Министерство образования Российской Федерации

Владимирский государственный университет

Кафедра информатики и вычислительной техники

История и методология ИВТ

Rational Software

Выполнил: студент группы ИМ-101
Огрызков С. А.

Проверил: Костров А. В.

Владимир 2001

Содержание

На вопрос «Кто есть *Rational Software*?» сама компания отвечает так:

- Одна из самых больших и самых прибыльных в мире компаний, занимающихся разработкой программного обеспечения, с годовым доходом в 815 млн. долларов (по результатам финансового года, закончившегося 31.03.2001).

- Более 3’700 профессионалов в 70 местах двух Америк, Европы, Среднего Востока и Азиатско-Тихоокеанского региона.

- Поставщик, выбранный 90 компаниями из списка 100 крупнейших компаний журнала *Fortune* и 49 компаниями из аналогичного списка 50 крупнейших компаний сферы электронного бизнеса.

- Признанный лидер во множественной разработке приложений и развёртывании рынка на протяжении четырёх лет подряд[1].

- *«Компания, которая добавила слово «унифицированный» к языкам моделирования»*[2].

- Поставщик лидирующих на рынке инструментов, таких как *Rational Rose*, *Rational ClearCase*, *Rational RequisitePro* и *Rational Unified Process*.

- Признанный промышленный инноватор, вкладывающий 17% доходов в исследования по разработке программного обеспечения.

Действительно, компания *Rational Software* несколько лет является лидером в области создания инструментальных средств и методологий в области разработки программного обеспечения (ПО). Решения *Rational* охватывают различные аспекты разработки, включая управление требованиями, визуальное моделирование, тестирование, управление конфигурациями и изменениями, увеличивая производительность команды разработчиков.

[1] По данным *IDC*.
[2] См. JavaPro

Rational Unified Process (RUP)

Rational Unified Process (RUP), то есть унифицированный процесс *Rational* – это методологическая основа для всего, что выпускает компания. Данный продукт является своего рода энциклопедией (методологическим руководством) того, как нужно строить эффективное информационное производство. Также *RUP* регламентирует этапы разработки ПО, документы, сопровождающие каждый этап, и продукты самой *Rational* для каждого этапа. В *RUP* заложены все самые современные идеи. Продукт постоянно обновляется, включая в себя всё новые и новые возможности. К достоинством данной методологии стоит отнести чрезвычайную гибкость – *RUP* не диктует, что необходимо сделать, а только рекомендует использовать то или иное средство.

Компания *Rational* выделяет существующий в мире парадокс разработки ПО – *скорость против качества*. И если раньше можно было выбирать что-то одно, то сегодня, чтобы оставаться конкурентоспособным, приходится соответствовать и тому, и другому. В этом и состоит парадокс – в необходимости работать *быстро и качественно*. Компания *Rational* вызывается разрешить этот парадокс, предлагая следующее:

- Выпускать ПО, пользуясь принципом промышленного подхода. То есть так, как поступают любые заводы и фабрики: определяя стадии, потоки, уточняя обязанности каждого участника проекта. Именно промышленный подход позволит достаточно оперативно выпускать новые версии ПО, которые при этом будут надёжными и качественными.

- Расширять кругозор специалистов для снятия барьеров. Ведь в подавляющем большинстве случаев специалисты из разных отделов просто говорят на разных языках. Соответственно, снятие языкового барьера должно вести к ускорению работы над ПО.

- Использовать итеративную разработку вместо каскадной, существующей в настоящее время. Принцип итерации заключается в повторяемости определённой последовательности процессов с целью доведения элемента до безошибочного состояния.

- Обязательное управление требованиями. Всем известно, что по ходу разработки в систему вносятся изменения (самой группой разработчиков или заказчиком – неважно). *Rational* предлагает мощную систему контроля управления требованиями: их обнаружение и документирование, поддержку соглашений между разработчиками и заказчиками.

- Полный контроль всего происходящего в проекте посредством создания специальных архивов.

- Унифицированный документооборот, приведённый в соответствие со всеми известными стандартами. Это значит, что каждый этап в разработке

(начало, работа и завершение) сопровождаются унифицированными документами, которыми должен пользоваться каждый участник проекта.

- Использование визуального моделирования.

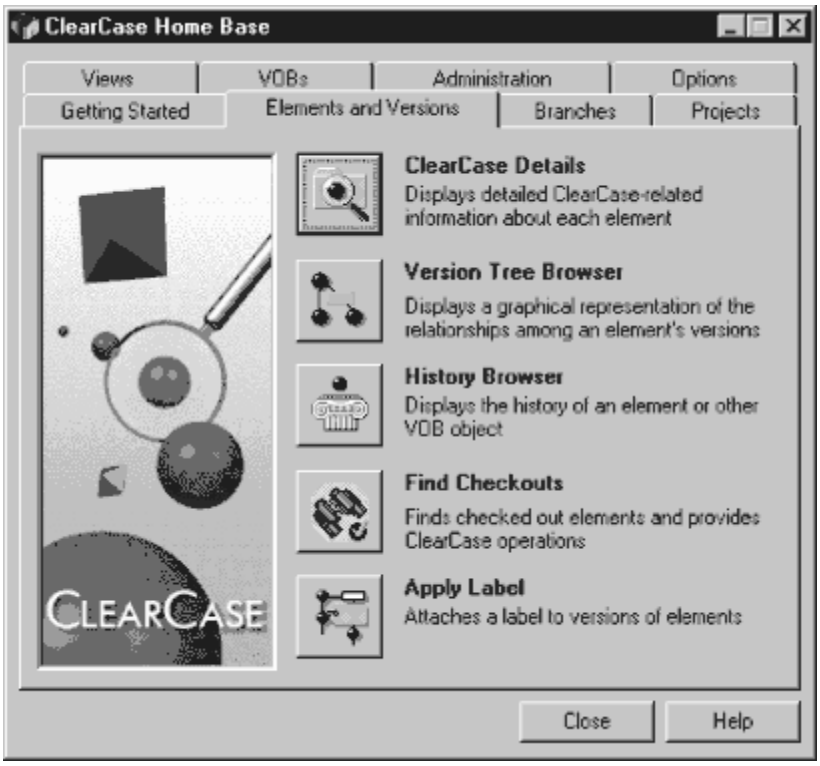
- Применение не только механизмов объектно-ориентированного программирования, но также объектно-ориентированного мышления и подхода.

Сам *RUP* поставляется в виде не обычного программного продукта, а в виде гипертекстовой документации (веб-страниц), что позволяет размещать его во внутренней сети предприятия с целью приобщения всех сотрудников к этому гигантскому источнику полезной информации.

Продукт ориентирован на всех участников проекта.

ClearCase

Рекомендованный как средство контроля для командной разработки, *ClearCase* превосходно справляется с возложенной на него задачей. Являясь, по сути, высоко масштабируемым клиент-серверным приложением, *ClearCase* объединяет всех участников проекта единой средой, хранящей всю возможную информацию, относящуюся к проекту, позволяя получать последние версии редактируемых файлов.



Продукт совмещает полный *SCM* (*Source Code Management* – управление исходным кодом), включая контроль над версиями, и управление рабочим пространством с помощью уникального инвариантного подхода. Посредством *ClearCase* команда разработчиков может ускорить циклы разработки, убедиться в точности выпусков, создавая новые, надёжные в эксплуатации продукты, а также дорабатывать и поддерживать ранее реализованные продукты, организовывать эффективный процесс разработки – и всё это без изменения среды, инструментальных средств и подхода к работе.

