Содержание:

ВВЕДЕНИЕ

Экономическая информация стала предметом труда сравнительно недавно. До конца XX века самыми значимыми предметами труда являлись материальные предметы. К концу столетия ведущим предметом труда становится информация. Объемы информационных ресурсов увеличиваются в геометрической прогрессии. Существуют математические методики, позволяющие измерять процесс накопления информации во времени и дающие возможность оценить скорость данного процесса. В частности, можно определить, с какой скоростью происходит удвоение объемов информации. Если до девятнадцатого века удвоение объема происходило примерно каждые 50 лет, то в период до 1950 г. этот процесс занимал 10 лет, до 1970 г. – около пяти лет, а к настоящему времени это происходит каждый год. [4] Подобная динамика требует вовлечения в сферу работы с информацией все больших трудовых и материальных ресурсов.

На развитие информационных технологий оказывают влияние следующие факторы:

- Построение инфраструктуры предприятий на основе информационных технологий.

- Увеличение объемов инвестирования в информационные технологии, с целью повышения адаптируемости предприятия к изменяющимся условиям.

- Увеличение количества и повышение интенсивности взаимосвязей между отдельными объектами внутри предприятий и между ними.

- Увеличение количества информационных продуктов и их разнообразия; развитие рынка информационных услуг.

Понятие информационной технологии подразумевает собой комплекс из трех основных элементов: хранения, обработки и передачи информации. Для обеспечения данных процессов применяются информационные системы.

Под методами информационных технологий понимают способы моделирования, разработки механизмов обработки данных. Средствами информационных технологий являются алгоритмы анализа данных, математические методы, программное обеспечение для моделирования процессов и проектирования систем [11].

Объектом управления в экономических информационных системах является комплекс взаимосвязанных структурных элементов предприятия. К данным подразделениям относятся: менеджмент, финансовый, обеспечивающий, производственный и прочие отделы. В составе системы они выполняют следующие задачи:

1.разработка планов

2.контроль выполнения

3.проведение учета

4.регуляторные функции

5.проведение анализа деятельности

Целью данной работы является анализ и оценка средств объектно-ориентированного анализа и проектирования информационных систем.

В связи с обозначенной целью в работе решаются следующие задачи:

1. Ознакомиться с понятием информационной системы.
2. Изучить особенности экономических информационных систем
3. Рассмотреть сущность объектно-ориентированного подхода к проектированию информационных систем.
4. Ознакомиться со средствами языка UML
5. Ознакомиться со средствами реализации объектно-ориентированного анализа и проектирования экономической информационной системы.

ГЛАВА.1 ОБЪЕКТНО-ОРИЕНТИРОВАННАЯ ПАРАДИГМА АНАЛИЗА И ПРОЕКТИРОВАНИЯ

ИНФОРМАЦИОННЫХ СИСТЕМ

1.1.Понятие информационной системы.

Согласно определению, в соответствии со стандартом ISO/IEC 2382:2015, под информационной системой понимают систему, предназначенную для хранения, поиска и обработки информации, и соответствующие организационные ресурсы (человеческие, технические, финансовые и т. д.), которые обеспечивают и распространяют информацию. [13]

Российский стандарт ГОСТ РВ 51987 определяет информационную систему как автоматизированную систему, результатом функционирования которой является представление выходной информации для последующего использования. [5]

Существует несколько классификаций информационных систем. В частности, различают:

По архитектуре:

1.Локальные информационные системы, все компоненты которых реализуются с помощью одного компьютера.

2.Распределенные информационные системы, в которых составляющие распределены по нескольким компьютерам.

По степени автоматизации:

1.Автоматизированные информационные системы, работа которых требует вмешательства персонала.

2.Автоматические информационные системы, работа которых полностью автономна, либо требует эпизодического вмешательства персонала.

По характеру обработки данных:

1.Информационно-поисковые информационные системы, в которых отсутствуют сложные алгоритмы анализа данных, а основной задачей является поиск и предоставление информации в удобной для пользователя форме.

2.Информационные системы обработки данных, в которых происходит анализ данных по сложным алгоритмам (к данной категории относятся системы поддержки принятия решений).

По сфере применения информационные системы разделяют в зависимости от прикладного характера выполняемых задач. В качестве наиболее распространенных типов можно привести следующие:

1.Экономическая информационная система - комплекс организационных, программных, технических и информационных ресурсов, организованных в единую систему с целью сбора, хранения, обработки и выдачи требуемой информации, предназначенной для выполнения функций управления.

2.Медицинская информационная система – совокупность методологических, технических, программных, информационных, организационных и правовых средств, поддерживающих процессы функционирования лечебно-профилактического учреждения.

3.Географическая информационная система - комплекс информационных технических, программных средств, обеспечивающих ввод, хранение, обработку, математико-картографическое моделирование и образное интегрированное представление географических и связанных с ними атрибутивных данных для решения проблем территориального планирования и управления. [9]

1.2.Экономические информационные системы.

В общем случае под информацией принято понимать совокупность сведений, необходимых для принятия какого-либо решения.

Информационные ресурсы являются неотъемлемой частью современной экономики. С точки зрения конкретного предприятия, информация - это данные в определенном формате, значимые для принятия решения, определенным образом собираемые, хранящиеся и обновляемые.

Информацию внутри экономических систем подразделяют на непосредственно экономическую и техническую.

Экономическая информация ассоциирована с управлением коллективами, занятыми производством и управлением, техническая – с управлением техническими объектами. Экономическая информация связана с процессами производства, распределения, обмена и потребления материальных благ и услуг, обусловленных общественным производством. К числу ее свойств относится: большие объемы, многократное применение, выполнение сравнительно несложных расчетных операций. Для нее характерна определенная структура, минимальной структурной единицей является показатель. [10]

1.3.Методы анализа и проектирования экономической информационной системы.

Проектирование и анализ экономических информационных систем – трудоемкий процесс, требующий высококвалифицированных специалистов и значительного количества временных ресурсов. Следует помнить, что на этапе разработки информационной системы требования со стороны пользователей могут видоизменяться и корректироваться, что в свою очередь усложняет и продлевает этот процесс.

