Методологии разработки ПО

SCRUM, KANBAN, DSDM, MSF, RUP, AUP

SCRUM

Методология, предназначенная для небольших команд (до 10 человек).

Весь проект делится на спринты продолжительностью 30 дней каждый.

Выбирается список функций системы, которые планируется реализовать в течение следующего спринта.

Роли SCRUM

- Scrum Master
- Product Owner
- Team

Scrum Master

Скрам Мастер ведет Daily Scrum Meeting и отслеживает прогресс команды при помощи Sprint Backlog, отмечая статус всех задач в спринте.

Как правило, эту роль в проекте играет менеджер проекта или тимлид.

Скрам Мастер не раздает задачи членам команды. Команда является самоорганизующейся и самоуправлямой.

Product Owner

Человек, отвечающий за разработку продукта. Как правило, это:

- product manager для продуктовой разработки,
- менеджер проекта для внутренней разработки,
- представитель заказчика для заказной разработки.

Product Owner

Product Owner ставит задачи команде, но он не вправе ставить задачи конкретному члену проектной команды в течении спринта.

Координирует и приоритизирует Product Backlog.

Команда (Team)

Команда берет на себя обязательства по выполнению объема работ на спринт перед Product Owner.

Работа команды оценивается как работа единой группы.

В Scrum вклад отдельных членов проектной команды не оценивается, так как это разваливает самоорганизацию команды.

Product Backlog

Приоритезированный список имеющихся на данный момент бизнес-требований и технических требований к системе.

Product Backlog постоянно пересматривается и дополняется - в него включаются новые требования, удаляются ненужные, пересматриваются приоритеты.

Product Backlog

Feature	Estimation, man*it	Priority
Интеграция с IntHR		
Синхронизация сотрудников, орг структуры, позиций и т.д. с учетом multisite	2	Must
ФИН		
Расчет утилизации и проч. финансовых показателей	1	Must
Расчет затрат по реальным зарплатам	1	Must
Редактирование справочных таблиц	1	Must
Закрытый период		
Заказчики		
Внутренние ставки		
Закрытый период по проектам	1	Should
Поддержка разных типов проектов	1	Should
Административный проект подразделения		
Проект обучения		
Multisite		
Синхронизация данных между серверами, управление синхронизацией	4	Must
Отчеты уровня компании	2	Must
Внешний бюджет		
Доработка интерфейса (общий стиль, тулбары, usability, AJAX)	2	Should
Additional view	2	Should
By Month		
By Person		
Прочие доходы	1	Should
Автоматизация PSR	1	Must

Sprint Backlog

Содержит функциональность, выбранную Product Owner из Product Backlog.

Все функции разбиты по задачам, каждая из которых оценивается командой.

Каждый день команда оценивает объем работы, который нужно проделать для завершения задач.

Sprint Backlog

		Days Left in Sprint	15	13	10	8	
Who	Description		7/22/2002	7/24/2002	7/26/2002	7/31/2002	
		Total Estimated Hours:	554	458	362	270	0
-	**User's Guide**		-	-	-	-	-
SM	Start on Study Variable chapter first draft		16	16	16	16	
SM	Import chapter first draft		40	24	6	6	
SM	Export chapter first draft		24	24	24	6	
	Misc. Small Bugs						
JM	Fix connection leak		40				
JM	Delete queries		8	8			
JM	Delete analysis		8	8			
TG	Fix tear-off messaging bug		8	8			
JM	View pedigree for kindred column in a result set		2	2	2	2	
AM	Derived kindred validation		8				
	Environment						
TG	Install CVS		16	16			
TBD	Move code into CVS		40	40	40	40	
TBD	Move to JDK 1.4		8	8	8	8	
	Database						
KH	Killing Oracle sessions		8	8	8	8	
KH	Finish 2.206 database patch		8	2			
KH	Make a 2.207 database patch		8	8	8	8	
KH	Figure out why 461 indexes are created		4				

Спринт (Sprint)

Длительность составляет 1 месяц (30 дней)

Результатом Sprint является готовый продукт (build), который можно передавать (deliver) заказчику (по крайней мере, система должна быть готова к показу заказчику).

