МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ОДЕСЬКА НАЦІОНАЛЬНА АКАДЕМІЯ ХАРЧОВИХ ТЕХНОЛОГІЙ

XI МІЖНАРОДНА НАУКОВО-ПРАКТИЧНА КОНФЕРЕНЦІЯ

ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ І АВТОМАТИЗАЦІЯ – 2018

Збірник доповідей

Частина II

Одеса,
4-5 жовтня 2018

ЗМІСТ

УДК 004.424+004.423

ХАРАШ К.М., ОЛЬШЕВСЬКА О.В.
ОНАХТ (Україна)

ПЕРЕВАГИ ФУНКЦІОНАЛЬНОЇ ПАРАДИГМИ ПРОГРАМУВАННЯ

Наразі функціональне програмування не так широко розповсюджено серед розробників програмного забезпечення (ПЗ), і під час розробки частіше використовують імперативний або інакше декларативний стиль програмування. Імперативний стиль являє собою встановлення чіткого алгоритму та послідовності виконуваних дій. Імперативне програмування є більш простим для вивчення і розуміння тому ширше використовується у навчанні. В зв'язку з тим, що засвоївши імперативний стиль, розробник не часто або взагалі не звертає уваги на інші парадигми, він має менші можливості для росту та вдосконалення своїх навичок. Програмування у функціональному стилі, окрім іншого, змінює патерни мислення розробника, що призводить до написання гарнішого, структурованішого та загалом чистішого коду.

Функціональне програмування як парадигма являє собою відхід від послідовностей та зведення програми до набору чітких функцій. Такий підхід у більшій степені математичний і для його освоєння, необхідно суттєво змінити бачення процесу програмування.

Функціональне програмування бере свій початок у 1930-х роках, коли вчений з Принстона на ім'я Алонзо Чорч сумісно з іншими вченими розробив формальну систему для вирішення математичних задач, яка в результаті отримала назву Лямбда-числення. Система була чимось на кшталт мови програмування для уявної машини, з неіснуючою на той час, архітектурою. Лямбда-числення засноване на функціях, які приймають функції у якості аргументів та повертають функції у якості результату. Варто зазначити, що в той самий час, Алан Тьюрінг незалежно працював над іншою формальною системою, що надалі отримала назву Машина Тьюрінга. Два вчених у той час змагалися в рішенні математичних задач та намагалися створити для них найпотужнішу формальну систему. В результаті виявилося, що обидві системи були однаково потужні [1].

Пізніше, в 1950-х роках роботами Алонзо Чорча зацікавився професор Масачусетського технологічного інститута Джон Маккарті і в 1958 році представив мову обробки листів Lisp, яка фактично була імплементацією Лямбда-числення для комп'ютерів з архітектурою фон Неймана. Апаратна ж реалізація Лямбда-числення побачила світ в 1973 році, коли програмісти з того ж Масачусетського технологічного інституту створили комп'ютер, названий Lisp-машиною.

Таким чином, функціональне програмування, яким ми його нині знаємо, є реалізацією ідей Чорча з певними змінами та виключеннями з початкової ідеї.

Функціональне програмування, як будь-яка технологія, має як переваги так і недоліки, однак в рамках даної роботи увага зосереджена на перевагах. Одним із головних є відладка програми. Існує можливість відтворити проблему, що виникла, так як помилка в функції не обов'язково залежить від коду, що виконувався попередньо, але залежить від функції, у якій помилка і виникла. Під час відладки можна рухатися по стеку викликів і безпомилково та швидко виявити джерело проблеми, а отже, і усунути її.

Багатопоточність у функціональному стилі реалізується легше, ніж у звичному імперативному. Це пов'язано з функціональною логікою. У функціональній програмі не можна два чи більше рази змінювати дані ні одним і тим самим потоком, ні різними. Таким чином, розробник не повинен замислюватися про проблеми, з якими часто стикаються в декларативних мовах програмування [2].

Функціональні програми можна запускати без зупинки серверів, що надзвичайно важливо в телекомунікаційних системах, коли робочий процес не можна переривати, а необхідні оновлення потрібно обов'язково встановити.

Цікава властивість функціональної парадигми, яка не зустрічається в імперативній - це ліниві обчислення. Суть їх полягає в тому, шо функції запускаються лише у тому випадку, якщо в них дійсно є необхідність, на відміну від звичного для розробників стилю програмування, коли всі обчислення виконуються послідовно, навіть якщо вони на даний момент не потрібні. Це, в свою чергу, призводить до збільшення ефективності кода при збереженні цілісності програми.

Безперечно, ліниві обчислення тягнуть за собою ряд недоліків, однак математики знайшли засоби, які дозволяють обійти проблеми, що виникають і, при тому, не втратити функціонального стилю.

Елементи Лямбда-числення присутні у останніх версіях таких відомих мов програмування як Java, Kotlin, Python, що звичайно дає можливості зробити програму більш математичною, ефективною та значно оптимізувати стандартні рішення. Проте, існують також і чисто функціональні мови, серед яких найпопулярнішими є Haskell, Erlang, Lisp та Scala. Вказані мови мають доволі широке співтовариство розробників, великі можливості для розробки, широкі набори засобів розробки, повну всеохоплюючу документацію. Однак, майже вся література за даними технологіями доступна у більшості англійською мовою. Ланка входження до даних мов висока, однак повністю відповідає можливостям, які відкриваються розробнику.

Підсумовуючи, можна зробити висновок, що функціональний стиль програмування більш надійний, стабільний і для серйозних завдань підходить набагато більше за імперативний. Функціональні мови мають свою бажану область застосування, проте не знають собі рівних у вирішенні існуючих проблем. Використання розглянутого стилю в розробці змінює мислення розробника, що неминуче призводить до покращення стилю написаного ним програмного коду та розроблюваного ПЗ загалом. Врешті, програмування у функціональному стилі є цікавим процесом, тому за можливості, варто звернути увагу саме до такого підходу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. “Functional Programming For The Rest of Us,” *The ten project management commandments*, 19-Jun-2006. [Online]. Available: http://www.defmacro.org/2006/06/19/fp.html. [Accessed: 12-Sep-2018].
2. Щекн-Итрч “Функциональное программирование для всех,” / *Хабр*, 23-Apr-2012. [Online]. Available: https://habr.com/post/142351/. [Accessed: 20-Sep-2018].

XI МІЖНАРОДНА НАУКОВО-ПРАКТИЧНА КОНФЕРЕНЦІЯ

ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ І АВТОМАТИЗАЦІЯ – 2018

ОДЕСА
4 – 5 ЖОВТНЯ, 2018

Збірник включає доповіді учасників XI Міжнародної науково-практичної конференції «Інформаційні технології і автоматизація – 2018»

Редакційна колегія: Котлик С.В., Хобін В.А.

Комп'ютерний набір і верстка: Шамрай О.А.

Відповідальний за випуск: Котлик С.В.

ДЛЯ НОТАТОК