Особенности программных средств, используемых в разработке информационных систем

Silverrun

Silverrun

CASE-средство Silverrun американской фирмы Computer Systems Advisers, Inc. (CSA) используется для анализа и проектирования ИС бизнес-класса и ориентировано в большей степени на спиральную модель ЖЦ.

Оно применимо для поддержки любой методологии, основанной на раздельном построении функциональной и информационной моделей (диаграмм потоков данных и диаграмм "сущность-связь").

Silverrun

Настройка на конкретную методологию обеспечивается выбором требуемой графической нотации моделей и набора правил проверки проектных спецификаций. В системе имеются готовые настройки для наиболее распространенных методологий: DATARUN (основная методология, поддерживаемая Silverrun), Gane/Sarson, Yourdon/DeMarco, Merise, Ward/Mellor, Information Engineering. Для каждого понятия, введенного в проекте имеется возможность добавления собственных описателей. Архитектура Silverrun позволяет наращивать среду разработки по мере необходимости.

Silverrun (структура и функции)

Silverrun имеет модульную структуру и состоит из четырех модулей, каждый из которых является самостоятельным продуктом и может приобретаться и использоваться без связи с остальными модулями.

Модуль построения моделей бизнес-процессов в форме диаграмм потоков данных (BPM - Business Process Modeler) позволяет моделировать функционирование обследуемой организации или создаваемой ИС. В модуле BPM обеспечена возможность работы с моделями большой сложности: автоматическая перенумерация, работа с деревом процессов (включая визуальное перетаскивание ветвей), отсоединение и присоединение частей модели для коллективной разработки. Диаграммы могут изображаться в нескольких предопределенных нотациях, включая Yourdon/DeMarco и Gane/Sarson. Имеется также возможность создавать собственные нотации, в том числе добавлять в число изображаемых на схеме дескрипторов определенные пользователем поля.

Silverrun (структура и функции)

Модуль концептуального моделирования данных (ERX - Entity-Relationship eXpert) обеспечивает построение моделей данных "сущность-связь", не привязанных к конкретной реализации. Этот модуль имеет встроенную экспертную систему, позволяющую создать корректную нормализованную модель данных посредством ответов на содержательные вопросы о взаимосвязи данных. Возможно автоматическое построение модели данных из описаний структур данных. Анализ функциональных зависимостей атрибутов дает возможность проверить соответствие модели требованиям третьей нормальной формы и обеспечить их выполнение. Проверенная модель передается в модуль RDM.

Silverrun (структура и функции)

Модуль реляционного моделирования (RDM - Relational Data Modeler) позволяет создавать детализированные модели "сущность-связь", предназначенные для реализации в реляционной базе данных. В этом модуле документируются все конструкции, связанные с построением базы данных: индексы, триггеры, хранимые процедуры и т.д. Гибкая изменяемая нотация и расширяемость репозитория позволяют работать по любой методологии. Возможность создавать подсхемы соответствует подходу ANSI SPARC к представлению схемы базы данных. На языке подсхем моделируются как узлы распределенной обработки, так и пользовательские представления. Этот модуль обеспечивает проектирование и полное документирование реляционных баз данных.

Silverrun (структура и функции)

Менеджер репозитория рабочей группы (WRM - Workgroup Repository Manager) применяется как словарь данных для хранения общей для всех моделей информации, а также обеспечивает интеграцию модулей Silverrun в единую среду проектирования.

Платой за высокую гибкость и разнообразие изобразительных средств построения моделей является такой недостаток Silverrun, как отсутствие жесткого взаимного контроля между компонентами различных моделей (например, возможности автоматического распространения изменений между DFD различных уровней декомпозиции). Следует, однако, отметить, что этот недостаток может иметь существенное значение только в случае использования каскадной модели ЖЦ ПО.

Silverrun
(Взаимодействие с другими средствами)

Для автоматической генерации схем баз данных у Silverrun существуют мосты к наиболее распространенным СУБД: Oracle, Informix, DB2, Ingres, Progress, SQL Server, SQLBase, Sybase. Для передачи данных в средства разработки приложений имеются мосты к языкам 4GL: JAM, PowerBuilder, SQL Windows, Uniface, NewEra, Delphi. Все мосты позволяют загрузить в Silverrun RDM информацию из каталогов соответствующих СУБД или языков 4GL. Это позволяет документировать, перепроектировать или переносить на новые платформы уже находящиеся в эксплуатации базы данных и прикладные системы. При использовании моста Silverrun расширяет свой внутренний репозиторий специфичными для целевой системы атрибутами. После определения значений этих атрибутов генератор приложений переносит их во внутренний каталог среды разработки или использует при генерации кода на языке SQL. Таким образом можно полностью определить ядро базы данных с использованием всех возможностей конкретной СУБД: триггеров, хранимых процедур, ограничений ссылочной целостности. При создании приложения на языке 4GL данные, перенесенные из репозитория Silverrun, используются либо для автоматической генерации интерфейсных объектов, либо для быстрого их создания вручную.

Silverrun (Взаимодействие с другими средствами)

Для обмена данными с другими средствами автоматизации проектирования, создания специализированных процедур анализа и проверки проектных спецификаций, составления специализированных отчетов в соответствии с различными стандартами в системе Silverrun имеется три способа выдачи проектной информации во внешние файлы:

- **Система отчетов**. Можно, определив содержимое отчета по репозиторию, выдать отчет в текстовый файл. Этот файл можно затем загрузить в текстовый редактор или включить в другой отчет;
- **Система экспорта/импорта**. Для более полного контроля над структурой файлов в системе экспорта/импорта имеется возможность определять не только содержимое экспортного файла, но и разделители записей, полей в записях, маркеры начала и конца текстовых полей. Файлы с указанной структурой можно не только формировать, но и загружать в репозиторий. Это дает возможность обмениваться данными с различными системами: другими CASE-средствами, СУБД, текстовыми редакторами и электронными таблицами;
- **Хранение репозитория во внешних файлах через ODBC-драйверы**. Для доступа к данным репозитория из наиболее распространенных систем управления базами данных обеспечена возможность хранить всю проектную информацию непосредственно в формате этих СУБД.

Silverrun (Групповая работа)

Групповая работа поддерживается в системе Silverrun двумя способами:

- В стандартной однопользовательской версии имеется механизм контролируемого разделения и слияния моделей. Разделив модель на части, можно раздать их нескольким разработчикам. После детальной доработки модели объединяются в единые спецификации;
- Сетевая версия Silverrun позволяет осуществлять одновременную групповую работу с моделями, хранящимися в сетевом репозитории на базе СУБД Oracle, Sybase или Informix. При этом несколько разработчиков могут работать с одной и той же моделью, так как блокировка объектов происходит на уровне отдельных элементов модели.

Silverrun
(Среда функционирования)

Имеются реализации Silverrun трех платформ - MS Windows, Macintosh и OS/2 Presentation Manager - с возможностью обмена проектными данными между ними.

Для функционирования в среде Windows необходимо иметь компьютер с процессором модели не ниже i486 и оперативную память объемом не менее 8 Мб (рекомендуется 16 Мб). На диске полная инсталляция Silverrun занимает 20 Мб.

JAM
*(*JYACC's Application Manager*)*

JAM

Средство разработки приложений JAM (JYACC's Application Manager) - продукт фирмы JYACC (США). В настоящее время поставляется версия JAM 7 и готовится к выходу JAM 8.

Основной чертой JAM является его соответствие методологии RAD, поскольку он позволяет достаточно быстро реализовать цикл разработки приложения, заключающийся в формировании очередной версии прототипа приложения с учетом требований, выявленных на предыдущем шаге, и предъявить его пользователю.

JAM (Структура и функции)

JAM имеет модульную структуру и состоит из следующих компонент:

- Ядро системы;
- JAM/DBi - специализированные модули интерфейса к СУБД (JAM/DBi-Oracle, JAM/DBi-Informix, JAM/DBi-ODBC и т.д.);
- JAM/RW - модуль генератора отчетов;
- JAM/CASEi - специализированные модули интерфейса к CASE-средствам (JAM/CASE-TeamWork, JAM/CASE-Innovator и т.д.);
- JAM/TPi - специализированные модули интерфейса к менеджерам транзакций (например, JAM/TPi-Server TUXEDO и т.д.);
- Jterm - специализированный эмулятор X-терминала.

