УДК 004.891.3:004.3

**Майер Ілля Сергійович**
*студент кафедри автоматизації та управління в технічних системах*
*факультету інформатики та обчислювальної техніки*
*Національного технічного університету України*
*«Київський політехнічний інститут імені Ігоря Сікорського»*
**Майер Илья Сергеевич**
*студент кафедры автоматизации и управления в технических системах*
*факультета информатики и вычислительной техники*
*Национального технического университета Украины*
*«Киевский политехнический институт имени Игоря Сикорского»*
**Maier Illia**
*Student of the Department of Automation and Control in Technical Systems of the*
*Faculty of Computer Science and Computer Engineering of the*
*National Technical University of Ukraine*
*«Igor Sikorsky Kyiv Polytechnic Institute»*

**Хмелюк Марина Сергіївна**
*асистент кафедри автоматизації та управління в технічних системах*
*факультету інформатики та обчислювальної техніки*
*Національний технічний університет України*
*«Київський політехнічний інститут імені Ігоря Сікорського»*
**Хмелюк Марина Сергеевна**
*ассистент кафедры автоматизации и управления в технических системах*
*факультета информатики и вычислительной техники*
*Национальный технический университет Украины*
*«Киевский политехнический институт имени Игоря Сикорского»*
**Khmeljuk Marina**
*Assistant of the Department of Automation and Control in Technical Systems*
*of the Faculty of Computer Science and Computer Engineering*
*National Technical University of Ukraine*
*«Igor Sikorsky Kyiv Polytechnic Institute»*

# ПІДТРИМКА ЖИТТЄВОГО ЦИКЛУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

# ПОДДЕРЖКА ЖИЗНЕННОГО ЦИКЛА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

# SOFTWARE LIFE CYCLE SUPPORT

***Анотація.*** *Процес сучасної розробки програмного забезпечення орієнтований на життєвий цикл програмного продукту. Всі існуючі в даний час технології, методики та стандарти безпосередньо або опосередковано стосуються або регламентують етапи життєвого циклу, як за функціональним наповненням, так і за змістом. Процес розробки програмних систем тісно пов'язаний з областю управління проектами, тому що будь-який програмний продукт є унікальним результатом. Від організації цього процесу безпосередньо залежать основні характеристики виконання програмного проекту – терміни виконання, запланований бюджет, якість готового продукту. Але професійне управління проектами саме по собі не може забезпечити досягнення зазначених характеристик. Важливу роль у цьому відіграє архітектура програмної системи, досвід і кваліфікація учасників команди розробки, а також правильне документування всіх процесів розробки програмного забезпечення.*

***Ключові слова:*** *програмний сервіс, життєвий цикл програмного забезпечення.*

***Аннотация.*** *Процесс современной разработки программного обеспечения ориентирован на жизненный цикл программного продукта. Все существующие в настоящее время технологии, методики и стандарты непосредственно или косвенно касаются или регламентирующих этапы жизненного цикла, как по функциональному наполнению, так и по содержанию. Процесс разработки программных систем тесно связан с областью управления проектами, потому что любой программный продукт является уникальным результатом. От организации этого процесса напрямую зависят основные характеристики выполнения программного проекта – сроки выполнения, запланированный бюджет, качество готового продукта. Но профессиональное управление проектами само по себе не может обеспечить достижения указанных характеристик. Важную роль в этом играет архитектура программной системы, опыт и квалификация участников команды разработки, а также правильное документирование всех процессов разработки программного обеспечения.*

***Ключевые слова:*** *программный сервис, жизненный цикл программного обеспечения.*

***Summary.*** *The process of modern software development focuses on the life cycle of a software product. All currently existing technologies, techniques and standards directly or indirectly relate or regulate the stages of the life cycle, both in terms of functional content and content. The process of developing software systems is closely linked to the project management area, because any software product is a unique result. From the organization of this process directly depend on the main characteristics of the implementation of the program project – the timing, the planned budget, the quality of the finished product. But professional project management alone can not achieve the achievement of these characteristics. An important role in this is played by the architecture of the software system, the experience and qualifications of the team development team, as well as the proper documentation of all software development processes.*

***Key words:*** *software service, software life cycle.*

**Постановка проблеми.** Комп'ютеризація в світі збільшує свою швидкість з кожним днем, а кількість розроблюваного програмного забезпечення збільшується з ще більшою швидкість, що створює велике навантаження на керуючі гілки відповідних підрозділів.

