Бинго! Ты только что нашел решение своей проблемы! Только давай договоримся – ты прочтешь текст до конца, окей? :)

Давай начистоту - тут один шлак, лучше закажи работу на[author-24.site](http://author-24.site/) и не парься – мы все сделаем за тебя! Даже если остался **день** до сдачи работы – мы справимся, и ты получишь «Отлично» по своему предмету! Только представь: ты занимаешься своим любимым делом, пока твои лохи-одногруппники теряют свои нервные клетки…

Проникнись… Это бесценное ощущение :)

Курсовая, диплом, реферат, статья, эссе, чертежи, задачи по матану, контрольная или творческая работа – всё это ты можешь передать нам, наслаждаться своей молодостью, гулять с друзьями и радовать родителей отличными оценками. А если преподу что-то не понравится, то мы **бесплатно** переделаем так, что он пустит слезу от счастья и поставит твою работу в рамочку как образец качества.

Еще сомневаешься? Мы готовы подарить тебе **сотни часов свободного времени** за смешную цену – что тут думать-то? Жизнь одна – не трать ее на всякую фигню!

Перейди на наш сайт [author-24.site](http://author-24.site/) - обещаю, тебе понравится! :)

А работа, которую ты искал, находится ниже :)

Министерство образования Омской области

Государственное образовательное учреждение Омской области среднего профессионального образования

«Омский государственный колледж управления и профессиональных технологий»

Методология RAD разработки информационных систем

**Курсовая работа**

2007

RAD (от англ. rapid application development — быстрая разработка приложений) — концепция создания средств разработки программных продуктов, уделяющая особое внимание быстроте и удобству программирования, созданию технологического процесса, позволяющего программисту максимально быстро создавать компьютерные программы. С конца XX века RAD получила широкое распространение и одобрение. Концепцию RAD также часто связывают с концепцией визуального программирования.

## Основные принципы RAD

* Инструментарий должен быть нацелен на минимизацию времени разработки.
* Создание прототипа для уточнения требований заказчика.
* Цикличность разработки: каждая новая версия продукта основывается на оценке результата работы предыдущей версии заказчиком.
* Минимизация времени разработки версии, за счёт переноса уже готовых модулей и добавления функциональности в новую версию.
* Команда разработчиков должна тесно сотрудничать, каждый участник должен быть готов выполнять несколько обязанностей.
* Управление проектом должно минимизировать длительность цикла разработки.

## История

Концепция RAD стала ответом на неуклюжие методы разработки программ 1970-х и начала 1980-х годов, такие как «модель водопада» (англ. Waterfall model). Эти методы предусматривали настолько медленный процесс создания программы, что зачастую даже требования к программе успевали измениться до окончания разработки. Основателем RAD считается сотрудник IBM Джеймс Мартин, который в 1980-х годах сформулировал основные принципы RAD, основываясь на идеях Барри Бойма и Скотта Шульца. А в 1991 году Мартин опубликовал известную книгу, в которой детально изложил концепцию RAD и возможности её применения. В настоящее время RAD становится общепринятой схемой для создания средств разработки программных продуктов. Именно средства разработки, основанные на RAD, имеют наибольшую популярность среди программистов.

## Среды разработки, использующие принципы RAD

* Borland Delphi
* Borland C++ Builder
* Microsoft Visual Studio
* Macromedia Flash
* Macromedia Authorware
* Macromedia Director
* Omnis Studio
* Visual DataFlex
* IntraWeb

Быстрая разработка приложений

**Rapid Application Development (RAD)** – это жизненный цикл процесса проектирования, созданный для достижения более высоких скорости разработки и качества ПО, чем это возможно при традиционном подходе к проектированию.



RAD предполагает, что разработка ПО осуществляется небольшой командой разработчиков за срок порядка трех-четырех месяцев путем использования инкрементного прототипирования с применением инструментальных средств визуального моделирования и разработки. Технология RAD предусматривает активное привлечение заказчика уже на ранних стадиях – обследование организации, выработка требований к системе. Причины популярности RAD вытекают из тех преимуществ, которые обеспечивает эта технология.

Наиболее существенными из них являются:

* высокая скорость разработки;
* низкая стоимость;
* высокое качество.

