ПРИНЦИПИ ПОБУДОВИ СИСТЕМИ ДІЯЛЬНОСТІ ПРОГРАМНОГО ПРОЕКТУ

Розглядаємо питання управління програмними проектами за допомогою методик/наборів методик. Будь-яка програмна розробка унікальна за своєю суттю, за умовами виконання проекту, а тому застосування методології і методик вимагає їх адаптації.

Цікаво було б розробити понятійну базу процесу виробництва програмного забезпечення, яка не була б прив'язана до конкретних методологій.

Виробничі функції і виконавці

Процес реалізації програмного проекту можна розглядати як одну велику виробничу функцію, виконання якої приводить проект від задуму до програмного продукту і далі, від поставки виробу споживачам до завершення його експлуатації. Для такої виробничої функції природно говорити про узагальненого виконавця, об'єднуючого всіх реальних дійових осіб, що виконують проект, а також про узагальненого користувача, який застосовує одержувані в ході реалізації проекту результати в своїй діяльності. Для продуктивного розгляду вказану виробничу функцію потрібно уміти розбивати на частини. Є два види такого розбиття:

- Можна структурувати узагальненого виконавця, іншими словами, конкретизувати виконавців, що відповідають за різні аспекти виконання функції.
- Можна структурувати виконання функції, розбиваючи її на складові, визначаючи призначення кожної з складових і зв'язку між ними так, щоб результат спільного виконання збігався з необхідним результатом цієї функції.

Обидва види розбиття виробничих функцій допускають продовження в глибину: для виконавців - до рівня груп виконавців і конкретних індивідуумів, для функцій - до таких фрагментів, які розглядаються як неподільні одиниці. Обидва вони існують одночасно і взаємопов'язано.

Розбиття проекту на складові двох видів не є незалежним і однозначним хоча б тому, що виконавцям різних функцій висуваються різні кваліфікаційні вимоги. Є і інші причини варіативності структуризації проектів. Так, оскільки метою будь-якої програмної розробки є задоволення потреб користувачів, а самі ці потреби з часом змінюються, то виконавцям доводиться постійно адаптувати проект до нових потреб. Змінюються і умови проведення розробки, які також потрібно враховувати при управлінні проектом.

Процес виконання проекту є цілеспрямованою динамічною системою дій, що реалізують виробничі функції за допомогою їх виконання виконавцями, яка розвивається в часі. Цілеспрямованість тут означає визначеність задач, для вирішення яких виконується кожна з функцій на всіх рівнях розбиття, а динамічність - мінливість структури (розбиття) в ході реалізації відповідно до потреб вирішення певних задач.

Системи і елементи проектних дій

- Динаміка - одна з основних характеристик процесу розвитку проекту, яка разом з розбиттям функцій дозволяє розглядати проект як систему взаємопов'язаних цілеспрямованих видів діяльності суб'єктів-виконавців, що реалізують функції. З погляду теорії діяльності кожна робота цієї системи характеризується такими елементами.
- Суб'єкт - виконавець дії, що володіє певними властивостями і можливостями, що дозволяють займатися її виконанням. Суб'єкти, виконавці програмних проектів, можуть бути індивідуумами або групами індивідуумів. У жорстких підходах до проектування планування часто поширюють до індивідуальних завдань, гнучкі методології не вимагають розгляду структури суб'єкта-виконавця (приклад - робоча група MSF).

  Стисло: суб'єкт - це той, хто в змозі виконувати дану дію.
- Мета - призначення дії, явно визначає напрям її розвитку. Мета завжди співвідноситься з іншими складовими системи дій: яке місце дана дія займає в системі, як і яким чином використовуються результати дії в інших проектних і зовнішніх діях.

  Стисло: мета - це те, для чого дана дія виконується.
- Матеріали і ресурси - те, що використовується і переробляється при виконанні дії. Термін "матеріали" частіше використовується, коли аспект витрат не цікавить або відсутній зовсім. Приклад - специфікації програмного виробу використовуються, але не витрачаються в різних видах діяльності по виробництву програм: завдання на розробку, еталон для перевірки готової програми, шаблон складання документації тощо. Ресурси можна визначити як матеріали, що витрачаються.

  Стисло: матеріали і ресурси - це те, з чого в даній діяльності продукуються результати.
- Засоби і інструменти. Звична діяльність припускає застосування допоміжних засобів, які підтримують її виконання або є необхідними для отримання змістовних результатів. Засоби - загальне поняття, що відображає можливість застосування об'єкту в різних діях. Інструменти, як правило, спеціалізуються для конкретних видів дії.

  Стисло: засоби і інструменти - це те, за допомогою чого в даній діяльності продукуються результати.

- Методи - сукупність прийомів для діяльності даного типу, розпоряджень, угод і рекомендацій, регламентів, що забороняють або обмежують можливість застосування засобів, інструментів, інших методів, доступ до загальних ресурсів. Проектні рішення, що стосуються методів, за різними методологіями програмування можуть бути протилежними. Так, для традиційних методологій звичними є регламенти розмежування доступу до файлів, що містять інтегрований код системи, тоді як в рамках методології екстремального програмування принципово, що для цих файлів встановлюється загальний доступ для всіх розробників проекту.

  Стисло: методи - це те, що вказує на спосіб виконання даної дії.
- Результат. При виконанні дії суб'єкт-виконавець створює різні продукти, обумовлені і не обумовлені цілями. Продукти можна розглядати як матеріалізовані наслідки діяльності, які з'являються в ході її виконання — артефакти. Сукупність всіх продуктів, одержаних в ході виконання дії, називається її результатом. Результат може містити продукти, корисні з погляду цілей проекту і цілей дії, і продукти, не сприяючі просуванню проекту до його глобальних цілей. Корисність продукту або її відсутність визначається проектними угодами. Загальний критерій корисності - включення продукту в інші дії системи проекту як атрибут.

  Стисло: результат - це те, що фактично продукується в даній діяльності.

Будь-яка діяльність є цілком певна частина деякої загальної системи дій, що охоплює групу суб'єктів-виконавців. Дії, суб'єкти яких не потрапляють у виділену групу, є оточенням даної системи. Оточення пов'язане з виділеною системою такими способами:

