Рациональный Унифицированный Процесс (Rational Unified Process, RUP) - это процесс-ориентированная методология разработки программного обеспечения, разработанная фирмой Rational Software (позднее приобретенной IBM). RUP предоставляет стройный и систематический подход к разработке информационных систем, учитывая лучшие практики и методы управления проектами. Вот основные характеристики методологии RUP:

1. **Процессориентированность**: RUP предоставляет обширное описание процессов разработки, включая все этапы от начала до конца жизненного цикла проекта. Эти процессы состоят из задач, ролей и артефактов.
2. **Артефакты и документация**: RUP акцентирует внимание на создании документации, такой как планы, спецификации, диаграммы и другие артефакты. Эта документация служит средством коммуникации между участниками проекта.
3. **Роли и обязанности**: RUP определяет различные роли в процессе разработки, такие как аналитик, архитектор, разработчик, тестировщик и т.д. Каждая роль имеет свои обязанности и задачи.
4. **Итеративность и инкрементальность**: RUP поддерживает итеративный и инкрементальный подход к разработке. Проект делится на короткие итерации, каждая из которых привносит новый функционал или улучшения.
5. **Архитектурная целостность**: Методология акцентирует внимание на создании устойчивой и надежной архитектуры системы. Она подразумевает проектирование и документирование архитектурных решений.
6. **Управление изменениями**: RUP предоставляет инструменты для эффективного управления изменениями в проекте. Это важно, учитывая, что требования могут меняться в ходе разработки.
7. **Управление качеством и тестирование**: RUP включает в себя процессы управления качеством и тестирования. Это позволяет обеспечить высокое качество разрабатываемых систем.
8. **Артефакты UML**: RUP часто использует язык моделирования UML (Unified Modeling Language) для создания диаграмм, описывающих архитектуру и дизайн системы.
9. **Гибкость**: RUP можно адаптировать к различным типам проектов и организационным потребностям. Он поддерживает гибкость в выборе процессов и инструментов.
10. **Эффективное управление проектом**: Методология включает в себя инструменты для управления проектом, включая планирование, контроль и отслеживание.

Рациональный Унифицированный Процесс (Rational Unified Process, RUP) организован вокруг процессов и стадий разработки, предоставляя структурированный и методологический подход к проектированию информационных систем. RUP определяет следующие ключевые стадии и процессы:

**1. Начальная стадия (Inception):**

* **Процесс начальной стадии (Inception Process)**: На этой стадии происходит инициализация проекта. Оцениваются его цели, требования и область применения. Выявляются ключевые риски, исследуется техническая и экономическая целесообразность. В результате процесса начальной стадии создается "Обоснование проекта" (Project Vision) и "Исходная архитектура" (Initial Architecture).

**2. Разрабатывающая стадия (Elaboration):**

* **Процесс разрабатывающей стадии (Elaboration Process)**: На этой стадии углубляется анализ требований, проектируется архитектура системы, и разрабатывается детальный план проекта. Осуществляется управление рисками. Результатом разрабатывающей стадии являются "Спецификация системы" (System Specification) и "Архитектурный прототип" (Architectural Prototype).

**3. Строительная стадия (Construction):**

* **Процесс строительной стадии (Construction Process)**: На этой стадии осуществляется фактическое программирование, тестирование и развертывание системы. Происходит углубление в детали реализации. Результатом строительной стадии является "Выполненная система" (System in Operation).

**4. Переходная стадия (Transition):**

* **Процесс переходной стадии (Transition Process)**: На этой стадии система переводится в рабочий режим и внедряется в окружение пользователя. Происходит обучение пользователей и технических специалистов, а также осуществляется поддержка системы.

Метод DSDM (Dynamic Systems Development Method) - это методология разработки программного обеспечения, ориентированная на быструю и гибкую разработку информационных систем. DSDM входит в семейство агильных методологий и акцентирует внимание на сотрудничестве, коммуникации и поставке ценности заказчику. Вот основные характеристики методологии DSDM:

