ОДЕСЬКА НАЦІОНАЛЬНА АКАДЕМІЯ ЗВ’ЯЗКУ ІМ. О. С. ПОПОВА

Кафедра мереж зв’язку

ЗВІТ
З виробничої практики для студентів 4-го курсу
ННІ ІКПІ ОНАЗ ім. О.С. Попова

Студента групи ТЕ-4.2.01
Корнійчука Миколи

Одеса 2016

Зміст

1. ЗАГАЛЬНІ ПИТАННЯ

1.1 Організаційна структура підприємства

Більшою мірою підприємсво є аутсорсинговим, проте має і власні розробки і програмні продукти. Підприємство представляє групу середніх підприємств і включає в себе 100 осіб.

Підприємство не має філій або представництв.

Умовно підприємство можна розділити на 3 частини:

Менеджмент,

Розробка та підтримка ПО,

Обслуговуючий персонал офісу.

Частина менеджменту включає в себе засновників, директорів, фінансових директорів, адміністраторів, рекрутерів. Зв’язок між ланкою менеджменту та найчисленнішим відділом розробки ПО утворює проектний менеджер, або менеджери, що відповідають за комунікацію між клієнтами та командами розробки.

Відділ розробки умовно розділений на команди за спеціалізацією та навичками. Окремо від команд розробників також сконцентрована команда технічної підтримки продуктів розробки.

Третя ланка робітників – ті що відповідають за надання робітникам умов праці. Сюди відносяться системні адміністратори, що контролюють, адмініструють та покращують мережу, юристи, що займаються оформленням працівників в державному реєстрі, бухгалтери, що займаються обліком фінансових справ робітників, тощо. Сюди ж відносяться й постачальники води, продуктів, ланчів.

1.2 Технічне оснащення підприємства

Головною частиною мережі підприэмства є датацентр до якого підключено кожне робоче місце, точки доступу, телефони та додаткові комутатори.

Кожне робоче місце відділу розробки та підтримки ПО оснащене:

- IP-телефоном Cisco 7945 з підтримкою PoE,
- Ноутбуком HP Probook 4740s * (або іншим, зі схожими характеристиками)
- Монітором Монитор 24" Dell UltraSharp *(або іншим схожим за характеристиками)
- Гарнітурою/навушниками для skype-конференцій.

2. КОМП'ЮТЕРНА МЕРЕЖА ПІДПРИЄМСТВА

2.1 Склад мережевого обладнання та програмного забезпечення.

Для підключення обладнання співробітників використовуються комутатори серії Cisco Catalyst 2960 на 24 порти з підтримкою PoE:

Інтелектуальні Ethernet-комутатори Cisco Catalyst серії 2960 (Cisco Catalyst 2960 Series Intelligent Ethernet Switch) дозволяють реалізувати розширені сервіси в локальних мережах великих і середніх підприємств, а також в мережах філій. Представники цього сімейства автономних комутаторів з фіксованою конфігурацією забезпечують підключення робочих місць на швидкостях 10/100 Fast Ethernet і 10/100/1000 Gigabit Ethernet.

Можливості моделей серії:

Підключення: підключення Fast Ethernet і Gigabit Ethernet в конфігураціях з 8, 24 і 48 портами

Живлення пристроїв по витій парі: конфігурації з 24 портами з повною підтримкою PoE і 24 портами (з підтримкою PoE на 8 портах)

Інтегровані функції безпеки, включаючи контроль доступу в мережу (NAC)

Розширені можливості управління якістю обслуговування (QoS) і забезпечення відмовостійкості

Інтелектуальні сервіси на кордоні мережі

З програмних рішень одним з базових є Checkpoint mobile від Checkpoint software technologies LTD що являє собою VPN доступ до інтранету.

VPN може використовуватися для того щоб один віддалений комп'ютер міг, через Інтернет підключитися до локальної мережі. Для цього на комп'ютері, який хоче підключитися до локальної мережі через VPN, повинно бути встановлено спеціальне програмне забезпечення, яке буде виконувати функції VPN.

2.2 Архітектура мережі, організація виходу у зовнішні мережі.

Незважаючи на рекомендації ITU-T, на практиці нині найбільшого поширення набула чотирирівнева архітектура NGN, у якій рівень послуг і транспортний рівень в свою чергу зазнали подальшої декомпозиції на такі рівні (рис. 2.1):

- рівень управління послугами (четвертий рівень);
- рівень мережного контролю й управління (третій рівень);
- транспортний рівень (другий рівень);
- рівень доступу (перший рівень);
- термінальне обладнання (нульовий рівень).

