Основні відомості про CASE-засоби IBM Rational

Деякі проектні команди розглядають CASE-засоби як "милиці" для новачків, інші вважають їх не менш важливими, ніж текстові процесори.
Е. Йордон „Шлях камікадзе”

1. Вступ до Rational Rose

Rational Rose – сімейство об'єктно-орієнтованих CASE-засобів фірми Rational Software Corporation – призначене для автоматизації процесів аналізу та проектування ПЗ, а також для генерації кодів різними мовами та випуску проектної документації. Rational Rose використовує метод об'єктно-орієнтованого аналізу та проектування, що базується на мові UML.

Поточна версія Rational Rose реалізує генерацію кодів програм для C++, Visual C++, Visual Basic, Java, PowerBuilder, CORBA Definition Language (IDL), генерацію описів баз даних для ANSI SQL, Oracle, MS SQL Server, IBM DB2, Sybase, а також дозволяє розробляти проектну документацію у вигляді діаграм та специфікацій. Крім того, Rational Rose містить засоби реверсного інжинірингу програм та баз даних, які забезпечують повторне використання програмних компонентів у нових проектах.

Структура та функції.

В основі роботи Rational Rose лежить побудова діаграм та специфікацій UML, що визначають архітектуру системи, її статичні та динамічні аспекти. У складі Rational Rose можна виділити шість основних структурних компонентів:

- **репозиторій,**
- **графічний інтерфейс користувача,**
- **засоби перегляду проекту (браузер),**
- **засоби контролю проекту,**
- **засоби збору статистики та**
- **генератор документів.**

До них додаються генератор кодів (індивідуальний для кожної мови) та аналізатор для C++, що забезпечує реверсний інжиніринг.

Репозиторій є базою даних проекту. Браузер забезпечує "навігацію" за проектом, у тому числі переміщення по ієрархіях класів і підсистем, перемикання від одного виду діаграм до іншого і т.д. його описи. Генератор звітів формує тексти вихідних документів на основі інформації, що міститься в репозиторії.

Засоби автоматичної генерації кодів програм мовою C++, використовуючи інформацію, що міститься у діаграмах класів та компонентів, формують файли заголовків та файли описів класів та об'єктів. Скелет програми, що створюється таким чином, може бути уточнений шляхом прямого програмування мовою C++. Аналізатор кодів C++ реалізований як окремого програмного модуля. Його призначення - створювати модулі проектів Rational Rose на основі інформації, що міститься у вихідних текстах, що визначаються користувачем, на C++. У процесі роботи аналізатор здійснює контроль правильності вихідних текстів та діагностику помилок. Модель, отримана внаслідок його роботи, може цілком або фрагментарно використовуватись у різних проектах. Аналізатор має широкі можливості налаштування по входу та виходу. Наприклад, можна визначити типи вихідних файлів, базовий компілятор, задати, яка інформація повинна бути включена в модель, що формується, і які елементи вихідної моделі слід виводити на екран. Таким чином Rational Rose забезпечує можливість повторного використання програмних компонентів.

В результаті розробки проекту за допомогою CASE-засобу Rational Rose формуються такі документи:

- діаграми UML, у сукупності являють собою модель програмної системи, що розробляється;
- специфікації класів, об'єктів, атрибутів та операцій;
- заготівля текстів програм.

Тексти програм є заготівлями для подальшої роботи програмістів. Склад інформації, що включається до програмних файлів, визначається або за умовчанням, або на розсуд користувача. Надалі ці вихідні тексти розвиваються програмістами у повноцінні програми.

Взаємодія з іншими засобами та організація групової роботи.

Для підтримки командної роботи над проектом на кожній стадії життєвого циклу є інтегрований набір продуктів Rational Suite. Rational Suite існує у таких варіантах:

- Rational Suite AnalystStudio - призначений для визначення та управління повним набором вимог до системи, що розробляється;
- Rational Suite DevelopmentStudio - призначений для проектування та реалізації ПЗ;

- Rational Suite TestStudio - є набір продуктів, призначених для автоматичного тестування додатків;
- Rational Suite Enterprise – забезпечує підтримку повного життєвого циклу ПЗ та призначений як для менеджерів проекту, так і окремих розробників, що виконують кілька функціональних ролей у команді розробників.