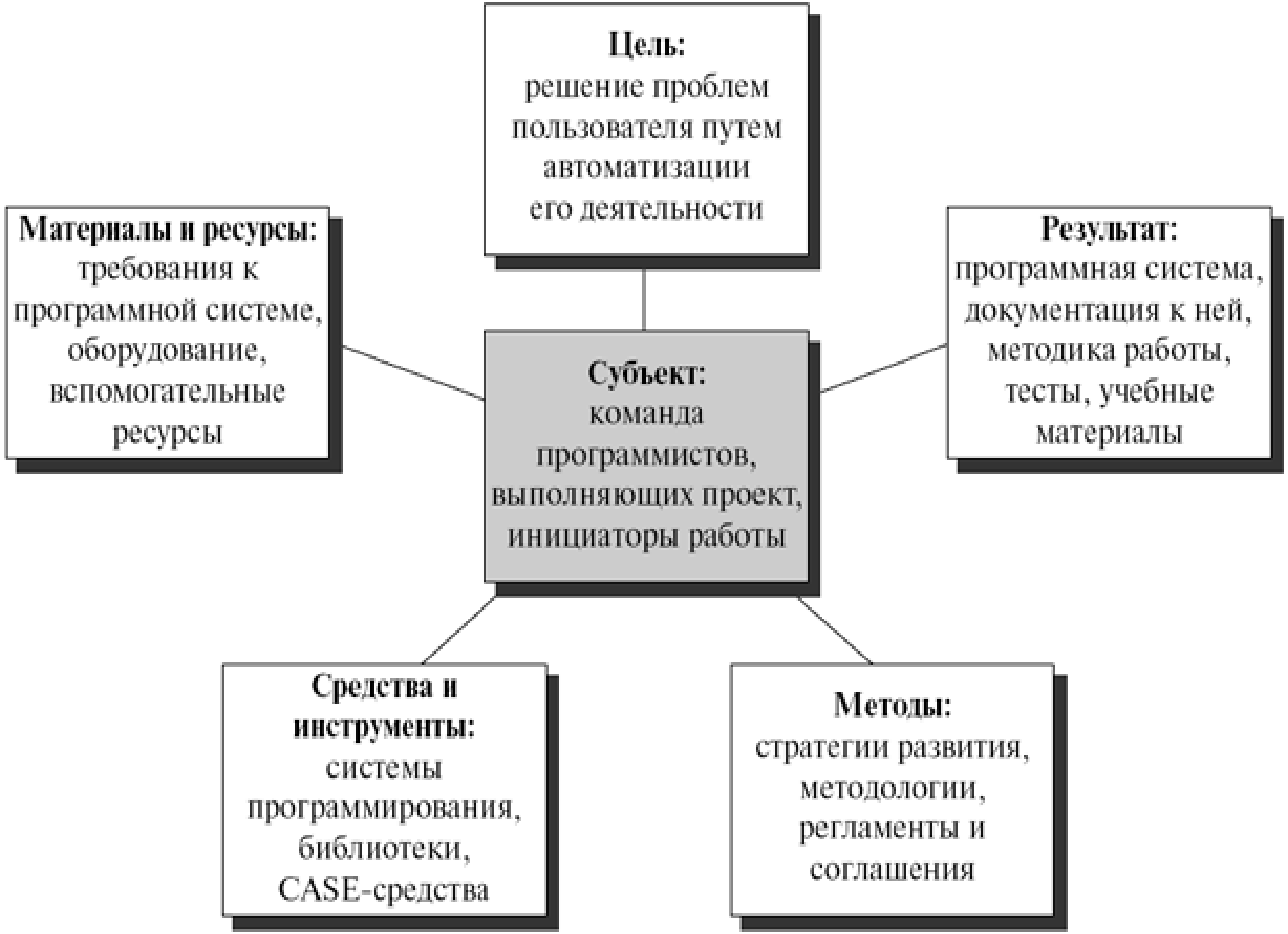
- з оточення поставляються елементи дій системи;
- діям оточення передаються результати дій системи;
- система в цілому або її окремі дії є елементами дій оточення.

Розгляд дії і системи дій відособлений від інших дій - абстрактний прийом, що дозволяє побачити структуру дії або системи, описати елементи і зв'язки. Представлений список елементів відображає структуру будь-якої змістовної дії, а способи зв'язку задають все розмаїття систем дій. З його допомогою ми в змозі дати поняття автоматичної і автоматизованої дій.

Діяльність називається автоматизованою, якщо вона призводить до результатів, які продукуються іншою, неавтоматизованою діяльністю (реальною або, можливо, умоглядною), але з меншими витратами часу і/або зі скороченням витрати ресурсів.

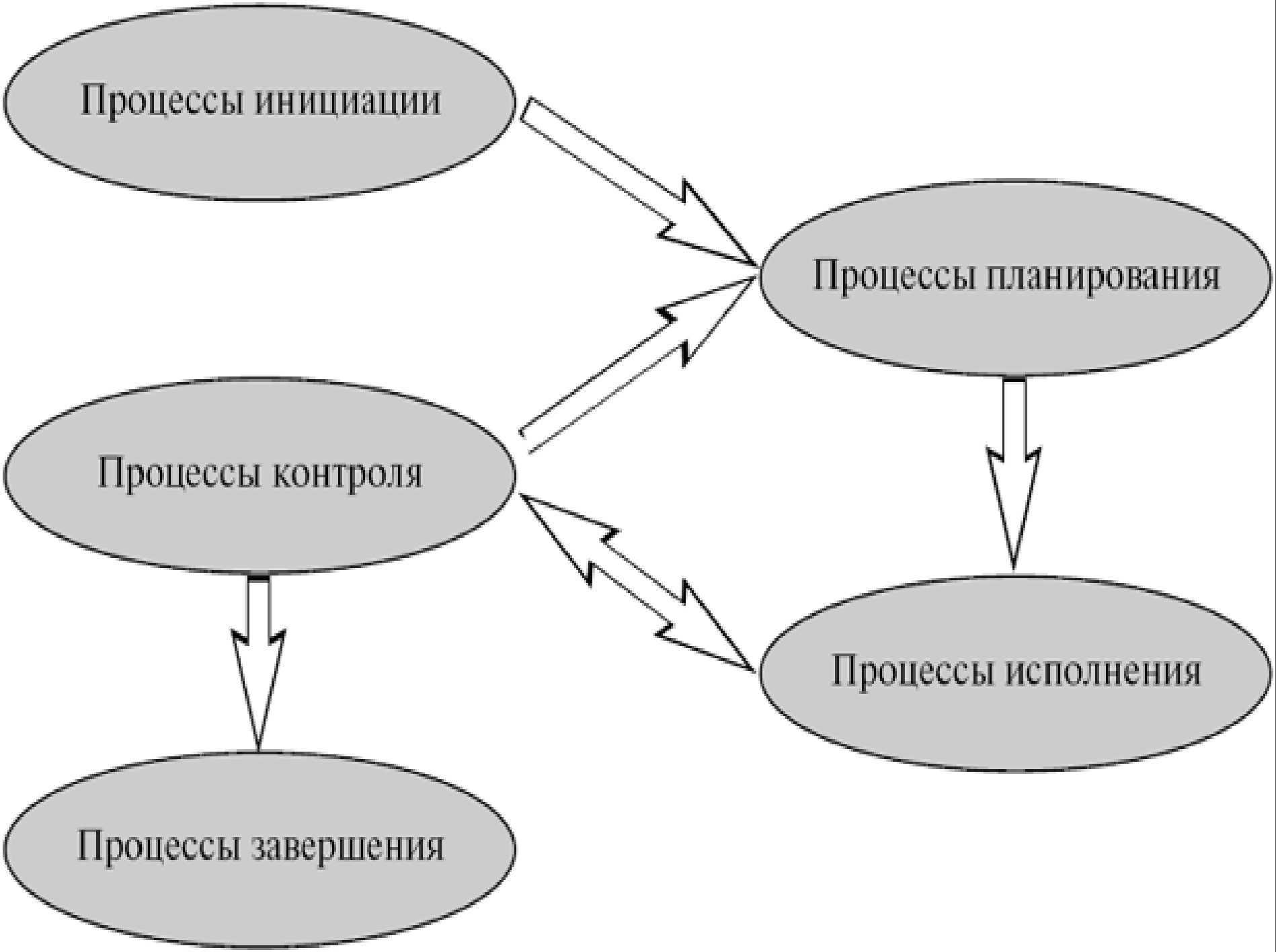
Діяльність називається автоматичною або, точніше, автоматично виконуваною, якщо вона має неживий суб'єкт, діючий з іншими елементами за заданою зовнішнім чином програмою. Це те саме, що вироджена діяльність, що не має суб'єкта, акт виконання якої здійснюється шляхом зовнішньої активізації її єдиного засобу. Третя еквівалентна точка зору - визначення автоматичної дії як такої, у якої (живий) суб'єкт виконує єдину дію-активізацію.

