Лекція 3 Принципи встановлення вимог розробки системи

3.1 Принципи встановлення вимог

Встановлення вимог - перший етап життєвого циклу розробки системи. Система, що підлягає розробці, визначається за допомогою виду діяльності, котрий отримав назву системного планування. Мета встановлення охоронних вимог полягає в тому, щоб дати розгорнуте визначення функціональних - а також не функціональних - вимог, які учасники проекту очікують затвердити в реалізованої і розгортається системі.

Вимоги визначають послуги, очікувані від системи (формулювання сервісів) і обмеження, яким система повинна підкорятися (формулювання обмежень). Ці формулювання сервісів можна об'єднати в кілька груп: одна з груп що описує кість кордону системи, друга - необхідні бізнесфункції (функціональні вимоги), а третя - необхідні структури даних (вимоги до даних). Формулювання обмежень можна класифікувати відповідно до різними категоріями обмежень, що накладаються на систему, такі як необхідну "враження і відчуття від використання програми", продуктивність, безпеку і т.д.

Вимоги необхідно отримати від замовників (користувачів і власників сис теми). Цей вид діяльності, званий виявленням вимог (requirements elicita tion), здійснюється аналітиком бізнес-процеси (або системним аналітиком). Для цього підходять різні методи, починаючи з традиційних інтерв'ю з замов кому і закінчуючи (при необхідності) побудовою програмного прототипу, з по міццю якого розкриваються додаткові вимоги.

Зібрані вимоги повинні бути піддані ретельному аналізу для виявила лення в них повторів і суперечностей. Це незмінно призводить до перегляду охоронних вимог і їх повторному погодженню з замовниками.

Вимоги, задовільні з точки зору замовника, документуються. При цьому вимогам дають визначення, класифікують, нумерують і привласнюють їм пріоритети. Структура документа, що описує вимоги, відповідає шаблона (template), заданої в організації для цієї мети.

Хоча документ, що фіксує вимоги, носить в значній мірі описатель ний характер, цілком можливо включити в нього високорівневу схематичну біз несмодель. Як правило, бізнесмодель складається з моделі кордонів системи, моделі бізнес неспрецедентов і моделі бізнескласса.

Вимоги користувача подібні рухомій цілі. Щоб впоратися з зрад стійкими проти вимог, необхідно вміти управляти змінами. Управління вимогами включає такі види діяльності як оцінка впливу змін одних вимог на інші, а також на незмінну частину системи.

3.2. Виявлення вимог

Бізнес Аналітик виявляє вимоги до системи за допомогою консультацій. До участі в консультаціях залучаються замовники і експерти в проблемній області. У деяких випадках БІЗНЕСАНАЛІТИКА володіє достатнім досвідом у проблемної області, і допомога експерта може бути зайвими. В цьому випадку БІЗНЕСАНАЛІТИКА є різновидом Експерта проблемної області, що відображено в моделі, показаної на рис. 3.1, за допомогою відносини узагальнення. (Слід, однак, мати на увазі, що рис. 3.1 - це не модель прецедентів: нотація для прецедентів використовується тут тільки для зручності.)

Вимоги, виявлені за допомогою експерта проблемної області, становлять основу знань про проблемою області. Вони фіксують широко визнані, чи не залежать від часу бізнес-правила, застосовні до більшості організацій і систем. Вимоги, виявлені в ході консультацій з замовниками, виражаються в сценаріях прецедентів. Вони виходять за рамки базових знань про проблемну область і фіксують унікальні риси організації - спосіб ведення бізнесу "тут і зараз" Чи бо побажання щодо того, як слід вести бізнес.

Завдання БІЗНЕСАНАЛІТИКА полягає в тому, щоб об'єднати два набору вимог в бізнес-моделі. Як показано на рис. 3.1, Бізнес-модель містить Модель бізнес-класів та Модель бізнес-прецедентів. Модель бізнес-класів перед ставлять собою діаграму класів верхнього рівня, яка ідентифікує і зв'язує між собою бізнесоб'екти. Модель бізнес-прецедентів - це діаграма прецедентів верхнього рівня, яка ідентифікує основні функціональні будівельні блоки системи.

