5. Лекция: Стандарты ISO, SW-CMM. CASE-технологии: версия для печати и PDA

Стандарты ISO

В 1947 году в Лондоне представители 25 стран решили создать международную организацию, основной задачей которой стала бы координация разработок и унификация международных стандартов. Новая организация получила название International Organization for Standardization (ISO). В настоящее время ее членами являются около 100 стран.

Главной причиной появления единых стандартов стало желание инициаторов создания этой организации устранить технические барьеры в торговле, которые возникли вследствие того, что в разных странах для одних и тех же технологий и товаров действовали разнородные стандарты. Сегодня стандартами ISO "перекрыты" многие технологические отрасли – от программирования и телекоммуникаций до банковской и финансовой сферы.

По уставу членом ISO может стать "самая авторитетная в стране организация, занимающаяся выработкой стандартов". Таким образом, интересы какой-либо страны в ISO может представлять только одна организация. Помимо основных членов, в ISO входят так называемые члены-корреспонденты (как правило, ими становятся относительно крупные "стандартообразующие" организации отдельных стран, однако еще недостаточно мощные, чтобы распространить свое влияние на стандартизацию технологий по всей стране). Члены-корреспонденты не принимают активного участия в разработке международных стандартов, но имеют полный доступ к интересующей их информации. Наконец, есть еще и члены-подписчики – они представлены организациями развивающихся стран. Россия на сегодня имеет статус члена-корреспондента ISO.

Выполнение технической работы в ISO возложено на 2700 технических комитетов, подкомитетов и рабочих групп, в состав которых входят представители правительственных, промышленных, научно-исследовательских и юридических кругов (всего около 500 организаций). Каждая организация – член ISO – имеет право включить своего представителя в любой комитет, в деятельности которого она особо заинтересована.

Новые стандарты рождаются в соответствии с тремя принципами.

Во-первых, они являются **результатом консенсуса всех заинтересованных сторон** – производителей, поставщиков, потребителей, профессиональных разработчиков, правительственных и исследовательских организаций.

Во-вторых, стандарты имеют действительно **мировое распространение** и удовлетворяют как производителей, так и потребителей.

В-третьих, появление новых стандартов **диктуется исключительно требованиями свободного рынка**, а не чьей-то злой или доброй волей. Если рынок созрел для нового стандарта, то такой стандарт появляется.

Процесс создания нового стандарта включает три этапа. Обычно инициатива его разработки исходит от производителей, которые доводят базовые предложения стандарта до своего представителя в ISO. Если эта организация признает целесообразность создания нового стандарта, то соответствующая рабочая группа определяет техническую область, на которую предполагаемый стандарт будет распространяться. На втором этапе происходит выработка технических спецификаций, в ходе которой представители различных стран стремятся

достичь консенсуса. На заключительном этапе первая версия стандарта утверждается (за стандарт должно проголосовать 75% кворума) и публикуется.

По мере совершенствования технологий, появления новых материалов, методов обработки, повышения требований к качеству и надежности изделий возникает необходимость в пересмотре стандартов. В ISO существует правило: все стандарты должны пересматриваться не реже чем раз в пять лет. Сегодня "перу" ISO принадлежит около 9300 различных стандартов, описание которых занимает 171000 страниц текста на английском языке.

Серия ISO 9000 [18] (управление качеством) включает в себя следующие стандарты:

- ISO 9000-1 (1994 г.). Управление качеством и гарантии качества. Часть 1. Руководство по выбору и использованию.
- ISO 9000-2 (1993 г.). Управление качеством и гарантии качества. Часть 2. Общее руководство по применению стандартов ISO 9001, ISO 9002 и ISO 9003.
- ISO 9000-3 (1991 г.). Управление качеством и гарантии качества. Часть 3. Руководство по применению стандарта ISO 9001 при разработке, установке и сопровождении ПО.
- ISO 9000-4 (1993 г.). Управление качеством и гарантии качества. Часть 4. Руководство по управлению надежностью программ.

Основополагающий стандарт ISO 9001 (1994 г.) задает модель системы качества для процессов проектирования, разработки, производства, установки и обслуживания (продукта, системы, услуги).

Основным преимуществом моделей ISO серии 9000 является их известность, распространенность, признание на мировом уровне, большое количество экспертов и аудиторов и невысокая стоимость услуг сертификации. Универсальность же моделей ISO серии 9000 имеет определенные недостатки: они являются достаточно высокоуровневыми, задают абстрактные модели и не содержат конкретных методологических разработок.