Мета програмування – побудова автоматичних і автоматизованих дій. Мета методологій програмування при такому розгляді - це побудова автоматичних і автоматизованих дій, замінюючих собою неавтоматизовані аналоги в системі дій програмного проекту.


Цель:
решение проблем пользователя путем автоматизации его деятельности
Материалы и ресурсы:
требования к программной системе, оборудование, вспомогательные ресурсы
Субъект:
команда программистов, выполняющих проект, инициаторы работы
Результат:
программная система, документация к ней, методика работы, тесты, учебные материалы
Средства и инструменты:
системы программирования, библиотеки, CASE-средства
Методы:
стратегии развития, методологии, регламенты и соглашения

Стандарт PMBOK (Project Management Body of Knowledge) інституту менеджменту проектів PMI, який фактично є загальновизнаним на світовому рівні (достатньо вказати на підтримку його з боку ISO), фіксує, що "процес - це серія дій, що приводить до результату", не визначаючи поняття "дія" і "результат". ISO 9000 говорить точніше: "Процес - будь-яка діяльність, в якій використовуються ресурси для перетворення входів у виходи". Останнє, мабуть, слід вважати загальновизнаним визначенням процесу, формалізація якого для опису систем процесів представлена в так званій IDEF-технології. Відзначимо, що наше розуміння дії ширше за процес. Воно включає і інші елементи, що звичайно розглядаються неформально, а значить, недостатньо регламентовано. Проте з точністю до цієї відмінності в термінології базові положення з того, що обговорюється і стандартизується у області менеджменту проектів, можна без особливих зусиль перенести і на поняття системи проектних дій. Так, PMBOK пропонує оперувати поняттям групи процесів, визначаючи для проектів такі групи:

- Процеси ініціації - починають проект, визначають для нього початкові ресурси і матеріали, цілі і елементи решти дії. По суті, це є початком передпроектної дії.
- Процеси планування - визначають, якими способами (методами) передбачається організувати проектну діяльність, на які терміни і ресурси розраховується проект, на які вкладені процеси і етапи він розбивається і як передбачається перевіряти їх результати.
- Процеси виконання - поетапне виконання запланованих процесів в конкретних умовах.
- Процеси контролю - перевірка ходу процесів і результатів, що відповідає запланованій методиці контролю.
- Процеси завершення - завершується проект.

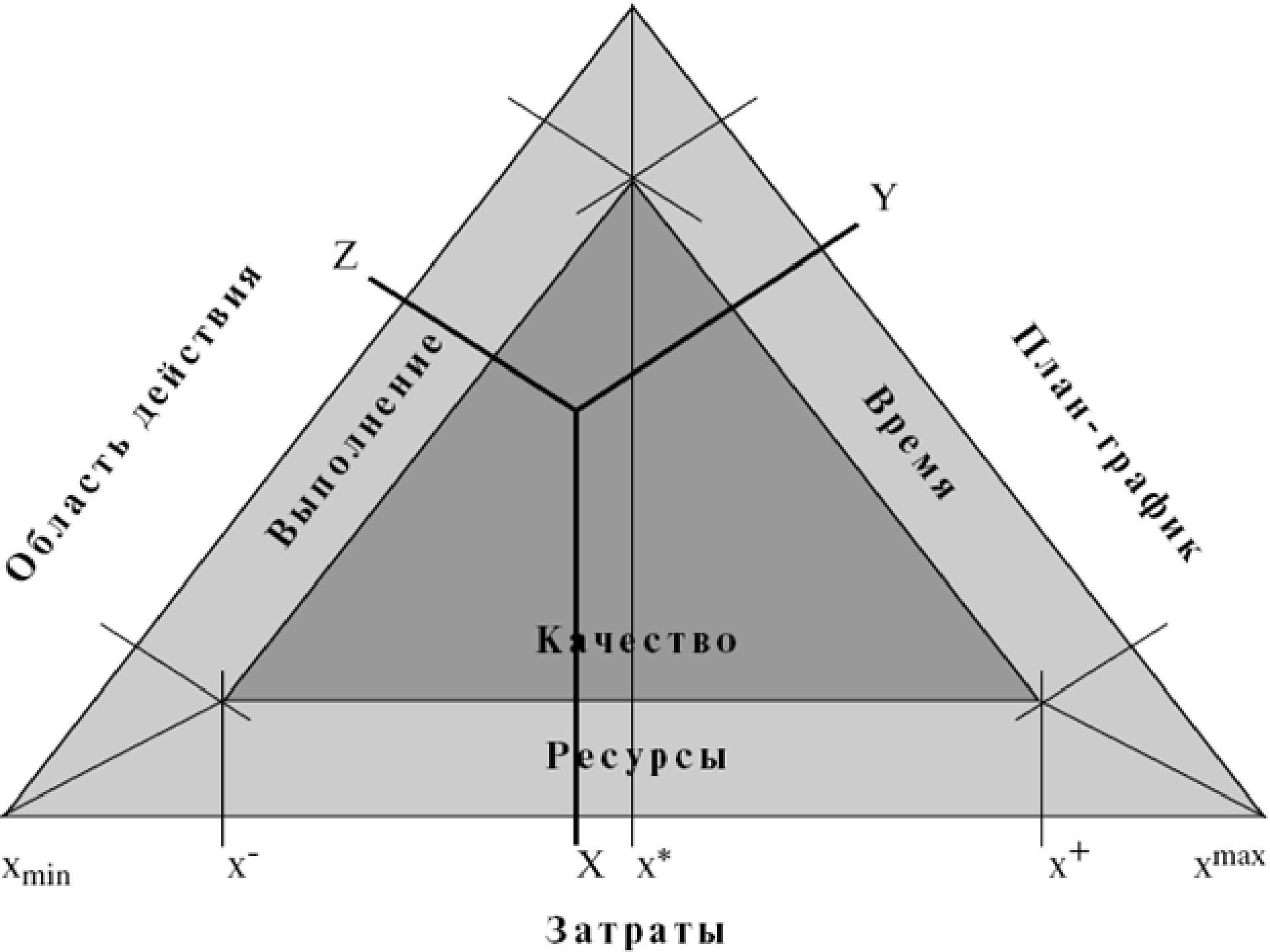

Процессы инициации
Процессы планирования
Процессы контроля
Процессы исполнения
Процессы завершения

Менеджмент в системі дій проекту

Відділення істотного від неістотного - важливий аспект формування системи як об'єкту управління. Багато в чому це мистецтво менеджера, оскільки чітких критеріїв для нього не існує. Як універсальна рекомендація – орієнтуватися на мету найзагальнішого процесу і на використання ролевого складу необхідних виконавців, який визначається змістом проектних робіт, з подальшою побудовою проекції реальних співробітників на ролі.

