**ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ**

CMS – Content Management System

СУБД – Система управління базами даних

MVC – Model – View – Controller

SQL – Structured Query Language

VS –Visual Studio

СКВ – Система Керування Вмістом

LINQ – Language Integrated Query

**ВСТУП**

З кожним роком електронна комерція в Україні набуває все більшого розмаху. Швидко збільшується кількість наших співвітчизників, котрі використовують інтернет для здійснення покупок, замовлення різноманітних послуг чи пошуку інформації про компанії, корті їх надають.

Ціни в інтернет магазинах зазвичай привабливіші ніж в звичайних за рахунок того що ресурусу електронної комерції не потрібно сплачувати за оренду торгової площадки, а асортимент товарів в них значно більший. Через те що прибуток від інтернет – магазину прямо залежить від довіри до нього потенційних покупців, власники приділяють величезні зусилля для підтримки і збереження репутації свого ресурсу і випадки обману чи шахрайства в серйозних магазинах трапляються дуже рідко. Завдяки поширеності і простоті у використанні сучасних пошукових систем, таких як «Google», «Bing», «Яндекс» знайти потрібний сайт можна дуже легко і це не вимагає майже ніяких зусиль. Знайти інформацію про певну компанію в інтернеті можна значно швидше ніж в довіднику, а повноцінний корпоративний сайт надає значно більший обєм інформації ніж може поміститися в оголошенні чи короткій статті в довідниках чи засобах масової інформації.

Дуже поширені випадки коли діючий бізнес відкриває свій сайт для збільшення продаж, залучення нових клієнтів та інформування вже існуючих. Найкраще для цих цілей підходить корпоративний сайт. Корпоративний сайт – це повноцінне представництво компанії в інтернеті, дуже часто саме по його виконанню та наповненню формується загальне враження про компанію. Крім того, багато компаній крім надання повної інформації про компанію, сферу її діяльності та послуги які вона надає розміщують на сайті калькулятори цін, форми для звязку, здійснення покупок чи замовлення послуг а іноді й вбудовані повноцінні магазини.

Проте зі збільшенням складності сайту, кількості сторінок та обсягу функціоналу і контенту в ньому різко зростає складність і вартість його обслуговування. Найефективнішим рішенням даної проблеми є застосування системи управління контентом. Зазвичай це порожній шаблон корпоративного сайту який після зміни дизайну і наповнення його контентом перетворюється в повноцінний унікальний корпоративний ресурс, але на відміну від звичайного сайту він має адміністративну частину, закриту для звичайних користувачів, де можна не володіючи версткою чи програмуванням швидко поміняти наповнення сайту та отримати повну інформацію про його роботу.

Такі системи завоювали дуже велику популярність через свою єфективність, доступність і простоту у використанні.

**1 АНАЛІЗ ТА ДОСЛІДЖЕННЯ ВПРОВАДЖЕННЯ СИСТЕМ УПРАВЛІННЯ КОТЕНТОМ В РЕСУРСАХ ЕЛЕКТРОННОЇ КОМЕРЦІЇ**

* 1. **Актуальність використання електронної торгівлі**

Завдяки суттєвому розвитку та поширенню сфери застосування компютерів та Інтернету на нові галузі економіки почав розвиватися принципово новий напрямок ведення бізнесу - електронний бізнес, який у західних країнах називають "економікою XXI сторіччя".

Найбільш прогресивною складовою електронного бізнесу є електронна торгівля (e-com), масштаби якої зростають неймовірними темпами. За даними різних моніторингових компаній обсяги електронної комерції зросли у світі з 120—150 млрд. дол. США в 1997 році до 1 трлн в 2002 році.

Неймовірні масштаби і швидкість розвитку електронної комерції у світі зумовлені сукупністю чинників економічного, соціального, електронно-технологічного, організаційно-правового характеру. Серед них можна відокремити три найсуттєвіші:

1. Лібералізація економічної діяльності і глобалізація економіки. Дія цих чинників виявляється у вільному переміщенні капіталу, товарів, послуг, технологій як усередині окремих країн, так і в міжнародному масштабі.
2. Мультифункціональність Інтернету. Постійне оновлення ресурсів, інструментаріїв, комп'ютерних технологій в Інтернеті зумовлює перспективність і надає універсальний характер цій мережі, суттєво розширює всі сфери бізнесу — у тому числі і комерційного. Переваги Інтернету полягають у тому, що він надійно забезпечує не лише окремі комерційні операції, а всі стадії комерційного процесу.
3. Доступність і ефективність електронної торгівлі. Електронна торгівля доступна для широкого кола суб'єктів ринку в організаційно-технічному та фінансово-економічному плані. Крім того, цей вид торгівлі відрізняється високою ефективністю і окупністю витрат.

Електронна торгівля як поняття має багато авторських та кілька офіційних визначень. Якщо узагальнити офіційні визначення (європейської комісії, організації економічної кооперації та розвитку, світової організації торгівлі СОТ), то сутність електронної торгівлі полягає в організації процесу товарно-грошового обміну у формі купівлі-продажу на базі електронних технологій. Більш широко трактується це поняття Комісією ООН з міжнародного торгового права (UNCITRAL): електронна торгівля — це організація і технологія купівлі-продажу товарів, послуг електронним способом з використанням телекомунікаційних мереж та електронних фінансово-економічних інструментів.

Отже, обидва визначення містять купівлю-продаж, що здійснюється електронними засобами, причому у віртуальному режимі.

Електронна торгівля, на відміну від електронної комерції, охоплює не весь процес товарно-грошового обміну, а тільки ту його частину, яка безпосередньо пов'язана з купівлею-продажем. Таке твердження належить до електронної роздрібної торгівлі. Що стосується електронної оптової торгівлі, то діапазон функцій тут значно ширше. Але в цілому електронна комерція поняття складніше, ніж електронна торгівля. Остання є частиною електронної комерції.

Електронна роздрібна торгівля є динамічною динамічною моделлю ведення бізнесу, і в процесі свого розвитку вона починає охоплювати дедалі більше нових операцій та процесів:

- установлення контакту (не фізичного) між "покупцем" і "продавцем"; обмін комерційною інформацією;

- повний цикл інформаційного супроводу покупця;

- пошук, демонстрація, вибір, консультація, конкурентоспроможність товару;

- продаж товарів, у тому числі електронних продуктів, надання послуг;

- взаєморозрахунки (у тому числі з використанням електронного переказу грошей, кредитних карток, електронних грошей);

- управління процесом доставки товарів безпосередньо покупцю або за вказаною ним адресою;

- післяпродажне обслуговування.

Залежно від формату електронної торгівлі наведені функції можуть бути розширені або звужені. Отже, за змістом своєї діяльності електронна торгівля суттєво відрізняється від реальної роздрібної торгівлі. їй притаманні характерні риси, особливості. Серед них можна виділити головні:

1. Віртуальність — брак особистого контакту між фізичними особами-суб'єктами процесу купівлі-продажу, тобто електронна роздрібна торгівля здійснюється в режимі on-line. Реально ж роздрібна торгівля здійснюється в режимі off-line, де безпосередній фізичний контакт обов'язковий.
2. Інтерактивність — адекватне інформаційне забезпечення покупця (споживача) його запиту у вигляді інтерфейсу, тобто німого діалогу.
3. Глобальність — брак часових, просторових, адміністративних, соціально-демографічних, асортиментно-товарних меж.
4. Динамічність — спроможність on-line торгівлі до моментальних змін і адаптації до нових умов.
5. Ефективність — спроможність забезпечити прибуток, інші економічні вигоди, а також соціальний ефект.

Отже, електронну торгівлю відрізняють особливості і характерні риси, яких бракує в реальних традиційних формах торгівлі.

**1.2 Види електроної комерції**

Існують такі види електронної комерції:

1. Бізнес-до-бізнесу (B2B)

B2B електронна комерція — це електронна комерція між компаніями. Це — тип електронної комерції, що має справу з відносинами між і серед видів комерційної діяльності. Приблизно 80% електронної комерції належать до цього типу, і більшість експертів пророкує, що B2B електронна комерція продовжить поширюватися швидше ніж B2C. Найільш загальні B2B приклади й найкращіращі практичні моделі — IBM, Hewlett Packard, Cisco and Dell.

1. Бізнес-до-споживача (B2C)

Електронна комерція бізнес-до-користувача, або торгівля між компаніями й споживачами, включає збирання інформації клієнтами; купівлю фізичних речей чи інформаційних/електронних товарів; і, для інформаційних товарів, одержування товару (програми, електронної книги) по електронній мережі. Це друга по величині й сама рання форма електронної комерції. Приклади B2C моделей — мережеві компанії, які займаються роздрібною торгівлею типу Amazon.com, Drugstore.com, Beyond.com. B2C електронна комерція зменшує ціну покупки (особливо ціну її пошуку), збільшуючи доступ споживачів до інформації й дозволяючи споживачам знайти найбільш конкурентоспроможну ціну за товар або послугу. B2C електронна комерція також зменшує ринкові бар'єри входу, тому що вартість створення й розкручування сайту набагато менша ніж створення фірми. У випадку інформаційних товарів, B2C електронна комерція ще привабливіша, тому що це зберігає фірми від факторингу в додатковій вартості фізичної мережі розподілу. Крім того, для країн зі зростаючою кількістю користувачів Internet, постачання інформаційних товарів стає усе більше й більше доступним.

1. Споживач-до-споживача (C2C)

Електронна комерція споживач-до-споживача або C2C — просто торгівля між приватними індивідуумами або споживачами. Цей тип електронної комерції характеризований ростом електронних ринків і мережевих аукціонів, особливо у вертикальних галузях промисловості. С2С можливо має найбільший потенціал для того, щоб розвивати нові ринки. Цей тип електронної комерції входить принаймні в три форми: 1) аукціони, типу eBay, що дозволяє інтерактивно пропонувати ціну в реальному часі; 2) однорангові системи, типу моделі Napster (протокол спільного використання файлів між користувачами в системах типу IRC) і пізніші моделі обміну грошей; 3) оголошення тематичних категорій у портальних сайтах типу тематичних категорій Excite і eWanted (діалоговий мережевий ринок, де покупці і продавці можуть вести переговори).

1. Мобільна торгівля (m-commerce)

Мобільна торгівля — закупівля й продаж товарів і послуг через бездротову технологію, тобто, кишенькові пристрої типу мобільних телефонів і особистих цифрових помічників.

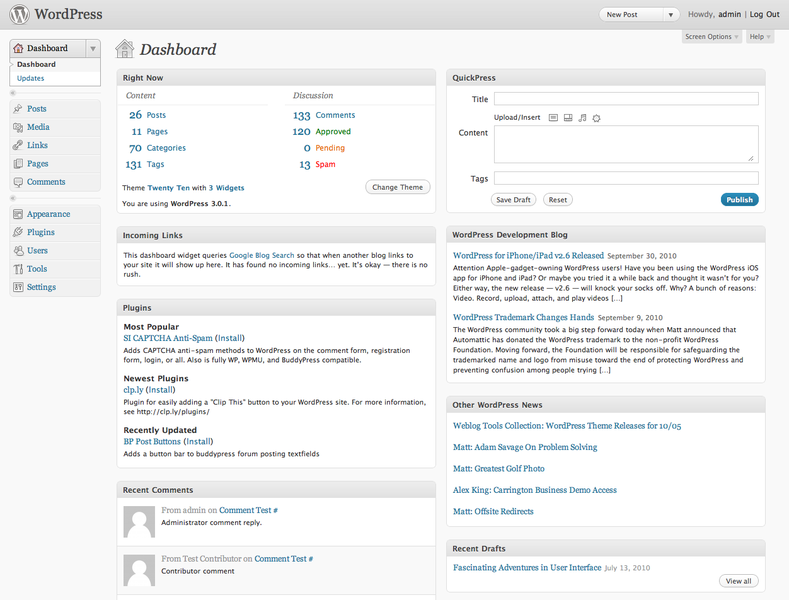
Для того щоб ознайомитися зі структурою і функціоналом CMS та створити якісну і функціональну CMS систему, котра буде відповідати всім поставленим до неї вимогам і буде придатною до використання в реальних умовах на ІТ - підприємствах і конкретних бізнес-проектах необхідно провести короткий аналіз існуючих CMS, в котрих можна запозичити вдалі ідеї, функціональні та архітектурні рішення. В якості CMS для ознайомлення я обрав такі популярні системи управління контентом, які використовуються для виконання широкого кола завдань, в тому числі і для створення сайтів електронної комерції (корпоративних сайтів):

* Wordpress
* Joomla
* Drupal
* 1C Бітрікс
* Netcat

**1.3.1 WordPress (WP e-Commerce)**

WordPress — це проста у встановленні та використанні [система керування вмістом](http://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B2%D0%BC%D1%96%D1%81%D1%82%D0%BE%D0%BC) з [відкритим кодом](http://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D0%BA%D1%80%D0%B8%D1%82%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F), яка широко використовується для створення [веб-сайтів](http://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B0%D0%B9%D1%82), зокрема, [блогів](http://uk.wikipedia.org/wiki/%D0%91%D0%BB%D0%BE%D0%B3).

Написана на мові програмування [PHP](http://uk.wikipedia.org/wiki/PHP) з використанням бази даних [MySQL](http://uk.wikipedia.org/wiki/MySQL). Ліцензія — [GNU General Public License](http://uk.wikipedia.org/wiki/GNU_General_Public_License). Сфера застосування — від блогів до складних [веб-сайтів](http://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B0%D0%B9%D1%82). Вбудована система тем і [плаґінів](http://uk.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D2%91%D1%96%D0%BD) в поєднанні з вдалою архітектурою дозволяє конструювати на основі WordPress практично будь-які веб-проекти.



Мал 1.1 Панель адміністратора WordPress

Загальні можливості:

* простота встановлення, простота налаштувань;
* підтримка веб-стандартів ([XHTML](http://uk.wikipedia.org/wiki/XHTML), [CSS](http://uk.wikipedia.org/wiki/CSS));
* модулі для підключення (плаґіни) з унікально простою системою їх взаємодії з кодом; можливість автоматичного встановлення та оновлення версії безпосередньо з панелі адміністратора;
* підтримка так званих «тем», з допомогою яких легко змінюється як зовнішній вигляд, так і способи виведення даних;
* можливість редагувати шаблони одразу в панелі адміністратора;
* «теми» реалізовані як набори файлів-шаблонів на PHP (у HTML-розмітку вставляються PHP-мітки);
* багато бібліотек «тем» і «плаґінів»;
* потенціал архітектури дозволяє легко реалізовувати складні рішення;
* СЕО-оптимізована система;
* наявність українського перекладу.

Можливості публікації та редагування:

* миттєва публікація;
* підтримка RSS, Atom, trackback, pingback;
* наявність ЛЗУ (людино-зрозумілий URL);
* редагування [WYSIWYG](http://uk.wikipedia.org/wiki/WYSIWYG)-редактором з можливістю вставлення форматованого тексту (наприклад з програми [Microsoft Word](http://uk.wikipedia.org/wiki/Microsoft_Word)) або редагування за допомогою [HTML](http://uk.wikipedia.org/wiki/HTML)-розмітки.

Можливості управління контентом:

* наперед заплановані публікації;
* багатосторінкові записи;
* прикріплення файлів та зображень до записів;
* можливість створення статичних сторінок;
* можливість створення свого типу контенту у власних темах;
* категорії, теги, коментування тощо.

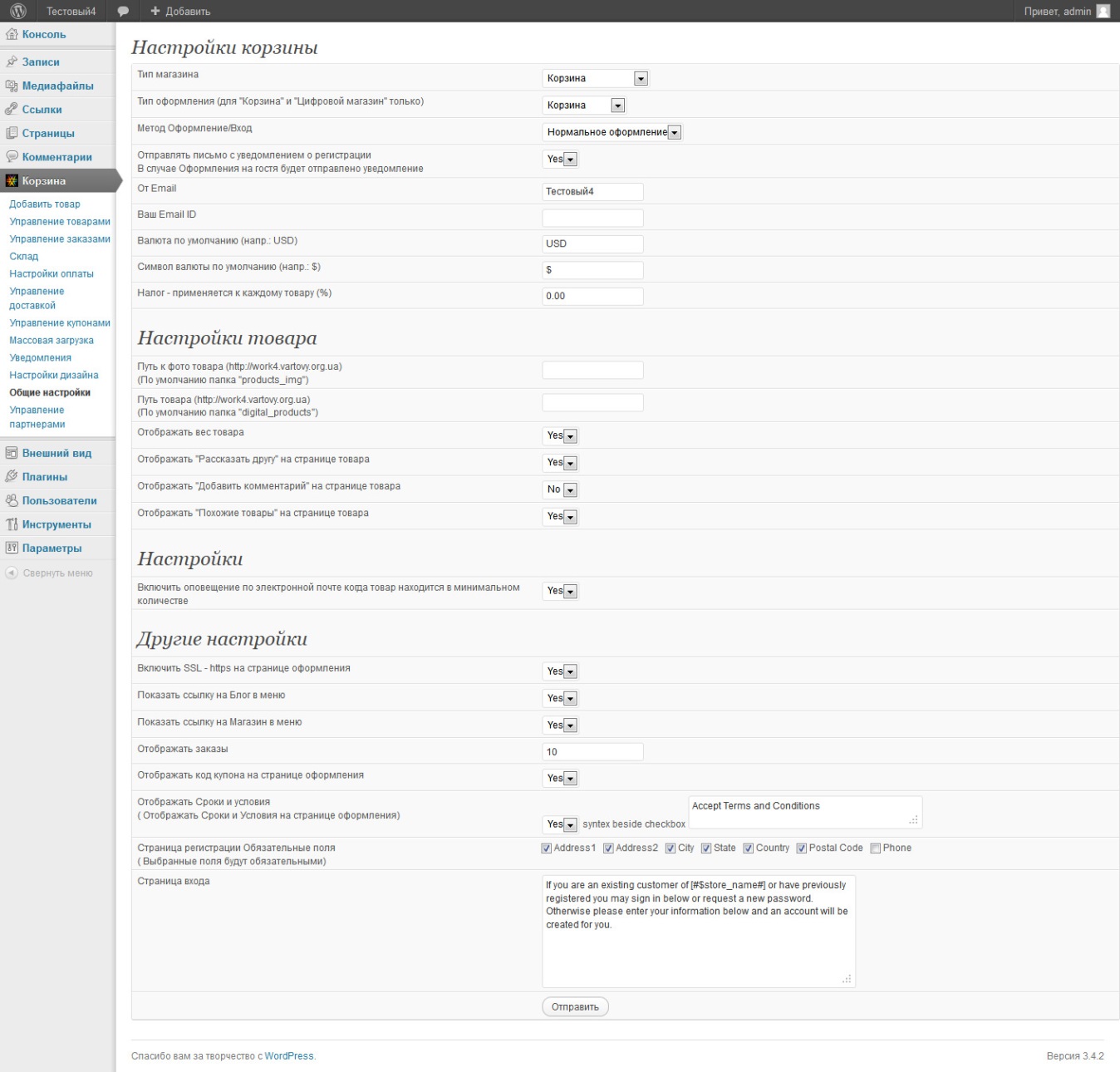
WP e-Commerce - плагін для створення електронного магазину на базі Wordpress, для якого характерні безліч налаштувань і опцій для створення каталогу товарів, деталізація роботи. Існує також платна версія плагіна з розширеним функціоналом.

Можливості WP e-Commerce:

* Оптимальні URL для пошукових систем
* Модуль тільки для зареєстрованих користувачів
* Необмежена кількість продуктів і категорій
* Велика частина функціоналу реалізована на AJAX
* Отримання інформації у вигляді CSV файлів
* Деталізована інформація про продукти і замовлення
* Робота з податками та зборами
* інтеграція з Google Checkout

Віджети:

* Кошик на AJAX
* Категорії та товари
* Пожертвування
* Спеціальна інформація про продукт



# Мал 1.2 Адіміністративна панель WordPress e-Commerce

**1.3.2 Joomla! E-commerce(VirtueMart)**

Joomla! - система управління вмістом (CMS), написана на мовах PHP і JavaScript, що використовує як сховища бази даних СУБД MySQL або інші індустріально-стандартні реляційні СУБД. Є вільним програмним забезпеченням, поширюваним під ліцензією GNU GPL.  
CMS Joomla! включає в себе різні інструменти для розробки веб-сайту. Важливою особливістю системи є мінімальний набір інструментів при початковій установці, який доповнюється в міру необхідності. Це знижує захаращення адміністративної панелі непотрібними елементами, а також знижує навантаження на сервер і економить місце на хостингу.  
Joomla! дозволяє відображати інтерфейс фронтальної та адміністративної частини на будь-якій мові. Каталог розширень містить безліч мовних пакетів, які встановлюються штатними засобами адміністрування. Доступні пакети російської, української, білоруської та ще деяких мов країн СНД.  
Функціональність можна збільшувати за допомогою додаткових розширень (компонентів, модулів і плагінів).  
Є модуль безпеки для багаторівневої аутентифікації користувачів та адміністраторів (використовується власний алгоритм аутентифікації і «ведення» сесій).  
Система шаблонів дозволяє легко змінювати зовнішній вигляд сайту або створити свій унікальний. У мережі існує величезний вибір готових шаблонів, як платних, так і безкоштовних.  
Передбачені різні схеми розташування модулів, що налаштовуються, включаючи ліве, праве, центральне і будь-яке інше довільне положення блоку. При бажанні вміст модуля можна включити у вміст матеріалу. До переваг системи можна віднести те, що всі компоненти, модулі, плагіни і шаблони можна написати самому, розмістити їх в структурованому каталозі розширень або відредагувати існуюче розширення на свій розсуд.  
Відбувається регулярний вихід оновлень. Існує публічний «баг-трекер» (система відслідковування помилок)

Починаючи з версії 1.6 вбудована багатомовність.  
Починаючи з версії 2.5 розширена підтримка баз даних. Реалізована підтримка Microsoft SQL Server, а з версії 3.0 - PostgreSQL . Надалі планується додати підтримку Oracle, SQLite.

Для кожної динамічної сторінки можна створити свій опис і ключові слова з метою підвищення рейтингу в пошукових системах;  
Початок і закінчення публікації будь-яких матеріалів можна запрограмувати за календарем;  
Можливість обмежити доступ до певних розділів сайту тільки для зареєстрованих користувачів, а з виходом Joomla 1.6 доступ як до розділу, так і до певного матеріалу з точністю до конкретного зв'язку матеріал ↔ користувач;  
Власні схеми розташування елементів по областях шаблону;  
Різні модулі (останні новини, лічильник відвідувань, докладна статистика відвідувань, гостьова книга, форум та інші);  
У версії 1.6 була сильно покращена система установки і управління розширеннями. Тепер можливо одночасно встановлювати кілька розширень, об'єднаних в один інсталяційний пакет. Більше того, реалізована можливість автоматичного оновлення встановлених розширень (за умови, що розробник розширення задіює цей механізм);  
У версії 1.6 з'явилася можливість публікації вмісту на декількох мовах;  
У версії 1.6 з'явилася можливість визначити час початку і завершення публікації модулів. Так само в новій версії Joomla поліпшені можливості по управлінню відображенням вмісту:  
Можливість створення не однієї, а декількох форм зворотного зв'язку для кожного контакту;  
Модуль прийому від віддалених авторів новин, статей і посилань;  
Ієрархія об'єктів;  
Менеджер розсилки новин. Підтримка більше ніж 360 служб розсилки новин по всьому світу;

Вбудований візуальний редактор TinyMCE.

