Содержание:

ВВЕДЕНИЕ

На сегодняшний день в программной инженерии существуют два основных подхода к разработке программного обеспечения экономической информационной системы, принципиальное различие между которыми обусловлено разными способами декомпозиции (разбиения) систем. Первый подход называют функционально-модульным или структурным, второй - объектно-ориентированным.

Цель данной работы – рассмотреть теоретические аспекты объектно-ориентированного подхода к построению экономической информационной системы.

Данная тема актуальна тем, что в условиях развития технологий, потребность в создании и поддерживании информационных систем растет с огромной скоростью, и важным вопросом является определение наиболее удобного, практичного и экономичного подхода к построению этих систем.

Основные задачи в процессе выполнения работы:

1. Рассмотреть понятие и сущность объектно-ориентированного подхода к построению информационной системы,
2. Определить наиболее подходящий язык,
3. Рассмотреть средства реализации подхода и определить их плюсы и минусы.

Глава 1. Сущность объектно-ориентированного подхода к построению информационной системы

1.1 Определение, понятия и основные принципы объектно-ориентированного подхода

Объектно-ориентированный подход (ООП) использует объектную декомпозицию. При этом структура системы описывается в терминах объектов и связей между

ними, а поведение системы описывается в терминах обмена сообщениями между объектами. Каждый объект системы обладает своим собственным поведением, моделирующим поведение объекта реального мира [6]. Рассмотрим более подробно ООП.

Понятие "объект" впервые было использовано около 30 лет назад в технических средствах при попытках отойти от традиционной архитектуры фон Неймана и преодолеть барьер между высоким уровнем программных абстракций и низким уровнем абстрагирования на уровне компьютеров. С объектно-ориентированной архитектурой также тесно связаны объектно-ориентированные операционные системы. Однако наиболее значительный вклад в объектный подход был внесен объектными и объектно-ориентированными языками программирования: Simula, Smalltalk, C++, Object Pascal. На объектный подход оказали влияние также развивавшиеся достаточно независимо методы моделирования баз данных, в особенности подход "сущность-связь"[6].

Основой ООП является объектная модель. Эта модель состоит из следующих четырех главных элементов:

1. Абстракция,
2. Инкапсуляция,
3. Модульность,
4. Иерархия.

Термин «главный элемент» означает, что без любого из них модель не является объектно-ориентированной. Кроме главных, в этой модели еще существует три дополнительных элемента:

1. Контроль типов,
2. Параллелизм,
3. Персистентность.

Термин «дополнительный элемент» означает полезный, но не существенный компонент объектной модели [1].

Абстракция (абстрагирование) - это выделение существенных характеристик некоторого объекта, которые отличают его от всех других видов объектов и, таким образом, четко определяют его концептуальные границы относительно дальнейшего рассмотрения и анализа. Абстрагирование концентрирует внимание на внешних особенностях объекта и позволяет отделить самые существенные

особенности его поведения от деталей их реализации. Выбор правильного набора абстракций для заданной предметной области представляет собой главную задачу объектно-ориентированного проектирования.

Инкапсуляция — это процесс отделения друг от друга отдельных элементов объекта, определяющих его устройство и поведение. Инкапсуляция служит для того, чтобы изолировать интерфейс объекта, отражающий его внешнее поведение, от внутренней реализации объекта. Объектный подход предполагает, что собственные ресурсы, которыми могут манипулировать только методы самого класса, скрыты от внешней среды. Абстрагирование и инкапсуляция являются взаимодополняющими операциями: абстрагирование фокусирует внимание на внешних особенностях объекта, а инкапсуляция (или, иначе, ограничение доступа) не позволяет объектам-пользователям различать внутреннее устройство объекта.

Модульность — это свойство системы, связанное с возможностью ее декомпозиции на ряд внутренне связных, но слабо связанных между собой модулей. Инкапсуляция и модульность создают барьеры между абстракциями.

Иерархия — это ранжированная или упорядоченная система абстракций, расположение их по уровням. Основными видами иерархических структур применительно к сложным системам являются структура классов (иерархия по номенклатуре) и структура объектов (иерархия по составу). Примерами иерархии классов являются простое и множественное наследование (один класс использует структурную или функциональную часть соответственно одного или нескольких других классов), а иерархии объектов – агрегация [6].

Контроль типов – правила использования объектов, не допускающие или ограничивающие взаимную замену объектов разных классов.