Продукт обеспечивает продвинутое управление версиями исходных текстов, библиотек и на протяжении всего жизненного цикла проекта, позволяя тем самым разработчику вернуться к любой версии редактируемого файла и откорректировать его, создав новую версию.

Каждый участник проекта может иметь доступ как ко всем файлам проекта, так и к только определённой его части. Для достижения подобного эффекта *ClearCase* использует мощную систему настраиваемых фильтров, скрывающих ненужную информацию. Система видов разительно отличает *ClearCase* от продуктов конкурирующих фирм, поскольку позволяет осуществить параллельную разработку, а также позволяет отдельному участнику проекта выходить из общего состава разработки, забирая работу

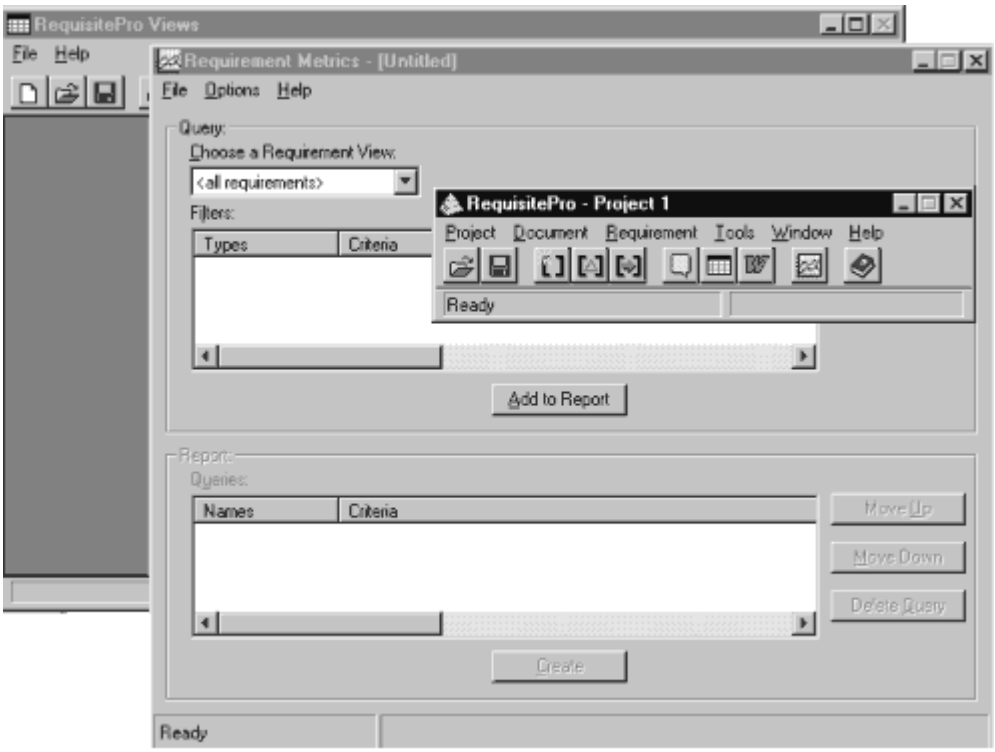
«на дом», а после всех внесённых изменений вернуть версии снова в проект. При этом *ClearCase* осуществит автоматическое слияние версий.

В дополнение к описанным возможностям, *ClearCase* позволяет объединять географически удаленные команды разработчиков посредством *MultiSite* – специального модуля, осуществляющего репликацию (передачу) текущего состояния проекта на указанный сайт.

В условиях бурно развивающейся и подверженной изменениям информационной индустрии становится всё сложнее и сложнее давать оценку программному продукту как чему-то независимому, вырванному из общего контекста разработки. Поэтому принимается во внимание степень поддержки данного продукта теми или иными средствами компаний, создающих средства разработки. В частности, продукт версионного контроля не может быть функционально полным без определённых механизмов интеграции со средствами разработки, с различными дополнительными генераторами отчётов и пр. И ещё, в дополнении к сказанному, хочется отметить, что продукт осуществляет при помощи специальной утилиты *OMake* сборку (компиляцию) проекта не только по последним версиям файлов, но и по любым версиям, взятым с любых параллельных проектов.

Продукт ориентирован на всех участников команды: директоров, менеджеров, разработчиков, аналитиков, тестировщиков, технических писателей.

ИС любых размеров невозможно строить в отрыве от консультаций с заказчиком. Естественно, подобные консультации ведут к постоянному уточнению функций системы (выяснение требований). Подобные переделки строящейся системы редко ведут к её улучшению, поскольку стороны изначально не договаривались о том, что система меняется (уточняется) в процессе разработки. Стало быть, все изменения носят шоковый характер, нанося немалый ущерб как моральный, так и материальный. С точки зрения компании *Rational*, если на первом этапе выяснения функций планируемой системы у заказчика использовался инструмент *Rational Rose*, то 90% функций будет оговорено на самом начале пути разработки. И всё же требования будут уточняться, но пусть это будет обыденный и безболезненный процесс. *RequisitePro* занимается как раз управлением требованиями.



RequisitePro – это удобный инструмент для ввода и управления требованиями, который может использоваться всеми участниками команды. Продукт позволяет в наглядной форме получать, выводить и структурировать наборы вводимых требований.

Для каждого требования поддерживается набор атрибутов, позволяющий эффективно управлять проектом на основе задания иерархий требований, установки их приоритетов, сортировки, назначения требований конкретным исполнителям. Для требований предусмотрен, предопределён набор атрибутов, который может быть расширен пользователем по своему усмотрению, что позволяет характеризовать требования в соответствии с представлениями пользователя.

Развитые возможности прослеживания требований позволяют визуально определять схожие требования в рамках одного или нескольких

проектов. Это даёт возможность применения готовых апробированных решений в новом проекте. Возможность задания связей между требованиями позволяет легко проследить, какие требования следует подвергнуть анализу (и, возможно, пересмотру) при модификации некоторого конкретного требования или атрибута. Тем самым упрощается процесс внесения изменений.

Для каждого требования хранится его история, позволяющая отследить, какие изменения были внесены в требование, кем, когда и почему.

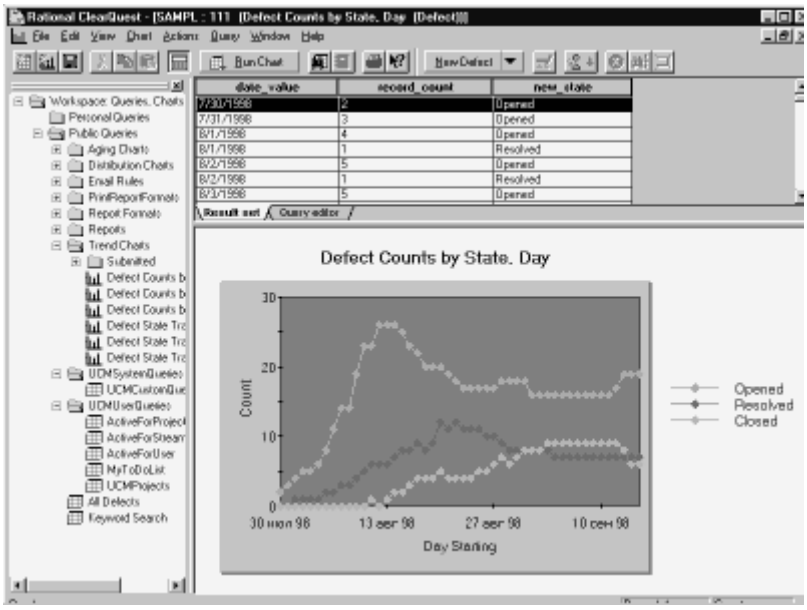
Выгоды эффективного управления требованиями с помощью *RequisitePro* увеличиваются экспоненциально при использовании его всей командой разработчиков. *RequisitePro* упрощает общение между разработчиками путем предоставления общего доступа ко всем требованиям проекта, либо к их части.