Последнее из указанных свойств подразумевает полное выполнение требований заказчика как функциональных, так и нефункциональных, с учетом их возможных изменений в период разработки системы, а также получение качественной документации, обеспечивающей удобство эксплуатации и сопровождения системы. Это означает, что дополнительные затраты на сопровождение сразу после поставки будут значительно меньше. Таким образом, полное время от начала разработки до получения приемлемого продукта при использовании этого метода значительно сокращается.

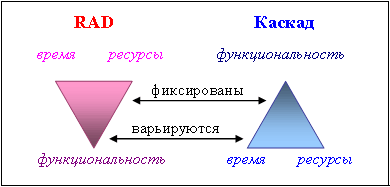
Когда применяется RAD

Применение технологии RAD целесообразно, когда:

* требуется выполнение проекта в сжатые сроки (90 дней). Быстрое выполнение проекта позволяет создать систему, отвечающую требованиям сегодняшнего дня. Если система проектируется долго, то весьма высока вероятность, что за это время существенно изменятся фундаментальные положения, регламентирующие деятельность организации, то есть, система морально устареет еще до завершения ее проектирования.
* нечетко определены требования к ПО. В большинстве случаев заказчик весьма приблизительно представляет себе работу будущего программного продукта и не может четко сформулировать все требования к ПО. Требования могут быть вообще не определены к началу проекта либо могут изменяться по ходу его выполнения.
* проект выполняется в условиях ограниченности бюджета. Разработка ведется небольшими RAD группами в короткие сроки, что обеспечивает минимум трудозатрат и позволяет вписаться в бюджетные ограничения.
* интерфейс пользователя (GUI) есть главный фактор. Нет смысла заставлять пользователя рисовать картинки. RAD технология дает возможность продемонстрировать интерфейс в прототипе, причем достаточно скоро после начала проекта.
* проект большой, но поддается разделению на более мелкие функциональные компоненты. Если предполагаемая система велика, необходимо, чтобы ее можно было разбить на мелкие части, каждая из которых обладает четкой функциональностью. Они могут выпускаться последовательно или параллельно (в последнем случае привлекается несколько RAD групп).
* ПО не обладает большой вычислительной сложностью.

Современные средства быстрой разработки windows-при-ложений, так называемые rad-средства (rad расшифровывается как rapid application development), обладают в той или иной степени почти всеми возможностями реализации в приложениях подобных интерфейсных элементов. Многие из них позволяют осуществлять доступ к базам данных, в том числе и к серверным БД. borland delphi (как версия 1.0, так и версия 2.0), на взгляд автора, является в этом отношении наиболее простым и удобным в использовании средством.

RAD-технология не является универсальной, то есть ее применение целесообразно не всегда. Например, в проектах, где требования к программному продукту четко определены и не должны меняться, вовлечение заказчика в процесс разработки не требуется и более эффективной может быть иерархическая разработка (каскадный метод). То же касается проектов, ПО, сложность которых определяется необходимостью реализации сложных алгоритмов, а роль и объем пользовательского интерфейса невелик.



Принципы организации RAD

Принципы RAD технологии направлены на обеспечение трех основных ее преимуществ – высокой скорости разработки, низкой стоимости и высокого качества. Достигнуть высокого качества программного продукта весьма непросто и одна из главных причин возникающих трудностей заключается в том, что разработчик и заказчик видят предмет разработки (ПО) по-разному.

Главная идея RAD технологии состоит в том, чтобы как можно быстрее донести до заказчика результаты разработки, пусть и не в полном виде. Например, реализация только пользовательского интерфейса и предъявление его заказчику позволяет уже на ранней стадии разработки получить замечания по экранным и отчетным формам и внести необходимые коррективы. В этом случае значительно возрастает вероятность успеха проекта, то есть возникает уверенность в том, что конечный продукт будет делать именно то, что ожидает заказчик. Кроме того, не следует забывать и тот факт, что разница стоимости ошибки определения требований в начале проекта и в конце равна 1:200.