Охарактеризуємо діяльність менеджера і можливі його дії як складові проектувальної системи дій. Її метою є спрямування виконання всіх інших видів проектних дій так, щоб вони в сукупності просували проект до виконання робіт, що задаються поза системою, в умовах обмежень за часом, фінансами і якістю. Отже, діяльність менеджера полягає в досягненні цілей дії, що задає проект в цілому. Ця діяльність поєднується із зовнішньою по відношенню до робіт проекту діяльністю менеджера, яка полягає в узгодженні параметрів проекту: об'єм робіт, терміни, необхідні фінанси.

Таким чином, менеджмент проекту розглядається як забезпечення поставок продукту, розробка якого вимагає виконання певного об'єму робіт (область дії) , що укладається в задані рамки часу і задовольняє прийнятному рівню якості, із залученням витрат, що не виходять за певні межі. Це широко відомий "трикутник менеджменту проектів“.


Область действия
Z
Выполнение
Y
План-график
Время
Качество
Ресурсы
x_{min}
x^-
X
x^*
x^+
x^{max}
Затраты

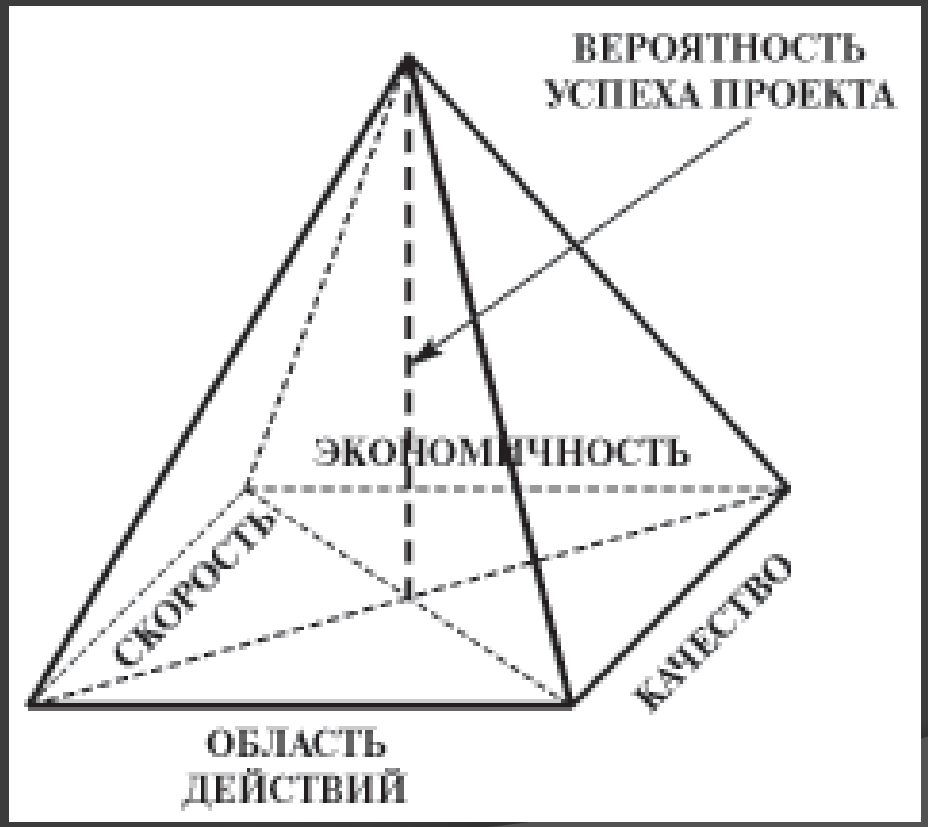
Потрібно врівноважувати продуктивність робіт проекту (область дії), час (план-графік поставок) і витрати ресурсів (витрати), задовольняючи вимогам якості. З різних причин дотримувати баланс по всіх параметрах найчастіше не вдається, а тому менеджер змушений вибирати первинною метою тільки один або два параметри, ставлячи інші в залежність від них. У літературі можна знайти безліч варіацій на тему трикутника менеджменту проектів, або, як його ще називають, "залізного трикутника". Всі вони обумовлені прагненням до того, щоб враховувати в моделях різні аспекти процесу, і в першу чергу змінні, якими можна оперувати, взаємозалежності змінних і обмеження на їх значення. Звідси з'являються варіанти змінних, пов’язаних трикутником, і різні інтерпретації діаграм.

Досить часто як значення змінних інтерпретуються довжини сторін трикутника менеджменту, а взаємозалежність задається як інваріант трикутника (наприклад, площа), який для конкретного проекту і умов його виконання даною командою залишається незмінним. В цьому випадку вибір значень двох змінних, тобто довжин відповідних сторін вимагає обчислення третьої подібно тому, як це робилося в інтерпретації з відновленням перпендикулярів.

У традиційному трикутнику менеджменту якість продуктів і процесу виконання програмного проекту або не відображається зовсім, або задається за допомогою визначення деякої внутрішньої області трикутника, наприклад як в наведеній вище моделі. Це вкладений трикутник, круг (як пропонується в компанії Microsoft) та ін.

М. Відеман пропонує розглядати якість (QUALITY) як ще одну рівноправну змінну . В результаті замість трикутника виходить зірка з чотирма променями . Таке уявлення не дозволяє будувати перпендикуляри і ускладнює вибір інваріанта. Проте в стандарті PMBOK в 1987 році воно було прийняте . У 1991 році Відеман в роботі, виконаній в рамках PMI, приводить більш строгу інтерпретацію чотирикутної зірки, але і вона не так зрозуміла, як традиційні для початкового трикутника трактування. Як виявилося, простоти інтерпретації можна досягнути, якщо скористатися третім виміром. В результаті зберігається можливість і побудови перпендикулярів, і трактування площі - тепер об'єму - як інваріант.

Ідея третього виміру досить плідна. Зокрема, нею скористався Дж. Марасько для віддзеркалення в моделі ще однієї змінної: вірогідність успіху проекту (див. мал. 4.6). Автор вибудовує "проектну піраміду“, в основі якої лежить чотирикутник із сторонами Швидкість (Speed), Економічність (Frugality), Якість (Quality) і Область дій (Scope), а висотою є вірогідність успіху (Probability of Success - фактично використовується не вірогідність, а деяка величина, з нею пов'язана, яка дозволяє застосувати простий інваріант). Ці п'ять характеристик проекту розглядаються як змінні, значення яких пов'язані інваріантом для наявних умов розвитку проекту силами даної команди. Як інваріант вибирається об'єм піраміди.