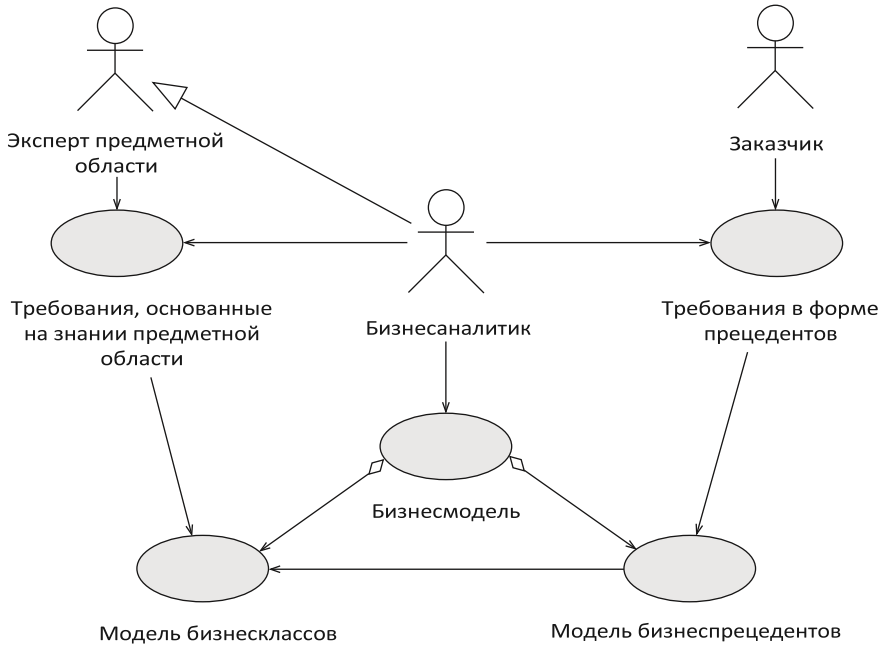

Рис. 3.1. Взаємні впливу моделей, характерні для процесу визначення вимог

У загальному випадку, класи проблемної області (бізнесоб'екти) не повинні виводи дивіться з прецедентів [75]. Однак, на практиці правильність моделі бізнес класів повинна підтверджуватися за допомогою порівняння з

Моделлю бізнеспрецедентов. Це порівняння, як правило, призводить до деякої налаштуванні або розширенню Моделі бізнес-класів.

Дотримуючись Хоффер (Hoffer) [37] ми розрізняємо традиційні і сучасні мето ди виявлення фактів і збору інформації.

3.2.1. Традиційні методи виявлення вимог

До традиційних методів виявлення вимог відносяться використання інтерв'ю і анкети, спостереження і вивчення ділових документів. Це прості і економічних методи.

Ефективність традиційних методів обернено пропорційна ризику проекту. Високий ризик означає, що систему важко реалізувати - не цілком ясні навіть узагальнені вимоги. Для таких проектів традиційних методів навряд чи буде достатньо.

3.2.1.1. Інтерв'ювання замовників і експертів в проблемної області

Використання інтерв'ю являє собою основний метод виявлення фактів і збору інформації. Більшість інтерв'ю проводяться з замовниками. Інтерв'ю з замовниками дозволяють виявити здебільшого "прецедентні" вимоги, тобто вимоги випливають з "прецедентів" (рис. 3.1). Якщо БІЗНЕСАНАЛІТИКА не володіючи достатнім досвідом в проблемній області, можна також взяти інтерв'ю у відповідних експертів.

Інтерв'ю з експертами в прикладної області найчастіше є про сто процес передачі знань - заняття з навчання БІЗНЕСАНАЛІТИКА. Інтерв'ю з замовниками відрізняє велика складність [46], [81].

Замовники можуть мати дуже туманне уявлення про свої вимоги. Вони можуть не бажати співпрацювати або не вміють висловлювати свої вимоги в зрозумілій формі. Вони також можуть запитувати реалізацію вимог, які перевершують бюджет проекту або реалізовуються. І нарешті, досить імовірно, що вимоги, ис ходять від різних груп користувачів, можуть виявитися суперечливими.

Існують два основних типи інтерв'ю: структуроване (формальне) і не структуроване (неформальне). Структуроване інтерв'ю готується заздалегідь, відрізняється ясністю постановки питань, а багато питань можуть бути задані заздалегідь. Заздалегідь сформульовані питання можна розділити на дві категорії: питання с від критим безліччю відповідей (openended question) (відповіді на цю категорію питань зарані не готуються) і питання с замкнутим безліччю відповідей (closedended question) (відповіді на цю категорію питань можна вибрати зі списку підготовлених відповідей).

Структурованого інтерв'ю повинно супроводжувати неструктуроване інтерв'ю. Неструктуроване інтерв'ю більше нагадує неформальну зустріч, до торою не властиві заготовлені про запас питання або заздалегідь поставлені це чи. Мета неструктурованого інтерв'ю - спонукати замовника до того, щоб він

поділився своїми думками і в процесі бесіди підійшов до вимог, яких бізнес-аналітик міг і не чекати і, отже, не міг поставити потрібні питання.

Як структуроване, так і неструктуроване інтерв'ю потребують до деяких законів рій відправної точки і контексті для обговорення. Це може бути невеликий документ або записка, відправлена по електронній пошті беруть інтерв'ю перед зустріччю, мета яких - пояснити цілі інтерв'юера або поставити деякі запитання.

Існують три категорії питань, яких, в загальному випадку, необхідно мати за [91].

1. Небеспристрасні питання, в яких інтерв'юер висловлює (прямо чи опосередковано) свою думку з питання ("Чи повинні ми працювати так, як ми працюємо?").
2. Предвзяті питання, аналогічні небезстороннім, але відрізняються від останніх тим, що думка інтерв'юера є, очевидно, тенденційним ("Ви ж не станете цього робити, чи не так?").
3. Наводящі питання, які припускають відповідь в самому питанні ("Ви ж зробите саме так, чи не так?").

Успіх інтерв'ю залежить від багатьох факторів, але чи не головними серед них є навички інтерв'юера в області комунікації і міжособистісного спілкування. Хоча основне завдання інтерв'юера - задавати питання і володіти ситуацією, не менше важлива але в ході бесіди уважно слухати і проявляти терпіння до беруть інтерв'ю так, щоб він відчував себе невимушено. Для збереження хороших особистих від відносин і в розрахунку на позитивний відгук з боку інтерв'юйованого необхідно відправити йому протягом одного-двох днів після інтерв'ю записку, утримуючи короткі підсумки інтерв'ю.