Для даної CMS існує багато плагінів, які розширюють її можливості в сфері електронної комерції. Найпопулярнішими є такі плагіни:

* [VirtueMart](http://extensions.joomla.org/extensions/e-commerce/shopping-cart/129)
* [SimpleCaddy](http://extensions.joomla.org/extensions/e-commerce/shopping-cart/4148)
* [JoomShopping](http://extensions.joomla.org/extensions/e-commerce/shopping-cart/5378)
* [OSE Membership™](http://extensions.joomla.org/extensions/e-commerce/membership-a-subscriptions/7591)
* [HikaShop](http://extensions.joomla.org/extensions/e-commerce/shopping-cart/14275)

Найпопулярнішим компонентом для створення інтернет-магазинів на базі СКВ Joomla є [VirtueMart](http://extensions.joomla.org/extensions/e-commerce/shopping-cart/129). VirtueMart, раніше відомий як mambo-phpShop є вільним програмним забезпеченням для створення Інтернет-магазину, створеним для доповнення таких СКВ, як Mambo та Joomla. VirtueMart написаний на PHP і використовує базу даних MySQL. Найбільш підходить для веб-сайтів з низьким або середнім рівнем завантаження. VirtueMart захищений ліцензією GNU GPL.

VirtueMart з'явився вперше у вигляді автономної програми phpShop для створення інтернет-магазину. Оригінально названий mambo-phpShop, він став першим спеціально створеним компонентом електронної комерції для СКВ Mambo. Після того як від Mambo відкололася Joomla, розробник поміняв бренд mambo-phpShop на VirtueMart, який офіційно підтримує більш нову систему управління сайтами Joomla. Поточні версії VirtueMart можуть функціонувати з Mambo. Старі версії mambo-phpShop все ще доступні для завантаження, але вже не мають активної підтримки.

VirtueMart 1.0.0 вийшла 23 листопада 2005

VirtueMart 1.1.0 вийшла 23 квітня 2008 і закінчила свій розвиток версією 1.1.9, випущеної 20 червня 2011

VirtueMart 2.0.0 вийшла 19 грудня 2011.

VirtueMart 1.1.x був викачаний з офіційного сайту більше 500 тисяч разів. На ньому було зроблено кілька сотень тисяч інтернет-магазинів у всьому світі. На сьогоднішній день VirtueMart - найпопулярніший компонент для Joomla в галузі електронної комерції. Причиною цього є безкоштовність, розширюваність і величезний функціонал.

VirtueMart підтримує необмежену кількість продукції і категорій, необмежену кількість валют для одного продукту, призначення продукції безлічі категорій, можливість продавати продукцію, призначену для скачування, а також надає можливість відключити функцію продажу і використовувати VirtueMart в режимі каталогу. VirtueMart дозволяє мати різні ціни для однієї продукції, засновані на кількості або приналежності покупця до певної групи, і надає можливість використання різних платіжних систем. У VirtueMart реалізована система знижок, купонів, а також великий вибір систем оплати та способів доставки. Функціонал, що забезпечує можливість роботи декількох продавців, заявлений в VirtueMart 1.1.0, в даній гілці не до кінця був реалізований, тому його остаточна реалізація перенесена у версію 2.0.0.

VirtueMart є вільним програмним забезпеченням для реалізації функціоналу інтернет-магазину і весь PHP-код повністю доступний. Це дозволяє PHP-розробникам переглядати, оновлювати або змінювати можливості і функції інтернет-магазину. На додаток до всього, VirtueMart використовує спрощені шаблони (звані 'fly pages' в VirtueMart), що мають можливість змінювати зовнішній вигляд інтернет-магазину і мають HTML і CSS формат. При перенесенні VirtueMart з Mambo на Joomla розробники не приділили належної уваги відмінностям між Mambo та Joomla, в результаті VirtueMart гілки 1.0.x містить в собі багато старого процедурного коду та компоненти, які вже існують в Joomla, але з якихось причин не використовуються (шаблонізатор , з'єднання з базою і т. д.). У сучасну гілку VirtueMart 1.1.x була включена можливість швидко змінювати теми для категорії товарів, сторінки замовлення і сторінки кошика.

Можливості панелі адміністратора:

Маркетингові інструменти

* Багаторівневе ціноутворення
* Купони
* Правила ціноутворення

Визначення податкових ставок

* Визначення правил для регіону або країни.
* Створення класів податку, наприклад «Нормальний» або «Оптовий»
* Виставлення ставки ПДВ.

Продукти та каталог

* Можливість завдання різних атрибутів (властивостей) для товару.
* Необмежена вкладеність категорій.
* Конфігуровані продукти - продукти з вибірковими властивостями (наприклад: колір, розмір і т. д.)
* Сортування товару за наперед визначеними атрибутами (виконується за допомогою додаткового модуля).

Адміністрування

* Групи покупців (прості, VIP і т. д.)
* Імпорт / Експорт (у вигляді сторонніх розробок)
* Система контролю доступу
* Вибір валюти
* Вибір країн
* Звіти продажів
* Вибір способів доставки
* Вибір способів оплати

Можливості фронт-енду:

* Пропозиція покупцям супутніх товарів
* Пропозиція покупцям недавно переглянутих товарів
* Пошукова оптимізація (SEO) - поки у вигляді сторонніх розробок
* Зв'язок з покупцем
* Повідомлення по електронній пошті
* RSS-підписка на категорію або всі групи товарів
* Пошук і розширений пошук
* Порівняння продуктів - у вигляді сторонніх розробок
* Теги для продуктів - у вигляді сторонніх розробок
* Відгуки покупців про товари
* Оформлення замовлення
* Кошик покупок
* Акаунти покупців
* Рекомендація товару одним покупцем іншому
* Топ-10 товарів
* Показ спецпропозицій на товари

Імпорт і експорт даних:

У стандартну складання не входять модулі імпорту / експорту, тільки у вигляді сторонніх розробок.

* Імпорт та експорт товарів
* Імпорт і експорт товарних залишків на складах
* Імпорт і експорт покупців

Можливість створення шаблонів імпорту / експорту для різних цілей. Існують сторонні розробки модулів експорту в Яндекс. Маркет.

Способи оплати:

Ті, що входять у стандартну збірку:

* Кредитною картою
* Оплата при доставці
* PayPal
* eCheck.net
* MerchantWarrior

Сторонні:

* WebMoney
* Яндекс. Гроші
* RBKMoney
* Пошта Росії.

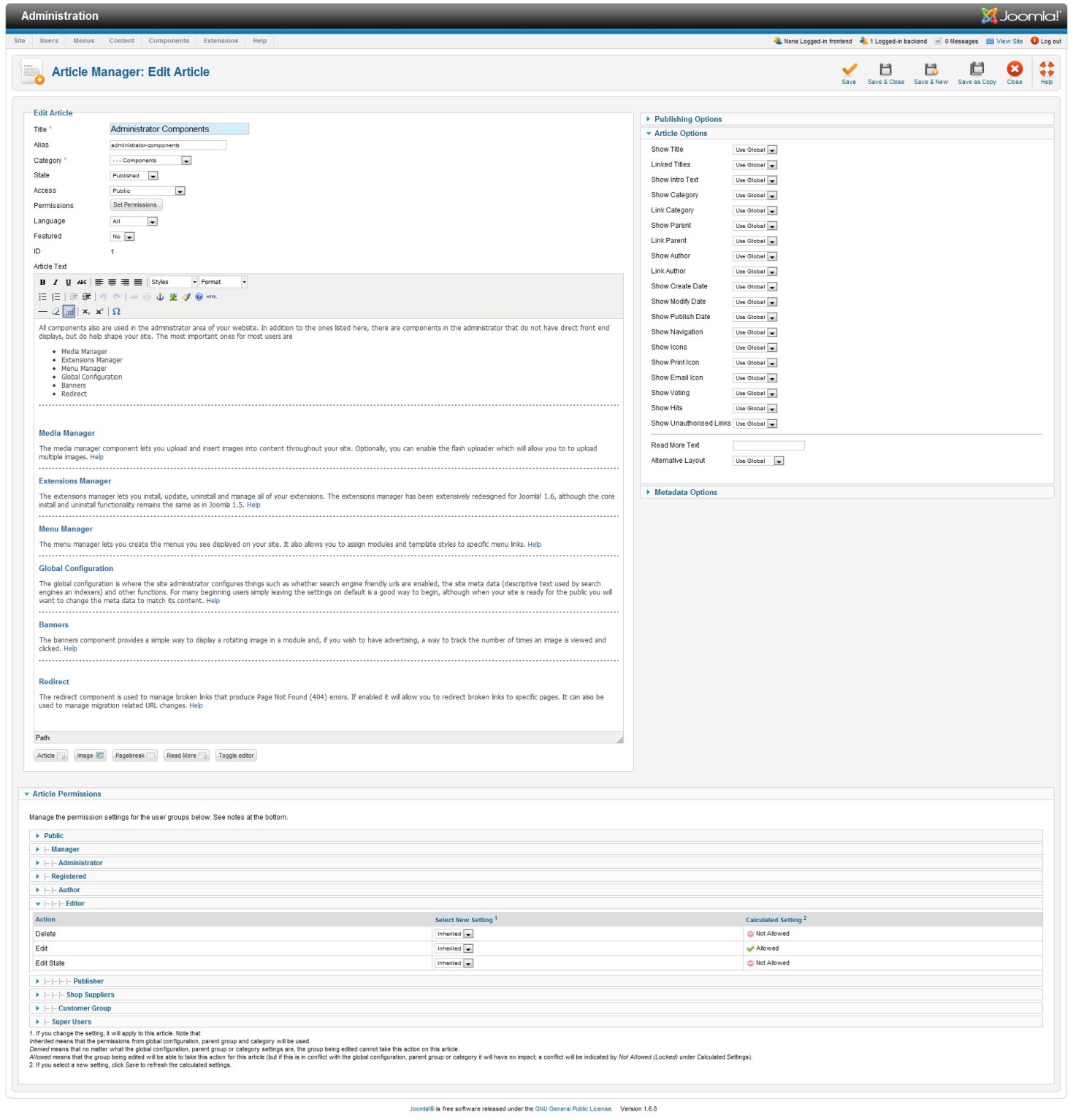
Способи доставки (ті, що входять у стандартну збірку):

* auspost
* Canada Post
* DHL
* FedEXdc
* Flex
* InterShipper
* shipvalue
* UPS
* USPS
* Стандартний модуль доставки
* Доставка в залежності від регіону або країни
* Самовивезення

Так само існують багато сторонніх розробок модулів доставки.

Виробники:

* Можливість угруповання виробників за категоріями.
* Можливість виведення опису виробника і його координат.



Мал 1.3 Адміністративна панель CMS Joomla!

**1.3.3 Drupal** **E-Commerce (Ubercart)**

Drupal- система управління контентом, написана на мові php і зберігає дані в реяційній базі даних. Вона може працювати у таких популярних системах як Windows, Mac OS X, Linux, на будь-якій платформі, яка підтримує роботу web-сервера Apache, Nginx, Lighttpd або Microsoft IIS.

Інтерфейс системи можна локалізувати на більш ніж на 100 мов. В Drupal використовується власний механізм для локалізації зі зберіганням перкладів у базі даних, нарівні з іншим вмістом сайту. При цьому не дуже важкою є адаптація до роботи зі стандартною бібліотекою gettext. Імпорт та експорт перекладів сайту здійснюється у вигляді ро-файлів (формат, що використовується бібліотекою gettext).

Кожен документ сайту може належати одній або кільком рубрикам. Самі ж рубрики можуть складати списки або складні ієрархічні структури довільної вкладеності (з множинними предками і перехресними посиланнями елементів).

Наскрізна рубрикація є можливою за всіма типами документів сайту (наприклад список ключових слів, загальний для форумів та блогів). Форум із фисновком цікавих новин на головну сторінку або сайт новин із блогами та відеопрезентації-все це можна укласти в єдиний рубрикатор (або декілька рубрикаторів) і це буде виглядати частинами єлиного сайту, а не розрізненими сторінками об`єднаними лише загальним дизайном.

Сайт новин, сайт-візитка компанії, блог або форум - такі сайти можна будувати, користуючись тільки модулями CMS, що йдуть у поставці, потрібно тільки включити відповідні модулі, налаштувати їх і перенести сайт на хостинг.

Для зручності доступу до архівних матеріалів служать рубрикація контенту і пошук з урахуванням видів контенту, рубрик та вмісту. Документи зберігають незмінні посилання весь час свого життя. Також за допомогою коротких посилань і псевдонімів сайт набуває імен розділів і окремих сторінок, що добре запам`ятовуються, які не використовують спеціальних символів і тому добре індексовані пошуковими системами.

До базового пакету системи, окрім модулів створення статичних сторінок та нових статей входять модулі для організаціх блогів, форумів, сигдикації, модуль керування інформаційними блоками на сторінках, що полегшують керування їх виглядом, модуль керування меню. Drupal підтримує різні теми оформлення та дозволяє створювати свої теми оформлення. Спільнотою розробників даної CMS створено багато додаткових модулів, серед яких варто згадати модулі інтернаціоналізації (створення багатомовних сайтів), модулі керування файлами, що дозволяють викладати на сайтах звукові та відео-файли, модулі категоризації вмісту, модулі організації користувачів у групи та спільноти.

У Drupal пропонується гнучка схема організації структури сайту на основі таксономії. Таксономія-механізм, що дозволяє створювати довільну кількість тематичних категорій для вмісту сайту і асоціювати їх з модулями, що забезпечують введення і виведення інформації. Категорії можуть представляти плоскі або ієрархічні списки, чи складні структури, де елемент може мати декілька «батьків» і кілька дочірніх елементів. За допомогою подібної схеми одними і тими ж модулями можлива організація різних варіантів структуризації вмісту. Наприклад, легко створюється наскрізний список «ключових слів» для всіх документів сайту тощо.

Інша парадигма з`явилася зі створенням в Drupal модуля Content Construction Kit (CCK). (З виходом версії Drupal 7-перенесено в ядро). ССК дозволяв доповнювати документи новими полями різних типів-від полів вводу URL і email, до полів зберігання і відображення мультимедійних файлів. Також за допомогою додаткових модулів до ССК (наприклад Node reference) можна організувати зв`язок між документами, не використовуючи механізм таксономії. В Drupal 7 майже весь функціонал ССК перенесений в ядро системи. В модулі ССК залишилися хелпери (наприклад підтримка php коду).

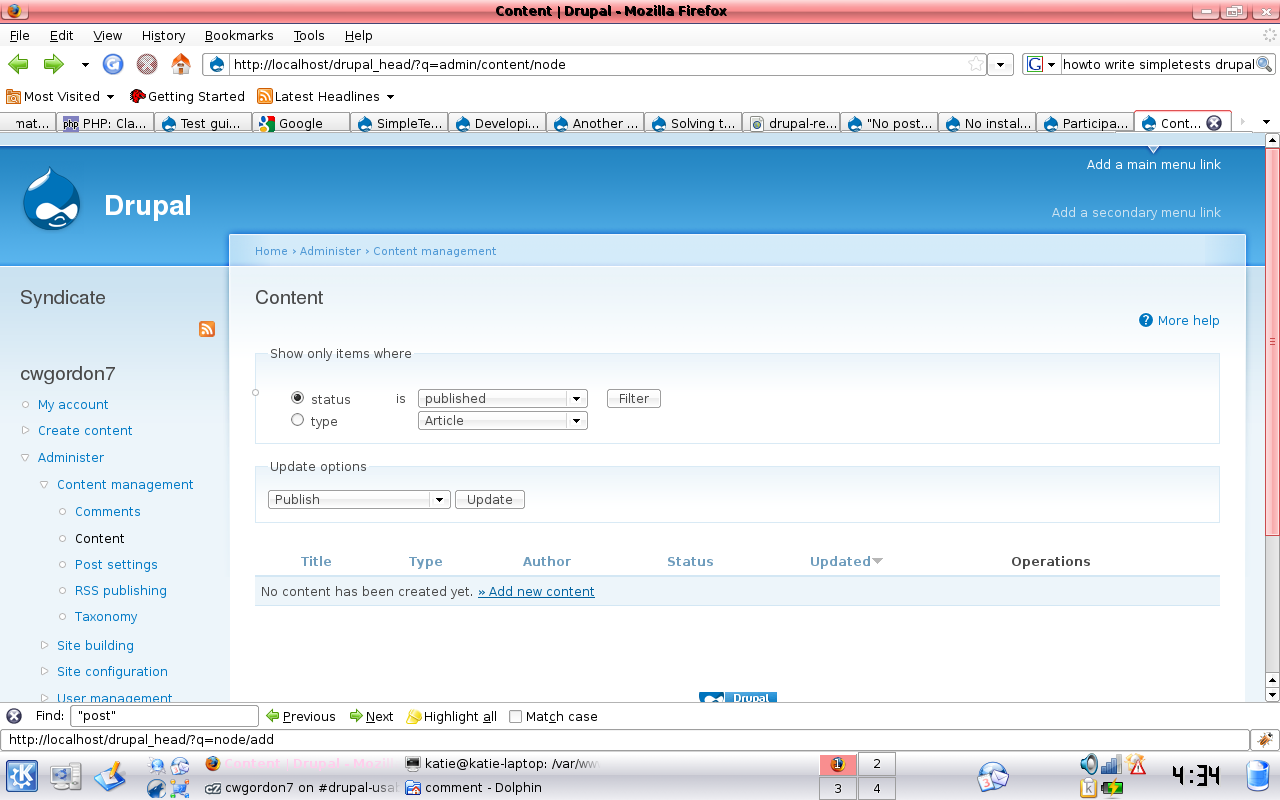
Drupal має модульну архітектуру з компактним ядром, що надає API, до якого можуть звертатися модулі. Стандартний набір модулів включає такі функції, як новинна стрічка, блог, форум, завантаження файлів, збирач новин, голосування, пошук тощо. Дизайн сайту змінюється за допомогою спеціальних модулів-«тем оформлення».

Для розширення можливостей СКВ Drupal в електронній комерції використовують додаткові модулі. одним з найпопулярніших доповнень такого типу є Ubercart. Ubercart є вільним програмним забезпеченням для відкриття торгових майданчиків, інтернет-магазинів, аукціонів, створюваних для доповнення такої системи керування вмістом, як CMF Drupal. Модуль Ubercart може бути встановлений на будь-якому сервері з підтримкою PHP і MySQL, поширюється по ліцензії GNU GPL.

Ubercart повністю інтегрована з Drupal'ом, що означає 100% сумісність з ним будь-якого веб-сайту, який використовує Drupal. Модуль може використовуватися як у випадках продажу товарів і послуг, так і при скачуванні онлайнової продукції. Крім того, його можна застосовувати для найму і працевлаштування, створення подій. Головним же призначенням Ubercart є можливість відкривати необмежену кількість платіжних шлюзів різних електронних світових валют і банківських систем. Ubercart дозволяє проводити цілий ряд комплексних дій всередині CMF Drupal, а саме: додавати новий продукт, здійснювати мультипродажу товарів, робити знижки і видавати купони, обмінювати валюту, здійснювати будь-які внутрішні і міжнародні платежі, підтримувати партнерські програми, виплачувати комісії, видавати купони на ліцензовані товари та послуги, проводити комплексну класифікацію продукції підприємства та ін.

Ubercart може працювати на різних мовних платформах, так само як і Drupal.

Стабільною версією для Drupal'а 5 є Ubercart 1.9. А для Drupal'а 6 - Ubercart 2.10.



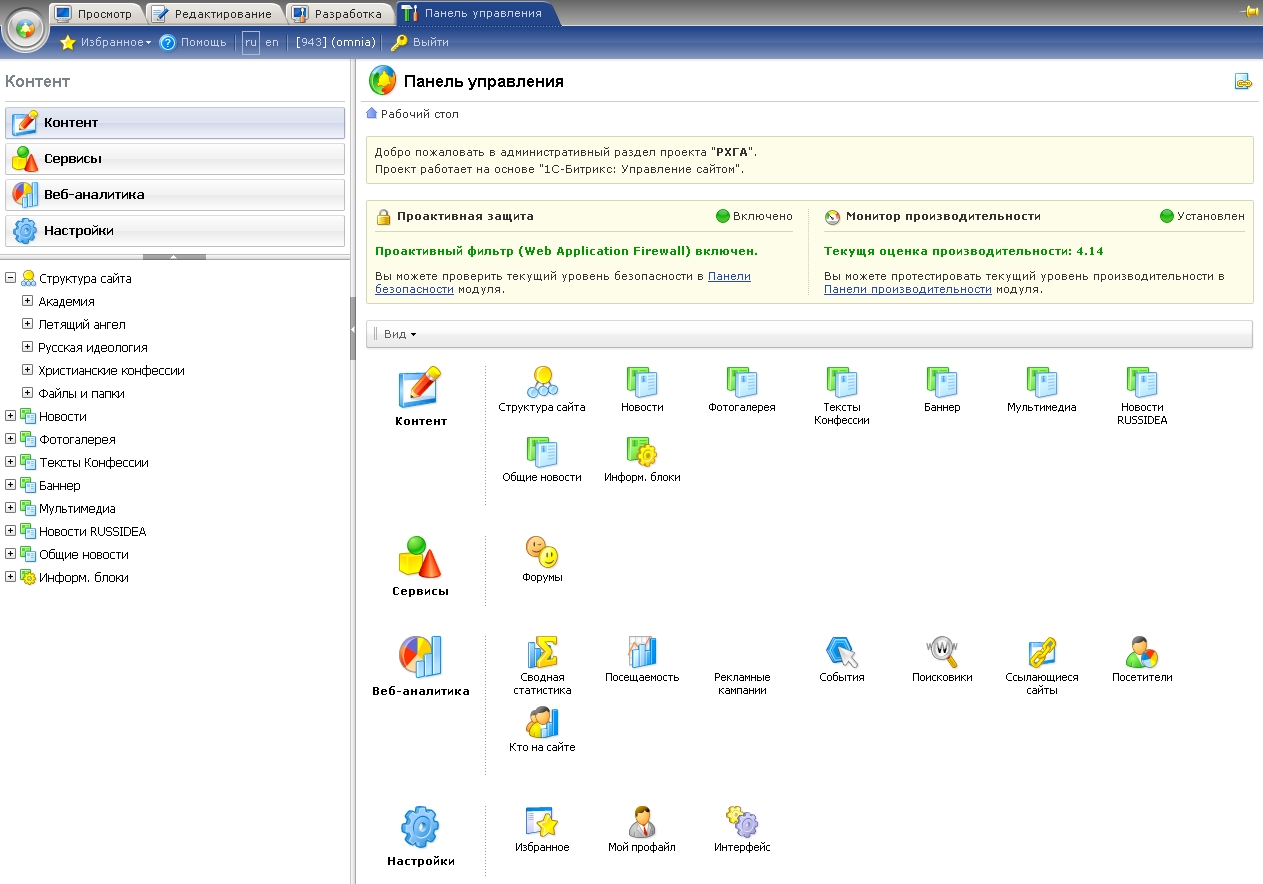
Мал 1.4 Адміністративна панель CMS Drupal