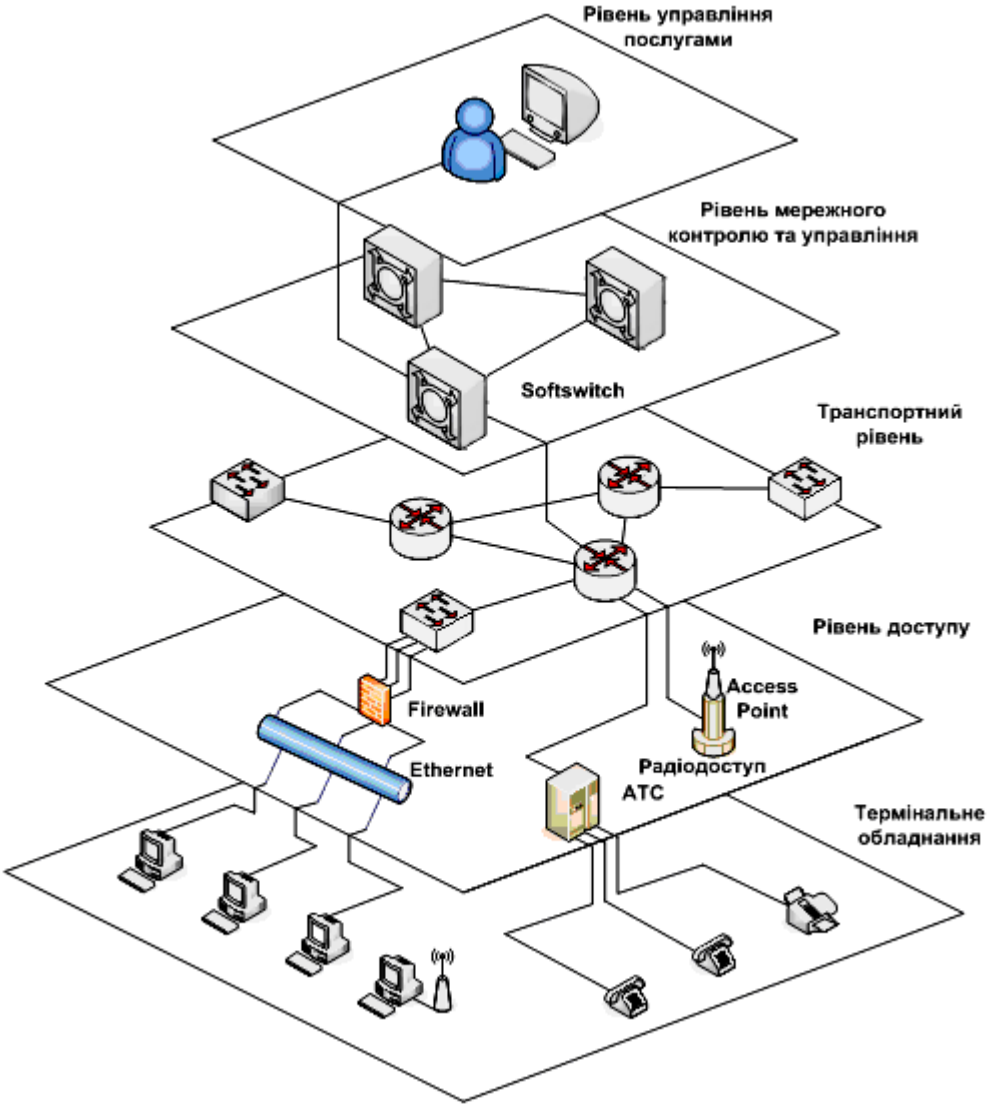

Рис.2.1 : Рівні NGN

2.3 СКС підприємства зв’язку, тип кабелю, пасивного обладнання, структура.

СКС підприємства представляє собою горизонтальну систему двох поверхів (11 та 12). Тут розташовано:

Датацентр

Кроссову

Серверну

Додаткове оснащення аппаратних кімнат:

Охоронна та пожежна сигналізація

Аварійне освітлення і кондиціонування

Джерела безперебійного живлення будівлі

Робоче місце адміністратора, обладнане телефонним зв'язком

Горизонтальна підсистема, утворена внутрішніми горизонтальними кабелями між РПЕ та інформаційними розетками робочих місць, комутаційним обладнанням у РПЕ, комутаційними шнурами і перемичками. Горизонтальні підсистеми обох поверхів нашої СКС виконані по топології «зірка».