3.2.1.2. Анкетування

Використання анкет або анкетування (questionnaires) - ефективний спосіб збору інформації від багатьох замовників. Зазвичай анкети використовуються в доповнення ня до інтерв'ю, а не замість них. Виняток можуть становити проекти з низьким ризиком, цілі яких ясно окреслені. Для таких проектів зазвичай досить іс користуватися анкети з питаннями, які носять пасивний характер і не відрізняються великою глибиною.

У загальному випадку, анкетування менш продуктивно, ніж використання інтерв'ю, оскільки в питання або можливі відповіді не можна внести додаткову ясність. Анкетування пасивно - це можна розцінювати і як гідність, і як недостатньо струм. Це гідність, оскільки у респондента є час подумати над відповіддю, а анкета може залишитися анонімною. Це недолік, оскільки респонденту нелегко прояснити для себе питання.

Анкета (або опитувальний лист) повинна бути розроблена таким чином, щоб, по можли вості, полегшити відповіді на питання. Зокрема, слід уникати питань з невизначеним ленним безліччю відповідей - більшість питань повинні

ставитися до питань із замкнутим списком відповідей. Подібні питання можуть набувати трьох форм [91].

1. Багатоальтернативні питання (multiplechoice questions). При відповіді на ці питання респондент повинен вказати один або більше відповідей, вибравши їх із прикладеного списку. Крім того, іноді допускаються додаткові коментарі до питань з боку респондентів.
2. Рейтінгові питання (rating questions). При відповіді на цей тип питань респондент повинен висловити свою думку щодо висловленого твердження. Для цього можуть використовуватися такі рейтингові значення, як "абсолютно згоден", "згоден", "ставлюся нейтрально", "не згоден", "абсолютно не згоден" і "не знаю".
3. питання з ранжуванням (ranking questions). Цей тип питань передбачає ранжування відповідей за допомогою привласнення ним послідовних номерів, процентних значень і використання інших засобів упорядкування.

Ретельно продумана, що полегшує відповіді анкета заохочує респондента для того, щоб відразу повернути заповнений документ. Однак, при оцінці результатів анкетування БІЗНЕСАНАЛІТИКА повинен передбачити можливість спотворення інформації через те, що ті люди, які не відповідали на питання, могли б відповісти на них інакше, ніж брали участь в опитуванні [37].

3.2.1.3. Спостереження

У деяких ситуаціях БІЗНЕСАНАЛІТИКА знаходить важким отримати повну ін формацію з використанням інтерв'ю і анкет. У замовника з тих чи інших причи нам може бути відсутнім можливість (або вміння) надати необхідну ін формацію, або він може не мати повного уявлення про бізнес-процеси в ціло. У подібних випадках в якості ефективного методу виявлення фактів може виступати спостереження (observation). Какнікак, кращий спосіб навчитися зав'язувати гал стукіт - поспостерігати за процесом.

Спостереження може виступати в двох формах.

1. Пасивне спостереження (passive observation), в ході якого БІЗНЕСАНАЛІТИКА спостерігає за різними видами ділової діяльності без втручання або прямої участі в них. У деяких випадках для того, щоб спостереження було якомога менше нав'язливим, можна навіть використовувати відеокамери.
2. Активне спостереження (active observation), в ході якого БІЗНЕСАНАЛІТИКА бере участь в діяльності і стає фактично частиною команди.

Щоб результати спостережень були представницькими, спостереження необхідно проводити протягом тривалого періоду часу, в різні часові інтервали і при різній робочому навантаженні (вибіркові періоди). Основна праця пов'язана з наглядом, полягає в тому, що люди, за якими спостерігають, схильні поводитися інакше, ніж у звичайній обстановці. Зокрема, вони прагнуть

працювати відповідно до формальних правил і процедур. Це призводить до спотворення реального стану справ за рахунок приховування "раціональних" прийомів рабо ти - як позитивних, так і негативних. Слід пам'ятати, що "робота по правилам" - це одна з ефективних форм страйкового руху.

3.2.1.4. Вивчення документів і програмних систем

Вивчення документів і програмних систем є неоціненним методом виявлення як вимог типу прецедентів, так і вимог, пов'язаних зі знанням про проблемної області *(див. Розд. 3.2)*. Цей метод використовується завжди, хоча він може стосується тільки окремих сторін системи.

Вимоги, що формулюються у вигляді прецедентів, або прецедентні вимоги (use case requirements) виявляються за допомогою вивчення існуючих організаційних до ментів і системних форм і звітів (якщо, звичайно, для поточної системи існують автоматизовані рішення, що типово для великих організацій). Одним з найбільш корисних способів осягнення суті вимог на основі прецедентів є фіксація дефектів (природно, при їх наявності) та формування запитів на зміну для існуючої системи.

До організаційних документів, що підлягають вивченню, належать: форми ділових документів (по можливості - заповнені), опису робочих процедур, має обов'язки, методичні керівництва, бізнес-план, схеми організаційних них структур, внутріофісну кореспонденція, протоколи нарад, облікові записи, зовнішня кореспонденція, скарги клієнтів і т.д.

Системні форми і звіти, що підлягають вивченню, включають: копії екранів, від подружжя разом з відповідною документацією - системні керівництва по експлутації, призначена для користувача документація, технічна документація, системні моді чи аналізу і проектування і т.д.