**1.3.4 1С-Бітрікс**

1С-Бітрікс: Управління сайтом - професійна система управління веб-проектами, універсальний програмний продукт для створення, підтримки та успішного розвитку:

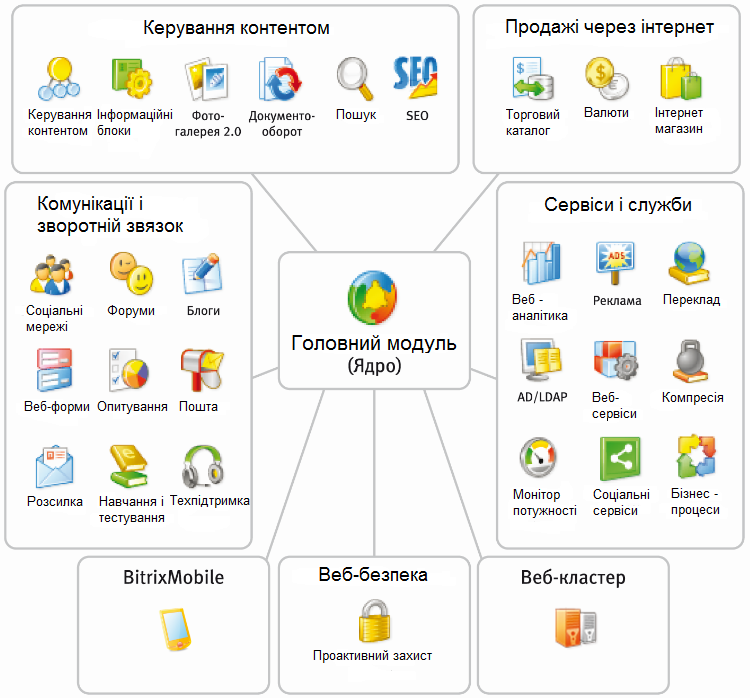
* корпоративних сайтів;
* інтернет-магазинів;
* інформаційних порталів
* сайтів спільнот
* соціальних мереж та інших веб-проектів

Основні особливості:



Мал 1.5 Панель керування CMS 1С-Бітрікс

* Просте управління сайтом  
  Адаптивний інтерефейс «Ермітаж». Управляти сайтом так само просто, як і працювати зі звичайним текстовим редактором - технічну частину роботи за вас виконає система. Адаптивний інтерфейс «Ермітаж» спрощує освоєння системи управління сайтом, знижує частку помилкових дій користувачів, економить час при внесенні змін на сайт. Інтерфейс запам'ятовує уподобання клієнта і дозволяє витрачати менше часу на управління контентом сайту.  
  івапівпп
* Готовий функціонал  
  До складу програмного продукту входять більше 30 модулів для управління інформаційним наповненням і структурою, продажами через Інтернет, соціальною мережею, медіафайлами і фотогалереями, форумами, блогами, рекламою і багатьма іншими можливостями сайту.
* Веб-безпека  
  Продукт забезпечує високий рівень захищеності сайтів від злому, що підтверджує незалежний аудит компанії Positive Technologies. Якість реалізації комплексу «Проактивний захист»: проактивний фільтр, вбудований веб-антивірус, одноразові паролі (OTP) та інші технології захисту надають впевненість не тільки в надійності ядра системи, але і в безпеці інтернет-рішення на її основі, з урахуванням надбудов і доробок .
* Висока продуктивність  
  Продуктивність сайту клієнта завжди під контролем. Система автоматично виробляє діагностику роботи і видає рекомендації щодо поліпшення продуктивності інтернет-проекту. Продукт дозволяє досягти чудових результатів за швидкодією навіть в умовах обмеженості ресурсів, а також будувати високопродуктивні системи для веб-сайтів з дуже великою відвідуваністю і високими піковими навантаженнями.
* Підтримка «хмарних» сховищ  
  Продукт сумісний з «хмарами» Google Storage, Amazon S3, Windows Azure Storage від Microsoft, відкритого проекту OpenStack та іншими, а також з CDN (мережі доставки і дистрибуції контенту -. Англ мережі доставки контенту або Content Distribution Network), інтегрованими з цими хмарними сховищами .  
  «1С-Бітрікс» надає своїм клієнтам безкоштовну можливість «хмарного» бекапа. Клієнти можуть зберігати резервну копію свого сайту в «хмарі» Amazon S3 і робити це штатними засобами системи.
* Інтеграція з «1С»  
  Схема взаємодії програмних продуктів Продукт повністю сумісний з «1С: Підприємство 8.2», завдяки чому можна створити інтернет-магазин, інтегрований в інформаційне середовище компанії: автоматично публікувати на сайті каталоги товарів з «1С», прайс-листи, вивантажувати замовлення, їх статуси, а також дані по залишках на складі з сайту в «1С» і назад.
* Мобільний сайт  
  Мобільний інтернет-магазин BitrixMobile - технологія розробки веб-додатків для мобільних і планшетних пристроїв-, на основі якої можна створювати мобільні сайти і інші веб-додатки. У постачання продукту включений готовий мобільний інтернет-магазин, який працює на iPhone, IPAD, Android, Blackberry. Користувачі можуть на мобільних пристроях вибирати товари та оформляти замовлення.



Мал 1.6 Структура CMS 1С Бітрікс

**1.3.5 Netcat E-commerce**

Система управління сайтами Netcat дозволяє створювати [веб-сайти](http://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B0%D0%B9%D1%82) різних рівнів складності і управляти створеним сайтом без необхідності знання [веб-сервера](http://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80) та [веб-технологій](http://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D1%96%D1%97). Простий та інтуїтивно зрозумілий [інтерфейс](http://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81) системи адміністрування дозволить управляти сайтом користувачеві, знайомому з комп'ютером на рівні роботи в [Microsoft Windows](http://uk.wikipedia.org/wiki/Microsoft_Windows)/[Microsoft Office](http://uk.wikipedia.org/wiki/Microsoft_Office).

Перша версія системи була розроблена в 1999 році. Згідно з дослідженням ринку CMS, проведеного інтернет-виданням Webinform, система Netcat є універсальним засобом управління сайтами. Система розрахована для наступних видів сайтів:

* корпоративні представництва;
* інтернет сервери портального типу;
* бібліотеки даних, файл-архіви;
* інтернет-видання, [ЗМІ](http://uk.wikipedia.org/wiki/%D0%97%D0%9C%D0%86);
* [електронні магазини](http://uk.wikipedia.org/wiki/%D0%95%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B8%D0%B9_%D0%BC%D0%B0%D0%B3%D0%B0%D0%B7%D0%B8%D0%BD)
* і інше, в тому числі складні інтерактивні веб-системи.

Система адміністрування в Netcat розділена на дві частини: [інтерфейс](http://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81) користувача і [інтерфейс](http://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81) розробника. Для використання системи не потрібне знання [інтернет-технологій](http://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82-%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D1%96%D1%97), [мов програмування](http://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B8_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) і [розмітки](http://uk.wikipedia.org/wiki/HTML). Інтерфейс системи простий і інтуїтивно зрозумілий для користувача, що має досвід роботи на комп'ютері.

Типові можливості системи (створення [рубрикатора](http://uk.wikipedia.org/w/index.php?title=%D0%A0%D1%83%D0%B1%D1%80%D0%B8%D0%BA%D0%B0%D1%82%D0%BE%D1%80&action=edit&redlink=1), адаптація [дизайну](http://uk.wikipedia.org/wiki/%D0%94%D0%B8%D0%B7%D0%B0%D0%B9%D0%BD), наповнення вмістом, адміністрування), необхідні для більшості [сайтів](http://uk.wikipedia.org/wiki/%D0%A1%D0%B0%D0%B9%D1%82), можуть легко доповнюватися нестандартними рішеннями для [електронної комерції](http://uk.wikipedia.org/wiki/%D0%95%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B0_%D0%BA%D0%BE%D0%BC%D0%B5%D1%80%D1%86%D1%96%D1%8F), каталогами різного типа, системами [статистики](http://uk.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D1%82%D0%B8%D1%81%D1%82%D0%B8%D0%BA%D0%B0), системами управління рекламою.

Netcat дозволяє гнучку адаптацію системи під потреби замовника.

Система управління сайтами Netcat надає користувачам і розробникам такі можливості:

* Створення і впровадження за допомогою Netcat інформаційних систем різного рівня — від простих сайтів з мінімальним контентом до складних веб-систем, порталів, торгівельних майданчиків і ЗМІ.
* На базі однієї встановленої системи можливо управляти декількома різними сайтами.
* Система може бути встановлена практично на будь-який сучасний хостинг без установки додаткових програм.
* Управління інформацією на сайті здійснюється користувачем або групою користувачів з можливістю чіткого розділення прав управління.
* Сайт на основі системи Netcat дозволяє використовувати будь-які графічні і анімаційні елементи.
* У системі Netcat реалізовані різні функціональні можливості, наприклад, можливість інтерактивного спілкування з відвідувачами сайту, голосування, опити, форуми різних типів.

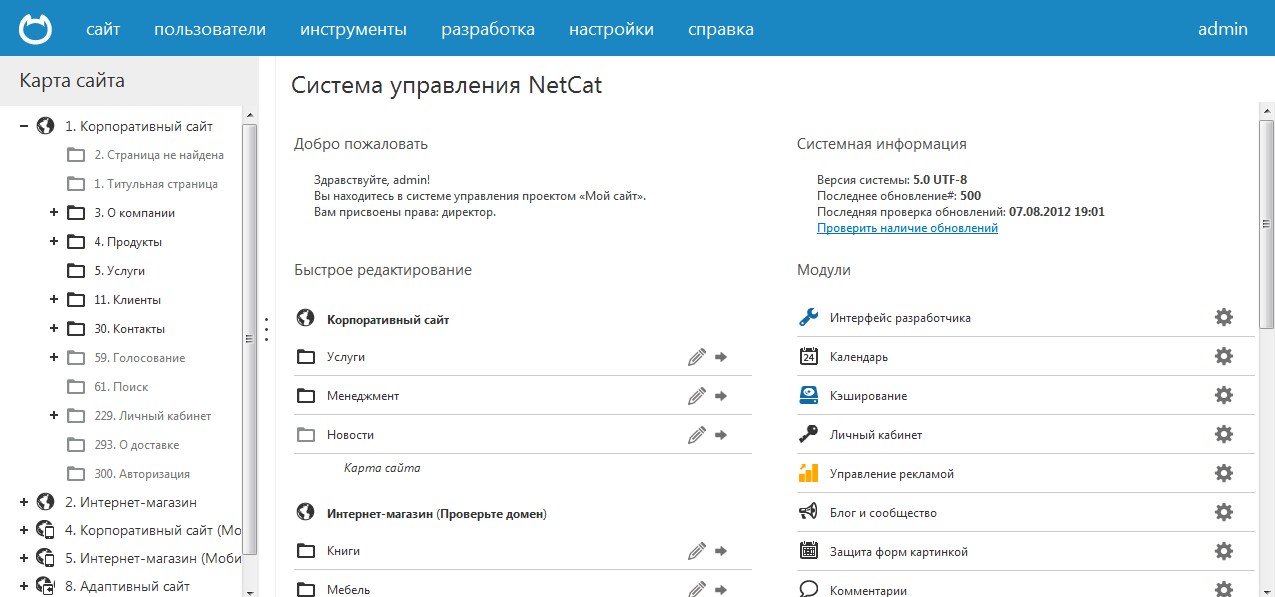
Розробники мають можливість створювати додаткові модулі самостійно.

Для Netcat розроблено декілька додаткових модулів:

* модулі інтерактивних функціоналів
* модулі електронної комерції
* модулі електронного документообігу
* модулі взаємодії з партнерами або співробітниками, що перебувають поза офісом, модулі взаємодії між різними офісами в рамках однієї системи.

Також за допомогою Netcat можуть бути вирішені завдання створення орієнтованих для веб-сервера інформаційних систем:

* розрахованих на багато користувачів, з розділенням прав доступу до інформації і функцій по її зміні
* метою яких є структуризація і каталогізація інформації різних типів
* [ASP](http://uk.wikipedia.org/wiki/ASP)-сервісів і ін.



Мал 1.7 Система управління CMS Netcat

NetCat E-Commerce – це редакція, орієнтована на системи електронної комерції: прості інтернет-магазини або складні торговельні системи, b2b-і b2c-майданчики або закриті торгові системи.

E-commerce відмінно підходить і для корпоративних сайтів з елементами електронної комерції, завдяки тому, що редакція повністю інтегрована з 1С: Бухгалтерією, проектами Яндекс.Маркет і Товари@Mail.Ru, включає можливість виставлення рахунків і прийому платежів кредитними картами або електронними грошима.

Дана редакція складається з таких модулів і компонентів:

Модулі:

1. Мінімагазини. Даний модуль дозволяє з мінімальними зусиллями організувати нескладний магазин на сайті.
2. Пошук по сайту. Повнотекстовий багатофункціональний пошук по сайту.
3. Особистий кабінет. Модуль відповідає за роботу з користувачами.
4. Файл-менеджер. Зручний сервіс для будь-яких операцій з файлами.
5. Коментарі. Можливість висловлювати свою думку щодо будь-якій публікації.
6. Кешування. Інструмент, що підвищує продуктивність в часі і скорочує навантаження на сервер.
7. Календар. Календар публікацій.
8. Статистика відвідувань. Збір та аналіз статистичних даних про відвідування сайту.
9. Інтерфейс розробника. Засіб розширення API проекту власними функціями.
10. Захист форм картинкою (CAPTCHA). Додаткова перешкода для автоматизованих засобів розміщення інформації на сайтах.
11. Управління підписками і розсилками. Організація підписки на оновлення сайту.
12. Блог і співтовариство. Організація і ведення колективних, персональних і корпоративних блогів.
13. Логування. Інформація про всі зміни в системі, які відбуваються користувачами або скриптами.
14. Інтернет-магазин. Організація як простого магазину, так і закритих торгових систем.

Компоненти:

1.Базові

* Проста сторінка
* Список підрозділів
* Посилання на сайти

2.Банери

* Банери прості

3.Корпоративні

* Адреси компанії
* Клієнти
* Корпоративні контакти
* Відгуки про роботу
* Проекти

4.Вітрина товарів

* Замовлення
* Каталог товарів

5.Голосування

* Рейтинг об'єктів

6.Інтерактив

* Гостьова книга / FAQ
* Коментарі

7.Веб-форми

* Заявка на участь у заході
* Лист з сайту

8.Люди, робота

* Вакансії компанії
* Особисті контакти
* Онлайн-консультанти
* Персоналії
* Резюме

9.Мультимедіа та файли

* Проста фотогалерея
* Список файлів простий
* Файл-менеджер
* Фотоальбоми
* Фотогалерея

10.Пошук по сайту

* Документи
* Пошук по сайту

11.Користувачі

* Друзі
* Приватні повідомлення
* Особистий рахунок

12.Простий блог

* Записи в блозі

13.Публікації

* Новини (ЗМІ)
* Новини компанії
* Прості статті

14.Товари та послуги

* Адаптивний каталог товарів
* Прайс лист
* Простий каталог товарів

15.Блог

* Записи блогу
* Коментарі блогу
* Налаштування блогу
* Користувачі блогу
* Список блогів

16.Інтернет магазин

* Валюти
* Варіанти доставки
* Варіанти оплати
* Замовлення
* Інтернет-магазин
* Компонент корзини
* Курси валют ЦБ
* Налаштування розділу магазину
* Знижка
* Відповідність цін і груп користувачів
* Товар
* Шаблони листів

17.Передплати та розсилки

* Список підписок

18.Мінімагазини

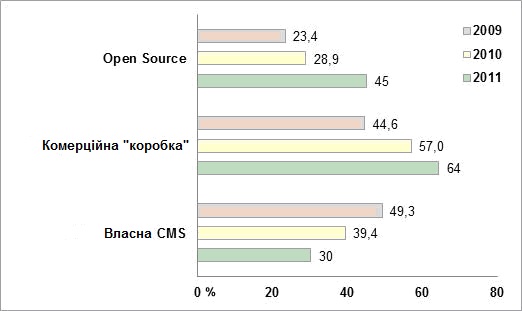
* Замовлення
* Кошик

19.Коментарі

* Останні коментарі
  1. **Статистика використання програмних рішень в електронній торгівлі**

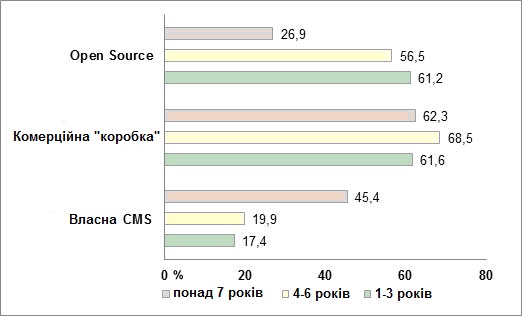
Статистика використання різних CMS дозволяє визначити лідерів ринку, яких потрібно використовувати як зразок при написанні власної CMS. Також вона допомагає проаналізувати тенденції розвитку ринку створення і підтримки сайтів електронної комерції та визначити найперспективніші напрямки побудови та розвитку створюваної системи. Поскільки створювана система швидше за все буде експлуатуватися саме в Україні, то найактуальнішою для неї буде саме статистика по секторі інтернету пострадянського простору ( так званого «Рунету»).

Веб-студіями використовуються три основних типи систем управління ресурсами електронної комерції – комерційні «коробкові» рішення, безкоштовні «коробкові» рішення (Open Source) та системи, розроблені силами студії (власні CMS). В кінці 2011-го року співвідношення цих систем виглядало так:



Мал. 1.8. Розповсюдженість типів CMS.

Загальні тенденції прекрасно помітні - частка «коробок», як комерційних, так і безкоштовних, з кожним роком зростає, а використання власних CMS зменшується. При цьому в різних регіонах картина розрізняється.



Мал. 1.8. Залежність між віком веб – студії і використовуваною нею CMS.

За минулий рік Open Source-системи для підтримки електронної комерції дуже серйозно заявили про себе. У той час як лідери комерційного сегменту - 1С-Бітрікс і NetCat - демонструють стагнацію і деяке погіршення своїх показників, Joomla!, Wordpress і Drupal активно набирають прихильників серед веб-розробників.

**1.5 Переваги і недоліки використання електронної торгівлі**

Електронна торгівля має значні переваги навіть перед прогресивними формами реальної (off-line) торгівлі, але вона має також і деякі вади. Переваги і вади можуть бути поділені між основними суб'єктами процесу купівлі-продажу таким способом (табл. 1.1, табл 1.2).

Таблиця 1.1 Основні переваги і недоліки електронної торгівлі для компанії

|  |  |
| --- | --- |
| **Переваги** | **Вади** |
| 1. Розширення торгівлі, вільний рух товару, послуги без обмежень і бар'єрів | 1. Складність реалізації технічних, технологічних проектів, відсутність кадрів |
| 1. Безперервність роботи торгівельного об'єкта. Максимальна інтенсивність торговельного обслуговування | 1. Складність реалізації технічних, технологічних проектів, відсутність кадрів |
| 1. Зменшення витрат обігу, а також утримання інфраструктури | 1. Брак правових механізмів |
| 1. Вбудовані засоби реклами, маркетингу, інформації. | 1. Можливий несанкціонований доступ до товару та грошових засобів, конфіденційної інформації |
| 1. Прискорення розрахунків |  |
| 1. Відсутність обмежень комерційного бізнесу |  |

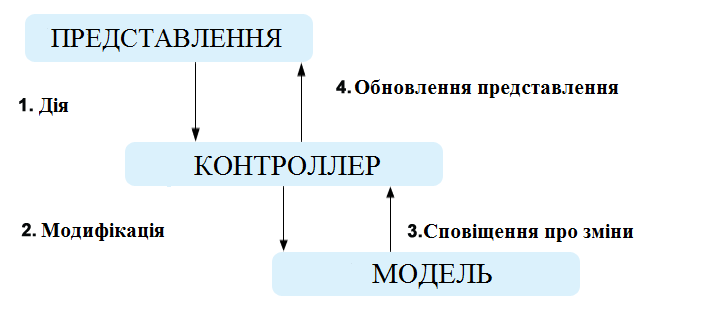
Таблиця 1.1 Основні переваги і недоліки електронної торгівлі для покупця

|  |  |
| --- | --- |
| **Переваги** | **Вади** |
| 1. Доступність до товару: територіальна, демографічна, часова | 1. Невпевненість, скутість при здійсненні дорогих покупок |
| 1. . Економія часу і грошей | 1. Перебої з доставкою товару |
| 1. Зручність і високий рівень торговельного обслуговування | 1. Складність процесу повернення та обміну товару |
| 1. Доставка товару за конкретною адресою |  |
| 1. Морально-психологічна перевага при відвіданні сайта, ознайомлення з товаром, його характеристиками |  |

**2 ПРОЕКТУВАННЯ СТРУКТУРИ ПРОГРАМНОГО ЗАБЕЗПЕЧННЯ. ОГЛЯД ВИКОРИСТАНИХ ТЕХНОЛОГІЙ ТА ПРОГРАМНИХ ЗАСОБІВ**

**2.1 Огляд архітектури MVC**

Model-View-Controller (MVC, «Модель-вигляд-поведінка», «Модель-представлення-контролер») - схема використання декількох шаблонів проектування, за допомогою яких модель даних програми, користувальницький інтерфейс і взаємодія з користувачем розділені на три окремих компоненти так , що модифікація одного з компонентів чинить мінімальний вплив на інші. Дана схема проектування часто використовується для побудови архітектурного каркасу, коли переходять від теорії до реалізації в конкретній предметній області .