Параллелизм – свойство, отличающее активные объекты от пассивных. Реальная параллельность достигается только на многопроцессорных системах, а системы с одним процессором могут лишь имитировать параллельность за счет алгоритма разделения времени.

Персистентность – способность объекта преодолевать временные рамки (т.е. продолжать свое существование после исчезновения своего создателя) или пространственные пределы (т.е. выходить за пределы своего первоначального адресного пространства) [1].

Основные понятия объектно-ориентированного подхода – объект и класс.

Объект определяется как осязаемая реальность (tangible entity) - предмет или явление, имеющие четко определяемое поведение. Объект обладает состоянием, поведением и индивидуальностью; структура и поведение схожих объектов определяют общий для них класс. Термины "экземпляр класса" и "объект" являются эквивалентными. Состояние объекта характеризуется перечнем всех возможных (статических) свойств данного объекта и текущими значениями (динамическими) каждого из этих свойств. Поведение характеризует воздействие объекта на другие объекты и наоборот относительно изменения состояния этих объектов и передачи сообщений. Иначе говоря, поведение объекта полностью определяется его действиями. Индивидуальность - это свойства объекта, отличающие его от всех других объектов. Определенное воздействие одного объекта на другой с целью вызвать соответствующую реакцию называется операцией. Как правило, в объектных и объектно-ориентированных языках операции, выполняемые над данным объектом, называются методами и являются составной частью определения класса.

Класс — это множество объектов, связанных общностью структуры и поведения. Любой объект является экземпляром класса. Определение классов и объектов — одна из самых сложных задач объектно-ориентированного проектирования.

Следующую группу важных понятий объектного подхода составляют наследование и полиморфизм. Понятие полиморфизма может быть интерпретировано, как способность класса принадлежать более чем одному типу. Наследование означает построение новых классов на основе существующих с возможностью добавления или переопределения данных и методов.

Объектно-ориентированная система изначально строится с учетом ее эволюции. Наследование и полиморфизм обеспечивают возможность определения новой функциональности классов с помощью создания производных классов — потомков базовых классов. Потомки наследуют характеристики родительских классов без изменения их первоначального описания и добавляют при необходимости собственные структуры данных и методы. Определение производных классов, при котором задаются только различия или уточнения, в огромной степени экономит время и усилия при производстве и использовании спецификаций и программного кода.

Важным качеством объектного подхода является согласованность моделей деятельности организации и моделей проектируемой системы от стадии формирования требований до стадии реализации. Требование согласованности

моделей выполняется благодаря возможности применения абстрагирования, модульности, полиморфизма на всех стадиях разработки. Модели ранних стадий могут быть непосредственно подвергнуты сравнению с моделями реализации. По объектным моделям может быть прослежено отображение реальных сущностей моделируемой предметной области (организации) в объекты и классы информационной системы [6].

1.2. Унифицированный язык UML

Модель предметной области может служить основой общего языка для коммуникации в рамках проекта по разработке программного обеспечения. Модель - это набор понятий, существующих в головах у создателей проекта, вместе с названиями (терминами), отношениями и взаимосвязями, отражающими их понимание предмета. Термины и взаимосвязи образуют семантику языка, адаптированного к предметной области, но достаточно точного и для технических нужд разработки. Это та нить, которая соединяет модель с кодом и позволяет ей органично вплетаться в процесс разработки.[5]

Унифицированный язык моделирования UML – язык для определения, визуализации, конструирования и документирования артефактов программных систем, а также для моделирования экономических процессов и других не программных систем. [4]

Унифицированный язык моделирования (Unified Modeling Language, UML) – это универсальный язык визуального моделирования систем. Хотя чаще всего UML ассоциируется с моделированием объектно-ориентированных программных систем, он имеет намного более широкое применение благодаря свойственной ему расширяемости. UML объединил лучшие современные технические приемы моделирования и разработки программного обеспечения. По сути, язык UML был задуман так, чтобы его можно было реализовать посредством его же инструментальных средств. Фактически это признание того, что большие современные программные системы, как правило, нуждаются в инструментальной поддержке. UML диаграммы легко воспринимаются и при этом без труда генерируются компьютерами. Важно понимать, что UML не предлагает нам какой-либо методологии моделирования. Конечно, некоторые методические аспекты подразумеваются элементами, составляющими модель UML, но сам UML пре доставляет собой лишь визуальный синтаксис, который можно использовать для

создания моделей.