Вимоги, засновані на знанні проблемної області, з'ясовуються за допомогою вивчення журналів і підручників, які відносяться до даної сфери. Вивчення патентованих програмних пакетів на зразок ERPсістем (Enterprise Resource Planning Systems - системи планування ресурсів підприємства) також може стати багатим джерелом знань про проблемну область. Отже, відвідування бібліотек і постачальників ПО є частиною процесу виявлення вимог (звичайно, Inter net дозволяє здійснити багато такі "візити", не залишаючи офісу).

3.2.2. Сучасні методи виявлення вимог

До сучасних методів виявлення вимог відноситься використання про програмних прототипів, а також такі методи, як JAD (Joint Application Develop ment - спільна розробка додатків) і RAD (Rapid Application Development - швидка розробка додатків). Ці підходи пропонують більш глибоке проник новение в суть вимог, але за рахунок більшої ціни і зусиль. Однак, долговремен ні вкладення в ці методи можуть окупитися з лишком.

Застосування сучасних методів зазвичай пов'язане з високим проектним ризиком. Існує безліч факторів, що обумовлюють високий ризик проекту. До таких факторів належать неясні цілі, недокументовані процедури, нестабільні вимоги, слабке знання справи користувачами, недосвідчені розробники, недостатньо точна прихильність користувачів розробці і т.д.

3.2.2.1. Прототипування

Прототипування (prototyping) - це найбільш часто використовуваний сучасний метод виявлення вимог. Програмні прототипи конструюються для візуалізації системи або її частини для замовників з метою отримання їх відгуків.

Прототип являє собою демонстраційну систему - "нашвидку і грубо" зроблену робочу модель рішення, яка представляє користувальницький GUI інтерфейс і моделює поведінку системи при ініціюванні користувачем раз особистих подій. Інформаційне наповнення екранів частіше жорстко запрограмованих в програмі прототипу, ніж виходить автоматично з бази даних.

Складність (і зростаючі "апетити" замовників) сучасних GUIінтерфейсов роблять прототипування обов'язковим елементом розробки ПО. Прототипи дозволяють досить непогано оцінити достовірність, і корисність системи до початку її реалізації.

У загальному випадку, прототип - це дуже ефективний спосіб виявлення охоронних вимог, які важко отримати від замовника за допомогою інших засобів. Найчастіше така ситуація зустрічається для систем, які повинні надати в розпорядження користувачів нові бізнес-функції. Подібна ситуація також характерна для випадків суперечливих вимог і наявності проблем в кооперації між замовниками і розробниками.

Існують дві основні різновиди прототипів [46].

1. "Одноразовий" прототип ("throwaway" prototype), який після того, як виявлення вимог завершено, просто відкидається. Розробка "одноразового" прототипу націлена тільки на етап встановлення вимог ЖЦ ПО. Як правило, цей прототип концентрується на найменш зрозумілих вимогах.
2. Еволюційний прототип (evolutionary prototype), який зберігається після виявлення вимог і використовується для створення кінцевого програмного продукту. Еволюційний прототип націлений на прискорення поставок товарів. Як правило, він концентрується на ясно викладених вимогах, так що першу версію продукту можна надати замовнику досить швидко (хоча її функціональні можливості, як правило, неповні).

Додатковим аргументом на користь використання саме "одноразового" прототипа може служити прагнення уникнути ризику "консервації" швидких і

грубих і, як наслідок, неефективних рішень в кінцевому продукті. Однак міць і гнучкість з тимчасових засобів створення ПО послаблюють цей аргумент. Якщо управління проектом здійснюється належним чином, то причини, по якій можна було б позбутися неефективних запропонованих для прототипу рішень, не існує.

3.2.2.2. Спільна розробка додатків (JAD метод)

JAD метод повністю відповідає своїй назві - це спільна розробка додатків (Joint Application Development), що здійснюється в ході одного або декількох нарад із залученням всіх учасників проекту (замовників і розробників) [95]. Хоча ми відносимо JAD-підхід до сучасних методів виявлення вимог, цей метод був вперше введений в кінці 1970х років компанією IBM.

Існує багато різновидів JAD-метода і багато фірм, що пропонують послуги по організації і проведенню JAD-нарад. Проведення JAD-нарад може займати кілька годин, кілька днів або навіть пару тижнів. Кількість учасників не повинно перевищувати 2530 осіб. У нараді бере участь певне коло осіб [37], [91].

1. Ведучий - людина, яка проводить і модерує зустріч (тому іноді його називають модератором). Ця людина повинна володіти винятковими здібностями в області комунікації, не повинен ставитися до числа учасників проекту (крім того, що він є лідером JADсессіі), володіє ґрунтовним знанням проблемної області (проте не обов'язково добре володіє проблемами розробки ПО).
2. Секретар - людина, яка фіксує хід JAD-сесії з використанням комп'ютера. Ця людина повинна вміти швидко вводити текст в комп'ютер і добре володіти питаннями розробки ПО. Для документальної фіксації ходу сесії і розробки первинних моделей рішень секретар може використовувати CASEсредства.
3. Замовник (користувачі або керівники) - це основні учасники, які викладають і обговорюють вимоги, приймають рішення, стверджують проектні цілі і т.д.
4. Разробник - бізнесаналітик і інші члени проектної бригади. Ці учасники більше слухають, що говорять - вони присутні на нараді для того, щоб усвідомити фактичний бік справи і зібрати інформацію, а не позичати цілком увагу інших учасників в процесі наради.