До складу Rational Suite, крім Rational Rose, входять такі компоненти:

- Rational Requisite Pro – засіб управління вимогами, призначений для організації спільної роботи групи розробників. Воно дозволяє команді розробників створювати, структурувати, встановлювати пріоритети, відстежувати, контролювати зміни вимог, які виникають будь-якому етапі розробки компонентів докладання;
- Rational ClearCase – засіб управління конфігурацією ПЗ;
- Rational SoDA – засіб автоматичної генерації проектної документації;
- Rational ClearQuest - засіб для керування змінами та відстеження дефектів у проекті на основі засобів e-mail та Web;
- Rational TeamTest – засіб автоматичного виявлення помилок під час виконання програми та генерації сценаріїв для проведення регресійного тестування;
- Rational Robot - засіб для створення, модифікації та автоматичного запуску тестів;
- Rational Purify - засіб для локалізації трудновиявлених помилок часу виконання програми;
- Rational PureCoverage – засіб ідентифікації ділянок коду, пропущених при тестуванні;
- Rational Quantify – засіб кількісного визначення вузьких місць, що впливають на загальну ефективність роботи програми;
- Rational Suite Performance Studio – засіб навантажувального тестування додатків «клієнт-сервер» та Web-додатків.

Для організації групової роботи у Rational Rose можливе розбиття моделі на керовані підмоделі. Кожна з них зберігається на диску або завантажується в модель. Як підмодель може бути пакет або підсистема.

Середовище функціонування. Rational Rose працює на різних платформах: IBM PC (Windows), Sun SPARCstations (UNIX, Solaris, SunOS), Hewlett-Packard (HP UX), IBM RS/6000 (AIX).