По ISO качество – это совокупность свойств и характеристик продукта, процесса или услуги, которые обеспечивают способность удовлетворять заявленным или подразумеваемым потребностям. Современные способы обеспечения качества базируются на подходах TQM (Total Quality Management). Это управление ресурсами и применение количественных методов анализа для улучшения: разработок, материалов и услуг, поставляемых в организацию; всех процессов внутри организации; степени удовлетворенности настоящих и будущих потребностей клиентов.

В модели ISO 9000 лишь упоминаются требования, которые должны быть реализованы, но не говорится, как это можно сделать. Поэтому для построения полноценной системы качества по ISO помимо основной модели ISO 9001 (1994 или 2000 года), необходимо использовать вспомогательные отраслевые и рекомендательные стандарты. Для организации, занимающейся разработкой программного обеспечения, такими стандартами являются: ISO 9004-1:94 (ISO 9004:2000), ISO 8402:94 (ISO 9000:2000), ISO 9000-3:91, ISO 10007:95, ISO 10013:95, ISO 12207:95.

Семейство стандартов ISO 9000 версии 2000 года разработано с тем чтобы преодолеть недостатки ISO 9000 версии 1994 года и помочь организациям всех специализаций, типов и размеров внедрить и использовать эффективные системы менеджмента качества.

Подход к системам менеджмента качества является общим и применяется к организациям в любой отрасли экономики, поэтому данный стандарт не устанавливает каких-либо конкретных требований к программным продуктам. Требования к ним могут определяться

заказчиками или третьими лицами и содержаться в технических спецификациях, стандартах на продукт, стандартах на процесс, контрактных соглашениях и нормативных документах.

В основу построения системы качества в соответствии с моделью ISO 9000:2000 закладываются следующие принципы:

- концентрация на потребностях заказчика;
- активная лидирующая роль руководства;
- вовлечение исполнителей в процессы совершенствования;
- реализация процессного подхода;
- системный подход к управлению;
- обеспечение непрерывных улучшений;
- принятие решений на основе фактов;
- взаимовыгодные отношения с поставщиками.

При этом методически в полном соответствии с дисциплиной построения сложных систем в стандарте ISO 9000:2000 предусматривается, с одной стороны, построение организационной системы "сверху — вниз": от целей предприятия и его политики – к организационной структуре и формированию бизнес-процессов, и с другой – итеративное развитие организационной системы через механизмы измерения и улучшения.

Внедрение системы менеджмента качества организацией–разработчиком программных продуктов по ISO 9000 версии 2000 года состоит из нескольких этапов. В их числе выделяются:

- измерение характеристик продуктов для определения эффективности каждого процесса, направленного на достижение соответствующего качества;
- применение результатов измерений для определения текущей эффективности процессов создания и внедрения продуктов;
- определение способов предотвращения дефектов, снижения изменяемости продукции и минимизации доработок;
- поиск возможностей по снижению рисков и улучшению эффективности и производительности технологических и иных процессов;
- выявление и расстановка в порядке важности тех улучшений, которые могут давать оптимальные результаты с приемлемыми рисками;
- планирование стратегии, процессов и ресурсов для получения идентифицированных улучшений продукции;
- контроль результатов улучшений;
- сравнение полученных результатов с ожидаемыми;
- определение подходящих корректирующих действий.

Реализация этих этапов возможна только при наличии в организации системы критериев, показателей и факторов качества, а также методов их измерения и оценки.

Capability Maturity Model for Software (Модель SEI SW-CMM)

В 1982 году Министерство обороны США образовало комиссию, основной задачей которой стало исследование проблем, возникающих при разработке программных продуктов в организациях министерства. В результате деятельности комиссии в декабре 1984 году был создан Институт инженеров-разработчиков программного обеспечения (Software Engineering Institute, SEI) на базе Университета Карнеги-Меллона в Питсбурге.

В 1986 году SEI и корпорация Mitre под руководством Уоттса Хамфри (Watts Humphrey) приступили к разработке критериев оценки зрелости технологических процессов – Capability Maturity Model (CMM) [19].

Далее события развивались в следующем порядке.

1987 г. SEI публикует: краткое описание структуры CMM; методы оценки процессов разработки ПО; методы оценки зрелости процессов производства ПО; анкету для выявления степени зрелости процессов (для проведения самостоятельного, внутреннего аудита и внешнего аудита).

1991 г. Выпуск версии CMM v1.0.

1992 г. Выпуск версии CMM v1.1.

1997 г. Выпуск очередной (усовершенствованной) версии CMM.

Методология CMM разрабатывалась и развивалась в США как средство, позволяющее выбирать лучших производителей ПО для выполнения госзаказов. Для этого предполагалось создать критерии оценки зрелости ключевых процессов компании-разработчика и определить набор действий, необходимых для их дальнейшего совершенствования. В итоге методология оказалась чрезвычайно полезной для большинства компаний, стремящихся качественно улучшить существующие процессы проектирования, разработки, тестирования программных средств и свести управление ими к понятным и легко реализуемым алгоритмам и технологиям, описанным в едином стандарте.

