МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Опорний конспект лекцій з курсу

"МОБІЛЬНІ ІНФОРМАЦІЙНІ СИСТЕМИ "

для студентів спеціальності

"Інформаційні управляючі системи і технології"

ТЕРНОПІЛЬ – 2016

ЗМІСТ

1. ВСТУП. ПОНЯТТЯ МОБІЛЬНА ІНФОРМАЦІЙНА СИСТЕМА (МІС), ОСОБЛИВОСТІ, ХАРАКТЕРИСТИКИ ТА ОБМЕЖЕННЯ

Розглянемо в якості типового прикладу мобільної іформаційної системи інформаційну систему спостереження за транспортом "Skylock-Inform".

Система спостереження за транспортом призначена для забезпечення зв'язку з віддаленими та рухомими об'єктами – автотранспортом. За допомогою системи автотранспортні компанії можуть здійснювати якісний контроль та управління автоперевезеннями.

Моніторингова система дозволяє підвищити ефективність використання автотранспорту, оптимізувати транспортні витрати, збільшити рентабельність автоперевезень, контролювати місцеположення транспорту, швидкість та напрямок його руху, запобігає нецільовому використанню ресурсів.

Можливості системи

1. Ведення бази даних об'єктів - автомобілів, водіїв, клієнтів;
2. Розділений доступ груп операторів до об'єктів спостереження і функцій системи;
3. Використання різних способів телекомунікації (протоколу SMPP (Short message peer-to-peer protocol), пакетного радіозв'язку GPRS (General Packet Radio Service), стандартної технології передачі даних CSD (Circuit Switched Data) або GSM Data в мережі GSM (Global System for Mobile Communications));
4. Обмін повідомленнями з об'єктами спостереження (подача команд, прийом даних);
5. Відображення спостережуваних об'єктів на цифровій карті (поточного положення і треків переміщення за період часу);
6. Контроль за використанням пального.

Переваги системи

1. Можливість отримувати об'єктивну й достовірну інформацію по використанню автотранспорту в будь-який момент часу та за будь-який період;
2. Формування оптимальних схем руху автотранспорту;
3. Запобігання нецільовому використанню транспортних засобів водіями, у тому числі для власних потреб;
4. Можливість складання графіків завантаженості водіїв і автотранспорту, і як результат – збільшення об'єму вантажоперевезень;
5. Виключення можливості розкрадання палива та фальсифікації показників спідометру;
6. Можливість оперативного втручання в процес перевезень при необхідності.

Принцип роботи системи моніторингу Skylock

Інформаційна система моніторингу автотранспорту базується на використанні технології глобального позиціювання GPS (Global Positioning System). За допомогою системи GPS ведеться спостереження за місцеположенням рухомих об'єктів з високою точністю, визначаються головні параметри руху (швидкість, напрямок, пройдений шлях, відстань до кінцевого пункту, витрачений час).

На транспортний засіб встановлюється компактний блок обладнаний GPS, розміри блоку дозволяють встановлювати його приховано. Для обміну інформацією між транспортним засобом і диспетчерським пунктом використовують мережу мобільного зв'язку GSM. Робота системи в режимі "реального часу" забезпечується через канал GPRS.

Встановлений GPS-приймач визначає місцеположення транспортного засобу в просторі, обробляє отриману інформацію і передає її в Диспетчерський центр за допомогою GSM-мережі.

Центр спостереження Skylock представляє собою комп'ютеризоване робоче місце диспетчера, що виконує аналітичні функції та функції контролю. В Диспетчерському центрі виконується збір інформації, яка надходить від транспортних засобів, її обробка, аналіз та формування звітів про основні параметри руху по кожному об'єкту окремо або по всім об'єктам.

Користувачі системи

Система моніторингу розрахована на два режими роботи: режим безпосереднього адміністрування бази даних об'єктів та режим оператора (режим обмеженого доступу).

В режимі адміністрування диспетчеру доступні всі можливі операції з об'єктами:

- перегляд та редагування інформації про доступних авто/водіїв;
- перегляд та редагування інформації про доступні юніти (блоки з GPS-приймачами);
- перегляд та редагування інформації про клієнтів;
- додавання/видалення об'єктів;
- настройка атрибутів об'єктів.

Режим оператора дозволяє лише отримувати інформацію про авто/водіїв та юніти, тобто робити запити та формувати звіти.

Формування бази даних об'єктів (автотранспорт, водії, юніти, клієнти)

Атрибути об'єктів описуються кожний в окремій формі.

Опис атрибутів юнітів включає: номер юніта, номер SIM, тип юніта, виробник, версія, протокол передачі даних та ін.

Опис автомобілів включає такі основні характеристики: назву, державний номер, модель, категорія, об'єм двигуна, витрата палива на 100 км, місткість паливного баку та ін.

Введення особистих даних про водіїв та призначення на транспортний засіб виконується у формі "Водії".

Форма "Автомобілі"
Форма "Водії"

Визначення місцеположення транспортного засобу в режимі реального часу

При роботі із системою з використанням карти реалізований режим відображення різних карт або однієї і тієї ж в різних вікнах, тобто можна отримати відображення місцеположення різних груп автомобілів на різних картах.

Визначення місцеположення транспортного засобу можна робити запитами по даті, за період, та запит місцеположення в даний момент часу.

При запиті місцеположення в даний момент часу на карті відображається одна точка, подається опис стану транспортного засобу і фактична адреса його місцеположення, якщо автомобіль знаходиться в межах населеного пункту, для якого є карта.

При запиті за період часу на карті відображається трек (шлях) автомобіля за вибраний період часу.

Запит місцеположення в даний момент часу
Запит місцеположення за період часу

Планування маршрутів

В системі SkyLock планування маршрутів виконується на карті вручну, також передбачена можливість автоматичного формування маршрутів по найкоротшій відстані в залежності від класу доріг. Для побудови оптимального шляху маршруту на карті України можна вибрати різні класи доріг: регіонального, державного або міжнародного значення. В залежності від вибраних класів доріг система може запропонувати декілька варіантів оптимального маршруту.

Звіти

Для аналізу виконаної роботи використовують функцію системи по створенню звітів. Форма звітів може бути різною в залежності від потреб користувача. Звіти можуть створюватися у форматах Microsoft Excel або HTML. Звіти складаються за будь-який період часу, по всім транспортним засобам або по окремим групам, детальний або зведений, по подіях, використаному пальному, по швидкості та ін.

Звіт за період часу по водіях

2. СТРУКТУРА І КЛАСИФІКАЦІЯ ІНФОРМАЦІЙНИХ СИСТЕМ

При класифікації інформаційних систем (ІС) завжди виникають проблеми, які пов'язані з математичними і алгоритмічними описами поставлених завдань. Структура і класифікація ІС проводиться за розглянутими нижче різноманітним критеріям.

За ступенем автоматизації

Інформаційні системи, в яких відсутній сучасні технічні засоби обробки інформації, і виконання всіх операцій здійснюється людиною, називаються ручними. Зараз їх практично немає.

Системи, що передбачають участь в процесі обробки даних людей, технічних засобів і програмного забезпечення - автоматизовані інформаційні системи.

Системи, що виконують всі операції по обробці даних без участі людини - це автоматичні інформаційні системи. Наприклад, деякі пошукові машини Internet. Робота з інформацією проводиться автоматично пошуковим роботом.

За характером використання інформації

Системи, для обробки, накопичення, пошуку і видачі необхідної інформації - інформаційно-пошукові.

Системи, призначені для аналізу даних з використанням експертних систем і баз знань - інформаційно-аналітичні.

ІС, які виконують накопичення та обробку даних з використанням прикладного ПЗ - інформаційно-вирішальні. Вони, в свою чергу діляться на: керуючі (що використовують бази даних і прикладні пакети) та експертні (застосовують практичні бази знань).

По архітектурі

Локальні, які працюють на одному ПК, не взаємодіючи з сервером.

Клієнт-серверні - системи, що працюють в локальній або глобальній мережі з єдиним сервером.

Розподілені - системи в гетерогенній мережі, керованої серверами.

За сферою застосування

Системи організаційного управління - забезпечують автоматизацію функцій управлінського складу.

ІС керування техпроцесами - забезпечують управління пристроями і техпроцесами на автоматизованому виробництві.

Системи наукових досліджень - призначені для наукових вишукувань.

САПР - програмні системи для виробництва проектних робіт з використанням математичних методів.

Навчальні-електронні підручники і довідники.

Інтегровані - забезпечують автоматизацію безлічі функцій підприємств.

Економічні - для роботи з управлінсько-економічною інформацією.

За ознакою структурованості поставлених завдань

Модельні ІС, що дозволяють досліджувати модель процесу або явища в процесі її вивчення.

Експертні ІС дозволяють обробляти відомості для установки можливих альтернативних рішень.

Ми розглянули основні критерії класифікації інформаційних систем.

3. ВИДИ ЗАБЕЗПЕЧЕННЯ МОБІЛЬНИХ ОПЕРАЦІЙНИХ СИСТЕМ

Організаційне забезпечення.

Під організаційним забезпеченням слід розуміти узгодження по місцю, часу і меті сумісне функціонування окремих виконавців, колективів і технічних засобів. Воно повинно здійснюватися і регулюватися деякими правилами взаємодії, які утворюють правовий та моральний кодекс і складають основу правового забезпечення. Тому організаційне забезпечення будується на нормативних актах правового забезпечення, а правове забезпечення знаходить своє втілення в організаційному забезпеченні.

Організаційне забезпечення інформаційної системи охоплює сукупність засобів, методів і відповідного персоналу. Воно повинно забезпечити:

- проведення техніко-економічного аналізу існуючої системи управління, вибору і постановки задач побудови інформаційної системи на етапі розробки і впровадження;

- регламентацію взаємодії персоналу з комплексом технічних засобів і між собою в процесі розв'язку задач управління, контролю ефективності роботи системи управління на етапі функціонування інформаційної системи.

На етапі проектування організаційне забезпечення виконує наступні задачі:

- аналіз існуючих систем управління і формулювання напрямів підвищення їх ефективності;

- вибір і постановку задач управління;

- формулювання вимог до комплексу технічних засобів;

- розробку організаційних рішень по складу, структурі, організації і методології розв'язку задач управління в інформаційній системі, склад робочих процедур і пояснення щодо їх виконання.

На етапі функціонування інформаційної системи організаційне забезпечення вирішує такі задачі:

- впровадження методів задач управління;

- організацію функціонування персоналу і комплексу технічних засобів інформаційної системи;

- контроль і аналіз ефективності управління;

- формування пропозицій по вдосконаленню і розвитку інформаційної системи.

В склад організаційного інформаційного забезпечення включаються схеми структури управління і списки штатних розкладів, уніфіковані форми документів, відомості про системи морального і матеріального стимулювання, посадові інструкції.

Правове забезпечення.

Основою ефективного розвитку підприємств є нормативно-правова база, яка грунтується на сукупності законів, потрібних для регулювання діяльності та створення сприятливих умов для її розвитку. Правове забезпечення - це сукупність норм, виражених в нормативних актах, які встановлюють і закріплюють організацію інформаційної системи, її мету, завдання, структуру і функції (правовий статус інформаційної системи і її підрозділів), призначених для регламентації створення і функціонування інформаційної системи. Правове забезпечення будується на базі юридичного підходу, який розглядає соціальне управління, як організаційну, в тому числі виконавчо-розпорядницьку діяльність державних органів, спрямовану на виконання законів та інших нормативних актів, прийнятих органами влади і управління. Юридичний підхід аналізує місце і роль права в управлінні, визначає зміст законної виконавської і розпорядчої діяльності державних і управлінських органів, виробляє рекомендації по її вдосконаленню.

Конструктивно функції управління в сфері права реалізуються у вигляді нормативно-правових актів, планів, положень і методик, обов'язкових для всіх ГОСТів, які визначають різноманітні заходи в сферах планування і управління. Нормативно-правові акти - це форма вираження і встановлення правових норм, сукупність яких утворює нормативно-правову базу управління, яка будується за ієрархічним принципом. Нормативно-правові акти поділяються на законодавчі, нормативні акти міністерств і відомств, акти місцевих органів влади і управління, локальні нормативні акти, індивідуальні акти.

На етапі функціонування інформаційної системи правове забезпечення включає:

- статус інформаційної системи в конкретних галузях державного управління;

- правове положення про компетенцію ланок інформаційної системи і організацію їх діяльності;

- права, обов'язки і відповідальність персоналу інформаційної системи;

- правове положення окремих видів процесів управління в інформаційній системі;

- порядок отримання і використання інформації в інформаційної системі, процедури її збору, реєстрації, зберігання, передачі і обробки;

- порядок отримання і використання комплексу технічних засобів, програмного, інформаційного та інших видів забезпечення.

Математичне забезпечення.

Математичне забезпеченнявключає сукупність математичних методів, моделей і алгоритмів для розв'язку задач управління і обробки інформації із застосуванням обчислювальної техніки. Математичне забезпечення містить засоби математичного забезпечення, методи вибору розв'язування, технічну документацію, персонал

(спеціалістів цієї галузі). До засобів математичного забезпечення відносяться: моделі процесів управління, алгоритми розв'язку задач управління, методи оптимізації за багатьма критеріями, методи математичного програмування, математичної статистики, теорії масового обслуговування та інші. До методів вибору рішення математичного забезпечення інформаційної системи відносяться методи вибору типів алгоритмів, оцінки їх точності, швидкодії і складності. Технічна документація включає описи задач і алгоритмів, постановки задач, описи пакетів прикладних програм, тести і контрольні приклади. Спеціалісти математичного забезпечення займаються постановкою задач та використання математичних аналітичних і числових методів.

Лінгвістичне забезпечення.

Лінгвістичне забезпечення охоплює сукупність науково-технічних термінів ти інших мовних засобів, правил формалізації мов, методів стиску запису інформації, засобів діалогу людини і обчислювальної системи. Лінгвістичне забезпечення включає в себе:

- інформаційні мови для опису структурних одиниць баз даних інформаційної системи (документів, показників, реквізитів);

- мови управління, маніпулювання і обміну даними в банку даних інформаційної системи;

- мовні засоби інформаційно-пошукових систем;

- мовні засоби системи автоматизованого проектування;

- діалогові мови;

- словники термінів і визначень.

Ергономічне забезпечення.

Ергономічне забезпеченняохоплює сукупність методів і засобів, призначених для створення оптимальних умов для ефективної діяльності і навчання операторів з складу персоналу інформаційної системи. Ергономічне забезпечення включає в себе:

- комплекс документації, яка містить ергономічні вимоги до робочих місць і здійснює експертизу робочих місць;

- комплекс методів, учбово-методичних матеріалів і технічних засобів підготовки персоналу до роботи;

- комплекс методів і засобів, які забезпечують професійний відбір.

В плані ергономічного забезпечення на етапах проектування інформаційної системи визначається ступінь і рівень участі людини в системі управління, вимоги до форми представлення інформації, умови оточуючого середовища діяльності людини, порядок роботи і відпочинку персоналу, нормативи навантаження і надійності персоналу; вимоги до технічних засобів, способи взаємодії персоналу і технічних засобів.

В реальних інформаційних системах загальне число видів забезпечення, що підтримують достатнє та повне функціонування організації може бути і меншим. Обов'язковими складовими інформаційної системи є лише підсистеми інформаційного, програмного і

технічного забезпечення. Функції інших видів забезпечення менш значимі і можуть об’єднуватися і групуватися або входити в основні підсистеми.

Центр ваги управлінської діяльності повинен бути зосереджений на функціональних задачах, таких, як багатофакторне планування, розрахунок балансу, розробка нормативів, облік і контроль. Органи управління повинні вирішувати складні і широкомасштабні інформаційні задачі, для чого вони повинні отримувати від об’єктів управління своєчасну, якісну, повну і достовірну інформацію; своєчасно обробляти її, виробляти команди управління і видавати звітність і результати аналізу вищестоячим організаціям. Одночасно повинні вирішуватися задачі прогнозування, перспективного і поточного планування, визначення основних напрямів розвитку науково-технічного прогресу.

4. ОПЕРАЦІЙНІ СИСТЕМИ ДЛЯ МОБІЛЬНИХ ПЛАТФОРМ

Symbian

Система від початку розроблялася консорціумом Symbian Ltd. Консорціум було засновано у червні 1998 року компаніями Psion, Nokia, Ericsson та Motorola. Пізніше до косорціуму приєдналися компанії Sony-Ericsson, Siemens, Panasonic, Fujitsu, Samsung, Sony та Sanyo. У 2008 сформовано некомерційну організацію Symbian Foundation, яка продовжила розробку операційної системи. З осені 2010 розробкою Symbian займається компанія Nokia, в той час як Symbian Foundation опікується лише юридичними питаннями, зокрема ліцензуванням платформи.