JAD-метод ґрунтується на груповій динаміці. Групові зусилля більш перспективні з точки зору отримання найкращих рішень проблем. Групи сприяють підвищенню продуктивності, швидше навчаються, схильні до більш кваліфікованим висновками, дозволяють виключити багато помилок, приймають ризиковані рівня (іноді це може носити негативний характер!), Концентрують увагу навчаючи на найбільш важливих питаннях, об'єднують людей і т.д.

Якщо JAD сесія проводиться відповідно до правил, можна домогтися хороших результатів. Однак, не слід забувати про попередження: "... компанія Ford Motor в 1950х роках зазнала невдачі, намагаючись вивести на ринок модель Edsel - автомобіль, розроблений комітетом" [95].

3.2.2.3. Швидка розробка додатків (RAD-метод)

Метод швидкої розробки додатків (Rapid Application Development - RAD) - це не що більше, ніж метод виявлення вимог - це цілісний підхід до розробки ПЗ [37]. Як зрозуміло з назви методу, він передбачає швидку поставку системних рішень. Технічна перевага відступає на друге місце в порівнянні з бо ростью поставки.

3.3. Узгодження і перевірка обґрунтованості вимог

Згідно Вуду (Wood) і Сильверу (Silver) [95] технологія RAD поєднує в собі п'ять методів, перерахованих нижче.

1. Еволюційне прототипування.
2. CASE-засоби з можливостями генерації програм і циклічної розробкою з переходом від проектних моделей до програми і назад.
3. Спеціалісти, які володіють розвиненими інструментальними засобами (Specialists with Ad vanced Tools - SWAT) – RAD-бригада розробників. Кращі аналітики, проектувальники і програмісти, яких тільки може залучити організація. Бригада працює в рамках суворого тимчасового режиму і розміщується разом з користувачами.
4. Інтерактивний JAD-метод – JAD-сесія, під час якої секретар замінюється бригадою SWAT, оснащеної CASEсредствамі.
5. Жорсткі тимчасові рамки (timeboxing) - метод управління проектом, який відводить бригаді SWAT фіксований період часу (timebox) для завершення проекту. Цей метод перешкоджає "розповзання рамок проекту"; якщо проект затягується, то рамки рішення звужуються, щоб дати можливість завершити проект вчасно.

Використання RAD-підходу може виявитися привабливим варіантом для багатьох проектів, особливо, для невеликих проектів, які не зачіпають сферу ключових бізнес-процеси організації, і які, таким чином, не ставлять план рішення для інших проектів з розробки ПЗ. Малоймовірно, щоб швидкі рішення були оптимальними або довготривалими для ключових сфер діяльності організації. З використанням RAD-метода пов'язаний ряд проблем; деякі з них перераховані нижче.

1. Несумісний проект GU-Iінтерфейса.

2. Замість загальних рішень, що сприяють багаторазовому використанню ПО, спеціалізовані рішення.

3. Неповна документація.

4. Важке для підтримки і масштабування ПО і т.д.

3.3. Узгодження і перевірка обґрунтованості вимог

Вимоги, отримані від користувачів, можуть дублюватися або суперечать один одному. Деякі вимоги можуть бути неясними або нереальними Інші вимоги можуть залишитися нез'ясованими. З цієї причини перш, ніж вимоги потраплять в документ опису вимог, їх необхідно узгодити.

Насправді узгодження і перевірка обґрунтованості вимог здійснює ся паралельно з виявленням вимог. Після того як вимоги виявлені, вони піддаються певному рівню перевірки. Для всіх сучасних методів ви явища вимог, які пов'язані з так званої "груповою динамікою", це цілком природно. Як би там не було, після того як виявлені вимоги зібрані разом, вони в будь-якому випадку повинні бути піддані ретельному обговоренню і перевірці.

Узгодження і перевірка вимог не можуть бути відокремлені від процесу підготовки документа опису вимог. Зазвичай погодження розбіжностей між вимогами засноване на чорновому варіанті документа. Вимоги, перераховані в чорновому варіанті до документу, у якому обговорюються і при необхідності модифікуються. Побічні вимоги видаляються. Нововиявлені вимоги додаються.

Для перевірки обґрунтованості вимог (requirements validation) необхідна більш повна версія документа, в якому всі вимоги чітко ідентифіковані і класифіковані. Учасники проекту вивчають документ і проводять наради з їх формальному перегляду. Перегляди (reviews) часто структуровані у вигляді так званих процедур наскрізного контролю (walkthroughs) або інспекцій (inspections). Пере огляди є різновидом тестування (testing) (розд. 10.1.1).

3.3.1. Вимоги, що виходять за рамки проекту

Вибір проекту в сфері IT і, отже, підлягає реалізації системи (і їх чітких меж) визначається в рамках виду діяльності, який отримав назву системного планування. Однак, деталізовані взаємозалежності між системами можуть бути розкриті лише на етапі аналізу вимог. Установлення меж системи (системних рамок) - завдання етапу аналізу вимог, так що проблема "розтягування рамок" може вирішуватися на ранніх етапах процесу розроблення як складова частина цього завдання.