В течение спринта делаются все работы по сбору требований, дизайну, кодированию и тестированию продукта.

Планирование спринта

Митинг первый

Участники: команда, Product Owner, Scrum Master, пользователи, менеджмент

Цель: Определить цель спринта (Sprint Goal) и Sprint Backlog -функциональность, которая будет разработана в течение следующего спринта для достижения цели спринта.

Митинг второй

Участники: Скрам Мастер, команда

Цель: определить, как именно будет разрабатываться определенная функциональность. Для каждого элемента Sprint Backlog определяется список задач и оценивается их продолжительность.

Daily Scrum Meeting

Этот митинг проходит каждое утро в начале дня.

Предназначен для того, чтобы все члены команды знали, кто и чем занимается в проекте.

Длительность этого митинга строго ограничена и не должна превышать 15 минут.

Скрам митинг

- Что сделано вчера?
- Что будет сделано сегодня?
- С какими проблемами столкнулся?

KANBAN

гибкая методология разработки программного обеспечения, ориентированная на задачи.

Основные правила

- визуализация разработки:
  - разделение работы на задачи;
  - использование отметок о положение задачи в разработке;
- ограничение работ, выполняющихся одновременно, на каждом этапе разработки;
- измерение времени цикла (среднее время на выполнение одной задачи) и оптимизация процесса.

Преимущества KANBAN

- уменьшение числа параллельно выполняемых задач значительно уменьшает время выполнения каждой отдельной задачи;
- быстрое выявление проблемных задач;
- вычисление времени на выполнение усредненной задачи

Разница между Канбан и SCRUM

- В Канбан нет таймбоксов ни на что (ни на задачи, ни на спринты)
- В Канбан задачи больше и их меньше
- В Канбан оценки сроков на задачу опциональные или вообще их нет
- В Канбан «скорость работы команды» отсутствует и считается только среднее время на полную реализацию задачи

DYNAMIC SYSTEM DEVELOPMENT METHOD

методика ведения проектов, нацеленная на получение быстрой отдачи оставаясь в рамках бюджета