**Аналіз останніх досліджень і публікацій.** Дослідження базується на працях таких видатних авторів у галузі розробки програмного забезпечення як К. Петерсона [1], Д. Шора [2], Л. Марка [3], К. Бека [4], Б. Бохема [5].

**Формулювання цілей статті (постановка завдання).** Аналіз різнобічних наукових праць та досліджень. Виділення найсильніших рис різних підходів до розробки програмного забезпечення.

**Виклад основного матеріалу.** В реальних умовах проектування інформаційних систем — це пошук способу, який забезпечить необхідну функціональність системи засобами існуючих технологій з урахуванням встановлених обмежень.

Під методологією розробки розуміють набір методів та критеріїв оцінки, які використовуються для постановки задачі, планування, контролю та в кінцевому підсумку — для досягнення поставленої цілі. Сам процес розробки описується моделлю, котра визначає послідовність найбільш загальних етапів та очікуваних результатів.

Зараз існує досить багато методологій управління проектами та відповідного програмного забезпечення. Всі вони мають свої переваги та недоліки. Розглянемо деякі основні методологій, які довели свою ефективність при розробці ПЗ.

1. Каскадна модель («Waterfall Model»)

Каскадна модель одна з найбільш традиційних та загально використовуємих методологій для програмної розробки. Ця модель життєвого циклу,, зазвичай вважається як класичний стиль програмної розробки. Вона висвітлює процес програмної розробки як лінійну послідовну течію — під цим розуміють, що будь-яка фаза в процесі розробки починається тільки якщо попередня фаза завершена. Цей підхід не надає можливості повертатися назад до попередньої фази, щоб внести зміни в вимогах. Цю методолгію застосовують коли вимоги вже обговорені, немає проблем з кваліфікованими фахівцями, у відносно невеликих проектах.

Переваги:

- каскадна модель дуже просто та легка для розуміння та використання, що дійсно добре для новачків-розробників;
- в цій моделі фази виконуються та завершуються лише один раз — це суттєво зберігає велику кількість часу;
- цей підхід до розробки більш ефективен на невеликих проектах, де вимоги дуже добре сформульовані;

Недоліки:

- ця модель може використовуватися тільки коли чітко визначені попередні вимоги;
- ця модель не прийнятна для підтримки проектів;
- головний недолік цього методу якщо додаток знаходиться в стадії тестування немає можливості повернутуся до попереднього етапу, щоб внести деякі зміни;
- немає можливості показувати працюючий додаток доки не досягнуто останньої стадії циклу;
- не ідеальна для проектів, де вимоги недостатньо визначені та мають багато місць для зміни.

2. Гнучка методологія розробки («Agile model»)

Гнучка методологія використовується для проектування впорядкованого управління процесом

розробки котрий дозволяє вносити постійні зміни в розробку проекту. Ця модель використовується для максимального зменшення ризику при розробці продукту в короткі часові проміжки котрі називаються ітераціями і зазвичай тривають від одного тижня до одного місяця.

Цю модель слід застосовувати коли потреби користувачів постійно змінюються в динамічному бізнесі. Зміни на Agile реалізуються за меншу ціну із-за постійних спринтів. На відміну від каскадної моделі, в гнучкій моделі для старту проекту досить лише невеликого планування.

Переваги:
- гнучка методологія має адаптивний підхід котрий дозволяє змінювати вимоги клієнтів;
- безпосередній зв'язок та постійні відгуки замовників або їх представників не залишає місця для невизначенностей.

Недоліки:
- ця методологія зосереджена на створенні програмного забезпечення раніше, ніж документації, звідси може бути нестача документації;
- процес розробки може вийти з під контролю, якщо замовник чітко не представляє кінцевий результат проекту.

3. Спіральна модель («Spiral model»)

Спіральна модель є досить комплексною моделлю що зосереджується на ранньому виявленні та зменшенні ризиків проектів. В цій методології розробники починають з малих масштабів потім виявляють можливі ризики в проекті, складаються план для запобігання їх і в завершенні вирішують чи варто переходити до наступної стадії проекту, щоб зробити наступну ітерацію спіралі. Успіх будь-якого життєвого циклу спіральної моделі залежить від надійного, уважного та грамотного керівництва проекту.

Ця модель не підійде для малих проектів, вона резонна для складних і дорогих, наприклад, таких, як розробка системи документообліку для банку, коли кожен наступний крок вимагає більшого аналізу для оцінки наслідків, ніж програмування.

Переваги:
- об'ємний аналіз ризиків звичайно ж зменшує можливість провалу
- ця модель досить добре підходить для великих та ризикованих проектів;
- в спіральній моделі, додаткову функціональність можна додати пізніше;
- більш задовільна для високоризикованих проектів, де бажання та потреби замовника можуть змінюватися час від часу.