UML это не методология, это унифицированный язык визуального моделирования. Унифицированный процесс (Unified Process, UP) – это методология. Она указывает на исполнителей, действия и артефакты, которые необходимо использовать, осуществить или создать для моделирования программной системы.

UML не привязан к какой-либо конкретной методологии или жизненному циклу. На самом деле он может использоваться со всеми существующими методологиями. UP использует UML в качестве базового синтаксиса визуального моделирования. Следовательно, UP можно рассматривать как предпочтительный метод для UML, поскольку он лучше всего адаптирован к нему. Однако сам UML способен предоставить (и предоставляет) поддержку визуального моделирования другим методам. Конкретным примером сложившегося метода, использующего UML в качестве визуального синтаксиса, является метод OPEN (Object-oriented Process, Environment, and Notation – объектно-ориентированный процесс, среда и нотация).

Неизменная цель UML и UP – способствовать объединению всего лучшего в опыте разработки программного обеспечения последнего десятилетия. Для этого UML и UP унифицируют опыт предшествующих языков визуального моделирования и процессов разработки программного обеспечения наиболее оптимальным образом. [3]

Унификация UML носит не только исторический характер. UML прилагает усилия (и в основном успешно) в унификации нескольких разных областей:

1. Жизненный цикл разработки: UML предоставляет визуальный синтаксис для моделирования на протяжении всего жизненного цикла разработки программного обеспечения – от постановки требований до реализации.
2. Области приложений: UML используется для моделирования всех аспектов – от аппаратных встроенных систем реального времени до систем поддержки принятия решений.
3. Языки реализации и платформы: UML является независимым от языков и платформ. Естественно, он прекрасно поддерживает чистые объектно-ориентированные языки (Smalltalk, Java, C# и др.), но также эффективен и для гибридных объектно-ориентированных языков, таких как C++, и основанных на концепции объектов, таких как Visual Basic. UML также используется для создания моделей, реализуемых на не объектно-ориентированных языках программирования, таких как C.

4. Процессы разработки: хотя UP и его разновидности, вероятно, являются предпочтительными процессами разработки объектно-ориентированных систем, UML может поддерживать (и поддерживает) множество других процессов разработки ПО.
5. Собственные внутренние концепции: UML поистине стойко стремится сохранить последовательность и постоянство применения небольшого набора своих внутренних концепций. До сих пор это не всегда удавалось, но в этом направлении наблюдается заметный прогресс по сравнению с предыдущими попытками. [3]

Глава 2. Методологии, инструментальные средства реализации объектно-ориентированного подхода и их сравнение

В настоящее время объектно-ориентированный анализ является одним из наиболее интенсивно развивающихся направлений анализа и программирования, так как он позволяет разрабатывать хорошо структурированные и достаточно просто модифицируемые программные системы. Объектно-ориентированный анализ - метод исследования системы, основанный на объектной декомпозиции предметной области, представляемой в виде совокупности объектов, взаимодействующих между собой посредством передачи сообщений и принимающих определенные состояния. Объектно-ориентированный анализ связан с применением различных объектно-ориентированных методологий. К наиболее популярным относятся следующие:

1. методология OMT (Object Modeling Technique, Дж. Рум- бау — Rumbaugh);
2. методология Booch (Г. Буч);
3. методология OOSE (Object-Oriented Software Engineering, И. Якобсон - I. Jacobson);
4. методология ARIS (Architecture of Integrated Information Systems). [2]

Говорить о преимуществе той или иной системы/нотации не имеет смысла, пока не определены тип и рамки проекта КИС, а также основные задачи, которые должен решить данный проект. В общем случае модель бизнес-процесса должна давать ответы на следующие вопросы:

1. какие процедуры (функции, работы) необходимо выполнить для получения требуемого конечного результата?
2. в какой последовательности выполняются эти процедуры?
3. какие механизмы контроля и управления существуют в рамках рассматриваемого бизнес-процесса?
4. в чем заключаются роли и ответственности исполнителей процедур процесса?
5. какие входящие документы/информацию использует каждая процедура процесса?
6. какие исходящие документы/информацию генерирует процедура процесса?
7. какие ресурсы необходимы для выполнения каждой процедуры процесса?
8. какая документация регламентирует выполнение настоящей процедуры?
9. какие условия должны соблюдаться при выполнении процедуры?
10. какие параметры характеризуют выполнение процедур и процесса в целом?
11. существует ли последовательность процессов, минимизирующая затраты?
12. насколько процесс поддерживается/будет поддерживаться информационной системой?