JAM (Структура и функции)

Ядро системы (собственно, сам JAM) является законченным продуктом и может самостоятельно использоваться для разработки приложений. Все остальные модули являются дополнительными и самостоятельно использоваться не могут.

Ядро системы включает в себя следующие основные компоненты:

- редактор экранов. В состав редактора экранов входят: среда разработки экранов, визуальный репозиторий объектов, собственная СУБД JAM - JDB, менеджер транзакций, отладчик, редактор стилей;
- редактор меню;
- набор вспомогательных утилит;
- средства изготовления промышленной версии приложения.

JAM (Структура и функции)

При использовании JAM разработка внешнего интерфейса приложения представляет собой визуальное проектирование и сводится к созданию экранных форм путем размещения на них интерфейсных конструкций и определению экранных полей ввода/вывода информации. Проектирование интерфейса в JAM осуществляется с помощью редактора экранов. Приложения, разработанные в JAM, имеют многооконный интерфейс. Разработка отдельного экрана заключается в размещении на нем интерфейсных элементов, возможной (но не обязательной) их группировке и конкретизации различных их свойств, включающих визуальные характеристики (позиция, размер, цвет, шрифт и т.п.), поведенческие характеристики (многообразные фильтры, форматы, защита от ввода и т.п.) и ряд свойств, ориентированных на работу с БД.

JAM (Структура и функции)

Редактор меню позволяет разрабатывать и отлаживать системы меню. Реализована возможность построения пиктографических меню (так называемые toolbar). Назначение каждого конкретного меню тому или иному объекту приложения осуществляется в редакторе экранов.

В ядро JAM встроена однопользовательская реляционная СУБД JDB. Основным назначением JDB является прототипирование приложений в тех случаях, когда работа со штатной СУБД невозможна или нецелесообразна. В JDB реализован необходимый минимум возможностей реляционных СУБД за исключением индексов, хранимых процедур, триггеров и представлений (view). С помощью JDB можно построить БД, идентичную целевой БД (с точностью до отсутствующих в JDB возможностей) и разработать значительную часть приложения.

Отладчик позволяет проводить комплексную отладку разрабатываемого приложения. Осуществляется трассировка всех событий, возникающих в процессе исполнения приложения.

JAM (Структура и функции)

Утилиты JAM включают три группы:

- конверторы файлов экранов JAM в текстовые. JAM сохраняет экраны в виде двоичных файлов собственного формата. В ряде случаев (например для изготовления программной документации проекта) необходимо текстовое описание экранов;
- конфигурирование устройств ввода/вывода. JAM и приложения, построенные с его помощью, не работают непосредственно с устройствами ввода/вывода. Вместо этого JAM обращается к логическим устройствам ввода/вывода (клавиатура, терминал, отчет). Отображение логических устройств в физические осуществляется с помощью средств конфигурирования;
- обслуживание библиотек экранов (традиционные операции с библиотеками).

JAM (Структура и функции)

Одним из дополнительных модулей JAM является генератор отчетов. Компоновка отчета осуществляется в редакторе экранов JAM. Описание работы отчета осуществляется с помощью специального языка. Генератор отчетов позволяет определить данные, выводимые в отчет, группировку выводимой информации, форматирование вывода и др.

Приложения, разработанные с использованием JAM, не требуют так называемых исполнительных (run-time) систем и могут быть изготовлены в виде исполняемых модулей. Для этого разработчик должен иметь компилятор C и редактор связей. Для изготовления промышленной версии в состав JAM входит файл сборки (makefile), исходные тексты (на языке C) ряда модулей приложения и необходимые библиотеки.

JAM (Структура и функции)

JAM содержит встроенный язык программирования JPL (JAM Procedural Language), с помощью которого в случае необходимости можно написать модули, реализующие специфические действия. Данный язык является интерпретируемым, что упрощает отладку. Существует возможность обмена информацией между средой визуально построенного приложения и такими модулями. Кроме того, в JAM реализована возможность подключения внешних модулей, написанных на каком-либо языке, совместимым по вызовам функций с языком C.

JAM (Структура и функции)

С точки зрения реализации логики приложения JAM является событийно-ориентированной системой. В JAM определен набор событий, включающий открытие и закрытие окон, нажатие клавиши клавиатуры, срабатывание системного таймера, получение и передача управления каждым элементом экрана. Разработчик реализует логику приложения путем определения обработчика каждого события. Например, обработчик события "нажатие кнопки на экране" (мышью или с помощью клавиатуры) может открыть следующее экранное окно. Обработчиками событий в JAM могут быть как встроенные функции JAM, так и функции, написанные разработчиком на C или JPL. Набор встроенных функций включает в себя более 200 функций различного назначения. Встроенные функции доступны для вызовов из функций, написанных как на JPL, так и на C.

JAM (Структура и функции)

Промышленная версия приложения, разработанного с помощью JAM, включает в себя следующие компоненты:

- исполняемый модуль интерпретатора приложения. В этот модуль могут быть встроены функции, написанные разработчиками на языках 3-го поколения;
- экраны, составляющие само приложение (могут поставляться в виде отдельных файлов, в составе библиотек экранов или же быть встроены в тело интерпретатора);
- внешние JPL-модули. Могут поставляться в виде текстовых файлов или в прекомпилированном виде, причем прекомпилированные внешние JPL-модули могут быть как в виде отдельных файлов, так и в составе библиотек экранов;
- файлы конфигурации приложения - файлы конфигурации клавиатуры и терминала, файл системных сообщений, файл общей конфигурации.

JAM
(Взаимодействие с другими средствами)

Непосредственное взаимодействие с СУБД реализуют модули JAM/DBi (Data Base interface). Способы реализации взаимодействия в JAM разделяются на два класса: ручные и автоматические. При ручном способе разработчик приложения самостоятельно пишет запросы на SQL, в которых как источниками, так и адресатами приема результатов выполнения запроса могут быть как интерфейсные элементы визуально спроектированного внешнего уровня, так и внутренние, невидимые для конечного пользователя переменные. Автоматический режим, реализуемый менеджером транзакций JAM, осуществим для типовых и наиболее распространенных видов операций с БД, так называемых QBE (Query By Example - запросы по образцу), с учетом достаточно сложных взаимосвязей между таблицами БД и автоматическим управлением атрибутами экранных полей ввода/вывода в зависимости от вида транзакции (чтение, запись и т.д.), в которой участвует сгенерированный запрос.

JAM
(Взаимодействие с другими средствами)

JAM позволяет строить приложения для работы более чем с 20 СУБД: ORACLE, Informix, Sybase, Ingres, InterBase, NetWare SQL Server, Rdb, DB2, ODBC-совместимые СУБД и др.

Отличительной чертой JAM является высокий уровень переносимости приложений между различными платформами (MS DOS/MS Windows, SunOS, Solaris (i80x86, SPARC), HP-UX, AIX, VMS/Open VMS и др.). Может потребоваться лишь "перерисовать" статические текстовые поля на экранах с русским текстом при переносе между средами DOS-Windows-UNIX. Кроме того, переносимость облегчается тем, что в JAM приложения разрабатываются для виртуальных устройств ввода/вывода, а не для физических. Таким образом при переносе приложения с платформы на платформу, как правило, требуется лишь определить соответствие между физическими устройствами ввода/вывода и их логическими представлениями для приложения.

JAM
(Взаимодействие с другими средствами)

Использование SQL в качестве средства взаимодействия с СУБД также создает предпосылки для обеспечения переносимости между СУБД. При условии переноса структуры самой БД в ряде случаев приложения могут не требовать никакой модификации, за исключением инициализации сеанса работы. Такая ситуация может сложиться в том случае, если в приложении не использовались специфические для той или иной СУБД расширения SQL.

При росте нагрузки на систему и сложности решаемых задач (распределенность и гетерогенность используемых ресурсов, количество одновременно подключенных пользователей, сложность логики приложения) применяется трехзвенная модель архитектуры "клиент-сервер" с использованием менеджеров транзакций. Компоненты JAM/TPi-Client и JAM/TPi-Server позволяют достаточно просто перейти на трехзвенную модель. При этом ключевую роль играет модуль JAM/TPi-Server, так как основная трудность внедрения трехзвенной модели заключается в реализации логики приложения в сервисах менеджеров транзакций.