Операційні маршрути і траєкторії дії

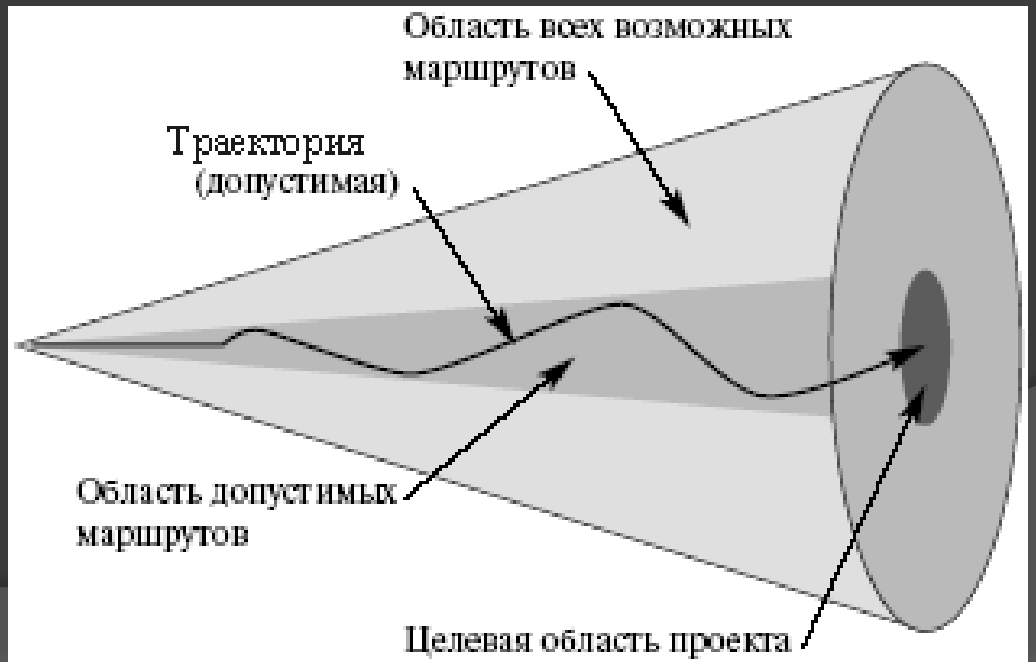
Оскільки в діяльності суб'єкта найчастіше є можливість вибору варіантів продовження, ми говоримо про **операційні маршрути дії як про можливі послідовності дій**, що характеризують кожен момент виконання дії. Маршрути можуть бути довільними і залежать тільки від запропонованих суб'єкту варіантів. Реалізований при виконанні дії операційний маршрут називається траєкторією дії.

Маршрути і траєкторії задають динамічний аспект дії. Проте без прив'язки до цілей і результатів ці поняття несуть мало змісту. При проходженні операційного маршруту продукуються якісь результати.

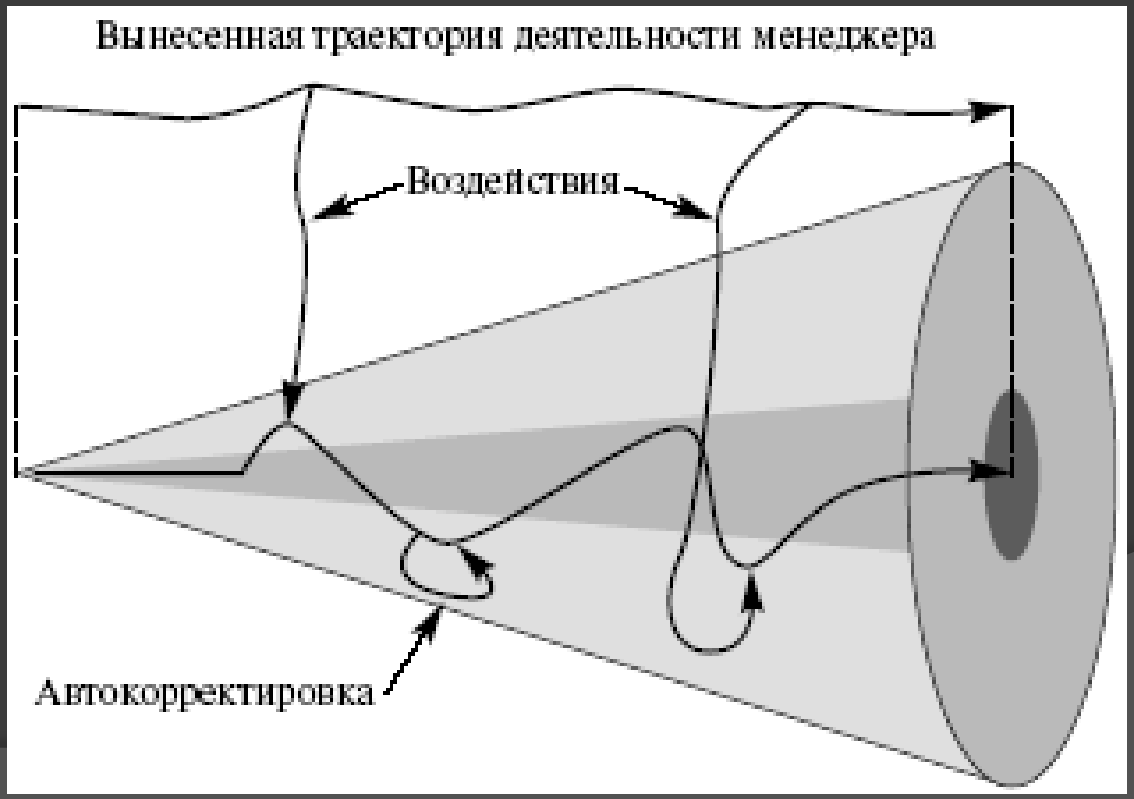
Терміни "операційний маршрут" і "траєкторія дії" завжди розуміються як послідовності дій, що властиві тій або іншій ролі. Це означає, що маршрут може виконувати індивідуум, з призначеною роллю, з якою пов'язаний маршрут, виконуваний в ті моменти, коли вимагається виконати відповідну виробничу функцію. Проте індивідуальні особливості співробітника впливають на те, які з послідовностей дій здійсненні, які з них реалізуються. Як наслідок, у будь-який момент розвитку проекту вибір операційних маршрутів, які належить проходити розробникам, індивідуалізується. Обумовлений обставинами, що виникають при розвитку проекту, цей вибір найчастіше не може бути зафіксований наперед. Управління проектом полягає в тому, щоб вибудовувати конкретні цілеспрямовані операційні маршрути, з одного боку, а з іншого - перевіряти, якою мірою результати траєкторій, реалізованих для цих маршрутів виконавцями, відповідають наперед встановленим цілям виконання функцій. Корисність прив'язки операційних маршрутів до ролей сприяє дотриманню принципу розділення ролей: якщо співробітнику доручається декілька ролей, то він завжди повинен знати, яку з них виконує в даний момент. Крім того, прив'язка дозволяє явно вказати умови, в яких принципово здійснено виконання функції, задавати ресурсні потреби, об'єктивніше визначати реалістичні терміни можливого отримання результатів. В той же час можна говорити і про індивідуалізацію маршруту, враховуючи чинники, які визначають можливі траєкторії для конкретного індивідуума. Якщо можна говорити про двоякість операційних маршрутів, то траєкторія - це атрибут виконання маршруту індивідуумом; до ролі, йому призначеної, вона має лише непряме відношення.

Методологічні стратегії

Якість планування розвитку проекту залежить і від того, як точно вдається встановити для кожного моменту операційні маршрути, оптимальні для досягнення цілей проекту (або виробничої функції). При такому розгляді методика (точніше, метод) – це здатність підтримати проходження операційного маршруту за допомогою певних розпоряджень, угод, рекомендацій і регламентів, а також, можливо, за допомогою інструментів. Методологія – це, по-перше, стратегія, призначена для побудови і виконання системи дій, тобто угоди про те, яким чином повинна розвиватися система, а по-друге, набір методів, засобів і інструментів, які узгоджені зі стратегією. Проект в цілому можна представити у вигляді конуса операційних маршрутів, центр якого відповідає задуму, а основа - множині всіх варіантів завершення проекту. На основі конуса виділяється цільова область - варіанти завершення проекту, що відповідають його цілям. Траєкторії, які ведуть в цільову область, є допустимими.