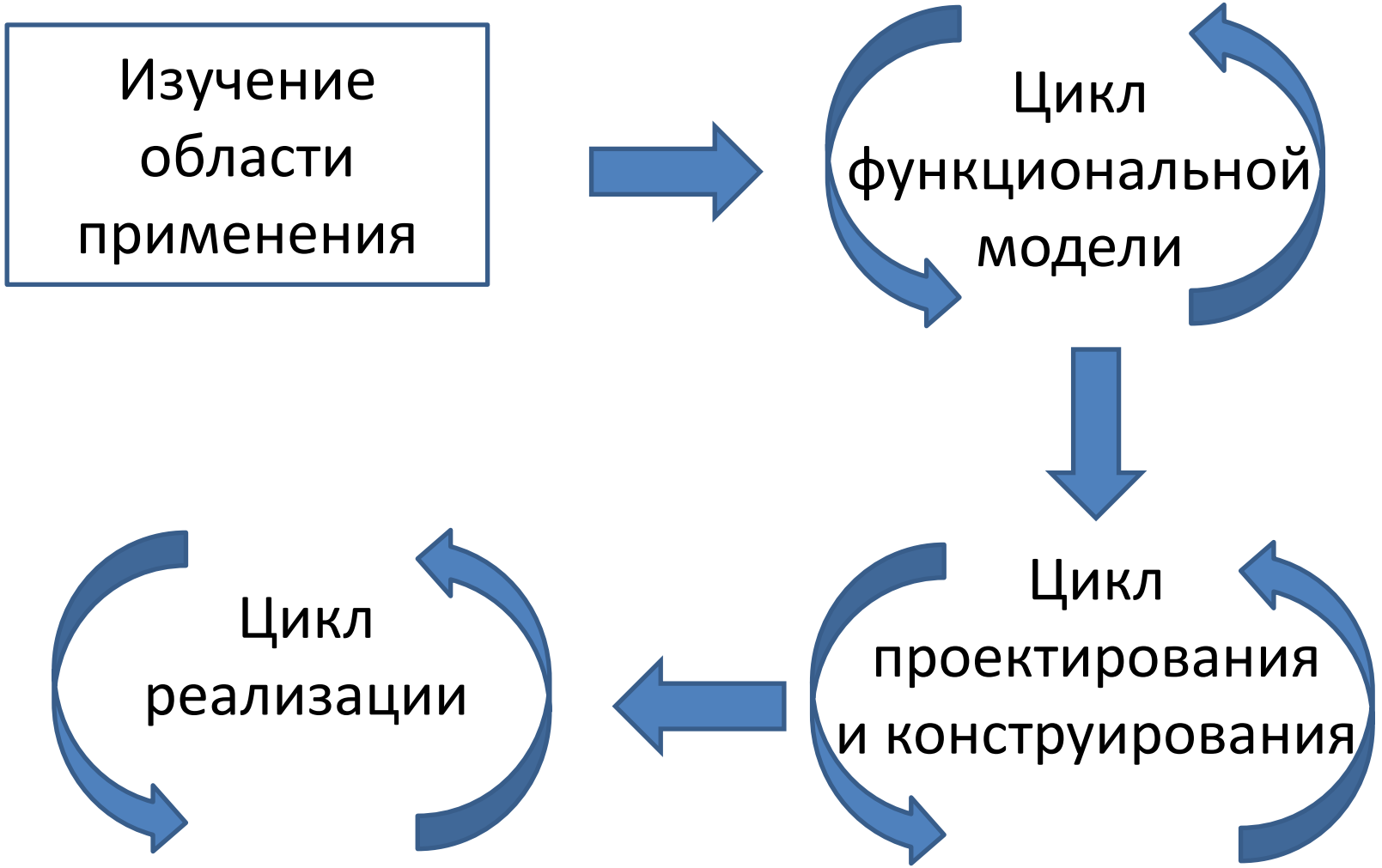
DSDM

Все начинается с изучения осуществимости программы и области ее применения.

Принципы DSSM

- Активное вовлечение пользователей
- Команда должна быть уполномочена принимать решения
- Частая поставка результатов
- Соответствие бизнес-задаче
- Итеративная и инкрементная разработка
- Любые действия могут быть отменены
- Требования устанавливаются на высоком уровне
- Тестирование интегрировано в жизненный цикл разработки
- Обязательное сотрудничество между всеми участниками

Три взаимосвязанных цикла



Цикл функциональной модели

Создание аналитической документации и прототипов.

Разрабатывается и анализируется функциональный прототип системы.

Функциональный прототип показывает, какими функциями должна обладать система и как она должна их выполнять.

Цикл проектирования и разработки

Приведение системы в рабочее состояние.

Продукт проектируется и разрабатывается.

Причем разработка сразу ведется на должном уровне качества.

Обычный проект содержит несколько итераций разработки.

Цикл реализации

Обеспечение развертывания программной системы.

Продукт подготавливается к выпуску, разрабатывается пользовательская документация, проводится обучение пользователей и т. д.

Проект может включать в себя несколько итераций реализации.

Роли DSDM

- **Менеджер проекта (Project Manager)** - обеспечивает общее руководство проектом.

- **Провидец (Visionary)*** - следит за соответствием проекта коммерческим целям и задачам. Провидцем часто является топ-менеджер, который инициировал проект.

- **Чемпион проекта (Project Champion или Executive Sponsor)*** - обладает возможностями и обязанностями по распоряжению ресурсами и фондами, которые необходимы данному проекту. Чемпион проекта несет ответственность за принятие любых решений, связанных с проектом.

Роли DSDM

- **Лидер команды (Team Leader)** - руководит командой разработчиков и обеспечивает эффективность ее работы.
- **Технический координатор (Technical Co-ordinator)** - отвечает за разработку архитектуры продукта. Технический координатор также отвечает общее техническое состояние проекта.
- **Разработчик (Developer)** - участвует в анализе требований, моделировании и проектировании. Очевидно, что основной обязанностью разработчика является программирование.
- **Тестировщик (Tester)** - отвечает техническое тестирование продукта.