JAM
(Взаимодействие с другими средствами)

Интерфейс JAM/CASE подобен интерфейсу к СУБД и позволяет осуществить обмен информацией между репозиторием объектов JAM и репозиторием CASE-средства аналогично тому, как структура БД импортируется в репозиторий JAM непосредственно из БД. Отличие заключается в том, что в случае интерфейса к CASE этот обмен является двунаправленным. Кроме модулей JAM/CASEi, существует также модуль JAM/CASEi Developer's Kit. С помощью этого модуля можно самостоятельно разработать интерфейс (т.е. специализированный модуль JAM/CASEi) для конкретного CASE-средства, если готового модуля JAM/CASEi для него не существует.

JAM
(Взаимодействие с другими средствами)

Мост (интерфейс) Silverrun-RDM <-> JAM реализует взаимодействие между CASE-средством Silverrun и JAM (перенос схемы базы данных и экранных форм приложения между CASE-средством Silverrun-RDM и JAM версии 7.0). Данный программный продукт имеет 2 режима работы:

- прямой режим (Silverrun-RDM->JAM) предназначен для создания объектов CASE-словаря и элементов репозитория JAM на основе представления схем в Silverrun-RDM. В этом режиме мост позволяет, исходя из представления моделей данных интерфейса в Silverrun-RDM, производить генерацию экранов и элементов репозитория JAM. Мост преобразует таблицы и отношения реляционных схем RDM в последовательность объектов JAM соответствующих типов. Методика построения моделей данных интерфейса в Silverrun-RDM предполагает применение механизма подсхем для прототипирования экранов приложения. По описанию каждой из подсхем RDM мост генерирует экранную форму JAM;
- обратный режим (JAM->Silverrun-RDM) предназначен для переноса модификаций объектов CASE-словаря в реляционную модель Silverrun-RDM.

Режим реинжиниринга позволяет переносить модификации всех свойств экранов JAM, импортированных ранее из RDM, в схему Silverrun. На этом этапе для контроля целостности базы данных не допускаются изменения схемы в виде добавления или удаления таблиц и полей

JAM
(Групповая работа)

Ядро JAM имеет встроенный интерфейс к средствам конфигурационного управления (PVCS на платформе Windows и SCCS на платформе UNIX). Под управлением этих систем передаются библиотеки экранов и/или репозитории. При отсутствии таких систем JAM самостоятельно реализует часть функций поддержки групповой разработки.

Использование PVCS является более предпочтительным по сравнению с SCCS, так как позволяет организовать единый архив модулей проекта для всех платформ. Так как JAM на платформе UNIX не имеет прямого интерфейса к архивам PVCS, то выборка модулей из архива и возврат их в архив производятся с использованием PVCS Version Manager. На платформе MS-Windows JAM имеет встроенный интерфейс к PVCS и действия по выборке/возврату производятся непосредственно из среды JAM.

JAM (Среда функционирования)

JAM, как среда разработки, и приложения, построенные с его использованием, не являются ресурсоемкими системами. Например, на платформе MS-Windows достаточно иметь 8MB оперативной памяти и 50 MB дискового пространства для среды разработки. На UNIX-платформах требования к аппаратуре определяются самой операционной системой.

Vantage Team Builder (Westmount I-CASE)

Vantage Team Builder (Westmount I-CASE)

Vantage Team Builder представляет собой интегрированный программный продукт, ориентированный на реализацию каскадной модели ЖЦ ПО и поддержку полного ЖЦ ПО.

Vantage Team Builder (Структура и функции)

Vantage Team Builder обеспечивает выполнение следующих функций:

- проектирование диаграмм потоков данных, "сущность-связь", структур данных, структурных схем программ и последовательностей экранных форм;
- проектирование диаграмм архитектуры системы - SAD (проектирование состава и связи вычислительных средств, распределения задач системы между вычислительными средствами, моделирование отношений типа "клиент-сервер", анализ использования менеджеров транзакций и особенностей функционирования систем в реальном времени);
- генерация кода программ на языке 4GL целевой СУБД с полным обеспечением программной среды и генерация SQL-кода для создания таблиц БД, индексов, ограничений целостности и хранимых процедур;
- программирование на языке C со встроенным SQL;
- управление версиями и конфигурацией проекта;
- многопользовательский доступ к репозиторию проекта;
- генерация проектной документации по стандартным и индивидуальным шаблонам;
- экспорт и импорт данных проекта в формате CDIF (CASE Data Interchange Format).

Vantage Team Builder (Структура и функции)

Vantage Team Builder поставляется в различных конфигурациях в зависимости от используемых СУБД (ORACLE, Informix, Sybase или Ingres) или средств разработки приложений (Uniface). Конфигурация Vantage Team Builder for Uniface отличается от остальных некоторой степенью ориентации на спиральную модель ЖЦ ПО за счет возможностей быстрого прототипирования, предоставляемых Uniface. Для описания проекта ИС используется достаточно большой набор диаграмм, конкретные варианты которого для наиболее распространенных конфигураций приведены ниже в таблице.

Vantage Team Builder (Структура и функции)

Тип диаграммы	Обозначение	Vantage Team Builder for ORACLE	Vantage Team Builder for Informix	Vantage Team Builder for Uniface
Сущность-связь	ERD	+	+	+
Потоков данных	DFD	+	+	+
Структур данных	DSD	+	+	+
Архитектуры системы	SAD	+	+	+
Потоков управления	CSD	+	+	+
Типов данных	DTD	+	+	+
Структуры меню	MSD	+		
Последовательности блоков	BSD	+		
Последовательности форм	FSD		+	+
Содержимого форм	FCD		+	+
Переходов состояний	STD	+	+	+
Структурных схем	SCD	+	+	+

Vantage Team Builder (Структура и функции)

При построении всех типов диаграмм обеспечивается контроль соответствия моделей синтаксису используемых методов, а также контроль соответствия одноименных элементов и их типов для различных типов диаграмм.

При построении DFD обеспечивается контроль соответствия диаграмм различных уровней декомпозиции. Контроль за правильностью верхнего уровня DFD осуществляется с помощью матрицы списков событий (ELM). Для контроля за декомпозицией составных потоков данных используется несколько вариантов их описания: в виде диаграмм структур данных (DSD) или в нотации БНФ (форма Бэкуса-Наура).

Для построения SAD используется расширенная нотация DFD, дающая возможность вводить понятия процессоров, задач и периферийных устройств, что обеспечивает наглядность проектных решений.

Vantage Team Builder (Структура и функции)

При построении модели данных в виде ERD выполняется ее нормализация и вводится определение физических имен элементов данных и таблиц, которые будут использоваться в процессе генерации физической схемы данных конкретной СУБД. Обеспечивается возможность определения альтернативных ключей сущностей и полей, составляющих дополнительные точки входа в таблицу (поля индексов), и мощности отношений между сущностями.

Наличие универсальной системы генерации кода, основанной на специфицированных средствах доступа к репозиторию проекта, позволяет поддерживать высокий уровень исполнения проектной дисциплины разработчиками: жесткий порядок формирования моделей; жесткая структура и содержимое документации; автоматическая генерация исходных кодов программ и т.д. - все это обеспечивает повышение качества и надежности разрабатываемых ИС.

Для подготовки проектной документации могут использоваться издательские системы FrameMaker, Interleaf или Word Perfect. Структура и состав проектной документации могут быть настроены в соответствии с заданными стандартами. Настройка выполняется без изменения проектных решений.

Vantage Team Builder (Структура и функции)

При разработке достаточно крупной ИС вся система в целом соответствует одному проекту как категории Vantage Team Builder. Проект может быть декомпозирован на ряд систем, каждая из которых соответствует некоторой относительно автономной подсистеме ИС и разрабатывается независимо от других. В дальнейшем системы проекта могут быть интегрированы.