Все документы и данные, относящиеся к требованиям, централизованно организуются при помощи *RequisitePro*. Требования заказчика, дизайн подсистем, сценарии, функциональные и нефункциональные спецификации и планы тестирования распределяются и связываются таким образом, чтобы максимально облегчить управление проектом.

Продукт также направлен на всех участников проекта.

ClearQuest

Ещё одно решение от *Rational* – это средство управления запросами на изменение – *ClearQuest*. Поскольку проект постоянно меняется, руководителям, менеджерам жизненно необходимо знать статистику того, что менялось, кем и в какое время. В большинстве же случаев необходимо видеть не просто объёмную информацию о том или ином событии, а хочется иметь полный объем необходимой информации в виде базы данных (БД), скажем, на *Oracle*. *ClearQuest* помогает в решении данной задачи.



ClearQuest является мощным средством управления запросами на изменение (*Change Request Management – CRM*), специально разработанным с учётом динамической и сложной структуры процесса разработки ПО. *ClearQuest* отслеживает и управляет любым типом действий, приводящих к изменениям в течение всего жизненного цикла продукта, помогая тем самым организациям более предсказуемым (правильным) образом создавать качественное ПО. Основные задачи, решаемые посредством *ClearQuest*:

- Управление изменениями, возникающими в ходе процесса разработки ПО.
- Оптимизация пути прохождения запросов на изменения, а также связанных с ним формы и процедуры.
- Поддержка связи внутри команд, разделённых территориально, при помощи веб-технологий и Интернета.
- Внедрение надёжный и проверенный процесса *CRM*, либо изменение уже существующего процесса для удовлетворения специфическим требованиям.

- Визуальный анализ полученного прогресса проекта с помощью богатых возможностей графического представления информации и отчётов.

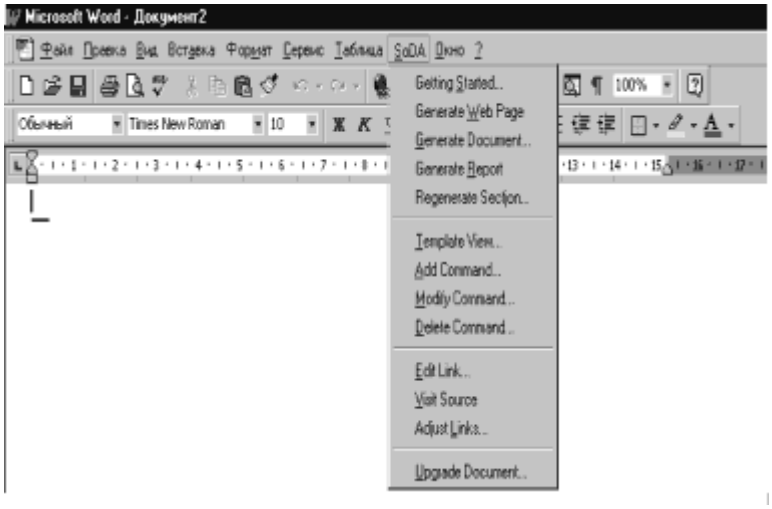
- Интеграция со средствами конфигурационного управления, такими как *Rational ClearCase*, позволяющая создавать связи между запросами на изменение и развитием кода.

В качестве положительных черт данного продукта можно отметить его адаптивность (то есть продукт можно на месте доработать с учётом конкретной функциональности), масштабируемость, простоту в администрировании и использовании.

При работе с программой каждый участник проекта может заходить в базу и определять собственные запросы. Запросы на изменения проходят цикл из нескольких состояний, начиная с подачи и заканчивая их разрешением. Например, только что поданный запрос находится в состоянии «подан» (*submitted*). После передачи запроса сотруднику он переходит в состояние «назначен» (*assigned*). Начало работы над запросом переводит его в «открытое» (*open*) состояние, и вся команда может видеть, что кто-то обрабатывает запрос. Наконец, когда запрос проверен и закрыт, он проходит соответственно стадии «проверки» (*verify*) и «закрытия» (*resolved*). Таким образом, любой участник проекта, от подчинённого до руководителя, может пользоваться базой в собственных целях.

SoDA

Результатом любой деятельности, а уж тем более по созданию информационной системы (ИС), является документ или отчёт заранее установленного образца. Ещё лучше, когда внутренние стандарты как-то соотносятся с общепринятыми мировыми. Последнее особенно важно интернациональным командам, работающим совместно с зарубежными партнёрами. На решение всех проблем с документооборотом направлен инструмент *Rational SoDA*. Его основная обязанность – подготовить отчёт по заранее установленному шаблону. Данные для отчёта берутся из любой программы *Rational*. Например, необходимо получить готовый документ по имеющейся модели в *Rational Rose*. *SoDA* позволит сгенерировать подобный отчёт в считанные минуты, не упустив при этом ни одной детали. Само собой разумеется, что всю работу по написанию может взять на себя человек, скажем, технический писатель, но ему будет очень трудно извлекать из моделей спецификации и комментарии, перенося в текстовый редактор – это неправильный подход, ведь все это *SoDA* сделает всё сама в автоматическом режиме, представив результат в виде обычного документа в формате *Microsoft Word*.



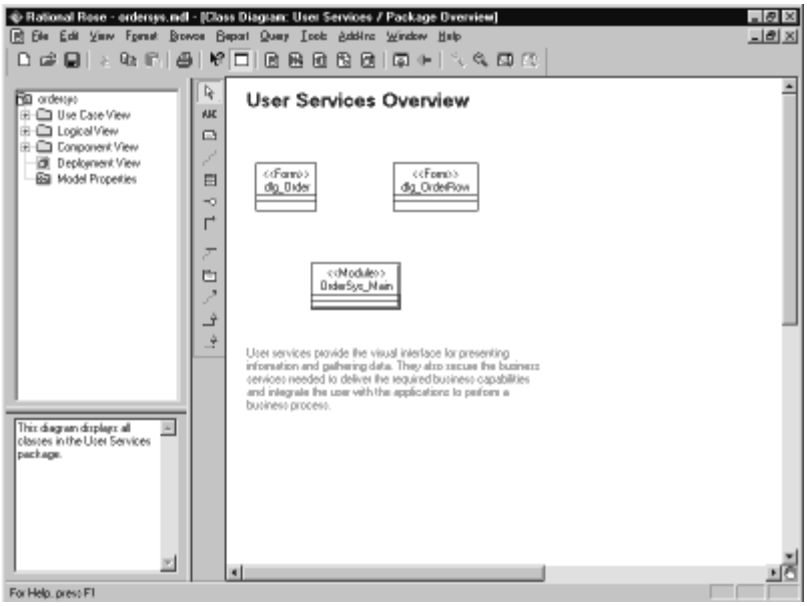
По сути дела, *SoDA* является макросом для *Microsoft Word*. Система вызовов и меню интегрирована с *Word* и позволяет генерировать шаблоны на базе имеющихся файлов. *SoDA* допускает к использованию как стандартных шаблонов, так и созданных пользователем при помощи специального мастера, также встроенного в систему меню *Word*.