В якості протокола канального рівня використовано Ethernet.

2.4 Технології, які застосовуються у компьютерній мережі підприємства.

VPN (Віртуальна приватна мережа, англ. Virtual Private Network) — це логічна мережа, створена поверх інших мереж, на базі загальнодоступних або віртуальних каналів інших мереж (Інтернет). Безпека передавання пакетів через загальнодоступні мережі може реалізуватися за допомогою шифрування, внаслідок чого створюється закритий для сторонніх канал обміну інформацією. VPN дозволяє об'єднати, наприклад, декілька географічно віддалених мереж організації в єдину мережу з використанням для зв'язку між ними непідконтрольних каналів.

Прикладом створення віртуальної мережі використовується інкапсуляція протоколу PPP в будь-який інший протокол — IP (ця реалізація називається також PPTP — Point-to-Point Tunneling Protocol) або Ethernet (PPPoE). Деякі інші протоколи так само надають можливість формування захищених каналів (SSH).

NAT — дана технологія дозволяє перетворювати IP-адреси пакетів, забезпечуючи зв’язок між глобальної та локальної мережею за допомогою маршрутизуючого пристрою. Таким чином, група комп’ютерів може функціонувати по одному зовнішньому адресою. Крім поповнення нестачі мережевих адрес, NAT сприяє посиленню безпеки комп’ютера і спрощення адміністрування — все перераховане відноситься до плюсів даної технології. Разом з тим, трансляція мережевих адрес має ряд мінусів, до яких відноситься нездатність деяких протоколів подолати NAT, додаткові складнощі з ідентифікацією користувачів і т.п.

2.5 Захист мережі

Firewall Cisco ASA series (між-мережевий екран або мережевий екран) - комплекс апаратних чи програмних засобів, що здійснює контроль і фільтрацію мережевих пакетів, які проходять через нього, відповідно до заданих правил.

Основним завданням мережевого екрану є захист комп'ютерних мереж або окремих вузлів від несанкціонованого доступу. Також мережеві екрани часто називають фільтрами, так як їх основне завдання - не пропускати (фільтрувати) пакети, що не підходять під критерії, визначені в конфігурації.

Деякі мережеві екрани також дозволяють здійснювати трансляцію адрес - динамічну заміну внутрішньомережевих (сірих) адрес або портів на зовнішні, які використовуються за межами ЛВС.

Мережеві екрани поділяються на різні типи залежно від таких характеристик:

- чи забезпечує екран з'єднання між одним вузлом і мережею або між двома і більше різними мережами;
- на рівні яких мережевих протоколів відбувається контроль потоку даних;
- чи відслідковується стан активних з'єднань чи ні.

Залежно від рівня, на якому відбувається контроль доступу, існує поділ на мережеві екрани, що працюють на:

- мережевому рівні, коли фільтрація відбувається на основі адрес відправника і одержувача пакетів, номерів портів транспортного рівня моделі OSI та статичних правил, заданих адміністратором;
- сеансовом рівні (також відомі як stateful) - відстежують сеанси між додатками, не пропускають пакети, які порушують специфікації TCP/IP, та часто використовуються у зловмисних операціях - скануванні ресурсів, зломи через неправильні реалізації TCP/IP, обриви/уповільнення з'єднань, ін'єкції даних.
- рівні додатків, фільтрація на підставі аналізу даних програми, переданих всередині пакету. Такі типи екранів дозволяють блокувати передачу небажаної і потенційно небезпечної інформації на підставі політик і налаштувань.

3. Мережі передачі даних підприємства зв'язку

3.1 Послуги передачі даних, що надаються підприємством для розробників програмного забезпечення

Розробка програмного забезпечення — процес, спрямований на створення та підтримку працездатності, якості та надійності програмного забезпечення, використовуючи технології, методологію та практики з інформатики, керування проектами, математики, інженерії та інших областей знання.

Вимоги до програмного забезпечення — набір вимог щодо властивостей, якості та функцій програмного забезпечення, що буде розроблено, або знаходиться у розробці. Вимоги визначаються в процесі аналізу вимог та фіксуються в специфікації вимог,діаграмах прецедентів та інших артефактах процесу аналізу та розробки вимог.

Розробка вимог до програмної системи може бути розділена на декілька етапів:

- Знаходження вимог (збір, визначення потреб заінтересованих осіб та систем).
- Аналіз вимог (перевірка цілісності та закінченості).
- Специфікація (документування вимог).
- Тестування вимог.