Под проектированием экономических информационных систем следует понимать процесс создания технической документации, ассоциированный с развертыванием системы получения и преобразования исходной информации в результирующую. Документ, являющийся результатом проектирования, называется проектом. Основной задачей проектирования является создание математического, информационного, программного, правового и организационного обеспечения. [9]

При проектировании экономической информационной системы используется ряд принципов:

1. Системность. Он заключается в том, что все явления внутри системы рассматриваются с позиции взаимосвязанности. При этом основным этапам организации системы относятся:

- точное определение цели;

-формирование системных требований (уточнение границ объектов);

-формирование функциональных подсистем, определение их структуры;

- поиск и уточнение связей между подсистемами;

- Организация функционирования и развития системы.

2. Непрерывность развития — подразумевает, что при организации информационной системы усматривается возможность оперативного и экономичного изменения информационной системы.

3. Совместимость — регламентирует взаимодействие отдельных информационных систем между собой.

4. Стандартизация регламентирует применение определенных стандартов при развертывании информационных систем.

5. Эффективность – отношение между предполагаемыми затратами на развертывание системы и предполагаемой эффективностью. [11]

Выделяют две ведущих методики проектирования информационных систем: объектное-ориентированное и структурное проектирование.

Основой структурного метода является декомпозиция системы на отдельные подсистемы в виде автоматизируемых функций, который в дальнейшем разделяются на подфункции, подзадачи и так далее. Процедура декомпозиции выполняется до конкретных процедур. При этом обеспечивается целостное представление системы, при котором сохраняются взаимосвязи между компонентами.

Объектно-ориентированное метод подразумевает разбиение системы на отдельные объекты. Под объектом понимается действительно существующий элемент, исполняющий важную функцию в пределах рассматриваемой предметной области. Объект обладает следующими характеристиками: структура, модель поведения, состояние. [14]

Состояние объекта характеризуется списком всех доступных атрибутов и их существующими показателями.

Невозможно проектировать информационную систему на основе двух методик одновременно. Необходимо проводить разбиение либо по функциональному признаку, либо на основе объектно-ориентированного подхода. В дальнейшем

предпринимается попытка оценки системы по альтернативной методике. [8]

Для объектно-ориентированной методики характерно наличие следующих преимуществ перед структурным:

- большая гибкость в проектировании и эксплуатации, что упрощает модернизацию в дальнейшем.

- разбиение на основе объектного подхода сокращает размер кода вследствие повторного использования общих методов.

1.4.Сущность объектно-ориентированного подхода к проектированию информационных систем.

Объектно-ориентированный способ проектирования информационных систем основан на модели объектно-ориентированного программирования. Последний стал популярен в связи с все более широким применением персональных компьютеров в малом и среднем бизнесе, что вывело на первый план задачи анализа данных. Объектно-ориентированное программирование регламентирует структуру программы как комплекс объектов, обладающих набором атрибутов и применимых методов, и является альтернативой процедурному методу программирования, при котором программа является набором последовательно запускаемых процедур.

Объектно-ориентированная парадигма — это метод разработки приложений, подразумевающие разбиение приложения на комплекс объектов, обладающих определенной независимостью. Завершенное приложение выглядит как композиция объектов. Одним из преимуществ данного подхода является возможность многократного применения компонентов в разных системах, что значительно сокращает время, необходимое для разработки и развертывания новой системы. [2]

Минимизация риска при использовании данной парадигмы реализуется за счет применения принципа итерационной разработки, связанного со спиральной моделью жизненного цикла системы. При этом создание системы проходит ряд ступеней (именуемых итерациями). Каждый этап включает в себя разработку части программы или обновленной версии системы и содержит ключевые моменты:

определений требований, анализ, проектирование, и так далее. В связи с тем, что тесты предусмотрены для каждой ступени, минимизация риска достигается уже в начале жизненного цикла системы.

Данные преимущества достигаются при разработке системы в соответствии с основными принципами объектно-ориентированной парадигмы:

- Инкапсуляция — свойство объекта, предусматривающее ограничения на внесение изменений в объект за счет сокрытия информации о его внутренней структуре. Взаимодействие с объектом возможно лишь посредством его методов и атрибутов. Методы и атрибуты объединяются под определением интерфейса объекта.
- Наследование — механизм, реализующий новые объекты на основе уже имеющихся. Данный механизм позволяет использовать атрибуты и методы одного объекта внутри другого объекта; при этом допускается их частичная или полная модификация. Объект, созданные по данному механизму носит определение потомка.
- Полиморфизм — свойство объектов, позволяющее реализовывать различные действия в ответ на одни и те же вызовы, в зависимости от разных внешних условий, сопровождающих данный вызов. Как правило под этим подразумевается способность подбирать операции в зависимости от типа данных, сопровождающих вызов из вне. [3]

Базовым элементом данной парадигмы является объект. Объект отражает некую сущность из окружающей действительности, или концепцию с определенной структурой. В частности, к объектам может относиться конкретная книга, либо (как пример концептуального) денежная транзакция.

Объект включает в себя не все свойства описываемого им предмета (понятия), а лишь те, что имеют значение для развертываемой информационной системы. То есть устройство объекта примитивнее, чем описываемого им сущности или концепции. Следует помнить, что объект является формальной конструкцией. Это дает возможность определить формальные зависимости между объектами, а также формальные действия.

Любой объект характеризуется следующими свойствами: индивидуальность, состояние, поведение.

- Состояние — одна из кондиций, возможных для данного объекта, способная изменяться с течением времени; зависит от атрибутов данного объекта и

взаимных вызовов объектов.
- Поведение — регламентирует реакцию объекта на вызовы из вне и его возможные действия. Данное свойство обеспечивается с помощью списка методов данного объекта.
- Индивидуальность определяет уникальность данного объекта, вне зависимости от совпадения его состояния с другим объектом. [2]

Одной из основных сущностей объектно-ориентированной парадигмы является класс. Класс — это описание объекта, включающее его атрибуты и методы, способы взаимодействия с иными объектами, на основе которого создаются новые объекты одного и того же типа. Все объекты являются экземплярами того или иного (единственного) класса. Принадлежность одного объекта к нескольким классам недопустима.