Основные принципы RAD можно сформулировать следующим образом:

* Работа ведется группами. Типичный состав группы - руководитель, аналитик, два программиста, технический писатель. Если проект сложный, то для него может быть выделено несколько RAD-групп. Разработка проекта выполняется в условиях тесного взаимодействия между разработчиками и Заказчиком.
* Разработка базируется на моделях. Моделирование позволяет оценить проект и выполнить его декомпозицию на составные части, каждая из которых может разрабатываться отдельной RAD-группой.
* Итерационное прототипирование. Разработка системы и предъявление ее заказчику осуществляется в виде последовательности развиваемых прототипов. Любой из прототипов реализует определенную часть функциональности, требуемой от конечного продукта. При этом каждый последующий прототип включает всю функциональность, реализованную в предыдущем прототипе, с добавлением новой. Число прототипов определяется на основе учета разных параметров – размера проекта, анализа рисков, пожеланий заказчика и т. д. Традиционно для проектов ПО средней сложности разрабатываются три прототипа. Первый содержит весь пользовательский интерфейс с нулевой функциональностью. Он дает возможность собрать замечания заказчика и после их устранения утвердить у него экранные и отчетные формы. Второй прототип содержит реализованную на 70-80% функциональность системы, третий – полностью реализованную функциональность. Основаниями для очередной итерации являются:
  1. Замечания заказчика. Привлечение заказчика и конечного пользователя к оценке выходных результатов прототипа с эффективной обратной связью с командой разработчиков является гарантией того, что созданная система будет делать то, что требуется заказчику. Если замечания носят характер исправлений, они учитываются в следующем прототипе, если же изменяются требования, то выполняется переоценка проекта и корректируются сроки и стоимость проекта.
  2. Детализация. Выполняется программирование нереализованной части системы в соответствии с составленным планом.
  3. Анализ результатов программирования. Исправляются ошибки, повышается эффективность программного кода и т. д.
* RAD группа всегда работает только над одним прототипом. Это обеспечивает единство целей, лучшую наблюдаемость и управляемость процессом разработки, что в итоге повышает качество конечного продукта. Соответственно используемые инструментальные средства должны обеспечивать групповую разработку и конфигурационное управление проектом.
* Если проект сложный, то для него может быть выделено несколько RAD групп. Большие системы разбиваются на подсистемы. Каждая подсистема разрабатывается независимой группой. Ключ успеха – правильное разбиение системы на подсистемы. Команды должны использовать общие стандарты. Обязательно финальное тестирование полной системы.
* Обязательное использование инструментальных средств, автоматизирующих процесс разработки, и методик их использования, следствием чего является сокращение сроков разработки и повышение качества конечного продукта.

Принципы RAD применяются не только при реализации, но и распространяются на все этапы жизненного цикла, в частности на этап обследования организации, построения требований, анализ и дизайн.

# Современные средства быстрой разработки

Рассуждая об "идеологической совместимости" Free Pascal и Delphi, необходимо отдать должное проекту Lazarus, в рамках которого и реализуется идеология быстрого визуального программирования.В основе проекта лежит библиотека визуальных компонентов Lazarus (LCL), для которой декларируется совместимость с визуальными компонентами VCL из Delphi. Библиотека LCL является платформонезависимой и утверждается, что исходные коды приложений могут быть портированы на любую из поддерживаемых платформ.

Современные средства быстрой разработки windows-при-ложений, так называемые rad-средства (rad расшифровывается как rapid application development), обладают в той или иной степени почти всеми возможностями реализации в приложениях подобных интерфейсных элементов. Многие из них позволяют осуществлять доступ к базам данных, в том числе и к серверным БД. borland delphi (как версия 1.0, так и версия 2.0), на взгляд автора, является в этом отношении наиболее простым и удобным в использовании средством.

Линейка VS.NET будет включать выпуски Academic, Professional и две версии Enterprise (Architect и Developer). Обсуждение конкретных вариантов поставок лучше отложить до официального объявления продукта.

Объектная модель среды разработки (IDE) базируется на корневом объекте Development Tools Extensibility (DTE), который находится в пространстве имен EnvDTE библиотек классов.NET Framework. Через него можно получить ссылки на все множество объектов, соответствующих отдельным элементам IDE, таким, как Windows, Documents, Solutions, Projects, Debugger и Events.

Каждый из этих объектов позволяет обратиться в собственной иерархической системе объектов, коллекций, свойств, методов и событий. Многие из этих компонентов имеют перекрестные связи, поэтому доступ к тому или иному объекту можно получить несколькими путями. Эта система объектов позволяет обращаться практически к любым компонентам и функциям среды разработки VS.NET.

**Разработка интерфейса средствами RAD**

Процедура разработки интерфейса средствами RAD сводится к последовательному выполнению трех операций:

* размещение компонентов интерфейса в нужном месте.
* задание моментов времени их появления на экране.
* настройка связанных с ними атрибутов и событий.