CMM де-факто стал именно таким стандартом. Его применение позволяет поставить разработку ПО на промышленную основу, повысить управляемость ключевых процессов и производственную культуру в целом, гарантировать качественную работу и исполнение проектов точно в срок. Основой для создания CMM стало базовое положение о том, что фундаментальная проблема "кризиса" процесса разработки качественного ПО заключается не в отсутствии новых методов и средств разработки, а в неспособности компании организовать технологические процессы и управлять ими.

Для оценки степени готовности предприятия разрабатывать качественный программный продукт CMM вводит ключевое понятие **зрелость организации** (Maturity). **Незрелой** считается организация, в которой:

- отсутствует долговременное и проектное планирование;
- процесс разработки программного обеспечения и его ключевые составляющие не идентифицированы, реализация процесса зависит от текущих условий, конкретных менеджеров и исполнителей;
- методы и процедуры не стандартизированы и не документированы;
- результат не предопределен реальными критериями, вытекающими из запланированных показателей, применения стандартных технологий и разработанных метрик;
- процесс выработки решения происходит стихийно, на грани искусства.

В этом случае велика вероятность появления неожиданных проблем, превышения бюджета или невыполнения сроков сдачи проекта. В такой компании, как правило, менеджеры и разработчики не управляют процессами – они вынуждены заниматься разрешением текущих и спонтанно возникающих проблем. Отметим, что на данном этапе развития находится большинство российских компаний.

Основные признаки **зрелой** организации:

- в компании имеются четко определенные и документированные процедуры управления требованиями, планирования проектной деятельности, управления конфигурацией, создания и тестирования программных продуктов, отработанные механизмы управления проектами;
- эти процедуры постоянно уточняются и совершенствуются;
- оценки времени, сложности и стоимости работ основываются на накопленном опыте, разработанных метриках и количественных показателях, что делает их достаточно точными;
- актуализированы внешние и созданы внутренние стандарты на ключевые процессы и процедуры;
- существуют обязательные для всех правила оформления методологической программной и пользовательской документации;
- технологии незначительно меняются от проекта к проекту на основании стабильных и проверенных подходов и методик;
- максимально используются наработанные в предыдущих проектах организационный и производственный опыт, программные модули, библиотеки программных средств;
- активно апробируются и внедряются новые технологии, производится оценка их эффективности.

CMM определяет **пять уровней** технологической зрелости компании, по которым заказчики могут оценивать потенциальных претендентов на заключение контракта, а разработчики – совершенствовать процессы создания ПО.

Каждый из уровней, кроме первого, состоит из нескольких **ключевых областей процесса** (Key Process Area), содержащих **цели** (Goal), **обязательства по выполнению** (Commitment to Perform), **осуществимость выполнения** (Ability to Perform), **выполняемые действия** (Activity Performed), их **измерение и анализ** (Measurement and Analysis) и **проверку внедрения** (Verifying Implementation). Таким образом, CMM фактически является комплексом требований к ключевым параметрам эффективного стандартного процесса разработки ПО и способам его постоянного улучшения. Выполнение этих требований, в конечном счете, увеличивает вероятность достижения предприятием поставленных целей в области качества.

Начальный уровень (Initial Level – Level 1).

К данному уровню относится компания, которой удалось получить заказ, разработать и передать заказчику программный продукт. Стабильность разработок отсутствует. Лишь некоторые процессы определены, результат всецело зависит от усилий отдельных сотрудников. Успех одного проекта не гарантирует успешности следующего. К этой категории можно отнести любую компанию, которая хоть как-то исполняет взятые на себя обязательства.

Ключевые области процесса этого уровня не зафиксированы.

Повторяемый уровень (Repeatable Level – Level 2).

Этому уровню соответствуют предприятия, обладающие определенными технологиями управления и разработки. Управление требованиями и планирование в большинстве случаев основываются на разработанной документированной политике и накопленном опыте. Установлены и введены в повседневную практику базовые показатели для оценки параметров проекта. Менеджеры отслеживают выполнение работ и контролируют временные и производственные затраты.

В компании разработаны некоторые внутренние стандарты и организованы специальные группы проверки качества (QA). Изменения версий конечного программного продукта и созданных промежуточных программных средств отслеживаются в системе управления конфигурацией. Имеется необходимая дисциплина соблюдения установленных правил. Эффективные методики и процессы **институционализируются** (устанавливаются), что обеспечивает возможность повторения успеха предыдущих проектов в той же прикладной области.