Набор отчётов, поддерживаемых *SoDA*, таков: *Rational Rose*, *RequisitePro*, *ClearCase*, *TeamTest*.

Продукт обязателен для использования в составе любого проекта. Также особенную направленность продукт имеет на разработчиков и постановщиков задач.

Rational Rose

Продукт № 1 в программном списке *Rational*. Следуя рекомендации и практическому опыту, данный продукт позиционируется для использования проектировщиками, аналитиками, разработчиками. *Rose* является уникальным CASE-средством, чьи графические возможности, основанные на *UML* (*Unified Modeling Language* – унифицированный язык моделирования), способны решить любые задачи, связанные с любым проектированием и моделированием: от общей модели процессов (абстрактной) предприятия до конкретной (физической) модели класса в создаваемом ПО. Работа в *Rational Rose* заключается в проектировании определённого вида диаграмм, задавая при этом все свойства, отношения и взаимодействие друг с другом.



При разработке любой ИС в первую очередь возникает проблема взаимопонимания подрядчика и заказчика уже на стадии договорённости о структуре системы. Имея такой инструмент, как *Rose*, проектировщик (аналитик) всегда может показать заказчику не абстрактное словесное описание процесса, а его конкретную модель (на экране монитора или в печатном виде – неважно). Значит, *Rose* позволит быстрее уладить с заказчиком все детали планируемой системы. Как говорилось выше, *RUP* описывает все артефакты (документы), возникающие по ходу проекта, так и в *Rose* результатом моделирования является файл с моделью, которую проектировщик передает следующему звену сотрудников – кодировщикам, которые дополняют полученную логическую модель системы моделями конкретных классов на конкретном языке программирования.

Необычайно богатый набор средств *Rose* предоставляет разработчикам следующие возможности:

- *Кодогенерация* – позволяет преобразовать нарисованную модель в описание на конкретном языке программирования. По умолчанию поддерживаются языки *Ada*, *BASIC*, *C++*, *Java*, *Oracle*, *XML*. Также к *Rose* сторонними компаниями разрабатываются специальные мосты к не

входящим стандартную поставку языкам, например, к *Object Pascal* (язык среды *Delphi*).

- *Реинжиниринг* – когда готовую ИС (например, на *C++*) или БД (на *Oracle*) загружают в *Rose* с целью получения наглядной визуальной (структурной) модели.

- *Двухстороннее проектирование* (*round-trip engineering*) – сочетает возможности первых двух подходов, когда создается система, а по прохождении некоторого времени эволюционного периода подвергается вновь реинжинирингу и вновь кодогенерации.

В настоящее время *Rational Rose* поставляется в следующих редакциях:

- *Rose DataModeler* – позволяет проектировать любые системы и БД без возможности кодогенерации; продукт направлен на аналитиков и проектировщиков.

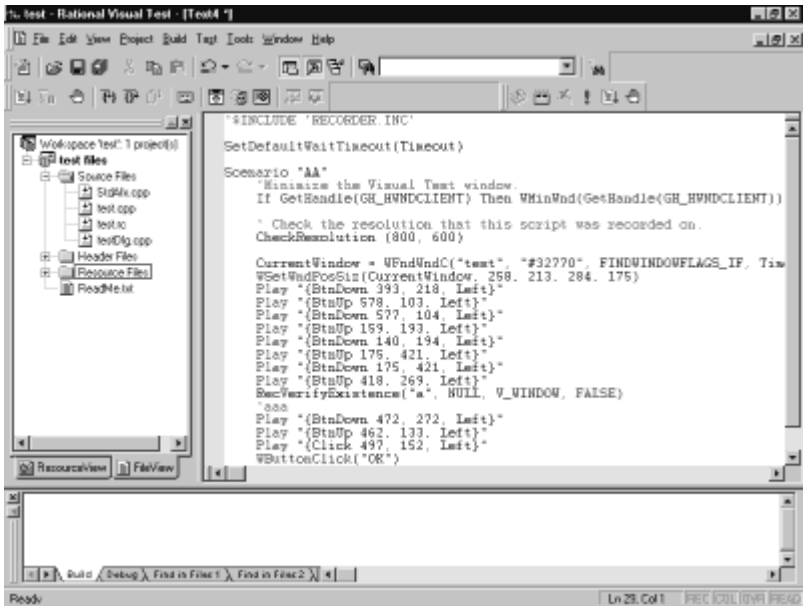
- *Rose RealTime* – узкоспециализированная версия, способная проводить 100%-ную кодогенерацию и реинжиниринг только на *C* и *C++* и имеющая неполный набор диаграмм; продукт направлен только на разработчиков.

- *Rose Enterprise* – наиболее полная версия, включает в себя все вышеописанные возможности.

В целом, *Rational Rose* направлен на проектировщиков, аналитиков и разработчиков широкого профиля.

Visual Test

Visual Test предоставит то, что разработчики программного обеспечения ищут в любом инструменте – гибкость и мощь, одновременно обеспечивая «любимое» свойство менеджмента – исключительный доход от инвестиций. Основное свойство программы должно особенно порадовать программистов, поскольку приставка *visual* в названии программы не случайна – продукт имеет интерфейс, сходный с *Microsoft Visual Studio*. То есть при изучении продукта тратится минимум времени.



Продукт по своей природе достаточно «всеяден», позволяя производить любое тестирование, 32-разрядных Windows-приложений, компонентов ActiveX, DLL, серверов автоматизации OLE или веб-приложений. *Visual Test* является автоматизированным инструментом тестирования для любых задач.

Visual Test дает возможность создавать поддерживаемые, расширяемые и пригодные для повторного применения компоненты тестирования, которые можно приспосабливать ко многим версиям и, после некоторого планирования, ко многим проектам.

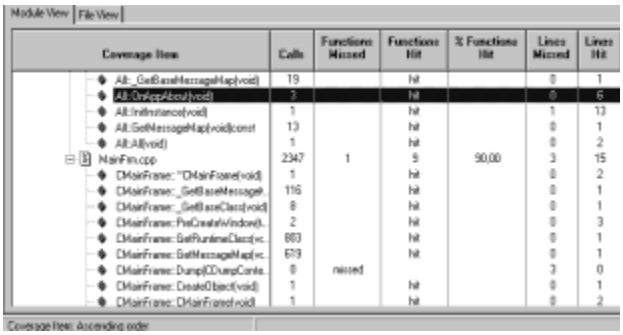
Основу гибкости и мощи *Visual Test* составляет производный от *Visual BASIC* расширенный язык программирования *Test BASIC*, с сотнями специфических для теста функций, специальных конструкций для облегчения тестирования, простого доступа к интерфейсу прикладных программ Windows (*WinAPI*) и открытой архитектурой, которая делает этот язык расширяемым.