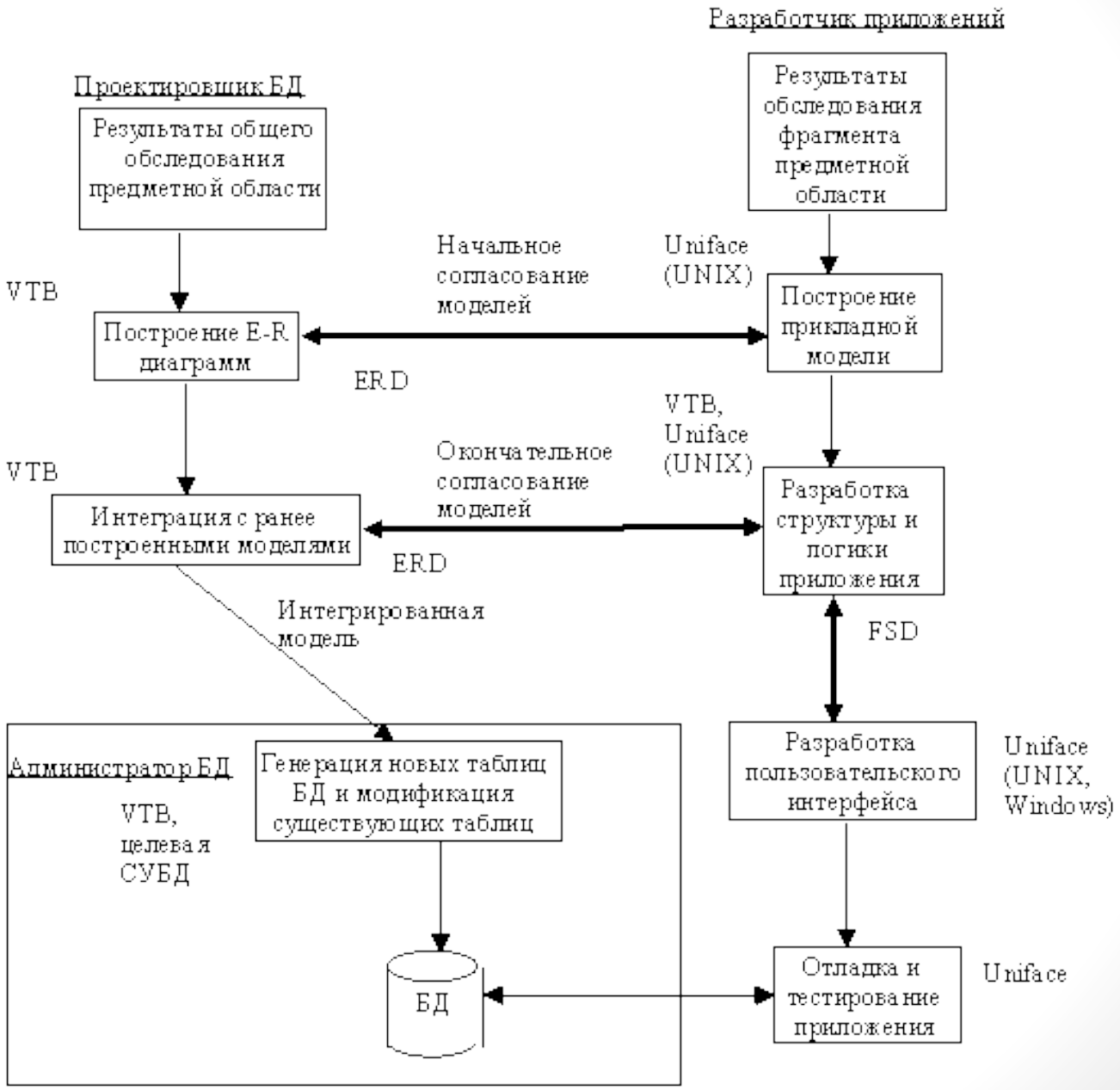
Процесс проектирования ИС с использованием Vantage Team Builder реализуется в виде 4-х последовательных фаз (стадий) - анализа, архитектуры, проектирования и реализации, при этом законченные результаты каждой стадии полностью или частично переносятся (импортируются) в следующую фазу. Все диаграммы, кроме ERD, преобразуются в другой тип или изменяют вид в соответствии с особенностями текущей фазы. Так, DFD преобразуются в фазе архитектуры в SAD, DSD - в DTD. После завершения импорта логическая связь с предыдущей фазой разрывается, т.е. в диаграммы могут вноситься все необходимые изменения.

Vantage Team Builder (Взаимодействие с другими средствами)

Конфигурация Vantage Team Builder for Uniface обеспечивает совместное использование двух систем в рамках единой технологической среды проектирования, при этом схемы БД (SQL-модели) переносятся в репозиторий Uniface, и, наоборот, прикладные модели, сформированные средствами Uniface, могут быть перенесены в репозиторий Vantage Team Builder. Возможные рассогласования между репозиториями двух систем устраняются с помощью специальной утилиты. Разработка экранных форм в среде Uniface выполняется на базе диаграмм последовательностей форм (FSD) после импорта SQL-модели. Технология разработки ИС на базе данной конфигурации показана на рисунке.

Структура репозитория (хранящегося непосредственно в целевой СУБД) и интерфейсы Vantage Team Builder являются открытыми, что в принципе позволяет интеграцию с любыми другими средствами.

Vantage Team Builder
(Взаимодействие с другими средствами)



Vantage Team Builder
(Среда функционирования)

Vantage Team Builder функционирует на всех основных UNIX-платформах (Solaris, SCO UNIX, AIX, HP-UX) и VMS.

Vantage Team Builder можно использовать в конфигурации "клиент-сервер", при этом база проектных данных может располагаться на сервере, а рабочие места разработчиков могут быть клиентами.

Uniface

Uniface

Uniface 6.1 - продукт фирмы Compuware (США) - представляет собой среду разработки крупномасштабных приложений в архитектуре "клиент-сервер" и имеет следующую компонентную архитектуру:

- Application Objects Repository (репозиторий объектов приложений) содержит метаданные, автоматически используемые всеми остальными компонентами на протяжении жизненного цикла ИС (прикладные модели, описания данных, бизнес-правил, экранных форм, глобальных объектов и шаблонов). Репозиторий может храниться в любой из баз данных, поддерживаемых Uniface;
- Application Model Manager поддерживает прикладные модели (E-R модели), каждая из которых представляет собой подмножество общей схемы БД с точки зрения данного приложения, и включает соответствующий графический редактор;
- Rapid Application Builder - средство быстрого создания экранных форм и отчетов на базе объектов прикладной модели. Оно включает графический редактор форм, средства прототипирования, отладки, тестирования и документирования. Реализован интерфейс с разнообразными типами оконных элементов управления (Open Widget Interface) для существующих графических интерфейсов - MS Windows (включая VBX), Motif, OS/2. Универсальный интерфейс представления (Universal Presentation Interface) позволяет использовать одну и ту же версию приложения в среде различных графических интерфейсов без изменения программного кода;

Uniface

- Developer Services (службы разработчика) - используются для поддержки крупных проектов и реализуют контроль версий (Uniface Version Control System), права доступа (разграничение полномочий), глобальные модификации и т.д. Это обеспечивает разработчиков средствами параллельного проектирования, входного и выходного контроля, поиска, просмотра, поддержки и выдачи отчетов по данным системы контроля версий;
- Deployment Manager (управление распространением приложений) - средства, позволяющие подготовить созданное приложение для распространения, устанавливать и сопровождать его (при этом платформа пользователя может отличаться от платформы разработчика). В их состав входят сетевые драйверы и драйверы СУБД, сервер приложений (полисервер), средства распространения приложений и управления базами данных. Uniface поддерживает интерфейс практически со всеми известными программно-аппаратными платформами, СУБД, CASE-средствами, сетевыми протоколами и менеджерами транзакций;
- Personal Series (персональные средства) - используются для создания сложных запросов и отчетов в графической форме (Personal Query и Personal Access - PQ/PA), а также для переноса данных в такие системы, как WinWord и Excel;
- Distributed Computing Manager - средство интеграции с менеджерами транзакций Tuxedo, Encina, CICS, OSF DCE.

Uniface

Объявленная в конце 1996 г. версия Uniface 7 полностью поддерживает распределенную модель вычислений и трехзвенную архитектуру "клиент-сервер" (с возможностью изменения схемы декомпозиции приложений на этапе исполнения). Приложения, создаваемые с помощью Uniface 7, могут исполняться в гетерогенных операционных средах, использующих различные сетевые протоколы, одновременно на нескольких разнородных платформах (в том числе и в Internet).

Uniface

В состав компонент Uniface 7 входят:

- Uniface Application Server - сервер приложений для распределенных систем;
- WebEnabler - серверное ПО для эксплуатации приложений в Internet и Intranet;
- Name Server - серверное ПО, обеспечивающее использование распределенных прикладных ресурсов;
- PolyServer - средство доступа к данным и интеграции различных систем.
- В список поддерживаемых СУБД входят DB2, VSAM и IMS; PolyServer обеспечивает также взаимодействие с ОС MVS.
- Среда функционирования Uniface - все основные UNIX - платформы и MS Windows.