Ключевые области процесса разработки ПО этого уровня:

- Управление требованиями (Requirements management).
- Планирование проекта разработки ПО (Software project planning).
- Отслеживание хода проекта и контроль (Software project tracking and oversight).
- Управление субподрядчиками разработки ПО (Software subcontract management).
- Обеспечение уверенности в качестве разработки ПО (Software quality assurance).
- Управление конфигурацией продукта (Software configuration management).

Определенный уровень (Defined Level – Level 3).

Уровень характеризуется детализированным методологическим подходом к управлению (то есть описаны и закреплены в документированной политике типичные действия, необходимые для многократного повторения: роли и ответственность участников, стандартные процедуры и операции, порядок действий, количественные показатели и метрики процессов, форматы документов и пр.).

Для создания и поддержания методологий в актуальном состоянии в организации уже подготовлена и постоянно функционирует специальная группа. Компания регулярно проводит тренинги для повышения профессионального уровня своих сотрудников.

Начиная с этого уровня, организация практически перестает зависеть от личностных качеств конкретных разработчиков и не имеет тенденции опускаться на нижестоящие уровни. Эта независимость обусловлена продуманным механизмом постановки задач, планирования мероприятий, выполнения операций и контроля исполнения.

Управленческие и инженерные процессы документированы, стандартизированы и интегрированы в унифицированную для всей организации технологию создания ПО. Каждый проект использует утвержденную версию этой технологии, адаптированную к особенностям текущего проекта.

Ключевые области процесса разработки ПО этого уровня:

- Цель упорядочивания работы организации (Organization Process Focus).
- Определение (стандартного) процесса организации (Organization Process Definition).
- Программа обучения (Training Program).
- Интегрированное управление разработкой ПО (Integrated Software Management).
- Технология разработки программных продуктов (Software Product Engineering).
- Межгрупповая координация (Intergroup Coordination).
- Экспертные (совместные) оценки коллег (Peer Reviews).

Управляемый уровень (Managed Level – Level 4).

Уровень, при котором разработаны и закреплены в соответствующих нормативных документах количественные показатели качества. Более высокий уровень управления проектами достигается за счет уменьшения отклонений различных показателей проекта от

запланированных. При этом систематические изменения в производительности процесса (тенденции, тренды) можно выделить из случайных вариаций (шума) на основании статистической обработки результатов измерений по процессам, особенно в хорошо освоенных и достаточно формализованных процессных областях.

Ключевые области процесса разработки ПО этого уровня:

- Количественное управление процессом (Quantitative Process Management).
- Управление качеством ПО (Software Quality Management).

Оптимизирующий уровень (Optimizing Level – Level 5).

Для этого уровня мероприятия по совершенствованию рассчитаны не только на существующие процессы, но и на внедрение, использование новых технологий и оценку их эффективности. Основной задачей всей организации на этом уровне является постоянное совершенствование существующих процессов, которое в идеале направлено на предотвращение известных ошибок или дефектов и предупреждение возможных. Применяется механизм повторного использования компонентов от проекта к проекту (шаблоны отчетов, форматы требований, процедуры и стандартные операции, библиотеки модулей программных средств).

Ключевые области процесса разработки ПО этого уровня:

- Предотвращение дефектов (Defect Prevention).
- Управление изменением технологий (Technology Change Management).
- Управление изменением процесса (Process Change Management).

CMM определяет следующий минимальный набор требований: реализовать 18 ключевых областей процесса разработки ПО, содержащих 52 цели, 28 обязательств компании, 70 возможностей выполнения (гарантий компании) и 150 ключевых практик.

В результате аудита и аттестации компании присваивается определенный уровень, который при последующих аудитах в дальнейшем может повышаться или понижаться. Следует отметить, что каждый следующий уровень в обязательном порядке включает в себя все ключевые характеристики предыдущих. В связи с этим сертификация компании по одному из уровней предполагает безусловное выполнение всех требований более низких уровней.

К преимуществам модели SEI SW-CMM относится то, что она ориентирована на организации, занимающиеся разработкой программного обеспечения. В данной модели удалось более детально проработать требования, специфичные для процессов, связанных с разработкой ПО. По этой причине в SEI SW-CMM приведены не только требования к процессам организации, но и примеры реализации таких требований.