Види вимог за рівнями:

- Бізнес-вимоги — визначають призначення ПЗ, можуть описуватися в документі про **бачення** (англ. *vision*) та документі про **межі проекту** (англ. *scope*).

- Вимоги користувача — визначають набір завдань користувача, які повинна вирішувати програма, а також сценарії їхнього вирішення в системі. Ці вимоги можуть мати вигляд тверджень, варіантів використання, історій користувача, сценаріїв взаємодії.
- Функціональні вимоги — визначають **«що»** повинен робити програмний продукт. Ці вимоги описуються в документі **Специфікації програмного забезпечення** (англ. *SRS*).

Види вимог за характером:

- Функціональний характер — вимоги до поведінки системи
  - Бізнес-вимоги
  - Вимоги користувача
  - Функціональні вимоги
- Нефункціональний характер — вимоги до характеру поведінки системи
  - Бізнес-правила — визначають обмеження, що витікають з предметної області.
  - Системні вимоги — вимоги до програмних інтерфейсів, надійності, обладнанню.
  - Атрибути якості
  - Зовнішні системи та інтерфейси
  - Обмеження

Джерела вимог:

- Законодавство
- Вимоги стандартів
- Бізнес-процеси
- Очікування та бачення користувачів системи

Методи знаходження вимог:

- Спілкування з майбутнім користувачем: інтерв'ю, анкетування.
- Мозковий штурм, семінар.
- Аналіз нормативної документації та законодавства.
- Аналіз бізнес-процесів.

Документування вимог

Зазвичай вимоги використовують як засіб комунікації між різними заінтересованими особами та системами. З цього виходить, що вимоги повинні бути простими та зрозумілими як для звичайних користувачів, так і для розробників. Для цього створюються наступні документи:

- Бачення (Vision)
- Специфікація вимог до програмного забезпечення (англ. *Software Requirements Specification, SRS*)

Вимоги до ПЗ можуть документуватися в текстовому або графічному вигляді. Текстові вимоги - це стислий та розгорнутий описи якогось прецеденту. Для графічного представлення використовують наступні нотації: ER (IDEF1FX), IDEF0, IDEF3, DFD, UML, OCL, SysML, ARIS (eEPC, VAD).

Вимоги в процесах розробки

Різні методології розробки програмного забезпечення по-різному працювали з вимогами. В дуже старій, та не актуальній моделі **водоспаду** (англ. *waterfall*) етап аналізу та розробки вимог є першим. Особливістю є те, що він повністю закінчується до початку проектування та розробки ПЗ, а останні не можуть початися до завершення аналізу вимог.

В ітеративних процесах розробки фаза аналізу та розробки вимог в різному об'ємі є на кожній ітерації.

Стадії розробки програмного забезпечення

У розробці програмного забезпечення, стадії розробки програмного забезпечення використовуються для позначення ступеня готовності програмного продукту. Також стадію розробки може відображати кількість реалізованих функцій, запланованих для певної версії програми. Стадії або можуть бути офіційно оголошені і регламентуються розробниками, або іноді цей термін використовується неофіційно для опису стану продукту. Слід зазначити, що стадії Beta і Alpha (або Pre-Alpha) не є показниками стабільності чи нестабільності релізу, оскільки присвоюються програмі один раз або один раз за серію (серією, в цьому випадку, вважається число до першої крапки), залежно від системи розробки. Вони можуть присвоюватися декільком релізам поспіль. Релізом в цьому випадку вважається завершена версія.

Pre-alpha — початкова стадія розробки; період часу зі старту розробки ПЗ до виходу стадії «Alpha» (або до будь-якої іншої, якщо стадії «Альфа» немає). Також, так називаються програми, що не увійшли ще в стадію альфа або бета, але минули стадію розробки, для первинної оцінки функціональних можливостей в дії. На відміну від альфа і бета версій, пре-альфа може включати в себе не весь спектр функціональних можливостей програми. У цьому випадку, маються на увазі всі дії, що виконуються під час проектування і розробки

програми аж до тестування. До таких дій відносяться — розробка дизайну, аналіз вимог, власне розробка програми, а також налагодження окремих модулів.

Alpha — стадія, під час якої ПЗ підлягає внутрішньому тестуванню. Стадія початку тестування програми в цілому фахівцями-тестерами, зазвичай, не розробниками програмного продукту, але, як правило, усередині організації або співтовариства, що розробляють цей продукт. Також це може бути стадія додавання нових функціональних можливостей. Програми на цій стадії можуть застосовуватися тільки для ознайомлення з майбутніми можливостями.