1. **Сотрудничество и коммуникация**: DSDM поддерживает сотрудничество между разработчиками, заказчиками и другими участниками проекта. Эффективная коммуникация считается ключевым фактором успеха.
2. **Итеративность и инкрементальность:** Методология DSDM использует итеративный подход, разбивая проект на короткие временные интервалы (итерации), в течение которых разрабатывается функциональность. Это позволяет быстро внедрять изменения и уточнять требования.
3. **Фиксированные сроки и ресурсы:** DSDM признает важность фиксированных сроков и ресурсов в проекте. Вместо того чтобы менять даты, методология рекомендует управлять объемом функциональности, чтобы соблюсти график.
4. **Сконцентрированность на бизнес-ценности**: Основной целью DSDM является достижение бизнес-ценности через разработку ПО. Процессы разработки и приоритеты определяются в соответствии с бизнес-целями.
5. **Управление рисками:** DSDM уделяет внимание управлению рисками, помогая их выявлять и минимизировать на ранних стадиях проекта.
6. **Роли и обязанности:** Методология определяет роли, такие как спонсор проекта, аналитик, разработчик, тестировщик и т. д., а также их обязанности в рамках процесса разработки.
7. **Обратная связь и обучение:** DSDM поощряет сбор обратной связи от заказчиков и пользователей, а также обучение и улучшение процессов на основе этой обратной связи.
8. **Фазы и процессы:** DSDM определяет фазы и процессы разработки, включая фазу предварительного исследования, моделирование, разработку, проверку и внедрение.
9. **Принципы DSDM:** Методология DSDM строится на нескольких ключевых принципах, таких как активное участие заказчика, вовлечение пользователей, предоставление полномочий командам разработки и другие.

Scrum - это агильная методология разработки программного обеспечения, которая акцентирует внимание на гибкости, сотрудничестве и управлении проектами. Scrum предоставляет каркас для эффективной работы над проектами, включая информационные системы. Вот основные характеристики методологии Scrum:

1. **Роли**: Scrum определяет три ключевые роли:
   * **Product Owner (Владелец продукта)**: Отвечает за определение приоритетов задач и создание списка требований (бэклога).
   * **Scrum Master (Скрам-мастер)**: Отвечает за соблюдение процесса Scrum и устранение преград для команды разработки.
   * **Development Team (Команда разработки)**: Отвечает за выполнение задач из бэклога и доставку функциональности.
2. **Бэклог**: Это список всех задач и требований для проекта, представленный в виде приоритетов. Бэклог постоянно обновляется, и Product Owner определяет, что будет реализовано на следующей итерации.
3. **Итерации (Спринты)**: Проект разбивается на короткие временные интервалы, называемые итерациями (спринтами). Обычно, спринт длится 2-4 недели. За время спринта команда разработки выбирает задачи из бэклога и работает над ними.
4. **Дневные совещания (Daily Stand-up)**: Каждый день члены команды разработки проводят короткое совещание, где обсуждают, что было сделано, что будет сделано и какие препятствия возникли.
5. **Инкрементальное и итеративное развитие**: Каждая итерация привносит новый функционал. Работа ведется итеративно, что позволяет быстро адаптироваться к изменяющимся требованиям.
6. **Приоритеты и гибкость**: Product Owner устанавливает приоритеты задач, что позволяет команде сосредотачиваться на наиболее важных функциональных возможностях. Гибкость вносить изменения в бэклог на каждой итерации.
7. **Оценка скорости команды (Velocity)**: Scrum использует оценку скорости для определения, сколько задач команда разработки способна завершить за итерацию. Это помогает в планировании спринтов.
8. **Управление качеством и тестирование**: Scrum поддерживает высокое качество кода и тестирование на протяжении всего процесса разработки.
9. **Инспекция и адаптация (Inspect and Adapt)**: Scrum основан на постоянном цикле инспекции и адаптации, что позволяет команде учиться на своих ошибках и улучшать процесс.
10. **Скрум-мастер и фасилитаторство**: Скрум-мастер не управляет командой, а скорее помогает ей следовать процессу Scrum и устранять преграды. Это роль фасилитатора, а не менеджера.

Scrum позволяет командам разработки быстро и гибко реагировать на изменяющиеся требования и создавать ценность для заказчика на ранних этапах проекта. Он широко используется в разработке информационных систем и программного обеспечения благодаря своей эффективности и адаптивности.

| **Вес модели** | **Преимущества** | **Недостатки** |
| --- | --- | --- |
| Тяжелые | 1. Процессы рассчитаны на среднюю квалификацию исполнителей. 2. Большая специализация исполнителей. 3. Ниже требования к стабильности команды. 4. Меньше непроизводительных расходов, связанных с управлением проектом, рисками, изменениями, конфигурациями. 5. Объем и сложность выполняемых проектов ограничены. | 1. Упрощенные стадии анализа и проектирования, основной упор на разработку функциональности, совмещение ролей. 2. Неформальные коммуникации. 3. Требуют существенной управленческой надстройки. 4. Более длительные стадии анализа и проектирования. 5. Более формализованные коммуникации. |
| Легкие | 1. Отсутствуют ограничения по объему и сложности выполняемых проектов. | 1. Эффективность сильно зависит от индивидуальных способностей, требуют более квалифицированной, универсальной и стабильной команды |