Данный продукт обеспечит надёжное функциональное тестирование.

Продукт ориентирован на разработчиков и тестировщиков.

PureCoverage

Rational PureCoverage позволит разработчикам довести собственные программы до состояния абсолютной эффективности, освободив от ошибок и странностей.



Основное и единственное назначение продукта – выявление участков кода, пропущенных при тестировании приложения. Вполне очевидно, что при тестировании (даже при самом дотошном) программы специалисту не удаётся оттестировать абсолютно все её функции. А невозможно это, как правило, по двум причинам: во-первых, разработчик не может сделать всё абсолютно правильно с учётом всех возможных нюансов, а во-вторых, даже учитывая все возможные реакции приложения на внешние «раздражители» невозможно на 100% быть уверенным в том, что всё оттестировано. Достаточно обидно будет получить системную ошибку после просьбы заказчика ткнуть мышью в красивую кнопку на экране. После такого провала уже никому не доказать, сколько бессонных ночей проведено в детальном всестороннем тестировании программы, и что как раз именно не прошедшая тестирование функция привела к фатальному сбою... Но если Вы, как разработчик, воспользуетесь средством *Rational PureCoverage*, то сможете раз и навсегда забыть о мучительном поиске невыполненного кода в собственной программе.

PureCoverage собирает статистику о тех участках программы, которые во время тестирования не были выполнены (пройдены). Дополнительно, программа считает так называемые *«хиты»* (*hits*) – активно исполнявшиеся строки.

Стало быть, разработчик может оценить не просто, сколько раз вызывалась та или иная функция, а сколько раз исполнилась каждая строка, составляющая её. Имея подобную статистику, очень просто выявить неисполнившиеся строки и проанализировать причину, по которой они не получили управления.

Подобные строки *PureCoverage* подсвечивает красным цветом, чётко указывая на наличие «чёрных дыр» в программе в виде неоттестированного кода и, тем самым, давая разработчику хорошую пищу для размышлений. В работе данный инструмент так же прост, как и вышеописанные.

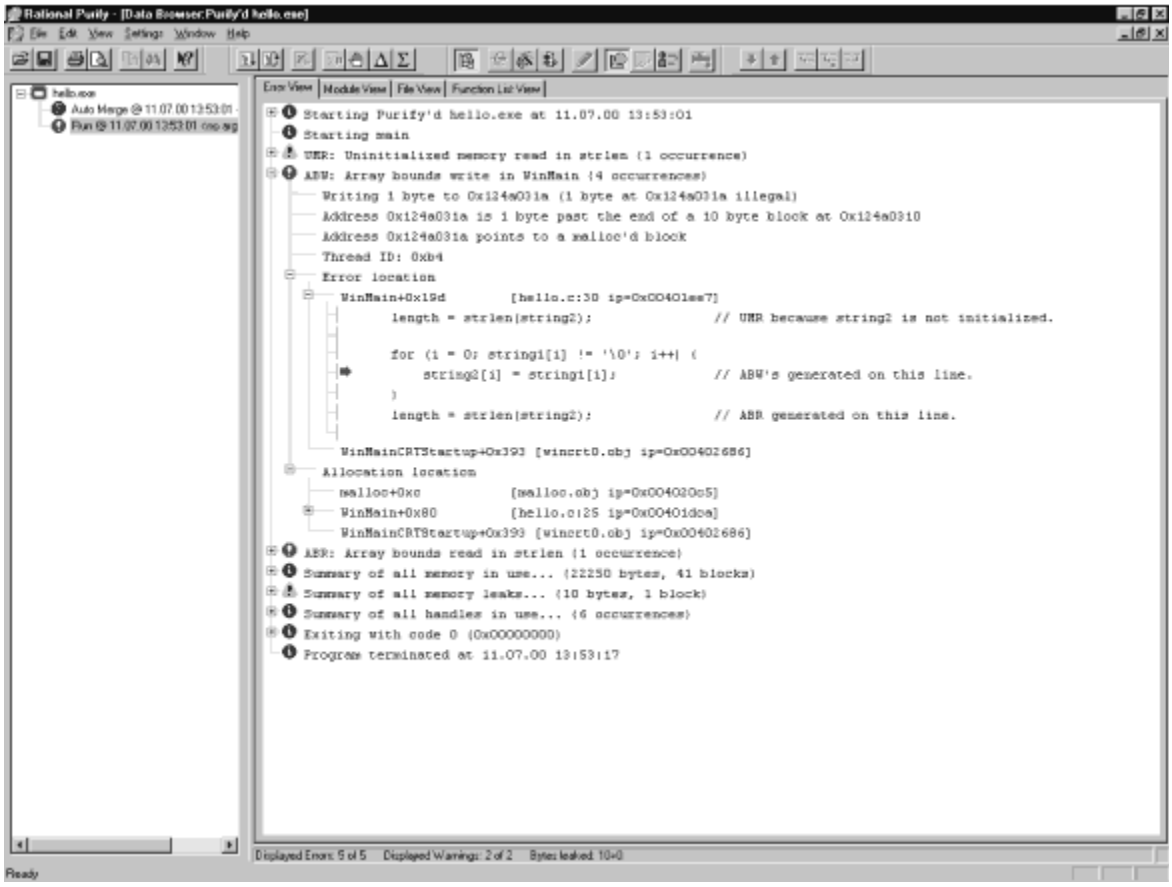
Продукт направлен на разработчиков.

Purify

Данный продукт направлен на разрешение всех проблем, связанных с утечками памяти и ошибками времени выполнения. Не секрет, что многие программные продукты ведут себя «не слишком скромно», замыкая на себя во время работы все системные ресурсы без большой на то необходимости. А это и есть путь, который приведёт готовую систему к краху в самый ответственный момент. Возникновение подобного рода ошибок достаточно трудно отследить стандартными средствами, имеющимися в арсенале разработчика. И дело тут не только в том, что разработчик может где-то недосмотреть, где-то пересмотреть, а в том, что в подавляющем большинстве случаев проектные сроки вынуждают смотреть «сквозь пальцы» на неточности.

Стандартного, общепризнанного рецепта для решения подобных проблем не было до недавнего времени (просто не было продукта, который бы взял на себя поиск подобных ошибок, освободив разработчика для более абстрактных, концептуальных материй).

Как видится, имея в своем распоряжении надёжный инструмент, который бы сам в процессе работы над проектом указывал на все «чёрные дыры» в использовании памяти, разработчики начали бы его повсеместное внедрение, повысив надёжность создаваемого ПО.



В общих чертах работа *Purify* сводится к выводу детальнейшей статистики об использовании памяти приложением. Программа собирает данные о любых потерях в памяти. К ним можно отнести и банальное невозвращение блока, и неиспользование указателей, и остановку

исполнения программы с выводом состояния среды при возникновении ошибки времени выполнения.

Purify предоставляет возможность разработчику не только видеть состояние исполнения (предупреждения, ошибки), но и переходить к соответствующему исходному тексту (естественно, такая возможность существует только для внутренних вызовов, поскольку исходные тексты динамических библиотек пока программистам недоступны).