Описание бизнес-процесса формируется с помощью нотации и инструментальной среды, позволяющих отразить все указанные выше аспекты. Только в этом случае модель бизнес-процесса окажется полезной для предприятия, так как ее можно будет анализировать и подвергнуть реорганизации.[3]

Инструменты проектирования обеспечивают проверку согласованности, ограничений и полноты, а также анализ моделей. Это позволяет разработчику свободно обозревать результаты анализа и проектирования. Например, взглянув на диаграмму компонентов, проектировщик может изучить конкретный механизм или локализовать все классы, связанные с определенным компонентом. Изучая диаграмму последовательностей, описывающую некий сценарий, разработчик может углубиться в иерархию наследования. Если исследуемый сценарий связан с активным объектом, то разработчик может использовать инструменты проектирования для идентификации процессора, через который проходит конкретный поток управления, а затем просмотреть анимацию конечного автомата, связанного с этим процессором. Избавившись от необходимости удерживать в памяти все детали, связанные с анализом и проектированием, разработчик, использующий такие инструменты, может сосредоточиться лишь на творческой работе. С другой стороны, инструмент проектирования не может подсказать проектировщику, что необходимо изобрести новый класс или упростить структуру какого-то класса. Все это может сделать лишь человек. Можно было бы попытаться

использовать для этого какую-нибудь экспертную систему, но для этого необходимы:

1. эксперт как в области объектно-ориентированного проектирования, так и в предметной области,
2. способность описывать эвристическую классификацию и большие общие знания. [1]

Одной из новейших и перспективных технологий реинжиниринга и анализа предметной области является технология ARIS.

Нотация ARIS eEPC (Extended Event Driven Process Chain) — расширенная нотация описания цепочки процесса, управляемого событиями. Каждый объект в системе ARIS Toolset имеет определенный набор атрибутов. Пользователю предлагается воспользоваться стандартными атрибутами для описания объектов или самостоятельно создать ограниченное количество пользовательских атрибутов. Бизнес-процесс в нотации ARIS eEPC представляет собой последовательность процедур, расположенных в порядке их выполнения. Используемые при построении модели символы логики позволяют отразить ветвление и слияние процедур. Следует отметить, что в нотации ARIS eEPC реальное время выполнения процедур не отражается визуально. Поэтому при создании моделей возможны ситуации, когда на исполнителя будет возложено одновременное выполнение нескольких задач. Для получения информации о реальном времени процессов необходимо использовать другие инструменты описания (например, диаграммы Ганта в системе MS Project). Таким образом, с помощью нотации ARIS eEPC можно описывать бизнес-процесс в виде потока последовательно выполняемых работ (процедур, функций). Диаграммы eEPC могут быть использованы для описания процессов управления рабочими потоками (WorkFlow). При создании диаграмм WorkFlow управляющие структуры моделируются более детально, включая описания:

1. временных характеристик начала и окончания работ;
2. исполнителей работ, их роли и функции;
3. прикладных автоматизированных систем, используемых в работах.

Нотация ARIS Organizational Chart является одной из основных нотаций ARIS и предназначена для построения схем организационной структуры предприятия. Как правило, эта модель строится в начале проекта, в ней отражаются существующие подразделения предприятия в виде иерархической структуры. Заложенные в

нотацию типы связей позволяют отразить различные виды отношений, существующих между объектами организационной структуры. [2]

Методология ARIS рассматривает предприятие как совокупность четырех взглядов: взгляд на организационную структуру, взгляд на структуру функций, взгляд на структуру данных, взгляд на структуру процессов. При этом каждый из этих взглядов разделяется еще на три подуровня: описание требований, описание спецификации, описание внедрения. Таким образом, ARIS предлагает рассматривать организацию с позиции 12 аспектов, отображающих разные взгляды на предприятие, а также разную глубину этих взглядов.

Положительные стороны:

1. "Могучая" репрезентативная графика.
2. Наличие большого числа стандартных объектов для описания бизнес процессов.
3. Наличие инструмента имитационного моделирования.
4. Наличие внутреннего языка управления ARIS-Basic.
5. Возможность тестирования проекта на соответствие требования стандарта качества ISO 9000.