2. Робота в середовищі Rational Rose

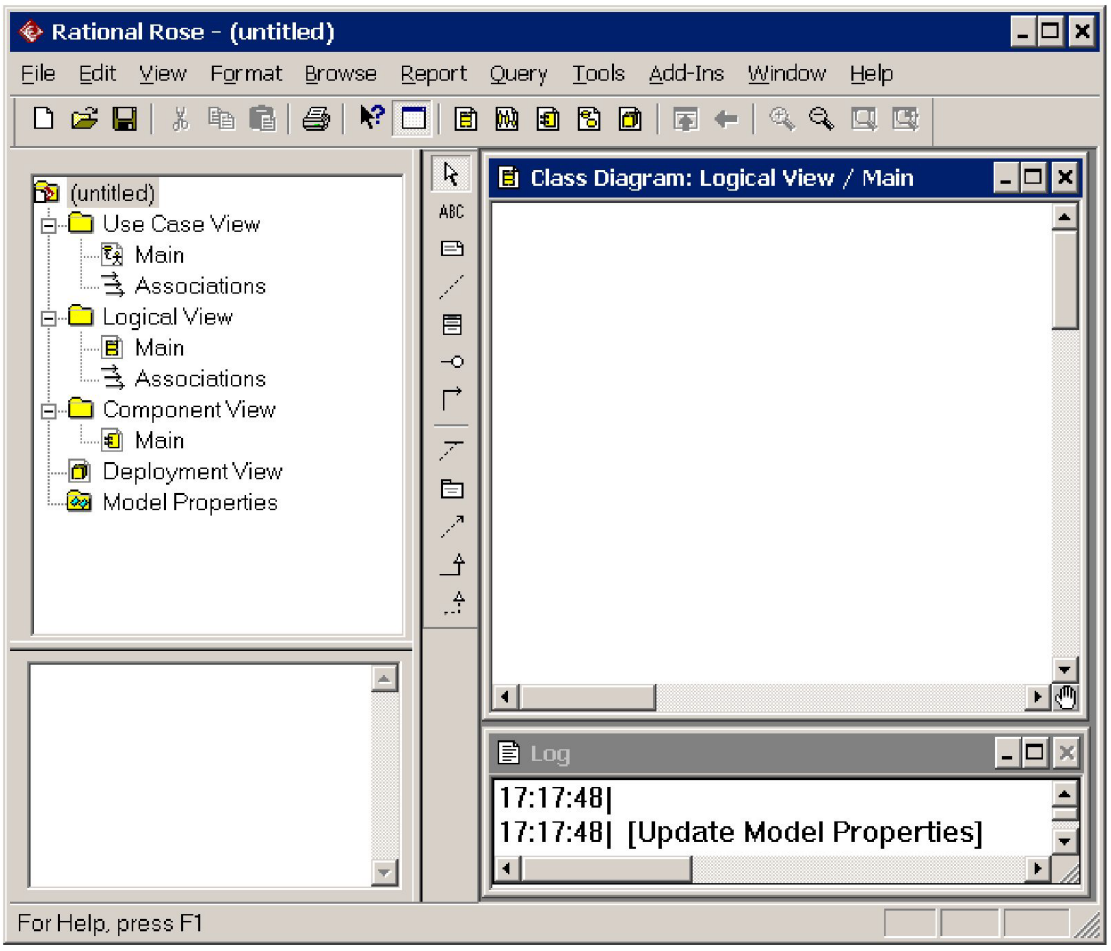
2.1. Елементи екрану

П'ять основних елементів інтерфейсу Rose – це браузер, вікно документації, панелі інструментів, вікно діаграми та журнал (log). Їхнє призначення полягає в наступному:

- браузер (browser) – використовується для швидкої навігації за моделлю;
- вікно документації (documentation window) - застосовується до роботи з текстовим описом елементів моделі;
- панелі інструментів (toolbars) – застосовуються для швидкого доступу до найпоширеніших команд;
- вікно діаграми (diagram window) - використовується для перегляду та редагування однієї або кількох діаграм UML;

- журнал (log) - застосовується перегляду помилок і звітів про результати виконання різних команд.

На рис. 2.1 показано різні частини інтерфейсу Rose.



Мал. 2.1. Інтерфейс Rational Rose.

Браузер

Браузер – це ієрархічна структура, що дозволяє здійснювати навігацію за моделлю. Все, що додається в модель – дійові особи, варіанти використання, класи, компоненти – буде показано у вікні браузера. За допомогою браузера можна:

- додавати до моделі елементи (дійові особи, варіанти використання, класи, компоненти, діаграми тощо);
- переглядати наявні елементи моделі;
- переглядати зв'язки між елементами моделі;
- переміщувати елементи моделі;
- перейменовувати ці елементи;
- додавати елементи моделі до діаграми;
- пов'язувати елемент із файлом або адресою Інтернет;
- групувати елементи пакети;
- працювати з деталізованою специфікацією елемента;
- відкривати діаграму.

Браузер підтримує чотири уявлення (view): подання варіантів використання, компонентів, розміщення та логічне подання. Всі вони і елементи моделі, що містяться в них, описані нижче в підрозділ. 2.2.

Браузер організований у деревоподібному стилі. Кожен елемент моделі може містити інші елементи, що знаходяться нижче за нього в ієрархії. Знак « - » біля елемента означає, що його гілка повністю розкрита. Знак «+» - що його гілка згорнута.

Вікно документації

З його допомогою можна документувати елементи моделі Rose. Наприклад, можна зробити короткий опис кожної дійової особи. При документуванні класу все, що буде написано у вікні документації, потім з'явиться у вигляді коментаря в згенерованому коді, що позбавляє необхідності згодом вносити ці коментарі вручну. Документація виводитиметься також у звітах, створюваних у середовищі Rose.

Панелі інструментів

Панелі інструментів Rose забезпечують швидкий доступ до найпоширеніших команд. У цьому середовищі є два типи панелей інструментів: стандартна панель та панель діаграми. Стандартна панель завжди видно, її кнопки відповідають командам, які можуть використовуватися для роботи з будь-якою діаграмою. Панель діаграми своя для кожного типу діаграм UML.