Если традиционные методы разработки предусматривали нахождение данных в базе данных, а их обработка доверялась непосредственно приложению, то при объектно-ориентированном подходе данные и применимые к ним методы хранятся внутри одной сущности, определяемой как объект.

Известным недостатком объектно-ориентированной парадигмы являются высокие затраты на начальном этапе. Для получения финансового преимущества необходимо выполнение нескольких проектов в данной предметной области, что позволяет накопить библиотеку компонентов для многократного применения.

Следует так же помнить, что объектно-ориентированная парадигма хуже воспринимается пользователями и в первую очередь обеспечивает преимущества для разработчиков. Для пользователей информационной системы более доступен подход на основе функционально-ориентированной декомпозиции.[1]

1.5.Объектно-ориентированный анализ.

При развертывании информационных систем больших масштабов была выявлена необходимость анализа предметной области, предшествующего разработке программного обеспечения. Процесс развертывания базы данных значительно отличается от разработки кода с целью вычисления задачи. Например, при развертывании баз данных существует необходимость в превентивном создании модели, отражающей взаимные связи внутри предметной области и особенности ее

структуры.

Предметная область — это та часть окружающего мира, которая непосредственно относится к функционированию информационной системы. Под предметной областью принято понимать только те объекты и связи, которые требуются для формализации требований и создания решения поставленной задачи.

Определение базовых элементов предметной области, их формализация, является одной из сложнейших задач на этапе проектирования информационной системы. Проблематичность определяется тем, что правила, применимые для этой цели, носят, как правило, неформальный характер. Кроме того, данная задача должна выполняться в сотрудничестве с экспертами, хорошо знакомыми с предметной областью.

Объектно-ориентированный анализ и проектирование (ООАП) — методика разработки информационных систем, в основе которой находится принцип объектно-ориентированного представления предметной области в форме объектов, являющихся экземплярами конкретных классов. [15]

Технология ООАП в значительной степени связана с системами автоматизированной разработки программного обеспечения (Computer Aided Software Engeniring — CASE). Изначально к CASE-инструментам было настороженное отношение. Впоследствии возникли как восхищенные отклики, так и критические замечания в их адрес. Для подобных противоречивых оценок был ряд оснований. Одно из них состоит в том, что первые CASE-средства являлись примитивной «надстройкой» над системами управления баз данных. Применение визуализации в процессе развертывания базы данных значительно ускоряет данный процесс, однако это не является решением при разработки программного обеспечения иных классов.

В основе другой причины лежит графическая нотация, реализованная в CASE-инструментарии. Предложения по созданию единого синтаксиса для визуального представления концепции структуры базы данных, по типу строгого синтаксиса в языках программирования, получили неоднозначную реакцию. Это обеспечило «радушный прием» сообществом разработчиков стандартизованного языка моделирования UML. [14]

В пределах объектно-ориентированного анализа и проектирования были определены три графические нотации:

- диаграммы «сущность-связь»
- диаграммы функционального моделирования
- диаграммы потоков данных

Диаграммы «сущность — связь» обеспечивают графическое представление моделей данных развертываемой информационной системы и предоставляют список стандартизованных обозначений описания данных и взаимосвязей между ними. Данный вид диаграмм позволяет представить элементы концептуальной модели данных и их взаимные связи.

К базовым понятиям этой нотации, как следует из ее названия, относятся понятия сущности и связи. В данном случае под сущностью принято понимать множество реальных объектов или концепций, каждого из которых характерны одинаковые свойства. При этом произвольный объект может являться экземпляром только одного класса (сущности, в данном контексте), должен иметь личный идентификатор и быть отличным от иных экземпляров этого класса.

Связь в данном случае определена как отношение, связывающее различные сущности. К примерам подобных связей можно отнести отношения между родственниками, или иерархические отношения между сотрудниками предприятия. Также вариантом связей является отношение «обладать свойством». Для графического изображения связей используется ромб с идентификатором данной связи.

При построении графической модели данных добиваются того, чтобы связи между сущностями описывали как семантику данного отношения, так и дополнительные аспекты, отражающие обязательность связи и кратность экземпляров, принимающих участие в данных отношениях. Нотация диаграмм «ERD» обладает реализацией в различных инструментальных средствах разработки. Ограниченность данных диаграмм проявляется на этапе перевода концептуальной модели в более детальное представление моделируемой информационной системы, в котором, помимо статических связей необходимо включать данные о работе ее отдельных элементов. [17]

ГЛАВА.2 СРЕДСТВА РЕАЛИЗАЦИИ ИНФОРМАЦИОННЫХ СИСТЕМ С ИСПОЛЬЗОВАНИЕМ

ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПОДХОДА

2.1.Язык проектирования UML.

Появление отдельных языков объектно-ориентированного проектирования было отмечено в середине 70-х годов. В то время отдельные авторы предлагали множество методик проектирования. К началу девяностых годов существовало около пятидесяти языков ООАП, что вызывало у пользователей определенные сложности с выбором конкретной методики, поскольку каждая из них, наравне с преимуществами, обладала и своими недостаткам. Появление к этому времени таких стандартов как IDEF0 и IDEF1X не решало проблему унификации методик в полной мере.

Однако ряд методов к этому времени приобрел несколько большую популярность, в сравнении с остальными. Такими методами стали:

- Booch (автор — Гриди Буч)
- Object Modeling Technique (автор — Джеймс Румбах)
- Object-Oriented Software Engineering (автор — Айвар Джекобсон)

Особенностью данных методов являлась специализация на конкретном уровне анализа и проектирования. В частности, методика Айвара Джекобсона имела инструменты визуализации вариативности применения, что важно на этапе анализа требований при развертывании информационных систем.

Подход Джеймса Румбаха проявлял максимальную эффективность на этапе анализа процессов обработки информации в приложениях.