Отрицательные стороны:

1. Невозможность генерации каких-либо кодов или баз данных.
2. Потребует очень большого времени (возможно до 5 мес.) на обучение персонала.
3. Авторитет разработчика ПО ничем пока не подтвержден.

Инструментальные средства для разработки информационных систем, в том числе и для процесса реинжиниринга, созданные фирмой Computer Associates, получили название AllFusion Modeling Suite 7.2. В состав интегрального пакета инструментальных средств AllFusion Modeling Suite 7.2 входят пять программных продуктов: AllFusion Process Modeler 7.2 (BPwin 4.0); AllFusion ERwin Data Modeler 7.2 (ERwin 4.0); AllFusion Data Model Validator 7.2 (ERwin Examiner); AllFusion Model Manager 7.2 (Model Mart); AllFusion Component Modeler 7.2 (Paradigm Plus). В скобках указано название программного продукта предыдущей версии.

Нотация IDEF0 была разработана на основе методологии структурного анализа и проектирования SADT и успешно применяется во многих проектах с целью создания функциональных моделей деятельности предприятия. Работы на

диаграмме IDEF0 располагаются в соответствии с временем начала работы - слева направо. В левом верхнем углу размещается работа, выполняемая первой. Одной из целей описания бизнес-процессов в нотации IDEF0 является анализ организации бизнес-процессов. Если в построенной модели отсутствуют обратные связи, работы не имеют выхода или управления и работы дублируются, то эти работы требуют совершенствования организации процесса.

Нотация IDEF3 была разработана с целью описания информационных потоков (WorkFlow), для которых важно отразить логическую последовательность выполнения процедур с учетом временных показателей. Диаграммы WorkFlow применяются в моделировании бизнес-процессов для анализа завершенности процедур обработки информации. С их помощью описывают сценарии действий сотрудников организации, которые необходимо выполнять за конечное время. Каждый сценарий сопровождается описанием процесса и используется для документирования функций.

Нотация DFD предназначена для описания потоков данных, системы документооборота и процедур обработки информации на предприятии/в компании. Диаграммы DFD можно использовать как дополнение к модели IDEF0 для более детального отображения операций документооборота в корпоративных системах обработки информации. Наличие объектов «хранилище данных» и двунаправленных стрелок позволяет наиболее эффективно описать требования к информационной системе. [2]

Организационные диаграммы AllFusion Process Modeler являются аналогом организационных диаграмм ARIS EPS и предназначены для описания иерархии в организациях. Для создания организационной диаграммы необходимо предварительно внести в словари следующую информацию:

1. изображения (bitmaps), если на организационной диаграмме предполагается использовать иконки;
2. группы ролей — могут соответствовать структурным подразделениям;
3. роли — могут соответствовать должности;
4. ресурсы — могут соответствовать фамилии конкретной персоны.

После формирования словарей нотации AllFusion Process Modeler позволяет создать иерархию, включающую группы ролей, роли и ресурсы. В диаграммах IDEFO, IDEF3, DFD каждой работе может быть назначен исполнитель-ресурс из словаря ресурсов. Нотация AllFusion Process Modeler позволяет создавать диаграммы SwimLane —

разновидность диаграмм IDEF3, на которых в виде полос отображаются зоны ответственности служащих предприятия, возникающие при выполнении служащими технологических операций. [2]

AllFusion Process Modeler - средство концептуального моделирования БД, использующее стандарт IDEF1X. AllFusion Process Modeler 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. Методология IDEF0, являющаяся официальным федеральным стандартом США, представляет собой совокупность методов, правил и процедур, предназначенных для построения функциональной модели объекта какой-либо предметной области. Функциональная модель IDEF0 отображает функциональную структуру объекта, т.е. производимые им действия и связи между этими действиями. Методология IDEF может использоваться для моделирования широкого круга систем и определения требований и функций, а затем для разработки системы, которая удовлетворяет этим требованиям и реализует эти функции. Для уже существующих систем IDEF может быть использована для анализа функций, выполняемых системой, а также для указания механизмов, посредством которых они осуществляются.