Щоб мати можливість вирішити, чи лежить конкретну вимогу в рамках системи або виходить за рамки, необхідна еталонна модель, по відношенню до якої і повинно прийматися рішення. Історично роль подібної еталонної моделі сиг рала контекстна діаграма (context diagram) - головна міжнародна діаграма популярно го методу структурного моделювання під назвою діаграма потоків даних (Data Flow Diagrams - DFD). Хоча в мові UML місце цієї діаграми зайняла діаграма прецедентів, контекстна діаграма, як і раніше залишається чудовим методом встановлення меж системи.

Існують, втім, і інші причини, за якими вимога може бути розцінено, як виходить за рамки проекту [81]. Наприклад, вимога може становити велику трудність для реалізації в комп'ютеризованої системи і залишається

прерогативою людини, що бере участь в процесі. Або вимога може мати низький пріоритет і виключається з першої версії системи. Вимога може бути також реалізовано за допомогою апаратного забезпечення або зовнішніх пристроїв і, таким чином, може виявитися поза контролем з боку програмного забезпечення.

3.3.2. Матриця залежності вимог

Вважаючи, що всі вимоги чітко ідентифіковані і пронумеровані, можна сконструювати матрицю залежностей вимог (requirements dependency matrix) (або матрицю взаємодії (interaction matrix вимог)) [81], [46]. У стовпці і рядку заголовка перераховані впорядковані ідентифікатори вимог, як показано на рис. 3.2.

3.3. Узгодження і перевірка обґрунтованості вимог

вимоги	**Т1**	**Т2**	**Т3**	**Т4**
Т1	X	X	X	X
Т2	Конфлікт	X	X	X
Т3			X	X
Т4		Перекриття	Перекриття	X

Рис. 3.2. Матриця залежності вимог

Верхня права частина матриці (над діагоналлю включно) не використовується. Решта осередки вказують на те, перекриваються чи два будь-яких вимоги, протиречивими один одному або незалежні один від одного (порожні клітинки). Суперечливі вимоги необхідно обговорити з замовниками і при можливості Переформулювати для пом'якшення протиріч (фіксацію протиріччя, видиму для наступної розробки, необхідно зберегти). Перекриваються вимоги також повинні бути сформульовані заново, щоб виключити збіги.

Матриця залежності вимог являє собою простий, але ефективний метод виявлення протиріч перекриттів, коли кількість вимог відносно невелика. В іншому випадку описаний метод все ж можна застосувати, якщо вдається згрупувати вимоги по категоріям (розд. 3.4.1), а потім порівняти їх окремо в межах кожної категорії.

3.3.3. Ризики і пріоритети вимог

Після того, як в результаті зняття протиріч і усунення повторив вироблений переглянутий набір вимог, їх необхідно піддати аналізу ризиків і призначити їм пріоритети. Аналіз ризиків (risk analysis) спрямований на ідентифікацію вимог, які є потенційними джерелами працю ностей в розробці. Призначення пріоритетів (prioritization) необхідно для того, що б забезпечити можливість без праці змінити рамки проекту в разі виникнення непередбачених затримок.

Вимоги можуть бути "ризикованими" внаслідок впливу різних чинників.

Вимогам властиві такі типові види ризиків [81].

1. Технічний ризик, коли вимога технічно важко реалізувати.
2. Ризик, пов'язаний зі зниженням продуктивності, коли вимога - будучи реалізовано - може несприятливо позначитися на часі реакції системи.
3. Ризик, пов'язаний з порушенням безпеки, коли вимога - будучи реалізовано - може створити проломи в захисті системи.
4. Ризик, пов'язаний з процесом розробки, коли для реалізації вимоги необхідно використання незвичайних методів розробки, незнайомих розробникам (наприклад, методів формальної специфікації).
5. Ризик, пов'язаний з порушенням цілісності баз даних, коли вимога не може бути легко перевірено і може привести до суперечливості даних.
6. Політичний ризик, коли вимога може виявитися важким виконати через внутрішньополітичні причини.
7. Ризик, пов'язаний з порушенням законності, коли вимога може привести до порушення чинних законів або очікуваної зміни закону.
8. Ризик пов'язаний з мінливістю, коли вимога може потенційно змінюватися або еволюціонувати протягом процесу розробки.

В ідеалі пріоритети вимогам призначають окремі замовники в процесі ви явища вимог. Потім вони узгоджуються на нарадах і знову змінюються після додатки до них факторів ризику.

Для виключення невизначеності і полегшення призначення пріоритетів коли чество груп пріоритетів не повинно бути занадто велике. Зазвичай використовується від трьох до п'яти пріоритетів. Їх можна позначити як "високий", "середній", "низький" і "невизначений". Альтернативний перелік пріоритетів може виглядати, на приклад, так: "істотне", "корисне", "важко визначити" і "відкладене".

3.4. Управління вимогами

Вимогами необхідно управляти. Управління вимогами в дійсності ності є частиною загального управління проектом. Воно пов'язане з трьома основними питаннями.

1. Ідентифікація, класифікація, організація і документування вимог.

2. Зміна вимог (за допомогою процесів, які встановлюють спосо б висунення, узгодження, перевірки достовірності та документування неминучих змін до вимог).

3. Відстеження вимог (за допомогою процесів, які підтримують відносини взаємозалежності між вимогами та іншими системними ар тефактамі, а також, власне, між вимогами) (зверніться до глави 10).