Designer/2000 + Developer/2000

Designer/2000 + Developer/2000

CASE-средство Designer/2000 2.0 фирмы ORACLE является интегрированным CASE-средством, обеспечивающим в совокупности со средствами разработки приложений Developer/2000 поддержку полного ЖЦ ПО для систем, использующих СУБД ORACLE.

Designer/2000 + Developer/2000 (*Структура и функции*)

Designer/2000 представляет собой семейство методологий и поддерживающих их программных продуктов. Базовая методология Designer/2000 (CASE*Method) - структурная методология проектирования систем, полностью охватывающая все этапы жизненного цикла ИС. В соответствии с этой методологией на этапе планирования определяются цели создания системы, приоритеты и ограничения, разрабатывается системная архитектура и план разработки ИС. В процессе анализа строятся модель информационных потребностей (диаграмма "сущность-связь"), диаграмма функциональной иерархии (на основе функциональной декомпозиции ИС), матрица перекрестных ссылок и диаграмма потоков данных.

На этапе проектирования разрабатывается подробная архитектура ИС, проектируется схема реляционной БД и программные модули, устанавливаются перекрестные ссылки между компонентами ИС для анализа их взаимного влияния и контроля за изменениями.

На этапе реализации создается БД, строятся прикладные системы, производится их тестирование, проверка качества и соответствия требованиям пользователей. Создается системная документация, материалы для обучения и руководства пользователей. На этапах эксплуатации и сопровождения анализируются производительность и целостность системы, выполняется поддержка и, при необходимости, модификация ИС;

Designer/2000 + Developer/2000 (*Структура и функции*)

Designer/2000 обеспечивает графический интерфейс при разработке различных моделей (диаграмм) предметной области. В процессе построения моделей информация о них заносится в репозиторий. В состав Designer/2000 входят следующие компоненты:

- Repository Administrator - средства управления репозиторием (создание и удаление приложений, управление доступом к данным со стороны различных пользователей, экспорт и импорт данных);
- Repository Object Navigator - средства доступа к репозиторию, обеспечивающие многооконный объектно-ориентированный интерфейс доступа ко всем элементам репозитория;
- Process Modeller - средство анализа и моделирования деловой деятельности, основывающееся на концепциях реинжиниринга бизнес-процессов (BPR - Business Process Reengineering) и глобальной системы управления качеством (TQM - Total Quality Management);
- Systems Modeller - набор средств построения функциональных и информационных моделей проектируемой ИС, включающий средства для построения диаграмм "сущность-связь" (Entity-Relationship Diagrammer), диаграмм функциональных иерархий (Function Hierarchy Diagrammer), диаграмм потоков данных (Data Flow Diagrammer) и средство анализа и модификации связей объектов репозитория различных типов (Matrix Diagrammer);

Designer/2000 + Developer/2000 (*Структура и функции*)

- Systems Designer - набор средств проектирования ИС, включающий средство построения структуры реляционной базы данных (Data Diagrammer), а также средства построения диаграмм, отображающих взаимодействие с данными, иерархию, структуру и логику приложений, реализуемую хранимыми процедурами на языке PL/SQL (Module Data Diagrammer, Module Structure Diagrammer и Module Logic Navigator);
- Server Generator - генератор описаний объектов БД ORACLE (таблиц, индексов, ключей, последовательностей и т.д.). Помимо продуктов ORACLE, генерация и реинжиниринг БД может выполняться для СУБД Informix, DB/2, Microsoft SQL Server, Sybase, а также для стандарта ANSI SQL DDL и баз данных, доступ к которым реализуется посредством ODBC;
- Forms Generator (генератор приложений для ORACLE Forms). Генерируемые приложения включают в себя различные экранные формы, средства контроля данных, проверки ограничений целостности и автоматические подсказки. Дальнейшая работа с приложением выполняется в среде Developer/2000;
- Repository Reports - генератор стандартных отчетов, интегрированный с ORACLE Reports и позволяющий русифицировать отчеты, а также изменять структурное

Designer/2000 + Developer/2000 (*Структура и функции*)

Репозиторий Designer/2000 представляет собой хранилище всех проектных данных и может работать в многопользовательском режиме, обеспечивая параллельное обновление информации несколькими разработчиками. В процессе проектирования автоматически поддерживаются перекрестные ссылки между объектами словаря и могут генерироваться более 70 стандартных отчетов о моделируемой предметной области. Физическая среда хранения репозитория - база данных ORACLE.

Генерация приложений, помимо продуктов ORACLE, выполняется также для Visual Basic.

Designer/2000 + Developer/2000 (*Взаимодействие с другими средствами*)

Designer/2000 можно интегрировать с другими средствами, используя открытый интерфейс приложений API (Application Programming Interface). Кроме того, можно использовать средство ORACLE CASE Exchange для экспорта/импорта объектов репозитория с целью обмена информацией с другими CASE-средствами.

Developer/2000 обеспечивает разработку переносимых приложений, работающих в графической среде Windows, Macintosh или Motif. В среде Windows интеграция приложений Developer/2000 с другими средствами реализуется через механизм OLE и управляющие элементы VBX. Взаимодействие приложений с другими СУБД (DB/2, DB2/400, Rdb) реализуется с помощью средств ORACLE Client Adapter для ODBC, ORACLE Open Gateway и API.

Designer/2000 + Developer/2000 (*Среда функционирования*)

Среда функционирования Designer/2000 и Developer/2000
- Windows 3.x, Windows 95, Windows NT.

Локальные средства
(ERwin, BPwin, S-Designor, CASE.Аналитик)

Локальные средства (ERwin, BPwin, S-Designor, CASE.Аналитик)

ERwin - средство концептуального моделирования БД [24], использующее методологию IDEF1X (см. подраздел 2.5). ERwin реализует проектирование схемы БД, генерацию ее описания на языке целевой СУБД (ORACLE, Informix, Ingres, Sybase, DB/2, Microsoft SQL Server, Progress и др.) и реинжиниринг существующей БД. ERwin выпускается в нескольких различных конфигурациях, ориентированных на наиболее распространенные средства разработки приложений 4GL. Версия ERwin/OPEN полностью совместима со средствами разработки приложений PowerBuilder и SQLWindows и позволяет экспортировать описание спроектированной БД непосредственно в репозитории данных средств.

Для ряда средств разработки приложений (PowerBuilder, SQLWindows, Delphi, Visual Basic) выполняется генерация форм и прототипов приложений.

Сетевая версия Erwin ModelMart обеспечивает согласованное проектирование БД и приложений в рамках рабочей группы.

BPwin - средство функционального моделирования, реализующее методологию IDEF0.

Возможные конфигурации и ориентировочная стоимость средств (без

Локальные средства (ERwin, BPwin, S-Designor, CASE.Аналитик)

Конфигурация	Стоимость, $
ERwin/ERX	3,295
Bpwin	2,495
ERwin/ERX for PowerBuilder, Visual Basic, Progress	3,495
ERwin/ERX for Delphi	4,295
ERwin/Desktop for PowerBuilder, Visual Basic	495
ERwin/ERX for SQLWindows / Designer/2000/ Solaris	3,495 / 5,795 / 6,995
ModelMart 5 / 10 user	11,995 / 19,995
Erwin/OPEN for ModelMart	3,995

Локальные средства (ERwin, BPwin, S-Designor, CASE.Аналитик)

S-Designor 4.2 представляет собой CASE-средство для проектирования реляционных баз данных. По своим функциональным возможностям и стоимости он близок к CASE-средству ERwin, отличаясь внешне используемой на диаграммах нотацией. S-Designor реализует стандартную методологию моделирования данных и генерирует описание БД для таких СУБД, как ORACLE, Informix, Ingres, Sybase, DB/2, Microsoft SQL Server и др. Для существующих систем выполняется реинжиниринг БД.

S-Designor совместим с рядом средств разработки приложений (PowerBuilder, Uniface, TeamWindows и др.) и позволяет экспортировать описание БД в репозитории данных средств. Для PowerBuilder выполняется также прямая генерация шаблонов приложений.

Локальные средства (ERwin, BPwin, S-Designor, CASE.Аналитик)

CASE.Аналитик 1.1 является практически единственным в настоящее время конкурентоспособным отечественным CASE-средством функционального моделирования и реализует построение диаграмм потоков данных. Его основные функции:

- построение и редактирование DFD;
- анализ диаграмм и проектных спецификаций на полноту и непротиворечивость;
- получение разнообразных отчетов по проекту;
- генерация макетов документов в соответствии с требованиями ГОСТ 19.XXX и 34.XXX.