Beta — стадія, під час якої ПЗ підлягає публічному тестуванню; стадія активного бета-тестування і налагодження програми, що пройшла альфа-тестування (якщо таке було). Програми на цій стадії розробки можуть бути використані іншими розробниками програмного забезпечення для випробування сумісності. Тим не менш, програми цього етапу можуть містити достатньо велику кількість помилок.

Оскільки бета-продукт не є фінальною версією, і публічне тестування проводиться на страх і ризик користувача, виробник не несе ніякої відповідальності за збиток, заподіяний в результаті використання бета-версії. Таким чином, багато виробників уникають відповідальності, надаючи користувачам тільки бета-версії продукту. Так, ICQ у версії 2003 року використала цей трюк, випустивши 2003b (b означає бета) версію свого інтернет-месенджера. Фінальної версії ICQ 2003 так і не з'явилося, натомість два роки по тому вийшли версії ICQ 4 та ICQ 5.

RTM (**r**elease **t**o **m**anufacturing) або **Release** — видання продукту, готового до тиражування. Це стабільна версія програми, що пройшла всі попередні стадії розробки, в яких виправлені основні помилки, але існує ймовірність появи нових, раніше не помічених, помилок. RTM-стадія передує загальній доступності (GA), коли продукт випущений для громадськості.

Процес розробки конкретного програмного рішення можна повною мірою описати через поняття «життєвого циклу».

Життє́вий цикл програ́много забезпечення — сукупність окремих етапів робіт, що проводяться у заданому порядку протягом періоду часу, який починається з вирішення питання про розроблення програмного забезпечення і закінчується припиненням використання програмного забезпечення

В загальному випадку,життєвий цикл визначається моделлю й описується у формі методології (методу). Модель або парадигма життєвого циклу визначає загальну організацію і, як правило, основні його фази та принципи переходу між ними. Методологія (метод) визначає комплекс робіт, їх детальний зміст і рольову відповідальність спеціалістів на всіх етапах вибраної моделі.

Життєвий цикл програмного забезпечення супроводжується розробленням, обігом та використанням програмної документації.

Програмна документація — сукупність документів, що містять відомості, необхідні для розробки, виготовлення, супроводу та експлуатації програм. Програмна документація є одним з видів технічної документації.

Комплекс державних стандартів, що встановлюють взаємопов'язані правила розробки, оформлення та обігу програм і програмної документації називається «Єдина система програмної документації» (ЄСПД)

Модель життєвого циклу - це структура, що складається із процесів, робіт та задач, які включають в себе розробку, експлуатацію і супровід програмного продукту; охоплює життя системи від визначення вимог до неї до припинення її використання. На сьогодні найбільшого розповсюдження набули дві моделі:

- каскадна модель;
- спіральна модель.

Водоспадна (каскадна) модель життєвого циклу ПЗ — послідовний метод розробки програмного забезпечення, названий так через діаграму схожу на водоспад (як на ілюстрації справа).

Цей метод з'явився раніше ніж перше програмне забезпечення. Його застосовували для створення складних інженерних конструкцій (літаків, мостів і

подібного). Зрозуміло, що зміни в проекті після того, як вже натягнуті розтяжки коштують дуже дорого, тому метод передбачає перфекціонізм на кожному етапі. А так, як колись ще не існувало формальних методів розробки ПЗ, для неї просто перейняли цю модель.

Плюси методу:

- Ніяких переробок
- Гарна специфікація перетікає в гарну документацію
- Зрозуміла модель
- Кодери можуть мати низьку кваліфікацію.

Мінуси

- Необхідний перфекціонізм на кожному етапі.
- Важко вносити зміни (якщо взагалі можливо)
- Надлишкове проектування
- Поділ розробників на "perfect" та "code monkeys"

Спіральна модель

Розробка ітераціями відображає об'єктивно існуючий спіральний цикл створення системи. Неповне завершення робіт на кожному етапі дозволяє переходити на наступний етап, не чекаючи повного завершення роботи на поточному. При ітеративному способі розробки відсутню роботу можна буде виконати на наступній ітерації. Головне ж завдання - щонайшвидше показати користувачам системи працездатний продукт, тим самим активізуючи процес уточнення і доповнення вимог.

Виходячи з можливості внесення змін, як в процес, так і в проміжний продукт було створено спіральну модель ЖЦ

Внесення змін орієнтоване на задоволення потреби користувачів одразу, як тільки буде встановлено, що створені артефакти або елементи документації не відповідають дійсному стану розробки.