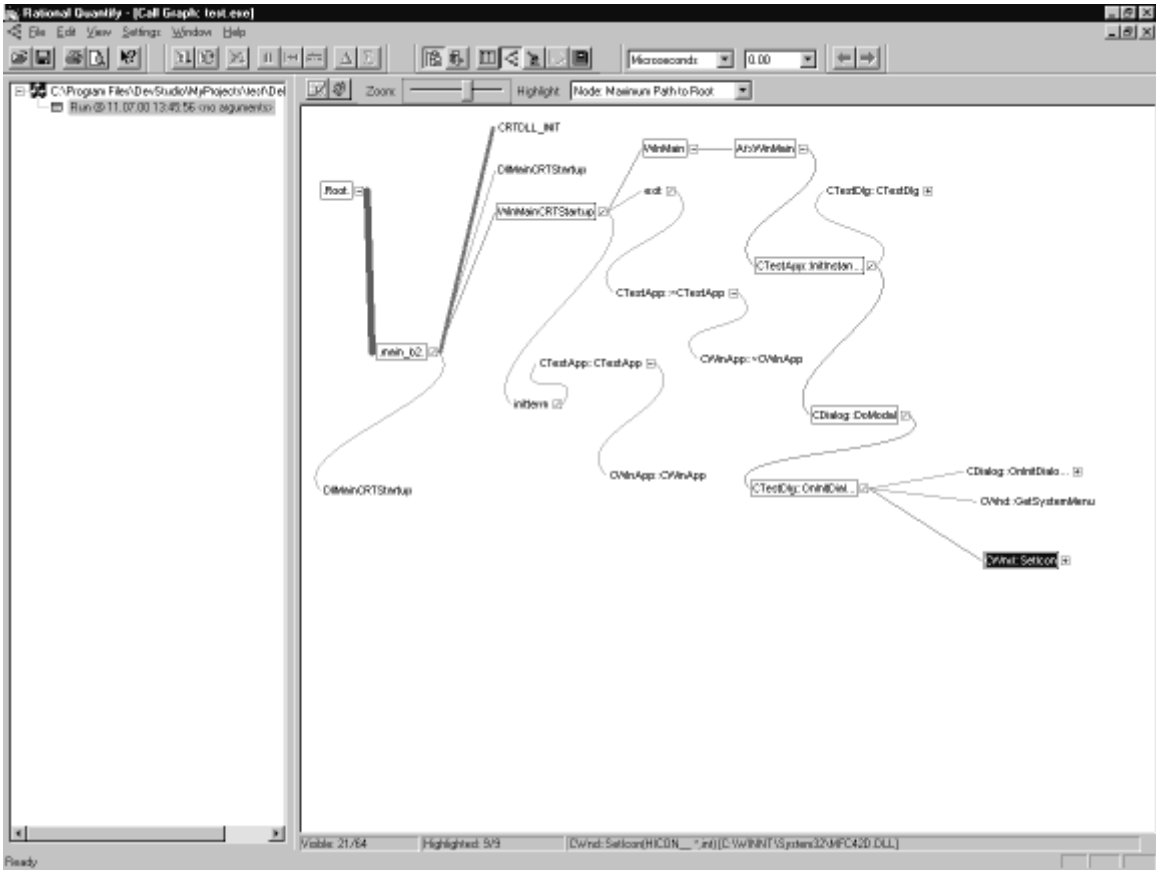
Очень мощный продукт, обладает достаточно простым интерфейсом и осваивается специалистом за 2-3 дня.

Продукт ориентирован на разработчиков.

Quantify

Разработчик постоянно сталкивается с проблемой необходимости увеличения производительности собственного приложения в сжатые сроки и с максимально возможной эффективностью. Проблема вечная по определению и нетривиальная по сути, поскольку, из-за отсутствия необходимого инструментария, придётся вручную замерять скоростные характеристики разрабатываемых приложений, дописывая для этого килобайты отладочных функций и циклов вызова. А если недосуг набивать горы новых тест-функций, придется обзавестись таймером и замерять фрагменты кода с точностью до секунды. Естественно, что подобную работу нельзя назвать продуктивной.

Rational Quantify позволяет решить описанные выше проблемы за один проход. Это простое, но в то же время мощное и гибкое средство учёта производительности приложений – незаменимый инструмент для сбора и анализа информации о производительности созданного программного продукта.



Quantify генерирует в табличной форме список всех вызываемых в процессе работы приложения функций, указывая временные характеристики каждой из них.

Quantify предоставляет полную статистическую выкладку по всем вызовам (внешним и внутренним), невзирая на размеры тестируемого приложения и время его тестирования. Сбор данных осуществляется посредством технологии *OCI* (*Object Code Insertion* – вставка объектного кода). Суть способа состоит в подсчёте всех машинных циклов путём

вставки счётчиков в код для каждого функционального блока тестируемой программы (все циклы приложения просчитываются реально, а не при помощи произвольных выборок, как в большинстве пакетов тестирования). Уникальность данного подхода заключается в том, что, во-первых, тестируется не только сам исходный код, но и все используемые компоненты, (например: библиотеки DLL, системные вызовы), а во-вторых, для подобного анализа совсем необязательно иметь исходные тексты тестируемого приложения (правда, в этом случае нет возможности отслеживать внутренние вызовы).

Статистическая информация по вызовам может быть перенесена в *Microsoft Excel*, где можно построить как графики, так и сводные таблицы для разных запусков программы.

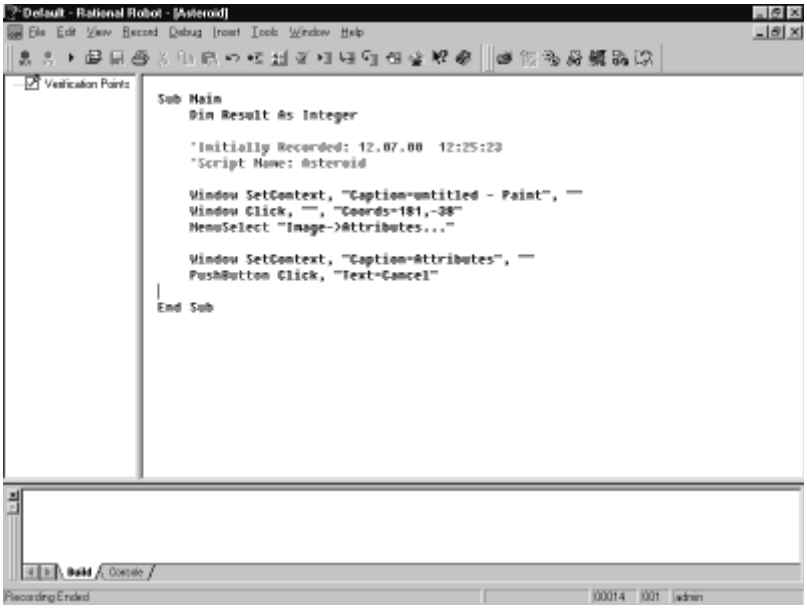
Ещё хочется поставить акцент на то, что тестируемое приложение можно перекомпилировать и запустить повторно, при этом *Quantify* способен запомнить все предыдущие вызовы и дать сравнительную оценку.

Продукт ориентирован на разработчиков.

Robot

Rational Robot – средство функционального тестирования, базирующееся на объектно-ориентированной технологии, что позволяет существенно превзойти традиционные средства тестирования графического интерфейса, так как здесь тестируются сотни и тысячи свойств всех объектов приложения (даже скрытых объектов), как вместе, так и каждого в отдельности.