Среда функционирования: процессор - 386 и выше, основная память - 4 Мб, дисковая память - 5 Мб, MS Windows 3.x или Windows 95.

Ориентировочная стоимость:

- однопользовательская версия - 605 $;
- многопользовательская версия (одно рабочее место) - 535 $.

База данных проекта реализована в формате СУБД Paradox и является открытой для доступа.

Локальные средства (ERwin, BPwin, S-Designor, CASE.Аналитик)

С помощью отдельного программного продукта (Catherine) выполняется обмен данными с CASE-средством ERwin. При этом из проекта, выполненного в CASE.Аналитике, экспортируется описание структур данных и накопителей данных, которое по определенным правилам формирует описание сущностей и их атрибутов.

Объектно-ориентированные CASE-средства (Rational Rose)

Объектно-ориентированные CASE-средства (Rational Rose)

Rational Rose - CASE-средство фирмы Rational Software Corporation (США) - предназначено для автоматизации этапов анализа и проектирования ПО, а также для генерации кодов на различных языках и выпуска проектной документации. Rational Rose использует синтез-методологию объектно-ориентированного анализа и проектирования, основанную на подходах трех ведущих специалистов в данной области: Буча, Рамбо и Джекобсона. Разработанная ими универсальная нотация для моделирования объектов (UML - Unified Modeling Language) претендует на роль стандарта в области объектно-ориентированного анализа и проектирования. Конкретный вариант Rational Rose определяется языком, на котором генерируются коды программ (C++, Smalltalk, PowerBuilder, Ada, SQLWindows и ObjectPro). Основной вариант - Rational Rose/C++ - позволяет разрабатывать проектную документацию в виде диаграмм и спецификаций, а также генерировать программные коды на C++. Кроме того, Rational Rose содержит средства реинжиниринга программ, обеспечивающие повторное использование программных компонент в новых проектах.

Объектно-ориентированные CASE-средства (Rational Rose) (*Структура и функции)*

В основе работы Rational Rose лежит построение различного рода диаграмм и спецификаций, определяющих логическую и физическую структуры модели, ее статические и динамические аспекты. В их число входят диаграммы классов, состояний, сценариев, модулей, процессов .

В составе Rational Rose можно выделить 6 основных структурных компонент: репозиторий, графический интерфейс пользователя, средства просмотра проекта (browser), средства контроля проекта, средства сбора статистики и генератор документов. К ним добавляются генератор кодов (индивидуальный для каждого языка) и анализатор для C++, обеспечивающий реинжиниринг - восстановление модели проекта по исходным текстам программ.

Репозиторий представляет собой объектно-ориентированную базу данных. Средства просмотра обеспечивают "навигацию" по проекту, в том числе, перемещение по иерархиям классов и подсистем, переключение от одного вида диаграмм к другому и т. д. Средства контроля и сбора статистики дают возможность находить и устранять ошибки по мере развития проекта, а не после завершения его описания. Генератор отчетов формирует тексты выходных документов на основе содержащейся в репозитории информации.

Объектно-ориентированные CASE-средства (Rational Rose) (*Структура и функции)*

Средства автоматической генерации кодов программ на языке C++, используя информацию, содержащуюся в логической и физической моделях проекта, формируют файлы заголовков и файлы описаний классов и объектов. Создаваемый таким образом скелет программы может быть уточнен путем прямого программирования на языке C++. Анализатор кодов C++ реализован в виде отдельного программного модуля. Его назначение состоит в том, чтобы создавать модули проектов в форме Rational Rose на основе информации, содержащейся в определяемых пользователем исходных текстах на C++. В процессе работы анализатор осуществляет контроль правильности исходных текстов и диагностику ошибок. Модель, полученная в результате его работы, может целиком или фрагментарно использоваться в различных проектах. Анализатор обладает широкими возможностями настройки по входу и выходу. Например, можно определить типы исходных файлов, базовый компилятор, задать, какая информация должна быть включена в формируемую модель и какие элементы выходной модели следует выводить на экран. Таким образом, Rational Rose/C++ обеспечивает возможность повторного использования

Объектно-ориентированные CASE-средства (Rational Rose) (*Структура и функции)*

В результате разработки проекта с помощью CASE-средства Rational Rose формируются следующие документы:

- диаграммы классов;
- диаграммы состояний;
- диаграммы сценариев;
- диаграммы модулей;
- диаграммы процессов;
- спецификации классов, объектов, атрибутов и операций
- заготовки текстов программ;
- модель разрабатываемой программной системы.

Объектно-ориентированные CASE-средства (Rational Rose) (*Структура и функции)*

Последний из перечисленных документов является текстовым файлом, содержащим всю необходимую информацию о проекте (в том числе необходимую для получения всех диаграмм и спецификаций).

Тексты программ являются заготовками для последующей работы программистов. Они формируются в рабочем каталоге в виде файлов типов .h (заголовки, содержащие описания классов) и .cpp (заготовки программ для методов). Система включает в программные файлы собственные комментарии, которые начинаются с последовательности символов //##. Состав информации, включаемой в программные файлы, определяется либо по умолчанию, либо по усмотрению пользователя. В дальнейшем эти исходные тексты развиваются программистами в полноценные программы.

Объектно-ориентированные CASE-средства (Rational Rose) (*Взаимодействие с другими средствами и организация групповой работы)*

Rational Rose интегрируется со средством PVCS для организации групповой работы и управления проектом и со средством SoDA - для документирования проектов. Интеграция Rational Rose и SoDA обеспечивается средствами SoDA.

Для организации групповой работы в Rational Rose возможно разбиение модели на управляемые подмодели. Каждая из них независимо сохраняется на диске или загружается в модель. В качестве подмодели может выступать категория классов или подсистема.

Для управляемой подмодели предусмотрены операции:

- загрузка подмодели в память;
- выгрузка подмодели из памяти;
- сохранение подмодели на диске в виде отдельного файла;
- установка защиты от модификации;
- замена подмодели в памяти на новую.

Объектно-ориентированные CASE-средства (Rational Rose) (*Взаимодействие с другими средствами и организация групповой работы)*

Наиболее эффективно групповая работа организуется при интеграции Rational Rose со специальными средствами управления конфигурацией и контроля версий (PVCS). В этом случае защита от модификации устанавливается на все управляемые подмодели, кроме тех, которые выделены конкретному разработчику. В этом случае признак защиты от записи устанавливается для файлов, которые содержат подмодели, поэтому при считывании "чужих" подмоделей защита их от модификации сохраняется и случайные воздействия окажутся невозможными.

Объектно-ориентированные CASE-средства (Rational Rose) (*Среда функционирования)*

Rational Rose функционирует на различных платформах: IBM PC (в среде Windows), Sun SPARC stations (UNIX, Solaris, SunOS), Hewlett-Packard (HP UX), IBM RS/6000 (AIX).

Для работы системы необходимо выполнение следующих требований:

- Платформа Windows - процессор 80386SX или выше (рекомендуется 80486), память8Мб (рекомендуется 12Мб), пространство на диске 8Мб + 1-3Мб для одной модели.
- Платформа UNIX - память 32+(16*число пользователей)Мб, пространство на диске 30Мб + 20 при инсталляции + 1-3Мб для одной модели.

Совместимость по версиям обеспечивается на уровне моделей.

Вспомогательные средства поддержки жизненного цикла ПО

Вспомогательные средства поддержки жизненного цикла ПО

- *Средства конфигурационного управления*
- *Средства документирования*
- *Средства тестирования*

Средства конфигурационного управления

Цель конфигурационного управления (КУ) - обеспечить управляемость и контролируемость процессов разработки и сопровождения ПО. Для этого необходима точная и достоверная информация о состоянии ПО и его компонент в каждый момент времени, а также о всех предполагаемых и выполненных изменениях.

Для решения задач КУ применяются методы и средства обеспечивающие идентификацию состояния компонент, учет номенклатуры всех компонент и модификаций системы в целом, контроль за вносимыми изменениями в компоненты, структуру системы и ее функции, а также координированное управление развитием функций и улучшением характеристик системы.

Наиболее распространенным средством КУ является PVCS фирмы Intersolv (США), включающее ряд самостоятельных продуктов: PVCS Version Manager, PVCS Tracker, PVCS Configuration Builder и PVCS Notify.