В октябре 1994 года Буч и Румбах начали работу по интеграции и унификации собственных подходов. При этом проводились исследования всех доступных на тот момент средств объектно-ориентированного проектирования и анализа с целью выявления их преимуществ. Впервые унифицированный метод был представлен в 1995 году и имел к тому моменту версию 0.8. Позже в состав рабочей группы вошел Джекобсон, его методика Object-Oriented Software Engineering так же была интегрирована в Unified Method.

К этому моменту Unified Method Language получил поддержку консорциума Object Managment Group, который был создан раннее с целью разработки стандартов в области объектных и компонентных технологий CORBA. В составе данного консорциума была создана группа разработчиков, куда вошли вышеуказанные авторы методик, и которая взяла на себя задачу по дальнейшему развитию UML.

Позже был создан так называемый консорциум партнеров UML, куда вошли такие компании как DEC, IBM, HP, Oracle, Unisys, Rational Software. С их участием версия UML 1.0 была создана к январю 1997 года. Данная версия имела хорошую определенность, предоставляла средства, обеспечивающие высокую выразительность, обладала гибкостью, позволяющей решать задачи широкого спектра. [1]

История разработки и последующего развития языка UML графически представлена на рисунке 1.

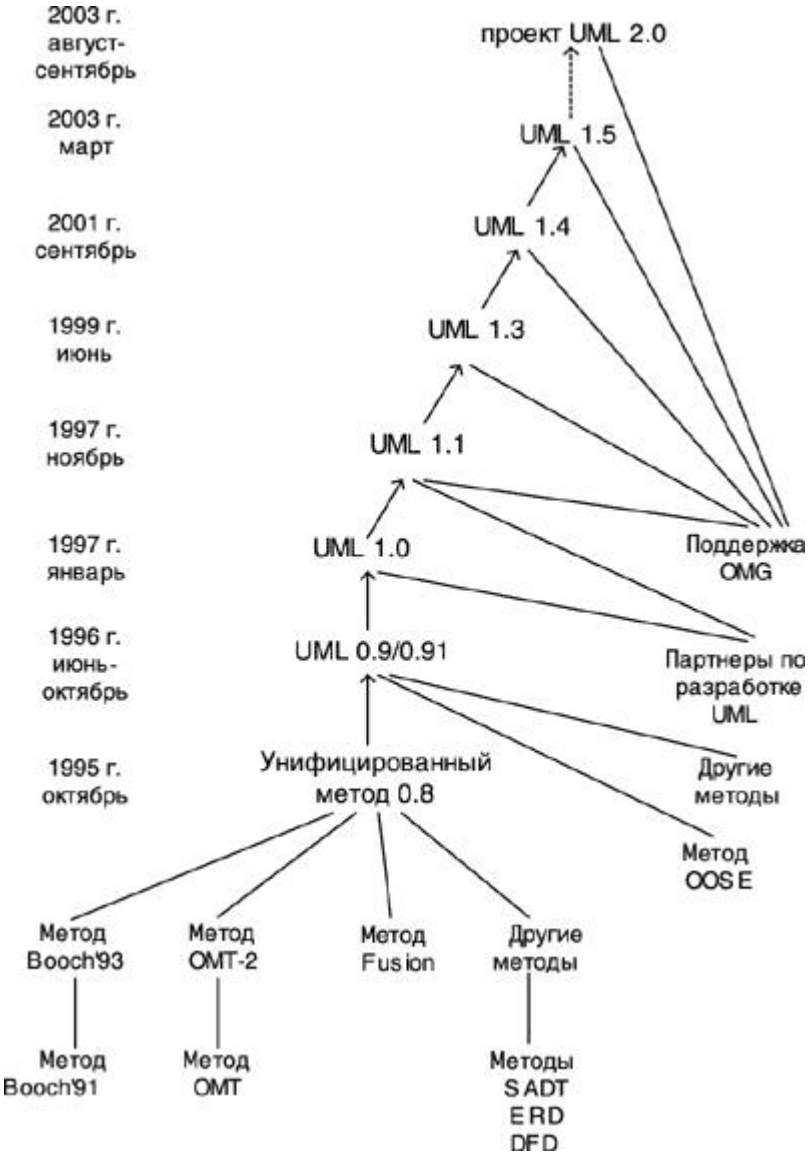

Рисунок 1. История развития UML

На сегодняшний день разработка и стандартизация языка UML сосредоточена в пределах консорциума OMG. При этом отмечается, что данный язык открыт для всех предложений по его дальнейшему развитию. Новейшая версия UML — 2.5; представлена в июне 2015 года.

UML 2.4.1 принят в качестве международного стандарта ISO/IEC 19505-1, 19505-2.

На сегодняшний день на рынке программного обеспечения существует множество CASE-средств, поддерживающих UML и предоставляющих генерацию программ на популярных языках программирования, таких как Java, C++, Delphi и т. д.

Заинтересованность разработчиков в UML возрастает с течением времени. Он является базисом в CASE-средствах визуального моделирования. Кроме того, он

применяется не только для объектно-ориентированного анализа и проектирования, но и для документирования бизнес-процессов. [8]

UML является языком визуального моделирования общего назначения, обеспечивающий спецификацию, проектирование, документирование и визуализацию элементов информационных систем, приложений и иных систем. Это эффективный инструмент моделирования, который может с успехом применяться для создания моделей (таких, как логические, концептуальные, графические), а также различных целевых систем.

Методика объектно-ориентированного проектирования и анализа описывает достаточно полную модель сложной системы как некоторое количество сущностей, имеющих взаимные связи, каждая из которых максимально полно описывает компонент поведения или структуры системы. Максимально общими сущностями сложной системы являются статическое и динамическое. Они делятся на более частные.

С точки зрения принципа иерархического развертывания моделей сложных систем процесс создания модели на рассматривается на различных уровнях как абстрагирования, так и детализации в пределах определенных представлений.

Уровень представления — метод формирования модели с определенным уровнем абстракции, представляющий срез архитектуры модели в горизонтальной плоскости, при этом ее срез в вертикальной плоскости является разбиением.