Положительные стороны:

1. Авторитетность (множество положительных отзывов).
2. "Изобразительные" средства системы соответствуют федеральному стандарту США IDEF на моделирование организационных процессов.
3. Распространенность (99,9% проектов организационного реинжиниринга исполняются с использованием стандарта IDEF).
4. Возможность генерации исполняемого кода по разработанной модели информационной системы.
5. Пожалуй одно из лучших средств проектирования баз данных.

Отрицательные стороны:

1. Репрезентативные свойства низки.
2. Отсутствие стандартных объектов для описания бизнес процессов.
3. Довольно узкие возможности для проведения экономического анализа.

В ряде случаев для реинжиниринга бизнес-процессов и моделирования предметной области используется технология Rational Rose. Функциональная модель бизнес-процессов предметной области, построенная средствами Rational Rose в нотации

UML, представляется в виде диаграммы вариантов использования (use case diagram). Данная диаграмма графически отображает подмножество активных субъектов, взаимодействующих с системой посредством тех или иных вариантов использования. При проектировании системы в первую очередь создается основная диаграмма, представляющая множество пользователей (активных субъектов) и ключевые функции (варианты использования) системы. Диаграммы вариантов использования позволяют формализовать процесс постановки целей и задач проекта. Данный подход к построению диаграмм вошел в стандарт языка UML. Варианты использования характеризуются рядом свойств:

1. охватывают некоторую очевидную для пользователя функцию;
2. могут быть различного масштаба;
3. решают некоторую дискретную задачу пользователя.

В простейшем случае вариант использования - это функция, которую должна реализовывать проектируемая система. Основными элементами диаграммы вариантов использования являются действующие лица (активные субъекты), варианты использования и отношения между ними. Действующее лицо - это роль, которую пользователь играет по отношению к системе. На этапе анализа процессов предметной области средствами Rational Rose на базе нотаций UML используются диаграммы действий, которые позволяют отображать динамические характеристики системы:

1. функции управления процессом;
2. участки процесса, которые могут выполняться параллельно;
3. синхронность начала и окончания действий;
4. альтернативные пути достижения целей.

Диаграммы действий могут охватывать от одного до нескольких вариантов использования. С каждым вариантом использования связан определенный поток событий, происходящих по мере выполнения соответствующих функций системы. При описании потока событий важно описать действия, которые необходимо осуществить, а не то, как выполняются эти действия. По мере реализации проекта детализируются процессы предметной области и диаграммы действий дополняются новыми объектами, позволяющими проиллюстрировать особенности реализации отдельных операций. Нотация диаграмм действий Rational Rose для построения диаграмм WorkFlow. Используя нотацию диаграмм действий Rational Rose, можно построить диаграммы рабочих потоков (WorkFlow). Моделирование диаграмм рабочих потоков применяется с целью:

1. понять структуру организации и динамику процессов, проходящих в ней;
2. сформировать общую точку зрения у сотрудников, конечных пользователей и разработчиков на процессы, которые проходят в организации;
3. определить требования к системе с учетом организационной структуры.

Процесс функционирования системы может содержать стадии, которые могут выполняться параллельно. Полосы синхронизации на диаграммах позволяют показать, какие действия допускается выполнять параллельно или подлежит подвергнуть логическому объединению. Для того чтобы описать на диаграмме действий, кто ответствен за выполнение соответствующей последовательности действий, используется такой прием, как разделение диаграммы на зоны (SwimLane). Каждая зона на диаграмме действий связана с определенным активным субъектом, ответственным за выполнение работы или процедуры.

Таким образом, процесс анализа предметной области средствами Rational Rose можно представить как поуровневый переход от наиболее общих моделей и представлений концептуального уровня к более частным и детальным представлениям логического и физического уровней. При этом на каждом из этапов объектно-ориентированного анализа процессов данные модели последовательно дополняются все большим количеством деталей, что позволяет адекватно отражать различные аспекты конкретной реализации сложной системы. [2]

Rational Rose - 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. 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.

Положительные стороны:

1. В наибольшей степени подходит для разработки крупных информационных систем.
2. Реализует большую часть функций ARIS и ERwin/BPwin.

3. Мощные функциональные возможности по генерации исполняемых кодов.

Отрицательные стороны:

1. Политика разработчика непрозрачна.
2. Отсутствие стандартных объектов для описания бизнес процессов.
3. Очень противоречивые отзывы пользователей.