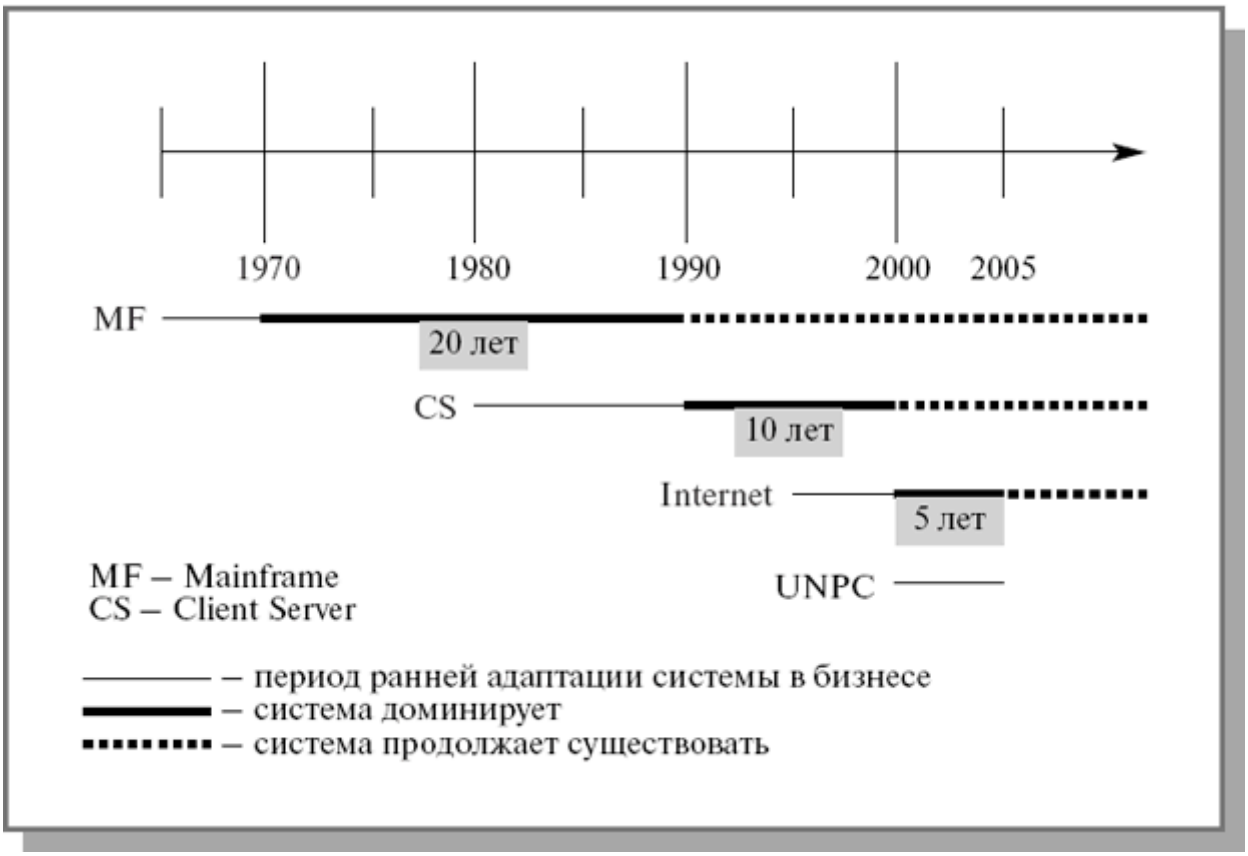
Основной же недостаток SW-CMM заключается в том, что модель не авторизована в качестве стандарта ни международными, ни национальными органами по стандартизации. Впрочем, CMM давно уже стала промышленным стандартом де-факто.

К недостаткам данной модели необходимо отнести также большие внешние накладные расходы на приведение процессов компании в соответствие модели CMM, нежели к моделям ISO 9000. Это связано с меньшей распространенностью модели в мире, меньшим количеством консалтинговых органов и экспертов и, в результате, с гораздо большими внешними затратами на консалтинг и на подтверждение соответствия процессов независимой третьей стороной. Тем не менее, CMM, несомненно, полезнее ISO 9000.