Исходная модель при этом обладает максимально общим представлением и принадлежит к уровню концепции. Подобная модель, носящая определение концептуальное, формируется на первых этапах проектирования, при этом не обязательно включает многие детали и нюансы моделируемой системы. Дальнейшие модели делают концептуальную модель более конкретной, внося в нее представления логического и физического уровня. [2]

В общем случае объектно-ориентированные анализ и проектирование представляется как процесс формирования наиболее детализированных представлений на физическом и логическом уровнях, начиная от уровня концептуальной модели. Данные процесс включает в себя последовательное дополнение данных модели возрастающим количеством деталей на каждом последующем уровне, что в итоге дает возможность максимально адекватного отражения различных аспектов той или иной сложной модели.

Одним из основных средств языка является пакет, применяющийся для объединения в группы компонентов модели. Непосредственно компоненты, включая произвольные представления, относящиеся к одному пакет, являются единым целым. Все многообразие компонентов графической нотации Unified Method Language объединены в пакеты.

2.2.Пакеты в языке UML.

Пакет — механизм организации компонентов модели в упорядоченное множество на основе принципа декомпозиции модели сложной системы и предоставляющий возможность вложения пакетов друг в друга [15].

Пакет — это ведущий метод организации модели в Unified Method Language. Пакет обладает всем множеством включенных в него компонентов. Про соответствующие компоненты пакета принято говорить о их принадлежности к данному пакету. При этом отдельный компонент может иметь принадлежность только к одному пакету. Допускается вложенность одних пакетов в другие.

Пакет, который является составной частью другого носит название подпакета. Таким образом, для компонентов модели, входящих в состав пакетов, которые, в свою очередь, являются подпакетами более общих пакетов, создается отношение вложенности пакетов, которое представляет собой иерархию.

Для представления пакетов на диаграммах используется специальный графический символ — большой прямоугольник, объединенный с прямоугольником меньшего размера в левой верхней части (Рисунок 2). При этом в поле большого прямоугольника записывается информация, имеющая отношение к этому пакету. При отсутствии подобной информации внутри записывается уникальное имя данного пакета. При наличии подобной информации имя прописывается в меньшем прямоугольнике.

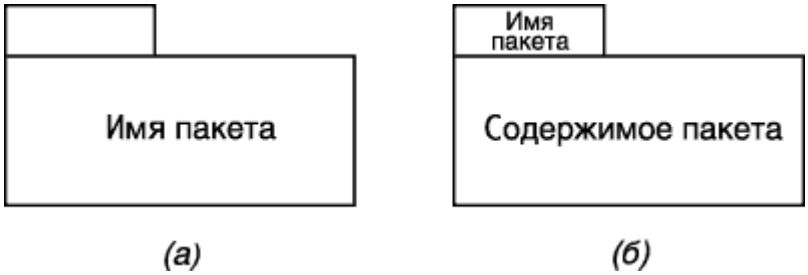

Рисунок 2. Графическое изображение пакетов в языке UML

Имя пакета может предваряться строкой текста, содержащей ключевое слово, ранее определенное в UML, и имеющее название стереотипа.

Содержимым пакета могут быть имена отдельных компонентов и их характеристики, например, область видимости.

К типовым отношениям между пакетами относится отношение вложенности. Данное отношение изображается с помощью помещения одного прямоугольника внутрь другого, обозначающего более общий пакет (Рисунок 3).

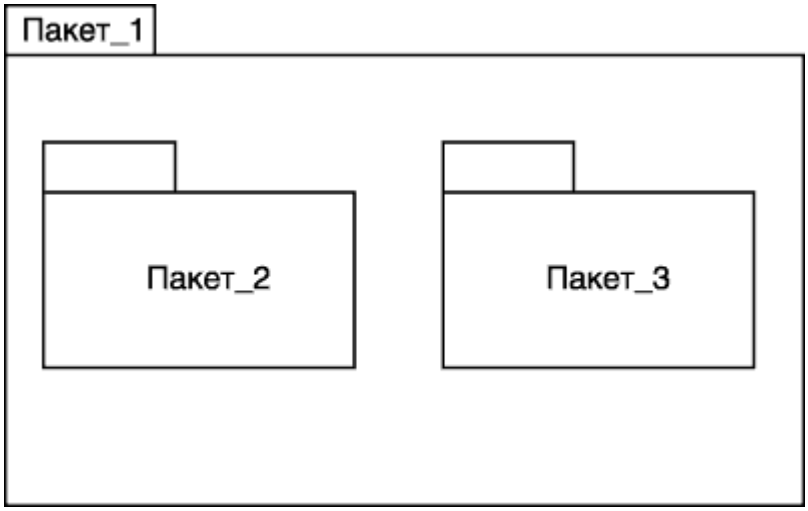

Рисунок 3. Графическое изображение вложенности пакетов друг в друга

Другим способом изображения данного отношения служат отрезки линий по аналогии с графическим представлением дерева (Рисунок 4). При подобном отображении более общий пакет располагается в верхней части рисунка, а относящиеся к нему подпакеты — на более низком уровне. Контейнер связывается с подпакетами сплошной линией, на конце которой, со стороны контейнера, изображается специальный символ, обозначающий принадлежность подпакетов к данному контейнеру, и что иных он не содержит.

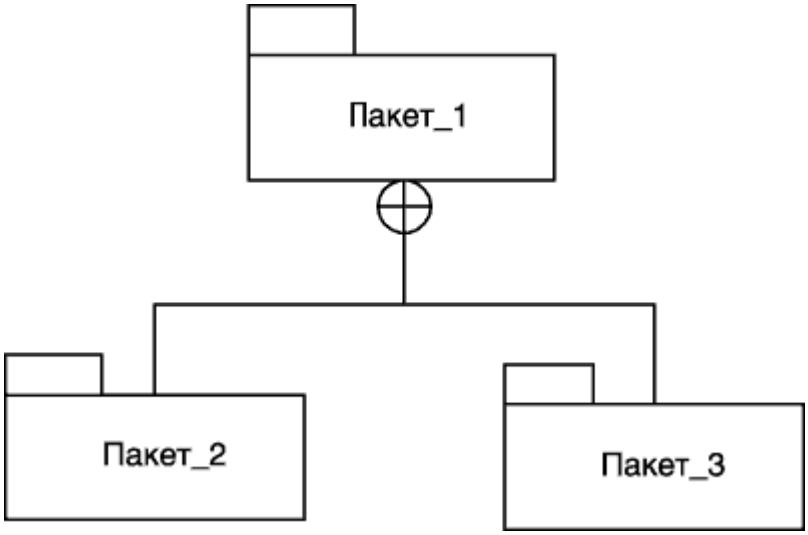

Рисунок 4. Графическое изображение языка UML для вложенности пакетов друг в друга с помощью явной визуализации отношения включения

Модель является подклассом пакета и отображает абстракцию физической системы, предназначенной для конкретной цели. Данная цель определяет не элементы, которые необходимо включить в модель, и те, которые учитывать не обязательно. Иначе говоря, модель представляет релевантные аспекты физической системы, имеющие значение для выполнения поставленной задачи.

