Содержание

1 Информационные ресурсы, идентификационная информация, идентификационные данные, идентификационный атрибут, идентификация

Информационные ресурсы — это отдельные документы и отдельные массивы документов, документы и массивы документов в информационных системах (билиотеках, архивах, фондаъ, банках данных, других информационных системах).

Идентификационная информация — это совокупность значений идентификационных атрибутов, которая связана с конкретным субъектом доступа или конкретным объектом доступа.

Идентификационные данные — это совокупность идентификационных атрибутов и их значений, которая связана с конкретным субъектом доступа или конкретным объектом доступа.

Идентификационный атрибут — это атрибут, который характеризует субъект доступа или объект доступа и может быть использован для его распознавания.

Идентификация — это действия по присвоению субъектам и объектам доступа идентификаторов и (или) по сравнению предъявляемого идентификатора с перечнем присвоенных идентификаторов.

1.1 Дополнительная (желательная к прочтению и повторению информация)

Идентификация, аунтификация, авторизация

1.2 Сопутствующая информация

Субъект доступа — это одна из сторон информационного взаимодействия, которая инициирует получение и получает доступ.

Субъектами доступа могут являться как физические лица (пользователи), так и ресурсы стороны информационного взаимодействия, а также вычислительные процессы, инициирующие получение и получающие доступ от их имени.

Объект доступа — это одна из сторон информационного взаимодействия, предоставляющая доступ.

2 Аунтефикационная информация, аунтефикация. Многофакторная аунтефикация. Типы аунтефикационных данных

Аунтефикационная информация — это информация, используемая при аунтефикации субъекта доступа или объекта доступа.

Аунтефикация — это действия по проверке подлинности субъекта доступа и (или) объекта доступа, а так же по проверке принадлежности субъекту и (или) объекту доступа предъявленного идентификатора доступа или аунтефикационной информации.

Многофакторная аунтефикация — это аунтефикация, при выполнении которой используется не менее двух различных факторов аунтефикации.

Типы аунтификационных данных (пока что информация не найдена).

2.1 Дополнительная (желательная к прочтению и повторению информация)

Идентификация, аунтификация, авторизация

2.2 Сопутствующая информация

Субъект доступа — это одна из сторон информационного взаимодействия, которая инициирует получение и получает доступ.

Субъектами доступа могут являться как физические лица (пользователи), так и ресурсы стороны информационного взаимодействия, а также вычислительные процессы, инициирующие получение и получающие доступ от их имени.

Объект доступа — это одна из сторон информационного взаимодействия, предоставляющая доступ.

3 Авторизация. Определение, практическое значение и применение

Авторизация — это предоставление определённому лицу или группе лиц прав на выполнение определённых действий.

4 Классические модели процесса: водопадная модель, спиральная модель. Фазы и виды дкятельности.

4.1 Водопадная модель

Водопадная модель является одной из самых старых методологий разработки. Данная методология зиждится на следующих прнципах:

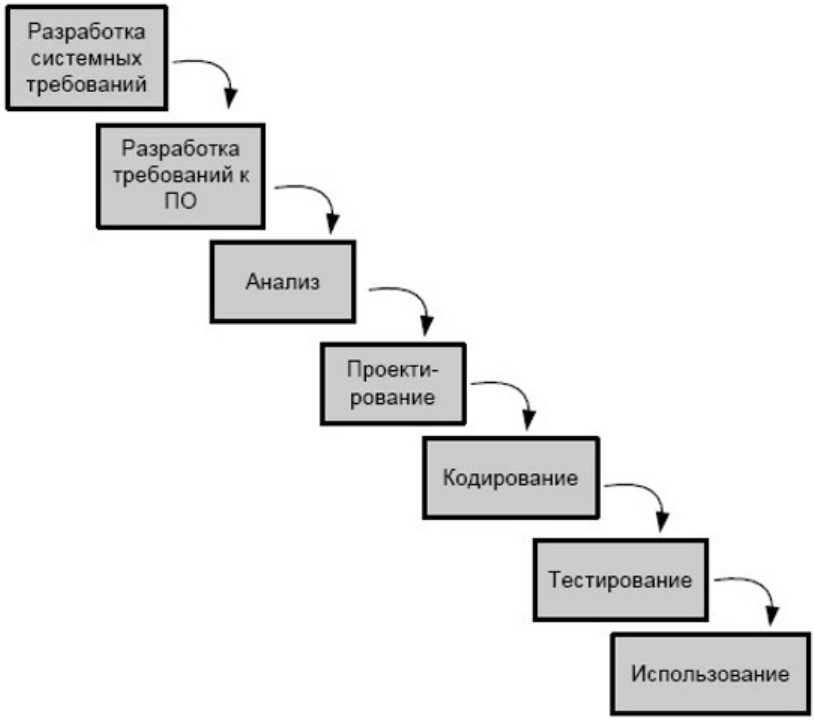
- Полное планирование всех фаз разработки и жёсткое следование ему. В водопадной модели всё планируется и прописывается заранее, каждый этап описан и спланирован, никакие изменения не вносятся до завершения проекта;
- Последовательное выполнение фаз без возможности вернуться на предыдущую фазу;
- Не предполагается, что заказчик имеет доступ к проекту до его сдачи, также предполагается, что правки вноститься не будут.