Якщо проект некерований, то ймовірність того, що його траєкторія виявиться допустимою, досить невисока - дуже різними можуть бути дії, які спонтанно формуються у виконавців. З цієї причини вирізняється діяльність менеджера, одна з цілей якої – не допустити відхилення траєкторій діяльності виконавців від цільової області проекту. Для цього у нього мають бути засоби, що дозволяють виявляти відхилення, і інструменти впливу, призначені для коректування відхилень. Так, саме за принципом виявлення відхилень і швидкого коректування будується робота менеджера в рамках екстремального програмування.



Визначення етапів проекту: послідовний розвиток проекту

Для реалізації загальної глобальної задачі проекту вона розбивається на підзадачі, що виконуються послідовно. Кожне виконання приводить до результатів, що використовуються наступною задачею. Зокрема, воно дає можливість уточнити постановку задачі і коло дій, які доцільно активізувати. Таким чином, операційні маршрути проекту в цілому розбиваються на послідовність етапів, кожен зі своєю локальною метою.

Постановка задачі кожного етапу характеризується:

суб'єктом-виконавцем;

термінами виконання задачі;

виділеними ресурсами;

засобами, інструментами і методами виконання задач;

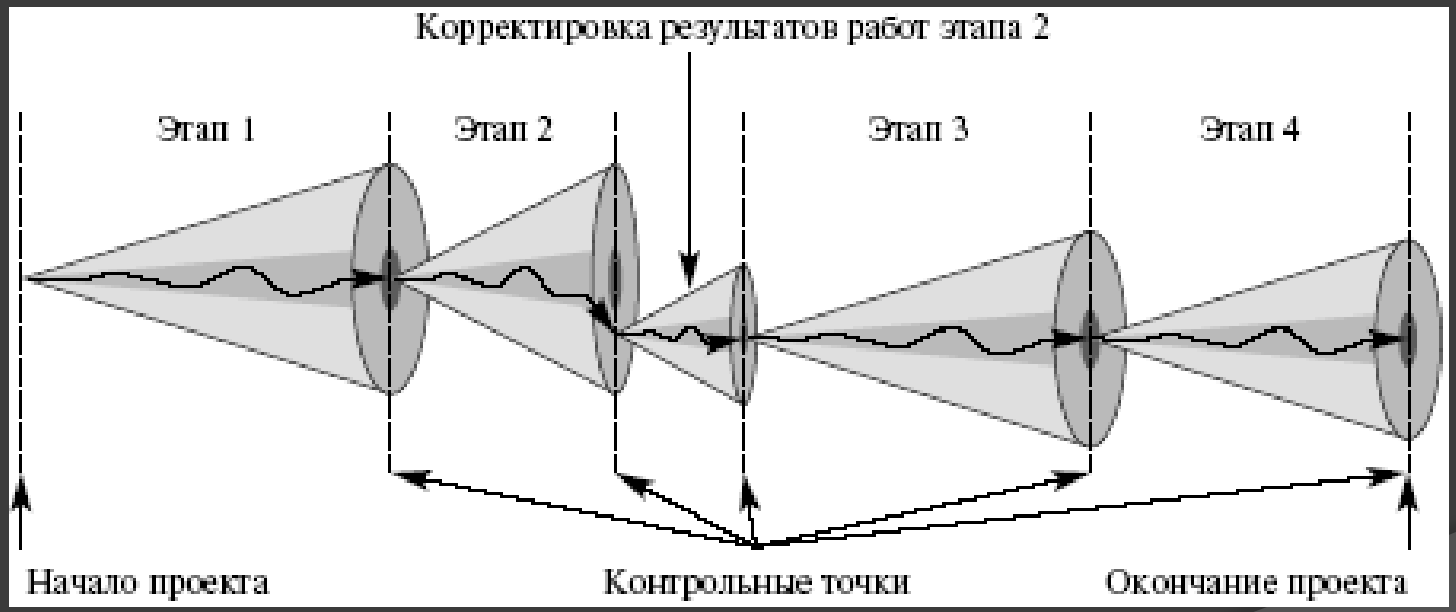
контрольними заходами, що дозволяють впевнитися, що задачі етапу виконані.

До кінця етапу приписуються так звані контрольні точки, призначені для підведення локальних підсумків і дозволу переходити до робіт наступного етапу. Інструментами діяльності в контрольних точках є контрольні заходи. Якщо з'ясовується, що завдання етапу може бути невиконане, тобто траєкторія, що реалізує дію етапу, вийшла з області допустимості, виконуються заходи з метою повернення траєкторії в цю область. Зокрема, використовуються схеми з'ясування відхилень і коректування траєкторії. Контрольна точка – це своєрідний бар'єр, який треба подолати, щоб перейти до наступного етапу.

Продовження робіт після заходів в контрольних точках залежить від того, в якому стані опиниться проект. Стан визначає і зміст робіт чергового етапу, і навіть розбиття наступних робіт на етапи. Так, якщо задача пройденого етапу включала з'ясування ринкової ситуації, то результатом її виконання може виявитися те, що слід віддати перевагу придбанню продукту, а не його розробці. Як наслідок, наступні роботи по досягненню кінцевої мети проекту повинні змінити характер: виготовлення повинне бути замінене закупівлею і поставкою готового виробу. Інші варіанти результату виконання цієї задачі - форсування робіт, додаткові інвестиції (наприклад, в рекламу) тощо.

Якщо стан робіт етапу залишається незадовільним, то він, у свою чергу, розбивається на локальні етапи, роботи або завдання, що допускають відносне автономне управління тим або іншим способом, зокрема шляхом з'ясування відхилень і коректування, а також за допомогою додаткових контрольних точок. При відповідному ресурсному забезпеченні проекту, можливостях виконавців роботи можуть виконуватися паралельно, а для синхронізації встановлюються додаткові контрольні точки. Розпаралелювання може підтримуватися діяльністю менеджерів за напрямами.