Усі панелі інструментів можуть бути змінені та налаштовані користувачем. Для цього виберіть пункт Tools > Options, а потім виберіть вкладку Toolbars.

Щоб показати або приховати стандартну панель інструментів (або панель інструментів діаграми):

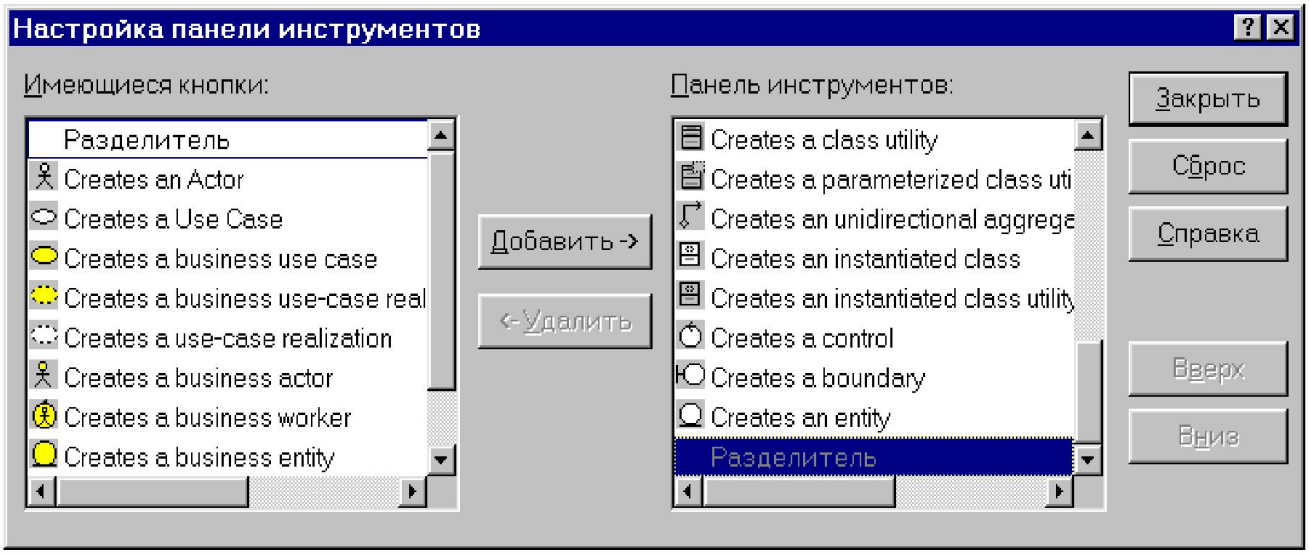
1. Виберіть Tools > Options.
2. Виберіть вкладку Toolbars.
3. Щоб зробити видимою або невидимою стандартну панель інструментів, позначте (або зніміть позначку) контрольний перемикач Show Standard ToolBar (або Show Diagram ToolBar)

Щоб збільшити розмір кнопок на панелі інструментів:

1. Клацніть правою кнопкою миші на потрібній панелі.
2. Виберіть у спливаючому меню пункт Use Large Buttons (Використовувати великі кнопки)

Щоб налаштувати панель інструментів:

- Клацніть правою кнопкою миші на потрібній панелі.
- Виберіть Customize (налаштувати)
- Щоб додати або видалити кнопки, виберіть відповідну кнопку, а потім клацніть мишею на кнопці Add (додати) або Remove (видалити), як показано на рис. 2.2.



Мал. 2.2.Налаштування стандартної панелі інструментів.

Вікно діаграми

У діаграмі видно, як виглядає одна або кілька діаграм UML моделі. При внесенні в елементи діаграми змін Rose автоматично оновить браузер. Аналогічно, при внесенні змін до елемента за допомогою браузера Rose автоматично оновить відповідні діаграми. Це допомагає підтримувати модель у несуперечливому стані.