В идеале среда визуальной разработки должна позволять быстро перетаскивать компоненты мышью и задавать значения изменяемых параметров. Если на выполнение каждой операции будет уходить ощутимое время, то проектирование сложного интерфейса превратится в очень длительную процедуру. Эффективность визуального программирования определяется не столько наличием визуальных компонентов, сколько их взаимосвязью и взаимодействием с традиционными средствами.

Интегрированная среда разработки является средством, с помощью которого выполняются проектирование, отладка, тестирование и дальнейшее распространение прикладных программ. Для повышения эффективности данного процесса каждый из инструментов (конструкторы, отладчики и т. д.) должен быть реализован на очень привычном уровне уровне.

Даже если среда не содержит достаточного количества требуемых компонентов, она все равно будет востребована, если позволяет использовать имеющиеся на рынке средства, альтернативные отсутствующим в ней.

Что обеспечивает RAD технология

Технология RAD обеспечивает:

* быстроту продвижения программного продукта на рынок;
* интерфейс, устраивающий пользователя;
* легкую адаптируемость проекта к изменяющимся требованиям;
* простоту развития функциональности системы.

#### Методология RAD

Одним из возможных подходов к разработке ПО в рамках спиральной модели ЖЦ является получившая в последнее время широкое распространение методология быстрой разработки приложений RAD (Rapid Application Development). Под этим термином обычно понимается процесс разработки ПО, содержащий 3 элемента:

* небольшую команду программистов (от 2 до 10 человек);
* короткий, но тщательно проработанный производственный график (от 2 до 6 мес.);
* повторяющийся цикл, при котором разработчики, по мере того, как приложение начинает обретать форму, запрашивают и реализуют в продукте требования, полученные через взаимодействие с заказчиком.

Команда разработчиков должна представлять из себя группу профессионалов, имеющих опыт в анализе, проектировании, генерации кода и тестировании ПО с использованием CASE-средств. Члены коллектива должны также уметь трансформировать в рабочие прототипы предложения конечных пользователей.

Жизненный цикл ПО по методологии RAD состоит из четырех фаз:

* фаза анализа и планирования требований;
* фаза проектирования;
* фаза построения;
* фаза внедрения.

На фазе анализа и планирования требований пользователи системы определяют функции, которые она должна выполнять, выделяют наиболее приоритетные из них, требующие проработки в первую очередь, описывают информационные потребности. Определение требований выполняется в основном силами пользователей под руководством специалистов-разработчиков. Ограничивается масштаб проекта, определяются временные рамки для каждой из последующих фаз. Кроме того, определяется сама возможность реализации данного проекта в установленных рамках финансирования, на данных аппаратных средствах и т.п. Результатом данной фазы должны быть список и приоритетность функций будущей ИС, предварительные функциональные и информационные модели ИС.

На фазе проектирования часть пользователей принимает участие в техническом проектировании системы под руководством специалистов-разработчиков. CASE-средства используются для быстрого получения работающих прототипов приложений. Пользователи, непосредственно взаимодействуя с ними, уточняют и дополняют требования к системе, которые не были выявлены на предыдущей фазе. Более подробно рассматриваются процессы системы. Анализируется и, при необходимости, корректируется функциональная модель. Каждый процесс рассматривается детально. При необходимости для каждого элементарного процесса создается частичный прототип: экран, диалог, отчет, устраняющий неясности или неоднозначности. Определяются требования разграничения доступа к данным. На этой же фазе происходит определение набора необходимой документации.

После детального определения состава процессов оценивается количество функциональных элементов разрабатываемой системы и принимается решение о разделении ИС на подсистемы, поддающиеся реализации одной командой разработчиков за приемлемое для RAD-проектов время - порядка 60 - 90 дней. С использованием CASE-средств проект распределяется между различными командами (делится функциональная модель). Результатом данной фазы должны быть:

* общая информационная модель системы;
* функциональные модели системы в целом и подсистем, реализуемых отдельными командами разработчиков;
* точно определенные с помощью CASE-средства интерфейсы между автономно разрабатываемыми подсистемами;
* построенные прототипы экранов, отчетов, диалогов.

Все модели и прототипы должны быть получены с применением тех CASE-средств, которые будут использоваться в дальнейшем при построении системы. Данное требование вызвано тем, что в традиционном подходе при передаче информации о проекте с этапа на этап может произойти фактически неконтролируемое искажение данных. Применение единой среды хранения информации о проекте позволяет избежать этой опасности.