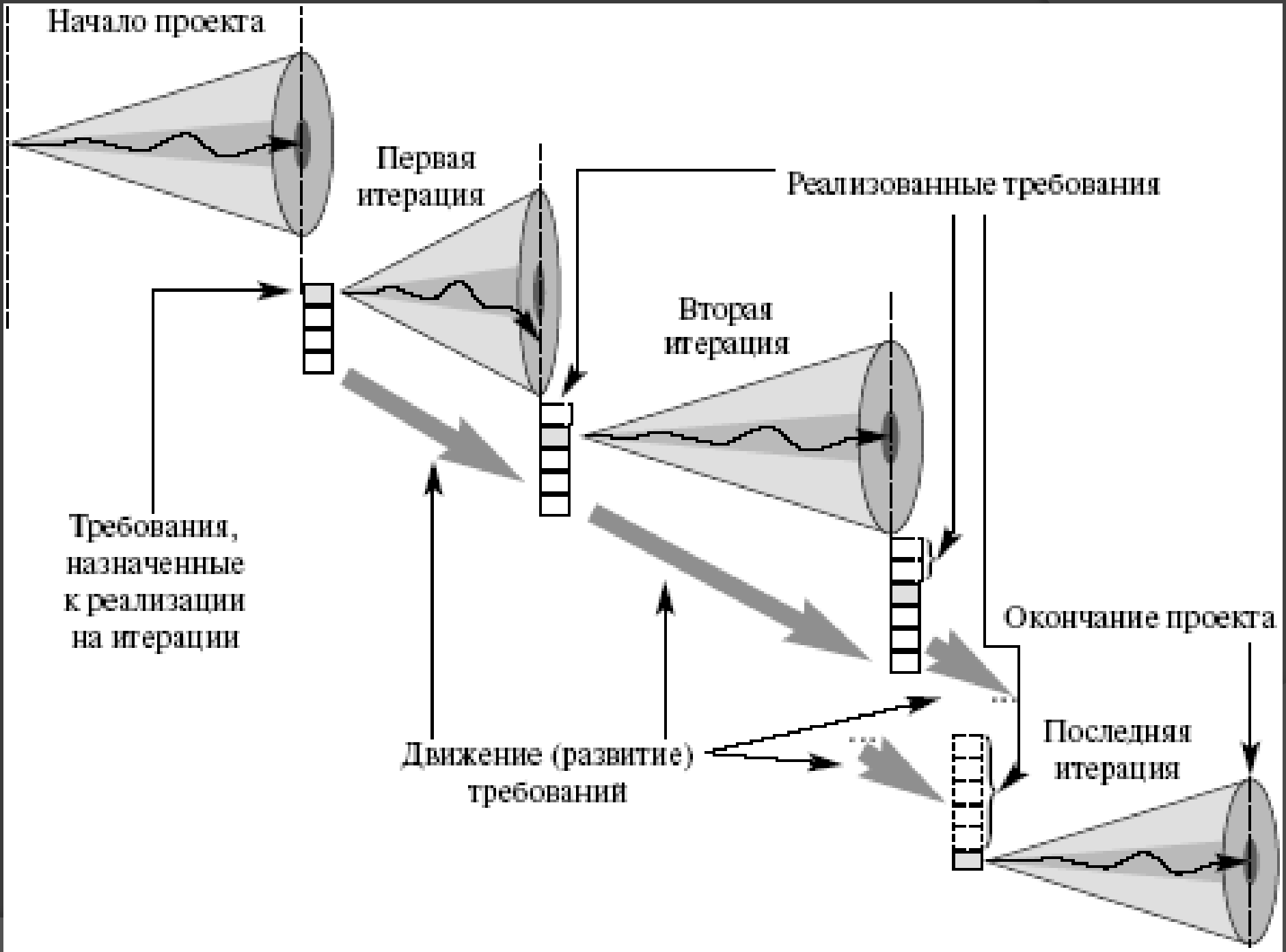
Стратегія визначення етапів одержала назву послідовного розвитку проектів. Це історично перша стратегія, якої дотримувалися при розробці програмних проектів, а тому для неї існує багато методик і їх варіантів.



Звуження поточної задачі проекту: ітеративне нарощування можливостей

Зменшення об'єму конуса операційних маршрутів можна досягнути шляхом виділення із загальної задачі проекту таких її частин, які стають прийнятними (осяжними) для управління. На відміну від попередньої стратегії, тут як перша так і наступні виділені задачі частково вирішують проблеми користувача так, що кожного разу будується програмний виріб, який може застосовуватися на практиці, хай навіть не до кінця задовольняючи потреби в цілому. Таким чином, в даній стратегії замість підзадач виділяються вимоги, які планується реалізовувати. Із сукупності вимог, що визначає проект, і, можливо, не повністю сформованої до початку робіт (так найчастіше і буває), виділяється перша порція, потім наступна, які послідовно, від релізу до релізу наближають програмний виріб до стану, що задовольняє як задуму проекту, так і потребі користувачів.

Кожне з часткових виконань називається ітерацією проекту, а стратегія такої розробки - ітеративним розвитком проекту або ітеративним нарощуванням можливостей конструйованої системи. В результаті виконання однієї ітерації будується довершений програмний виріб, але, взагалі кажучи, залежний від результатів попередніх ітерацій. Принципово, що реліз ітерації ніколи не скасовує користувацьких можливостей, досягнутих на попередніх ітераціях.


Начало проекта
Первая итерация
Реализованные требования
Вторая итерация
Требования, назначенные к реализации на итерации
Окончание проекта
Последняя итерация
Движение (развитие) требований

Прямокутники, намальовані біля основ конусів операційних маршрутів ітерацій, представляють вимоги: заливкою виділений блок вимог, призначених для реалізації, пунктиром - відпрацьовані вимоги. Кількість прямокутників при переході від ітерації до ітерації росте, що відображає надходження нових вимог в ході розвитку проекту.

При розгляді схеми ітеративного нарощування можливостей системи легко помітити, що конуси ітерацій можна, принаймні наочно, представити як конуси етапів. А чи не можна у зв'язку з цим вважати ітеративну стратегію окремим випадком послідовної схеми визначення етапів, в якому кожен етап є ітерацією? Відповідь на це питання найчастіше негативна. Річ у тому, що розподіл діяльності, який ми назвали стратегією визначення етапів, припускає мету виконання, пов'язану з використанням одержаних результатів на наступному етапі. А ця умова для ітеративного нарощування не завжди виконується. Але якщо крім того, що в ході ітерації готується реліз системи, вона припускає побудову бази для реалізації наступної ітерації, то розгляд ітерацій як етапів проекту достатньо припустимий. Проте залишається ще одна відмінність: при поетапному розподілі проекту зазвичай цілі і зміст всіх етапів визначаються на ранніх стадіях, тоді як ітеративне нарощування не пов'язане з цим припущенням і навіть кількість ітерацій, не кажучи вже про їх зміст, не фіксується наперед.

Важливо відзначити, що стратегія ітеративного нарощування не фіксує методику роботи в рамках ітерації. Виконання вимоги осяжності задач ітерації дає підставу для того, щоб розробку ітерації підпорядковувати будь-якій стратегії. За рахунок звуження задачі проекту розробники можуть ослабити вимогу осяжності і вирішувати задачу ітерації як розробку самостійного проекту, наприклад підпорядкувати його послідовній стратегії, і це звичайний шлях реалізації.

Жорсткі і гнучкі стратегії в методологіях програмування

Визначаючи стратегії послідовного і ітеративного розвитку проектів, ми виходили з того, що контроль діяльності проекту - задача, що вимагає спеціальних організаційних підходів. Загальною метою є перетворення діяльності зі створення програмного продукту у впорядкований процес, в рамках якого розвиток проекту можна зробити прогнозованішим і ефективнішим. Для цього, зазвичай, створюється детальний опис процесу розробки системи, особливе місце в якому займає планування. Інакше кажучи, створюється метод, за допомогою якого передбачається конструювання системи. Вдалі методи узагальнюються, в результаті досвід їх застосування перетворюється на методику, або, як зараз прийнято говорити, методологію. Нерідко при такому формуванні методики-методології фіксуються часткові і випадкові рішення, що виявилися корисними через специфіку вдалих проектів. Разом зі всім корисним ці рішення оголошуються обов'язковими, стандартними, їх вимушені застосовувати і тоді, коли початкові передумови втрачені. Щоб дотримуватись таких методологій, доводиться виконувати безліч різних розпоряджень, що уповільнює темп основних робіт програмістів. Тому їх називають жорсткими або монументальними.