Symbian OS є спадкоємцем операційної системи EPOC32, котра була розроблена компанією Psion для своїх кишенькових комп'ютерів. У 1998-2000 роках значану частину системи було переписано з метою оптимізації коду для подальшої роботи на пристроях з обмеженими ресурсами. Розробникам вдалося досягти значної економії пам'яті, покращення кешування коду та, як наслідок, прискорення роботи програм, при знижених вимогах до енерговитрати. З точки зору розробки, виключною особливістю системи є повністю об'єктно-орієнтована архітектура (на рівні API). Починаючи з версій системи 9.x з'явився серйозний механізм захисту — розмежування API відповідно до прав (англ. capabilities) застосунків. Основна мова розробки застосунків — C++, є підтримка Java.

На 2010 рік найбільш розповсюдженою (за кількістю пристроїв) версією є Symbian OS Series 60 2nd Edition. У 2005 році вийшла Symbian OS Series 60 3rd Edition, що призвело до порушення зворотньої сумісності з програмами, розробленими для попередніх версій.

Windows Phone

Windows Phone (*Windows Mobile*) — операційна система для мобільних пристроїв з основним набором програм, таких як Windows Marketplace for Mobile, My Phone, Windows Live, заснованих на Microsoft Win32 API. Windows phone може працювати на ряді пристроїв, включаючи Pocket PC, смартфони, комунікатори.

Поточна версія називається "Windows Phone Classic 6.5. Вона заснована на Windows CE 5.2, має основний набір програм, розроблених з використанням Microsoft Windows API. Вона дещо аналогічна для настільних версій Windows, функціонально і естетично. Доступна третю частину software development для Windows Phone Classic, а також програмне забезпечення можна придбати через Windows Marketplace.

Початкова Pocket PC 2000 більше призначалася для управління стилусом, який використовується для введення команд, торкнувшись екрану. Windows Mobile була оновлена кілька разів, версію Windows Phone 7 Series, оголошено з великим торжеством на Mobile World Congress в Барселоні 15 лютого 2010.

Частка Windows Phone ринку смартфонів знижується. Зменшення 20% в 3 кварталі 2009. У Сполучених Штатах, це 3-тя найбільш популярна операційна система для бізнес-користувачів (після BlackBerry OS і iPhone OS), з 24% акцій серед корпоративних користувачів. У загальних продажів, це 4-а найбільш популярна операційна система, 7,9% частки світового ринку смартфонів.

iOS

iOS (відома як *iPhone OS* до червня 2010 року) — це власницька мобільна операційна система від Apple. Розроблена спочатку для iPhone, вона стала операційною системою також для iPod Touch, iPad і Apple TV. Apple не дозволяє роботи ОС на мобільних телефонах інших фірм. Станом на 31 травня 2011 року інтернет-магазин App Store містить

понад 500 тисяч застосунків для iOS, які були завантажені понад 15 мільярдів разів. Станом на травень 2010 року, App Store становив 15,4% ринку операційних систем для смартфонів, третій після Symbian і Blackberry.

Користувальницький інтерфейс IOS заснований на концепції прямої маніпуляції з використанням Multi-Touch жестів. Елементи інтерфейсу управління складаються з повзунків, перемикачів і кнопок. Він призначений для безпосереднього контакту користувача з екраном пристрою. Внутрішній акселерометр використовуються деякими програмами для реагування на струшування пристрою, яке є також загальною командою скасування, або обертати пристрій у трьох вимірах, що є загальною командою перемикання між книжковим та альбомним режимами. iOS є похідною від Mac OS X, отже, є за своєю природою Unix-подібною операційною системою.

Android

Android — операційна система і платформа для мобільних телефонів створена компанією Google на базі ядра Linux. Підтримується альянсом Open Handset Alliance (OHA).

Хоча Android базується на ядрі Лінукс, він стоїть дещо осторонь Лінукс-спільноти та Лінукс-інфраструктури. Базовим елементом цієї операційної системи є реалізація Dalvik віртуальної машини Java, і все програмне забезпечення і застосунки спираються на цю реалізацію Java.

MeeGo

MeeGo — мобільна операційна система на основі Linux з відкритими кодами, анонсована на Mobile World Congress в Барселоні в лютому 2010 Intel і Nokia на їхній спільній прес-конференції. Метою проекту стало поєднання зусиль Intel з його системою Moblin та Nokia з його Maemo в один проект. Згідно з Intel, MeeGo розробляється через недостатню підтримку процесорів Atom компанією Microsoft у Windows 7. 15 листопада 2010 року на MeeGo-конференції у Дубліні компанія AMD також приєдналася до проекту.

MeeGo націлений на різноманітні апаратні платформи, включаючи ручні комп'ютери і комунікатори, інформаційні системи автомобілів, нетбуки і телевізори зі з'єднанням з мережею. Всі платформи будуть використовувати ядро MeeGo, і різні рівні розширень UX (User eXperience) для кожного типу пристроїв.

Каркас інтерфейсу користувача заснований на Qt, але GTK+ і Clutter будуть включатися для забезпечення сумісності із застосунками Moblin. Залежно від пристрою, застосунки будуть забезпечені системою цифрового розповсюдження або Intel AppUp, або Nokia Ovi.

MeeGo забезпечує підтримку для процесорних архітектур ARM та Intel x86.

Bada

Bada має багаторівневу архітектуру. Ядром bada може бути Linux у потужних сматфонах або пропрієтарна операційна система реального часу у бюджетних варіантах. API платформи для розробки (офіційно надаються засоби розробки на C++) дає доступ до всіх рівнів платформи.[12]

Підсистеми Linux в Bada використовуються тільки на самому низькому рівні платформи, для роботи застосунків задіяний власний власницький фреймворк, що надає розробникам набір класів для керування телефоном, побудови користувацького інтерфейсу та організації обміну даними. Платформа побудована на основі сервіс-орієнтованої архітектури, в якій всі програми виступають в ролі сервісів, що підключаються. У застосунки можуть бути легко інтегровані компоненти, які забезпечують доступ до функцій виконання дзвінка, відправлення повідомлень, доступу до адресної книги. З цікавих можливостей, доступних для застосунків, можна відзначити детектор руху, тонке управління вібросигналів і систему розпізнавання облич.

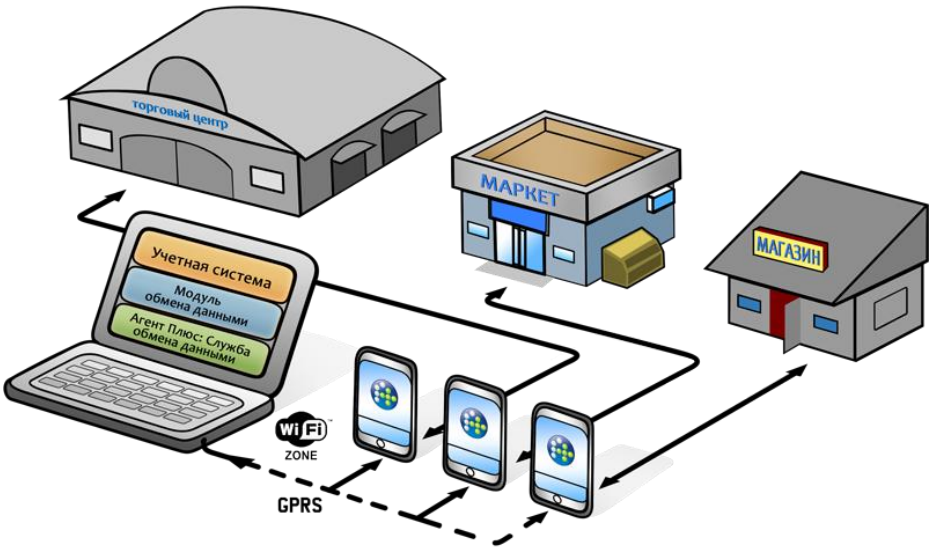
Для користувацького оточення в платформі Bada задіяна власницька оболонка TouchWiz власної розробки. Безпосередньо в користувацьке оточення інтегрований заснований на WebKit браузер з підтримкою Adobe Flash, що дозволяє використовувати його можливості з усіх програмах. Основний акцент в користувацькому оточенні робиться на простому і інтуїтивно зрозумілому візуальному оформленні, робота якого забезпечується спеціальним фреймворком, що надає єдине зовнішнє оформлення для всіх виконуваних на платформі програм.

Для платформи Bada у вільному доступі поширюються тільки сирцеві тексти змінених вільних компонентів і інструментарій Bada SDK. SDK складається з набору стандартних GNU-інструментів і інтегрованого середовища на основі Eclipse, яке містить симулятор

телефону, зневаджувач і візуальний будівник інтерфейсу. В якості мови програмування для створення застосунків підтримується C++ і JavaScript.

5. ОРГАНІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ МІС

Розглянемо приклад використання мобільних додатків для оптимізації бізнес-процесів підприємства.



Успішність сучасного бізнесу багато в чому залежить від інформації: своєчасної, повної, точної, доступної. Наявність такої інформації - це не тільки гарантія якісного надання послуг, але й успішний контроль діяльності співробітників, збільшення швидкості обробки замовлень, підвищення продуктивності праці.

Найбільш дієвим способом підвищення ефективності роботи мобільних співробітників є автоматизація їх бізнес-процесів. Для вирішення цього завдання компаніями "1С" та “Агент +” розроблено платформи для створення високопродуктивних мобільних додатків, що дозволяють оптимізувати бізнес-процеси на підприємстві та підвищити ефективність мобільних співробітників.

Сьогодні на базі платформи “Агент Плюс 2.0” розроблені чотири мобільні додатки.

Агент Плюс: Мобільна торгівля (Android)

Програмне забезпечення для автоматизації мобільної торгівлі за допомогою мобільних пристроїв (смартфони, планшетні комп’ютери) під управлінням ОС Android версії 2.2 і вище. Автоматизує різні види діяльності торгових представників і мерчендайзерів оптових і дистриб’юторських фірм, підприємств-виробників.

Автоматизована мобільна система Агент + (Windows Mobile)

Програмне забезпечення для автоматизації мобільної торгівлі за допомогою мобільних пристроїв (комунікатори, термінали збору даних, смартфони, кишенькові персональні комп’ютери) під управлінням ОС Microsoft ® Windows Mobile 2003/2003

SE/5.0/6.0/6.1/6.5. Автоматизує різні види діяльності торгових представників і мерчендайзерів оптових і дистриб'юторських фірм, підприємств-виробників.

Агент Плюс: Офіціант (Android)

Програмне забезпечення для автоматизації та оптимізації роботи офіціантів у закладах HoReCa (ресторани, кафе, бари) за допомогою мобільних пристроїв (смартфони, планшетні комп'ютери) під управлінням ОС Android версії 2.2 і вище.

Система моніторингу мобільних співробітників і об'єктів Map9

Програмне забезпечення для автоматизації роботи керівників, супервайзерів, менеджерів, які контролюють роботу мобільних співробітників (наприклад, торгових представників, експедиторів, страхових агентів). "Map9" встановлюється на настільний комп'ютер (ноутбук) керівника (менеджера, супервайзера) в офісі, і є ефективним інструментом контролю переміщень мобільних співробітників з використанням технології GPS.

Крім спеціалізованих систем, до категорії «Мобільні пристрої» входять програми створені для **взаємодії мобільних пристроїв (телефонів, смартфонів та планшетів) з персональним комп'ютером**. До цієї групи програмного забезпечення можна віднести програмне забезпечення для синхронізації пристроїв з ПК, доповнення та розширення їхнього функціоналу, тестування та налаштування.

iTunes — програвач, що використовується для синхронізації файлів з iPod, iPhone, і Apple TV.

Samsung Kies дає змогу підключити ПК до телефону Samsung, щоб полегшити синхронізацію даних і пошук програм.

Nokia PC Suite — безкоштовна програма для синхронізації даних між телефонами Nokia та комп'ютером.

Samsung PC Studio — пакет програм для редагування і перенесення персональної інформації на телефонах SAMSUNG (випущених по 2008 рік).

Samsung New PC Studio — програма для керування особистими даними та мультимедійними файлами телефонів Samsung (2009-2010 років).

Nokia Suite — безкоштовна програма для синхронізації даних телефонів Nokia з комп'ютером.

Sony PC Companion — безкоштовна програма, що виступає в якості порталу, на якому представлені функціональні інструменти і додатки Sony Xperia.

MyPhoneExplorer — безкоштовна програма для керування вмістом пристроїв на базі операційної системи Android і телефонами компанії Sony.

 **Microsoft ActiveSync** — безкоштовний синхронізатор даних мобільного пристрою під ОС Windows з комп'ютером.

 **Skype** — безкоштовна програма, що дозволяє робити телефонні дзвінки прямо з комп'ютера.

 **Mail.ru Агент** — інтернет-пейджер поштової служби Mail.ru, з багатьма просунутими можливостями.

 **ICQ**— популярна програма для спілкування в режимі реального часу через інтернет.

 **iSendSMS** — невелика безкоштовна програма, призначена для безкоштовної надсилання повідомлень в форматі SMS і MMS

Останні новини, які стосуються програмного забезпечення мобільних систем можна знайти на сайті http://www.itechnology.org.ua.

6. МОВИ ПРОГРАМУВАННЯ ДЛЯ МОБІЛЬНИХ СИСТЕМ.

Проаналізуємо основні технології, які використовуються для розробки додатків для мобільних телефонів.
Перша технологія - це Java 2 Micro Edition (J2ME). Це набір специфікацій і технологій, призначених для різних типів портативних пристроїв. Існують два основні напрями: Connected Device Configuration (CDC) і Connected Limited Device Configuration (CLDC). Напрям визначає тип конфігурації центральних бібліотек Java, а так само параметрів віртуальної машини Java (в якій будуть використовуватися додатки). Пристрої, які використовують технологію CDC будуть більш розвиненими, в якості прикладу можна навести комунікатори.
До пристроїв CLDC відносяться звичайні мобільні телефони, які апаратно володіють більш скромними
можливостями (ресурсами). Спеціальні режими дозволяють визначати функціональність конфігурацій для різних типів пристроїв.
Режим Mobile Information Device Profile (MIDP) призначений для CLDC портативних пристроїв з можливістю спілкування. Режим MIDP визначає функціональність - роботу користувальницького інтерфейсу, збереження налаштувань, роботу в мережі і модель додатка. CLDC і MIDP закладають основу реалізації J2ME [1]. Java-код інтерпретується безпосередньо самим пристроєм за допомогою так званої Java Virtual Machine. Цей механізм робить можливим вільне розповсюдження Java-додатків, так як вони працюють на всіх пристроях з аналогічною Java-платформою [2].
Програмування Java-додатків і на сьогоднішній день займає більшу частину, так як більшість мобільних пристроїв (в основному мобільні телефони) в світі мають вже встановлену Java-систему.
Наступна популярна сьогодні технологія – це технологія Qt. Вона в основному використовується в якості крос-платформного середовища, яке дозволяє використовувати написані з її допомогою додатки на різних пристроях і операційних системах, у тому числі Windows, Mac OS X, Linux, Symbian, Android та інших [3]. Починаючи з версії Qt 4.0 з'явилася можливість програмувати для мобільних пристроїв. Із зростаючою користувальницької базою Qt, зростає і потреба у вбудованих, мобільних додатках

і UI-розробників. Qt є однією з найбільш вдалих бібліотек для C++. Налагодження додатків, розроблених для мобільних пристроїв, відбувається за допомогою емулятора, який міститься в середовищі розробки. Таким чином, можна писати складні програми для мобільних пристроїв з використанням бібліотек C++ і підтримкою платформ.
На даний час остання версія це - Qt 5 бета. Для роботи Qt на мобільних пристроях необхідна установка певного фреймворку.
Ще однією технологією є технологія розробки Windows Phone SDK. На даний момент, остання версія інструментарію доступна у версії Windows Phone SDK 7.1 Release Candidate в ліцензії «Go Live» з можливістю розробляти свої додатки і публікувати їх в Windows Phone Marketplace. Windows Phone SDK 7.1
Release Candidate містить наступні компоненти [4]:
а) Windows Phone SDK 7.1;
б) Windows Phone Emulator;

в) Windows Phone SDK 7.1 Assemblies;

г) Silverlight 4 SDK and DRT;

д) Windows Phone SDK 7.1 Extensions for XNA Game Studio 4.0;

е) Expression Blend SDK for Windows Phone 7;

ж) Expression Blend SDK for Windows Phone OS 7.1;

з) WCF Data Services Client for Windows Phone;

і) Microsoft Advertising SDK for Windows Phone.

Код додатка, що розроблюється, описується на мові XAML. Хоча насправді - це просто XML файли з мовою розмітки XAML.