Недоліки:
- досить витратна модель з точки зору розробки;
- в цілому успіх проекту залежить від фази аналізу ризиків, невдача в цій фазі може спричинити провал проекту;
- не підходить для низькоризикованих проектів;

4. Екстремальне програмування («Extreme Programming»)

Екстремальне програмування — гнучка методологія розробки програмного забезпечення. Ця методологія, відома ще як «XP», в основному використовується для створення додатків в дуже нестабільному середовищі. Це додає великою гнучкості під час процесу моделювання. Головна перевага цієї методології — це те, що вона досить малозатратна. В моделі XP доволі часто буває, що вартість змін вимог на більш пізній стадії проекту може бути досить високою.

Переваги:
- методологія екстремального програмування робить акцент на залученості замовника;
- ця модель допомагає визначити доцільні плани і графіки та дає можливість розробникам самостійно зафіксувати їхні графіки що є безумовно найбільшою перевагою;

Недоліки:
- ця методологія ефективна лише якщо розробники повністю зосереджені та захоплені розробкою;
- ця модель вимагає занадто багато змін в розробці, що дуже трудозатратно для розробника;
- в цій методології, як правило, неможливо визначити точні трудозатрати тому, що на початку проекту ніхто не уявляє цілий об'єм робіт та вимог.

**Порівняння методологій**

Найбільш популярними та одними із батьків усіх інших є каскадна та Agile методології. З однієї сторони (каскадна) ми плануємо все до найменших деталей, визначаємо жорсткі терміни, маємо фіксований бюджет, але проблеми приходять коли потрібно зробити якісь правки або взагалі змінити русло куди йде розробка програмного продукту. З іншої сторони (Agile) маємо гнучку систему спринтів, замовник платить за спринти, досить легко вносити зміни, але досить складно встановлювати терміни для випуску продукту та важко прорахувати бюджет. У таблиці 1 проведено порівняння вищезгаданих методологій за певними критеріями.

**Висновки з даного дослідження і перспективи подальших розвідок у даному напрямі.** Зростаючи кожного дня попит на програмне забезпечення створює необхідність правильної постановки процесів в підрозділах розробки, аби вони могли задовольняти потреби бізнесу та створювати якісний, підтримуваний та функціональний продукт. Існуючі методології розробки були створені та перевірені часом, але їх недоліки інколи є критичними для їх користувачів, що відкриває простір для покращення та створення новий підходів до розробки програмного забезпечення.

*Таблиця 1*

**Порівняння методологій**

| | Каскадна | Agile | Спіральна | XP |
|---|---|---|---|---|
| Детермінованість вимог | повністю | основні | бажано основні | початкові |
| Розмір проекту | нижче середн. | великий | великий | нижче середн. |
| Контроль витрат | високий | низький | нижче середн. | низька |
| Гарантія успіху | невисока | висока | висока | середня |
| Рівень ризику | високий | низький | низький | вище середн. |
| Залученість замовника | низька | вище середн. | середня | висока |
| Простота використання | висока | вище середн. | нижче середн. | низька |
| Повернення до попередньої фази | – | + | –/+ | + |

**Література**

1. Петерсон К. The Waterfall Model in Large-Scale Development / Кай Петерсон. — 386 с.
2. Шор Д. The Art of Agile Development / Джеймс Шор., 2007. — 440 с.
3. Марк Л. Agile Project Management For Dummies / Лейтон Марк. — Хобокен, NJ: For Dummies, 2012. — 360 с.
4. Бек К. Extreme Programming Explained: Embrace Change / К. Бек, К. Андрес. — 224 с.
5. Бохем Б. The Incremental Commitment Spiral Model: Principles and Practices for Successful Systems and Software / Б. Бохем, Д. Лейн, С. Кулманойвонг, Р. Тьорнер., 2014. — 336 с.

**References**

1. Peterson K. The Waterfall Model in Large-Scale Development / Cay Peterson. — 386 p.
2. Shore D. The Art of Agile Development / James Shore, 2007. — 440 p.
3. Mark L. Agile Project Management For Dummies / Leyton Mark. — Hoboken, NJ: For Dummies, 2012. — 360 p.
4. Beck K. Extreme Programming Explained: Embrace Change / K. Beck, C. Andres. — 224 p.
5. Boehm B. The Incremental Commitment Spiral Model: Principles and Practices for Successful Systems and Software / B. Boehm, J. Lane, S. Koolmanojwong, R. Turner, 2014. — 336 p.