Журнал

У міру роботи над вашою моделлю певна інформація надсилатиметься у вікно журналу. Наприклад, туди розміщуються повідомлення про помилки, що виникають під час генерації коду. Немає способу закрити журнал зовсім, та його вікно може бути мінімізовано.

2.2. Чотири вистави моделі Rose

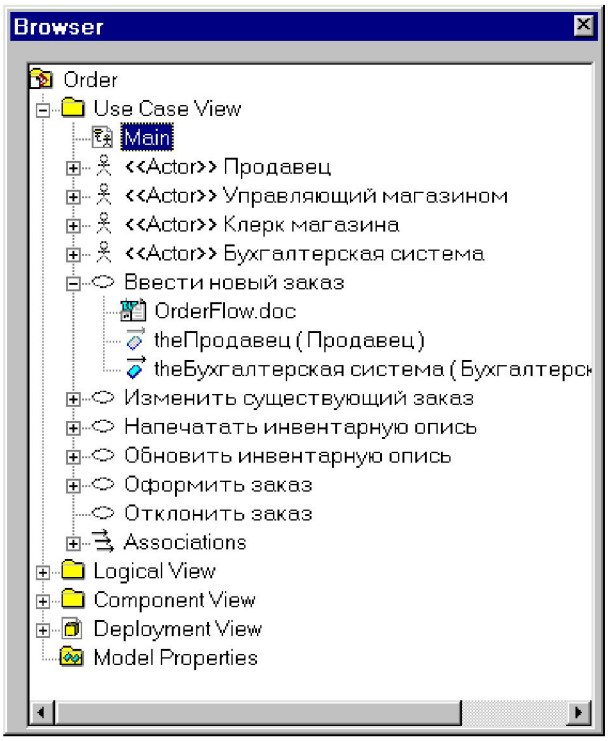
У моделі Rose підтримується чотири уявлення (views) - подання варіантів використання, логічне подання, подання компонентів та подання розміщення. Кожна з них призначена для своїх цілей та для відповідної аудиторії. У наступних розділах цього розділу ми коротко розглянемо кожне із зазначених уявлень, а в решті книги детально обговоримо елементи моделі, що містяться в них.

Подання варіантів використання

Це представлення містить всіх дійових осіб, всі варіанти використання та їх діаграми для конкретної системи. Воно може також містити деякі діаграми послідовності та кооперативні діаграми. На рис. 2.3 показано, як виглядає уявлення варіантів використання у браузері Rose.

Подання варіантів використання містить:

- Чинних осіб.
- Використання.
- Документацію за варіантами використання, що деталізує процеси (потоки подій), що відбуваються в них, включаючи обробку помилок. Піктограми зображують зовнішні файли, прикріплені до моделі Rose. Вигляд піктограми залежить від програми, яка використовується для документування потоку подій. У разі (рис. 2.3) застосовувався Microsoft Word.
- Діаграми варіантів використання. Зазвичай система має кілька таких діаграм, кожна з яких показує підмножину дійових осіб та/або варіантів використання.
- - Пакети, які є групами варіантів використання та/або дійових осіб.



Мал. 2.3. Подання варіантів використання.