Платформа Windows Phone не просто чергова платформа для мобільних пристроїв. Вона містить у собі не тільки технологічну складову, але і повністю опрацьовану концепцію дизайну інтерфейсу і взаємодії з користувачем під назвою Metro-дизайн або стиль Metro [4].
Вся розробка під Windows Phone ведеться в середовищі Visual Studio. Для мобільних додатків під Windows Phone відладка та тестування відбувається за допомогою емулятора Windows Phone у середовищі розробки Windows Phone.
Ще одна технологія розробки - це iPhone SDK. Розробка під операційну систему iOS можлива тільки з використанням Mac OS X. Але в Інтернеті можна знайти статті, як можна проводити розробку безпосередньо на Macintosh і навіть на VM. Варто зауважити, що Apple надає інструменти безкоштовно, але платити доведеться за підписку розробника [5].
Для написання програм під iPhone пропонується використовувати Objective-C.При цьому є можливість писати так само і на C і C++ (для цього необхідно змінювати розширення файлів з .M на .Mm). Правда при цьому повністю звільнитись від Obj-C не вдасться, майже весь API розрахований саме на Obj-C, виключення складають наприклад OpenGL (хоча для його ініціалізації доведеться використовувати кілька рядків коду на Obj-C), так само повністю доступні стандартні бібліотеки C/C++ (так, наприклад, з файловою системою можна працювати як засобами SDK на Obj-C, так і використовуючи стандартну бібліотеку C для вводу / виводу (fopen (), fgetc (), etc)) [5].

Налагодження додатка відбувається за допомогою середовища XCode і емулятора iPhone встановленого в ній.
Для розробки під Android можна використовувати середовище Eclipse з встановленим плагіном ADT. Розробка ведеться на мові програмування Java. Є можливість налагодження з використанням емулятора вбудованого в ADT або безпосередньо на мобільному пристрої з ОС Android.
Існують різні версії SDK, які використовуються для написання коду для різних версій Android. В даний час велике поширення отримали версії 2.2 і 2.3. Підтримується майже повна зворотна сумісність версій.
Крім розробки на мові Java підтримується можливість більш низькорівневої розробки з використанням Android NDK (Native Development Kit) на мові C/C++.
Для написання додатків під Symbian можна використовувати мову
програмування C++. В основному даний підхід використовується для Symbian OS v6.1, 7.0, 7.0s і 8.0. Розробка для Symbian OS (якщо говорити про C++) зазвичай ведеться на ПК.
Середовище розробки – звична для багатьох програмістів Visual Studio, це також можуть бути IDE Metrowerks CodeWarrior Development Studio, Borland C++ BuilderX Mobile Edition, Carbide. C++ (відносно нова IDE, створена компанією Nokia на базі Eclipse), забезпечена додатковими інструментальними пакетами (SDK). Розробнику доступні практично всі звичні можливості, як створення програмне забезпечення (ПЗ), так і налагодження (трасування, перегляд змінних, стека викликів, структур класів). Відлагодження програми запускається в емуляторі Symbian OS. Цю підсистему правильніше було б назвати симулятором, оскільки імітуються не апаратні засоби, а лише програмне оточення (відповідні API операційної системи, реалізовані поверх API Win32). При цьому програмні модулі, які завантажуються в емулятор, являють собою виконані файли для архітектури x86 (не ARM, на базі якої побудовані смартфони), відповідне ПЗ для цільової платформи формується після підсумкової компіляції. Це передбачає певну специфіку (раніше була досить поширена ситуація, коли програма, нормально функціонувала в середовищі емулятора, відмовлялася працювати на реальному пристрої), але сьогодні емулятор забезпечує досить високу ступінь подібності та проблеми виникають лише при створенні програм, які нестандартно використовують API.
Платформа Android надає розробникам найбільшу свободу вибору ОС, на якій розробляти додатки. Android розробники можуть використовувати Windows, Mac або Linux. BlackBerry користувачі можуть вибирати між Mac, Windows і Linux.
Якщо ви хочете розробити для iOS і пристроїв Windows, єдиним варіантом є Mac і Windows відповідно.
Всі чотири мобільні платформи вимагають від розробників підписувати свої програми, перш ніж вони можуть бути представлені в магазинах постачальників додатків. Microsoft надає цілий ряд інструментів для надання допомоги у розвитку Windows Phone, але інструменти доступні тільки в Windows 7 і Vista. Розробка робиться в Visual Studio з комбінацією Silverlight / XAML, C # і Visual Basic NET, HTML / JavaScript і Expression Blend. Безкоштовні інструменти доступні для персонального використовування. Для професійних розробників потрібно придбати ліцензію для використання Visual Studio.
Для розробки крос-платформеного мобільного додатку можна також використовувати веб-технології, а саме HTML5, CSS3 та JavaScript. Розроблені веб-програми копіюють рідний стиль мобільної системи, але такі програми є доступними прямо з браузера (так само, як веб-сторінка) і тому використовують ті самі технології.

Дизайн мобільного додатку повинен бути максимально простий і зрозумілий. З огляду на те, що мобільні пристрої мають невеликий за розмірами екран, при проектуванні додатків для них не можна керуватися тими ж правилами, що і для ПК.

Основні вимоги до дизайну інтерфейсу мобільних додатків:

а) Мінімум елементів. Не слід перевантажувати невеликий простір дисплея мобільного пристрою великою кількістю об'єктів. Необхідно намагатися вмістити максимум функціоналу в лаконічний і дружній інтерфейс. Ця вимога є головною для розробки будь-якого ПЗ, а не тільки мобільного.

б) Управління сучасними телефонами з сенсорними екранами здійснюється за допомогою чуттєвого дотику. З огляду на те, що площа дотику пальця значно більшерозмірів покажчика комп'ютерної миші, а також стілуса, інтерфейс не повинен містити дрібних елементів. По-перше, вони погано помітні на невеликому екрані. По-друге, торкаючись деякого ділянки дисплея, користувач може натиснути не той елемент, який йому потрібен, що в кращому випадку може призвести до зниження зручності використання програми, а в гіршому - до небажаних наслідків.

в) Розмір всіх написів повинен бути достатнім для того, щоб користувач міг прочитати їх з відстані не менше 30 см.

г) Найбільш важливі і часто використовувані елементи інтерфейсу повинні знаходитися в центрі екрану і мати достатній розмір для того, щоб виділятися серед інших.

д) При перенесенні додатків з ПК на мобільний платформу можна обмежитися створенням зменшеної копії додатка. Необхідно оптимізувати весь інтерфейс, прибрати всі зайві елементи, згрупувавши схожі по функціоналу. При великій кількості об'єктів слід зробити додаткові «вікна», що змінюють один одного на дисплеї. На відміну від ПК, на мобільних платформах під вікнами розуміються елементи інтерфейсу, що займають весь простір екрану пристрою. Користувач здійснює переходи між такими вікнами за допомогою графічних елементів-навігаторів, або перетягуючи їх за допомогою пальця (в залежності від тієї або іншої платформи і переваг творців додатка).

При проектуванні дизайну мобільного додатку може виникнути необхідність враховувати культурні особливості регіону, для якого воно призначене (читання справа наліво або правильний підбір колірної гами).

У правильно спроектованому мобільному додатку повинні поєднуватися три основні властивості:

а) Зручність у використанні (інтуїтивний дизайн, об'єднання та використання всіх можливостей мобільного пристрою). Найбільш популярними мобільними платформами є iPhone і Android. Вони мають багато спільного. Грамотно спроектований додаток буде включати в себе той функціонал, який використовує особливості кожної з них.

б) Мобільний додаток повинен бути ціковим користувачу. Саме тому найбільшою популярністю на сьогодні користуються мобільні ігри. Кращий засіб змусити людину використовувати додаток з великою кількістю функціоналу -внести елемент розваги.

в) Корисність. Максимальний рейтинг мають ті додатки, які здатні бути дійсно потрібними.

7. РОЗРОБКА ПРОГРАМ ДЛЯ МОБІЛЬНИХ ПРИСТРОЇВ.

Основні характеристики середовища розробки для платформи Android

Головним скарбом Android як середовища розробки став її API. – Application Programming Interface

Android як нейтральна до додатків платформа надає можливість створювати програми, які стануть такою ж невід'ємною частиною телефону, як і компоненти, що поставляються в комплекті.

Наступний список ілюструє основні характеристики Android:

• відсутність витрат на використання ліцензії, поширення та розробку, а також будь-яких механізмів сертифікації готових програмних продуктів;

• доступ до Wi-Fi-пристрою;

• в мережах GSM, EDGE і 3G, призначених для телефонії і передачі даних, можна дзвонити або приймати дзвінки і SMS, відправляти і отримувати дані;

• комплексний API для роботи з навігаційними службами, наприклад GPS;

• повний контроль над мультимедійними пристроями, включаючи програвання або запис інформації з камери і мікрофону;

• API для роботи з сенсорними пристроями, наприклад акселерометром і компасом;

• бібліотеки для роботи з Bluetooth з можливістю передачі даних за протоколом P2P;

• передача IPC-повідомлень (IPC – *Inter-Process Communication)*;

• сховища для загальних даних;

• фонові додатки і процеси;

• віджети для робочого стола, Живі каталоги (Live Folders) і Живі шпалери (Live Wallpaper);

• можливість інтеграції результатів пошуку додатка в системний пошук;

• вбудований браузер на базі WebKit з відкритими початковими кодами і підтримкою HTML5;

• повна підтримка додатків, які використовують функціонал роботи з картами в своєму користувальницькому інтерфейсі;

• оптимізована під мобільні пристрої графічна система з апаратним прискоренням, що включає бібліотеку для роботи з векторною 2D-графікою і підтримку тривимірної графіки з використанням OpenGL ES 2.0;

• мультимедійні бібліотеки для програвання і запису аудіо, відеофайлів або зображень;

• локалізація за допомогою інструментів для роботи з динамічними ресурсами;

• набір програмних компонентів для повторного використання компонентів і заміщення вбудованих додатків.

Робота з апаратними ресурсами, включаючи камеру, GPS-навігатор і акселерометр

До складу Android входять бібліотеки API, які спрощують розробку програм, що використовують апаратні ресурси пристрою. Це означає, що не потрібно щоразу створювати спеціальні версії програми для різних пристроїв. Ви можете створити додаток на платформі Android, яке буде працювати на будь-якому сумісному з нею пристрої.

Середовище розробки для Android включає API для роботи з навігаційними пристроями (зокрема, з GPS-навігатором), камерою, звуковою системою, мережними з'єднаннями, Wi-Fi, Bluetooth, акселерометром, сенсорним дисплеєм і системою управління живленням. Більш детальна інформація про можливості API для роботи з апаратним забезпеченням Android розглянемо пізніше.

Вбудовані служби Google Maps, Geocoding (геокодування) і сервіси навігації.

Android підтримує роботу з картами, а значить можна створювати навігаційні додатки, які будуть ефективно використовувати мобільні переваги пристроїв під управлінням Android. Дана платформа дозволяє програмам включати сервіс Google Maps в

інтерфейс і забезпечує повний доступ до карт, якими можна управляти програмно і при необхідності забезпечувати коментарями, використовуючи багаті можливості графічної бібліотеки Android.

Навігаційні сервіси платформи працюють з GPS і технологією визначення положення по базовим станціям мереж GSM від Google, за допомогою яких встановлюється поточне місцезнаходження пристрою.

Дані сервіси дозволяють абстрагуватися від особливостей тієї чи іншої технології, при цьому ви задаєте мінімальний набір налаштувань (наприклад, точність або вартість) і вибираєте потрібну технологію. Крім цього платформа гарантує, що ваші навігаційні програми будуть працювати незалежно від того, яку з технологій підтримує той або інший пристрій.

Для з'єднання карт з навігаційними сервісами до складу Android включений API для прямого і зворотного геокодування, який дозволяє знаходити на карті координати по заданій адресі або визначати адресу для певної позиції на карті.

Фонові служби

На платформі Android можна створювати додатки і служби, які працюють у фоновому режимі.

Сучасні мобільні пристрої, як правило, мультизадачності. Однак через невеликі розміри екранів ви можете бачити тільки один інтерактивний додаток. Платформи, які не підтримують фонову роботу додатків, обмежують час життя програм, які не потрібні вам постійно.

Фонові служби дозволяють створювати невидимі компоненти додатків, які в автоматичному режимі виконують будь-які операції без прямої участі користувача. Завдяки фоновим процесам можна працювати з подіями або регулярно проводити оновлення. Крім того, вони ідеально підходять для моніторингу біржових зведень та результатів ігор, відображення навігаційних повідомлень або фільтрації вхідних дзвінків і повідомлень.

Використання баз даних SQLite для зберігання та вилучення інформації

Для пристроїв, розміри яких не дозволяють використовувати великі обсяги пам'яті, як ніколи актуально швидке і ефективне збереження та вилучення інформації.

У кожного додатку, що працює на платформі Android, є доступ до легковагової реляційної бази даних SQLite. Ваша програма може використовувати всі переваги движка цієї бази даних для безпечного і ефективного зберігання інформації.

За замовчуванням окремі бази даних додатків ізольовані один від одного, тобто їх вміст може бути використано тільки додатком, яке створило ту чи іншу базу. Однак Джерела даних (Content Providers) забезпечують можливість спільного використання баз даних додатків.

Загальні дані та міжпрограмна взаємодія.

Android підтримує три технології передачі інформації з додатку будь-якому іншому джерелу: повідомлення, класи переходів і Джерела даних.

Повідомлення - це стандартні засоби, за допомогою яких мобільні пристрої що не будь повідомляють користувачеві. За допомогою API ви можете викликати звукові повідомлення, створювати вібрацію або відображати флеш-повідомлення на екрані пристрою, а також змінювати статус значків повідомлень в рядку стану.

Класи переходів - це механізм передачі повідомлень всередині додатків і між ними. З їх допомогою ви можете транслювати потрібну дію (наприклад, набір номера на телефоні або редагування контакту) по всій системі в інші додатки, які повинні його обробити.

Класи переходів - дуже важливий компонент ядра платформи Android.

Нарешті, ви можете використовувати Джерела даних, щоб відкрити доступ до баз даних програми. Вбудовані додатки, наприклад менеджер контактів, забезпечують доступ до інформації також через Джерела даних, так що ви можете створювати програми, які будуть зчитувати або змінювати ці дані.

Робота з віджетами, Живими каталогами та Живими шпалерами для розширення можливостей стандартного Робочого столу

Віджети, Живі каталоги і Живі шпалери покликані створювати динамічні компоненти додатків, які, з одного боку, відкривають інтерактивне вікно в програму, а з іншого - дозволяють відображати важливу або змінну в режимі реального часу інформацію прямо на Робочий стіл пристрою.

Ви як би створюєте динамічний ярлик на програму, яка дає можливість взаємодіяти з нею безпосередньо з Робочого столу, так що користувач в режимі реального часу має доступ до інформації, що цікавить без запуску програми.

Розширена підтримка мультимедіа і 2D/3D-графіки

Великі екрани і яскраві дисплеї високої дозволяючої здатності дозволили назвати мобільні телефони мультимедійними пристроями. Для використання усіх можливостей доступного апаратного забезпечення Android забезпечили графічними бібліотеками для двомірного малювання на полотні і роботи з тривимірною графікою допомогою OpenGL.

Android включає також комплексні бібліотеки для роботи із статичними зображеннями, відео-і аудіофайлами, в тому числі підтримку форматів MPEG4, H.264, MP3, AAC, AMR, JPG, PNG і GIF.

Оптимізоване управління пам'яттю і процесами

Управління пам'яттю і процесами в Android трохи незвичне. Як і платформи Java і. NET, Android використовує власне середовище виконання і віртуальну машину для управління пам'яттю додатку. Але на відміну від двох вищезгаданих платформ в Android середовище виконання управляє часом життя процесів. Це дозволяє підвищити чутливість додатків шляхом зупинки або примусового завершення процесів, якщо необхідно звільнити ресурси для більш пріоритетних програм.

У цьому контексті високий пріоритет має програма, з якою в даний момент працює користувач. У середовищі, де програми не можуть управляти своїм часом життя, важливо забезпечити готовність додатків до швидкого завершення із збереженням їх чутливості, поновлення та перезапуску у фоновому режимі, якщо така необхідність виникне.

Про Альянс відкритих мобільних пристроїв (Open Handset Alliance, OHA)

Альянс відкритих мобільних пристроїв - це співтовариство з більш ніж 50 компаній, що включає виробників апаратного і програмного забезпечення, а також мобільних операторів. Серед найбільш значних членів Альянсу можна назвати компанії Motorola, HTC, T-Mobile і Qualcomm. Ось як формулюють основні ідеї OHA учасники цієї спільноти:

Прихильність відкритості, спільне бачення майбутнього і конкретні завдання для втілення мрії в реальність. Прискорення впровадження інновацій у сфері мобільних технологій і надання споживачам функціональних, менш дорогих і більш просунутих мобільних пристроїв. (Http://www.openhandsetalliance.com/).

Мета OHA - донести до покупців свій багатий досвід у сфері розробки програмного забезпечення. У зв'язку з цим представлена платформа, яка ідеально підходить для впровадження інноваційних технологій, яка відрізняється більш високою продуктивністю і покращеною якістю в порівнянні з існуючими. При всьому цьому, використовуючи дану платформу, ні розробники ПЗ, ні виробники мобільних пристроїв не повинні платити будь які ліцензійні відрахування.