CASE-технологии

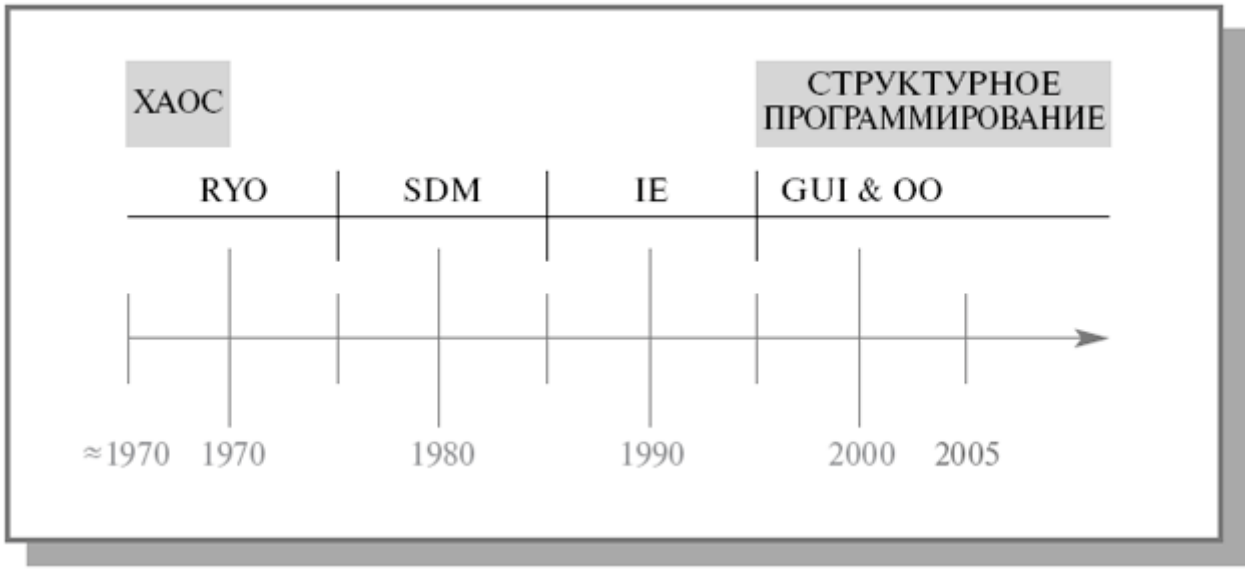
На протяжении всей истории программирования программные проекты все более и более усложнялись, объем работ стремительно увеличивался (особенно это проявилось в бизнес-приложениях), возникла потребность в таком универсальном средстве, которое могло бы помочь как-то структурировать, упорядочить и даже автоматизировать создание ПО. Проблема была глубже — необходимо было как-то объединить заказчиков, разработчиков, программистов, пользователей — причем в условиях постоянно меняющейся ситуации. А для того, чтобы о чем-то договориться, нужен какой-то общий язык. Традиционные языки программирования в силу малой наглядности, избыточности и многословия для этой роли не подходили, и, в конце концов, стали предприниматься попытки создания четкого графического языка. Реализации графических языков и методологии их использования способствовали появлению программно-технологических средств специального класса — CASE-средств [21]. Аббревиатура CASE расшифровывается как Computer-Aided Software Engineering, т.е. разработка ПО с помощью компьютера.

Изобразим ось времени, в качестве начальной точки возьмем 1965 год. Именно тогда начался период раннего внедрения компьютеров в бизнесе. В то время применялись в основном мэйнфреймы (mainframes) — большие ЭВМ коллективного пользования.



увеличить изображение

С 1970 по 1990 год мэйнфреймы доминировали на рынке. Мэйнфрейм в то время – это, прежде всего, IBM 360/370. Все данные хранились и обрабатывались на одной очень большой ЭВМ, пользователи работали за экранами терминалов, которые могли только отображать информацию, но никакой обработки не производили. В СССР скопировали мэйнфреймы под именем ЕС ЭВМ. Были клоны IBM 360/370 в Англии, ФРГ и Японии.



увеличить изображение

В середине 1980-х годов появляется альтернатива мэйнфреймам: системы клиент-сервер (client-server), которые занимали ведущее положение с 1990 по 2000-е годы. В этом случае на клиентском рабочем месте уже могла осуществляться какая-то обработка данных, например, их форматирование, распаковка, простые расчеты. Основные вычисления по-прежнему выполнялись в центре (на сервере).

В середине 1990-х начинается адаптация Internet-систем, которые доминируют с середины 2000-го года. Сейчас начинают появляться новые системы, которым раньше не было аналогов; их принято называть "не ПК" (un-PC). В их число можно включить мобильные телефоны, карманные компьютеры, системы управления автомобилем (например, стоимость программного обеспечения автомобиля Ford уже превышает стоимость железа, из которого он сделан) и многое другое.

Характерно, что с развитием компьютерных технологий они становятся дешевле, а сами компьютеры – более доступными (особенно это проявилось в 1980-90-е гг.); как следствие, создается больше систем на базе каждой конкретной технологии.

Прослеживается такая закономерность: период доминирования каждой последующей технологии сокращается вдвое; одновременно все более многочисленными и масштабными становятся создаваемые системы.

Посмотрим на эволюцию подхода к проектированию систем.