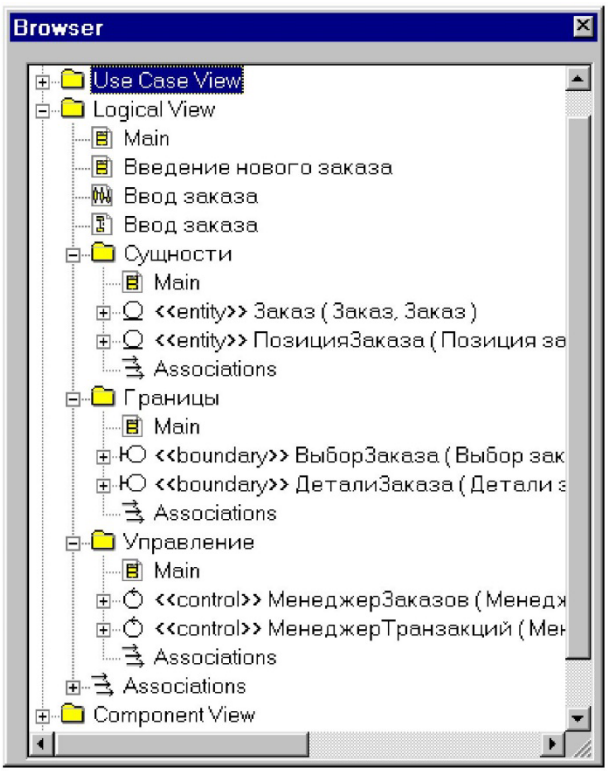
Логічне уявлення

Логічне уявлення, показане на рис. 2.4, концентрується на тому, як система реалізовуватиме поведінку, описану у варіантах використання. Воно дає докладну картину

складових частин системи та визначає взаємодію цих елементів. Логічне уявлення включає, окрім іншого, конкретні необхідні класи, діаграми класів та діаграми станів. З їхньою допомогою конструюється детальний проект створюваної системи.

Логічне уявлення містить:

- Класи.
- Діаграми класів. Як правило, для опису системи використовується кілька діаграм класів, кожна з яких відображає деяке підмножина всіх класів системи.
- Діаграми взаємодії, які застосовуються для відображення об'єктів, що беруть участь в одному потоці подій варіанти використання.
- Діаграма станів.
- - пакети, які є групами взаємопов'язаних класів.



Мал. 2.4. Логічне уявлення системи.

Подання компонентів

Подання компонентів містить:

- Компоненти є фізичними модулями коду.
- Діаграма компонентів.
- Пакети є групами зв'язаних компонентів.

Подання розміщення

Останнє уявлення Rose - це подання розміщення. Воно відповідає фізичному розміщенню системи, що може відрізнятиметься від її логічної архітектури. До представлення розміщення входять:

- Процеси, що є потоками (threads), що виконуються у відведеній для них області пам'яті.
- Процесори, що включають будь-які комп'ютери, здатні обробляти дані. Будь-який процес виконується одному чи кількох процесорах.
- Пристрої, тобто будь-яка апаратура, не здатна обробляти дані. До таких пристроїв відносяться, наприклад, термінали введення-виводу та принтери.
- Діаграма розміщення.

2.3. Параметри налаштування відображення

Зображення атрибутів та операцій на діаграмах класів

Rose має можливість налаштувати діаграми класів так, щоб:

- показувати всі атрибути та операції;
- приховати операції;
- приховати атрибути;
- показувати лише деякі атрибути чи операції;
- показувати операції разом з їх повними сигнатурами або лише їхні імена;
- показувати чи не показувати видимість атрибутів та операцій;
- показувати чи не показувати стереотипи атрибутів та операцій.

Значення кожного параметра за замовчуванням можна встановити за допомогою вікна, що відкривається при виборі пункту меню Tools > Options.

У даного класу на діаграмі можна:

- показати всі атрибути;
- приховати всі атрибути;
- показати лише вибрані вами атрибути;
- придушити висновок атрибутів.

Пригнічення виведення атрибутів призведе не тільки до зникнення атрибутів з діаграми, але й до видалення лінії, що показує місце розташування атрибутів у класі.

Існує два способи зміни параметрів подання атрибутів на діаграмі. Можна встановити потрібні значення кожного класу індивідуально. Можна також змінити значення параметрів за

замовчуванням до початку створення діаграми класів. Внесені таким чином зміни вплинуть тільки на новостворені діаграми.

Щоб показати всі атрибути класу:

1. Виділіть на діаграмі потрібний клас.
2. Клацніть правою кнопкою миші на ньому, щоб відкрити контекстно-залежне меню.
3. У ньому виберіть Options > Show All Attributes.