Роли DSDM

- **Представительный пользователь (Ambassador User)** - отвечает за то, чтобы разработчики вовремя получали обратную связь со стороны пользователей.
- **Пользователь-консультант (Advisor User)*** - привносит в проект знание по некоторому аспекту использования разрабатываемого продукта.
- **Секретарь (Scribe)** - отвечает за протоколирование всех соглашений и решений, принятых во время семинаров.
- **Посредник (Facilitator)** - отвечает за проведение семинаров. Посредник также отвечает за эффективность коммуникации между всеми членами команды.

MICROSOFT SOLUTIONS FRAMEWORK

методология разработки программного обеспечения, предложенная корпорацией Microsoft

Базовые принципы

- единое видение проекта
- управление компромиссами
- гибкость
- концентрация на бизнес-приоритетах
- поощрение свободного общения внутри проекта
- создание базовых версии

Роли MSF

- управляющий программой (program manager) — разработка архитектуры решения, административная служба;
- разработчик (developer) — разработка приложений и инфраструктуры, технологические консультации;
- тестировщик (QAE) — планирование, разработка тестов и отчетность по тестам;

Роли MSF

- управляющий выпуском (release manager) — инфраструктура, сопровождение, бизнес-процессы, выпуск готового продукта;
- взаимодействующий с пользователем (user experience) — обучение, эргономика, графический дизайн, техническая поддержка;
- управляющий продуктом (product manager) — бизнес-приоритеты, маркетинг, представительство интересов заказчика.

Объединение ролей

- Роль разработчика не может быть объединена ни с какой другой ролью.

- Избежание сочетания ролей, имеющих предопределенные конфликты интересов.

Особенности MSF

MSF предлагает проверенные методики для планирования, проектирования, разработки и внедрения успешных IT-решений.

Благодаря своей гибкости, масштабируемости и отсутствию жестких инструкций MSF способен удовлетворить нужды организации или проектной группы любого размера.

Методология MSF

Методология MSF состоит из принципов, моделей и дисциплин по управлению персоналом, процессами, технологическими элементами и связанными со всеми этими факторами вопросами, характерными для большинства проектов.

RATIONAL UNIFIED PROCESS

методология разработки программного обеспечения, созданная компанией Rational Software

6 основных принципов

- компонентная архитектура, реализуемая и тестируемая на ранних стадиях проекта;
- работа над проектом в сплочённой команде, ключевая роль в которой принадлежит архитекторам;
- ранняя идентификация и непрерывное устранение возможных рисков;

6 основных принципов

- концентрация на выполнении требований заказчиков к исполняемой программе;
- ожидание изменений в требованиях, проектных решениях и реализации в процессе разработки;
- постоянное обеспечение качества на всех этапах разработки проекта.

Жизненный цикл RUP

Полный жизненный цикл разработки продукта состоит из четырех фаз, каждая из которых включает в себя одну или несколько итераций:

- начальная стадия (inception);
- стадия разработки (elaboration);
- стадия конструирования (construction);
- стадия ввода в действие (transition).

Начальная стадия (Inception)

- Формируются видение и границы проекта.
- Создается экономическое обоснование (business case).
- Определяются основные требования, ограничения и ключевая функциональность продукта.
- Создается базовая версия модели прецедентов.
- Оцениваются риски.

Уточнение (Elaboration)

- Документирование требований (включая детальное описание для большинства прецедентов).
- Спроектированную, реализованную и оттестированную исполняемую архитектуру.
- Обновленное экономическое обоснование и более точные оценки сроков и стоимости.
- Сниженные основные риски.

Построение (Construction)

Реализация большей части функциональности продукта.

Фаза Построение завершается первым внешним релизом системы и вехой начальной функциональной готовности (Initial Operational Capability).

Внедрение (Transition)

- Бета-тестирование.
- Обучение пользователей.
- Определение качества продукта.
- Передача продукта заказчику.

Гибкость формализации

- Можно по окончании каждого этапа и каждой итерации создавать все требуемые документы и достигнуть максимального уровня формализации.

- Можно создавать только необходимые для работы документы, вплоть до полного их отсутствия.