Развитие методологии проектирования

С середины 1960-х до середины 1970-х годов программы преимущественно писались по принципу RYO (Roll Your Own), т. е. каждый писал так, как хотел и как умел — не было определенных подходов к процессу разработки ПО. В середине 70-х годов прошлого столетия возникла идея структурного программирования (SDM, Structure Design Methodology), происходит развитие методологии программирования и технологии моделирования. В Европе главным идеологом структурного программирования считается Дейкстра, а в США – ДеМарко. Разработчики приходят к пониманию, что систему (программу) можно описывать без использования конструкций языка программирования, а на более абстрактном уровне. В этот период времени программисты (и не только) поняли, что моделирование играет немаловажную роль, а вместе с ним и правила для моделирования. К этому моменту относится и введение блок-схем (flow charts). Для этого этапа развития

технологии моделирования характерно, что центром программного продукта является процесс, а не данные (для хранения данных использовались индексные файлы и иерархические базы данных).

В середине 1980-х годов происходит очередной прорыв: появляются реляционные базы данных, один из авторов которых — **James Martin**. Главной частью программного продукта становятся данные и базы данных. Теоретиками была создана реляционная алгебра [21], ставшая основой для построения реляционных баз данных. Отсюда и новый подход к моделированию систем — IE (Information Engineering — методы и средства проектирования прикладных и информационных программ), в которых за основу принимаются не процессы, а структуры обрабатываемых данных.

С появлением персональных компьютеров на уровне систем клиент-сервер развивается графический интерфейс (GUI, Graphic User Interface). В связи с этим рождается объектно-ориентированный подход к проектированию (ОО), объединивший в одной сущности программу и данные.

Стоит отметить два вида объектно-ориентированного программирования, которые по сей день сосуществуют, хотя и велись споры о правильности каждого из них и нелогичности другого. Object Action заключается в том, что сначала выбирается объект, а потом уже реализовываются его действия, которые впоследствии будут использованы. Action Object же, наоборот, предлагает продумать необходимые действия, а затем выбрать, какой именно объект будет их реализовывать.

Главные составляющие CASE-продукта таковы:

- **методология (Method Diagrams)**, которая задает единый графический язык и правила работы с ним. CASE-технологии обеспечивают всех участников проекта, включая заказчиков, единым, строгим, наглядным и интуитивно понятным графическим языком, позволяющим получать обозримые компоненты с простой и ясной структурой. При этом программы представляются двумерными диаграммами (которые проще в использовании, чем многостраничные описания), позволяющими заказчику участвовать в процессе разработки, а разработчикам — общаться с экспертами предметной области, разделять деятельность системных аналитиков, проектировщиков и программистов, облегчая им защиту проекта перед руководством, а также обеспечивая легкость сопровождения и внесения изменений в систему.
- **графические редакторы (Graphic Editors)**, которые помогают рисовать диаграммы; возникли с распространением PC и GUI. Этими двумя составляющими (так называемые upper case технологии) CASE-технологии поначалу и были ограничены. Диаграммы стало легко рисовать, их появилось множество, но пользы от них было мало – проектирование было развито лишь на уровне рисования. Существовало много проблем: никто не знал все используемые в тот момент технологии (не мог писать и для мэйнфреймов, и для клиента, и для сервера); неясно было, как объединять написанное для разных платформ.
- **генератор**: по графическому представлению модели можно сгенерировать исходный код для различных платформ (так называемая low case часть CASE-технологии). Генерация программ позволяет автоматически построить до 85-90% объектного кода или текстов на языках высокого уровня, но только для хорошо формализуемых частей программы (прежде всего, для описания баз данных и для задания форм ввода-вывода информации). Сложная обработка, как обычно, может быть описана с помощью ручного программирования.
- **репозиторий**, своеобразная база данных для хранения результатов работы программистов (сложилась парадоксальная ситуация: к тому моменту базами данных

пользовались все, кроме программистов), происходит переход от "плоских" файлов к системе хранения информации о разработке проекта.

Примеры CASE-средств

Если сначала диаграммы рисовались вручную, то в середине 1980-х годов появляются первые продукты, реализующие CASE-технологию. Компания TI (Texas Instruments) выпускает продукт IEF (Information Engineering Facility), компания KW (Knowledge Ware) создает ADW. Целевыми платформами обоих продуктов были только мэйнфреймы,что являлось их основным недостатком. В 1986 году начинаются разработки продукта HPS (High Productivity System), а в 1990 году образуется компания SEER, которая выпускает HPS на рынок. В 1992-1993гг. по заказу этой компании мы полностью переписали HPS, сохранив их замечательные бизнес-идеи.

Технологии оказались востребованными на рынке: на 1995 год объемы продаж IEF равнялись 200 млн. долл., ADW — 150 млн. долл., HPS — 120 млн. долл. У продукта HPS был более высокий уровень генерации кода (lower case), но более слабый уровень методологии и графических редакторов (upper case).