В прикладных задачах цель, как правило, представляется в виде первоначальных требований к системе, которые в UML отображаются в виде вариантов применения системы.

Внутри UML для единственной физической системы могут быть определены различные модели, описывающие систему с точки зрения различных подходов (Рисунок 5). В качестве примеров подобных моделей могут выступать модель проектирования, логическая модель, и другие. Каждая из этих моделей обладает собственными представлением физической системы и уровнем абстракции. Модели могут быть вложены друг в друга, подобно пакетам. Некоторое множество моделей, принадлежащих одной и той же системе могут быть включены в состав одного пакета. Это обстоятельство является одним из важнейших механизмов создания моделей в среде UML.

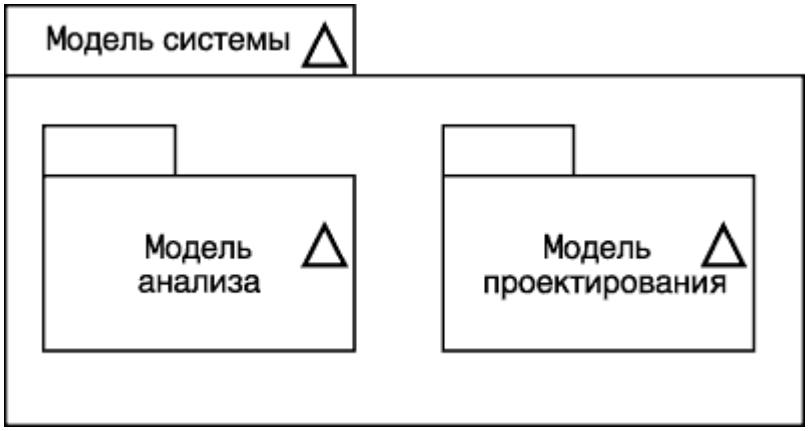

Рисунок 5. Изображение модели системы в виде пакетов моделей анализа и проектирования

Подсистема — это группировка компонентов модели, которые определяют простейшее поведение физической системы. В процессе группировки компоненты подразделяются на спецификацию поведения и его реализацию. Графическим изображением подсистемы также является прямоугольник, но в отличии от пакета он разделен на три секции (рисунок 6). В верхнем, меньшем, прямоугольнике отображается символ, указывающий на подсистему. Уникальное имя подсистемы, вместе с необязательным стереотипом, вписывается внутрь большого прямоугольника. При наличии текста внутри большого прямоугольника имя подсистемы может быть вписано рядом со специальным символом, указывающим на подсистему. [15]

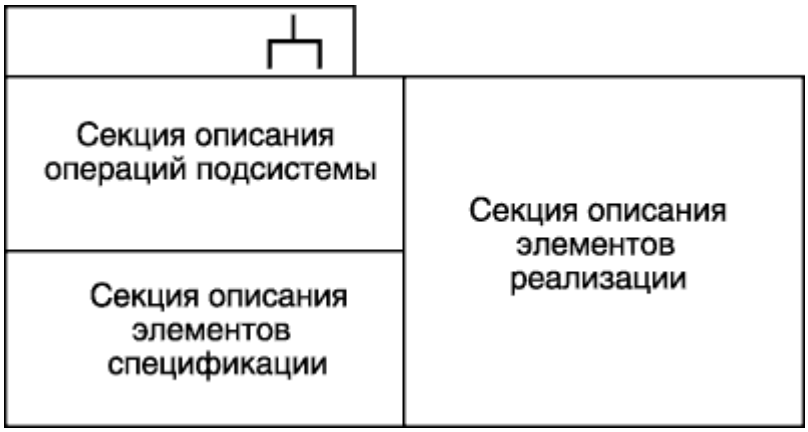

Рисунок 6. Графическое изображение подсистемы в языке UML

2.3.Канонические диаграммы языка UML.

В представлении языка UML вся структура модели представляется в форме специальных графических конструкций, имеющих название диаграмм.

Диаграмма — изображение комплекса компонентов модели, имеющий вид связного графа, вершины и ребра которого имеют строго определенную семантику.

Нотация канонических диаграмм — основополагающий метод создания моделей с применением языка UML.

В UML представлены следующие разновидности канонических диаграмм:

- классов
- кооперации
- вариантов использования
- последовательности
- развертывания
- компонентов
- состояний
- деятельности

Список данных диаграмм и их имена носят название канонических, так как именно они определяют собой базовую часть графической нотации UML.

Сам процесс объектно-ориентированного анализа и проектирования во многом определяется процессом создания данных диаграмм. Комплекс выстроенных подобным образом диаграмм определяется как самодостаточный, поскольку он включает всю информацию, необходимую для развертывания сложной системы.

Каждая отдельная диаграмма уточняет те или иные представления о модели системы, используя терминологию UML. Максимально общей концептуальной моделью при этом является диаграмма вариантов использования, которая является базовой для создания прочих диаграмм.

Диаграмм классов — это логическая модель, дающая представления о статических аспектах структуры системы.

Диаграмма кооперации и диаграмм последовательности являются разновидностями логической модели, отражающими динамические аспекты сложной системы.

Основная функция диаграмм состояний и диаграмм деятельности — моделирование реакции системы.