Средства конфигурационного управления

PVCS Version Manager предназначен для управления всеми компонентами проекта и ведения планомерной многоверсионной и многоплатформенной разработки силами команды разработчиков в условиях одной или нескольких локальных сетей. Понятие "проект" трактуется как совокупность файлов. В процессе работы над проектом промежуточное состояние файлов периодически сохраняется в архиве проекта, ведутся записи о времени сохранения, соответствии друг другу нескольких вариантов разных файлов проекта. Кроме этого, фиксируются имена разработчиков, ответственных за тот или иной файл, состав файлов промежуточных версий проекта и др. Это позволяет вернуться при необходимости к какому-либо из предыдущих состояний файла (например, при обнаружении ошибки, которую в данный момент трудно исправить).

PVCS Version Manager предназначен для использования в рабочих группах. Система блокировок, реализованная в PVCS Version Manager позволяет предотвратить одновременное внесение изменений в один и тот же файл. В то же время, PVCS Version Manager позволяет разработчикам работать с собственными версиями общего файла с полуавтоматическим разрешением конфликтов между ними.

Средства конфигурационного управления

Доступ к архивам PVCS Version Manager возможен не только через сам Version Manager, но и из более чем 50 инструментальных средств, в том числе MS Visual C и MS Visual Basic, Uniface, PowerBuilder, SQL Windows, JAM, Delphi, Paradox и др.

Результатом работы PVCS Version Manager является созданный средствами файловой системы репозиторий, хранящий в компактной форме все рабочие версии программного продукта вместе с необходимыми комментариями и метками.

PVCS Version Manager функционирует в среде MS Windows, Windows 95, Windows NT, OS/2, SunOS, Solaris, HP-UX, AIX и SCO UNIX и может исполняться на любом персональном компьютере с процессором 80386 или выше, рабочих станциях Sun, HP и IBM (RS-6000).

Средства конфигурационного управления

Другим средством конфигурационного управления является PVCS Tracker - специализированная надстройка над офисной электронной почтой, предназначенная для обработки сообщений об ошибках в продукте, доставке их исполнителям и контроля за исполнением. Интеграция с PVCS Version Manager дает возможность связывать с сообщениями те или иные компоненты проекта. Отчетные возможности PVCS Tracker включают множество разновидностей графиков и диаграмм, отражающих состояние проекта и процесса его отладки, срезы по различным компонентам проекта, разработчикам и тестировщикам. С их помощью можно наглядно показать текущее состояние работы над проектом и ее временные тенденции.

Средства конфигурационного управления

Персонал, работающий с PVCS Tracker делится на пять групп в зависимости от их обязанностей: пользователи, разработчики, группа тестирования и контроля качества, группа технической поддержки и сопровождения, управленческий персонал. Этим пяти группам персонала соответствуют пять предопределенных групп PVCS Tracker:

- *пользователи (Submitters)* - имеют ограниченные права на внесение замечаний и сообщений об ошибках в базу данных PVCS Tracker;
- *разработчики (Development Engineers)* - имеют право производить основные операции с требованиями и замечаниями в базе данных PVCS Tracker. Если разработчики делятся на подгруппы, то для каждой подгруппы могут быть заданы отдельные списки прав доступа;
- *тестировщики (Quality Engineers)* - имеют право производить основные операции с требованиями и замечаниями;
- *сопровождение (Support Engineers)* - имеют право вносить любые замечания, требования и рекомендации в базу данных, но не имеют прав по распределению работ и изменению их приоритетности и сроков исполнения;
- *руководители (Managers)* - имеют право распределять работы между исполнителями и принимать решения о их надлежащем исполнении. Руководителям разных групп могут заданы различные права доступа к базе данных PVCS Tracker.

Средства конфигурационного управления

В дополнение к этим пяти предопределенным группам, существует группа администратора базы данных и 11 дополнительных групп, которые могут быть настроены в соответствии со специфическими должностными обязанностями сотрудников, использующих PVCS Tracker.

Требование или замечание поступающее в PVCS Tracker проходит четыре этапа обработки:

- регистрация - внесение замечания в базу данных;
- распределение - назначение ответственного исполнителя и сроков исполнения;
- исполнение - устранение замечания, которое в свою очередь может вызвать дополнительные замечания или требования на дополнительные работы;
- приемка - приемка работ и снятие их с контроля или направление на доработку.
- Требования и замечания, поступающие в базу данных PVCS Tracker оформляются в виде специальной формы, которая может содержать до 18 полей выбора стандартных значений и до 12 произвольных текстовых строк. При разработке формы следует определить оптимальный набор информации, характерный для всех записей в базе данных.

Средства конфигурационного управления

Для получения содержательной информации о ходе разработки PVCS Tracker позволяет получать три типа статистических отчетов: частотные, тренды и диаграммы распределения.

Частотные отчеты содержат информацию о частоте поступающих замечаний за один час тестирования программного продукта. Однако универсального частотного отчета не существует, т.к. на оценку качества влияют тип методов тестирования, серьезность выявленных ошибок и значение дефектных модулей для функционирования всей системы. Малое число фатальных ошибок, приводящих к полной остановке разработки, хуже большого числа замечаний к внешнему виду интерфейса пользователя. Следовательно, частотные отчеты должны быть настроены на выявление какого-либо конкретного аспекта качества для того, чтобы их можно было использовать для прогнозирования окончания работ над проектом.

Средства конфигурационного управления

Тренды содержат информацию об изменениях того или иного показателя во времени и характеризуют стабильность и непрерывность процесса разработки. Они позволяют ответить на вопросы:

- успевает ли группа разработчиков справляться с поступающими замечаниями;
- улучшается ли качество программного продукта и какова динамика этого процесса;
- как повлияло то или иное решение (увеличение числа разработчиков, введение скользящего графика, внедрение нового метода тестирования) на работу группы и т.п.

Диаграммы распределения - наиболее разнообразные и полезные для осуществления оперативного руководства формы отчетов. Они позволяют ответить на вопросы: какой метод тестирования более эффективен, какие модули вызывают наибольшее число нареканий, кто из разработчиков лучше справляется с конкретным типом заданий, нет ли перекоса в распределении работ между исполнителями, нет ли модулей, тестированию которых было уделено недостаточно внимания и т. д.

Средства конфигурационного управления

PVCS Tracker предназначен для использования в рабочих группах, объединенных в общую сеть. В этом случае центральная база или проект PVCS Tracker находится на общедоступном сервере сети, доступ к которому реализуется посредством ODBC-драйверов, входящих в состав PVCS Tracker. Главной особенностью PVCS Tracker по сравнению с обычным приложением СУБД является его способность автоматически уведомлять пользователя о поступлении интересующей его или относящейся к его компетенции информации и гибкая система распределения полномочий внутри рабочей группы. При необходимости PVCS Tracker может использовать для уведомления удаленных членов группы электронную почту.

PVCS Tracker поддерживает групповую работу в локальных сетях и взаимодействует с СУБД dBase, ORACLE, SQL Server и SYBASE посредством ODBC.

PVCS Tracker может быть интегрирован с любой системой электронной почты, поддерживающей стандарты VIM, MAPI или SMTP.

PVCS Version Manager и PVCS Tracker окружены вспомогательными компонентами: PVCS Configuration Builder и PVCS Notify.

PVCS Configuration Builder предназначен для сборки окончательного продукта из компонент проекта. PVCS Configuration Builder позволяет описывать процесс сборки как на стандартном языке MAKE, так и на собственном внутреннем языке, имеющем существенно большие возможности. PVCS Configuration Builder позволяет осуществлять сборку программного продукта на основании файлов, хранящихся в репозитории PVCS Version Manager.

Средства конфигурационного управления

Обычная процедура сборки программного продукта с помощью PVCS Configuration Builder состоит из трех шагов:

- строится файл зависимостей между исходными модулями;
- в полученный файл вносятся изменения с целью его настройки и оптимизации;
- осуществляется сборка программного продукта из исходных модулей.

Результатом работы PVCS Configuration Builder является специальный файл, описывающий оптимальный алгоритм сборки программного продукта, построенный на основе анализа дерева зависимостей между исходными модулями.

PVCS Notify обеспечивает автоматическую рассылку сообщений об ошибках из базы данных пакета PVCS Tracker по рабочим станциям назначения. При этом используется офисная система электронной почты cc:Mail или Microsoft Mail. PVCS Notify расширяет возможности PVCS Tracker и используется только совместно с ним.