Дана модель ЖЦ допускає аналіз продукту на витку розробки, його перевірку, оцінку правильності та прийняття рішення про перехід на наступний виток або повернення на попередній виток для доопрацювання на ньому проміжного продукту.

Відмінність цієї моделі від каскадної полягає в можливості багато разів повертатися до процесу формулювання вимог і до повторної розробки версії системи з будь-якого процесу моделі.

Для програмного продукту така модель не дуже підходить з декількох причин. По-перше, висловлення вимог замовником носить суб'єктивний характер, вимоги можуть багаторазово уточнюватися протягом розробки ПС і навіть після завершення та випробовування, і часом може з'ясуватися, що замовник «хотів зовсім інше». По-друге, змінюються обставини та умови використання системи, тому загальновизнаним законом програмної інженерії є закон еволюції, який сформулюємо так: кожна діюча ПС з часом потребує внесення змін або виводиться з експлуатації.

При необхідності внесення змін до системи на кожному витку з метою отримання нової версії системи обов'язково вносяться зміни в заздалегідь зафіксовані вимоги, після чого повертаються на попередній виток спіралі для продовження реалізації нової версії системи з урахуванням усіх змін.

Еволюційна модель

У разі еволюційної моделі система послідовно розробляється з блоків конструкцій. На відміну від інкрементної моделі в еволюційній моделі вимоги встановлюються частково і уточнюються в кожному наступному проміжному блоці структури системи.

Використання еволюційної моделі припускає проведення дослідження предметної області для вивчення потреб її замовника і аналізу можливості застосування цієї моделі для реалізації. Модель використовується для розробки нескладних і некритичних систем, де головною вимогою є реалізація функцій системи. При цьому вимоги не можуть бути визначені відразу і повністю. Тому розробка системи здійснюється ітераційним шляхом її еволюційного розвитку з отриманням деякого варіанта системи–прототипу, на якому перевіряється реалізація вимог. Іншими словами, такий процес за своєю суттю є ітераційним, з етапами розробки, що повторюються, починаючи від змінених вимог і закінчуючи отриманням готового продукту. В деякому розумінні до цього типу моделі можна віднести спіральну модель.

Розвитком цієї моделі є модель еволюційного прототипування в рамках усього ЖЦ розробки ПС

У літературі вона часто називається моделлю швидкої розробки програм RAD (Rapid Application Development). У даній моделі наведені дії, які пов'язані з

аналізом її застосовності для конкретного виду системи, а також обстеженням замовника для визначення потреб користувача при розробці плану створення прототипу.

У моделі є дві головні ітерації розробки функціонального прототипу, проектування і реалізації системи з метою перевірки, чи задовольняє вона всі функціональні і нефункціональні вимоги. Основною ідеєю цієї моделі є моделювання окремих функцій системи в прототипі і поступове еволюційне його доопрацювання до виконання всіх заданих функціональних вимог.

Ітерацій з отримання проміжних варіантів прототипу може бути декілька, в кожній з яких додається функція і повторно моделюється робота прототипу. І так до тих пір, поки не будуть промодельовані всі функції, задані у вимогах до системи. Після цього виконується ще одна ітерація – остаточне програмування для отримання готової системи.

Ця модель застосовується для систем, в яких найбільш важливими є функціональні можливості, і які необхідно швидко продемонструвати на CASE-засобах.

Оскільки проміжні прототипи системи відповідають реалізації деяких функціональних вимог, їх можна перевіряти і під час супроводу і експлуатації, тобто разом з процесом розробки чергових прототипів системи. При цьому допоміжні і організаційні процеси можуть виконуватися разом з процесом розроблення і накопичувати відомості за даними кількісних і якісних оцінок на процесах розроблення.

При цьому враховуються такі чинники ризику:

– реалізація всіх функцій системи одночасно може призвести до громіздкості;

– обмежені людські ресурси зайняті розробкою протягом тривалого часу.

Переваги застосування даної моделі ЖЦ такі:

– швидка реалізація деяких функціональних можливостей системи і їх апробація;

– використання проміжного продукту в наступному прототипі;

– виділення окремих функціональних частин для реалізації їх у вигляді прототипу;

– можливість збільшення фінансування системи;

– зворотний зв'язок із замовником для уточнення функціональних вимог;

– спрощення внесення змін у зв'язку із заміною окремих функції.

Модель розвивається у напряму додавання нефункціональних вимог до системи, пов'язаних із захистом і безпекою даних, несанкціонованим доступом до них і ін.