Программа способна работать в двух режимах: в автоматическом и ручном. В ручном режиме пользователь сам задаёт на специальном языке сценарий тестирования, в автоматическом – пользователь тестирует приложение, а *Robot* автоматически генерирует необходимый сценарий для дальнейшего повторного тестирования.



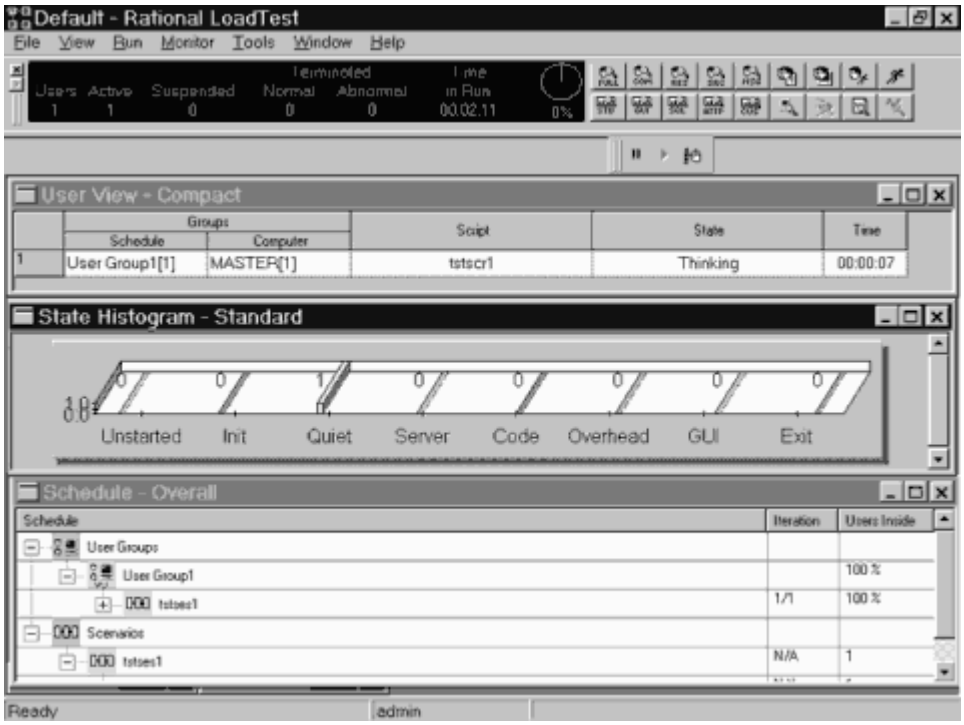
Сценарии, создаваемые в *Rational Robot*, обеспечивают поиск ошибок в приложении, оставаясь виртуально независимыми от внесённых изменений и платформы. Без дополнительных изменений сценарии могут использоваться на *Microsoft Windows 95*/98/NT. Объектное тестирование обеспечивает быстрое создание сценариев, которые в дальнейшем можно легко изменять, создавать заново и воспроизводить. *Rational Robot* поддерживает широкий спектр языков программирования и ERP-решений. *Rational Robot* позволяет редактировать, отлаживать и настраивать сценарии. Допускает также тестирование сложных клиент-серверных систем на платформе Windows.

Внедрение подобного универсального инструмента для всеобъемлющего тестирования графического интерфейса даст существенный прирост в эффективности проводимого тестирования.

Продукт ориентирован на разработчиков и тестировщиков.

LoadTest

LoadTest – средство автоматизированного тестирования характеристик распределенных сетевых приложений на платформах Windows и Unix.



Тесты производительности выполняются с помощью программы *LoadTest*. При этом тестировании типично используется нагрузка сервера большим количеством виртуальных пользователей (*Virtual Users – VU*). Например, Вы можете установить таймер для одного *VU*, чтобы определить, сколько времени займёт выполнение запроса, когда тысячи других *VU* посылают запросы на тот же самый сервер в то же самое время. Термин «тесты производительности» включает нагрузочные, стрессовые, конкурирующие и конфигурационные тесты. Совокупность этих тестов позволяет ускорить цикл тестирования производительности и достигнуть значимых и точных результатов.

Нагрузочное тестирование с использованием *LoadTest* выполняется тогда, когда нужно определить время отклика серверов или клиентских приложений при изменяющейся нагрузке. Нагрузочное тестирование также используется тогда, когда нужно вычислить, какое максимальное количество транзакций может выполнить сервер за определённый временной отрезок. Если клиент-серверная система использует распределённую архитектуру или средства балансировки нагрузки, то нагрузочное тестирование может быть использовано для того, чтобы проверить правильность выбранных методов для балансирования или конструирования системы.

Нагрузочное тестирование выполняется как с использованием режима только виртуальных пользователей, когда измеряется время отклика только серверной части или комбинации виртуальных пользователей, так и пользователей графического интерфейса для измерения времени отклика системы для конкретного клиентского приложения.

Таким образом, становится возможным при использовании данной программы проверять на производительность любую клиент-серверную систему.

Продукт направлен на тестировщиков, разработчиков, веб-разработчиков.

Наборы Rational

Продукты компании *Rational* полностью оправдывают свою не очень низкую стоимость, поэтому компания, идя навстречу пожеланиям заказчиков, объединяет их в специальные тематические группы – *наборы* (*suite*).

В набор входят основные программные продукты, направленные на покрытие одного или нескольких этапов разработки программного обеспечения. Данный подход вполне оправдывает себя, поскольку, например, команде аналитиков незачем переплачивать за средства тестирования, которые им не нужны, и, наоборот, тестировщикам ни к чему ставить *Rose DataModeler* для проектирования баз данных. Соответственно ни к чему не только иметь такой продукт, но и переплачивать за него. Второе преимущество наборов заключается в том, что их легче устанавливать и администрировать, поскольку продукты идут комплектом, а не разрозненно.

Бесспорно, что покупка наборов более выгодна как с финансовой точки зрения, так и с практической, но если Вам всё же необходимо нечто разрозненное, то Вы можете приобретать продукты по отдельности. Данный подход никак не скажется на степени интеграции программных компонентов друг с другом. Все разрозненные программы будут работать также слажено, как если бы входили в один набор. Мало того, Вы можете к уже купленному набору подключать разрозненные программы, повышая, тем самым, его насыщенность без ущерба совместимости и интегрируемости.

Итак, как говорилось выше, различные наборы направлены на покрытие определенных этапов, описанных в *RUP*. Давайте подробнее разберёмся со всем многообразием наборов, предлагаемых *Rational*, а для начала приведем сводную таблицу наличия продуктов в том или ином наборе (табл. 1):

Таблица 1

Наборы Rational и их комплектность

Продукт	**Наборы**				
	Analyst Studio	**Development Studio**	**Test Studio**	**Enterprise**	**Performance Studio**
RUP	**+**	**+**	**+**	**+**	**+**
ClearCase	распространяется отдельно				
RequisitePro	**+**	**+**	**+**	**+**	**+**
ClearQuest	**+**	**+**	**+**	**+**	**+**
SoDA	**+**	**+**	**+**	**+**	**+**
Rose[3]	DM	RT, E		**+**	E
Visual Test	распространяется отдельно				
PureCoverage		**+**	**+**	**+**	**+**
Purify		**+**	**+**	**+**	**+**
Quantify		**+**	**+**	**+**	**+**
Robot			**+**	**+**	**+**
LoadTest					**+**

[3] Варианты *Rational Rose*: *DM – DataModeler*, *RT – RealTime*, *E – Enterprise*.