Плюсы водопадной модели:

- Чёткость и предсказуемость. Всё от требований заказчика, до способов разработки и сроков определено заранее, все следуют плану;
- Понятно, что и как делать. Есть полное ТЗ, которому следует команда, кроме того каждый этап и требования к нему прописаны;
- Подходит крупным проектам, где важно контролировать бюджет. Перед разработкой стоимость и сроки проекта уже известны;
- Подходит для типовых проектов, знакомых команде;
- Заказчик не участвует в каждом этапе разработки. Удобно для тех заказчиков, кто значет чего хочет и у него нет времени для связи с проектом.

Минусы водопадной модели: на мой взгляд все минусы водопадной модели заключаются в невозможности заранее всё спланировать для непредсказуемых проектов или заказчиков. Водопадная модель работает только в тех случаях, когда есть возможность с высокой долей вероятности предусмотреть все варианты развития проекта и спланировать их.

Фазы водопадной модели:



Применяется водопадная модель в гос сфере, так как там важно всё заранее спланировать, в типовых проектах, когда непредвиденных ситуаций благодаря опыту команды не может случиться и в очень крупных проектах тоже, как правило используются данная методология разработки.

4.2 Спиральная модель

Спиральная модель — одна из наиболее важных моделей жизненного цикла разработки ПО, которая обеспечивает поддержку управления рисками. В схематическом представлении она выглядит как спираль с множеством петель. Точное количество витков спирали не известно и может изменяться от проекта к проекту. Каждый цикл спирали называется фазой. Точное количество этапов, необходимых для разработки продукта, моет варьироваться менеджером проекта в зависимости от рисков проекта.

Радиус спирали в любой точке представляет собой затраты (стоимость проекта) на данный момент,а угловой размер представляет прогресс, достигнутый на текущий момент.

Spiral Model Спиральная модель и архитектура разработки программного обеспечения

Фазы спиральной модели:

Фазы спиральной модели	**Действия, выполненные на этапе**
Определение целей и определение альтернативных решений	требования собираются от клиентов, а цели определяются, разрабатываются и анализируются в начале каждого этапа. Затем в этом квадранте предлагаются альтернативные решения, возможные для фазы.
Выявление и устранение рисков	во втором квадранте оцениваются все возможные решения, чтобы выбрать наилучшее из возможных. Затем выявляются риски, связанные с этим решением, и риски устраняются с использованием наилучшей стратегии. В конце этого квадранта создается прототип для наилучшего возможного решения.
Разработка следующей версии продукта	в третьем квадранте выявленные функции разрабатываются и проверяются посредством тестирования. В конце третьего квадранта доступна следующая версия программного обеспечения.
Обзор и планирование следующего этапа	В четвертом квадранте заказчики оценивают уже разработанную версию программного обеспечения. В конце концов, начинается планирование следующего этапа.

Плюсы спиральной модели:

- **Обработка рисков:** проекты с множеством неизвестных рисков, возникающих в процессе разработки, в этом случае спиральная модель является лучшей моделью разработки, которой следует следовать, благодаря анализу рисков и управлению рисками на каждом этапе.
- **Подходит для больших проектов:** рекомендуется использовать спиральную модель в больших и сложных проектах.
- **Гибкость в требованиях:** запросы на изменение в требованиях на более позднем этапе могут быть точно включены с помощью этой модели.
- **Удовлетворенность клиентов:** заказчик может наблюдать за развитием продукта на ранней стадии разработки программного обеспечения и, таким образом, привык к системе, используя ее до завершения работы над продуктом.

Минусы спиральной модели:

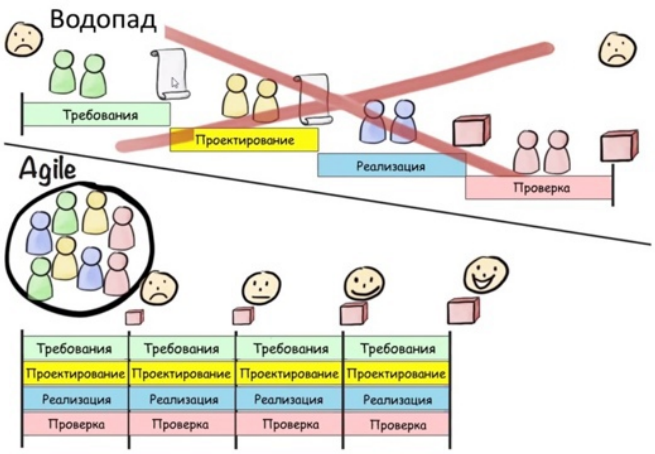
- **Сложность:** спиральная модель намного сложнее других моделей SDLC.
- **Дорого:** спиральная модель не подходит для небольших проектов, так как она дорогая.
- **Слишком сильно зависит от анализа рисков:** успешное завершение проекта во многом зависит от анализа рисков. Без очень большого опыта невозможно разработать проект с использованием этой модели.
- **Сложность в управлении временем:** поскольку количество этапов неизвестно в начале проекта, поэтому оценка времени очень сложна.

