

КОМПОНЕНТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ ЯК СУЧАСНИЙ ПІДХІД ДО РОЗРОБКИ СКЛАДНИХ ПРОГРАМНИХ СИСТЕМ

Огляд переваг компонентно-орієнтованого програмування над об'єктно-орієнтованим та способи вирішення поширених проблем.

З моменту створення першого програмованого комп'ютера у 1946 році програмування не збавляло обертів. Вимоги до програм росли, а разом з ними з'являлися все потужніші мови та революційніші підходи програмування. Так і зараз, компонентно-орієнтоване програмування починає плавно, але впевнено витісняти об'єктно-орієнтоване.

Необхідно визначити об'єктивні причини, через які компонентно-орієнтоване програмування в наш час набирає все більшу популярність.

Зараз дуже гостро стоять задачі швидкої та злагодженої розробки великих та складних продуктів. Будь яка програма повинна мати гнучку структуру та здатність до виправлення і доповнювання функціональності, при мінімальних затратах ресурсів. Навіть найменша помилка у питанні вибору структури або підходу програмування може призвести до фатальних наслідків при спробі виправити, змінити або додати функціональність.

Об'єктно-орієнтоване програмування відоме практично кожному програмісту. Згідно з цим підходом, програма – це сукупність об'єктів, які містять набір базових характеристик своїх класів та взаємодіють між собою. Разом з цим підхід пропонує чотири базові концепції: інкапсуляцію, успадкування, поліморфізм та абстракцію. Це, безумовно, чудові принципи з тисячами можливих реалізацій [1].

Компонентно-орієнтоване програмування (КОП) розглядає програму, як набір інкапсульованих компонентів, які взаємодіють з іншими компонентами через контейнер. Кожний компонент являється окремою підпрограмою, яка описує тільки одну функціональність та надає інтерфейс доступу до себе. Контейнери компонентів реалізують взаємодію компонентів та їх функціональність. При цьому компонентно-орієнтоване програмування практично виключає використання успадкування та привносить нові концепції: інтроспективність (здатність само опису), модульність, персистентність (здатність зберігати та відновлювати свій певний стан) та здатність до багатократного використання одного й того ж компонента [2].

При розробці великих програмних продуктів зі складною архітектурою у об'єктно-орієнтованого програмування з'являється ряд проблем. Компонентно-орієнтоване програмування дає можливість виправити ці недоліки і привносить більшу функціональність.

Взаємозалежність об'єктів і проблема крихкого базового класу. Так як КОП оперує тільки незалежними між собою компонентами та контейнерами, що містять їх екземпляри, то відпадає необхідність у використанні наслідування [3].

Недостатній контроль інформації. Небезпека витоків пам'яті. Компоненти являються частиною своїх контейнерів, а тому при деструкції контейнера вивільняється пам'ять усіх його компонентів [3].

Залежність від мови програмування і платформи. Згідно ідеології КОП, кожен компонент – це окрема незалежна функціональність, а отже її запросто можна виділити в окремий модуль програми, який матиме певний інтерфейс доступу, але може бути скомпільований на будь якій мові і платформі [3].

Важкість організації архітектури. Вся структура спрощується до набору контейнерів, до яких можна легко підключати необхідні компоненти із доступного переліку, створюючи необхідну функціональність [3].

Складність контролю життєвого циклу об'єкта. Кожен компонент має здатність зберігати та відновлювати власний стан та вивантажувати його [3].

Відповідно до компонентно-орієнтованого програмування програма подрібнена на велику сукупність модулів, що дає можливість швидко та легко виправляти, видозмінювати або доповнювати функціональності [4].

Отже, компонентно-орієнтоване програмування виправляє більшість недоліків попередника та привносить багато переваг, таких як легка структурованість, виправлення та доповнюваність, що особливо важливо у розробці масштабних програмних продуктів.

Література

1. Герберт Шилдт. Полный справочник по C++ 4 видання. – М.: Osborne, 2016.– 800 с.
2. <http://www.uchi-it.ru/7/1/12.html>
3. <https://habrahabr.ru/sandbox/96615/>
4. https://ru.wikibooks.org/wiki/Компонентно-ориентированное_программирование