Мал 2.1 схематичне зображення архітектури MVC

Розрізняють дві основні модифікації:  
1. Пасивна модель - модель не має жодних способів впливати на представлення або контролер, і використовується ними в якості джерела даних для відображення. Всі зміни моделі відстежуються контролером і він же відповідає за перемальовування представлення, якщо це необхідно. Така модель частіше використовується в структурному програмуванні, поскільки в цьому випадку модель являє собою просто структуру даних, без методів які їх оброблюють.  
2. Активна модель - модель оповіщає представлення про те, що в ній відбулися зміни, а представлення, які зацікавлені в оповіщенні, підписуються на ці повідомлення. Це дозволяє зберегти незалежність моделі як від контролера, так і від представлення.  
Класичною реалізацією концепції MVC прийнято вважати версію саме з активною моделлю.  
Основна мета застосування цієї концепції полягає в поділі бізнес-логіки (моделі) від її візуалізації (представлення, виду). За рахунок такого поділу підвищується можливість повторного використання. Найбільш корисне застосування даної концепції в тих випадках, коли користувач повинен бачити ті ж самі дані одночасно в різних контекстах і / або з різних точок зору. Зокрема, виконуються наступні завдання:  
1. До однієї моделі можна приєднати кілька представлень, при цьому не зачіпаючи реалізацію моделі. Наприклад, деякі дані можуть бути одночасно представлені у вигляді електронної таблиці, гістограми та кругової діаграми.  
2. Не зачіпаючи реалізацію представлень можна змінити реакції на дії користувача (натискання мишею на кнопці, введення даних), для цього досить використовувати інший контроллер.  
3. Ряд розробників спеціалізується тільки в одній з областей: або розробляють графічний інтерфейс, або розробляють бізнес-логіку. Тому можливо домогтися того, щоб програмісти, які займаються розробкою бізнес-логіки (моделі), взагалі не були проінформовані про те, яке представлення буде використовуватися.  
Концепція MVC дозволяє розділити дані, представлення та обробку дій користувача на три окремих компоненти:

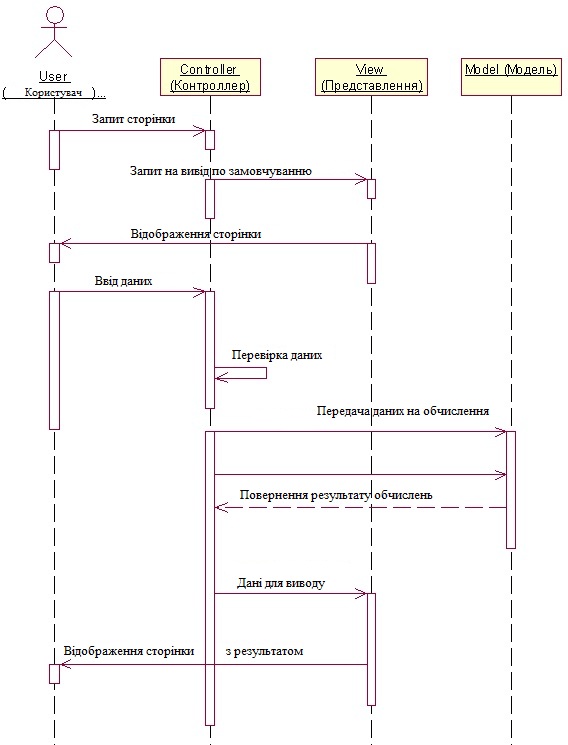
* Модель (англ. Model). Модель надає знання: дані та методи роботи з цими даними, реагує на запити, змінюючи свій стан. Не містить інформації, як ці знання можна візуалізувати.
* Представлення, вид (англ. View). Відповідає за відображення інформації (візуалізацію). Часто як представлення виступає форма (вікно) з графічними елементами.
* Контроллер (англ. Controller). Забезпечує зв'язок між користувачем і системою: контролює введення даних користувачем і використовує модель і представлення для реалізації необхідної реакції.

Важливо відзначити, що як представлення, так і контролер залежать від моделі. Однак модель не залежить ні від представлення, ні від контролера. Тим самим досягається призначення такого поділу: воно дозволяє будувати модель незалежно від візуального представлення, а також створювати кілька різних представлень для однієї моделі.Отже:  
*Представлення.* Модуль виведення інформації. Це може бути шаблонізатор чи щось подібне, метаою якого є тільки в подання інформації у вигляді HTML на основі будь-яких готових даних.  
*Контроллер.* Модуль управління введенням і виведенням даних. Даний модуль повинен стежити за переданими в систему даними (через форму, рядок запиту, куки або будь-яким іншим способом) та на основі введених даних вирішити:  
• Передавати їх у модель  
• Вивести повідомлення про помилку і запросити повторне введення (змусити модуль представлення оновити сторінку з урахуванням мінливих умов)  
Крім того, контролер зобов'язаний визначати тип даних, отриманих від моделі (чи це готовий результат, його відсутність, або повідомлення про помилку) і передавати інформацію в модуль представлення.  
*Модель.* Модуль, що відповідає за безпосередній розрахунок чого-небудь на основі отриманих від користувача даних. Результат, отриманий цим модулем, повинен бути переданий в контролер, і не повинен містити нічого, що відноситься до безпосереднього висновку (тобто має бути представлений у внутрішньому форматі програми).  
Уявімо собі як представлення як який-небудь клас, який за допомогою шаблонізатора виводить результат або повідомлення про помилку. На його вхід подається або масив з даними (об'єкт або що-небудь інше), або змінна, що містить текст з помилкою.  
В якості контролера буде виступати клас, що проводить всі необхідні перевірки коректності даних і генерує повідомлення про помилки. Перевірки даних доцільно помістити саме в клас контролера, їх використовують досить часто. Як варіант, можна просто наслідувати клас контролера від більш загального класу, що реалізовує перевірку вхідних даних за заданими правилами. Або, якщо так буде зручніше, включити в клас контролера клас або серію функцій перевірки даних.  
Цей же клас повинен передати дані, отримані в результаті роботи моделі, в клас представлення для виводу.  
На цій діаграмі показана послідовність дій, а також послідовність переданих даних: від користувача, до користувача і між модулями.  
Схема відображає типовий процес виведення форми, заповнення її користувачем і повернення користувачеві результатів. Ніяких помилок в даному випадку не відбувається.

*Діаграма послідовностей*

Суть архітектурного шаблону MVC полягає в тому, щоб чітко розділити представлення, управління та модель системи. Це дуже зручно, адже якщо що-небудь зміниться в одній з частин системи, інших частин ці зміни не торкнуться.

Це дуже зручно, коли треба зробити якісь зміни в програмі, наприклад змінити джерело даних. Або, наприклад, в моделі змінити пару розрахункових формул. Або в контролері прибрати пару обмежень, або змінити метод прийому-передачі даних. Якщо ж узяти до уваги принципи спадкування в ООП, то архітектура MVC стане ще зручнішою. Скажімо, коли є дві форми, що виглядають однаково, але з дещо відмінними алгоритмами розрахунку.



Мал 2.2 Діаграма послідовностей архітектури MVC

Як видно з діаграми, звернення до моделі відбувається лише у випадку передачі користувачем вірних даних. На внутрішньому ж рівні додатка, модель відділена від представлення і контролера. Контролер також відокремлений від моделі і представлення, і його функція полягає в управлінні і перевірці.  
Шаблони в MVC  
Для реалізації схеми Model-View-Controller використовується досить велика кількість шаблонів проектування (в залежності від складності архітектурного рішення), основні з яких «спостерігач», «стратегія», «компонувальник».  
Найбільш типова реалізація відокремлює представлення від моделі шляхом встановлення між ними протоколу взаємодії, використовуючи апарат подій (підписка / сповіщення). При кожній зміні внутрішніх даних в моделі вона оповіщає всіх залежних від неї представлень, і представлення оновлюється. Для цього використовується шаблон «спостерігач». При обробці реакції користувача представлення вибирає залежно від потрібної реакції, потрібний контролер, який забезпечить той чи інший зв'язок з моделлю. Для цього використовується шаблон «стратегія», або замість цього може бути модифікація з використанням шаблону «команда». А для можливості однотипного поводження з підоб'єктами складно-складового ієрархічного представлення може використовуватися шаблон «компонувальник». Крім того, можуть використовуватися й інші шаблони проектування, наприклад, «фабричний метод», який дозволить задати за замовчуванням тип контролера для відповідного представлення.

**2.2 Мова програмування C#**

**C#** (вимовляється Сі-шарп) — [об'єктно-орієнтована](http://uk.wikipedia.org/wiki/%D0%9E%D0%B1%E2%80%99%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [мова програмування](http://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) з безпечною [системою типізації](http://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%82%D0%B8%D0%BF%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%97) для платформи [.NET](http://uk.wikipedia.org/wiki/.NET). Розроблена [Андерсом Гейлсбергом](http://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B4%D0%B5%D1%80%D1%81_%D0%93%D0%B5%D0%B9%D0%BB%D1%81%D0%B1%D0%B5%D1%80%D0%B3), [Скотом Вілтамутом](http://uk.wikipedia.org/w/index.php?title=%D0%92%D1%96%D0%BB%D1%82%D0%B0%D0%BC%D1%83%D1%82_%D0%A1%D0%BA%D0%BE%D1%82&action=edit&redlink=1) та [Пітером Гольде](http://uk.wikipedia.org/w/index.php?title=%D0%93%D0%BE%D0%BB%D1%8C%D0%B4%D0%B5_%D0%9F%D1%96%D1%82%D0%B5%D1%80&action=edit&redlink=1) під егідою [Microsoft Research](http://uk.wikipedia.org/wiki/Microsoft_Research) (при фірмі [Microsoft](http://uk.wikipedia.org/wiki/Microsoft)).

[Синтаксис](http://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%81) C# близький до [С++](http://uk.wikipedia.org/wiki/%D0%A1%2B%2B) і [Java](http://uk.wikipedia.org/wiki/Java). [Мова](http://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0) має строгу статичну типізацію, підтримує [поліморфізм](http://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D1%96%D0%BC%D0%BE%D1%80%D1%84%D1%96%D0%B7%D0%BC_%28%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F%29), перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, [винятки](http://uk.wikipedia.org/wiki/%D0%9E%D0%B1%D1%80%D0%BE%D0%B1%D0%BA%D0%B0_%D0%B2%D0%B8%D0%BD%D1%8F%D1%82%D0%BA%D1%96%D0%B2), коментарі у форматі [XML](http://uk.wikipedia.org/wiki/XML). Перейнявши багато що від своїх попередників — мов [С++](http://uk.wikipedia.org/wiki/%D0%A1%2B%2B), [Delphi](http://uk.wikipedia.org/wiki/Delphi_%28%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F%29), [Модула](http://uk.wikipedia.org/w/index.php?title=%D0%9C%D0%BE%D0%B4%D1%83%D0%BB%D0%B0&action=edit&redlink=1) і [Smalltalk](http://uk.wikipedia.org/wiki/Smalltalk) — С#, спираючись на практику їхнього [використання](http://uk.wikipedia.org/wiki/%D0%92%D0%B8%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D0%B0%D0%BD%D0%BD%D1%8F), виключає деякі моделі, що зарекомендували себе як проблематичні при розробці програмних систем, наприклад [множинне спадкування](http://uk.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B6%D0%B8%D0%BD%D0%BD%D0%B5_%D1%81%D0%BF%D0%B0%D0%B4%D0%BA%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) класів (на відміну від C++).

C# є дуже близьким родичем мови [програмування](http://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [Java](http://uk.wikipedia.org/wiki/Java). Мова Java була створена компанією Sun Microsystems, коли глобальний розвиток інтернету поставив задачу роззосереджених обчислень. Взявши за основу популярну мову C++, Java виключила з неї потенційно небезпечні речі (типу вказівників без контролю виходу за межі). Для роззосереджених обчислень була створена концепція віртуальної машини та машинно-незалежного байт-коду, свого роду посередника між вихідним текстом програм і апаратними інструкціями комп'ютера чи іншого інтелектуального пристрою.

Java набула чималої популярності, і була ліцензована також і компанією [Microsoft](http://uk.wikipedia.org/wiki/Microsoft). Але з плином часу Sun почала винуватити Microsoft, що та при створенні свого клону Java робить її сумісною виключно з платформою [Windows](http://uk.wikipedia.org/wiki/Windows), чим суперечить самій концепції машинно-незалежного середовища виконання і порушує ліцензійну угоду. Microsoft відмовилася піти назустріч вимогам Sun, і тому з'ясування стосунків набуло статусу [судового](http://uk.wikipedia.org/wiki/%D0%A1%D1%83%D0%B4) процесу. Суд визнав позицію Sun справедливою, і зобов'язав Microsoft відмовитися від позаліцензійного використання Java.

У цій ситуації в Microsoft вирішили, користуючись своєю вагою на ринку, створити свій власний аналог Java, мови, в якій корпорація стане повновладним господарем. Ця новостворена мова отримала назву C#. Вона успадкувала від Java концепції віртуальної машини (середовище .NET), байт-коду (MSIL) і більшої безпеки вихідного коду програм, плюс врахувала досвід використання програм на Java.

Нововведенням C# стала можливість легшої взаємодії, порівняно з мовами-попередниками, з кодом програм, написаних на інших мовах, що є важливим при створенні великих проектів. Якщо програми на різних мовах виконуються на платформі [.NET](http://uk.wikipedia.org/wiki/.NET), .NET бере на себе клопіт щодо сумісності програм (тобто типів даних, за кінцевим рахунком).

Станом на сьогодні C# визначено флагманською мовою корпорації Microsoft, бо вона найповніше використовує нові можливості .NET. Решта мов програмування, хоч і підтримуються, але визнані такими, що мають спадкові прогалини щодо використання .NET.

Символ # у назві мови можна інтерпретувати і як дві пари плюсів ++, що натякають на новий крок в розвитку мови в порівнянні з C++ (подібно до кроку від C до C++), і як музичний символ [дієз](http://uk.wikipedia.org/wiki/%D0%94%D1%96%D1%94%D0%B7), разом з буквою C, що становить в англійській мові назву ноти [до-дієз](http://uk.wikipedia.org/w/index.php?title=%D0%94%D0%BE-%D0%B4%D1%96%D1%94%D0%B7&action=edit&redlink=1). Останнє і дало назву мові. Попри те, що символ [#](http://uk.wikipedia.org/wiki/) (октоторп) насправді є символом для позначення номера на більшості клавіатур і відрізняється від символу дієз ♯ (Unicode U+266F), Microsoft, як автор мови, неодноразово зверталася до своїх клієнтів з проханням прийняти таку стилізацію.

C# розроблявся як мова програмування прикладного рівня для [CLR](http://uk.wikipedia.org/wiki/CLR) і, як така, залежить від можливостей самої CLR. Це стосується, перш за все, системи типів C#. Присутність або відсутність тих або інших виразних особливостей мови диктується тим, чи може конкретна мовна особливість бути трансльована у відповідні конструкції CLR. Так, з розвитком CLR від версії 1.1 до 2.0 значно збагатився і сам C#; подібної взаємодії слід чекати і надалі. CLR надає C#, як і всім іншим .NET-орієнтованим мовам, багато можливостей, яких позбавлені «класичні» мови програмування. Наприклад, збірка сміття не реалізована в самому C#, а проводиться CLR для програм, написаних на C# точно так, як і це робиться для програм на VB.NET, J# тощо.

Версія 1.0

[Проект](http://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B5%D0%BA%D1%82) C# був початий в грудні 1998 і отримав кодову назву COOL (C-style Object Oriented Language). Версія 1.0 була анонсована разом з платформою .NET у червні 2000 року, тоді ж з'явилася і перша загальнодоступна бета-версія; C# 1.0 остаточно вийшов разом з Microsoft Visual Studio .NET у лютому 2002 року.

Перша версія C# нагадувала за своїми можливостями Java 1.4, дещо їх розширюючи: так, в C# були властивості (що виглядають у коді як поля об'єкта, але, при зверненні до них, можуть викликати пов'язані методи класу), індексатори (подібні до властивостей, але приймають параметр як індекс масиву), події, делегати, цикли foreach, структури, що передаються за значенням, автоматичне перетворення вбудованих типів в об'єкти при необхідності (boxing), атрибути, вбудовані засоби взаємодії з некерованим кодом (DLL, COM) тощо. Крім того, в C# вирішено було перенести деякі можливості C++, відсутні в Java: беззнакові типи, перевизначення операцій (з деякими обмеженнями, на відміну від C++), передача параметрів у метод за посиланням, методи зі змінним числом параметрів, оператор goto. Також у C# залишили обмежену можливість роботи з вказівниками — в місцях коду, спеціально позначених словом unsafe і при вказівці спеціальної опції компілятору.

Версія 2.0

Проект специфікації C# 2.0 вперше був викладений Microsoft в жовтні 2003 року; у 2004 році виходили бета-версії (проект з кодовою назвою Whidbey), C# 2.0 остаточно вийшов 7 листопада 2005 року разом з [Visual Studio](http://uk.wikipedia.org/wiki/Visual_Studio) 2005 і .NET 2.0.

**Нові можливості у версії 2.0**:

Часткові типи (розділення реалізації класу більш ніж на один файл).

Узагальнені, або параметризовані типи (generics, «дженерики»). На відміну від шаблонів C++, вони підтримують деякі додаткові можливості і працюють на рівні віртуальної машини. Разом з тим, параметрами узагальненого типу не можуть бути вирази.

Нова форма ітератора, що дозволяє створювати співпрограми за допомогою ключового слова yield, подібно [Python](http://uk.wikipedia.org/wiki/Python) і [Рубі](http://uk.wikipedia.org/wiki/%D0%A0%D1%83%D0%B1%D1%96).

Анонімні методи, що забезпечують функціональність замикання.

Оператор ??: return obj1 ?? obj2; означає (у нотації C# 1.0) return obj1!=null ? obj1 : obj2;.

Типи-значення, що обнуляються (nullable), (що позначаються знаком питання, наприклад, int? i = null;) є тими ж самими типами-значеннями, що можуть nfrj; приймати також значення null. Такі типи дозволяють поліпшити взаємодію з базами даних через мову SQL.

Версія 3.0

В червні 2004 року [Андерс Гейлсберг](http://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B4%D0%B5%D1%80%D1%81_%D0%93%D0%B5%D0%B9%D0%BB%D1%81%D0%B1%D0%B5%D1%80%D0%B3) вперше розповів на сайті Microsoft про плановані розширення мови в C#3.0.. У вересні 2005 року було випущено проект [специфікації](http://uk.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B5%D1%86%D0%B8%D1%84%D1%96%D0%BA%D0%B0%D1%86%D1%96%D1%8F) C# 3.0 і бета-версія C# 3.0, що встановлюється у вигляді доповнення до існуючих [Visual Studio](http://uk.wikipedia.org/wiki/Visual_Studio) 2005 і [.NET](http://uk.wikipedia.org/wiki/.NET) 2.0. Офіційно версія C# 3.0 побачила світ [19 листопада](http://uk.wikipedia.org/wiki/19_%D0%BB%D0%B8%D1%81%D1%82%D0%BE%D0%BF%D0%B0%D0%B4%D0%B0) [2007 року](http://uk.wikipedia.org/wiki/2007) у складі [.NET Framework 3.5](http://uk.wikipedia.org/w/index.php?title=.NET_Framework_3.5&action=edit&redlink=1).

В C# 3.0 з'явилися такі радикальні доповнення та зміни:

Ключові слова select, from, where, що дозволяють робити запити з [SQL](http://uk.wikipedia.org/wiki/SQL), [XML](http://uk.wikipedia.org/wiki/XML), колекції тощо (запит, інтегрований в мову, [англ.](http://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) [Language Integrated Query](http://uk.wikipedia.org/w/index.php?title=Language_Integrated_Query&action=edit&redlink=1), або LINQ)

Методи-розширення — додавання методу в існуючий клас за допомогою ключового слова this при першому параметрі статичної функції.

C# 3.0 буде сумісний з C# 2.0 за генерованим [MSIL](http://uk.wikipedia.org/w/index.php?title=MSIL&action=edit&redlink=1)-кодом; поліпшення в мові — чисто синтаксичні і реалізуються на етапі [компіляції](http://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%96%D0%BB%D1%8F%D1%82%D0%BE%D1%80). Наприклад, багато з інтегрованих запитів LINQ можна реалізувати в поточних версіях використовуючи безіменні делегати в поєднанні з предикативними методами над контейнерами, на кшталт List.FindAll і List.RemoveAll.

Версія 4.0

Нові можливості в версії 4.0:

* Динамічна типізація об'єктів:
* Динамічний перехід,
* Динамічний виклик методу.

Приклад:

dynamic calc = GetCalculator();

int sum = calc.Add(10, 20);

C# має «препроцесорні директиви» (хоча насправді він не має [препроцесора](http://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%B5%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D0%BE%D1%80)) на основі препроцесора C, це дає програмісту можливість визначити символи, але не [макроси](http://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D0%BA%D1%80%D0%BE%D1%81). Умовні директиви, такі як #if, #endif, чи #else також можливі. Директиви типу #region дають натяк редактору для згортання фрагментів коду.

Специфікація C# визначає мінімальний набір бібліотек типів і класів, на який має розраховувати [компілятор](http://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%96%D0%BB%D1%8F%D1%82%D0%BE%D1%80). На практиці, C# найчастіше використовується з якоюсь реалізацією [Common Language Infrastructure](http://uk.wikipedia.org/w/index.php?title=Common_Language_Infrastructure&action=edit&redlink=1) (CLI), яка стандартизована як ECMA-335 Common Language Infrastructure (CLI).

C# стандартизований в [ECMA](http://uk.wikipedia.org/wiki/ECMA) та [ISO](http://uk.wikipedia.org/wiki/ISO).

У серпні 2000 [Microsoft Corporation](http://uk.wikipedia.org/wiki/Microsoft_Corporation), [Hewlett-Packard](http://uk.wikipedia.org/wiki/Hewlett-Packard) та [Intel Corporation](http://uk.wikipedia.org/wiki/Intel_Corporation) виступили ко-спонсорами стандартизації специфікації мови C#, а також [Common Language Infrastructure](http://uk.wikipedia.org/w/index.php?title=Common_Language_Infrastructure&action=edit&redlink=1) ([CLI](http://uk.wikipedia.org/w/index.php?title=CLI&action=edit&redlink=1)) в організації зі стандартизації [ECMA International](http://uk.wikipedia.org/wiki/ECMA_International). У грудні 2001 ECMA випустила ECMA-334 Специфікація мови C#. C# стала стандартом ISO у 2003 (ISO/IEC 23270:2006 — Information technology—Programming languages—C#). До того ECMA ще встигла адоптувати еквівалентну специфікацію як другк редакцію C# у грудні 2002.

У червні 2005 ECMA схвалила редакцію 3 специфікації C#, і відредагувала ECMA-334. Доповнення включали часткові класи, анонімні методи, тип null, і генерики (аналоги шаблонів C++).

У липні 2005 ECMA подала стандарти і відповідні технічні умови на ISO/IEC JTC 1 через пришвидшену процедуру (Fast-Track). Цей процес звичайно займає 6-9 місяців.

Хоча визначення мови C# і CLI стандартизовані ISO та Ecma, що забезпечує розумний і недискримінаційний ліцензійний захист (RAND) від патентних позовів, Microsoft використовує C# і CLI у своїй бібліотеці Base Class Library (BCL), яка є фундаментом їхньої платформи [.NET framework](http://uk.wikipedia.org/wiki/.NET_framework), і яка забезпечує безліч нестандартизованих класів (розширений I/O, [GUI](http://uk.wikipedia.org/wiki/GUI) Windows Forms, веб-служби тощо). У деяких випадках, де патенти Microsoft відносяться до стандартів, використаних у .NET framework, документовані Microsoft, і застосовані патенти доступні через інші RAND умови або через Обітницю Відкритої Специфікації Microsoft (Microsoft's Open Specification Promise, OSP), які випускають патентні права публічно. Але є деякі застереження і обговорення, що існують додаткові аспекти, патентовані Microsoft, що не покриті, які можуть утримувати незалежних реалізаторів повного фреймворку. Microsoft також погодився не позиватися проти розробників [відкритого програмного забезпечення](http://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D0%BA%D1%80%D0%B8%D1%82%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) щодо порушення прав у неприбуткових проектах для частини свого фреймворку, покритого OSP. Microsoft погодився не порушувати патентних вимог щодо продуктів [Novell](http://uk.wikipedia.org/wiki/Novell) проти платних клієнтів Novell за винятком переліку продуктів, що явно не згадують C#, .NET чи реалізацію .NET від Novell (проект [Mono](http://uk.wikipedia.org/wiki/Mono)). Проте Novell дотримується точки зору, що Mono не порушує жодного патенту Microsoft. Microsoft також уклав спеціальну угоду не позиватися проти браузерного [плагіну](http://uk.wikipedia.org/wiki/%D0%9F%D0%BB%D0%B0%D0%B3%D1%96%D0%BD) [Moonlight](http://uk.wikipedia.org/wiki/Moonlight), який спирається на Mono, отриманного від [Novell](http://uk.wikipedia.org/wiki/Novell).