8. СТВОРЕННЯ ІНТЕРФЕЙСУ КОРИСТУВАЧА.

Інженерна наука USABILITY займається питаннями створення і втілення ефективних людино-комп'ютерних інтерфейсів (ЛКІ) або іншими словами інтерфейсу користувача (ІК). Помилки та неефективність ІК можуть дорого обійтися як користувачу (від низької продуктивності до катастроф), так і виробникові (від втрати ринків до судових розглядів і фінансових претензій).

Назва походить від англійських слів USER (користувач) і ABILITY (здатність, вміння, а ще: талант, обдарування, компетенція та правомочність). Першими на ІК звернули увагу, звичайно, військові. Для них створення якісного інтерфейсу для льотчика, ракетника, і взагалі будь-якого оператора - найчастіше питання життя і смерті (насамперед самого оператора). Поступово на Заході склалося уявлення і наука про ІК. Тим не менш, до цих пір в середовищі розробників і особливо програмістів поширені наступні помилкові уявлення (міфи):

- «я бачив стільки інтерфейсів, що можу вважатися компетентним експертом в USABILITY». В дійсності найчастіше досягається рівень кваліфікованого користувача;
- «користувач завжди правий». Я зроблю те, що він скаже, а там його справа. Користувач ще менш кваліфікований в питаннях USABILITY, ніж програміст;
- «користувач нічого не розуміє». У всякому разі, він набагато краще знає, що йому треба, хоча не знає (не зобов'язаний!) як цього досягти;
- «чим більше, тим краще». У будь-яких вікні та формі слід застосовувати правило «7 ± 2»;
- основна вимога користувача «Зробіть нам красиво»! Може й так, але якщо неефективно, то спочатку захват, потім роздратування і потім повне відкидання.

Як вважає авторитетна в області USABILITY Nielsen Norman Group, торгові інтернет-системи втрачають до половини своїх потенційних клієнтів просто через те, що ті виявляються не в змозі розібратися, як скористатися такими системами. Поради фахівців з USABILITY часто досить прості: наприклад, скоротити число кліків, які повинен зробити потенційний клієнт торгової інтернет-системи, щоб оформити свою покупку. Або змінити колір, тип і розмір шрифту, для того щоб зробити текст на екрані комп'ютера більш читабельним. Але за допомогою цих простих змін іноді можна досягти приголомшливих результатів. Так, компанія Dell Computers, застосувавши принципи USABILITY до своєї системи продажів в Інтернеті, збільшила прибуток від інтернет-торгівлі з 1 млн. доларів в день (у вересні 1998 року) до 34 млн. доларів в день (в березні 2000-го). Інший приклад: за даними компанії Oracle, зміну навігаційної структури баз даних дозволило на 20% підвищити ефективність роботи адміністраторів. В той же час в Україні працює не більше двох-трьох десятків фахівців, які професійно займаються проблемами інтерфейсів. Тим не менше, в СНД існує деяка традиція ергономіки і USABILITY програмних продуктів. У радянські часи подібними дослідженнями займалися в основному в закритих НДІ, що працювали на «оборонку». І це не випадково, адже людська помилка при роботі зі складною військовою технікою може призвести до плачевних результатів. Там, де ризики людської помилки занадто великі, наприклад на транспорті, залізничному або повітряному, у використанні військової техніки, USABILITY давно надають особливого значення.

До цивільних областей, де USABILITY ПЗ критично важлива, можна віднести фінанси та торгівлю. При роботі в фінансових системах клієнти кожну хвилину ризикують чималими сумами, недарма перша в Росії команда фахівців з оптимізації інтерфейсів з'явилася саме на біржі. Але авіадиспетчерам, військовим і брокерам все ж легше - системи для них не тільки розробляють, але й аналізують професіонали.

А ось приватний користувач, підбираючи або замовляючи необхідний йому продукт, найчастіше залишається з проблемою USABILITY один на один, тому що зручність відіграє далеко для нього не першу роль і тому питання про якість інтерфейсу виникає не при покупці або замовленні системи, а в процесі її експлуатації. Проте, варто зазначити, що системи, які призначені для масового користувача, наприклад традиційні офісні пакети, ігри, мультимедіа, найчастіше мають дуже хороші інтерфейси. Причина тому - висока конкуренція на масовому ринку.

. Проблеми проектування інтерфейсів користувача (ІК)

Невід'ємною частиною будь-якого програмного продукту, який призначений для використання в інтерактивному режимі, є ІК. Він об'єднує в собі всі елементи і компоненти програми, які здатні впливати на взаємодію користувача з програмним забезпеченням.
Елементи користувацького інтерфейсу:

• засоби відображення інформації, пристрої та технології введення даних;
• відображувана інформація, формати і коди;
• командні режими, мова користувач-інтерфейс;
• сценарії діалогу і самі діалоги;
• зворотний зв'язок з користувачем;
• підтримка прийняття рішень у конкретній предметній області;
• порядок використання програми і документація на неї.

Проектування ІК перетворилося на самостійну проблему, яка найчастіше перевершує по складності проблему розробки кодів програми, і вимагає, як і процес проектування будь-якої складної системи, відповідних методів, засобів, і, природно, зусиль кваліфікованих фахівців. Саме застосування терміну ЛКІ являє собою спробу розробників програмного забезпечення відокремити, принаймні концептуально, функціональне призначення програмних продуктів від проблем, пов'язаних з організацією взаємодії користувача з цими продуктами. Такий поділ є необхідною умовою створення «дружніх» інтерфейсів по відношенню до користувачів програмних продуктів. Зауважимо, що саме поняття дружності - це вектор, який містить відповідно до міжнародної класифікації сім вимог до ІК:

- відповідність завдань, що вирішуються користувачем;
- легкість застосування;
- керованість;
- відповідність очікуванням користувача;
- стійкість до помилок;
- адаптованість / індивідуалізоване;
- легкість вивчення.

Згідно з американською класифікацією таких компонент дев'ять.

Методологічні основи ІК

Сучасні підходи до проектування ІК базуються на визначеній методологічній основі, яка виділяє три ключові проблеми організації процесу проектування ІК:

- ідентифікація інформації, необхідної для проектування;
- визначення і структурування власне процесу проектування;
- цілі та порядок проведення експертизи ефективності ІК.

Узагальнена структура інформації для проектування інтерфейсу АС:

1. Оператор:

- ідентифікація;
- пріоритетність;
- сфери інтересів;
- очікування;
- вік, стать;
- антропометрія, відхилення;
- культура;
- мова (яка може залежати від завдання);
- ідентифікація груп;
- фактори стресу;
- асоціативні та ролеві моделі;
- освіта;
- рівень знання комп'ютера.

2. Організація праці:

- стратегія інформаційної системи;
- робочий процес;
- ідентифікація завдання;
- атрибути завдання; процедури, методи;
- вимоги інформації;
- відповідальність;
- процес рішення;
- рівень автоматизації;
- розпорядок роботи;
- робоче навантаження;
- гарантії;
- заробітна платня, стимули, вигоди;
- заохочення;
- набір і навчання;
- активність профоб'єднань.

3. Фізична середа:

- основна робоча середа (макросередовище);
- робоче місце (мікросередовище);
- обладнання, інструментальні засоби тощо;
- підтримка та експлуатація;
- освітленість;
- клімат;
- шум;
- вібрація;
- випромінювання;
- токсичні речовини.

Більшість методів проектування, що реалізують ту чи іншу методологію, сьогодні тяжіє до технологічних рішень та інструментальних засобів, і здебільшого орієнтовано на реалізацію технічних рішень. При цьому майже не приділяється уваги розгляду таких важливих завдань, пов'язаних з проектуванням, як збір та аналіз інформації, синтез і оцінка проектних рішень на ранніх етапах проектування. А між тим, інструментальні

програмні засоби є лише інструментом і самі по собі не гарантують правильних проектних рішень.

Інтерфейс за допомогою інструментальних засобів «збирається» з готових елементів – «кубиків». Однак з одних і тих же кубиків можна «побудувати» інтерфейси різної якості та з різними ергономічними властивостями. Тут багато що залежить від проектувальників, їх знань, досвіду і кваліфікації в області організації людино-комп'ютерної взаємодії.

Хто може проектувати ІК

Дуже часто подібним «конструюванням» доводиться займатися програмістам, рідше прикладним фахівцям, наприклад, технологам. Насправді, ж рішення цих проблем відноситься до сфери компетенції фахівців з людино-комп'ютерних інтерфейсів, або, як кажуть за кордоном, фахівцям в області USABILITY. Зараз вже в більшості посібників з розробки ІК, що включають компоненти візуального програмування вказується, що розробкою ІК повинні займатися спеціально підготовлені співробітники і категорично не рекомендується залучати для цієї мети виключно програмістів.

При цьому можна відзначити одну цікаву особливість. Більшість фахівців, особливо програмістів, які досить довго просиділи за екранами моніторів, вважають себе досвідченими цінувальниками та знавцями людино-комп'ютерних інтерфейсів. Переконати їх у зворотному буває вкрай важко, хоча по суті вони є користувачами, правда, дуже кваліфікованими.

Висновок очевидний - залучення консультантів зі сторони. Їх небагато, але вони є. Підхід найбільш доцільний, коли робота разова. Якщо ж фірма реалізує безліч проектів або в рамках одного проекту передбачається тривала еволюція системи, то бажано мати (готувати) своїх фахівців.

Методологія створення ІК існує в зародковому стані. Програміст змушений перетворюватися на художника і психолога одночасно, тому потрібно адекватне розуміння потреб людини. Авторами найбільш вдалих інтерфейсів є фахівці одночасно в області обчислювальної техніки, психології та дизайну.
Хто ж може проектувати ІК? У цій комплексній роботі повинні брати участь:

- постановник задач, який досконально знає вирішувані завдання;
- інженерний психолог (ергономіст), що враховує особливості психофізіології людини;
- програміст, який створює додаток;
- дизайнер, який робить інтерфейс привабливим.

Природно, в одній людині, такі різнорідні знання практично не зустрічаються. Отже - бригада. Основним, звичайно ж, є постановник завдань. Кращим постановником є людина, яка вміє ефективно спілкуватися з Замовником.

Нормативно-технічна база – стандарти ІК

За кордоном існує ціла індустрія проектування ІК, в будь-якій фірмі, що займається розробкою програмного забезпечення існують спеціалізовані ергономічні служби або відділи, є величезна кількість методичних посібників та, що особливо важливо, велике число нормативно-технічної документації, в тому числі на рівні міжнародних стандартів. У СНД ж ситуація зовсім інша: мізерно мало кваліфікованих спеціалістів, практично немає довідкової літератури, не застосовуються професійні стандарти.

Ситуація ускладнюється ще й тим, що інтерфейси більшості інструментальних систем, з якими доводиться мати справу розробникам програмного забезпечення, відносяться до класу так званих офісних інтерфейсів, які беруть свій початок від офісних систем. Інтерфейси же програм АСНД (АС наукових досліджень) і АСУТП (АС управління технологічним процесом), наприклад, відносяться до класу інформаційно-керуючих, ергономіка яких кардинально відрізняється від офісних і, як правило, абсолютно незнайома нашим розробникам.

У тих випадках, коли фірма реалізує безліч проектів необхідно підготувати також одне або декілька корпоративних керівництв з проектування ІК. Цей підхід достатньо ефективний і не пов'язаний зі значними витратами, тому має сенс зупинитися на ньому докладніше.

Стандартизація є одним з найбільш доступних інструментів якісних інтерфейсів. Сучасна концепція стандартизації людино-комп'ютерних інтерфейсів базується на ієрархії концепції:

1. Проектування користувацького інтерфейсу.
2. Програмна реалізація інтерфейсу.
3. Стилі інтерфейсів.

Перший рівень ієрархії визначає основні методи та інструменти проектування графічного інтерфейсу, другий - особливості його програмної реалізації, зокрема, особливості реалізації та сумісності для різних комп'ютерних платформ, а третій - особливості реалізації для різних класів систем.
В цілому розробка і дотримання ІК стандартів дозволяють забезпечити:

- високу продуктивність роботи користувачів, тому що користувачі отримають і будуть використовувати легкосприйнятні засоби вирішення їх професійних завдань з мінімальним ризиком зустріти труднощі або перешкоди;
- малий час навчання - користувачу буде достатньо вивчити одну систему і отримані знання та навички стануть базовими при використанні інших систем;
- скорочення часу розробки - не буде необхідності проектувати кожну компоненту окремо, оскільки з'являться попередньо розроблені відповідно до нормативів універсальні програмні компоненти. Використання цих компонент в поєднанні з керівництвами по людино-комп'ютерним інтерфейсам, формам, що деталізуються, застосування цих компонент на рівні конкретних типів додатків, дозволить мінімізувати відмінності інтерфейсів і буде сприяти скороченню часу розробки.

Стандарти можуть мати різну форму і можуть одночасно служити цілям:

- керівництва є джерелами проектних рішень для проектувальників у частині розробки форматів відображуваної інформації та інтерактивних процедур;
- керівництва забезпечують загальний підхід до систематизованого проектування на основі фундаментальних принципів ергономічного проектування;
- керівництва сприяють розширенню рамок критеріїв персонального вибору та зменшення вимог до підготовки та якостей операторів всіх систем.

Для досягнення цих цілей в процесі розробки керівництв ставляться наступні завдання:

- ідентифікувати та встановити всі функції і завдання, які вирішуються системою і операційним середовищем. Це сприяє розумінню загальної динаміки систем;

- виконати аналіз можливостей і обмежень користувачів системи. Це дозволить краще розподілити функції між людиною і машиною і гарантує виконання приписаних функцій з боку людини;
- систематизувати правила проектування інтерфейсу.

Стилі інтерфейсу

Зауважимо, що корпоративні керівництва з проектування ІК мають всі без винятку відомі фірми, що займаються програмуванням. Керівництва деяких фірм стали основою відповідного ергономічного стилю інтерфейсу. Прикладами найбільш часто використовуваних комерційних стилів є:

- OSF/Motif,
- SUN/OpenLook;
- Microsoft Windows;
- IBM;
- Apple.

Наші розробники з ряду причин найкраще знайомі зі стилем Microsoft Windows. Цей стиль найкраще пристосований до офісних додатків і менш за все - до діяльності типу оперативно-тактичного управління.

Між всіма стилями існують відмінності, які можуть бути згруповані в наступні три досить широкі категорії:

- термінологія - це розходження в назвах, що присвоюються та описують функції та елементи. Основна відмінність полягає у використанні різних назв і формулювань для опису еквівалентних або подібних функцій та елементів. Але іноді одна й та ж назва використовується для абсолютно різних елементів. Прикладом різних назв для подібних елементів: Motif використовує термін «радіокнопка», а OpenLook – «виключний перемикач».
- зовнішність - це розходження в графічному поданні;
- сприйняття - це розходження в діях, які повинен вжити користувач, взаємодіючи з додатком. Відмінності пов'язані із застосуванням і використанням кнопок миші, функціональних клавіш, мнемоніки і прискорювачів, деяких підходів по керуванню.

. Вимоги до ІК. Принципи реалізації інтерфейсу

Мінімізація зусиль користувача при виконанні роботи:

- скорочення тривалості операцій читання, редагування і пошуку інформації;
- зменшення часу навігації і вибору команди;
- підвищення загальної продуктивності користувача, що полягає в обсязі оброблених даних за певний період часу;
- збільшення тривалості стійкої роботи користувача тощо.

Стильова гнучкість - можливість використовувати різні інтерфейси з одним і тим же додатком, на практиці реалізується за допомогою таблиці стилів, в тому числі можливості вибору користувачем власних установок ІК (колір, ікони, підказки тощо).

Нарощування функціональності - можливість розвивати додаток без руйнування (тобто залишаючись в рамках) існуючого інтерфейсу.

Масштабованість - можливість легко налаштовувати і розширювати як інтерфейс, так і сам додаток при збільшенні кількості користувачів, робочих місць, обсягу і характеристик даних.

Адаптивність до дій користувача - додаток повинен допускати можливість введення даних і команд множиною різних способів (клавіатура, миша, інші пристрої) і багато варіативністю доступу до прикладних функцій (ікони, «гарячі клавіші», пункти меню), крім того, програма повинна враховувати можливість переходу і повернення від вікна до вікна, від режиму до режиму, і правильно обробляти такі ситуації.

Незалежність у ресурсах - для створення користувацького інтерфейсу повинні надаватися окремі ресурси, спрямовані на зберігання і обробку даних, необхідних для підтримки користувача (словники користувача, контекстно-залежні списки, набори даних за замовчуванням або за останнім запитом, історії запитів тощо).

Переносимість - при переході на іншу апаратну (програмну) платформу, повинно здійснюватися автоматичне перенесення і користувацького інтерфейсу, і кінцевого додатка.