Диаграммы компонентов и развертывания используются для изображения физических элементов системы. На этом основании их относят к физической модели сложной системы. [2]

Обобщенная модель сложной системы определенная с помощью нотаций UML может быть представлена в виде комплекса данных диаграмм (Рисунок 7).

Рисунок 7. Интегрированная модель сложной системы в нотации UML

Помимо графических компонентов, определенных для всех канонических диаграммы, они могут содержать текстовую информацию, расширяющую семантику базовых компонентов. В UML предусмотрено три способа расширения, включающие в себя следующие структурные компоненты:

Стереотип — новый тип компонента модели, расширяющий семантику метамодели. Для стереотипов обязательно создаваться на основе существующих классов, имеющих описание в метамодели UML.

Основная функция стереотипов заключается в расширении семантики, не затрагивая структуры существующих классов. Стереотипы могут быть как описаны в UML, так и определены разработчиком. На диаграммах стереотипы обозначаются в виде текста, ограниченного угловыми кавычками. Исходно определенные стереотипы относятся к ключевым словам UML, использующимся на канонических

диаграммах на языке оригинала.

Помеченное значение — явное представление свойства в качестве пары «имя — значение». Внутри помеченного значения имя определяется как «тег».

На диаграммах помеченные значения обозначаются в виде строки текста в определенном формате, ограниченном фигурными скобками. Для этого применяется специальный формат записи: {тег = значение}. Теги упоминаются в нотации UML, при этом их определение нестрогое. Следовательно, теги могут указываться непосредственно разработчиком.

Ограничение — это логическое условие, вводящие ограничение для семантики определенного элемента модели.

Спецификации ограничений вводятся, как правило, самим разработчиком. На диаграммах ограничения обозначаются в виде строки текста, ограниченного фигурными скобками. Формальная запись ограничений проводится с помощью специального языка объектных ограничений, который входит в состав UML. [15]

2.4.CASE-средства объектно-ориентированного анализа и проектирования.

Для реализации процесса проектирования информационных систем с использованием объектно-ориентированной парадигмы применяются CASE-средства (Computer Aided Software Engneering). В частности, к подобным средствам реализации систем с применением языка UML относятся: IBM Rational Rose, Borland Together, Sparx Systems Enterprise Architect. [16]

Rational Rose — популярное средство анализа, моделирования и разработки информационных систем. Применяется для решения широкого круга задач в ходе развертывания информационной системы, таких как анализ бизнес-процессов, или генерация кода на выбранном языке программирования. Подобные возможности позволяют не только создать новую систему, но и модернизировать имеющуюся, с помощью процесса обратного проектирования. [1]

С целью охвата как можно большей части рынка CASE-средств, произведена сегментация продукта на следующие версии:

- Rational Rose Modeler — данная версия предоставляет возможность проведения анализа бизнес-процессов и проектирования информационной системы. Отсутствует возможность генерации кода.
- Rational Rose Professional — данная версия предоставляет возможность как прямого, так и обратного проектирования на выбранном языке программирования. Поставляется в определенной заказчиком конфигурации с поддержкой определенного языка программирования. В результате использования данного инструментария разработчик получает каркасный код на выбранном языке программирования.
- Rational Rose RealTime — данная версия позволяет получить исполняемый код в реальном масштабе времени; предоставляет возможность прямого и обратного проектирования на C и C++. Предоставляет разработчикам скомпилированный исполняемый файл.
- Rational Rose Enterprise — данная версия предоставляет функционал всех иных редакций.

В зависимости от конфигурации поставки Rational Rose может содержать различное количество диаграмм.

К основным возможностям продукта относятся:

- Прямое и обратное проектирование с применением таких языков программирования как ADA, Java, C, C++, Basic
- поддержка следующих технологий: COM, DDL, XML
- Генерация схем баз данных Oracle и SQL

Данное программное обеспечение обладает открытым API, дающим возможность самостоятельной разработки модулей для иных языков программирования. Кроме того, на рынке предоставлено значительное число готовых модулей с поддержкой языков программирования и RAD-систем, таких как Delphi, Jbuilder, SmallTalk и т. д.

Одной из востребованных возможностей программы является обратное проектирование, при котором по исходному коду идет восстановление диаграммы классов, дающей возможность оценить структуру программы.

2.5.Borland Together.

Интегрированная среда разработки с поддержкой UML, предоставляющая широкие возможности по оптимизации процессов анализа и разработки информационной системы [16].

К ее особенностям относятся:

- Поддержка принципов экстремального программирования
- Оптимизация процесса разработки с использованием паттернов. Применение определенных стандартных решений ускоряет создание моделей и позволяет привести ее в соответствие с корпоративными стандартами.
- Быстрое развертывание на нескольких серверах приложений.
- Встроенные функции контроля качества, дающие возможность определить проблемы еще в процессе разработки, до ввода в эксплуатацию, что позволяет избежать значительных финансовых затрат.
- Поддержка всех основных типов диаграмм UML.
- Предоставляет возможность генерации кода и обратного проектирования.
- Функция быстрой генерации проектной документации.
- Поддержка схем баз данных
- Визуальные средства проектирования пользовательского графического интерфейса.
- Редактор кода с широкими возможностями настройки

Как и у предыдущего инструментария, для Borland Together существует сегментация на несколько версий:

- Together Control Center — интегрированное средство разработки, позволяющее оптимизировать процессы анализа, проектирования и разработки информационных систем.
- Together Solo — версия для малого бизнеса, дающее возможность быстрого моделирования небольших проектов с применением UML.
- Borland Together Editions — набор версий с поддержкой выбранных сред разработки.

2.6.Sparx Systems Enterprise Architect.

CASE-инструментарий для UML-моделирования в средах Windows и Linux, предоставляющий возможность многопользовательской разработки и обладающий

дружественным интерфейсом пользователя.

К его возможностям относится следующее:

• поддержка всех типов диаграмм UML.

• поддержка популярных языков программирования, в частности C++, Java, C#, VisualBasic, .NET, PHP, Delphi.

• моделирование баз данных.

• возможность как прямого, так и обратного проектирования.