Жорсткі методології привабливі для замовників програмних проектів, які завжди можуть перевірити, чи дійсно процес розробки впорядкований і результати відповідають планам. Природно пов'язувати цю можливість з організацією, яка одержує замовлення і забезпечує його менеджмент, оскільки в організації можливий адміністративний контроль.

На сьогодні є розроблено стандарти зрілості процесів розробки програмного забезпечення в організаціях. Серед них найрозвиненішою є пропозиція Інституту програмної інженерії при Університеті Карнегі-Меллона – так звана модель SW-CMM (Capability Maturity Model for Software). Цю модель можна вважати загальноприйнятою, оскільки на неї найчастіше орієнтуються замовники, зокрема з Міністерства оборони США, пропонуючи проекти тільки тим організаціям, які сертифіковані на рівнях зрілості 4 і 5. Модель CMM була сформована за істотного впливу практики військових замовлень з їх жорсткою процедурою контролю і звітності. Пропозиції CMM для визначення зрілості організації спираються на те, які процедури управління програмними проектами, відстежування їх розвитку і інші менеджерські методи прийняті як фірмовий стандарт. Методології програмування, побудовані на базі цих пропозицій, часто розглядають як еталон жорсткості.

На противагу жорстким методологіям останнім часом сформувався компромісний підхід, методології якого об'єднані загальним терміном agile development. На українську мову його перекладають як швидкий розвиток, гнучкі методології і навіть "спритні технології". У новому підході намагаються надати можливість полегшеної організованої роботи колективів програмістів, коли перевантаженість стандартизованого процесу перешкоджає ефективності. Вони претендують на те, що їх застосування можливе навіть коли не вдається достатньо точно представити проект при його зародженні, тобто в досить типовій ситуації невизначеності реальної користувацької потреби. У цих випадках пропонується залучення замовника до формування задач і коректування припущень протягом всього розвитку проекту. З його допомогою намагаються виявляти найточніше і без зайвих бюрократичних процедур актуальні потреби користувачів, що склалися на даний момент. В результаті з'являється можливість вказати саме ті вимоги, реалізація яких необхідна і допускає максимально короткі терміни випуску релізу.

Всі методології швидкого розвитку орієнтуються на стратегію ітеративного нарощування можливостей системи, але з частковою відмовою від постулату незмінності апріорної архітектури (як наслідок, не тільки допускається, але навіть передбачається, що архітектура системи, а значить, і програмний код мінятимуться при переході від релізу до релізу). Всі вони надають розробникам значно більшу свободу, ніж, наприклад, вимоги стандартів CMM. Але не слід вважати, що цей підхід повністю скасовує жорсткі методології. Як вказують Боем і Тюрнер, необхідний баланс між свободою швидких методологій і дисципліною.

Детальніше охарактеризувати всі методології швидкого розвитку загалом не є можливим - дуже відрізняються їх початкові принципи, дуже залежать вони від специфіки колективів, що займаються розробкою програмного забезпечення, від умов, що часто стихійно складаються, від звичок. На стратегічному рівні їх дійсно об'єднує так званий Agile Manifesto, прийнятий ентузіастами в лютому 2001 року в містечку Сноуберд (США), який зводиться до чотирьох постулатів:

- **індивідууми і їх взаємодія важливіші за процеси і інструменти;**
- **працездатне програмне забезпечення важливіше за детальну документацію;**
- **співпраця із замовником важливіша за укладення контракту;**
- **готовність до змін важливіша за дотримання плану.**

Жорсткі методології	Швидкі методології
Орієнтація на передбачені процеси розробки програмного забезпечення з чітко визначеною метою	Усвідомлення того, що процеси розробки програмного забезпечення загалом непередбачувані
Розпізнавання ситуацій і застосування готових методів	Розпізнавання ситуацій і конструювання методів для роботи в них
Планування, в якому визначаються етапи з об'ємом робіт, ресурсами, термінами і рівнем якості робіт	Дотримання балансу між параметрами проекту: об'єм робіт, ресурси, терміни і рівень якості робіт
Замовник - зовнішній по відношенню до проекту суб'єкт, що впливає на розробку тільки через надання ресурсів і контроль результатів, зокрема по поетапних термінах виконання проекту	Замовник (його представник) - член команди розробників, наділений правом впливати на розробку; його головною метою є відстежування актуальності вирішуваних задач
Ролевий розподіл праці працівників проекту	Спільна діяльність співробітників і деперсоніфікована відповідальність
Дисципліна і підпорядкування	Самодисципліна і співпраця
Знеособлений процес, виконавці якого визначаються тільки по кваліфікаційних вимогах	Процес, що максимально враховує індивідуальні якості виконавців

Таким чином, вже на стратегічному рівні можна розмежувати сфери адекватного використання двох підходів. Це розмежування безпосередньо виходить з їх зіставлення. Зрозуміло, що у багатьох випадках можливі обидва варіанти організації проектної діяльності. Проте пристосованість жорстких методологій до імперативності вказує на можливість і виправданість впровадження технологій, тоді як творчому процесу швидкого підходу технології можуть сприяти лише як набір інструментів, далеко не завжди достатньо повний.

Технологія будь-якої діяльності – це середовище підтримки діяльності, що володіє засобами і інструментами, а також методами їх застосування. Неухильне дотримання цих методів будь-яким виконавцем з певною кваліфікацією гарантовано забезпечить виробництво, тобто отримання з наданих ресурсів і матеріалів продукту-результату, що відповідає цілям, в необхідному об'ємі, за відомий час і з прийнятним рівнем якості.

Коли говорять про детерміновану діяльність, зазвичай під цим розуміють відсутність непередбачених, а значить, і не забезпечених методами ситуацій. У контексті поняття технології, крім того, слід виділити і інший аспект детермінізму: зв'язку методів і цілей, якому підпорядковані засоби і інструменти. Коли детермінізм порушується, тобто доводиться вирішувати проблеми вибору, або стикатися з непередбаченими ситуаціями, наступає кінець технології, і розробка вступає в область ремісничого виробництва, яке, мабуть привабливіше (за рахунок творчої складової), і цілком може призводити до підвищення якості, але не гарантує ні результату, ні дотримання термінів.

Розробники жорстких методологій старанно і послідовно виганяли недетермінізм з виробництва програмного забезпечення. Там, де це не вдавалося, вони виставляли своєрідні огорожі: з одного боку, це типові прийоми, вказівки і рекомендації, а з іншою - вимірювання процесу і результатів, контрольні заходи і багато іншого. Але, не зважаючи на появу різноманітних методик, засобів і інструментів підтримки управління, які змушені застосовувати менеджери (що якраз і стало предметом критики прихильників полегшених методологій), особливого успіху досягнути не вдалося. Причиною тому – і вже не раз згадувана мінливість вимог, і властива людській природі схильність припускатися помилок на всіх рівнях роботи.

Захищати ситуації від можливих помилок, допомагати їх пошуку, використовувати засоби автоматизації, здатні ліквідовувати багато рутинних операцій, ми навчилися, а тому можна говорити про деяку технологізацію.