3.4.1. Ідентифікація та класифікація вимог

Вимоги описуються природною мовою, наприклад, наступним чином.

1."Система повинна запланувати наступний телефонний дзвінок клієнту за запитом Телемаркетер".

2."Система повинна автоматично набирати запланований телефонний номер і одночасно відображати на екрані Телемаркетер інформацію, що включає телефонний номер, номер клієнта, ім'я клієнта".

3."У разі успішного з'єднання система повинна відобразити вступний текст, з яким телемаркетер може звернутися до клієнта для того, щоб зав'язати розмову".

Типова система може складатися з сотень або тисяч формулювань вимог, на зразок тих, що наведені вище. Для належного управління такою величезною кількістю вимог їх необхідно пронумерувати за допомогою деякої схеми ідентифікації (identification scheme) Схема може включати класифікацію вимог у вигляді груп, які легше піддаються управлінню.

Існує кілька методів ідентифікації та класифікації вимог [46].

Унікальний ідентифікатор - зазвичай послідовний номер, присвоєний вручну або згенерований з використанням бази даних CASE-засоби (тобто бази даних (або сховища), в якій CASE-засоби зберігають артефакти, вироблені в результаті аналізу або проектування).

3.4.2. Управління вимогами

1. Послідовний номер всередині ієрархії документа - присвоюється з урахуванням положення вимог в межах документа опису вимог (наприклад, сьоме вимога в третьому розділі другого розділу може бути пронумеровано як 2.3.7).
2. Послідовний номер в межах категорії вимог - присвоюється на додаток до мнемонічному імені, яке позначає категорію вимог (де під категорією вимог увазі функціональне вимога, вимога до даних, вимоги до продуктивності, вимоги до безпеки і т.д.)

Кожен з методів ідентифікації має свої за і проти. Найбільшою гнучкістю і захищеністю від помилок відрізняються унікальні ідентифікатори, генеруємі за допомогою бази даних. Бази даних мають вбудовані можливості ми генерації

унікальних ідентифікаторів для кожного нового запису даних в умовах одночасного доступу до даних з боку багатьох користувачів.

Деякі бази здатні додатково підтримувати супровід не скількох версій однієї і тієї ж записи (за рахунок розширення значення унікального ідентифікатора за допомогою номера версії). Нарешті, бази даних можуть мати можливості підтримки посилальної цілісності зв'язків між артефактами моделювання, включаючи вимоги, і можуть, таким чином, забезпечити необхідну підтримку управління та простежуваності вимог.

3.4.3. Ієрархії вимог

Вимоги можна впорядкувати у вигляді ієрархічно впорядкованої структури, що представляє відношення батько-нащадок. Ставлення родитель-потомок подібно відношенню композиції (розд. 2.1.4). Батьківське вимога складено з дочірніх вимог. Дочірнє вимога - це фактично "підвимоги" батьківського вимоги.

Ієрархічні відносини дозволяють ввести додатковий рівень класифікації вимог. Це може безпосередньо відображатися (або не відбиватися) в ідентифікаційнім номері (за допомогою використання точки в запису складеного име ні). Тому вимога, пронумерована як 4.9, може бути дев'ятим "нащадком" "батька" з ідентифікаційним номером, рівним 4.

Наведений нижче набір формулювань є ієрархічно упорядковані вимоги.

1. "Система повинна запланувати наступний телефонний дзвінок клієнту за запитом Телемаркетер".

1.1. "Система повинна активізувати клавішу Next Call (Наступний дзвінок) при відкритті форми Telemarketing Control (Управління Телемаркена Тінг) або при завершенні попереднього виклику".

1.2. "Система повинна видалити дзвінок з початку черги запланованих дзвінків і надати йому статус поточного дзвінка".

1.3. Інші вимоги ...

Ієрархії вимог дозволяють визначати вимоги на різних рівнях абстракції. Це узгоджується із загальним принципом моделювання, відповідно документу, котрим при переході до більш низького рівня абстракції моделі систематично збагачуються все новими деталями. В результаті для батьківських вимог можна сконструювати узагальнені вимоги, а деталізовані моделі можна замовити з дочірніми вимогами.

3.4.4. Управління змінами

Вимоги змінюються. Вимоги можуть змінюватися, вони можуть бути видалені, а ти нові вимоги можуть бути додані на будь-якому етапі розробки. Зміни можна рас розглядати як "удар", а ось некеровані зміни - це справжній удар по проекту.

Чим далі просунулася розробка, тим дорожче обходиться внесення змін. Фактично, мінімальні витрати, необхідні для виправлення проекту після внесення змін, завжди ростуть і часто ростуть експоненціально. Зміна щойно сформульованих вимог, які не пов'язані з іншими вимогами, - це проста вправа в редагуванні. Зміна тих же вимог, після того як вони реалізовані, може обійтися надзвичайно дорого.

Зміни можуть бути пов'язані з суб'єктивними помилками, але часто вони вимушені звертатися покликані змінами внутрішньої політики або зовнішніми факторами, такими як конкурентні сили, глобальні ринки або технологічні досягнення. Поза залежності від причин, для документування запитів на зміни (change request) оцінки впливу, що чиниться змінами (change impact), і здійснення змін необхідна сильна стратегія управління змінами.

