УДК 004.04
А. Стягар
(Тернопільський національний технічний університет імені Івана Пулюя)

СЕРВЕРНА ПЛАТФОРМА NODE.JS

З глобальним ростом маркетплейсів масштабованість для серверних додатків стала важливою як ніколи. Ось чому розробники вибирають Node.js і в якості платформи для серверного програмного забезпечення. У веб-розробці програма повинна взаємодіяти з кількома користувачами та обробляти великі обсяги даних. Для того щоб програма яка виконується на платформі Node.js була стійкою до відмов та масштабованою, критично важливо щоб розробники слідували стратегії неблокуючого вводу/виводу.

Основною відмінністю Node.js від інших технологій є використання однопотокової асинхронної моделі з циклом подій. В асинхронній моделі завдання виконуються почергово в одному потоці. Це простіше, ніж у випадку потоків, оскільки програміст впевнений, що коли одне завдання виконується, в цей же час інше завдання чекає своєї черги. Хоча в однопроцесорній системі в потоковій моделі завдання все-одно будуть виконуватись почергово, програміст, що використовує потоки, має мислити в термінах кількох незалежно виконуваних процесів. Натомість однопотокова асинхронна система завжди буде чергувати виконання завдань навіть на багатопроцесорній системі

Механізм подій допомагає серверу опрацьовувати запити неблокуючим способом і забезпечує високу масштабованість, в порівнянні з традиційними серверами, які для кожного нового запиту користувача створюють новий потік, що підвищує споживання пам’яті та ресурсів процесора. Використання однопотокової моделі дозволяє Node.js обслуговувати набагато більше запитів, ніж традиційні сервери, такі як HTTP-сервер Apache.

Всі бібліотеки Node.js є асинхронними. Це означає, що всі функції Node.js не блокують потік, а виконуються в фоновому режимі. Дана асинхронна модель краще підходить для програм, де багато користувачів одночасно виконують деякі дії, що не навантажують процесор. Наприклад, програми, що отримують температуру з датчиків в режимі реального часу, зображення з відеокамер, пишуть нові повідомлення в чат, отримують нові повідомлення з чату тощо. Вимога дій, які не навантажують процесор, пояснюється тим, що система має один глобальний потік подій, і, якщо в цей потік додати тяжке обчислювальне завдання, тоді система не зможе обслуговувати нових користувачів допоки це завдання не буде виконано, зникнуть всі переваги асинхронної моделі. Тому сервери, написані на Node.js, підходять тільки для виконання завдань, які не є тяжкими обчислювальними процесами.

Аналізуючи все вище сказане можна зробити висновок що для звичайного веб-додатку, де активно працюють з базою даних, підхід Node.js передбачає помітне підвищення продуктивності в порівнянні з іншими технологіями. Також це працює і для додатків де є велика кількість HTTP запитів до внутрішніх та зовнішніх сервісів. Тому сервери, написані на Node.js, підходять тільки для виконання завдань, які не є тяжкими обчислювальними процесами. Також вони підходять як сервери для обслуговування клієнтів з вузьким каналом даних, там де великий час відклику на дії, для повільних запитів. Сервери на Node.js добре виконують роль посередників в клієнт-серверній системі. На основі цих факторів можна зробити висновок що на даний час платформа Node.js є однією з найкращих серверних платформ для на писання програм на JavaScript.