ТЕМА 2

1 Автоматизация построения и управления программным проектом.

2. Треугольник успеха.

3. Модели процесса разработки.

4. Унифицированный процесс разработки.

2.1. Автоматизация построения и управления программным проектом

Программные средства являются исполнительными элементами многих компьютерных систем различного назначения, например, CASE, гибких проектируемых систем, автоматизированных систем управления, компьютерных игр и др. В связи с этим ПС становятся продуктом научно-технического назначения, создаются в строгом соответствии с действующими стандартами и утвержденной технологией, сопровождаются научно-технической документацией и обеспечиваются гарантиями поставщика. Это объясняется тем, что технические возможности, адаптируемость и эффективность компьютерных систем определяются качеством используемого ПО, а без современной индустриальной технологии его создания требуемое качество недостижимо.

В области программирования так же, как и в промышленном производстве, значительный эффект может дать многократное использование хорошо отработанных компонентов в качестве комплектующих изделий. Такие компоненты выполняют типовые функции или функции, характерные для определенных предметных областей применения компьютерных систем, и называются программными модулями. Для обеспечения повторного использования ПС необходима стандартизация их создания на всех этапах ЖЦ. Это позволяет значительно сократить дублирующие разработки, внедрить сборочное программирование и ввести на предприятиях накопление высококачественных программных продуктов для их многократного использования в качестве типовых комплектующих изделий. Для того чтобы ПО отвечало всем вышеназванным требованиям, процесс его проектирования должен включать в себя следующие этапы:

1. Анализ технических требований – определение функций, которые должна реализовать создаваемая система.

2. Проектирование – переход от спецификации системы к тому, каким образом она будет выполнять заданные ей функции.

3. Реализация – создание программного средства.

4. Аттестация – контроль корректности ПС.

5. Верификация – формальная проверка то го, что созданное ПС удовлетворяет всем техническим требованиям.

6. Сопровождение конфигураций и управление версиями – учет истории разработки отдельных компонентов системы и их взаимосвязей в рамках программного комплекса.

7. Документирование – фиксация технических решений для тех, кто будет сопровождать систему, и создание учебных материалов и инструкций для пользователей.

8. Управление проектом – планирование, слежение и руководство при разработке ПО.

В настоящее время эти этапы выполняются самыми разными способами, что находит отражение в широком спектре существующих инструментальных средств поддержки автоматизированной разработки ПО.

При создании ПО приходится решать целый ряд задач, объединенных под общим названием – *проект*.

Проект – это уникальный комплекс взаимосвязанных мероприятий, направленных на достижение конкретной цели при определенных требованиях к срокам, бюджету и характеристикам ожидаемых результатов.

В этом определении следует обратить внимание на следующее:

1) Каждый проект характеризуется конкретной целью, ради которой он затевается.

2) Каждый проект в чем-то уникален.

3) Любой проект ограничен по времени “жизни”.

4) Каждый проект характеризуется конкретными ресурсами, выделенными на его выполнение.

Еще одним понятием, актуальным для проекта, является его масштаб.

Масштаб проекта – это совокупность цели проекта и планируемых для ее достижения затрат времени и средств. Другими словами, это своеобразное трехмерное пространство (цель-время-деньги), в котором живут участники проекта и сам проект. Опираясь на введенное понятие масштаба, можно сказать, что управление проектом направлено на сохранение его исходного масштаба – содержания и границ.

Управление проектом – это процесс планирования, организации и контроля состояния задач и ресурсов проекта, направленный на своевременное достижение цели проекта.

В ходе управления любым проектам должно быть обеспечено решение следующих задач:

- соблюдение директивных сроков завершения проекта;

- рациональное распределение материальных ресурсов и исполнителей между задачами проекта, а также во времени;

Своевременная коррекция исходного плана в соответствии с реальным положением дел. Чтобы проект оказался успешным, в его реализации должны быть предусмотрены три главные фазы:

1) Формирование плана.
2) Контроль реализации плана и управление проектом.
3) Завершение проекта.

Чем качественнее будут реализованы эти фазы, тем выше вероятность успешного выполнения проекта в целом.