У зауваженні, опублікованому на сайті новин [Free Software Foundation](http://uk.wikipedia.org/wiki/Free_Software_Foundation) у червні 2009 [Річард Столлман](http://uk.wikipedia.org/wiki/%D0%A0%D1%96%D1%87%D0%B0%D1%80%D0%B4_%D0%A1%D1%82%D0%BE%D0%BB%D0%BB%D0%BC%D0%B0%D0%BD) попереджає, що він вважає, що «Microsoft можливо планує одного дня оголосити всі вільні реалізації C# такими, що використовують програмні патенти» і рекомендував розробникам уникати того, що він називає «безвідплатним ризиком», пов'язаним із «залежністю вільних реалізацій C#». Free Software Foundation пізніше повторила свої попередження, стверджуючи, що розширення Microsoft Community Promise на специфікації ECMA C# і CLI[[14]](http://uk.wikipedia.org/wiki/C_Sharp#cite_note-14) можуть не вберігти від шкідництва Microsoft відкритим реалізаціям C#, оскільки багато специфічних для Windows бібліотек, включених у .NET та Mono, не покриті цими обіцянками. Тому більшість провідних дистрибутивів Лінукс, за винятком Novell [SUSE Linux](http://uk.wikipedia.org/wiki/SUSE_Linux), не включають Mono в установку за замовчанням (хоча його можна підгрузити із репозитаріїв).

Титульним [компілятором](http://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%96%D0%BB%D1%8F%D1%82%D0%BE%D1%80) C# є [Microsoft Visual C#](http://uk.wikipedia.org/w/index.php?title=Microsoft_Visual_C&action=edit&redlink=1).

Існують інші компілятори C#, часто вони включають реалізації Common Language Infrastructure і бібліотеки класів .NET:

Проект Microsoft Rotor (який тепер зветься Shared Source Common Language Infrastructure, ліцензований тільки для навчального і дослідницького використання) забезпечує реалізації [CLR](http://uk.wikipedia.org/wiki/CLR) runtime і компілятор C#, і підмножину бібліотек фреймворка Common Language Infrastructure, відповідно до специфікації ECMA (до C# 2.0, і з підтримкою тільки [Windows XP](http://uk.wikipedia.org/wiki/Windows_XP)).

Проект [Mono](http://uk.wikipedia.org/wiki/Mono), початий компанією [Ximian](http://uk.wikipedia.org/w/index.php?title=Ximian&action=edit&redlink=1) і продовжений її покупцем і наступником [Novell](http://uk.wikipedia.org/wiki/Novell), забезпечує відкритий компілятор C#, повну відкриту реалізацію Common Language Infrastructure, включаючи потрібні бібліотеки фреймворка відповідно до специфікації ECMA, і близьку до повної реалізацію власницьких бібліотек класів Microsoft .NET до .NET 2.0, але не специфічних бібліотек .NET 3.0 і .NET 3.5, як для Mono 2.0.

Проект [DotGNU](http://uk.wikipedia.org/w/index.php?title=DotGNU&action=edit&redlink=1) також надає відкритий компілятор C#, близьку до повної реалізацію Common Language Infrastructure, включаючи потрібні бібліотеки фреймворка відповідно до специфікації ECMA, і підмножину деяких залишених власницьких біліотек класів Microsoft .NET до .NET 2.0 (які не документовані або не включені у специфікації ECMA, але включені у стандартне визначення Microsoft .NET Framework).

[DotNetAnywhere](http://dotnetanywhere.org/) Micro Framework Common Language Runtime націлений на вбудовані системи, і підтримує майже всі специфікації C# 2.0.

**2.3 Середовище розробки Microsoft Visual Studio**

Microsoft Visual Studio — серія продуктів фірми [Майкрософт](http://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D0%B9%D0%BA%D1%80%D0%BE%D1%81%D0%BE%D1%84%D1%82), які включають [інтегроване середовище розробки](http://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D1%81%D0%B5%D1%80%D0%B5%D0%B4%D0%BE%D0%B2%D0%B8%D1%89%D0%B5_%D1%80%D0%BE%D0%B7%D1%80%D0%BE%D0%B1%D0%BA%D0%B8) програмного забезпечення та ряд інших інструментальних засобів. Ці продукти дозволяють розробляти як [консольні програми](http://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D1%83%D0%B2%D0%B0%D1%87%D0%B0), так і програми з [графічним інтерфейсом](http://uk.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D1%96%D1%87%D0%BD%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81), в тому числі з підтримкою технології [Windows Forms](http://uk.wikipedia.org/w/index.php?title=Windows_Forms&action=edit&redlink=1), а також [веб-сайти](http://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B0%D0%B9%D1%82), [веб-застосунки](http://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA), [веб-служби](http://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%BB%D1%83%D0%B6%D0%B1%D0%B0) як в [рідному](http://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%B8%D0%B9_%D0%BA%D0%BE%D0%B4), так і в [керованому](http://uk.wikipedia.org/wiki/%D0%9A%D0%B5%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B9_%D0%BA%D0%BE%D0%B4) кодах для всіх платформ, що підтримуються [Microsoft Windows](http://uk.wikipedia.org/wiki/Microsoft_Windows), [Windows Mobile](http://uk.wikipedia.org/wiki/Windows_Mobile), [Windows CE](http://uk.wikipedia.org/wiki/Windows_CE), [.NET Framework](http://uk.wikipedia.org/wiki/.NET_Framework), [.NET Compact Framework](http://uk.wikipedia.org/wiki/.NET_Compact_Framework) та [Microsoft Silverlight](http://uk.wikipedia.org/wiki/Microsoft_Silverlight).

Visual Studio включає один або декілька з наступних компонентів:

* [Visual Basic .NET](http://uk.wikipedia.org/wiki/Visual_Basic_.NET), а до його появи — [Visual Basic](http://uk.wikipedia.org/wiki/Visual_Basic)
* [Visual C++](http://uk.wikipedia.org/wiki/Microsoft_Visual_C%2B%2B)
* Visual C#
* Visual J#
* Visual F# (входить до складу Visual Studio 2010);
* [Visual Studio Debugger](http://uk.wikipedia.org/wiki/Visual_Studio_Debugger)

Багато варіантів постачання також включають:

* [Microsoft SQL Server](http://uk.wikipedia.org/wiki/Microsoft_SQL_Server) або

MSDE Visual Source Safe — файл-серверна [система управління версіями](http://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%96%D0%BD%D0%BD%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D1%96%D1%8F%D0%BC%D0%B8)

У минулому, до складу Visual Studio також входили продукти:

* Visual InterDev;
* Visual J++;
* Visual J#;
* [Visual FoxPro](http://uk.wikipedia.org/w/index.php?title=Visual_FoxPro&action=edit&redlink=1);
* Visual Source Safe – файл-серверна [система управління версіями](http://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%96%D0%BD%D0%BD%D1%8F_%D0%B2%D0%B5%D1%80%D1%81%D1%96%D1%8F%D0%BC%D0%B8).

Найзначиміші версії пакету:

**Visual Studio 97** — перша випущена версія Visual Studio, в ній вперше були зібрані разом різні засоби розробки ПЗ. Вона була випущена в двох версія Professional і Enterprise і включала Visual Basic 5.0, Visual C++ 5.0, Visual J++ 1.1, Visual FoxPro 5.0, вперше з'явилося середовище розробки [ASP](http://uk.wikipedia.org/wiki/Active_Server_Pages) — Visual InterDev.Visual Studio 97 — була першою спробою Microsoft створити єдине середовище для розробки на різних мовах програмування: Visual C++, Visual J++, Visual InterDev, і MSDN використовували одне середовище, зване Developer Studio. Visual Basic і Visual FoxPro використовували окремі середовища для розробки.

**Visual Studio 6.0** — випущена в червні 1998 — остання версія Visual Studio, що працює на платформі Win9x. Як і раніше популярна серед програмістів, що використовували [Visual Basic](http://uk.wikipedia.org/wiki/Visual_Basic). Дана версія була основним середовищем розробки [застосунків](http://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) під [Windows](http://uk.wikipedia.org/wiki/Windows) від Microsoft, до появи платформи .NET. Ця версія була базовою для розробників Microsoft’у в продовж наступних чотирьох років. Visual Studio 6.0 була останньою версією яка включала в себе COM версію Visual Basic’а. Також це була остання версія яка включала мову програмування Visual J++. Існували дві версії Visual Studio 6.0: Professional та Enterprise. Enterprise містив у собі додаткові плагіни, які не були присутні в Professional, включаючи:Application Performance Explorer, Automation Manager, Microsoft Visual Modeler, RemAuto Connection Manager, Visual Studio Analyzer.

**Visual Studio .NET** (кодове ім'я Rainier; внутрішня версія 7.0) — випущена в лютому 2002 (влючала .NET Framework 1.0). Service Pack 1 для Visual Studio .NET (2002) випущений в березні 2005.Бета – версія була доступною в 2001 році. Найбільшою зміною було впровадження менеджера коду. [Застосунки](http://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) які були розроблені за допомогою Visual Studio .NET не компілювались в машинну мову, а перетворювались в формат, який мав назву Microsoft Intermediate Language (MSIL) або Common Intermediate Language (CIL). Коли MSIL – [застосунок](http://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) використовували, він автоматично компілювався в машинну мову для даної платформи, це робило код кросплатформенним, що дозволяло виконувати його на різних платформах. Проте такі [застосунки](http://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) могли використовуватись тільки в платформах що підтримували Common Language Infrastructure. Це робило можливим використання [застосунків](http://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) в операційних системах [Linux](http://uk.wikipedia.org/wiki/Linux) або [Max OS](http://uk.wikipedia.org/w/index.php?title=Max_OS&action=edit&redlink=1) використовуючи спеціальні програми як, Mono та DotGNU.

Visual Studio .NET 2002 вийшла одразу в чотирьох версіях:

* Academic
* Professional
* Enterprise Developer
* Enterprise Architect

Було вперше представлено нову мову програмування C# (сі – шарп), яка була спеціально розроблена для використання в Visual Studio .NET. Також було представлено наслідника Visual J++ що мав назву Visual J#. За допомогою Visual Studio .NET можна було створювати звичайні [застосунки](http://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) та [веб](http://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1) – [сайти](http://uk.wikipedia.org/wiki/%D0%A1%D0%B0%D0%B9%D1%82) (використовуючи ASP.NET та Web сервіси). Було випущено пакет оновлень для Visual Studio .NET в травні 2005 року.

Visual Studio .NET 2003 (кодове ім'я Everett; внутрішня версія 7.1) — випущена в квітні 2003 (влючаєт .NET Framework 1.1). Це була перша версія, що дозволяла розробку за стосунків для мобільних пристроїв, використовуючи ASP.NET або .NET Compact Framework. Внутрішній номер версії Visual Studio .NET 2003 був 7.1, але версії файлів були 8.0. Visual Studio .NET 2003 також було випущено в чотирьох варіантах: Academic, Professional, Enterprise Developer, та Enterprise Architect. Версія Enterprise Architect містила спеціальний [застосунок](http://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) Microsoft Visio 2002, що використовувався для побудови [UML](http://uk.wikipedia.org/wiki/UML) об’єктів. Пакет оновлень для Visual Studio .NET 2003 було випущено 13 вересня 2006 року.

Visual Studio 2005 (кодове ім'я Whidbey; внутрішня версія 8.0) — випущена в кінці жовтня 2005 (влючає .NET Framework 2.0). На початку листопада 2005 також вийшла серія продуктів в редакції Express: Visual C++ 2005 Express, Visual Basic 2005 Express, Visual C# 2005 Express і інше. 19 квітня 2006 редакція Express стала безоплатною. Service Pack 1 для VS2005 і всіх Express-редакцій випущений 14 грудня 2006 року. Додаткова латка для SP1, що вирішує проблему сумісності з [Windows Vista](http://uk.wikipedia.org/wiki/Windows_Vista) випущена 3 червня 2007.

Visual Studio 2005 підтримує ASP .NET версії 2.0, і дозволяє підтримувати он–лайн сервіси ASP .NET. Також підтримує усі тими SQL Server’ів до 2005 року. Було надано можливості для розробки 64 – бітних [застосунків](http://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA). Можна було компілювати коди ваших [застосунків](http://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) як 32 – або 64 – розрядні. Visual Studio 2005 містила 64 – бітні версії стандартних бібліотек. Також було випущено ще два продукти, які мали назву, Visual Studio Tools for Applications (VSA) та Visual Basic for Applications (VBA). В них було включено підтримку [Microsoft Office 2007](http://uk.wikipedia.org/wiki/Microsoft_Office_2007). Згодом була добавлена підтримка таких додатків як, WPF, WCF, WF, LINQ та .NET Framework 3.5.

В листопаді 2007 корпорація Microsoft оголосила про випуск нових продуктів для розробників Visual Studio 2008 (кодове ім'я Orcas) і .NET Framework 3.5.

Visual Studio 2008 сконцентрувала свою увагу на розробці застосунків для [Windows Vista](http://uk.wikipedia.org/wiki/Windows_Vista), [Microsoft Office 2007](http://uk.wikipedia.org/wiki/Microsoft_Office_2007) та [веб](http://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1) – [застосунків](http://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA). Для візуальної розробки було презентовано Windows Presentation Foundation та новий HTML/CSS редактор. Маючи понад 250 нових функціональних можливостей, Visual Studio 2008 пропонує істотні поліпшення кожної з версій, включаючи Visual Studio Express та Visual Studio Team System.

Visual Studio Team System підтримує управління збіркою програм, включаючи виконання планових збірок та збірок в результаті процесу безперервної інтеграції. Team Build забезпечує інтегровану підтримку статичного аналізу коду під час виконання збірки і проведення контрольних випробувань збірки.

Значне спрощення розробки для Web завдяки новій техніці обміну інформацією з [веб-сервером](http://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80) для веб-сайтів, які підтримують [AJAX](http://uk.wikipedia.org/wiki/AJAX)/[JSON](http://uk.wikipedia.org/wiki/JSON). Нові елементи управління ASP.NET передбачають покращене управління сторінками і шаблони, а Windows Communication Foundation передбачає вбудовану підтримку [RSS](http://uk.wikipedia.org/wiki/RSS) і [REST](http://uk.wikipedia.org/wiki/REST).

.NET Framework 3.5 також містить декілька нових функціональних можливостей, серед яких можна назвати можливості для [Web 2.0](http://uk.wikipedia.org/wiki/Web_2.0), сервіс-орієнтовану архітектуру (Service-Oriented Architecture, SOA) та програми на базі технології ПЗ + Сервіси (Software+Services). Сервіси з підтримкою послідовності операцій надають нові класи моделі програмування, які спрощують створення сервісів з підтримкою послідовності операцій за рахунок використання Windows Communication Foundation і Windows Workflow Foundation. Це дозволяє розробникам на .NET Framework створювати бізнес-логіку сервісу, використовуючи WF, та організовувати обмін повідомленнями з цим сервісом за допомогою [WCF](http://uk.wikipedia.org/wiki/WCF).

Підтримка додаткових протоколів веб-сервісів у Windows Communication Foundation, включаючи протоколи Web Services Atomic Transaction (WS-AtomicTransaction) 1.1, WS-ReliableMessaging 1.1, WS-Secure Conversation та Web Services Coordination (WS-Coordination) 1.1.

[Text Template Transformation Toolkit](http://uk.wikipedia.org/wiki/Text_Template_Transformation_Toolkit) шаблонно орієнтований генератор коду включений як частина середовища.

Visual Studio 2010 була редставлена [12 квітня](http://uk.wikipedia.org/wiki/12_%D0%BA%D0%B2%D1%96%D1%82%D0%BD%D1%8F) [2010](http://uk.wikipedia.org/wiki/2010) року. Включає [.NET Framework](http://uk.wikipedia.org/wiki/.NET_Framework) 4.0. З'явилася нова мова [F#](http://uk.wikipedia.org/wiki/F_Sharp), [Visual C++](http://uk.wikipedia.org/wiki/Visual_C%2B%2B) підтримує стандарт C++0x.

Інструменти Visual Studio 2010 допоможуть не тільки в створенні звичних програм для мобільних телефонів і персональних комп'ютерів, але в розробці [хмарних](http://uk.wikipedia.org/wiki/%D0%A5%D0%BC%D0%B0%D1%80%D0%BD%D1%96_%D0%BE%D0%B1%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%BD%D1%8F) застосунків. При цьому процес тестування, [зневадження](http://uk.wikipedia.org/wiki/%D0%97%D0%BD%D0%B5%D0%B2%D0%B0%D0%B4%D0%B6%D0%B5%D0%BD%D0%BD%D1%8F) і розгортання програм в «хмарі» аналогічний створенню .NET-застосунків. Іншим важливим доповненням в Visual Studio 2010 є інструменти для [багатониткової](http://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B3%D0%B0%D1%82%D0%BE%D0%BD%D0%B8%D1%82%D0%B5%D0%B2%D1%96%D1%81%D1%82%D1%8C) розробки з використанням як некерованого коду, так і [.NET Framework](http://uk.wikipedia.org/wiki/%D0%9A%D0%B5%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B9_%D0%BA%D0%BE%D0%B4).

У Visual Studio 2010 повністю перероблений інтерфейс з використанням [Windows Presentation Foundation](http://uk.wikipedia.org/wiki/Windows_Presentation_Foundation) (WPF), упроваджено наступне покоління інструментів [ASP.NET](http://uk.wikipedia.org/wiki/ASP.NET), є підтримка динамічних розширень в мовах програмування C# і Visual Basic, використовуються нові шаблони проектів, інструментарій для [документування](http://uk.wikipedia.org/wiki/%D0%94%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) тестових сценаріїв і велика кількість нових [бібліотек](http://uk.wikipedia.org/wiki/%D0%91%D1%96%D0%B1%D0%BB%D1%96%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_%28%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F%29), що підтримують [Windows 7](http://uk.wikipedia.org/wiki/Windows_7).

Visual Studio Ultimate 2010, формально це Visual Studio Team System 2010, кодове ім’я Rosario, це новий інструмент для спільної розробки застосунків.

**Visual Studio 2012**

Представлений [2 серпня](http://uk.wikipedia.org/wiki/2_%D1%81%D0%B5%D1%80%D0%BF%D0%BD%D1%8F) [2012](http://uk.wikipedia.org/wiki/2012) року. Включає [.NET Framework](http://uk.wikipedia.org/wiki/.NET_Framework) 4.5. Головні нововведення це підтримка [Windows RunTime](http://uk.wikipedia.org/wiki/Windows_RunTime), [C++/CX](http://uk.wikipedia.org/w/index.php?title=C%2B%2B/CX&action=edit&redlink=1)(Component Extensions), бібліотека [C++ AMP](http://uk.wikipedia.org/wiki/C%2B%2B_AMP) для [GPGPU](http://uk.wikipedia.org/wiki/GPGPU) програмування, [компілятор](http://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%96%D0%BB%D1%8F%D1%82%D0%BE%D1%80) [Visual C++](http://uk.wikipedia.org/wiki/Visual_C%2B%2B) майже підтримує стандарт [C++11](http://uk.wikipedia.org/wiki/C%2B%2B11). З'явився новий тип проектів, котрі дозволяють писати рідні [застосунки](http://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) (у стилі [Windows Metro](http://uk.wikipedia.org/w/index.php?title=Windows_Metro&action=edit&redlink=1)) для операційної системи [Windows 8](http://uk.wikipedia.org/wiki/Windows_8).

Доповнення (Add-Ins)

Visual Studio побудована в архітектурі, що підтримує можливість використання доповнень (Add-Ins), — плагінів від сторонніх розробників, що дозволяє розширювати можливості середовища розробки.

Деякі з найпопулярніших доповнень:

* DevPartner Studio
* Visual Assist
* [ReSharper](http://uk.wikipedia.org/wiki/ReSharper)
* [IncrediBuild](http://uk.wikipedia.org/w/index.php?title=IncrediBuild&action=edit&redlink=1)
* Workspace Whiz
* Viva64

**2.4 Опис технології ASP.NET**

ASP.NET — технологія створення [веб-застосунків](http://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) і [веб-сервісів](http://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D1%96%D1%81) від компанії [Майкрософт](http://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D0%B9%D0%BA%D1%80%D0%BE%D1%81%D0%BE%D1%84%D1%82). Вона є складовою частиною платформи [Microsoft .NET](http://uk.wikipedia.org/wiki/Microsoft_.NET) і розвитком старішої технології [Microsoft ASP](http://uk.wikipedia.org/w/index.php?title=Microsoft_ASP&action=edit&redlink=1). У цей час останньою версією цієї технології є ASP.NET 4.5.

ASP.NET зовні багато в чому зберігає схожість із старішою технологією ASP, що дозволяє розробникам відносно легко перейти на ASP.NET. У той же час внутрішній устрій ASP.NET істотно відрізняється від ASP, оскільки вона заснована на платформі. NET і, отже, використовує всі нові можливості, що надаються цією платформою.

|  |
| --- |
|  |

Після випуску сервера Internet Information Services 4.0 в 1997 році, компанія Microsoft почала досліджувати можливість нової моделі веб-застосунків, яка задовольнить скарги на ASP, особливо пов'язані з відділенням оформлення від змісту, і яка дозволить писати «чистий» код. Робота з розробки такої моделі була доручена Марку Андерсу, менеджеру команди IIS, і Скотту Гутрі, що надійшов на роботу в Microsoft в 1997. Андерс і Гутрі розробили початковий проект протягом двох місяців, і Гутрі написав код первісного прототипу під час різдвяних канікул 1997 року.

Початковий проект називався «XSP»; Гутрі пояснив в інтерв'ю 2007 року що, «завжди запитують, що означає буква X. У той час вона нічого не значила. XML починається з неї; XSLT починається з неї. Все кльове починається з X, тому ми його так і назвали.» Прототип XSP був написаний на Java, але скоро було вирішено побудувати нову платформу на основі Common Language Runtime (CLR), бо на платформу Java у компанії Microsoft закінчувалась ліцензія. Гутрі описав це рішення як «величезний ризик», тому що успіх нової розробки був пов'язаний з успіхом CLR, яка, як і XSP, перебувала на ранній стадії розробки.

Хоча ASP.NET бере свою назву від старої технології Microsoft ASP, вона значно від неї відрізняється. Microsoft повністю перебудувала ASP.NET, грунтуючись на [Common Language Runtime](http://uk.wikipedia.org/wiki/Common_Language_Runtime) (CLR), який є основою всіх [застосунків](http://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) Microsoft. NET. Розробники можуть писати код для ASP.NET, використовуючи практично будь-які мови програмування, що входять у комплект. NET Framework (C#, Visual Basic.NET, і JScript. NET). ASP.NET має перевагу у швидкості в порівнянні зі скриптовими технологіями, тому що при першому зверненні код компілюється і поміщається в спеціальний кеш, і згодом тільки виконується, не вимагаючи витрат часу на парсинг, оптимізацію, і т. д.