Етапи проектування ІК

Розробка ІК ведеться паралельно розробці програмного продукту в цілому і в основному передує його впровадженню. Процес розробки ергономічного ІК розбивається на наступні етапи:
1. Аналіз виробничої діяльності користувача, визначення і специфікація його бізнес-функцій. Формулювання вимог до роботи користувача. Побудова користувацької моделі даних (ERD), формування робочих місць.
2. Проектування ІК - вибір показників оцінки користувацького інтерфейсу. Розробка узагальненого сценарію взаємодії користувача з системою (функціональної моделі) і його попередня оцінка користувачами та Замовником (паперовий прототип ІК). Коригування і деталізація сценарію взаємодії, вибір і доповнення стандарту (керівництва) для побудови прототипу. Розробка макетів і прототипів ІК та їх оцінка у діловій грі, вибір остаточного варіанта. При проектуванні користувацького інтерфейсу наведена вище послідовність не є строго обов'язковою. Проектувальник може уявити діалог в екранних формах. Однак на цьому етапі головне погодити та затвердити не вид екрану (це вторинне і відображає швидше смак і майстерність розробника), а саме:

- структури і домени даних, що вводяться, коректуються і видаляються, дані NOT NULL;
- дії здійснювані при цьому користувачем, ІС, СКБД і при необхідності операційною системою;
- умови і шляхи переходу;
- умови підтримання цілісності та бізнес-правил;
- тригера;
- збережені процедури;
- навчання та/або функції допомоги користувачу;
- на цьому ж етапі узгоджуються SQL-запити.

Найбільш поширеною помилкою розробника є саме відсутність чіткого опрацювання виконуваних дій. Без цього подальша реалізація виявляється неузгодженою і може виявитися не відповідної кваліфікаційним вимогам, а на практиці вимогам користувача.
3. Реалізація ІК в коді, створення тестової версії (візуалізація). Розробка засобів підтримки користувача (користувацькі словники, підказки, повідомлення, допомога тощо) та їх вбудовування в програмний код.

4. Випробування ІК - Usability тестування тестової версії ІК по набору певних показників. Підготовка документації користувача та розробка програми навчання.

Аналіз діяльності користувача

Необхідно ретельно продумати і усвідомити сценарій взаємодії програми з користувачем, привівши його до оптимальної (щодо розглянутих показників) системи виконання завдань, і реалізувати ІК у відповідності з цією системою.

Для того, щоб розібратися в технології вирішення завдань користувачем, розробнику необхідно з'ясувати наступні моменти (досліджуючи діяльність користувача):

- яка інформація необхідна користувачеві для вирішення завдання?
- яку інформацію користувач може ігнорувати (не враховувати)?
- спільно з користувачем розділити всю інформацію на сигнальну, що відображається та редагується, пошукову і результуючу;
- які рішення користувачу необхідно приймати в процесі роботи з програмою?
- чи може користувач здійснювати кілька різних дій (вирішувати кілька завдань) одночасно?
- які типові операції виконує користувач при вирішенні задачі?
- що станеться, якщо користувач буде діяти не згідно з написаним алгоритмом, пропускаючи ті чи інші кроки або обходячи їх?

Пооперaційний аналіз ефективності ІК

Для оцінки продуктивності використовуються відповідні показники, що перевіряються фахівцями з ергономіки в процесі usability тестування робочого прототипу.

Формування таких показників відбувається в процесі визначення вимог до ІК при вивченні наступних питань:

- що від користувача потрібно в першу чергу?
- скільки інформації, що вимагає обробки, надходить користувачу за період часу?
- які вимоги до точності та швидкості введення інформації?
- на які операції користувач витрачає найбільше часу?
- чим ми можемо полегшити роботу користувача при вирішенні типових завдань?

Методи і критерії оцінки ІК

Оцінка ІК повинна бути об'єктивною. Тому її не можна довіряти ні користувачам (немає кваліфікації), ні дизайнерам (є зацікавленість). До даної роботи краще залучати фахівців з ергономіки. Для оцінки необхідного рівня зручності інтерфейсу використовуються спеціальні експертні анкети, опитувальники, формуляри, check-листи. В якості методів використовують:

- спостереження за користувачами до використання ІК, в процесі навчання і в роботі;
- відстеження мотивації користувача - думки вголос, пояснення своїх дій і намірів;
- постановка і протоколювання виконання тестових завдань.

Цілі та критерії оцінки користувацького інтерфейсу

Головна мета. З точки зору ергономіки (науку про ефективну взаємодію людини і техніки), найважливіше в додатку - створити такий користувацький інтерфейс, який

зробить роботу ефективною і продуктивною, а також забезпечить задоволеність користувача від роботи з програмою.

Ефективність роботи означає забезпечення точності, функціональної повноти і завершеності при виконанні виробничих завдань на робочому місці користувача. Ефективність роботи відображає обсяг витрачених ресурсів при виконанні завдання, як обчислювальних, так і психофізіологічних.

Створення ІК має бути націлене на показники ефективності людино-машинної системи, які можна виміряти кількісно і об'єктивно:

- продуктивність праці - визначається середньою кількістю вирішених завдань, отриманими за результатами роботи групи користувачів;
- точність роботи (кількість помилок) - показник точності включає відсоток помилок, які зробив користувач (число помилок набору, варіанти помилкових шляхів або відгалужень, число неправильних звернень до даних, запитів тощо);
- функціональна повнота - визначається тим, якою мірою вироблений користувачем продукт (результат роботи), відповідає пред'явленим до нього вимогам; відображає ступінь використання первинних і оброблених даних, списку необхідних процедур обробки або звітів, число пропущених технологічних операцій або етапів при виконанні поставленої задачі для користувача. Цей показник може визначатися через відсоток застосування окремих функцій в роботі;
- завершеність роботи - описує ступінь виконання виробничого завдання середнім користувачем за певний термін або період, частку (або довжину черги) незадоволених (необроблених) заявок, відсоток продукції, що знаходиться на проміжній стадії готовності, а також число користувачів, які виконали завдання у фіксовані терміни;
- простота освоєння - визначається часом освоєння інтерфейсу, виходу на продуктивний рівень.

Задоволеність користувача можна оцінити шляхом проведення опитування (експертної оцінки) користувачів і за ступенем стресу, втоми, емоційного стану - по фізіологічним і психологічним показниками.

Задоволеність користувача від роботи тісно пов'язана з комфортністю його взаємодії з додатком, і сприяє збереженню професійних кадрів на підприємстві Замовника за рахунок привабливості роботи на даному робочому місці.

Вимоги до зручності та комфортності інтерфейсу зростають зі збільшенням складності робіт і відповідальності користувача за кінцевий результат. Висока задоволеність від роботи досягається в разі:

- прозорої для користувача навігації і цільової орієнтації в програмі. Головне, щоб було зрозуміло, куди йдемо, і яку операцію програма після цього кроку зробить;
- ясності і чіткості розуміння користувачем текстів та значення ікон. У програмі повинні бути ті слова і графічні образи, які користувач знає чи зобов'язаний знати за характером його роботи або займаної посади;
- швидкості навчання при роботі з програмою, для чого необхідно використовувати переважно стандартні елементи взаємодії, їх традиційне або загальноприйняте розташування;
- наявності допоміжних засобів підтримки користувача (пошукових, довідкових, нормативних), в тому числі і для прийняття рішення у невизначеній ситуації (введення за замовчуванням, обхід «зависання» процесів тощо).

Зручний інтерфейс допомагає користувачеві справитися з втомою і напругою при роботі в умовах високої відповідальності за результат.

10 правил по проектуванню якісних ІК (по David F. Kelly):

1. Методологія створення ІК існує в зародковому стані. Програміст змушений перетворюватися на художника і психолога одночасно, тому потрібно адекватне розуміння потреб людини. Авторами найбільш вдалих інтерфейсів є фахівці одночасно в області обчислювальної техніки, психології та дизайну.
2. Суньте руки в кишені та забудьте на час про програмування. Займіться проектуванням. Єдиний спосіб створити хороший інтерфейс - розпочати розробку «з кінця» з інтерфейсу кінцевих користувачів.
3. Не слід чекати поки проект буде реалізований у вигляді програми. Набагато важливіше якомога швидше перевірити проект у користувача.
4. Структуруйте діалог. Створюйте підвікна.
5. Проектуйте легко сприймані не перевантажені екрани. Правило 7 ± 2.
6. Дотримуйтесь узгодженості шрифтів, розмірів, позначень у всіх вікнах. Стандартизуйте назви. Не виносьте на екран те, що в даний момент користувачеві не потрібно.
7. Починайте з рисунку на папері. Це швидше генерації вікон.
8. Використовуйте як мишу, так і клавіатуру.
9. Проаналізуйте інші проекти.
10. Не намагайтеся зібрати відразу всі вимоги користувачів. Проведіть дві 4-годинні наради і приступайте до розробки. Інші вимоги можна врахувати пізніше.

Основні відомості з інженерної психології

Прийом інформації оператором - абсолютний поріг сприйняття 1 кутова хвилина. Поле зору по ефективності сприйняття містить 4 зони:

- центральна 4 гр - ясне бачення;
- ясного бачення 30-35 гр - колір, форма, але без дрібних деталей;
- периферична 70-75 гр - об'єкт виявляється, але не сприймається;
- оперативна зона 10 гр.

Алфавіти кодування ІК:

- алфавітно-цифрові;
- кодування формою - довжина 20 знаків, оптимальна довжина - 12-15 знаків. Найбільш швидко сприймаються криві злами;
- колірне кодування - довжина - 15, оптимальне – 7;
- кодування величиною - довжина - 5, оптимальне – 3;
- кодування яскравістю - довжина - 4, оптимальне - 2. Яскравість 1:5;
- кодування частотою - довжина - 4, оптимальне - 2. Поріг 6-8 Гц.

Логарифмічний закон сприйняття:
Сприйняття = Lg (інтенсивність).
ІК як у прислів'ї - зустрічають по одягу, а якщо розуму немає, то і проводжають також.

Проблеми, що виникають на етапі розробки прототипу ІК та варіанти їх вирішення:
1. Врахування особливостей пристроїв введення/виводу інформації, що використовуються користувачем, наприклад:

- розмір екрану монітора;

- розподільча здатність екрану;
- колірна палітра;
- характеристики звукової (якість відтворення мови) і відеокарти (швидкість виведення при анімації);
- вид миші;
- тип клавіатури;
- необхідність додаткового обладнання (сканер штрих-коду, світлового пера, сенсорного екрану тощо).

2. Специфіка інтерактивних елементів, пов'язана з вибором платформи, стандартних бібліотек:

- програмна організація введення/виводу інформації;
- зміна і створення нових елементів форм (контролів);
- придбання нестандартних бібліотек у інших фірм.

3. Вибір технології та методів ведення діалогу програми з користувачем:

- ступінь активності користувача при взаємодії (автоматичний режим або перехоплення керування програмою на себе, Майстер, забезпечення доступу до всіх засобів інтерфейсу незалежно від дій користувача);
- ступінь врахування ситуації (контекстні підказки, меню подальших подій або об'єктів, запам'ятовування типових шляхів діалогу);
- відповідність очікуванням користувача (прогнозування, попередня обробка, попереднє форматування);
- стійкість, терпимість до помилок користувача шляхом виправлення типових помилок;
- дублювання в ручному режимі окремих функцій системи і додаткові контрольні процедури роботи окремих режимів;
- настройка ІК на різний рівень підготовки користувача (образність або метафоричність предметної області на противагу скорочень і гарячих клавіш);
- ступінь адаптивності ІК під уподобання користувача (зміна способу та порядку відображення, перекомпонування екрану, вибір окремих характеристик (стилю) тощо);
- настройка ІК на специфіку задачі (новий формат даних, зміна набору об'єктів, доповнення атрибутів об'єктів).

4. Розміщення інформації та керуючих елементів у полі екрана, у вікні. При композиції екрану необхідно враховувати обмежені розміри простору екрану, в зв'язку з чим виникає задача оптимального розташування максимально можливого обсягу інформації шляхом:

- логічною ув'язкою даних в залежності від алгоритму роботи користувача, а не орієнтацією на структуру і послідовність фізичних таблиць даних;
- визначення рівня «детальності – узагальненості» виведення інформації (знаходження компромісу між бажанням вивести багато записів одночасно і/або відразу побачити детальну інформацію про кожний з них);
- виділення важливої інформації на екрані;
- чіткого визначення основних і допоміжних блоків інформації;
- визначення статичних полів на екрані, а також полів, де інформація періодично змінюється;
- уникнення вікон, які перекриваються на екрані;
- застосування принципів гармонії при компонуванні екрану (симетрія, баланс мас, дотримання пропорцій, поєднання кольорів).

5. Формування зворотного зв'язку між користувачем і додатком:

- показ актуального стану системи, режиму роботи системи (автономного, штатного, захищеного тощо) і режиму взаємодії (наприклад, відображення, редагування або пошук даних);
- відображення окремих, важливих для робочої операції даних і показників;
- відображення дій користувача (натискання клавіш, запуск процесу, динаміка виконання процесу, отримання очікуваного та іншого результату);
- ясність та інформативність повідомлень системи.

6. Проектування панелей меню та інструментів і вибір пунктів в них:

- логічне і смислове угруповання пунктів;
- фіксована позиція панелей на екрані;
- обмеження на ширину списку виборів і кроків (глибини) меню;
- використання звичних назв, широко поширених ікон-піктограм, традиційних ікон-символів і акуратне введення скорочень;
- розміщення найбільш часто використовуваних пунктів (зазвичай на початку списку).

7. Розробка засобів орієнтації та навігації:

- легкість визначення свого місцезнаходження і вказівка напряму слідування;
- зручний перехід від узагальненого погляду до конкретних деталей (варіювання ступеня деталізації аналізованих об'єктів);
- швидкий пошук в списку або таблиці;
- вказівка на додатково існуючу інформацію і спосіб її отримання;
- використання засобів перегортання і прокручування.

8. Створення форм для введення даних:

- використання одного або декількох механізмів уведення в рамках режиму (клавіатура, миша, сканер штрих-коду, світлове перо тощо);
- визначення способів введення даних (таблиці, списки, проста форма, меню тощо);
- мінімізація обсягу введення;
- виділення редагованих обов'язкових і необов'язкових, а також нередагованих полів;
- використання механізмів швидкого введення (за замовчуванням, скорочення, з продовженням тощо).

Приклад стандарту ІК від компанії IBM

Для ознайомлення приведемо коротку характеристику стандарту американської фірми IBM, вживаного при проектуванні інтерфейсу користувача (ІК).
Вичерпний детальний опис або специфікація ІК повністю визначає структуру прикладної системи. При цьому з позицій користувача:

- якість ІК має вирішальне значення для забезпечення якості прикладної системи в цілому.
- всі вимоги, що стосуються взаємодії користувача з прикладною системою, які він ставить перед розробником, перш за все, повинні враховуватися при розробці ІК.

У відповідності з вимогами користувача простота, легкість вивчення, зрозумілість, однаковий доступ користувачів до функцій системи, логічність взаємодії з прикладною системою є базовими якісними характеристиками розробки.

Виходячи з вищевикладеного, можна сказати, цілісність, ефективність і якість ІК визначають успіх чи неуспіх тієї чи іншої розробки. Сучасні прикладні ІС є за своєю

природою - діалоговими, тобто припускають тісну взаємодію користувача з системою на протязі всього процесу рішення задачі.

З точки зору експлуатаційних характеристик прикладних програм значення для користувача інтерфейсу як засобу комунікації важко переоцінити (всі основні тенденції розвитку сучасних систем - удосконалення ІК).

Характеристики інтерфейсу майже повністю визначаються тривалістю часу, необхідного для навчання користувача роботі з системою, таким чином, в ІК з'являється і соціальний аспект. Послідовно розроблений і логічний ІК спрощує освоєння та використання системи, зменшує частоту помилок користувача і, завдяки цьому, збільшує ефективність її використання.

ІК визначає, як взаємодіяти з системою. Розробляючи ІК, в якості вихідних даних використовується архітектура програмного забезпечення системи, тобто ресурсів, що надаються системою і визначають її поведінку. Опис ІК є описом головних елементів системи і може вважатися специфікацією на систему в цілому.

Конструктори фірми IBM в Австрії розробили концепцію кінцевого користувача, в якій користувач виступає в головній ролі найважливішого чинника - процесу рішення задачі.

Підхід фірми IBM полягає у виконанні вимог відповідних стандартів форматів, протоколів і архітектур, і в наданні засобів розвитку і розширення систем при збереженні можливостей використання специфічних функцій конкретних систем. Ці архітектурні рішення були названі *System Applicacion Architecture* (SAA) - архітектура середовища для розробки додатків. SAA складається з трьох компонентів, названих відповідно:

- системою користувацьких інтерфейсів (Common User Access - CUA);
- системою комунікації (Common Communications Access - CCS);
- системою програмних інтерфейсів (Common Programming Interface - CPI).