Таким образом, каждая из рассматриваемых систем (ARIS Toolset, AllFusion Process Modeler и Rational Rose) имеет преимущества и недостатки, которые в зависимости от решаемых задач могут как усиливаться, так и ослабевать. Следует отметить, что для хранения моделей в системах ARIS используется объектная СУБД и под каждый проект создается новая база данных. В системах ARIS предоставляются различные функции по администрированию базы данных, управлению доступом, интеграцией данных и т.д. В AllFusion Process Modeler данные модели хранятся в файле. Для групповой работы над большими проектами предусмотрено размещение моделей в репозитарии Model Manager, который позволяет хранить модели, созданные в AllFusion Process Modeler и AllFusion ERwin Data Modeler. Репозитарий Model Manager поддерживает реляционные СУБД Oracle, Informix, MS SQLServer, Sybase. В нем предусмотрено администрирование, разграничение прав доступа до уровня объекта модели, сравнение версий моделей, слияние моделей и т.д. [2]

Сравнительный функциональный анализ представлен в таблице 1:

Таблица 1

Сравнительный функциональный анализ инструментальных средств ARIS, AllFusion PM и Rational Rose

	Функции, свойства	**ARIS**	**AllFusion PM**	**Rational Rose**
1	Моделирование организационных функций и процессов	+	+	+
2	Разработка технического задания	+	+/-	+/-

3	Функционально-стоимостной анализ	+	+	+/-
4	Оптимизация бизнес процессов	+	-	-
5	Имитационное моделирование, событийно-управляемое моделирование	+	+/-	-
6	Генерация кода приложения	-	+	+/-
7	Оформление проектной документации; генерация технологических инструкций для рабочих мест	+	+/-	+
8	Хранение моделей деятельности предприятий	+	+/-	+/-
9	Создание концептуальных и физических моделей структуры базы данных	+/-	+	+
10	Генерация программного кода, SQL-сценариев для создания структуры базы данных.	-	+	+/-
11	Стандартное представление основных бизнес процессов (более 100 типов)	+	-	-
12	Ведение библиотеки типовых бизнес моделей	+	+/-	+/-
13	Групповая работа над проектом	+	+	+
14	Выдача встроенных отчетов по стандарту ISO9000	+	-	-

«+» - да

«+/-» - частичная реализация, требующая доработки иными инструментальными средствами

«-» - нет

ЗАКЛЮЧЕНИЕ

В настоящее время информатизация жизни происходит с огромной скоростью, и важно понимать, как уменьшать все возможные затраты на данный процесс. Объектно-ориентированный подход является в данный момент наиболее востребованным.

При написании курсового проекта нами была изучена специальная литература, включающая в себя статьи и учебники по информационным технологиям, описаны теоретические аспекты ООП, раскрыты ключевые понятия исследования и сравнение инструментальных средств.

Как можно заметить, универсального программного обеспечения не существует, каждый продукт имеет свои плюсы и минусы, слабые и сильные стороны, поэтому вопрос выбора инструментального средства остается актуальным и в наше время. Этим обусловлена необходимость с полной серьезностью подходить к анализу задач и требований к информационной системе, что, в свою очередь, позволит более точно определить наиболее подходящее программное средство и, тем самым, уменьшить все возможные затраты на всех этапах жизненного цикла информационной системы.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Буч Гради Объектно-ориентированный анализ и проектирование с примерами приложений, 3-е изд.: пер. с англ. – М.: ООО «И.Д. Вильямс», 2008. – 720 с.
2. Сатунина А.Е. Управление проектом корпоративной информационной системы предприятия: учебн. пособие/ А.Е. Сатунина, Л.А. Сысоева. – М.: Финансы и статистика; ИНФРА-М, 2009. – 352 с.

3. Арлоу Д., Нейштадт И. UML 2 и Унифицированный процесс. Практический объектно-ориентированный анализ и проектирование, 2е издание. – Пер. с англ. – СПб: Символ Плюс, 2007. – 624 с.
4. Крэг Ларман Применение UML и шаблонов проектирования. 2-е издание: Пер. с англ. – М.: Издательский дом «Вильямс», 2004. – 624 с.
5. Эванс Эрик. Предметно-ориентированное проектирование (DDD): структуризация сложных программных систем: Пер. с англ. - М.: 000 "И.Д. Вильямс", 2011. - 448 с.
6. Вендров А. М. Проектирование программного обеспечения экономических информационных систем: Учебник. — М.: Финансы и статистика, 2002. - 352 с