Незалежно від обранної методики проектування та розробки програмного забезпечення, над одним проектом працює ціла команда розробників. Всім їм потрібно мати загальне середовище для спілкування та обміну інформацією, система контролю версії, декілька середовищ для готової версії ПЗ (здебільшого для тестування), сервер з базою даних. Таких команд може бути порівняно багато. У зв’язку з цим, все навантаження припадає на мережу підприємства. І для компанії може коштувати збитків навіть декілька годин неправильної роботи мережі.

Ще один напрямок роботи підприємства – технічна підтримка, як власних, так і сторонніх продуктів. Це завдання потребує від мережі стабільної роботи, великої швидкості та високого рівня захищеності мережі. Адже робота вестиметься з мережею та терміналами клієнта, включаючи таке обладнання, як IP-телефони, компютери та бази даних клієнта.

Служба технічної підтримки на кожному підприємстві може бути побудована різноманітними способами (мається на увазі реалізації процесів підтримки). Існує кілька моделей служби підтримки, наприклад: централізована, локальна, віртуальна - з єдиним телефонним центром і т. д. Служба технічної підтримки може бути організована як в цілях обслужити зовнішніх клієнтів (аутсорсинг обслуговування комп'ютерів і т. п.), так і внутрішніх (підрозділ IT-департаменту на великих підприємствах).

В описі концепції ITIL, побудованої на процесному підході, Service Desk є єдиним описаним функціональним підрозділом. Цей виняток зроблено через велику важливість підрозділи техпідтримки при впровадженні практичному використанні сучасних IT-підходів та методик.

Правильно організована техпідтримка (Service Desk) завжди починається з реєстрації всіх звернень кінцевих користувачів, служить єдиною точкою для спілкування користувача з IT-службою. Найпопулярніші рішення з практичної організації техпідтримки часто будуються на базі Call-center (іноді навіть користувачі їх ототожнюють). Він є початковою точкою контактів кінцевих

користувачів зі службою технічної підтримки і служить джерелом інформації про їх фактичної задоволеності рівнем сервісу, що доповнює інформацію про технічні параметри якості обслуговування компанії-клієнта (зовнішнього або внутрішнього).

У даній компанії аутсорсері та аутстафері служба технічної підтримки часто організована за наступним багаторівневому принципі:

- Користувач - звертається з питанням в службу підтримки по телефону або за допомогою електронної заявки.

- Оператор (1-я лінія підтримки, Call-center) - реєструє звернення, при можливості допомагає користувачеві самостійно, або ескалірует (передає і контролює виконання) заявку на другу лінію підтримки.

- Друга лінія підтримки - отримує заявки від першої лінії, працює за ним, при необхідності залучаючи до вирішення проблеми фахівців із суміжних відділів (системні адміністратори, підтримка POS-термінал ів, підтримка спеціального ПЗ, підтримка спеціального обладнання (Дилінг) і т. д.)

Для автоматизації роботи служби технічної підтримки існує велика кількість програмних комплексів, наприклад:

1. LANDesk Service Desk
2. BMC Remedy
3. BOAS Help Desk (Сервіс Деск)
4. CA Service Desk
5. GLPI
6. HP Service Desk
7. IntraService (Service Desk, SaaS)
8. Microsoft System Center Service Manager
9. Naumen Service Desk
10. Open-source Ticket Request System
11. Symantec Service Desk
12. Terrasoft Service Desk
13. Кларіс Service Desk SaaS

Розуміння різних рівнів служби клієнтської підтримки

Допомога стіл одним з найважливіших елементів залучення клієнтів і допомагає вирішувати проблеми, ваші клієнти, пов'язані з вашого продукту/послуги. Агресивно допомагає клієнтам вирішувати проблеми, можуть значно підвищити свій рівень задоволеності клієнтів і може позитивно відображати в майбутніх продажів. Довідкової служби, як правило, досягається через безкоштовний номер телефону, хоча багато компаній все більше покладаються на веб-інструментів чату і соціальних мереж для виконання підтримки клієнтів.

Зазвичай організації, починаючи з підтримки клієнтів це зробити в одному ярусі - де одна точка контакту обробляє ваші запити і вирішує цю проблему. Однак, як ваша організація дозріває ви зрозумієте, що вам потрібно вийти за рамки однієї моделі рівня, і мають більш зернисту, багаторівнева підтримка.