Переваги ASP.NET перед ASP :

* Компільований код виконується швидше, більшість помилок відловлюється ще на стадії розробки
* Значно поліпшена обробка помилок часу виконання, з використанням [блоків try .. catch](http://uk.wikipedia.org/wiki/%D0%9E%D0%B1%D1%80%D0%BE%D0%B1%D0%BA%D0%B0_%D0%B2%D0%B8%D0%BD%D1%8F%D1%82%D0%BA%D1%96%D0%B2)
* Користувальницькі елементи управління (controls) дозволяють виділяти часто використовувані шаблони, такі як меню сайту
* Використання метафор, які вже застосовуються в [Windows](http://uk.wikipedia.org/wiki/Windows)-застосунках, наприклад, таких як елементи керування та події
* Розширюваний набір елементів управління і бібліотек класів дозволяє швидше розробляти застосунки
* ASP.NET спирається на багатомовні можливості .NET, що дозволяє писати код сторінок на [VB.NET](http://uk.wikipedia.org/wiki/VB.NET), [Delphi.NET](http://uk.wikipedia.org/w/index.php?title=Delphi.NET&action=edit&redlink=1), [Visual C/C++](http://uk.wikipedia.org/wiki/Visual_C/C%2B%2B) тощо
* Можливість кешування всієї сторінки або її частини для збільшення продуктивності
* Можливість кешування даних, що використовуються на сторінці
* Можливість поділу візуальної частини та бізнес-логіки з різних файлів («code behind»)
* Розширювана модель обробки запитів
* Розширена подієва модель
* Розширювана модель серверних елементів керування
* Наявність master-сторінок для завдання шаблонів оформлення сторінок
* Підтримка [CRUD](http://uk.wikipedia.org/wiki/CRUD)-операцій при роботі з таблицями через GridView
* Вбудована підтримка [AJAX](http://uk.wikipedia.org/wiki/AJAX)
* ASP.NET має перевагу у швидкості в порівнянні з іншими технологіями, заснованими на [скриптах](http://uk.wikipedia.org/wiki/%D0%A1%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B2%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0).

Тут можна навести певні порівняння. Так, ASP — похідна від [Win32](http://uk.wikipedia.org/wiki/Win32), [XML](http://uk.wikipedia.org/wiki/XML) і [HTML](http://uk.wikipedia.org/wiki/HTML); [PHP](http://uk.wikipedia.org/wiki/PHP) — від XML, HTML, [Java](http://uk.wikipedia.org/wiki/Java) і CDI, тоді ASP.NET — від HTML і .NET (XML і [XAML](http://uk.wikipedia.org/w/index.php?title=XAML&action=edit&redlink=1) відповідно). При цьому, якщо зазвичай [Rich Media Application](http://uk.wikipedia.org/w/index.php?title=Rich_Media_Application&action=edit&redlink=1) створюють за допомогою [Flash](http://uk.wikipedia.org/wiki/Flash), тепер це робиться за допомогою модуля [Silverlight](http://uk.wikipedia.org/wiki/Silverlight), так само через сам ASP.NET. ASP.NET — багатше середовище для розробки та розгортання веб-ресурсів. У ASP.NET можна працювати з будь-якою .NET мовою, аж до [Managed C++](http://uk.wikipedia.org/wiki/%D0%9A%D0%B5%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B9_%D0%BA%D0%BE%D0%B4) і [Visual Basic](http://uk.wikipedia.org/wiki/Visual_Basic), що дозволяє не замислюватися про перехід на C#.

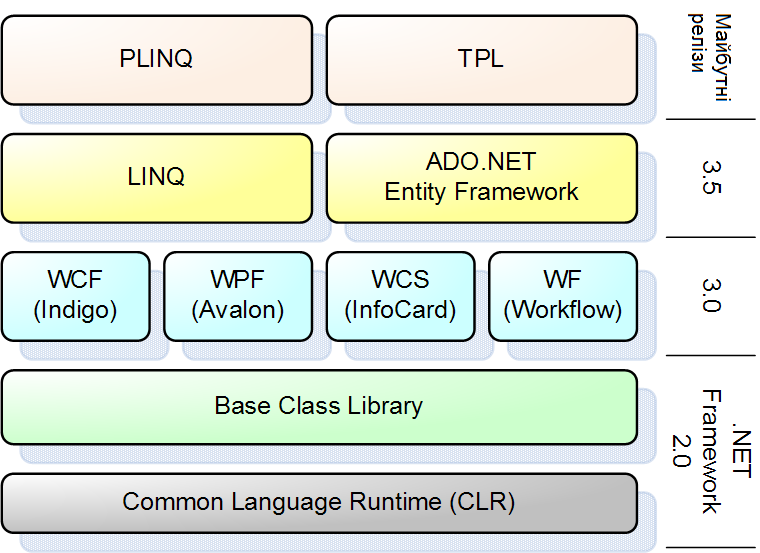
Корпорація Майкрософт випустила кілька розширень для ASP.NET:

ASP.NET [AJAX](http://uk.wikipedia.org/wiki/AJAX)

ASP.NET [MVC](http://uk.wikipedia.org/wiki/MVC) Framework

**2.5 Технологія доступу до даних ADO.NET Entity Framework**

ADO.NET Entity Framework (EF) - об'єктно-орієнтована технологія доступу до даних, є object-relational mapping (ORM) рішенням для. NET Framework від Microsoft. Надає можливість взаємодії з об'єктами як за допомогою LINQ у вигляді LINQ to Entities, так і з використанням Entity SQL. Для полегшення побудови web-рішень використовується як ADO.NET Data Services (Astoria), так і в зв'язці з Windows Communication Foundation і Windows Presentation Foundation, що дозволяє будувати багаторівневі програми, реалізуючи один з шаблонів проектування MVC, MVP або MVVM.  
LINQ to Entities  
Це альтернативний інтерфейс LINQ API, використовуваний для звернення до бази даних. Він відокремлює сутнісну об'єктну модель даних від фізичної бази даних, вводячи логічне відображення між ними. Так наприклад, схеми реляційних баз даних не завжди підходять для побудови об'єктно-орієнтованих додатків і в результаті ми маємо об'єктну модель програми, що істотно відрізняється від логічної моделі даних, в цьому випадку використовується LINQ to Entities, який використовує модель EDM (Entity Data Model). Тобто, якщо вам потрібно послабити зв'язок між вашою сутнісною об'єктною моделлю даних і фізичною моделлю даних, наприклад, якщо ваші сутнісні об'єкти конструюються з декількох таблиць або вам потрібна велика гнучкість в моделюванні ваших сутнісних об'єктів потрібно використовувати LINQ to Entities.



Мал 2.3 Структура . NET Framework

**2.6 СУБД Microsoft SQL Server**

Microsoft SQL Server — комерційна [система керування базами даних](http://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85), що розповсюджується корпорацією [Microsoft](http://uk.wikipedia.org/wiki/Microsoft). Мова, що використовується для запитів — [Transact-SQL](http://uk.wikipedia.org/w/index.php?title=Transact-SQL&action=edit&redlink=1), створена спільно Microsoft та [Sybase](http://uk.wikipedia.org/wiki/Sybase). Transact-SQL є реалізацією стандарту [ANSI](http://uk.wikipedia.org/wiki/ANSI)/[ISO](http://uk.wikipedia.org/wiki/ISO) щодо структурованої мови запитів ([SQL](http://uk.wikipedia.org/wiki/SQL)) із розширеннями. Використовується як для невеликих і середніх за розміром баз даних, так і для великих баз даних масштабу підприємства. Багато років вдало конкурує з іншими системами керування базами даних.

|  |
| --- |
|  |

Базовий код MS SQL Server (до версії 7.0) ґрунтувався на коді [Sybase SQL Server](http://uk.wikipedia.org/w/index.php?title=Sybase_SQL_Server&action=edit&redlink=1). Це дозволило [Microsoft](http://uk.wikipedia.org/wiki/Microsoft) вийти на ринок баз даних для підприємств, де конкурували [Oracle](http://uk.wikipedia.org/wiki/Oracle_Corporation), [IBM](http://uk.wikipedia.org/wiki/IBM), і, пізніше, сама Sybase. Microsoft, Sybase і [Ashton-Tate](http://uk.wikipedia.org/w/index.php?title=Ashton-Tate&action=edit&redlink=1) спочатку об'єдналися для створення і випуску на ринок першої версії програми, що отримала назву SQL Server 1.0 для [OS/2](http://uk.wikipedia.org/wiki/OS/2) (близько [1989 року](http://uk.wikipedia.org/wiki/1989)), яка фактично була еквівалентом Sybase SQL Server 3.0 для [Unix](http://uk.wikipedia.org/wiki/Unix), [VMS](http://uk.wikipedia.org/w/index.php?title=VMS&action=edit&redlink=1) та ін. Microsoft SQL Server 4.2 був випущений у [1992 році](http://uk.wikipedia.org/wiki/1992) та входив до складу операційної системи [Microsoft OS/2](http://uk.wikipedia.org/w/index.php?title=Microsoft_OS/2&action=edit&redlink=1) версії 1.3. Офіційний реліз Microsoft SQL Server версії 4.21 для ОС [Windows NT](http://uk.wikipedia.org/wiki/Windows_NT) відбувся одночасно з релізом самої [Windows NT](http://uk.wikipedia.org/wiki/Windows_NT) (версії 3.1). Microsoft SQL Server 6.0 був першою версією SQL Server, створеною виключно для архітектури NT і без участі в процесі розробки Sybase.

До того часу, як вийшла на ринок ОС Windows NT, Sybase і Microsoft розійшлися та створювали вже власні моделі цього програмного продукту. Microsoft намагалася отримати виняткові права на всі версії SQL Server для [Windows](http://uk.wikipedia.org/wiki/Microsoft_Windows). Пізніше Sybase змінила назву свого продукту на [Adaptive Server Enterprise](http://uk.wikipedia.org/w/index.php?title=Adaptive_Server_Enterprise&action=edit&redlink=1) щоб уникнути плутанини з Microsoft SQL Server. До [1994 року](http://uk.wikipedia.org/wiki/1994) Microsoft отримала від [Sybase](http://uk.wikipedia.org/wiki/Sybase) три повідомлення про авторські права як натяк на походження Microsoft SQL Server.

Після розділення компанії зробили декілька самостійних релізів програм. SQL Server 7.0 був першим сервером баз даних зі справжнім [графічним інтерфейсом](http://uk.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D1%96%D1%87%D0%BD%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D1%83%D0%B2%D0%B0%D1%87%D0%B0) адміністрування. Для усунення претензій з боку Sybase у порушенні авторських прав, весь успадкований код в сьомій версії був переписаний. Це забезпечило також й успіх SQL Server 2000, який був першою редакцією, орієнтованою на архітектуру [IA-64](http://uk.wikipedia.org/w/index.php?title=IA-64&action=edit&redlink=1).

Протягом подальших шести років [корпорація Microsoft](http://uk.wikipedia.org/wiki/Microsoft) працювала над вдосконаленням вже існуючої версії SQL Server 2000 доки не збудувала зручнішу систему Microsoft SQL Server 2005. Були вдосконалені продуктивність, кліентські інструменти [інтегрованого середовища розробки](http://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D1%81%D0%B5%D1%80%D0%B5%D0%B4%D0%BE%D0%B2%D0%B8%D1%89%D0%B5_%D1%80%D0%BE%D0%B7%D1%80%D0%BE%D0%B1%D0%BA%D0%B8), а також у декількох додаткових системах, що встановлюються разом із SQL Server 2005. Змінено: інструментарій [процесів керування сховищами даних](http://uk.wikipedia.org/wiki/ETL) (SQL Server Integration Services або [SSIS](http://uk.wikipedia.org/wiki/SQL_Server_Integration_Services)), сервер звітів, сервер [OLAP](http://uk.wikipedia.org/wiki/OLAP) та [інтелекутального аналізу даних](http://uk.wikipedia.org/w/index.php?title=%D0%86%D0%BD%D1%82%D0%B5%D0%BB%D0%B5%D0%BA%D1%83%D1%82%D0%B0%D0%BB%D1%8C%D0%BD%D0%B8%D0%B9_%D0%B0%D0%BD%D0%B0%D0%BB%D1%96%D0%B7_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85&action=edit&redlink=1) ([Analysis Services](http://uk.wikipedia.org/w/index.php?title=Microsoft_Analysis_Services&action=edit&redlink=1)), а також декілька технологій повідомлень, особливо Service Broker та Notification Services.

|  |  |  |  |
| --- | --- | --- | --- |
| Версії SQL Server | | | |
| **Версія** | **Рік** | **Повна назва** | **Кодове ім'я** |
| 1.0 ([OS/2](http://uk.wikipedia.org/wiki/OS/2)) | 1989 | SQL Server 1.0 | — |
| 4.2 | 1992 | SQL Server 4.2 | — |
| 4.21 ([WinNT](http://uk.wikipedia.org/wiki/Windows_NT)) | 1993 | SQL Server 4.21 | — |
| 6.0 | 1995 | SQL Server 6.0 | SQL95 |
| 6.5 | 1996 | SQL Server 6.5 | Hydra |
| 7.0 | 1998 | SQL Server 7.0 | Sphinx |
| — | 1999 | SQL Server 7.0 [OLAP](http://uk.wikipedia.org/wiki/OLAP) | Plato |
| 8.0 | 2000 | SQL Server 2000 32-бітна версія | Shiloh |
| 8.0 | 2003 | SQL Server 2000 64-бітна версія | Liberty |
| 9.0 | 2005 | SQL Server 2005 | Yukon |
| 10.0 | 2008 | SQL Server 2008 | Katmai |

SQL Server 2005, випущений в листопаді 2005, є наступником SQL Server 2000. На додаток до системи керування [реляційними базами даними](http://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D1%96_%D0%B1%D0%B0%D0%B7%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) включає також систему керування даними [XML](http://uk.wikipedia.org/wiki/XML). Для цього було визначено тип даних xml, який може використовуватись або як тип даних у стовпцях таблиць бази дани, або як літерал у запитах. XML-стовпці можуть бути асоційовані з схемами [XSD](http://uk.wikipedia.org/w/index.php?title=XSD&action=edit&redlink=1) (збережені дані XML перевіряються схемами). Перед збереженням у базі даних XML перетворюється на двійковий тип даних. Були розроблені спеціальні індексуючі методи для даних XML. Дані XML запитуються з використнням [XQuery](http://uk.wikipedia.org/wiki/XQuery) (до SQL Server 2005 доданий деякі розширення до мови [T-SQL](http://uk.wikipedia.org/w/index.php?title=T-SQL&action=edit&redlink=1), що дозволяють вкладення запитів XQuery до T-SQL). Крім того, були визначені нові розширення XQuery, названі XML DML, які дозволяють робити з даними XML модифікації на основі запитів. SQL Server 2005 також дозволяє серверу бази даних бути оприлюдненим через [веб-сервіси](http://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%BB%D1%83%D0%B6%D0%B1%D0%B0) з використанням пакетів TDS, що приховані у запитах [SOAP](http://uk.wikipedia.org/wiki/SOAP). Стосовно реляційних даних, до [T-SQL](http://uk.wikipedia.org/w/index.php?title=T-SQL&action=edit&redlink=1) були додані властивості керування помилками та підтримка рекурсивних запитів. SQL Server 2005 також включає нові алгоритми індексування та покращену систему відновлення після помилок. Сторінки даних стали містити [контрольну суму](http://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8C%D0%BD%D0%B0_%D1%81%D1%83%D0%BC%D0%B0) для кращого відновлення після помилок, також була додана підтримка оптимістичного паралелізму. Контроль дозволів і доступу був зроблений детальнішим, а процесор запитів став керувати паралельним виконанням запитів у ефективніший спосіб. Природно, підтримується поділ на таблиці та індекси, тому масштабування бази даних на [кластери](http://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%82%D0%B5%D1%80_%28%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0%29) відбувається легше. До SQL Server 2005 було введене CLR SQL, що дозволило йому об'єднатися з [.NET Framework](http://uk.wikipedia.org/wiki/.NET_Framework).

Наступна версія SQL Server — SQL Server 2008, кодова назва «Katmai» була представлена 6 серпня 2008 року. Мета випуску SQL Server 2008 — зробити керування даними самоналаштовуваним, самоорганізованим та самопідтримуваним. SQL Server 2008 також включає підтримку структурованих і напівструктурованих даних, у тому числі цифрові медіа-формати для зображень, звуків, відео й інших мультимедійних даних. Ключовим нововведенням SQL Server 2008 є розвинені засоби управління ресурсами (resource governor), що дозволяють ефективно управляти і розподіляти робоче навантаження за допомогою відстежування рівня завантаження процесора і обсягу пам'яті, що займають працюючі застосунки. Microsoft виділяє засоби управління на основі політик, розширені можливості з складання звітів і проведенню аналізу, а також розвинені засоби управління інтелектуальними ресурсами підприємства. У продукті з'явилася повноцінна підтримка неструктурованих даних і покращена система шифрування інформації. Крім того, варто виділити розширені функції роботи з геолокалізованими даними.

Серед нових можливостей і удосконалень Microsoft SQL Server 2008 також слід зазначити появу нових типів даних, а саме — просторових даних, кращу сумісність з застосунками сторонніх розробників, наприклад [Oracle](http://uk.wikipedia.org/wiki/Oracle_Database), тіснішу інтеграцію з [Office](http://uk.wikipedia.org/wiki/Microsoft_Office), оптимізовані засоби шифрування даних, засоби управління на основі політик, а також покращені інструменти звітності і аналізу.

Продукт SQL Server 2008 пропонується в семи модифікаціях, орієнтованих на різні категорії користувачів, зокрема, на робочі групи, крупних корпоративних замовників, розробників і інших. Ціни на SQL Server 2008 в порівнянні з попередніми версіями системи управління базами даних залишаться тим самим. Крім того, Microsoft пропонує користувачам безкоштовні версії продукту (з обмеженими можливостями) — SQL Server 2008 Express і SQL Server Compact.

SQL Server 2012 включає низку вдосконалень для роботи з критичними бізнес-[застосунками](http://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA) і бізнес-аналітикою як в традиційних [дата-центрах](http://uk.wikipedia.org/wiki/%D0%94%D0%B0%D1%82%D0%B0-%D1%86%D0%B5%D0%BD%D1%82%D1%80), так і в приватних, публічних і гібридних [хмарних середовищах](http://uk.wikipedia.org/wiki/%D0%A5%D0%BC%D0%B0%D1%80%D0%BD%D1%96_%D0%BE%D0%B1%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%BD%D1%8F). Серед нових можливостей SQL Server 2012 виділяються SQL Server AlwaysOn (рішення підтримки високого рівня доступності даних та аварійного відновлення), xVelocity (технологія збільшення продуктивності сховищ даних та програм бізнес-аналітики), нові рішення в області візуалізації PowerPivot і PowerView для створення звітів і аналітичних програм з [Excel](http://uk.wikipedia.org/wiki/Excel) і [SharePoint](http://uk.wikipedia.org/wiki/SharePoint), покращені інструменти для інтеграції даних і управління ними, включаючи SQL Server Data Quality Services і Master Data Services, нова семантична модель бізнес-аналітики та інструмент для адміністраторів баз даних і розробників застосунків SQL Server Data Tools.

Також Microsoft зробила значні інвестиції в області Big Data, а саме в інтеграцію SQL Server і популярних інструментів для бізнес-аналітики з неструктурованою інформацією.

Microsoft робить SQL Server доступним у різноманітних варіантах, які різняться наборами властивостей в залежності від цілей кінцевого користувача. Це такі редакції як:

[SQL Server Compact Edition (SQL CE)](http://uk.wikipedia.org/w/index.php?title=SQL_Server_Compact&action=edit&redlink=1)

Компактне видання — вкладений механізм бази даних. Завдяки малому обсягу (2 Мб для DLL) має зменшені властивості у порівнянні з іншими варіантами. Розмір бази даних обмежений 4 Гб і не може використовуватися як служба Windows.

[SQL Server Express Edition](http://uk.wikipedia.org/w/index.php?title=SQL_Server_Express&action=edit&redlink=1)

Раніше відомий під назвою MSDE (Microsoft SQL Server Desktop Engine), Microsoft SQL Server Express є вільно поширюваною версією SQL Server. Дана версія має деякі технічні обмеження, також відсутні графічні інструменти адміністрування. Такі обмеження роблять її непридатною для розгортання великих баз даних. В основному вона використовується у застосунках, при проектуванні, або для самостійного вивчення. Розмір бази даних обмежений 4 Гб, розмір пам'яті, що може бути адресованою — 1 Гб, підтримує лише один процесор.

SQL Server Workgroup Edition включає функціональність ядра бази даних, але не включає додаткові сервіси.

SQL Server Standard edition включає механізми ядра бази даних, а також автономні сервіси. Відрізняється від варіанту Enterprise Edition тим, що підтримує менше активних вузлів та не включає деякі функції збільшення продуктивності.

SQL Server Enterprise Edition — це повнофункціональна версія SQL Server

SQL Server Developer Edition включає ті самі функції, що й SQL Server Enterprise Edition, але містить обмеження щодо використання його лише для розробки та тестування. Його ліцензія не дозволяє використання в якості виробничого серверу.

Microsoft SQL Server як мову запитів використовує версію [SQL](http://uk.wikipedia.org/wiki/SQL), що отримала назву [TRANSACT-SQL](http://uk.wikipedia.org/w/index.php?title=TRANSACT-SQL&action=edit&redlink=1) (скорочено T-SQL), яка є реалізацією SQL-92 (стандарт [ISO](http://uk.wikipedia.org/wiki/ISO) для SQL) з багатьма розширеннями. T-SQL дозволяє використовувати додатковий синтаксис [процедур, що зберігаються](http://uk.wikipedia.org/w/index.php?title=%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D0%B4%D1%83%D1%80%D0%B0,_%D1%89%D0%BE_%D0%B7%D0%B1%D0%B5%D1%80%D1%96%D0%B3%D0%B0%D1%94%D1%82%D1%8C%D1%81%D1%8F&action=edit&redlink=1) і забезпечує підтримку транзакцій (взаємодія бази даних з керуючим застосунком). Microsoft SQL Server та Sybase ASE для взаємодії з мережею використовують протокол рівня застосунка під назвою [Tabular Data Stream](http://uk.wikipedia.org/w/index.php?title=Tabular_Data_Stream&action=edit&redlink=1) (TDS, протокол передачі табличних даних).

Microsoft SQL Server також підтримує [Open Database Connectivity](http://uk.wikipedia.org/wiki/ODBC) (ODBC) — інтерфейс взаємодії застосунків з СУБД. Версія SQL Server 2005 надає можливість підключення користувачів через веб-сервер-сервіси, що використовують протокол SOAP. Це дозволяє клієнтським програмам, не призначеним для [Windows](http://uk.wikipedia.org/wiki/Windows), кроссплатформенно з'єднуватися з SQL Server. Microsoft також випустила сертифікований драйвер [JDBC](http://uk.wikipedia.org/wiki/JDBC), що дозволяє застосункам під керування [Java](http://uk.wikipedia.org/wiki/Java) (таким як BEA і [IBM Websphere](http://uk.wikipedia.org/wiki/IBM_Websphere)) з'єднуватися з Microsoft SQL Server 2000 і 2005.

SQL Server підтримує дзеркалювання та кластеризацію баз даних. Кластер серверу SQL — це сукупність однаково конфігурованих серверів; така схема допомагає розподілити робоче навантаження між декількома серверами. Усі сервера мають одне віртуальне ім'я, а дані розподіляються за IP-адресами машин кластеру протягом робочого циклу. Також у разі відмови або збою на одному з серверів кластеру доступне автоматичне перенесення навантаження на інший сервер.