В отличие от традиционного подхода, при котором использовались специфические средства прототипирования, не предназначенные для построения реальных приложений, а прототипы выбрасывались после того, как выполняли задачу устранения неясностей в проекте, в подходе RAD каждый прототип развивается в часть будущей системы. Таким образом, на следующую фазу передается более полная и полезная информация.

На фазе построения выполняется непосредственно сама быстрая разработка приложения. На данной фазе разработчики производят итеративное построение реальной системы на основе полученных в предыдущей фазе моделей, а также требований нефункционального характера. Программный код частично формируется при помощи автоматических генераторов, получающих информацию непосредственно из репозитория CASE-средств. Конечные пользователи на этой фазе оценивают получаемые результаты и вносят коррективы, если в процессе разработки система перестает удовлетворять определенным ранее требованиям. Тестирование системы осуществляется непосредственно в процессе разработки.

После окончания работ каждой отдельной команды разработчиков производится постепенная интеграция данной части системы с остальными, формируется полный программный код, выполняется тестирование совместной работы данной части приложения с остальными, а затем тестирование системы в целом. Завершается физическое проектирование системы:

* определяется необходимость распределения данных;
* производится анализ использования данных;
* производится физическое проектирование базы данных;
* определяются требования к аппаратным ресурсам;
* определяются способы увеличения производительности;
* завершается разработка документации проекта.

Результатом фазы является готовая система, удовлетворяющая всем согласованным требованиям.

На фазе внедрения производится обучение пользователей, организационные изменения и параллельно с внедрением новой системы осуществляется работа с существующей системой (до полного внедрения новой). Так как фаза построения достаточно непродолжительна, планирование и подготовка к внедрению должны начинаться заранее, как правило, на этапе проектирования системы. Приведенная схема разработки ИС не является абсолютной. Возможны различные варианты, зависящие, например, от начальных условий, в которых ведется разработка: разрабатывается совершенно новая система; уже было проведено обследование предприятия и существует модель его деятельности; на предприятии уже существует некоторая ИС, которая может быть использована в качестве начального прототипа или должна быть интегрирована с разрабатываемой.

Следует, однако, отметить, что методология RAD, как и любая другая, не может претендовать на универсальность, она хороша в первую очередь для относительно небольших проектов, разрабатываемых для конкретного заказчика. Если же разрабатывается типовая система, которая не является законченным продуктом, а представляет собой комплекс типовых компонент, централизованно сопровождаемых, адаптируемых к программно-техническим платформам, СУБД, средствам телекоммуникации, организационно-экономическим особенностям объектов внедрения и интегрируемых с существующими разработками, на первый план выступают такие показатели проекта, как управляемость и качество, которые могут войти в противоречие с простотой и скоростью разработки. Для таких проектов необходимы высокий уровень планирования и жесткая дисциплина проектирования, строгое следование заранее разработанным протоколам и интерфейсам, что снижает скорость разработки.

Методология RAD неприменима для построения сложных расчетных программ, операционных систем или программ управления космическими кораблями, т.е. программ, требующих написания большого объема (сотни тысяч строк) уникального кода.

Не подходят для разработки по методологии RAD приложения, в которых отсутствует ярко выраженная интерфейсная часть, наглядно определяющая логику работы системы (например, приложения реального времени) и приложения, от которых зависит безопасность людей (например, управление самолетом или атомной электростанцией), так как итеративный подход предполагает, что первые несколько версий наверняка не будут полностью работоспособны, что в данном случае исключается.

Оценка размера приложений производится на основе так называемых функциональных элементов (экраны, сообщения, отчеты, файлы и т.п.) Подобная метрика не зависит от языка программирования, на котором ведется разработка. Размер приложения, которое может быть выполнено по методологии RAD, для хорошо отлаженной среды разработки ИС с максимальным повторным использованием программных компонентов, определяется следующим образом:

|  |  |
| --- | --- |
| **< 1000** функциональных элементов | один человек |
| **1000-4000** функциональных элементов | одна команда разработчиков |
| **> 4000** функциональных элементов | 4000 функциональных элементов на одну команду разработчиков |

Основные принципы RAD:

* разработка приложений итерациями;
* необязательность полного завершения работ на каждом из этапов жизненного цикла;
* обязательное вовлечение пользователей в процесс разработки ИС;
* необходимое применение CASE-средств, обеспечивающих целостность проекта;
* применение средств управления конфигурацией, облегчающих внесение изменений в проект и сопровождение готовой системы;
* необходимое использование генераторов кода;
* использование прототипирования, позволяющее полнее выяснить и удовлетворить потребности конечного пользователя;
* тестирование и развитие проекта, осуществляемые одновременно с разработкой;
* ведение разработки немногочисленной хорошо управляемой командой профессионалов;
* грамотное руководство разработкой системы, четкое планирование и контроль выполнения работ.