Наприклад, якщо ви продаєте програмне забезпечення з управління проектами для бізнесу, ви можете отримати запити, починаючи від Що я повинен робити, якщо я забув ім'я користувача? до вельми езотеричної помилка, яка запобігає деякі клієнти в повній мірі відслідковувати свій час на їх IPad. В цьому випадку вам потрібно більш низькі хлопці рівня для обробки простих запитів при використанні ваших фахівців для вирішення жорстких помилок. Багаторівневе допоможе вам виділити ваші ресурси підтримки краще і зберегти дорогоцінні ресурси ваших експертів.

Рівень - рівень підтримки I / 1

Це базовий рівень підтримки клієнтів. Представник клієнта універсалом з більш широким розумінням продукту, але не могли зрозуміти внутрішню роботу системи. Він / вона в цьому випадку буде ідентифікувати клієнта, зрозуміти проблему і основні поради щодо вирішення проблеми.

Типові рішення можна знайти в FAQ або базу знань, які можуть бути використані в більшості клієнтських дзвінків. Підтримка 1-го рівня працює цілодобово і передані 3-й боку, у багатьох випадках. Коли 1-го рівня не в змозі справлятися з цим питанням, репутація класифікує цю проблему в багато типи, а потім вона переросла в відповідний рівень-2 контакту, а також квиток на питання відстеження може бути виданий клієнту.

Tier - підтримка II

Підтримка рівня Tier-II включає в себе технічні знання і реєстрації працює більш досвідчених техніків, які мають сильний вплив на усунення неполадок. Технік тут більш спеціалізованих і буде в першу чергу визначити, якщо питання стосується його / її домен на основі даних, зібраних фахівцем Tier-I. Якщо він знаходиться в домені, то він повинен бути визначений, якщо це новий випуск або існуючий питання. Розширені засоби діагностики і аналізу даних можна зробити тут.

Якщо проблема існує, то фахівець з Tier-II з'ясовує, чи є рішення або обхідний шлях в базі даних. Розчин потім запропонував замовнику. Проте, в деяких випадках не може бути ніякого рішення, і це відкрита помилка. У цьому випадку, стіл Tier-II додає додатковий елемент до списку помилок і в залежності від кількості примірників може попросити їх розробникам виправити якомога швидше.

Якщо це нова проблема, подальший аналіз має бути зроблено, щоб побачити, якщо це можна було б обійти. Клієнт потім буде запропоновано виправити. Проте, якщо виправлення не представляється можливим, то легко вона загострилася до Tier-III, де він зазвичай призначений розробником в компанії, відповідальної за продукт. Стіл Tier-II може бути укомплектований або бере участь компанії або відданого 3-й партії.

Tier - підтримка III

Tier-III є дуже спеціалізованої робота забезпечується фахівцями, які, як правило, беруть участь в розробці продукту. Питання під рукою може бути досить складним, і вони будуть збирати стільки даних, скільки можливо з двох нижніх ярусів.

У моїй попередній роботі в якості розробника в Microsoft в команді ОС Windows, я використовував, щоб отримати більш складні помилки в операційній системі передається з допоміжного персоналу по всьому світу, і від аварії звалища ви повідомити, коли збій програми. Кілька разів виправлення передбачає більш глибокий аналіз операційної системи, і ми б штовхаючи виправлення як оновлення Windows.

Підтримка рівня -IV

Це існує тільки в рідкісних випадках, коли кілька постачальників беруть участь в продукті. Наприклад, якщо ви великий мобільний розробник додатків і згадуваний питання потреби в вирішуватися на рівні мобільних ОС, наданої іншою компанією, ви можете перерости запит в службу підтримки в іншу організацію і попросити їх, щоб забезпечити виправлення.

Висновок

Під час проходження практики ознайомився зі структурою підприємства що займається розробкою програмного забезпечення, наданням послуг підтримки; ознайомився зі специфікою роботи на підприємстві, специфікою роботи мережі підприємства, залежно від поставлених задач та навантажень, відповідно до специфіки підприємства.

За допомогою технічних спеціалістів підприємства, специфікацій обладнання та документації, я здобув навички, щодо налаштування мережі для забезпечення її невідмовності та стабільної праці навіть при великих навантаженнях і великої кількості передаваних даних. Адже мережа підприємства, яке займається розробкою програмного забезпечення, має забезпечувати стабільну роботу всіх відділів. Також набув навичок встановлення контролеру точок безпровідного зєднання. Таке рішення дозволяє реалізувати концепцію BYOD (Bring Your Own Device), що дозволяє користувачам мережі не витрачати час на перепідключення до нової мережі. Контролер надає можливість організувати всі точки доступу в єдину бездротову мережу.