• наличие UML-профилей расширения, позволяющих создавать узкоспециализированные модели.

• поддержка паттернов проектирования.

• возможность генерации документации с поддержкой форматов rtf и HTML.

• поддержка работы нескольких пользователей.

• встроенные функции тестирования.

• развитые средства проектирования пользовательского интерфейса. [16]

Существуют три версии данного продукта:

EA Desktop Edition — версия для индивидуальных аналитиклв и разработчиков. Содержит базовый инструментарий проектирования, с применением ряда ограничений. В частности не поддерживается импорт и экспорт кода, технология ActiveX, возможность совместной работы с диаграммами.

EA Professional Edition — расширенная версия, содержащая полный функционал разработки, ориентированная на совместную работу. Допускается многопользовательский доступ к моделям; имеется поддержка ActiveX, импорт и экспорт кода, схем баз данных Oracle, SQL Server, Microsoft Access.

EA Corporate Edition — наиболее полная версия, обладающая всеми доступными возможностями вышеуказанных версий. Также поддерживается интеграция с MySQL, SQL Server, Sybase Adaptive Server Anywhere, PostgreSQL, Oracle9. Кроме того имеются функции автоизации пользователей, их групп, блокировки элементов. Целевая аудитория данной редакции — большие предприятия.

ЗАКЛЮЧЕНИЕ

Применение экономических информационных систем позволяет повысить эффективность работы предприятия за счет автоматизации выполняемой работы, оптимизации численности персонала, его переориентации на решение иных задач, получения актуальной отчетности и ряда других преимуществ.

Анализ и проектирование экономической информационной системы является нетривиальной задачей, качественное выполнение которой определяет эффективность всего предприятия. К настоящему времени существует два основных подхода к решению данной проблемы: структурный и объектно-ориентированный. Объектно-ориентированная парадигма является появилась позже, и обладает рядом преимуществ. В частности, данный подход позволяет создавать модели меньшего размера, за счет применения общих механизмов; предоставляет возможность быстрой модернизации за счет подключения новых объектов и модификации имеющихся; обеспечивает более высокий уровень координации разработчиков, решающих различные задачи внутри одной системы; повышает скорость разработки новых информационных систем за счет повторного применения объектов. Кроме того, данный подход можно определить как «более естественный», поскольку он ориентирован на человеческое восприятие мира.

К недостаткам данной методики можно отнести более высокие затраты (по сравнению со структурным подходом) при первоначальной разработке типовой информационной системы. В дальнейшем этот недостаток нивелируется за счет повторного применения моделей.

В работе подробно рассмотрен объектно-ориентированный подход, указаны основные концепции его реализации.

Эффективность и скорость разработки информационных систем в данной парадигме в значительной степени повысились благодаря стандартизации и разработке языка Unified Method Language. UML остается динамично развивающимся средством моделирования. В работе рассмотрены основные стандарты и диаграммы UML.

Создание UML способствовало также активному развитию средств реализации информационных систем — Computer Aided Software Engineering. CASE-средства позволяют проводить быстрый анализ и проектирование информационных систем с применением UML, и генерацией кода на выбранном языке программирования. Данный инструментария позволяет проводить, зачастую, как прямое, так и обратное проектирование информационных систем.

За время существования данной направления программного обеспечения на рынке появились и стали популярными такие продукты, как Rational Rose, Borland Together, **S**parx Systems Enterprise Architect, и другие.

В данной работе рассмотрены перечисленные CASE-средства, указаны их особенности применения.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Боггс У., Боггс М. UML и Rational Rose секреты эффективного проектирования объектно-ориентированных приложений – М.: ЛОРИ, 2004. – 509 с.
2. Буч Г., Рамбо Д., Джекобсон А. Язык UML: Руководство пользователя: Пер. с англ. – М.: ДМК, 2000. – 432 с.
3. Буч Г. Объектно-ориентированный анализ и проектирование

с примерами приложений на C++ Второе издание. Перевод с английского под редакцией И. Романовского и Ф. Андреева СантаКлара, Калифорния.: Rational 2001. - 342 с.

1. Вовканыч С.И., Парфенцева Н.А. «Социальный интеллект»: метафора или научное понятие? // Социс, 1993, № 8, с.153.
2. Елашкин М. SAP Business One. Строим эффективный бизнес. М.: КУДИЦ-ПРЕСС, 2007. - 240 с.
3. Михеева Е.В. Информационные технологии в профессиональной деятельности. – М.: Академия, 2007. – 379 с.
4. Мюллер Р. Базы данных и UML. Проектирование М.: Лори, 2002 - 432 с.
5. Суркова Н.Е., Остроух А.В. Методология структурного проектирования информационных систем: Монография /Красноярск: Научно-инновационный центр, 2014. - 190 с.

6. Титоренко Г.А. Информационные системы в экономике. – М.: ЮНИТИ, 2007. – 463 с.
7. Усольцев А.А. «Информационные системы в экономике». Конспект лекций. Новокузнецкий филиал Томского политехнического университета, 2009. - 69 с.
8. Федотова Д.Э., Семенов Ю.Д., Чижик К.Н. CASE-технологии: Практикум. – М.: Горячая линия - Телеком, 2005 – 160 с.
9. Grady Booch, James Rumbaugh, Ivar Jacobson, «The Unified Modeling Language User Guide» Publisher: Addison Wesley, 1999 — 391 pages.
10. ГОСТ Р 53622-2009. Информационные технологии. Информационно-вычислительные системы. Стадии и этапы жизненного цикла, виды и комплектность документов
11. ISO/IEC 2382:2015 Information technology — Vocabulary
12. Грекул В. Национальный исследовательский университет «Высшая Школа Экономики» Проектирование информационных систем. [Электронный ресурс] URL: http://www.intuit.ru/studies/courses/2195/55/info (дата обращения: 24.08.2017)
13. Курс Введение в UML. Обзор CASE-средств для построения диаграмм в UML. [Электронный ресурс] URL: http://www.intuit.ru/studies/professional_retraining/941/courses/229/lecture/5963?page=1 (дата обращения: 24.08.2017)