Средства конфигурационного управления

PVCS Notify настраивается из среды PVCS Tracker. Настройка включает в себя определение интервала времени, через который PVCS Notify проверяет содержимое базы данных, определение критериев отбора записей для рассылки уведомлений, определение списков адресов для рассылки. После настройки PVCS Notify начинает работу в автономном режиме, автоматически рассылая уведомления об изменениях в базе данных PVCS Tracker.

PVCS Notify предназначен для использования в больших рабочих группах, часть членов которых хотя и доступна только через средства электронной почты, однако должна иметь оперативную информацию о требованиях на изменение программного продукта, замечаниях, ошибках, ходе и результатах его тестирования.

Результатом работы PVCS Notify являются оформленные в соответствии с одним из стандартов почтовые сообщения, готовые для рассылки посредством системы электронной почты.

Средства документирования

Средства документирования

Для создания документации в процессе разработки ИС используются разнообразные средства формирования отчетов, а также компоненты издательских систем. Обычно средства документирования встроены в конкретные CASE-средства. Исключением являются некоторые пакеты, предоставляющие дополнительный сервис при документировании. Из них наиболее активно используется SoDA (Software Document Automation).

Продукт SoDA предназначен для автоматизации разработки проектной документации на всех фазах ЖЦ ПО. Он позволяет автоматически извлекать разнообразную информацию, получаемую на разных стадиях разработки проекта, и включать ее в выходные документы. При этом контролируется соответствие документации проекту, взаимосвязь документов, обеспечивается их своевременное обновление. Результирующая документация автоматически формируется из множества источников, число которых не

Средства документирования

SoDA не зависит от применяемых инструментальных средств. Связь с приложениями осуществляется через стандартный программный интерфейс API. Переход на новые инструментальные средства не влечет за собой дополнительных затрат по документированию проекта.

SoDA содержит набор шаблонов документов, определяемых стандартом на программное обеспечение DOD 2167A. На их основе можно без специального программирования создавать новые формы документов, определяемые пользователями.

Пакет включает в себя графический редактор для подготовки шаблонов документов. Он позволяет задавать необходимый стиль, фон, шрифт, определять расположение заголовков, резервировать места, где будет размещаться извлекаемая из разнообразных источников информация. Изменения автоматически вносятся только в те части документации, на которые они повлияли в программе. Это сокращает время подготовки документации за счет отказа от перегенерации

Средства документирования

SoDA реализована на базе издательской системы FrameBuilder и предоставляет полный набор средств по редактированию и верстке выпускаемой документации. Разные версии документации могут быть для наглядности отмечены своими отличительными признаками. В системе создаются таблицы требований к проекту, по которым можно проследить, как реализуются эти требования. Разные виды документации, сопровождающие различные этапы ЖЦ, связаны между собой, и можно проследить состояние проекта от первоначальных требований до анализа, проектирования, кодирования и тестирования программного продукта.

Итоговым результатом работы системы SoDA является готовый документ (или книга). Документ может храниться в файле формата SoDA (Frame Builder), который получается в результате генерации документа. Вывод на печать этого документа (или его части) возможен из системы SoDA.

Среда функционирования SoDA - ОС типа UNIX на рабочих станциях Sun SPARCstation, IBM RISC System/6000 или Hewlett Packard HP 9000 700/800.

SoDA требует по крайней мере 32 MB оперативной памяти, 100-300 MB для установки и 64 MB рабочего пространства на диске.

Средства тестирования

Средства тестирования

Под тестированием понимается процесс исполнения программы с целью обнаружения ошибок. Регрессионное тестирование - это тестирование, проводимое после усовершенствования функций программы или внесения в нее изменений.

Одно из наиболее развитых средств тестирования QA (новое название - Quality Works) представляет собой интегрированную, многоплатформенную среду для разработки автоматизированных тестов любого уровня, включая тесты регрессии для приложений с графическим интерфейсом пользователя.

QA позволяет начинать тестирование на любой фазе ЖЦ, планировать и управлять процессом тестирования, отображать изменения в приложении и повторно использовать тесты для более чем 25 различных платформ.

Средства тестирования

Основными компонентами QA являются:

- QA Partner - среда для разработки, компиляции и выполнения тестов;
- QA Planner - модуль для разработки планов тестирования и обработки результатов. Для создания и выполнения тестов в процессе работы QA Planner вызывается QA Partner;
- Agent - модуль, поддерживающий работу в сети.

Процесс тестирования состоит из следующих этапов:

- создание плана тестирования;
- связывание плана с тестами;
- пометка и выполнение тестов;
- получение отчетов о тестировании и управление результатами.

Средства тестирования

Создание тестового плана в QA Planner включает в себя составление схемы тестовых требований и выделение уровней детализации. Для этого необходимо определить все, что должно быть протестировано, подготовить функциональную декомпозицию приложения, оценить, сколько тестов необходимо для каждой функции и характеристики, определить, сколько из них будет реализовано в зависимости от доступных ресурсов и времени. Эта информация используется для создания схемы тестовых требований.

Для связывания плана с тестами необходимо создать управляющие предложения (скрипты) на специальном языке 4Test и тесты, которые выполняют требования плана, и связать компоненты любым способом. Для избежания перегруженности тестов используют управление тестовыми данными.

При выполнении плана результаты записываются в формате, похожем на план. Все результаты связаны с планом. Есть возможность просмотреть или скрыть общую информацию о выполнении, слить файлы результатов, разметить неудавшиеся тесты, сравнить результаты предыдущего выполнения тестов, выполнить или отменить отчет.

Одним из атрибутов теста является имя его разработчика, что позволяет при необходимости выполнять тесты, созданные конкретным разработчиком.

Комплекс QA занимает на жестком диске не более 21MB. Поддерживаемые платформы: Windows 3.x, Windows 95, Windows NT, OS/2, Macintosh, VMS, HP-UX, AIX, Solaris.

Примеры комплексов CASE-средств

Примеры комплексов CASE-средств

В заключение приведем примеры комплексов CASE-средств обеспечивающих поддержку полного ЖЦ ПО. Здесь хотелось бы еще раз отметить нецелесообразность сравнения отдельно взятых CASE-средств, поскольку ни одно из них не решает в целом все проблемы создания и сопровождения ПО. Это подтверждается также полным набором критериев оценки и выбора, которые затрагивают все этапы ЖЦ ПО. Сравниваться могут комплексы методологически и технологически согласованных инструментальных средств, поддерживающие полный ЖЦ ПО и обеспеченные необходимой технической и методической поддержкой со стороны фирм-поставщиков. По мнению автора, на сегодняшний день наиболее развитым из всех поставляемых в России комплексов такого рода является комплекс технологий и инструментальных средств создания ИС, основанный на методологии и технологии DATARUN. В состав комплекса входят следующие инструментальные средства:

- CASE-средство Silverrun;
- средство разработки приложений JAM;
- мост Silverrun-RDM <-> JAM;
- комплекс средств тестирования QA;
- менеджер транзакций Tuxedo;
- комплекс средств планирования и управления проектом SE Companion;
- комплекс средств конфигурационного управления PVCS;
- объектно-ориентированное CASE-средство Rational Rose;
- средство документирования SoDA.

Примеры комплексов CASE-средств

Примерами других подобных комплексов являются:

- Vantage Team Builder for Uniface + Uniface (фирмы "DataX/Florin" и "ЛАНИТ");
- комплекс средств, поставляемых и используемых фирмой "ФОРС":
- CASE-средства Designer/2000 (основное), ERwin, Bpwin и Oowin (альтернатив-
  ные);
- средства разработки приложений Developer/2000, ORACLE Power Objects (ос-
  новные) и Usoft Developer (альтернативное);
- средство настройки и оптимизации ExplainSQL (Platinum);
- средства администрирования и сопровождения SQLWatch, DBVision, SQL Spy, TSReorg и др. (Platinum);
- средство документирования ORACLE Book.
- комплекс средств на основе продуктов фирмы CENTURA:
- CASE-средства ERwin, Bpwin и Oowin (объектно-ориентированный анализ);
- средства разработки приложений SQLWindows и TeamWindows;
- средство тестирования и оптимизации приложений "клиент-сервер" SQLBench (ARC);
- средства эксплуатации и сопровождения Quest и Crystal Reports.