### Инструментальные средства быстрой разработки приложений (RAD) и баз данных

Актуальная тенденция в средствах разработки рекламирует визуальные среды проектирования, объектные шаблоны и увеличиваемые возможности доступа к данным. Здесь объектность и реляционность мирно сосуществуют в виде объектно - ориентированного языка разработки и реляционного источника данных. Средства разработки эксплуатируют такие методологии как drag-and-drop, визуальное построение приложения, повторное использование кода, программные компоненты. В отношении поддержки базы данных, они обеспечивают конечное представление базы данных, управляемое формами, и связность базы данных через локальные драйверы и поддержку промышленных стандартов, например, ODBC.

Новые версии визуальных windows баз данных предлагают возможности, обычно типичные для RAD инструментальных средств, типа управляемого событиями программирования, способности визуально создать компоненты многократного использования, и интегрированную связность базы данных на клиенте и сервере. Они могут создавать специализированные классы и подклассы или в коде, или визуально с проектировщиком класса. И они обеспечивают SQL запросы к главным базам данных типа Oracle, Sybase, Microsoft SQL server, а также также поддержку ODBC драйверы.

Такая возможность баз данных использовать одиночное приложение или внешний интерфейс к данным для доступа к различным базам данных на различных вычислительных платформах иногда называется масштабируемостью, при этом различают горизонтальную масштабируемость (способность приложения поддерживать множественные, параллельные процессоры) и вертикальную масштабируемость(способность перемещать приложение среди множества вычислительных платформ).

Эти настольные базы данных функционируют обычно на DOS, Windows, Mac, и платформах OS/2, хотя некоторые могут выполняться на Unix платформах (например, dBASE на Unix платформах, и SQLBase Gupta's на SunOS). Также, большинство их клиент/серверных версий поддерживают объектную технологию OLE за исключением VisualAge для System Object Model (SOM) и DSOM.

ODBC - это API, основанный на спецификации Call Level Interface(CLI) и грамматике SQL от SQL Access Group. Первоначально предложенный Microsoft, ODBC обеспечивает нейтральный, не зависящий от продавца БД, MS Windows - механизм для независимого доступа к множественным хостам базы данных. ODBC таким образом разрешает, чтобы разработчики программного обеспечения создавали настольные приложения, не тратя времени на изучение API базы данных. Другое преимущество ODBC - способность сохранить данные для различных приложений или данных из различных источников в любой базе данных, при этом подробности внутренних структур данных скрыты от пользователя.

Как стандарт, ODBC имеет сильную промышленную поддержку. В настоящее время большая часть продавцов программного обеспечения и оборудования, включая и Microsoft и Apple, одобрила ODBC как стандарт способности к взаимодействию базы данных. Кроме того, большинство продавцов базы данных или обеспечивает, или будет скоро обеспечивать, ODBC -совместимые интерфейсы.

Современные средства быстрой разработки windows-при-ложений, так называемые rad-средства (rad расшифровывается как rapid application development), обладают в той или иной степени почти всеми возможностями реализации в приложениях подобных интерфейсных элементов. Многие из них позволяют осуществлять доступ к базам данных, в том числе и к серверным БД. borland delphi (как версия 1.0, так и версия 2.0), на взгляд автора, является в этом отношении наиболее простым и удобным в использовании средством.

Инструментальные средства быстрой разработки приложений (RAD) и баз данных. Актуальная тенденция в средствах разработки рекламирует визуальные среды проектирования, объектные шаблоны и увеличиваемые возможности доступа к данным. Здесь объектность и реляционность мирно сосуществуют в виде объектно - ориентированного языка разработки и реляционного источника данных. Средства разработки эксплуатируют такие методологии как drag-and-drop, визуальное построение приложения, повторное использование кода, программные компоненты. В отношении поддержки базы данных, они обеспечивают конечное представление базы данных, управляемое формами, и связность базы данных через локальные драйверы и поддержку промышленных стандартов.