Автоматизация проектирования должна помогать справляться со всеми аспектами сложности при разработке ПО. Поэтому независимо от прикладной области, к которой относится решаемая задача, и от принятого уровня абстракции необходимо, чтобы используемое CASE-средство поддерживала выполнение следующих функций:

1. Ввод описания проекта в систему.
2. Просмотр-обход – предоставление пользователям возможности выбора нужных фрагментов проекта и формирования необходимых запросов.
3. Декомпозиция системы – возможность представления проекта в виде удобных для обработки частей-модулей.
4. Контроль соблюдения правил и норм – проверка соответствия проекта всем требованиям с учетом количественных измерений.
5. Моделирование – обработка проекта на выбранном уровне абстракции с целью лучшего понимания его поведенческих характеристик.
6. Синтез (программирование) – преобразование проекта из одного вида представления в другой, как правило, с переходом на более низкий уровень абстракции.
7. Управление – возможность всем участникам процесса проектирования оставаться в рамках выбранной инженерной методологии.

Целенаправленное управление проектом предназначено для пропорционального распределения ресурсов между работами по созданию ПС на протяжении всего цикла проектирования вплоть до внедрения системы в серийное производство или ее массового использования. В общем случае при проектировании необходимо создать в соответствии с принятым критерием эффективности оптимальную систему управления или обработки информации.

2.2. Треугольник успеха

Успех любого программного проекта зависит от рех составляющих: *системы обозначений (нотации, языка), процесса и инструмента.* Одинаково нужны все три составляющие.

Система обозначений важна в любой модели. Это связующее звено между всеми составляющими процесса разработки проекта. Примером полной и надежной системы обозначений может служить унифицированный язык моделирования (*Unified Modeling Language – UML*). С его помощью можно описывать модели на любом этапе разработкиПС: от анализа требований до проектирования и реализации.

Современные проекты должны разрабатываться с помощью эффективных средств в соответствии со строгим графиком и обеспечивать возможность внесения изменений и адаптации к конкретным условиям и требованиям. ЖЦ проекта должен быть управляемым, что позволит гарантировать его непременное завершение. Примером процесса, подходящего для современных *IT*-проектов, является *унифицированный процесс*. Его методология основана на языке *UML* и находит поддержку в современных инструментах.

Ни одна из современных технологий разработки ПС не обходится без какого-либо инструмента. Сегодня на рынке представлен достаточно широкий спектр инструментов, использующих нотацию *UML* и объектно-ориентированный подход к разработке ПО.

2.3. Модели процесса разработки

В настоящее время существует большое количество стандартных процессов и методологий, применяя которые можно получить ту или иную модель производства ПО. Модель и параметры процесса производства ПО в значительной мере зависят от типа проекта. Можно выделить их основные типы.

- *Проект для постоянного заказчика*. Ситуация, при которой команда разработчиков в течение длительного времени обслуживает единственного заказчика.
- *Продукт под заказ*. Ситуация, при которой команда разрабочиков находит стороннего заказчика и договаривается с ним о разработки программного продукта, призванного решить те или иные проблемы заказчика.
- *Тиражируемый продукт*. Ситуация, при которой команда разрабочиков либо вообще не имеет конкретных заказчиков, либо довольно большое количество заказчиков желают иметь один и тот же продукт.
- *Аутсорсинг*. Это одна из наиболее новых моделей производства ПО. Суть ее состоит в том, что между крупной (обычно) фирмой по производству ПО и другой иностранной фирмой заключается договор о субподряде.

В настоящее время наиболее распространенным является *итерационный процесс* создания ПО. Он представляет собой набор итераций, в рамках каждой из которых проект проходит через все этапы ЖЦ. Итеративный цикл позволяет выявлять проблемы на самых ранних этапах разработки, управлять рисками, раньше начинать тестирование и т.п. Поэтому такой процесс становится более динамичным и управляемым.

В мире существует множество типовых процессов производства ПО: *ISO9001, ISO12207, ISO15504, CMM, Rational Unified Process (RUP), SCRUM, ICONIX, XP, Crystal Clear, ASD, Lean Development* и др.

Под методологией понимается набор методов, практик, метрик и правил, используемых в процессе производства ПО. Главными задачами современной методологии и основанного на ней процесса являются следующие:

- облегчить процедуру введения новых людей в курс процесса производства;
- обеспечить взаимозаменяемость людей;
- распределить ответственности;
- демонстрировать видимый прозрачный процесс;
- создать учебную базу для сотрудников.

Методологии можно условно разбить на три категории: *тяжелые, легкие и средние.* Упрощенно каждая из них предназначена для работы в условиях больших, малых и средних проектов соответсвенно.