Оскільки зміна вимог обходиться дорого, для кожного запиту на зміни необхідно створювати формальний бізнес-прецедент. Обґрунтована зміна, яке не зустрічалося раніше, оцінюється з точки зору його технічної реалізації, впливу на решті проект і витрат. Після узгодження вимога включає в відповідні моделі і реалізується в програмному забезпеченні.

Управління змінами, крім іншого, включає відстеження великих обсягів по мов взаємозалежної інформації протягом тривалого періоду часу. Без інструментів підтримки управління змінами приречене. В ідеалі, зміни охоронних вимог повинні зберігатися і відслідковуватися за допомогою засобів конфігураційного управління ПЗ, яке використовується розробниками для контролю версій моделей і програм протягом ЖЦ розробки. Відповідні CASE-засоби повинні або мати власні можливості конфігураційного управління, або можливості взаємодії з автономними засобами конфігураційного управління.

3.4.5. Відстеження вимог

Відстеження вимог (requirements traceability) - це всього лише частина, хоча і вкрай важлива частина, управління змінами (change management). Блок простежується мости вимог технології управління змінами підтримує відносини простежуваності, щоб фіксувати зміни, які виходять з або вносяться до вимог протягом ЖЦ розробки.

Розглянемо вимога: "Система повинна запланувати наступний телефонний дзвінок клієнту за запитом Телемаркетер". Ця вимога можна згодом смо деліровать за допомогою діаграми послідовностей, яка активізується з GUIінтерфейса за допомогою кнопки запуску дії, позначеної як Next Call, і тригера, запрограмованого в базі даних. Якщо між усіма цими елементами існує ставлення простежуваності, зміна будь-якого елементу знову відкритому кість відношення для обговорення - траєкторія системи "підпадає під підозру" (кажучи мовою одного з інструментальних засобів).

Ставлення простежуваності може охоплювати багато моделей послідовних при етапів ЖЦ. Безпосередньою модифікації підлягають тільки суміжні зв'язку з простежування. Наприклад, якщо елемент А сходить до

елементу B, а елемент B сходить до елементу C, зміна, внесена в будь-який з кінцевих точок відносини, має здійснюватися в два етапи: спочатку за допомогою модифікації зв'язку A-B, а потім B-C. (Більш докладно простежуваності і управління вимогами розглядають Ріва в главі 10).

3.5. Бізнес-модель вимог

На етапі встановлення вимог здійснюється виявлення вимог і їх визначення, переважно, у вигляді формулювань природною мовою. Формальне моделювання вимог з використанням мови UML проводиться пізніше на етапі специфікації вимог. Проте, під час встановлення вимог постійно ведеться діяльність по узагальненому візуального уявлення зібраних вимог, звана бізнес-моделюванням вимог.

Для встановлення рамок системи потрібно, щонайменше, Високорівнева візуальна модель, що дозволяє позначити ключові прецеденти і ввести найбільш істотні бізнес-класа. На рис. 3.3 показані залежності між цими трьома моделями етапу встановлення вимог і моделі інших етапів ЖЦ розробки.

Провідна роль діаграм прецедентів в ЖЦ розробки знайшла своє відображення на рис. 3.3, з якого ясно, що джерелом тестових прецедентів, призначеної для користувача документації і проектних планів виступають моделі прецедентів. Більш того, діа грами прецедентів і моделі класів використовуються паралельно і по черзі грають роль "лідера гонки" в рамках послідовних ітерацій розробки. Проектування і реалізація також тісно переплетені і можуть ініціювати зворотний зв'язок з моделями специфікації вимог.

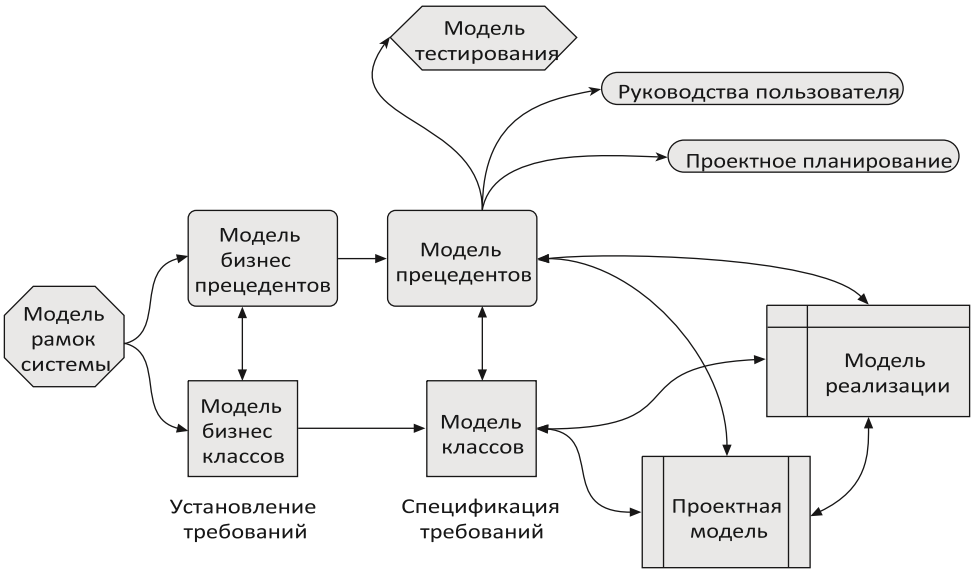

Рис. 3.3. Використання бізнес моделей на різних етапах вироблення вимог