Как видно из табл. 1, покупка любого набора подразумевает приобретение унифицированного набора из четырех продуктов (*RUP*, *RequisitePro*, *ClearQuest* и *SoDA*) плюс специальное программное обеспечение. Рассмотрим более подробно назначения приведенных наборов.

- *Analyst Studio Suite*. Направлен на определение и управление полным и чётким набором требований на разработку проекта. Данную редакцию продукта можно рекомендовать аналитикам. Данная поставка включает в себя для проектирования *Rose DataModeler*, которая неспособна производить кодогенерацию.

- *Development Studio Suite*. Обеспечивает все функции визуального моделирования ИС на основе известных и проверенных продуктов. *Development Studio* предоставляет всё необходимое для создания высококачественного программного продукта в установленные сроки и без превышения бюджета. Продукт ориентирован на аналитиков, разработчиков и проектировщиков. В зависимости от поставки может включать *Rose RealTime* или *Rose Enterprise*.

- *Test Studio Suite*. Представляет собой набор инструментов, предназначенных для детального тестирования приложений. Позволяет избавиться от рутинной работы по тестированию приложений. Редакция рекомендуется для тестировщиков.

- *Performance Studio Suite*. Обеспечивает поддержку полного жизненного цикла разработки ИС, позволяет проводить тестирование приложения под нагрузкой, быстро создавать тесты, эмулирующие работу большого числа пользователей, с целью определения производительности и надёжности распределённого приложения. Ориентирован на всех участников проекта.

- *Enterprise Suite*. Полнофункциональная редакция пакета, обеспечивает поддержку полного жизненного цикла разработки ИС. Продукт ориентирован на менеджеров проекта, отдельных программистов, выполняющих несколько функциональных ролей в команде разработчиков

Какова же общая рекомендация по использованию и приобретению наборов? По сути, выше все уже достаточно подробно расписано, но можно добавить следующее: если Вы не знаете конкретно, будет компания заниматься только проектированием или проектированием и разработкой, то, скорее всего, Вам необходимо приобрести *Analyst Studio*, и, поработав с ним некоторое время, модернизировать до следующей ступени. Также есть смысл приобретать *Analyst Studio* для проведения моделирования бизнес-процессов. Для данных целей набор содержит всё необходимое.

Если же ваша компания специализируется только на разработке программного обеспечения, то здесь есть смысл приобрести *Development*

Studio. А в зависимости от поставленных задач просто *Development Studio* (с *Rose Enterprise*) или *Development Studio RealTime*. По функциональности они абсолютно одинаковы, за исключением версии *Rose*. Соответственно, данный продукт не содержит мощных средств тестирования.

Для тех же, кому надо покрыть сразу весь цикл, можно порекомендовать *Performance Studio Suite* и *Enterprise Suite*.

Политика лицензирования

Пожалуй, самая «секретная» тема. Поскольку до того, как не встал вопрос о покупке, редко узнают о схемах лицензирования. По примеру наборов, схема лицензирования такая же увесистая. Обилие схем объясняется тем, что компания *Rational* пытается охватить как можно больший сегмент рынка программных средств и инструментов. В следствие даются различные схемы приобретения и скидки.

Итак, сначала о лицензиях. Лицензии бывают двух видов: *NodeLocked* и *Floating*:

- *NodeLocked* – одиночная лицензия, которая ставится только на один компьютер. Соответственно, запуск программы с данным видом лицензии возможен только на конкретном компьютере, для которого и была сгенерированна лицензия.

- *Floating* – «плавающая» лицензия. Данная лицензия ставится на сервер компании. При получении данной лицензии оговаривается количество работающих станций. Например, куплено 12 лицензий на какой-то продукт, соответственно одновременно с ним могут работать могут только 12 пользователей, а на скольких рабочих станциях будет установлено программное обеспечение – неважно. Данный вид лицензии предусматривает установку специального программного обеспечения на сервер и подразумевает наличие доступа к локальной сети с каждого рабочего места.

Ключи (лицензии) делятся не только на два вышеуказанных типа, но ещё и по времени действия. Ключи бывают *временные* и *постоянные*. Временные ключи выдаются пользователю с целью ознакомления с продуктом на срок 20-45 дней. В *Rational* отлично понимают, что потенциальный заказчик не будет вкладывать крупные деньги в автоматизацию производства, если не будет уверен в том, что данная покупка именно то, что нужно. Для этих целей предназначена временная лицензия, которая не ограничивает продукт функционально, а сокращает его срок службы. Соответственно, демо-версий программ у *Rational* нет – все продукты, – и пробные, и рабочие, – полнофункциональны, но в зависимости от типа лицензии имеют определённый срок работы.

С постоянной лицензии немного сложнее, для её получения необходимо заполнить специальную анкету, пропустить через программу лицензирования и отправить в головной офис *Rational*. Через некоторое время из *Rational* присылают «готовый к употреблению» ключ (работать который будет только на той машине, с которой был послан запрос).

Компания *Rational* является лидирующей компанией в области создания методологий и программных решений, ориентированных на программистов, аналитиков, тестировщиков. Спектр выпускаемого обеспечения целиком покрывает потребность всех участников проекта: от аналитиков до разработчиков и специалистов по внедрению. Все программно-методологические решения – плод многолетнего труда аналитиков и разработчиков как самой *Rational*, так и её партнеров. В итоге, все решения были собраны воедино. Так появился *RUP* – методологическая энциклопедия, в которой описаны все шаги, необходимые для создания качественного программного продукта. Пользуясь подобной энциклопедией и применяя соответствующие инструменты, рекомендуемые *Rational*, команда будет создавать обеспечение качественно и в срок. «Строй быстрей и качественней!» – вот лозунг, выдвигаемый *Rational*.

Список использованных источников

1. Официальный сайт компании *Rational Software* – http://www.rational.com/.

2. The Rational Platform for Software Development. Overcoming the Software Development Paradox. © Copyright 2001 by Rational Software Corporation.

3. Сервер информационных технологий – http://www.citforum.ru/.

4. Новичков А.Н. Эффективная разработка программного обеспечения с использованием технологий и инструментов компании *Rational*. – М.: *Interface Ltd.*, 2000.

5. Новичков А.Н. *Rational Rose* для разработчиков и ради разработчиков. В 3 ч. – М.: *Interface Ltd.*, 2000.

6. Новоженов Ю. Опыт реинжиниринга объектно-ориентированного комплекса программ с применением CASE-средства *Rational Rose* и *SilverRun*. – М.: Аргуссофт.

7. Умников А. *InterSystems* объединяет *Cache* и *Rational Rose*. – М.: *InterSystems GmbH*, 1999.

8. Новичков А.Н. Средства тестирования приложений для разработчиков. Инструменты от компании *Rational Software*. – М.: *Interface Ltd.*, 2000.

9. Новичков А.Н. *ClearCase* – система конфигурационного и версионного контроля. – М.: *Interface Ltd.*, 2000.

10. *Rational Concepts* представляет основанное на *Java* ПО для управления стоимостью. / Новости IT-компаний на *CITForum.ru* – М.: Центр информационных технологий, 2000.