Годы	Организации	Тип систем
1965 — 1975	немногие	Изолированные
1975 — 1985	многие	Изолированные
1985 — 1995	многие	Enterprise level
1995 — 2000	многие	Department level
с 2000	многие	SOA

В середине 1980-х годов мы также разработали технологию RTST, которая включала в себя все перечисленные выше компоненты, в том числе генератор в Алгол 68. В чем-то мы даже превзошли американцев, поскольку они умели генерировать только экранные формы, базы данных и стандартные действия CRUD (create, read, update, delete), а мы в дополнение к этому генерировали и бизнес-логику на основе SDL-диаграмм. Эта технология до сих пор активно используется при производстве ПО телефонных станций, но ни о каких продажах в нашей стране пиратского ПО мы и не думали.

Рассмотрим таблицу, характеризующую основные уровни развития использования CASE-технологии в бизнесе:

К середине 1980-х годов системы и проекты разрастаются. Поначалу немногие организации могли создавать мощные системы. Затем, с распространением мэйнфреймов, количество таких организаций увеличилось, но разработка систем по-прежнему никак не координировалась. Для ранних уровней развития проектирования характерная черта – изолированность "островков автоматизации" в "море" организации (этап с 1965 по 1985 гг.). Позже, при помощи IEF, ADW, HPS (на базе IE и CASE-технологий), преобладающим становится централизованное планирование в рамках всей организации (enterprise level). В то время проекты требовали участия 500-1000 человек в течение 1-2 лет.

В середине 1990-х годов такая "гигантомания" признается экономически нецелесообразной, разработка систем переходит на уровень department level, когда проектирование и планирование осуществляются в рамках одного отдела (департамента); для этого требуется работа 2-5 человек в течение одного-двух месяцев. В этот момент были очень популярны средства быстрого прототипирования – Rapid Application Development (RAD), такие как Power Builder, FORTE, Sun Microsystems Powery.

Средства быстрого прототипирования отодвинули на второй план CASE-средства, но, в свою очередь, были задавлены агрессивной политикой Microsoft. На какое-то время самым популярным языком стал Visual Basic. В последнее время наблюдается переход к уровню SOA (Service-Oriented Architecture), основанному на ОО-моделировании.

К 2000 году от графической разработки моделей практически отказались и, как оказалось, зря.

С 1998 года стала набирать силу технология Rational Rose, основанная на объектно-ориентированном подходе и на последовательно уточняющихся графических моделях. Первоначально промышленному использованию такого подхода мешало разнообразие похожих, но различающихся моделей. Заслугой фирмы Rational можно считать то, что она сумела объединить трех классиков объектного проектирования ("three amigos" Рэмбо, Буча и Якобсона), которые создали универсальный язык ОО-моделирования UML — Universal Modeling Language [22].

Технология стала настолько успешной и популярной, что IBM купила фирму Rational Software более чем за 2 млрд. долларов и включила Rational Rose в свою линейку продуктов.

Аналогичный путь проделала технология компании Together Soft, правда, значительно более скромная по своим возможностям. Их купила Borland за 56 млн. долларов.

Что же случилось с первыми CASE-технологиями?

В 1996 году известный собиратель устаревших средств – компания Sterling Software скупила почти все CASE-средства, кроме HPS фирмы SEER Technologies. Идея бизнеса фирмы Sterling Software понятна – есть сотни компаний, использующих CASE-средства, купленные в начале 1990-х годов. В США принято, что ежегодное сопровождение ПО стоит до 20% его продажной цены. Таким образом, "могильщик" Sterling Software собирал огромные деньги только на сопровождении, не ведя никаких новых разработок. Один раз они захотели расширить свой рынок и заказали нам конвертор Cobol →Coolgen (новое название, которое они дали IEF). Мы-то, конечно, все сделали как надо, но, на наш взгляд, компания допустила стратегическую ошибку: лучше было бы заказать нам конвертор Coolgen →Cobol, т.к. большинство пользователей старых средств хотят освободиться от зависимости от них.

В конце концов, компания Computer Associates купила Sterling Software, и они заказали нам конвертор ADW →Cobol, а потом и еще более пяти проектов по реинжинирингу ADW-проектов в Cobol.

Параллельно с этим два конкурента — SEER Technologies и Relativity Technologies — заказали нам конверторы HPS →Cobol. С их стороны был определенный риск, что жизненно важная для бизнеса информация утечет через нас к конкурентам, но мы их, разумеется, не подвели, и сейчас на рынке активно продаются оба варианта.

Таким образом, мы познакомились и так или иначе приняли участие в создании всех массовых CASE-средств. Этот опыт оказался бесценным для наших собственных работ.