CUA - це множина правил і принципів, що регулюють багато з основних аспектів розробки ІК. Цей компонент забезпечує інтерфейси кінцевого користувача, керує діалогом користувача. Саме він визначає сценарій взаємодії з системою.

Добре спроектований сценарій взаємодії з прикладної системою забезпечує простоту взаємодії і легкість використання. Уніфікація користувацьких інтерфейсів забезпечує їх застосування в різних обчислювальних середовищах. CUA забезпечує засоби організації вікна на екрані для взаємодії з прикладним процесом. CUA спроектований як система-посередник між користувачем і системою.

Наступні вимоги включені до CUA:

- використання вікон;
- проектування панелі, специфічне компонування панелі, панель типів, поля вибору й поля введення, керування курсором і прокрутка, колір і виділення;
- розробка діалогу в формі дій управління діалогом, і у формі діалогів спливаючих вікон;
- повідомлення та засоби Help надають допомогу користувачу;
- призначення клавіш;
- призначення користувацьких опцій;
- підтримка національної мови;
- термінологія.

Модель CUA для відображення інформації припускає віконне середовище. Ця модель розглядає безвіконні середовища як підмножину, в якому вікна є повним екраном і не можуть бути зроблені менше. Цей підхід забезпечує узгодженість між двома середовищами, коли вікна, що мають регульовані розміри, «максимізують» (робляться в розмірі на весь екран). CUA в поточний момент не визначає правила для управління вікном, такі як пересування і установка розмірів вікон.

Управління Уявленням забезпечує роботу з вікнами в середовищі персональних комп'ютерів. CUA визначає, як прикладні програми використовують вікна, визначаючи три типи вікон, які пов'язані з трьома типами діалогу: *первинним, вторинним* і *спливаючим*. Правила визначають спосіб появи кожного типу вікна, використання їх в діалозі та способи видалення.

Відобразимо положення, які розглядаються при розробці панелі. Модель CUA для відображення інформації визначає, що:

- робочий стіл - комп'ютеризоване представлення інформації одночасно зі створенням робочого середовища користувача;
- об'єктна орієнтація - зосереджено увагу користувача в процесі діалогу на дані та можливі дії з ними;
- вказівки та вибір - виділення варіантів з набору можливих варіантів, дія, яка значно простіше для користувачів, ніж пригадування. Крім того, використання інструментів вказівки значно простіше, ніж введення ланцюжка символів з клавіатури;
- використання графіки та піктограм - створення явного і конкретного процесу керування;
- організація вікон - інформація, надана користувачеві, відображається у вікнах. Панель - це специфічна комбінація і розташування інформації, а вікно - відображає механізм для цієї інформації. Прикладні розробники створюють панелі, такі як панелі меню, панелі для редагування тексту, які потім відображаються користувачеві в одне з трьох типів вікон, в залежності від контексту діалогу, в якому відображається панель;
- керування взаємодією - надання користувачу максимально можливої свободи взаємодії з постійним контролем над його діями і зменшенням обсягу інформації, що запам'ятовується.

Таким чином, можна відзначити, що інтерфейс людина-комп'ютер включає такі аспекти обчислювальної системи, які стосуються безпосередньо користувача.

Це важливий чинник, який забезпечує успішну роботу обчислювальної системи, так як ергономічні (як фізичні, так і психологічні) характеристики інтерфейсу справляють істотний вплив на продуктивність користувача.

Для зниження стресових ситуацій система повинна відповідати як фізичному стану користувача, так і його розуміння задачі; це вимога ускладнюється тим, що з більшістю систем працює багато користувачів і вимоги індивідуального користувача змінюються в міру його знайомства з системою.

Теоретичні основи побудови інтелектуалізованих ІК фірми IBM

Інтелектуалізовані ІК призначені для більш швидкого та ефективного спілкування людини - користувача з комп'ютером. У цьому спілкуванні істотне значення має діалогова взаємодія користувача та комплексу засобів автоматизації.
На практичному рівні інтерфейс - це набір прийомів взаємодії з комп'ютером. На теоретичному рівні інтерфейс включає три основні поняття:

- спілкування комп'ютера з користувачем (мова представлення);
- спілкування користувача з комп'ютером (мова дій);
- подання інтерфейсу користувача (графічна або текстова мова опису інтерфейсу).

Спосіб спілкування комп'ютера з користувачем, *мова подання*, визначається додатком, прикладною програмною системою. Додаток керує доступом і обробкою інформації, представленням її в зрозумілому для користувача вигляді.

Користувач повинен розпізнати інформацію, яку надає комп'ютер, зрозуміти, проаналізувати її і перейти до відповіді. Відповідь реалізується через інтерактивну технологію, елементами якої можуть бути такі дії, як вибір об'єкта за допомогою клавіш або миші. Все це складає другу частину інтерфейсу, а саме - *мову дій*.

Узгодженість інтерфейсу

Ефективність інтерфейсу полягає у швидкому, наскільки це можливо, розвитку у користувачів простої концептуальної моделі взаємодії. У CUA (Common User Access) це досягається через узгодженість. Концепція узгодженості полягає в тому, що при роботі з комп'ютером у користувача формується система очікування однакових реакцій на однакові дії, що постійно *підкріплює* (стимул-реакція) модель інтерфейсу, яка налаштовується. Іншою складовою інтерфейсу є властивість його конкретності та наочності. Воно забезпечується застосуванням в панелях різних кольорів та інших виразних засобів. Ідеї та концепції потім знаходять фізичне відображення на екрані, з яким безпосередньо взаємодіє користувач.

Три аспекти узгодженості

Інтерфейс може бути узгоджений у трьох аспектах або категоріях:

- фізична узгодженість;
- синтаксична узгодженість;
- семантична узгодженість.

Фізична узгодженість відноситься до технічних засобів: схема клавіатури, розташування клавіш, використання миші. Наприклад, для клавіші F3 фізична узгодженість має місце, якщо вона завжди знаходиться на одному і тому ж місці, незалежно від обчислювальної системи. Аналогічно кнопка вибору миші буде фізично узгоджена, якщо вона завжди розташовується під вказівним пальцем.

Синтаксична узгодженість відноситься до послідовності та порядку появи елементів на екрані (мова представлення) і послідовності запитів (мова дій). Наприклад, матиме місце синтаксична узгодженість, якщо заголовок панелі завжди розміщується в центрі та на верху панелі.

Семантична узгодженість відноситься до значення елементів, які складають інтерфейс. Наприклад, що означає «Вихід»? Де користувачі запитують вихід і що потім відбувається?

Переваги узгодженого інтерфейсу

Погоджений інтерфейс дає користувачам і розробникам економію часу та коштів.

Користувачі виграють від того, що знадобиться менше часу, щоб навчитися використовувати додатки, а потім для виконання роботи. Погоджений інтерфейс скорочує число помилок користувача і сприяє тому, що користувач відчуває себе з системою комфортніше.

Погоджений ІК вигідний і розробникам додатків, він дозволяє виділити загальні блоки інтерфейсу, стандартизувати елементи інтерфейсу і взаємодії з ними. Ці блоки дозволяють програмістам простіше і швидше створювати і змінювати додатки. Хоча користувацький інтерфейс встановлює правила для елементів інтерфейсу та інтерактивної взаємодії, він припускає досить високий ступінь гнучкості.

Діалогова взаємодія

Діалогова взаємодія (діалог) являє собою регламентований обмін інформацією між людиною і обчислювальною машиною, що здійснюється в реальному масштабі часу і спрямований на спільне вирішення конкретного завдання.

Ініціатором діалогового процесу є людина, яка вибирає мету і в якійсь мірі може впливати на способи її досягнення за допомогою запитів. Система забезпечує адекватну реакцію на запити користувача. Діалог є автоматизованим методом вирішення слабоформалізованих завдань. При цьому розподіл функцій між людиною і комп’ютером наступний: людина ставить завдання, а система надає засоби для вирішення підзадач. Потім виконує об'єднання результатів, а прийняття проектних рішень залишається за людиною.

Для людино-машинної взаємодії характерно істотне розходження властивостей партнерів діалогу. Людина, як учасник діалогу, має певні психофізичні потреби, володіє фізичними й моторними навичками, мовними знаннями і досвідом діалогового спілкування, а також може мати різну підготовленість у предметній області. Крім того, при розробці діалогу доводиться враховувати властивості людської особистості, ставлення користувача до системи і його очікування від роботи з системою. З урахуванням цих факторів користувачі системи підрозділяються на кілька категорій:

- керівники, які здійснюють прийняття рішень і керування проектними роботами;
- «проектувальники - непрограмісти», які працюють в конкретних прикладних областях без використання мовних засобів;
- «проектувальники - програмісти», які формують завдання на вхідних мовах пакетів і мовах програмування;
- прикладні програмісти, які розробляють або розширюють компоненти прикладних підсистем;
- системні програмісти, які супроводжують компоненти забезпечення і систему в цілому.

Додаткові відомості про ергономіку ІК можна отримати з джерел:

- ергономічні стандарти, які описують вимоги до процесу і процедур розробки та оцінки користувацького інтерфейсу, основний стандарт - ISO 9241 - (частини 1,2, 11);
- стандарти, що визначають вимоги до продуктів і елементам дизайну користувацького інтерфейсу, відображені у випусках ISO 9241 - (3-10, 12-17);
- керівництва по дизайну ІК від виробників програмних платформ, таких як Microsoft, Sun, Apple;
- внутрішні корпоративні керівництва і стандарти;
- пропозиції та рекомендації фахівців з ергономіки.

9.БАЗИ ДАНИХ ТА ДЖЕРЕЛА ДАНИХ

Історія, роль та значення мови SQL

SQL (зазвичай вимовний як «СІКВЕЛ» або «ЕСКЮЕЛЬ») символізує собою Структуровану Мову Запитів. Це - мова, яка дає Вам можливість створювати і працювати в реляційних базах даних, які є наборами зв'язаної інформації, що зберігається в таблицях.

Інформаційний простір з кожним днем стає більш уніфікованим. Це призвело до необхідності створення стандартної мови, яка могла би використовуватися у великій кількості різних видів комп'ютерних середовищ. Стандартна мова дозволяє користувачам, які знають один набір команд, використовувати їх для створення, знаходження, зміни та передачі інформації - незалежно від того, чи працюють вони на персональному комп'ютері, ноутбуці або смартфоні. У нашому все більш і більш взаємопов'язаному комп'ютерному світі, користувач з такою мовою, має величезну перевагу у використанні та узагальненні інформації з ряду джерел за допомогою великої кількості способів.

Елегантність і незалежність від специфіки комп'ютерних технологій, а також її підтримка лідерами промисловості в області технології реляційних баз даних, зробило SQL (і, ймовірно, протягом осяжного майбутнього залишить її) основною стандартною мовою. З цієї причини, кожен, хто хоче працювати з базами даних 22 століття, повинен знати SQL.

Стандарт SQL визначається ANSI (Американським Національним Інститутом Стандартів) і в даний час також приймається ISO (Міжнародна Організація по Стандартизації). Однак, більшість комерційних програм баз даних розширюють SQL без повідомлення ANSI, додаючи різні особливості в цю мову, які, як вони вважають, будуть вельми корисні. Іноді вони дещо порушують стандарт мови, хоча хороші ідеї мають тенденцію розвиватися і незабаром ставати стандартами «ринку» самі по собі в силу корисності своїх якостей.

Мова SQL стала фактично стандартною мовою доступу до баз даних. Всі СКБД, що претендують на назву «реляційні», реалізують той або інший діалект SQL. Багато нереляційних систем вже також мають засоби доступу до реляційних даних. Метою стандартизації є переносимість додатків між різними СКБД.

Слід зауважити, що в даний час, жодна система не реалізує стандарт SQL в повному обсязі. Крім того, у всіх діалектах мови є можливості, які не є стандартними. Таким чином, можна сказати, що кожен діалект - це підмножина деякого стандарту SQL. Це ускладнює переносимість додатків, розроблених для одних СКБД в інші СКБД.

Мова SQL оперує термінами, що декілька відрізняються від термінів реляційної теорії, наприклад, замість «відношень» використовуються «таблиці», замість «кортежів» - «рядки», замість «атрибутів» - «колонки» або «стовпці».

Стандарт мови SQL, хоча і заснований на реляційній теорії, але в багатьох місцях відходить він неї. Наприклад, відношення в реляційній моделі даних не допускає наявності однакових кортежів, а таблиці в термінології SQL можуть мати однакові рядки. Є й інші відмінності.

Мова SQL є реляційно повною. Це означає, що будь-який оператор реляційної алгебри може бути представлений оператором SQL.

Ми будемо, в основному, слідувати стандарту ANSI, але одночасно іноді показуватимемо і деякі найбільш загальні відхилення від цього стандарту.

Точний опис особливостей мови наводиться в документації на СКБД, яку Ви використовуєте. Далі всі SQL-приклади ми будемо розглядати на основі SQL системи Oracle 11g Express Edition, що відповідає стандартам ANSI-92 і частково SQL-2003.

Мови опису даних і маніпулювання даними

Існують *три аспекти роботи з даними*:

- визначення даних;
- маніпулювання (обробка) даних;
- керування даними (адміністрування даних).

Тому будь-яка СКБД має в своєму складі мову визначення даних, мову маніпулювання даними та мову адміністрування даних. Ці мови не обов'язково існують як окремі компоненти, а можуть включатися як частини в мову СКБД, наприклад, в SQL. Тому, в мову SQL в якості складових частин входять:

- мова визначення даних (Data Definition Language, DDL);
- мова маніпулювання даними (Data Manipulation Language, DML);
- мова керування даними (Data Control Language, DCL).

Підкреслимо, що це не окремі мови, а різні команди однієї мови. Такий поділ проведено тільки лише з точки зору різного функціонального призначення цих команд.

Мова визначення даних

Мова визначення даних (МВД) являє собою мову, яка призначена для опису схеми БД або її частини. За допомогою МВД виконується опис типів даних, їх структур та зв'язків між ними. МВД не є процедурною мовою. Вихідні тексти (опис даних), написані на цій мові після трансляції відображаються в керуючі таблиці адресу пам'яті. У відповідності з отриманим описом СКБД знаходить в базі необхідні дані, перетворює, передає їх, наприклад, в прикладну програму, якої вони потрібні, або визначає місце в пам'яті комп'ютера, куди дані потрібно помістити і в якому вигляді, а також як з даними встановити зв'язок і які дані необхідно скорегувати.

Основні команди мови визначення даних показані в табл. 9.1.

Таблиця 9.1. Команди DDL

Назва команди	Опис
CREATE DATABASE	Створити базу даних
CREATE TABLE	Створити таблицю
CREATE VIEW	Створити віртуальну таблицю
CREATE INDEX	Створити індекс
CREATE TRIGGER	Створити тригер
CREATE PROCEDURE	Створити збережену процедуру
ALTER DATABASE	Модифікувати базу даних
ALTER TABLE	Модифікувати таблицю

ALTER VIEW	Модифікувати віртуальну таблицю
ALTER INDEX	Модифікувати індекс
ALTER TRIGGER	Модифікувати тригер
ALTER PROCEDURE	Модифікувати збережену процедуру
DROP DATABASE	Видалити базу даних
DROP TABLE	Видалити таблицю
DROP VIEW	Видалити віртуальну таблицю
DROP INDEX	Видалити індекс
DROP TRIGGER	Видалити тригер
DROP PROCEDURE	Видалити збережену процедуру

В залежності від алгоритму роботи конкретної СКБД можливі різні варіанти обробки вихідних текстів описів даних, складених на МВД. В одних випадках опис спочатку транслюється і потім, якщо немає синтаксичних помилок, виконується подальша обробка запиту, в якому присутній цей опис. В інших випадках, можливі попередня трансляція описів для налагодження і виявлення помилок. Потім налагоджені описи поміщаються в (спеціальну) бібліотеку описів, звідки СКБД їх вибирає за ідентифікатором при обробці відповідних запитів (ідентифікатор необхідного опису є в запиті). По-третє, у випадках може використовуватися режим інтерпретації описів при обробці запитів тощо.

Мова маніпулювання даними

Мова маніпулювання даними (ММД) (або мова запитів до БД) представляється зазвичай системою команд маніпулювання даними.

Приклад типових команд:

- провести вибірку з бази даного по його найменуванню;
- провести вибірку з бази всіх даних певного типу, значення яких задовольняють певними ознаками;
- знайти в базі позицію даного і помістити туди його нове значення.

ММД складається з чотирьох основних команд (табл. 9.2.).

Таблиця 9.2. Команди DML

Назва команди	**Опис**
SELECT	Вибрати
INSERT	Вставити
UPDATE	Оновити
DELETE	Видалити

Мова керування даними

Мова керування даними (МКД) - призначена для адміністратора системи і дозволяє визначати повноваження користувачів, забезпечує його засобами захисту, оптимізації і підтримки цілісності. Більш точно її можна назвати «мова керування доступом». Вона складається з двох основних команд:

Таблиця 9.3. Команди DCL

Назва команди	Опис
GRANT	Надати права
REVOKE	Забрати права

З точки зору прикладного інтерфейсу існують два різновиди команд SQL:

- інтерактивний SQL;
- вбудований SQL.