SQL Server підтримує надлишкове дублювання даних за трьома сценаріями:

* Знімок: Виконується «знімок» бази даних, який сервер відправляє одержувачам.
* Історія змін: Всі зміни бази даних безперервно передаються користувачам.
* [Синхронізація](http://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%85%D1%80%D0%BE%D0%BD%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) з іншими серверами: Бази даних декількох серверів синхронізуються між собою. Зміни усіх баз даних відбуваються незалежно на кожному сервері, а під час синхронізації відбувається звірка даних. Дублювання такого типу передбачає можливість вирішення протиріч між базами даних.

SQL Server 2005 має вбудовану підтримку .NET Framework. Завдяки цьому, процедури бази даних, що зберігаються, можуть бути написані на будь-якій мові платформи .NET з використанням повного набору бібліотек, доступних для .NET Framework. На відміну від інших процесів, .NET Framework виділяє додаткову пам'ять і будує засоби керування SQL Server, не використовуючи вбудовані засоби Windows. Це підвищує продуктивність порівняно із загальними алгоритмами Windows, оскільки алгоритми розподілу ресурсів спеціально налагоджені для використання у структурах SQL Server.

Microsoft та інші компанії пропонують велику кількість програмних засобів розробки, які дозволяють розробляти [застосунки](http://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BA%D0%B8) для бізнесу з використанням баз даних Microsoft SQL Server. Microsoft SQL Server 2005 включає також [Common Language Runtime](http://uk.wikipedia.org/wiki/Common_Language_Runtime) (CLR) [Microsoft .NET](http://uk.wikipedia.org/wiki/Microsoft_.NET), що дозволяє застосункам, розробленим на мовах платформи [.ΝΕΤ](http://uk.wikipedia.org/w/index.php?title=.%CE%9D%CE%95%CE%A4&action=edit&redlink=1) (наприклад, [VB.NET](http://uk.wikipedia.org/wiki/VB.NET) або [C#](http://uk.wikipedia.org/wiki/C_Sharp)), реалізовувати [процедури, що зберігаються](http://uk.wikipedia.org/w/index.php?title=%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D0%B4%D1%83%D1%80%D0%B0,_%D1%89%D0%BE_%D0%B7%D0%B1%D0%B5%D1%80%D1%96%D0%B3%D0%B0%D1%94%D1%82%D1%8C%D1%81%D1%8F&action=edit&redlink=1) та різні функції. Попередні версії засобів розробки Microsoft використовували лише [API](http://uk.wikipedia.org/wiki/API) для надання функціонального доступу до Microsoft SQL Server.

**2.7 Вбудована мова запитів LINQ**

Language Integrated Query (LINQ) - проект Microsoft по додаванню синтаксису мови запитів, що нагадує SQL, в мови програмування платформи .NET Framework. Являє собою не що інше, як функціональне програмування, замасковане під синтаксис SQL. Раніше був реалізований в мовах C # і Visual Basic. NET. Безліч концепцій, які вводить LINQ, спочатку випробували в дослідницькому проекті Microsoft Cω.  
LINQ випущений разом з Visual Studio 2008 в кінці листопада 2007 року.  
  
Використовуючи деякі нові особливості мови, LINQ дозволяє використовувати SQL-подібний синтаксис безпосередньо в коді програми, написаної, наприклад, на мові C #:

* Анонімні типи
* Методи розширення
* Лямбда-числення
* Дерево виразів
* Стандартні оператори мови запитів
* Джерела даних

Спочатку підтримуючи механізм запитів для колекцій об'єктів в пам'яті, реляційних баз даних і даних у форматі XML, LINQ володіє розширюваною архітектурою, яка дозволяє стороннім розробникам реалізувати доступ до їх сховищ даних через механізм LINQ. Для цього необхідно реалізувати стандартні оператори запитів, використовуючи методи розширення, або реалізувати інтерфейс IQueryable, що дозволяє розбирати дерево виразу під час виконання, транслюючи його в свою мову запитів. У спільноті існує приклад користувальницької реалізації стандартних операторів запитів.   
Наприклад, LINQ для SQL (колишній DLinq), який перетворює LINQ-вирази в SQL-запити до бази даних, використовує можливості компілятора для побудови дерева виразів, грунтуючись на контексті програми, а не створюючи делегати функцій. Отримавши дерево виразів, що описує запит, спеціалізований провайдер бази даних може його проаналізувати і перетворити в запит на відповідній мові для бази даних, наприклад Microsoft SQL Server, Jet (яка використовується в Microsoft Access) або будь-якій інший. Деякі ентузіасти за допомогою подібної тактики вже створили для перевірки концепції LINQ бібліотеки для запитів до WMI , RSS, LDAP , колекціям даних ADO.NET, Amazon Web Services і SharePoint .

Існуюча попередня версія від Microsoft також включає в себе реалізацію LINQ для XML (раніше називалася XLinq), яка значно спрощує побудову XML документа і вилучення даних з нього, використовуючи схожі підходи. Крім того, Microsoft працює над ADO.NET vNext, також відомим як LINQ to Entities.  
В кінці 2008 року відповідальність за розробку LINQ до SQL поряд з ADO.NET Entity Framework (у тому числі і LINQ до Entities) була перекладена на команду, що займалася розвитком ADO.NET (т. зв. ADO.NET team), тоді як раніше розвитком LINQ до SQL займалася команда, пов'язана з розробкою компілятора для мови C # . Таким чином, стало очевидно, що обидва рішення націлені на вирішення одних і тих же завдань, а отже будуть конкурувати один з одним. Трохи пізніше Тім Маллалью роз'яснив, що Microsoft продовжить розробку LINQ до SQL на основі користувацьких відгуків. Однак, починаючи з версії платформи. NET 4.0, рекомендованим рішенням стає саме LINQ до Entities. Крім того, на підставі інформації, отриманої від користувачів, найбільш вживані можливості LINQ до SQL будуть додані і в LINQ до Entities. У результаті чого відбудеться поступове злиття рішень.  
Експерти в основному підтримали дане рішення. Так, наприклад, Марко Руссо хоча і обмовився, що переходити до LINQ до Entities варто не раніше, ніж воно перетвориться на повноцінну заміну LINQ до SQL, проте заявив, що об'єднання двох фреймворків, що частково перекривають один одного - хороша ідея .  
SQLMetal  
Бібліотека LINQ включає в себе інструмент SQLMetal, який дозволяє автоматично генерувати класи безпосередньо з підтримуваних .NET Framework баз даних. Альтернативою є реляційний конструктор об'єктів, що входить до складу Visual Studio, проте він може бути використаний тільки разом з Microsoft SQL Server.

**3 РОЗРОБКА СЕРВЕРНОЇ ЧАСТИНИ ПРОГРАМНОГО ШАБЛОНУ ТИПОВОГО РЕСУРСУ ЕЛЕКТРОННОЇ КОМЕРЦІЇ ЗАСОБАМИ ASP.NET**

**3.1 Розробка архітектури проекту**

Для того щоби створити досить складну програмну систему спочатку потрібно спланувати її архітектуру. Це позволить пришвидшити і спростити процес розробки а також уникнути багаторазових докорінних переробок, адже саме помилки в архітектурі програмного комплексу вимагають найбільше часу і зусиль на їх виправлення. В даній програмі я використовуватиму шаблон МВЦ і це вже накладає певний відбидок на її будову. Головною вимогою до архітектури даного проекту було чітке відокремлення бізнес – логіки від інтерфейсу. Однією з причин важливості даної вимоги є те, що розроблювана мною система є тільки складовою частиною цілісного програмного продукту, і розробкою користувацького інтерфесу займається інший програміст. Суть відокремлення бізнес- логіки від інтерфейсу закладена в самій концепції МВЦ, проте в даному випадку особливості роботи системи управління контентом і досить великий розмір частини програми, яка займається технічними операціями з даними і майже ніяк не повязана з інтерфейсом наштовхують на дещо складніше рішення. В результаті аналізу вимог до створюваного програмного продукту та приблизної схеми його використання була розроблена наступна архітектурна схема:

СКВ

ЯДРО

САЙТ

ОСНОВНІ КЛАСИ

МОДУЛЬ ВЗАЄМОДІЇ З БАЗОЮ ДАНИХ

БАЗОВІ СЕРВІСИ

АДМІНІСТРАТИВНА ЧАСТИНА

КЛІЄНТСЬКА ЧАСТИНА

Мал 3.1 Структура розроблюваної СКВ

Як видно з даної схеми, розроблювана СКВ розділяється на дві слабко взаємозалежні базові частини : ядро і безпосередньо сам шаблон сайту електронної комерції. Це надає можливість в подальшому розширювати ці модулі незалежно один від одного.

Адміністративна і клієнтська частини виконані за класичною схемою МВЦ – вони складаються з представлень, контроллерів, моделей і допоміжних класів.

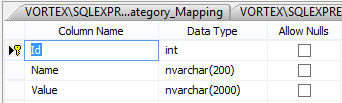
Поскільки я займаюся розробкою серверної частини, то розробкою будови і безпосередньою версткою представлень я не займаюся, мені потрібно створити контроллери і моделі, які повинні оперувати представленнями і надавати необхідні для їх роботи дані. Принципи роботи МВЦ додатку я вже описував в розділі 2.1, тому подробно на них я зупинятись не буду, єдиною відмінністю від простої схеми МВЦ є те, що вданому випадку дані отримуються не з бази даних через моделі, а з ядра.

Спрощена діаграма класів кожного з розділів ядра і сайту подана в додатках:

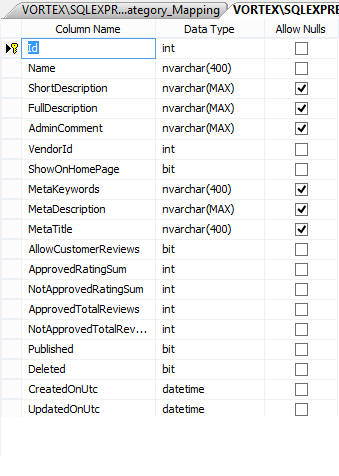
1. Основні класи ядра – додаток 1
2. Модуль взаємодії з базою даних – додаток 2
3. Базові сервіси – додаток 3
4. Адміністративна частина сайту – додаток 4
5. Клієнтська частина сайту – додаток 5

В цих діаграмах подані основні класи та взаємозвязки між ними, що дає змогу судити про більш деталізовану структуру створюваної системи.

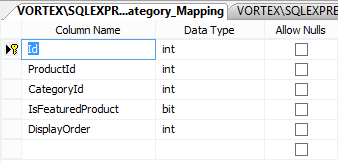
Структура бази даних повинна містити інформацію про сутності, які використовуються в системі, як от «Setting», «Product», «Order», які будуть детальніше розглянуті в розділі 3, а тому майже повністю повторює структуру основних класів, але до неї також додані службові і перехресні таблиці, котрі відповідають за взаємовідношення окремих сутностей між собою. Приклади таблиць, які зберігають дані про сутності:



Мал 3.2 Таблиця Setting



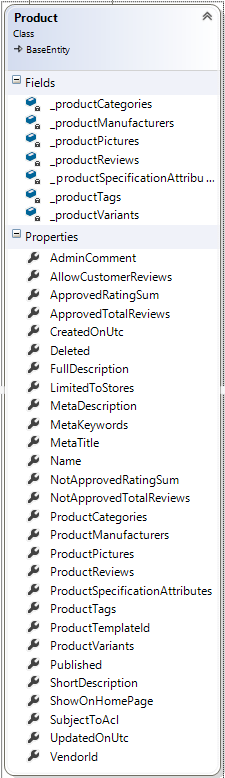
Мал 3.3 Таблиця Product



Мал 3.4 Перехресна таблиця, яка повязує продукт і категорію

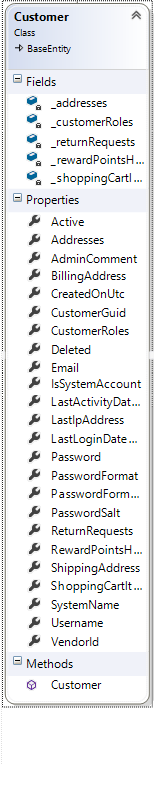
**3.2 Огляд основних класів і сутностей в проекті**

Як я вже згадуваав в розділі 2.8, класи, що описують базові сутності в СКВ, знаходяться в модулі ядра, в проекті «Основні класи». Згідно з концепцією ООП, саме екземпляри цих класів представляють окремі товари (послуги), користувачів, статті в блогах тощо. Одним з найголовніших понять в любому ресурсі електронної комерції є поняття продукту. В нашому випадку він описується класом «Product». Продукт місить всі базові властивості деякого узагальненого продукту, виражені як поля чи властивості класу, котрі використовуються для його ідентифікації, опису та здійснення з ним різноманітних операцій. Крім цього, даний клас містить деякі службові поля, які використовуються самою СКВ для відображення, добавлення даного продукту до замовлень та ін., такі як «Published», яке визначає чи опублікований даний продукт, чи ні (використовується контроллером, відповідальним за передачу продуктів через модель у представлення). Якщо продукт помічений як неопублікований, контроллер відсіє його в момент фільтрації результатів запиту. Крім того, продукт містить поля для його опису (інформація, яка міститься в цих полях, відображається користувачеві на сторінці інформації про продукт), це такі поля як «ShortDescription», «FullDescription» а також поля метаопису продукту, який використовується для просування сайту в пошукових системах - «MetaDescription», «MetaKeywords», «MetaTitle». Ці поля містять інформацію, котра є невидимою для користувача, вона розрахована спеціально для роботів пошукових систем, таких як Google Bot. При наявності цієї інформації якщо користувач шукатиме даний товар в пошуковій системі, вона видасть посилання на цей товар на вашому сайті вище у результатах, ніж посилання на сайти конкурентів.



Мал 3.5 Будова класу «Product»

Іншою базовою сутністю в любій комерції – як звичайній, так і електронній, є поняття покупця. В даній СКВ воно описується класом «Customer». Даний клас містить всю необхідну інформацію про покупця. В електронній комерції зазвичай здійснювати замовлення, а в багатьох випадках, наприклад на сайтах оптових складів чи виробників, і переглядати детальну інформацію про товари можуть тільки зареєстровані користувачі. Тому в даний клас були введені поля «Username», яке відповідає за імя користувача, поля, які відповідають за вхід користувача на сайті - «Email»,«Password» і службові поля «PasswordFormat» та «PasswordSalt», які служать для безпечного зберігання пароля.



Мал 3.6 Будова класу «Customer»

Також було введене поле «Active», яке може використовуватися для підтвердження аккауну терез електронну пошту і поля, які можуть бути використані для відстеження активності користувача на сайті - «LastIpAdress», «LastActivityDate» і «LastLoginDate»

Третім «китом» серед понять, на яких тримається електронна комерція, є поняття замовлення. В даному програмному продукті воно описується класом «Order», який містить всю необхідну інформацію про замовлення – дату замовлення, його суму, клієнта, який здійснив дане замовлення, товари в ньому і т.д.

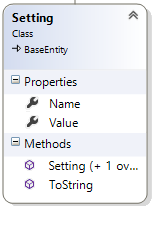
За дату замовлення відповідає поле «CreatedOnUtc», а за дату оплати - «PaidDateUtc». Дане поле разом з полем «PaymentStatus» можна використовувати для того, щоб відправляти товар замовнику тільки після оплати. Поля «Customer», «CustomerIp» та «CustomerId» надають короткий опис покупця. Детальнішу інформацію про нього можна отримати якщо зробити запит з ідентифікатором покупця до таблиці користувачів сайту. Поля «OrderTotal», яке описує загальну суму замовлення, «OrderTax» яке зберігає суму податку, «OrderDiscount», яке зберігає суму знижки та «CurrencyRate» яке може зберігати відношення поточної валюти до долара у випадку введення на ресурсі багатовалютності описують фінансову частину замовлення, а поля «OrderStatus», яке містить коротку інформацію про статус поточного замовлення «OrderStatusId», яке місить ідентифікатор статусу замовлення, «Deleted», яке служить маркером видалення (відміни) замовлення і «Shipment», яке зберігає інформацію про доставку служать для відстеження і збереження стану даного замовлення.

Крім описаних полів, даний клас також містить багато службових полів, які служать для пришвидшення обробки запитів, повязаних з полями, надання додаткової інформації, полегшення обліку та ін. Така повнота опису основних сутностей полегшує роботу з СКВ недосвідченим користувачам і надає можливості для подальшого розширення і вдосконалення системи без суттєвої переробки джерела даних та механізмів роботи з ним.



Мал 3.7 Будова класу «Order»

Крім базових класів, які описують основні поняття предметної області в системі керуванням вмістом є багато інших класів, не менш важлиих для її повноцінного функціонування. До таких класів належить наприклад клас «Setting», який являє собою описану через клас сутність одиночного налаштування. Незважаючи на свою простоту, роль даного класу в системі важко недооцінити, адже саме у вигляді екземплярів даного класу зберігаються і опрацьовуються всі настройки системи управління контентом.



Мал 3.8 Будова класу «Setting»

В даному продукті реалізована також система локалізації. Базовим класом в ній є клас «LocaleStringResource», який зберігає локалізовані значення змінних. Локалізація організована таким чином, що тексти на кнопках, гперпосиланнях та інших елементах інтерфейсу задані не напряму в HTML коді сторінки, а у вигляді змінних, значення яких підставляються серверною частиною в залежності від вибраної мови. Так наприклад виглядає код посилання на головну сторінку

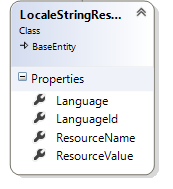
<li><a href="@Url.RouteUrl("HomePage")">@T("HomePage")</a></li>

, де @T("HomePage") – підстановка значення локалізованої змінної.

У випадку, якщо потрібне значення не було знайдене, виводиться назва змінної.

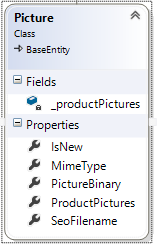
Клас «LocaleStringResource», містить необхідні для такого механізму поля:

1. «Language» і «LanguageId» - зберігають значення та ідентифікатор якій належить даний переклад
2. «ResourceName» -зберігає назву ресурсу(ім’я змінної)
3. «ResourceValue» - зберігає переклад змінної



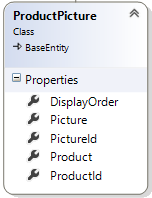
Мал 3.9 Будова класу «LocaleStringResource»

За зображення продуктів, які відображаються в магазині, відповідає клас «Picture». Він містить поля, які потрібні для опису картинки продукту (або всіх відповідних даному продукту зображень, якщо їх декілька). Цей клас також підтримує SEO – оптимізацію, завдяки наявності поля «SeoFilename». Для полегшення управління зображеннями і сприйняття інформації про зображення адміністратором ресурсу було також введене поле «IsNew», яке визначає чи є це зображення недавно доданим.



Мал 3.10 Будова класу «Picture»

Для повязування продукту з відповідними йому зображеннями використовується клас «ProductPicture». Це аналог перехресної таблиці в базі даних, містить поля які позволяють сумістити продукт і зображення - зберігаючи ідентифікатор зображення, ідентифікатор продукту та самі екземпляри цих класів.

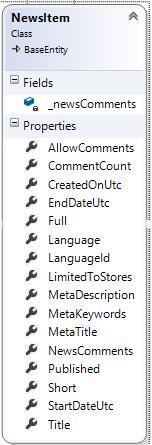


Мал 3.11 Будова класу «ProductPicture»

Крім функціоналу, який відповідає безпосередньо за здійснення покупок, кожен серйозний інтернет – магазин чи корпоративний сайт повинен містити розділи новин і блог, де він зміг би розміщувати різноманітні новини, статті, оголошувати акції і т.д.



Мал 3.12 Будова класу «BlogPost»



Мал 3.13 Будова класу «NewsItem»

Крім того, що ці статті чи пости роблять сайт значно цікавішим для користувача та інформують його про головні події в житті компанії чи ресурсу електронної комерції, регулярне обновлення і додавання новин служить показником активності сайту - особисто я, якщо на сайті в новинах чи блозі останнє повідомлення було більш ніж рік тому, вважаю такий сайт «мертвим». Іще однією перевагою наявності на сайті блогу чи стрічки новин є просування в пошукових системах – статті з блогу чи новини (якщо вони не скопійовані з якогось іншого ресурсу) розцінюються пошуковими системами як унікальний контент і піднімають рейтинг сайту.

Для опису сутностей «Новина» і «Повідомлення в блозі» використовуються класи «NewsItem» і «BlogPost» відповідно. ці класи досить схожі між собою за своєю структурою і призначенням, так як схожі і відповідні їм сутності з реального світу. Кожен з них містить поля, котрі зберігають дані про статтю чи пост. За тіло статті чи поста відповідають поля «Full» і «Body» відповідно. Для полегшення адміністрування та введення можливості коментування новин і постів були введені поля, які зберігають коментарі, їх кількість, дату створення та знищення. Для того щоб ввести можливість відображення скороченої версії новини, яка місить стисле подання основної інформації в статті було введене поле «Short». Крім того, дані класи також підтримують локалізацію (щоб користувач ресурсу, переглядаючи блог чи стрічку новин бачив тільки новини і пости на вибраній ним мові) завдяки полям «Language» і «LanguageId», які зберігають дані про мову статті і таким чином формують привязку статті до конкретної мови та SEO оптимізацію – завдяки полям «MetaDescription», «MetaKeywords», «MetaTitle», які зберігають ключові слова (метатеги), метаопис та метазаголовки для кожної статті чи посту.

**3.3 Принцип дії шаблону ресурсу електронної комерції**

Принцип дії шаблону електронного ресурсу не є дуже складним і найраще описується за допомогою реального прикладу. Розглянемо такий випадок: користувач хоче перейти в магазин через головне меню сайту. Спочатку він натискає на посилання в меню, сформоване таким куском коду :

<li> <a href="@Url.RouteUrl("Shop")">@T("Shop")</a></li>

Фреймворк опрацьовує натискання і переходить за посиланням, вказаним у вигляді переходу за маршрутом. Він шукає у списку маршрутів в файлі «RouteProvider.cs» маршрут "ShopPage", описаний таким чином:

routes.MapLocalizedRoute("Shop",

"shop/",

new { controller = "Catalog", action = "Shop" },

new[] { "Nop.Web.Controllers" });

Даний опис містить назву контроллера, який повинен опрацювати даний запит, назву дії (методу) даного контроллера, який повинен бути викликаний при цьому і кусок адреси, який повинен бути вставлений в адресний рядок. Того ж самого результату можна добитися, якщо скористатися стандартним підходом маршрутизації ASP.NET MVC 3(4), який передбачає формування посилань наступним чином

<http://mysite.com/ControllerName/ActionName?parameter1=...&parameter2=....&>.......

При введенні посилання такого типу в адресний рядок і переході по ньому буде викликаний метод «ActionName» контроллера «ControllerName» з передачею йому параметрів «parameter1» і «parameter2».

