

УДК 004.51

Гринчак Є.Г.¹, Василенко О.В.²

¹ студ. РТ-410 НУ «Запорізька політехніка»

² канд. техн. наук, доц. НУ «Запорізька політехніка»

ЕТАПИ ОПТИМАЛЬНОГО ПРОЕКТУВАННЯ САЙТУ

До сайтів можна використовувати критерії оптимальності та етапи проектування, як до будь-якого об'єкту матеріального світу. Критерії оптимальності традиційно розбиті на групи, які знаходяться у протиріччі одна до одної: якість та ціна; при цьому ціна не обов'язково має вартісний характер. Якість має в собі безліч підкритеріїв, головним з яких можна вважати ергономічність, для її максимізації необхідно дотримуватися правил оптимального проектування.

Визначимо процедуру оптимального проектування сайтів:
формування технічної пропозиції та технічного завдання;
визначення основних функцій (критеріїв якості, об'єму робіт);
розробка базового людино-машинного інтерфейсу (Human-Machine Interface, HMI) – (синтез системного рівню);
використання CASE-підходу для інтеграції модулів та інтерфейсів (структурний синтез);
визначення основних інструментів (параметризація);
розробка сайту згідно технічного завдання (ТЗ): робочий та технічний проект, декомпозиція та аналіз, композиція та аналіз, за необхідності оптимізації – повернення на рівні 5, або навіть 3;
розміщення сайту та прикінцеве тестування (натурний експеримент, «пуско-налагоджувальні» роботи), при необхідності оптимізації – повернення на рівні 5, 3, або навіть 1, в залежності від об'єму змін та вимог замовника;
підписання замовником прийому проекту, запуск проекту.

Розглянемо деякі етапи більш детально: 1. Після узгодження ТЗ, визначаються основні функції. Сучасний комерційний сайт є не просто набором пов'язаних гіперпосиланнями сторінок, а своєрідний інструмент для вирішення певного кола завдань, яке в залежності від призначення сайту може бути різним, тому дуже важливо ще до початку створення сайту визначити, які саме опції на сайті будуть потрібні, щоб досягти поставлених цілей з максимальною ефективністю.

2. В ході прийняття рішень необхідно виконувати все більше операцій і розглядати все більше даних, тому, якщо HMI розроблено на основі аналізу кращих варіантів його практичного застосування в конкретній галузі і з урахуванням певних базових принципів, він може поліпшити і спростити процедуру обробки інформації, щоб оператори швидше і точніше виконували

свої обов'язки в частині контролю та поточних завдань в технологічних і виробничих процесах.

3. Проблема зв'язку різномовних, різнорідних за кодом і середовищем програм загострилася у зв'язку зі швидкою зміною архітектури комп'ютерів, появою розподілених, клієнт-серверних середовищ тощо. Виявилася неоднорідність мов програмування (МП) в сенсі як уявлення в них типів даних, так і платформ комп'ютерів, на яких реалізовані відповідні системи програмування, а також в різних способах передачі параметрів між об'єктами в різних середовищах – маршаллінгом даних через різні види операторів віддаленого виклику. Єдиного підходу до вирішення проблеми інтерфейсу не існувало. Стандарт ISO/IEC 11404-1996 визначив підхід до вирішення питань інтерфейсу всіх видів МП за допомогою універсальної мови LI (Language Independent). Однак до теперішнього часу потужної інструментальної його підтримки не існує.

4. Є інструменти груп back-end та front-end. Front-end – це та частина, з якою користувач буде безпосередньо взаємодіяти, для неї доцільно використовувати HTML і CSS [1]. Back-end – це найважливіша частина, оскільки в ній виконується вся логічна частина сайту. Тут вибір мови проектування дуже широкий, але фаворитом є мова Python. Для збереження інформації доцільно використовувати бази даних, найбільш практичною і простою є PostgreSQL основна мова написання – SQL.

5. Для злагодженої роботи back-end і front-end рекомендовано використовувати API-інтерфейси, що в майбутньому дасть нам можливість розширювати можливості функціоналу [2].

6. Для безперервного доступу до ресурсу доцільно розміщувати його на сервісах хостингу або VPS, для цього слід використовувати інструменти Nginx або Apache, Gunicorn або UWSGI (Daphna – якщо сервер асинхронний). Кінцеве тестування проводиться для вчасного виявлення проблем, тут буде доцільно використовувати UNIT-тести, а також інші функції тестування конкретних вузлів і підсистем.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Розуміння основ роботи API і REST API [Електронний ресурс] – Режим доступу: <https://sebweo.com/rozuminnya-osnov-roboti-api-i-rest-api-korotkij-vstup/>.
2. Яка різниця між FRONT-END та BACK-END [Електронний ресурс] – Режим доступу: <https://lgs.lviv.ua/yaka-riznytsya-mizh-front-end-back-end>.