Інтерактивний SQL використовується в спеціальних утилітах (типу WISQL або DBD), що дозволяють в інтерактивному режимі вводити запити з використанням команд SQL, посилати їх для виконання на сервер і отримувати результати у призначеному для цього вікні. Вбудований SQL використовується в прикладних програмах, дозволяючи їм посилати запити до сервера і обробляти отримані результати, в тому числі комбінуючи set-орієнтований і record-орієнтований підходи.

Ми не будемо наводити точний синтаксис команд SQL, замість цього ми розглянемо їх на численних прикладах, що більш важливо для розуміння SQL, ніж точний синтаксис, який можна подивитися в документації на Вашу СКБД.

Отже, почнемо з розгляду команд мови маніпулювання даними.

Реляційні операції, як команди мови маніпулювання даними

Найбільш важливою командою мови маніпулювання даними є команда SELECT. За уявною простотою її синтаксису ховається величезне багатство можливостей. Нам важливо навчитися користовуватися цим багатством! Для того, щоб показати, що мова SQL є реляційно повною, потрібно показати, що будь-який реляційний оператор може бути виражений засобами SQL. Насправді достатньо показати, що засобами SQL можна виразити будь-який з примітивних реляційних операторів.

В якості інформаційної основи для прикладів ми будемо використовувати базу даних «Службовці підприємства» (схема Human Resources (HR)) (рис. 10.1.), що входить в поставку Oracle 11g EX. У записах про штат співробітників кожен співробітник має ідентифікаційний номер, адресу електронної пошти, ідентифікатор посади, оклад і (ідентифікатор) керівника. Деякі співробітники додатково до окладу заробляють комісійні. Також компанія зберігає інформацію про посади в рамках організації. У кожної посади є ідентифікатор, назва і діапазон (мінімальних та максимальних) окладів. Деякі співробітники довгий час працюють в компанії і займали в ній різні посади. При виході співробітника з посади записуються відомості про тривалість його роботи на даній посаді, її ідентифікатор і підрозділ, в якому працював співробітник. Компанія з даного прикладу працює в декількох регіонах, тому зберігаються відомості про місця розташування її складів і підрозділів. Співробітник може бути приписаним до одного з підрозділів, а

кожен підрозділ ідентифікується унікальним номером. Кожен підрозділ пов'язаний (знаходиться у взаємно однозначній відповідності) з одним місцем розташування, а для кожного місця розташування зберігається повна адреса, що містить назву вулиці, поштовий індекс, місто, штат або область та код країни. Для місцезнаходжень підрозділів і складів зберігається деталізована інформація: назва країни і географічний регіон, де розташована країна.

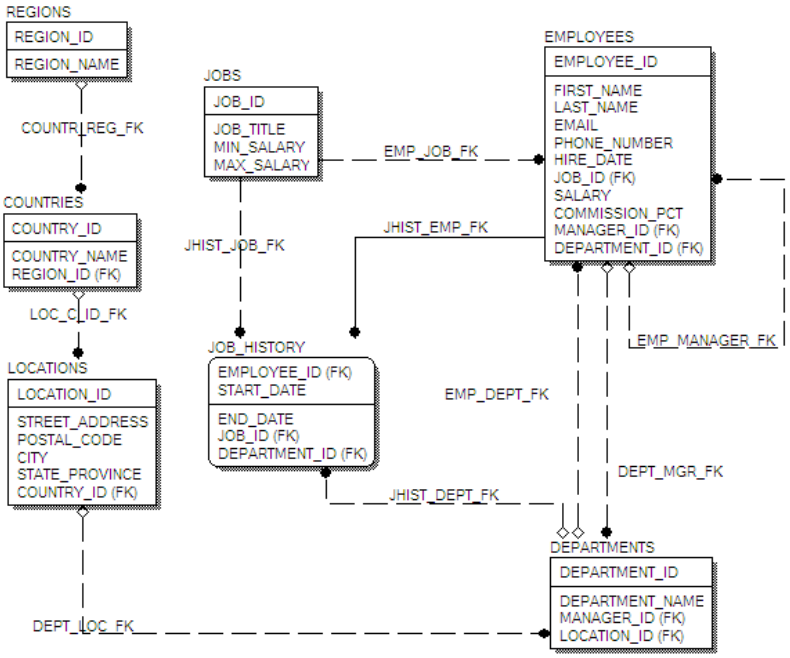


Рис. 9.1. Схема БД Human Resources

Наведена схема бази даних Human Resources для Oracle, зроблена за допомогою CASE-засобу ERwin Data Modeler. На схемі показані таблиці бази даних і взаємозв'язки, а також позначені первинні ключі та їх зв'язки з зовнішніми ключами. Багато з прикладів, особливо в наступній лекції, є досить складними. Однак, не слід на цій підставі робити висновок, що така складна сама мова SQL. Справа, скоріше за все, в тому, що звичайні (стандартні) операції настільки прості в SQL, що приклади таких операцій виявляються досить нецікавими і не ілюструють повної потужності цієї мови. Але в цілях системності ми пройдемо по всім можливостям SQL: від найпростіших - до надзвичайно складних.

Розглянемо операції реляційних баз даних:

- об'єднання (Union);
- перетин (Intersect);
- різниця (Minus);
- декартовий добуток (Times);
- вибірка (Restriction);
- проекція (Projection);
- з'єднання (Join);
- поділ таблиць (Divideby).

Операція вибірки (обмеження)

Реляційна алгебра: A WHERE c.

Операція вибірки (обмеження) дозволяє отримати всі рядки (записи) або частину рядків однієї таблиці.

Приклад 1. Отримати всі рядки таблиці Countries:
SELECT *
FROM countries

Таблиця 9.4. Всі рядки таблиці Countries

COUNTRY_ID	COUNTRY_NAME	REGION_ID
AR	Argentina	2
AU	Australia	3
BE	Belgium	1
BR	Brazil	2
CA	Canada	2
CH	Switzerland	1
CN	China	3
DE	Germany	1
DK	Denmark	1
EG	Egypt	4
FR	France	1
HK	HongKong	3
IL	Israel	4
IN	India	3
IT	Italy	1
JP	Japan	3
KW	Kuwait	4
MX	Mexico	2
NG	Nigeria	4
NL	Netherlands	1
SG	Singapore	3
UK	United Kingdom	1

US	United States of America	2
ZM	Zambia	4
ZW	Zimbabwe	4

У цьому прикладі та далі - для більшої наочності - всі зарезервовані слова мови SQL будемо писати великими літерами. Кожна конструкція для зручності буде писатися з нового рядку.

Приклад 2. Отримати підмножину рядків таблиці країн, що розташовані у регіоні 'Middle East and Africa' (region_id = 4):

```
SELECT *
FROM countries
WHERE region_id = 4
```

Таблиця 9.5. Підмножина рядків таблиці країн, що розташовані у регіоні 'Middle East and Africa'

COUNTRY_ID	COUNTRY_NAME	REGION_ID
EG	Egypt	4
IL	Israel	4
KW	Kuwait	4
NG	Nigeria	4
ZM	Zambia	4
ZW	Zimbabwe	4

Операція проекції

Реляційна алгебра: PROJECT A $\{a_1, a_2, \ldots, a_m\}$.

Операція проекції дозволяє виділити підмножину стовпців таблиці.

Приклад 3. Отримати список без повторів всіх країн для місць розташування:

```
SELECT country_id
FROM locations
GROUP BY country_id
```

або

```
SELECT DISTINCT country_id
FROM locations
```

Таблиця 10.6. Список без повторів всіх країн для місць розташування

COUNTRY_ID

US
SG
CA
CH
IT
MX
CN
DE
JP
IN
AU
UK
BR
NL

На практиці дуже часто потрібно отримати якусь підмножину стовпців і рядків таблиці, тобто виконати комбінацію Restriction і Projection. Для цього досить перерахувати стовпці таблиці та накласти обмеження на рядки.

Приклад 4. Знайти назву регіону для Китаю:
```
SELECT region_name
FROM countries, regions
WHERE countries.region_id = regions.region_id
AND country_name = 'China'
```

Таблиця 9.7. Назва регіону для Китаю

REGION_NAME
Asia

Приклад 5. Отримати прізвища та імена працівників, яких звуть 'Peter':
```
SELECT first_name, last_name
FROM employees
WHERE first_name = 'Peter'
```

Таблиця 10.8. Прізвища та імена працівників, яких звуть 'Peter'

FIRST_NAME	LAST_NAME
Peter	Vargas
Peter	Tucker
Peter	Hall

Ці приклади ілюструють загальну форму команди SELECT в мові SQL (для однієї таблиці):
SELECT (вибрати) специфіковані поля
FROM (з) специфікованої таблиці
WHERE (де) деяка специфікована умова є істинною.

Операція з'єднання

Реляційна алгебра: A JOIN B.

Операція з'єднання дозволяє з'єднувати рядки з більш ніж однієї таблиці (за деякою умовою) для утворення нових рядків даних.

Приклад 6. Отримати список керівників проектів:
SELECT first_name, last_name, department_name
FROM employees JOIN departments ON employees.employee_id = departments.manager_id

або

SELECT first_name, last_name, department_name
FROM employees, departments
WHERE employees.employee_id = departments.manager_id

Таблиця 9.9. Список керівників проектів

FIRST_NAME	LAST_NAME	PROJ_NAME
Jennifer	Whalen	Administration
Michael	Hartstein	Marketing
Den	Raphaely	Purchasing
Susan	Mavris	Human Resources
Adam	Fripp	Shipping
Alexander	Hunold	IT
Hermann	Baer	Public Relations
John	Russell	Sales
Steven	King	Executive
Nancy	Greenberg	Finance

Shelley	Higgins	Accounting

Приклад 7. Отримати список керівників проектів з зарплатнею більше за 10 000:
```
SELECT first_name, last_name, department_name
FROM employees JOIN departments ON employees.employee_id = departments.manager_id
WHERE employees.salary > 10000
```

Таблиця 9.10. Список керівників проектів з зарплатнею більше за 10 000

FIRST_NAME	LAST_NAME	PROJ_NAME
Michael	Hartstein	Marketing
Den	Raphaely	Purchasing
John	Russell	Sales
Steven	King	Executive
Nancy	Greenberg	Finance
Shelley	Higgins	Accounting

Операція об'єднання

Реляційна алгебра: A UNION B.

Операція об'єднання дозволяє об'єднувати результати окремих запитів по декількох таблицях в єдину результуючу таблицю. Таким чином, команда UNION об'єднує виведення двох або більше SQL-запитів в єдиний набір рядків і стовпців.

Приклад 8. Отримати список працівників, які працювали або працюють у відділі бухгалтерії (department_id = 110):
```
SELECT first_name, last_name, employees.department_id
FROM employees
WHERE department_id = 110
UNION ALL
SELECT first_name, last_name, job_history.department_id
FROM job_history, employees
WHERE employees.employee_id = job_history.employee_id
AND job_history.department_id = 110
```

Таблиці-операнди повинні бути сумісні, тобто мати однакову кількість стовпців і однакові типи (домени) стовпців в порядку їх перерахування. Не вимагається, щоб таблиці, які об'єднуються, мали б однакові імена колонок. Це відрізняє операцію об'єднання запитів в SQL від операції об'єднання в реляційній алгебрі. Найменування колонок в результуючій таблиці будуть автоматично взяті з першої таблиці в об'єднанні.

В СКБД Oracle для операції об'єднання використовуються команди UNION (з видаленням дублікатів) та UNION ALL (без видалення дублікатів).

Таблиця 9.11. Список працівників, які працювали або працюють у відділі бухгалтерії

FIRST_NAME	LAST_NAME	DEPARTMENT_ID
Shelley	Higgins	110
William	Gietz	110
Neena	Kochhar	110
Neena	Kochhar	110

Для довідки, наведемо загальну форму команди SELECT, що враховує можливість з'єднання декількох таблиць та об'єднання результатів:
SELECT [DISTINCT] список_вибираємих_елементів (полів)
FROM список_таблиць (або уявлень)
[WHERE предикат]
[GROUP BY поле (або поля)
[HAVING предикат]]
[UNION [ALL] інший_вираз_SELECT]
[ORDER BY поле (або поля) або номер (номери)]

Операція перетину

Реляційна алгебра: A INTERSECT B.

Операція перетину отримує таблицю, до складу якої входять лише такі рядки, які зустрічаються одночасно і в таблиці A, і в таблиці B.

Приклад 9. Отримати список посад та номери відділів, в яких працюють та працювали співробітники:
SELECT e.job_id, department_id
FROM employees e, jobs j
WHERE e.job_id = j.job_id
INTERSECT
SELECT job_id, department_id
FROM job_history

Таблиця 9.12. Список посад та номери відділів, в яких працюють та працювали співробітники

JOB_ID	DEPARTMENT_ID
AC_ACCOUNT	110
AC_MGR	110
IT_PROG	60
MK_REP	20

SA_MAN	80
SA_REP	80
ST_CLERK	50

Операція різниці

Реляційна алгебра: A MINUS B.

Операція різниці отримує таблицю, до складу якої входять лише ті рядки з таблиці A, яких нема в таблиці B.

Приклад 10. Отримати список країн, в яких не розташовані відділення:
```
SELECT country_id, country_name
FROM countries
MINUS
SELECT c.country_id, country_name
FROM countries c, locations l
WHERE c.country_id = l.country_id
```

Таблиця 9.13. Список країн, в яких не розташовані відділення

COUNTRY_ID	COUNTRY_NAME
AR	Argentina
BE	Belgium
DK	Denmark
EG	Egypt
FR	France
HK	HongKong
IL	Israel
KW	Kuwait
NG	Nigeria
ZM	Zambia
ZW	Zimbabwe

Операція поділу

Реляційна алгебра: A DIVIDEBY B.

Операція поділу таблиці A(a, b) на B(b) є таблиця C(a), що складається з рядків, які є частиною рядків з таблиці A, причому друга частина - це рядок з таблиці B. Кожний рядок з таблиці B входить як складова рядку в таблицю A.

Оператор SQL, що реалізує поділ відношень важко запам'ятати, тому дамо приклад еквівалентного перетворення виразів, що представляють суть запиту. Нехай таблиця A містить дані про історію посад співробітників, таблиця B містить список посад. Тоді розділити таблицю A на таблицю B означає в даному прикладі «Відібрати номери співробітників, які займали всі посади».

Приклад 11. Отримати список кодів співробітників, які займали всі посади:
```
SELECT DISTINCT jh1.employee_id
FROM job_history jh1
WHERE (SELECT COUNT(*) FROM jobs) =
(SELECT COUNT(DISTINCT(job_id))
FROM job_history jh2
WHERE jh1.employee_id = jh2.employee_id)
```

Таблиця 9.14. Список кодів співробітників, які займали всі посади

EMPLOYEE_ID

В нашому прикладі кількість посад - 19 і немає жодного співробітника, який займав таку велику кількість посад.

Операція декартового добутку

Реляційна алгебра: A TIMES B.

Операція декартового добутку отримує таблицю, що складається з рядків, які є конкатенацією (або злиттям) одного рядку першої таблиці та одного рядку другої таблиці.

Приклад 12. Отримати список можливих країн працевлаштування для співробітників:
```
SELECT e.first_name, e.last_name, c.country_name
FROM countries c, employees e
```

Таблиця 9.15. Список можливих країн працевлаштування для співробітників

FIRST_NAME	LAST_NAME	COUNTRY_NAME
Ellen	Abel	Argentina
Sundar	Ande	Argentina
Mozhe	Atkinson	Argentina
David	Austin	Argentina
Hermann	Baer	Argentina

Shelli	Baida	Argentina
Amit	Banda	Argentina
Elizabeth	Bates	Argentina
...		

При виконанні декартового добутку отримаємо 107*25 = 2675 рядків.

Оператор RENAME

Реляційний оператор перейменування RENAME виражається за допомогою ключового слова AS в списку полів, що вибираються, оператора SELECT.

Таким чином, мова SQL є реляційно повною.

10.4. Віртуальні атрибути і таблиці

Віртуальні дані - дані, які існують "only logic" (вони в БД не зберігаються, але постійно обчислюються). Наприклад, вік - це різниця між поточною датою та датою народження. При цьому саме поле «Вік» у БД відсутнє. Віртуальні дані є для користувачів, програмістів та адміністраторів.

Віртуальні дані є двох видів:

- віртуальні атрибути;
- віртуальні таблиці.

Віртуальний атрибут в БД не зберігається, це тимчасова змінна, яка обчислюється лише тоді, поки з нею працюють. *Віртуальна таблиця* підтримується СКБД. Вона створюється при старті додатку та існує весь час роботи прикладної програми, а потім після завершення роботи додатку ця таблиця знищується. В цих таблицях складніше порушити цілістність, тому що вони обчислюються наново при кожному старті додатку.