Тяжелая категория методологий появилась раньше других и служит неотъемлемой частью моделей качества ПО. Тяжелые методологии

охватывают все аспекты деятельности компании, производящей ПО, – от управления требованиями и планирования процесса до регламентирования отношений с заказчиком. Все методологии данной категории нетерпимы к изменениям и рассматривают людей как ресурс. К этой категории относятся *ISO9001, CMM, SPICE.*

Легкая категория методологий – это некоторя совокупность методов и практик, применявшихся небольшими командами разработчиков в небольших проектах. Все процессы данной категории предусматривают итерационный ЖЦ разработки ПО. Идея легких методологий – обеспечение максимальной скорости и качества разработки ПО при минимуме ограничений. Во всех легких методологиях предусмотрен лишь необходимый минимум докуметов. Примеры легких методологий - *SCRUM, ICONIX, XP, Crystal Clear*.

Средняя категория методологий включает в себя так называемые универсальные процессы. Наиболее популярным представителем этой категории является методология рационального унифицированного процесса – *Rational Unified Process (RUP*). Основная характеристика этой методологии – масштабируемость, т.е. процесс может быть настроен на работу как в малой команде над небольшим проектом, так и в большой команде над большим проектом.

Рассмотрим, какие типы методологий наиболее целесообразно использовать для того или иного типа проекта.

Проект для постоянного заказчика. Самый благоприятный тип проекта для внедрения легких методологий, поскольку заказчик всегда доступен и не прдъявляет сверхтребований к ПО. Однако необходимо учитывать количество разработчиков и степень их распределенности. Как правило, у таких команд не бывает необходимости в сертификации.

Продукт под заказ. Самый уязвимый тип проекта. Фирма целиком зависит от количества заключенных договоров. Постоянно идет поиск новых заказчиков. В таких условиях, конечно же, желательно наличие сертификата ISO. Сертификацию целесообразно проводить лишь при достижении определенной численности персонала, которой будет достаточно для внедрения тяжелой или средней технологии. Алтернативный вариант – одна из легких методологий.

Тиражируемый продукт. Самый устойчивый тип проекта. Выпуск такого проекта всегда характеризуется более низкими затратами на его производство по сравнению с выпуском единичных экземпляров. В данных условиях невозможно использование легкой методологии в чистом виде, т.к. нет возможности постоянно работать с заказчиком. В этом случае все зависит от способа управления командой, тактических и стратегических целей.

Аутсорсинг. Данный вид проектов характеризуется расределенной структурой и начальными предпосылками к утяжелению процесса, поскольку общение с заказчиком происходит в виде документов установленного образца. Если сторона фирмы-заказчика предоставляет команде свой

технологический процесс, то у нее нет совободы выбора. В противном случае стоит остановить свой выбор на каком-либо из процессов средней тяжести.

2.4. Унифицированный процесс разработки

Прежде всего унифицированный процесс – это процесс разработки ПО, т.е. это множество различных видов деятельности, необходимых для преобразования требований пользователей в программную систему.

Однако унифицированный процесс – это больше, чем единичный процесс. *UP* – это обобщенный каркас процесса, который может быть специализирован для широкого круга программных систем, различных областей применений, уровней компетенции и размеров проекта. Его основными принципами являются:

– итерационный и инкрементный подход к созданию ПО;

– управление вариантами использования;

– построение системы на базе архитектуры ПО.

Первый принцип является определяющим. В соответствии с ним разработка системы выполняется в виде нескольких краткосрочных мини-проектов фиксированной длительности (от 2 до 6 недель), называемых итерациями. Каждая итерация включает в себя свои собственные этапы анализа требований, проектирования, реализации, тестирования, интеграции и завершается созданием работающей системы.

Итерационный цикл основывается на постоянном расширении и дополнении системы в процессе нескольких итераций с периодической обратной связью и адаптацией добавляемых модулей к существующему ядру системы. Система постоянно разрастается, поэтому такой подход называют итерационным и инкрементным.

Вариант использования – это часть функциональности системы, необходимая для получения пользователем значимого для него, ощутимого и измеримого результата. Варианты использования обеспечивают функциональные требования. Все варианты использования в совокупности составляют модель вариантов использования, которая описывает полную функциональность системы.

Варианты использования в *UP* – это не только средство описания требований к системе. Они направляют далее весь процесс ее разработки. Основываясь на модели вариантов использования, разработчики создают все последующие модели.

Поскольку варианты использования управляют процессом, они не выделяются изолированно, а разрабатываются совместно с созданием архитектуры системы. Следовательно, варианты использования управляют архитектурой системы, которая, в свою очередь, оказывает влияние на их выбор. И архитектура системы, и варианты использования развиваются по мере хода жизненного цикла.

Созданная архитектура является основой всей дальнейшей разработки. В будущем неизбежны незначительные изменения в деталях архитектуры, однако серьезные изменения маловероятны.