Новые версии визуальных windows баз данных предлагают возможности, обычно типичные для RAD инструментальных средств, типа управляемого событиями программирования, способности визуально создать компоненты многократного использования, и интегрированную связность базы данных на клиенте и сервере. Они могут создавать специализированные классы и подклассы или в коде, или визуально с проектировщиком класса. И они обеспечивают SQL запросы к главным базам данных типа Oracle, Sybase, Microsoft SQL server, а также также поддержку ODBC драйверы.

Такая возможность баз данных использовать одиночное приложение или внешний интерфейс к данным для доступа к различным базам данных на различных вычислительных платформах иногда называется масштабируемостью, при этом различают горизонтальную масштабируемость (способность приложения поддерживать множественные, параллельные процессоры) и вертикальную масштабируемость(способность перемещать приложение среди множества вычислительных платформ).

Эти настольные базы данных функционируют обычно на DOS, Windows, Mac, и платформах OS/2, хотя некоторые могут выполняться на Unix платформах (например, dBASE на Unix платформах, и SQLBase Gupta's на SunOS). Также, большинство их клиент/серверных версий поддерживают объектную технологию OLE за исключением VisualAge для System Object Model (SOM) и DSOM.

ODBC - это API, основанный на спецификации Call Level Interface(CLI) и грамматике SQL от SQL Access Group. Первоначально предложенный Microsoft, ODBC обеспечивает нейтральный, не зависящий от продавца БД, MS Windows - механизм для независимого доступа к множественным хостам базы данных. ODBC таким образом разрешает, чтобы разработчики программного обеспечения создавали настольные приложения, не тратя времени на изучение API базы данных. Другое преимущество ODBC - способность сохранить данные для различных приложений или данных из различных источников в любой базе данных, при этом подробности внутренних структур данных скрыты от пользователя.

Как стандарт, ODBC имеет сильную промышленную поддержку. В настоящее время большая часть продавцов программного обеспечения и оборудования, включая и Microsoft и Apple, одобрила ODBC как стандарт способности к взаимодействию базы данных. Кроме того, большинство продавцов базы данных или обеспечивает, или будет скоро обеспечивать, ODBC -совместимые интерфейсы.

**Редактор метаданных**

Для обеспечения быстроты построения систем в рамках технологии "KERNEL", был необходим инструмент прикладного разработчика, который бы позволил ускорить и упростить процесс разработки как таковой. Решением стал уникальный специализированный редактор метаданных. Редактор реализован в соответствии с принципами методологии быстрой разработки (RAD) и визуального конструирования, и взаимного связывания объектов доступа к данным приложения и интерфейса пользователя. Для работы в редакторе метаданных не требуются специальные знания в области программирования. Важно отметить наличие в редакторе метаданных механизма перенесения объектов бизнес логики описанных в схеме разработчиков в рабочую схему. Это позволяет вести доработку уже внедренной системы в связи с изменившимися требованиями бизнеса без остановки рабочего процесса. Кроме того, механизмы переноса объектов бизнес логики позволяют реализовать параллельную разработку частей системы, (например специалистами фирмы – заказчика и программистами нашей компании, привлеченными для внесения в систему нового сложного функционала) с последующей синхронизацией на эксплуатирующейся версии.

**Заключение**

В моей работе были рассмотрена методология RAD технология ИС.

Основные принципы методологии RAD:

* разработка приложений итерациями;
* необязательность полного завершения работ на каждом из этапов жизненного цикла;
* обязательное вовлечение пользователей в процесс разработки ИС;
* необходимое применение CASE-средств, обеспечивающих целостность проекта;
* применение средств управления конфигурацией, облегчающих внесение изменений в проект и сопровождение готовой системы;
* необходимое использование генераторов кода;
* использование прототипирования, позволяющее полнее выяснить и удовлетворить потребности конечного пользователя;
* тестирование и развитие проекта, осуществляемые одновременно с разработкой;
* ведение разработки немногочисленной хорошо управляемой командой профессионалов;
* грамотное руководство разработкой системы, четкое планирование и контроль выполнения работ.

**Список источников**

1. http://ru.wikipedia.org
2. http://www.inforazrabotky.info
3. http://brain.botik.ru
4. http://promidi.by.ru
5. http://www.citforum.ru