10.5. Приклади використання операторів Insert, Update та Delete

INSERT - вставка рядків у таблицю

Приклад 13. Вставка в таблицю регіонів regions (region_id, region_name) одного рядка (5, 'East Europe'):

```
INSERT INTO regions (region_id, region_name)
VALUES (5, 'East Europe')
```

Приклад 14. Вставка в таблицю декількох рядків, вибраних з іншої таблиці (в таблицю tmp_table вставляються дані про регіони з таблиці regions, що мають номери, більше за 3):

```
INSERT INTO tm_table (region_id, region_name)
SELECT region_id, region_name
FROM regions
WHERE region_id > 3
```

UPDATE - оновлення рядків у таблиці

Приклад 15. Оновлення декількох рядків у таблиці країн:
```
UPDATE countries
SET region_id = 5
WHERE region_id = 4
```

DELETE - видалення рядків в таблиці
Приклад 16. Видалення декількох рядків у таблиці країн:
```
DELETE FROM countries
WHERE region_id = 3
```

Приклад 17. Видалення всіх рядків у таблиці країн:
```
DELETE FROM countries
```

Тригер — це SQL-оператор, що активізується під час виконання певних операцій над об'єктами бази даних. Об'єктами бази даних є таблиці (уявлення або вся БД), а операціями - додавання (INSERT), заміна (UPDATE) та видалення (DELETE) рядків. Тригери - це один із механізмів підтримки цілісності бази даних.

У найпростішому випадку синтаксис оголошення тригера є таким:
CREATE TRIGGER <назва тригера>
{BEFORE | AFTER} <операції над таблицею> [OF <список полів>] ON <назва таблиці>
[WHEN (<умова>)] <оператори SQL>
Якщо умова у фразі WHEN є істинною або ця фраза відсутня, до (BEFORE) або після (AFTER) виконання операції INSERT, UPDATE чи DELETE над таблицею, зазначеною після слова ON, буде виконано вказані нижче оператори SQL. Коли таких операторів кілька, їх слід помістити між ключовими словами BEGIN та END. Конструкція другого рядка визначення тригера називається *реченням ініціювання*, між *умовою ініціювання*, а <оператори SQL> — *дією тригера*.

Будь-яка СКБД розширює стандарт мови SQL, причому в багатьох випадках SQL трансформується в мову програмування, що надає усі можливості SQL. Наприклад, у Microsoft SQL Server таким розширенням є мова Transact-SQL, в Oracle - мова PL/SQL. Кожна з них має переваги звичайних мов програмування, підтримує всі можливості стандарту ANSI SQL, а також розширює його.

Приклади процедурних мов:

- Transact-SQL (MS SQL Server, Sybase);
- PL\SQL (Oracle);
- PSQL (FireBird);
- SQL PL (IBM DB2);
- JetSQL (MS Access);
- PL/pgSQL (PostgreSQL).

Розглянемо приклади застосування тригерів.

Приклад 1. Перед видаленням інформації про відділ видалити інформацію про всіх співробітників цього відділу:
```
CREATE OR REPLACE TRIGGER Departments_Delete_Trigger
BEFORE DELETE ON departments FOR EACH ROW
BEGIN
```

```
DELETE FROM employees
WHERE employees.department_id = :old.department_id;
END;
```

Приклад 2. Після видалення рядка з відомостями про співробітника встановити значення NULL в полі manager_id тих записів із таблиці відділів, в яких заданий співробітник був керівником:

```
CREATE OR REPLACE TRIGGER Employees_Delete_Trigger
BEFORE DELETE ON employees FOR EACH ROW
BEGIN
UPDATE departments
SET manager_id = null
WHERE departments.manager_id = :old.employee_id;
END;
```

Види тригерів

1) за типом події:

- INSERT;
- UPDATE;
- DELETE.

2) час спрацьовування:

- BEFORE;
- AFTER.

3) за об'єктом БД:

- для таблиці на рівні таблиці;
- для таблиці на рівні рядка;
- для уявлення View;
- для всієї БД.

4) за транзакціями:

- в тій же транзакції;
- автономна транзакція.

Приклад 3. Тригер для таблиці на рівні таблиці:

```
CREATE OR REPLACE TRIGGER Countries_Update_Trigger
AFTER UPDATE ON countries
BEGIN
INSERT INTO info
VALUES ('таблиця країн була змінена');
END;
```

Приклад 4. Тригер для таблиці на рівні рядка:

```
CREATE OR REPLACE TRIGGER Countries_Update_Trigger
AFTER UPDATE ON countries FOR EACH ROW
BEGIN
INSERT INTO info
```

```
VALUES ('один рядок з таблиці країн була змінена');
END;
```

Доступ до старих і нових значень рядків

Якщо виконується оновлення рядків таблиці, то в тригері допускається звернення до старих і нових значень рядків, що оновлюються. Для звернення до нового (старого) рядка слід вказати кваліфікатор NEW (OLD) перед назвою стовпця (табл. 13.1.). Такі кваліфікатори дозволяється використовувати як в умові тригера, так і в описі його дії.

Таблиця 10.1. Застосування кваліфікаторів NEW та OLD при різних операціях

Операція	**OLD**	**NEW**
INSERT	-	NEW.Назва_стовпця
UPDATE	OLD.Назва_стовпця	NEW.Назва_стовпця
DELETE	OLD.Назва_стовпця	-

Приклад 5. Заборонити підвищення зарплатні співробітника на більше ніж 10%:

```
CREATE OR REPLACE TRIGGER Employees_Salary_Trigger
AFTER UPDATE OF salary ON employees FOR EACH ROW
BEGIN
IF (:NEW.Salary > 1.1 * :OLD.Salary) THEN
RAISE_APPLICATION_ERROR(-20001,
'Завеликі зміни зарплатні');
END IF;
END;
```

Тригери й транзакції

Дії, які виконуються під час основної операції та в тригері, становлять єдину транзакцію.

- перед виконанням додавання, оновлення та видалення неявно ініціюється команда BEGIN TRANSACTION;
- реалізується операція додавання/оновлення/видалення;
- ініціюється та виконується тригер;
- тригер скасовує транзакцію або за замовчуванням його дія завершується.

Приклад 6. Дозволити додавання рядка з відомостями про кафедру лише в тому випадку, коли існує рядок із даними про факультет, якому належить кафедра (кафедри без факультету не буває):

```
CREATE OR REPLACE TRIGGER Employees_Audit_Trigger
AFTER UPDATE ON employees FOR EACH ROW
DECLARE
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
INSERT INTO info
VALUES ('table "employees" has changed');
COMMIT;
END;
```

Деякі СКБД накладають обмеження на оператори, які можуть бути використані в тригері: наприклад, може бути заборонено вносити зміни в таблицю, для якої створений тригер (називається «мутуючий» тригер для «мутуючої» таблиці), не можна виконувати оператори START TRANSACTION, COMMIT або ROLLBACK.

Вкладеність тригерів

Тригери бувають вкладеними. Це означає, що маніпулювання рядком однієї таблиці може ініціювати тригер, який маніпулює рядками іншої таблиці. У свою чергу, маніпулювання рядками другої таблиці може ініціювати тригер, що маніпулює рядками третьої таблиці тощо (рис. 13.1.).

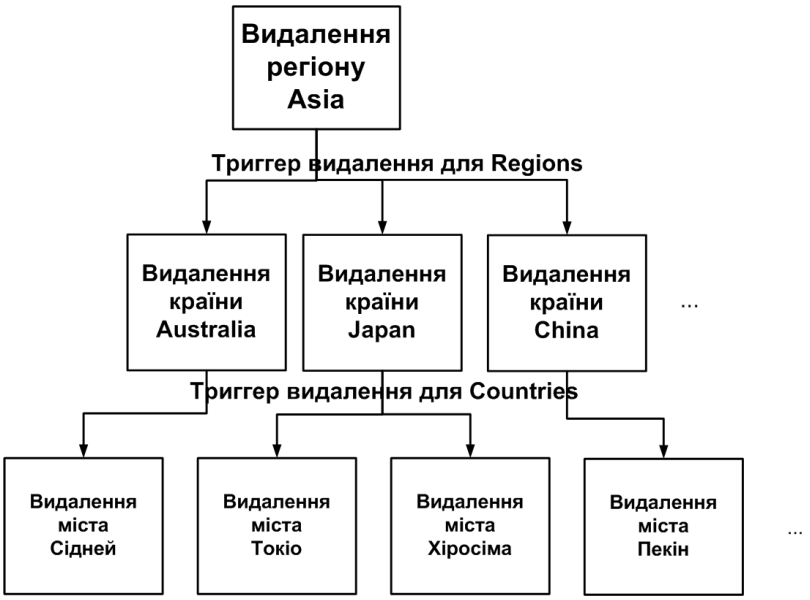


Рис. 10.1. Каскадне видалення за допомогою тригерів

Вкладеність тригерів може призвести до «зациклення».

Обмеження на використання тригерів:

- після видалення таблиці (або уявлення) всі зв'язані з нею тригери також видаляються;
- тригери визначаються лише для вже створених таблиць (або уявленнь).

В усіх наведених прикладах тригери використовувалися для підтримки цілісності бази даних. Але існує багато інших можливостей використання тригерів. Наведемо лише деякі з них:

- автоматичне обчислення значень похідних стовпців;
- запобігання виконанню недозволених транзакцій;
- виконання складних перевірок з метою захисту даних;
- підтримка складних обмежень цілісності;
- реєстрація подій, що відбуваються в базі даних;
- збирання статистики щодо доступу до таблиць бази даних.

Тригер для View

Тригери можуть бути прив'язані не до таблиці, а до представлення (VIEW). У цьому випадку з їх допомогою реалізується механізм «оновлюваного уявлення». Тоді ключові слова BEFORE і

AFTER впливають лише на послідовність виклику тригерів, так як власне подія (видалення, вставка або оновлення) не відбувається.

Збереженні процедури

Збережені процедури - це об'єкти бази даних, що розташовані на стороні сервера, зберігаються у відкомпільованому вигляді, які зазвичай містять велику кількість директив процедурної мови сервера СКБД. Такі процедури можуть містити сукупність команд (складних запитів, операцій оновлення, додавання та видалення), що часто використовуються як єдине ціле. Збережена процедура дає змогу звернутися до неї як до функції, замість того, щоб послідовно записувати команди SQL, які вона містить. Значне зниження навантаження на канали зв'язку під час роботи користувачів із сервером теж є важливою перевагою збережених процедур, адже замість окремих багаторазових звернень до бази даних виконується єдине звернення до збереженої процедури.

Збереженні процедури пишуться тією ж мовою, що і тригери. Але збережені процедури автоматично не викликаються сервером, тому їх необхідно викликати прикладному програмісту вручну.

Використання курсорів

Курсор є покажчиком на окремий запис тієї чи іншої таблиці. Курсори використовуються переважно в збережених процедурах, а також у програмах, що працюють з базою даних й ініціюють виконання операторів SQL. Використання курсорів дає можливість:

- вибрати певну сукупність даних (на зразок тимчасової таблиці, що містить результати виконання оператора SELECT);
- ініціювати послідовний перегляд сукупності записів;
- проаналізувати конкретний запис, на який вказує курсор;
- виконати зовнішню операцію над поточним записом перш ніж перейти до наступного.

Ще одним варіантом застосування курсору є тимчасове зберігання результатів запиту для подальшого використання. Якщо зовнішня програма вимагає багаторазового використання певного набору записів, то створюється курсор, яким оперують замість багаторазового виконання оператора SELECT.
Для використання курсору потрібно:

1. Створити курсор.
2. Відкрити курсор для використання в додатку чи збереженій процедурі.
3. Ініціювати за допомогою курсору послідовне перебирання записів з метою їхньої обробки.
4. Закрити курсор після завершення роботи з ним (з можливим наступним відкриттям у разі потреби).
5. Видалити курсор.

На відміну від таблиць, індексів, тригерів та збережених процедур, курсори не є об'єктами бази даних.

Наприклад, в PL/SQL підтримуються два типи курсорів: явні та неявні. Явний курсор оголошується розробником, а неявний курсор не вимагає оголошення.

Курсор може повертати один рядок, кілька рядків або жодного рядка. Для запитів, які повертають більше одного рядка, можна використовувати тільки явний курсор. Для

повторного створення результуючого набору для інших значень параметрів курсор слід закрити, а потім повторно відкрити.

Курсор може бути оголошений в секціях оголошень будь-якого блоку PL/SQL, підпрограми або пакета.
Оператори керування явним курсором:

- DECLARE виконує оголошення явного курсора;
- OPEN відкриває курсор, створюючи новий результуючий набір на базі зазначеного запиту;
- FETCH виконує послідовне вилучення рядків з результуючого набору від початку до кінця;
- CLOSE закриває курсор і звільняє займані ним ресурси.

Атрибути курсора:

- % ISOPEN - повертає значення TRUE, якщо курсор відкритий;
- % FOUND - визначає, чи знайдений рядок, який задовольняє умові;
- % NOTFOUND - повертає TRUE, якщо рядок не знайдений;
- % ROWCOUNT - повертає номер поточного рядка.

Динамічний SQL

Багато СКБД надають засоби для застосування SQL у традиційних мовах програмування. Існують два методи такого використання мови.

Статичний SQL припускає вкладання безпосередньо в програмний код речення, що не може бути змінене під час виконання програми. Зазвичай статичний SQL вимагає використання предкомпілятора. Наприклад, Oracle має предкомпілятори для використання SQL у мовах C, Pascal, Ada, COBOL та FORTRAN.

Динамічний SQL дає змогу програмувати побудову SQL-запитів, що створюються під час виконання програми і передаються СКБД, яка, в свою чергу, передає знайдені дані змінним програми.

Тимчасові таблиці

Деякі СКБД надають можливість створювати тимчасові таблиці, які є звичайними таблицями, що існують лише до завершення сеансу роботи користувача з базою даних. Як правило, такі таблиці входять до складу спеціальної службової бази даних, призначеної для їхнього зберігання. Синтаксис директив для роботи з тимчасовими таблицями аналогічний синтаксису відповідних команд для звичайних таблиць, за винятком того, що є конструкції, призначені для створення тимчасових таблиць. Тимчасові таблиці потрібні для зберігання тимчасових даних, які потрібні лише протягом поточного сеансу зв'язку із СКБД.

Зв'язані змінні

Етапи виконання запиту SQL-машиною без кешу SQL-запитів:

1. Синтаксичний аналіз - визначається, чи правильно взагалі з точки зору синтаксису SQL-мови сформульований запит.
2. Перевірка прав доступу користувача до об'єктів БД – таблиці, уявлення, процедури, функції тощо.

3. Генерація планів виконання запиту і вибір найкращого плану.
4. Виконання плану запиту.

Етапи виконання запиту SQL-машиною з кешем SQL-запитів:

1. Пошук в кеші SQL-запитів: а) якщо не знайдено – тоді «жорсткий (довгий) розбір». б) якщо знайдено – тоді «м'який (швидкий) розбір».
2. Генерація планів виконання запиту і вибір найкращого плану.
3. Виконання плану запиту.

Особливості застосування зв'язаних змінних:

- підвищення швидкості виконання, за рахунок використання кеша SQL-запитів;
- підвищення безпеки, тому що змінні передаються окремо в спеціальному вигляді (SQL-ін'єкція).

Приклад 7. Вірне використання зв'язаних змінних:

```
SELECT *
FROM employees e
WHERE e.employee_id = :employee_id
```

Приклад 8. Невірне використання зв'язаних змінних:

```
SELECT *
FROM employees e
WHERE e.employee_id = 101

    SELECT *
FROM employees e
WHERE e.employee_id = 117
```

Література

1. Діденко Д. Г**.** Бази даних та інформаційні системи. Електронний ресурс. Режим доступу: http://www.simulation.kiev.ua/dbis/index.html

2. Методологія проектування та інструментарій для створення мобільних додатків. Вісник НТУ “ХПІ». 2013. No56(1029)

3. ГрицуновО. В. Інформаційні системи та технології: навч. посіб. Для студентів за напрямом підготовки «Транспортні технології» / О. В. Грицунов; Харк. нац. акад. міськ. госп-ва. – Х.: ХНАМГ, 2010. – 222 с.

4.Голощапов А. Л. Google Android: программирование для мобильных устройств. — СПб.: БХВ-Петербург, 2011. — 448 с: ил.

5.Zigurd Mednieks, Laird Dornin, G. Blake Meike, Masumi Nakamura. Programming Android. – O'Reilly Media, Inc., 2011. – 524 p.

6. С. Хашими, С. Коматинени, Д. Маклин. Разработка приложений для Android. – СПб.: Питер, 2011.– 736 с.

7. Reto Meier. Professional Android 2. Application Development. – Wiley Publishing, Inc, 2010. – 580 p.