В нашому випадку посилання, сформоване за таким підходом, виглядатиме так :

<http://mysite.com/Catalog/Shop>

А посилання, сформоване через виклик заздалегідь визначеного маршруту, виглядатиме так:

http://mysite.com/shop

Як бачимо використання визначених маршрутів дає значно коротший і зрозуміліший результат. проти ці два підходи до маршрутизації не є взаємовиключаючими і можуть успішно використовуватись паралельно, залежно від того, який з них зручніший для конкретної сторінки і конкретної ситуації. Незалежно від методу маршрутизації, в результаті буде викликаний метод «Shop» контроллера «Catalog»:

public ActionResult Shop( CatalogPagingFilteringModel command)

{

Тут на основі поточної адреси формується посилання «продовжити покупки»

\_genericAttributeService.SaveAttribute(\_workContext.CurrentCustomer, SystemCustomerAttributeNames.LastContinueShoppingPage,

\_webHelper.GetThisPageUrl(false),

\_storeContext.CurrentStore.Id);

if (command.PageNumber <= 0) command.PageNumber = 1;

Створюється новий екземпляр класу «ShopModel», починається формування моделі

var model = new ShopModel();

Проводиться отримання параметру сортування з поточної адреси:

model.PagingFilteringContext.AllowProductSorting = \_catalogSettings.AllowProductSorting;

if (model.PagingFilteringContext.AllowProductSorting)

{

foreach (ProductSortingEnum enumValue in Enum.GetValues(typeof(ProductSortingEnum)))

{

var currentPageUrl = \_webHelper.GetThisPageUrl(true);

var sortUrl = \_webHelper.ModifyQueryString(currentPageUrl, "orderby=" + ((int)enumValue).ToString(), null);

var sortValue = enumValue.GetLocalizedEnum(\_localizationService, \_workContext);

model.PagingFilteringContext.AvailableSortOptions.Add(new SelectListItem()

{

Text = sortValue,

Value = sortUrl,

Selected = enumValue == (ProductSortingEnum)command.OrderBy

});

}

}

Вибирається вигляд результатів – у вигляді «плитки» чи таблиці:

model.PagingFilteringContext.AllowProductViewModeChanging = \_catalogSettings.AllowProductViewModeChanging;

var viewMode = !string.IsNullOrEmpty(command.ViewMode)

? command.ViewMode

: \_catalogSettings.DefaultViewMode;

if (model.PagingFilteringContext.AllowProductViewModeChanging)

{

var currentPageUrl = \_webHelper.GetThisPageUrl(true);

model.PagingFilteringContext.AvailableViewModes.Add(new SelectListItem()

{

Text = \_localizationService.GetResource("Categories.ViewMode.Grid"),

Value = \_webHelper.ModifyQueryString(currentPageUrl, "viewmode=grid", null),

Selected = viewMode == "grid"

});

model.PagingFilteringContext.AvailableViewModes.Add(new SelectListItem()

{

Text = \_localizationService.GetResource("Categories.ViewMode.List"),

Value = \_webHelper.ModifyQueryString(currentPageUrl, "viewmode=list", null),

Selected = viewMode == "list"

});

}

Вибираються кількості елементів на сторінці:

model.PagingFilteringContext.AllowCustomersToSelectPageSize = false;

var pageSizes = new string[] { "4", "8", "12","20","40","80" };

if (pageSizes.Any())

{

if (command.PageSize <= 0 || !pageSizes.Contains(command.PageSize.ToString()))

{

int temp = 0;

if (int.TryParse(pageSizes.FirstOrDefault(), out temp))

{

if (temp > 0)

{

command.PageSize = temp;

}

}

}

var currentPageUrl = \_webHelper.GetThisPageUrl(true);

var sortUrl = \_webHelper.ModifyQueryString(currentPageUrl, "pagesize={0}", null);

sortUrl = \_webHelper.RemoveQueryString(sortUrl, "pagenumber");

foreach (var pageSize in pageSizes)

{

int temp = 0;

if (!int.TryParse(pageSize, out temp))

{

continue;

}

if (temp <= 0)

{

continue;

}

model.PagingFilteringContext.PageSizeOptions.Add(new SelectListItem()

{

Text = pageSize,

Value = String.Format(sortUrl, pageSize),

Selected = pageSize.Equals(command.PageSize.ToString(), StringComparison.InvariantCultureIgnoreCase)

});

}

if (model.PagingFilteringContext.PageSizeOptions.Any())

{

model.PagingFilteringContext.PageSizeOptions = model.PagingFilteringContext.PageSizeOptions.OrderBy(x => int.Parse(x.Text)).ToList();

model.PagingFilteringContext.AllowCustomersToSelectPageSize = true;

if (command.PageSize <= 0)

{

command.PageSize = int.Parse(model.PagingFilteringContext.PageSizeOptions.FirstOrDefault().Text);

}

}

}

Отримуються і присвоюються діапазони цін:

if (command.PageSize <= 0) command.PageSize = 10;

decimal? minPriceConverted = null;

decimal? maxPriceConverted = null;

minPriceConverted = 0;

maxPriceConverted = 10000;

model.FeaturedProducts = new List<ProductOverviewModel>();

Відбувається вибірка продуктів з бази даних за вказаними параметрами

IList<int> alreadyFilteredSpecOptionIds = model.PagingFilteringContext.SpecificationFilter.GetAlreadyFilteredSpecOptionIds(\_webHelper);

IList<int> filterableSpecificationAttributeOptionIds = null;

var products = \_productService.SearchProducts(out filterableSpecificationAttributeOptionIds, true,

storeId: \_storeContext.CurrentStore.Id,

featuredProducts: true,

priceMin: minPriceConverted, priceMax: maxPriceConverted,

filteredSpecs: alreadyFilteredSpecOptionIds,

orderBy: (ProductSortingEnum)command.OrderBy,

pageIndex: command.PageNumber - 1,

pageSize: command.PageSize);

model.Products = PrepareProductOverviewModels(products).ToList();

model.PagingFilteringContext.LoadPagedList(products);

model.PagingFilteringContext.ViewMode = viewMode;

Повертається представлення, якому передається сформована вище модель. Поскільки представлення є строго типізованим, тобто прив’язане до конкретної моделі, то у випадку, коли повертається представлення, якому не передали перед тим модель, виникне помилка під час виконання. Якщо не передавати методу «View» назву представлення, то він шукатиме представлення з такою самою назвою як і в метода контроллера, звідки його викликали, у папці «Views» даного пректу. Поскільки наше представлення називається «Shop», тобто повністю відповідає назві методу контроллера, топередавати методу «View» назву потрібного представлення не потрібно

return View(model);

}

Оголошення строго типізованого представлення робиться дуже просто: достатньо вказати на початку коду представлення рядок типу

@model ModelName

В нашому випадку це

@model ShopModel

Код моделі, яка використовується в даному методі і відповідному йому представленні:

using System.Collections.Generic;

using Nop.Web.Framework.Mvc;

using Nop.Web.Models.Media;

namespace Nop.Web.Models.Catalog

{

public partial class ShopModel : BaseNopEntityModel

{

public ShopModel()

{

PictureModel = new PictureModel();

Products = new List<ProductOverviewModel>();

PagingFilteringContext = new CatalogPagingFilteringModel();

}

public string Name { get; set; }

public string Description { get; set; }

public string MetaKeywords { get; set; }

public string MetaDescription { get; set; }

public string MetaTitle { get; set; }

public string SeName { get; set; }

public PictureModel PictureModel { get; set; }

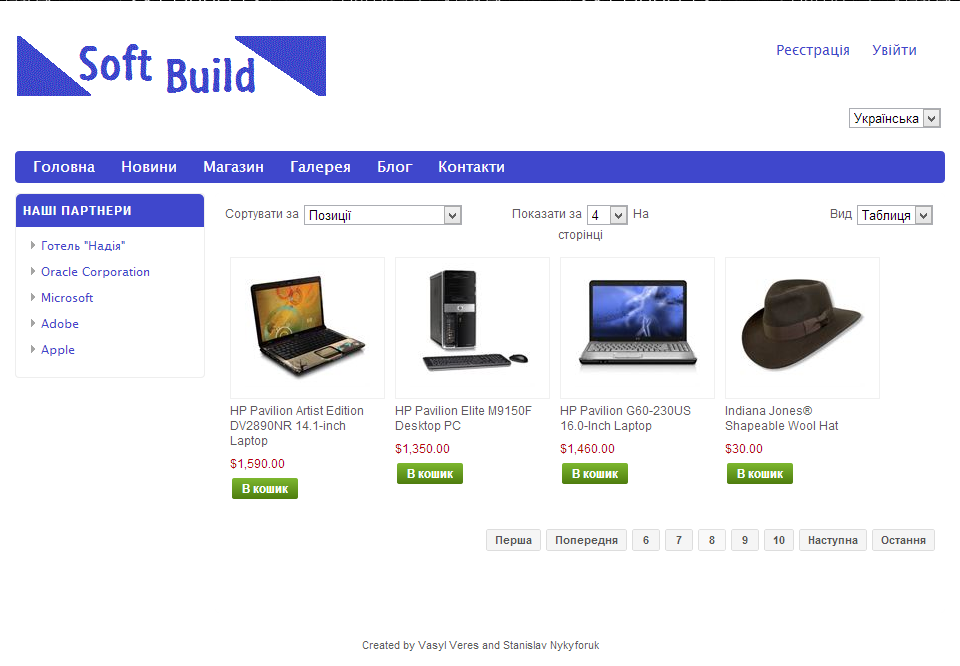
public CatalogPagingFilteringModel PagingFilteringContext { get; set; }

public IList<ProductOverviewModel> Products { get; set; }

}

}

В результаті виконання описаної вище послідовності операцій ми отримаємо такий результат:



Мал 3.14 Сторінка магазину

**4 ОХОРОНА ПРАЦІ**

**4.1 Законодавче та нормативно-правове забезпечення охорони праці**

ТзОВ «SoftJourn» – компанія, що займається розробкою та супроводом програмного забезпечення різних видів – від веб-застосунків до програм для різних мобільних платформ, таких як iOS, Android, Windows Phone 7, Windows Phone 8. Її офіс знаходиться за адресою вул. Тичини, 7, м.Івано-Франківськ.

Працівники на даному підприємстві майже увесь робочий день проводять працюючи з комп’ютером, тому воно повинне виконувати такі нормативно-правові акти з охорони праці:

* НПАОП 0.00-1.28-10 Правила охорони праці під час експлуатації електронно-обчислювальних машин;
* НПАОП 0.00-4.12-05 Типове положення про порядок проведення навчання і перевірки знань з питань охорони праці;
* НПАОП 0.00-4.15-98 Положення про розробку інструкцій з охорони праці;
* НПАОП 0.00-4.21-04 Типове положення про службу охорони праці;
* НПАОП 0.00-4.29-97 Типове положення про кабінет охорони праці;
* НПАОП 0.00-6.07-04 Порядок ведення обліку даних про технічний стан машин, механізмів, устаткування підвищеної небезпеки;
* НПАОП 0.00-6.23-92 Про порядок атестації робочих місць за умовами праці;
* НПАОП 0.00-7.06-94 Єдина державна система показників обліку умов та безпеки праці;

Крім того, для даного підприємства як і для всіх інших є обов`язковим дотримання Кодексу законів про працю та Закону України про охорону праці.

Вимоги безпеки при роботі з ПК:

В Україні діють державні санітарні правила та норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин від 10 грудня 1998 №7. У цих правилах, зокрема, регламентується, що приміщення, де працюють люди з ПЕОМ, повинно розміщуватися в північній або північно-східній частині будівлі. Площа одного робочого місця повинна становити не менше 6 квадратних метрів, об'єм – не менше 20 кубічних метрів, відстань між робочими столами – не менше 2,5 м у ряду і 1,2 м між рядами. Стіни приміщень повинні мати коефіцієнт відбиття 0,5-0,6.

Висота робочої поверхні столу має бути в межах 680-800 мм. Рекомендована ширина столу – 600-1400 мм, глибина – 800-1000 мм. Робочий стіл повинен мати простір для ніг висотою не менше 600 мм, шириною – не менше 500 мм, глибиною – на рівні колін – не менше 450 мм, на рівні витягнутої ноги – не менше 650 мм.

Робочий стіл має бути обладнаний підставкою для ніг шириною не менше 300 мм та глибиною не менше 400 мм, з можливістю регулювання по висоті в межах 150 мм та кута нахилу опорної поверхні її межах 20°. Підставка повинна мати рифлену поверхню та бортик на передньому краї заввишки 10 мм.

Екран та клавіатура повинні бути розташовані не ближче 600 мм від очей користувача з урахуванням розміру алфавітно-цифрових знаків та символів.

Робоче місце має відповідати основним антропометричним даним людини. Крісло або стілець на робочому місці повинні мати висоту сидіння 40-50 см від рівня підлоги, а також відповідний кут нахилу спинки. Стегна працюючих мають розміщуватися паралельно підлозі, а стопи ніг – на підлозі або підставці, передпліччя – вертикально; лікті – під кутом 70-90° до вертикальної площини; зап'ястя зігнуті під кутом не більше 20° відносно горизонтальної площини, нахил голови – 15-20° відносно вертикальної площини.

Робоче сидіння користувача повинно бути підйомно-поворотним, м`яким, що регулюється за висотою, кутом нахилу сидіння та спинки, за відстанню спинки до переднього краю сидіння. Регулювання кожного параметра має бути незалежним, плавним або ступінчатим, мати надійну фіксацію.

Хід ступінчастого регулювання елементів сидіння має становити для лінійних розмірів 15-20 мм, для кутових – 2-5°. Ширина та глибина сидіння повинні бути не меншими за 400 мм. Висота поверхні сидіння має регулюватися в межах 400-500 мм, а кут нахилу поверхні - від 15° «перед до 5° назад.

Поверхня сидіння має бути плоскою, передній край – закругленим. Висота спинки сидіння має становити 300x20 мм, ширина – не менше 380 мм, радіус кривизни в горизонтальній площині – 400 мм. Кут нахилу спинки повинен регулюватися в межах 0-30° відносно вертикального положення. Відстань від спинки до переднього краю сидіння повинна регулюватися в межах 260-400 мм.

**4.2 Організація роботи з охорони праці на підприємстві**

Робочий тиждень на підприємстві сорокагодинний, тривалістю 5 робочих днів. Кожному працівникові надається щорічна оплачувана відпустка згідно з чинним трудовим законодавством, а також щорічна додаткова відпустка за особливий характер праці тривалістю до чотирьох календарних днів, згідно з підрозділом «Інші види виробництв» розділу XXII «Загальні професії за всіма галузями господарства», п.58. Усім жінкам, що працюють на підприємстві надається оплачувана відпустка тривалістю 126 календарних днів зв`язку з вагітністю та пологами (70 днів до пологів і 56 після). Для інвалідів створюються особливі робочі місця згідно з рекомендаціями МСЕК, які повині відповідати їх потребам. Адміністрація підприємства проводить для працівників два обовязкові медичні огляди: попередній – при влаштуванні на роботу і періодичний – протягом трудової діяльності відповідно до чинного законодавства. Неповнолітні до роботи на підприємстві не допускаються.

Для кожного нового працівника проводять вступний інструктаж з охорони праці і техніки безпеки. Також на підприємстві проводять регулярний інструктаж для всіх працівників кожних 6 місяців, або позачергово у випадках зазначених у пункті 6.6 Типового положення про порядок проведення навчання і перевірки знань з питань охорони праці. Такі строки проведення інструктажів зумовлені тим, що на підприємстві не виконуються роботи, які несуть із собою підвищену небезпеку.

Всі інструктажі проводяться висококваліфікованими спеціалістами, відповідальними за охорону праці на підприємстві. Спеціалісти, що слідкують за охороною праці на підприємстві працюють на договірних засадах. Кожен з них має виробничий стаж роботи не менше трьох років та пройшов спеціальні навчання з охорони праці. Також ці спеціалісти відповідають за дотримання вимог правил і положень з охорони праці як працівниками, так і адміністрацією підприємства, вони ж відповідають за фіксацію і виправлення всіх виявлених порушень. Всі технічні несправності на підприємстві ліквідовуються в день виникнення.

Для створення комфортних і безпечних умов праці на підприємстві використовують високоякісне обладнання, яке відповідає всім вимогам правил і положень з охорони праці, люмінесцентні лампи з розсіювачами, які забезпечують рівномірне штучне освітлення та системи вентиляції, кондиціонування і обігріву приміщень, які забезпечують сталу вологість і температуру повітря. Також на підприємстві дозволена вільна форма одягу.

**4.3 Основні причини нещасних випадків на виробництві та профзахворювань і заходи щодо їх попередження**

Для програмістів та інших офісних працівників характерні специфічні захворювання, головною причиною яких є малорухомий спосіб життя та постійна перевтома очей. Зокрема дуже поширеними є захворювання спини та різноманітні проблеми із зором на кшталт порушення акомодації, міопії чи далекозорості.

Для уникнення розвитку цих захворювань у працівників компанії офіс даного підприємства має спортзал, до якого всі працівники підприємства мають доступ протягом робочого дня. Також усі робочі місця на підприємстві відповідають всім описаним вище вимогам. Для запобігання появи проблем із зором в компанії враховуються вимоги щодо розташування робочих місць та їх освітлення, використовуються високоякісні монітори, які мінімізують шкідливий вплив на очі та дозволяються щогодинні перерви тривалістю 15 хвилин, які дозволять зняти напруження очей. Крім того, в програмістів чи просто в людей, які багато часу працюють з клавіатурою виникає защемлення серединного нерва в зап`ястному каналі. Для його запобігання потрібно час від часу розминати пальці рук і зап`ястя. Це можна робити в час щогодинних перерв, про які згадувалося вище. З написаного вище очевидно, що адміністрація підприємства приділяє багато уваги питанням охорони праці і докладає всіх зусиль для того щоби попередити розвиток професійних захворювань у працівників, тому не дивно що отримані мною дані про компанію «SoftJourn» свідчать про те, що за весь час існування даного підприємства, на ньому не сталося жодного нещасного випадку. Також, ніхто зі співробітників компанії не хворів на професійні захворювання. Оскільки в компанії «SoftJourn» за весь час її існування не було зафіксовано жодного випадку виробничого травматизму чи нещасного випадку, а також жоден співробітник не постраждав від професійних захворювань, тому жодних економічних втрат підприємство не зазнало. Серед комплексних заходів з охорони праці, що проводяться в ТзОВ «SoftJourn» варто виділити:

* Щомісячний огляд обладнання, перевірка на непридатність для експлуатації, при необхідності проведення своєчасного ремонту;
* Регулярне поповнення аптечки медикаментами першої допомоги один раз в квартал;
* Проведення підготовки опалювальної системи у всьому приміщенні до 1 жовтня;
* Утримування вентиляційного обладнання у технічно-налагодженому стані;
* Щомісячне проведення дня охорони праці з метою усунень порушень норм та правил техніки безпеки;
* Наявність журналу, де фіксуються проведення інструктажів з техніки безпеки і протипожежної безпеки.

Між працівниками підприємства та адміністрацією укладений колективний договір, в якому передбачено виділення коштів на охорону навколишнього середовища та охорону праці.

Висновки до розділу

Керівництво компанії «SoftJourn» приділяє багато уваги дотриманню всіх необхідних нормативно-правових актів з охорони праці, профілактиці професійних захворювань та створенню комфортних умов праці для всіх її співробітників. Це досягається за рахунок періодичних медоглядів, своєчасних інструктажів з питань охорони праці, правильної організації робочих місць, наявності високоякісного обладнання, можливості щогодинних перерв та відвідування спортзалу.

Охорона праці в даній компанії знаходиться на дуже високому рівні, і це підтверджується відсутністю нещасних випадків та професійних захворювань серед працівників.

**ВИСНОВКИ**

В даному диполомному проекті був створений повноцінний шаблон ресурсу електронної комерції. Розроблене програмне забезпечення надає увесь необхідний функціонал для створення любого ресурсу такого типу – від корпоративного сайту до інтернет-магазину.

Для цього в продукті були введені всі базові поняття і функції електронної торгівлі – відображення списку товарів чи послуг, можливість здійснення покупки, розрахунок за покупку та ін. Незважаючи на те, що за кордоном здійснення покупок в інтернеті стало вже звичним явищем, в Україні електронна комерція ще тільки починає набирати обороти, і перспективи розвитку даного ринку є дуже великими, проте для створення великої кількості ресурсів електронної комерції вкрай потрібна система керування вмістом, надійний програмний шаблон на базі якого можна було би з невеликими зусиллями їх реалізовувати.

Поскільки більшість потенційних корисувачів не є спеціалістами з інформаційних технологій, то в даному програмному продукті була введена можливість вибору зручної для користувача мови і простий, інтуїтивно зрозумілий інтерфейс. Використання даної системи позволить дуже спростити та здешевити процес розробки та супроводу власного ресурсу електронної комерції.

Завдяки реалізованому об’єктно-орієнтованому підходу та MVC архітектурі даний програмний продукт дозволяє легко і швидко додавати, видаляти та змінювати його окремі модулі, що робить систему дуже гнучкою та адаптивною під потреби клієнта, а відокремлення інтерфейсу від бізнес – логіки надає можливість зробити на сайті унікальний корпоративний дизайн, не зачіпаючи програмного коду самого продукту.

**СПисок посилань на джерела**

1. Вовк Р.Б., Шекета В.І. Методичні вказівки до виконання дипломної роботи бакалавра для студентів напряму підготовки 6.050103 - “ Програмна інженерія ” : ІФНТУНГ, 2013. – 42 с.
2. <http://peredovik.org/korporativniiy_saiyt_pitannya_ta_vidpovidi.html>
3. http://pidruchniki.ws/18641210/marketing/elektronna\_torgivlya
4. <https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B2%D0%BC%D1%96%D1%81%D1%82%D0%BE%D0%BC>
5. <http://www.1c-bitrix.ua/products/cms/>
6. <http://ru.wordpress.org/>
7. <http://uk.wordpress.com/>
8. <http://uk.wikipedia.org/wiki/WordPress>
9. http://wordpress.org/plugins/wp-e-commerce/
10. <http://www.ratingruneta.ru/research/webanalytics2011/>
11. <http://uk.wikipedia.org/wiki/NetCat>
12. netcat.ru/products/editions/e-commerce/
13. <http://ru.wikipedia.org/wiki/Joomla>
14. <http://extensions.joomla.org/extensions/e-commerce>
15. http://www.virtuemart.ru/
16. http://ru.wikipedia.org/wiki/VirtueMart
17. <http://drupal.org/>
18. <http://uk.wikipedia.org/wiki/Drupal>
19. <http://ru.wikipedia.org/wiki/Ubercart>
20. http://www.ubercart.biz/articles/135
21. <http://uk.wikipedia.org/wiki/C_Sharp>
22. <http://uk.wikipedia.org/wiki/Microsoft_Visual_Studio>
23. <http://ru.wikipedia.org/wiki/ADO.NET_Entity_Framework>
24. НПАОП 0.00-1.28-10 Правила охорони праці під час експлуатації електронно-обчислювальних машин
25. НПАОП 0.00-4.12-05 Типове положення про порядок проведення навчання і перевірки знань з питань охорони праці
26. НПАОП 0.00-4.15-98 Положення про розробку інструкцій з охорони праці
27. НПАОП 0.00-4.21-04 Типове положення про службу охорони праці
28. НПАОП 0.00-4.29-97 Типове положення про кабінет охорони праці
29. НПАОП 0.00-6.07-04 Порядок ведення обліку даних про технічний стан машин, механізмів, устаткування підвищеної небезпеки
30. НПАОП 0.00-6.23-92 Про порядок атестації робочих місць за умовами праці
31. НПАОП 0.00-7.06-94 Єдина державна система показників обліку умов та безпеки праці