Щоб показати у класу лише вибрані атрибути:

1. Виділіть на діаграмі необхідний вам клас.
2. Клацніть правою кнопкою миші на ньому, щоб відкрити контекстно-залежне меню.
3. У ньому виберіть Options > Select Compartment Items.
4. Вкажіть потрібні атрибути у вікні Edit Compartment.

Щоб придушити виведення всіх атрибутів класу діаграми:

1. Виділіть на діаграмі необхідний вам клас.
2. Клацніть правою кнопкою миші на ньому, щоб відкрити контекстно-залежне меню.
3. У ньому виберіть Options > Suppress Attributes.

Щоб змінити прийнятий тип атрибута:

1. У меню моделі виберіть Tools > Options.
2. Перейдіть на вкладку Diagram.
3. Для встановлення значень параметрів відображення атрибутів за замовчуванням скористайтесь контрольними перемикачами Suppress Attributes та Show All Attributes. Зміна цих значень за замовчуванням впливатиме лише на нові діаграми. Вигляд існуючих діаграм класів не зміниться.

Як і у разі атрибутів, є кілька варіантів подання операцій на діаграмах.

- Показати всі операції;
- Показати лише деякі операції.

- Сховати всі операції.
- Придушити виведення операцій.

Крім того, можна:

- Показати лише ім'я операції. Це означає, що на діаграмі буде представлено лише ім'я операції, але не аргументи або тип значення, що повертається.
- Показати повну сигнатуру операції. На діаграмі буде представлено не тільки ім'я операції, а й усі її параметри, типи даних параметрів та тип значення операції, що повертається.

Щоб показати всі операції класу:

1. Виділіть на діаграмі необхідний вам клас.
2. Клацніть правою кнопкою миші на ньому, щоб відкрити контекстно-залежне меню.
3. У ньому виберіть Options > Show All Operations.

Щоб показати лише вибрані операції класу:

1. Виділіть на діаграмі необхідний вам клас.
2. Клацніть правою кнопкою миші на ньому, щоб відкрити контекстно-залежне меню.
3. У ньому виберіть Options > Select Compartment Items.
4. Вкажіть необхідні операції у вікні Edit Compartment.

Щоб придушити виведення всіх операцій класу діаграми:

1. Виділіть на діаграмі необхідний вам клас.
2. Клацніть правою кнопкою миші на ньому, щоб відкрити контекстно-залежне меню.
3. У ньому виберіть Options > Suppress Operations.

Щоб показати на діаграмі класів сигнатуру операції:

1. Виділіть на діаграмі необхідний вам клас.
2. Клацніть правою кнопкою миші на ньому, щоб відкрити контекстно-залежне меню.
3. У ньому виберіть Options > Show Operation Signature.

Щоб змінити прийнятий за умовчанням вид операції:

1. У меню моделі виберіть Tools > Options.
2. Перейдіть на вкладку Diagram.
3. Для встановлення значень параметрів відображення операцій за замовчуванням скористайтесь контрольними перемикачами Suppress Operations, Show All Operations та Show Operation Signatures.

Щоб показати видимість атрибута чи операції класу:

1. Виділіть на діаграмі необхідний вам клас.
1. Клацніть правою кнопкою миші на ньому, щоб відкрити контекстно-залежне меню.
2. У ньому виберіть Options > Show Visibility.

Щоб змінити прийняте за промовчанням значення параметра показу видимості:

1. У меню моделі виберіть Tools > Options.
2. Перейдіть на вкладку Diagram.
3. Для встановлення параметрів відображення видимості за промовчанням скористайтесь перемикачем Show Visibility.

Для перемикання між нотаціями видимості Rose та UML:

1. У меню моделі виберіть Tools > Options.
2. Перейдіть на вкладку Notation.
3. Для перемикання між нотаціями скористайтесь перемикачем Visibility as Icons. Якщо цей перемикач позначений, використовуватиметься нотація Rose. Якщо немає, то нотація UML. Зміна цього параметра вплине лише на нові діаграми. Існуючі діаграми класів залишаться незмінними.