5 Общее описание гибких методов разработки ПО. Ценности, принципы.

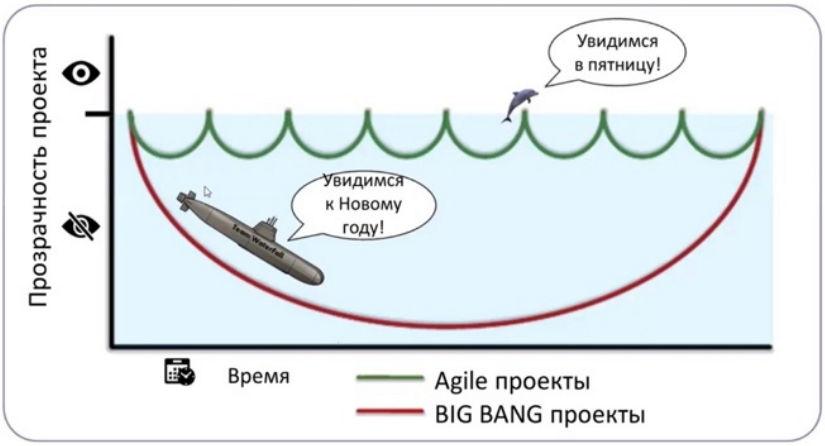
Agile — это гибкий подход к управлению проектами по разработке ПО, который часто применяют в небольших командах.

Как правило, для гибкого подхода Agile характерна работа с короткими итерациями по дветри недели. Внутри каждой итерации собрана серия задач: анализ, проектирование, непосредственно работа и тестирование. После каждой итерации команда анализирует результаты и меняет приоритеты для следующего цикла.

AGILE — ЭТО КРОССФУНКЦИОНАЛЬНО



AGILE — ЭТО ПРОЗРАЧНО



Ценности Agile:

1. **Люди и взаимодействие важнее процессов и инструментов**

   То, что общение и межличностные отношения важнее, чем строгие процессы — краеугольный камень Agile-управления проектами. Agile рекомендует персонализированный подход к управлению проектами, когда команды ориентируются на постоянное общение, а не на жестко распланированный выпуск обновлений.
2. **Работающий продукт важнее исчерпывающей документации**

   Agile-команды не очень любят бумажную работу. Для управления данными, отчетами и обновлениями статуса они предпочитают использовать гибкие программные решения, а не традиционную документацию.
3. **Сотрудничество с заказчиком важнее согласования условий контракта**

   Agile-команды любят сотрудничество — включая регулярные обновления и обратную связь о том, как продвигается проект, от клиентов и заинтересованных сторон. Чего Agile-команды не любят, так это долгих согласований объемных контрактов.
4. **Готовность к изменениям важнее следования первоначальному плану**

   Эта ценность прежде всего характеризует Agile-управление проектами. Agile-команды чутко реагируют на изменения и успешно адаптируются к новым условиям и вызовам.

Принципы Agile:

1. **Наивысшим приоритетом для нас является удовлетворение потребностей заказчика, благодаря регулярной и ранней поставке ценного программного обеспечения**
   Главное для Agile-команды — удовлетворенность клиентов, поэтому они обязательно представляют результаты своей работы через регулярные промежутки времени, а не заставляют заказчиков ждать финального результата в конце проекта.
2. **Изменение требований приветствуется даже на поздних стадиях разработки. Agile-процессы позволяют использовать изменения для обеспечения конкурентного преимущества заказчика**
   В этом их преимущество перед традиционными командами, которым обычно не так легко управлять изменениями.
3. **Работающий продукт следует выпускать как можно чаще, с периодичностью от двух недель до двух месяцев**
   Вспомним, что Agile-команды ценят постоянное общение, а не жестко распланированный выпуск обновлений, которые могут слишком далеко отстоять друг от друга по времени, что может оказаться неприемлемым для клиентов. Команды Scrum, которые тоже работают по методологии Agile, разбивают свою работу на периоды от одной до четырех недель, известные, как спринты.
4. **На протяжении всего проекта разработчики и представители бизнеса должны ежедневно работать вместе**
   Сотрудничество — краеугольный камень Agile, причем имеется в виду не только сотрудничество между членами команды, но и сотрудничество с заинтересованными сторонами, разработчиками, клиентами и другими партнерами.
5. **Над проектом должны работать мотивированные профессионалы. Чтобы работа была сделана, создайте им условия, обеспечьте поддержку — и полностью им доверьтесь**
   Agile-команды успешны, потому что в них работают только те люди, которые необходимы для проекта. Если участники Agile-команды получат поддержку, возможность работать вместе и инструменты, необходимые для работы, все остальное приложится.

6. **Непосредственное общение — наиболее практичный и эффективный способ обмена информацией как с самой командой, так и внутри команды**
   Все мы знаем, что главное в управлении проектами — личное сотрудничество. Этот принцип применим и во времена «новой нормы», при гибридных и удаленных моделях работы. Zoom и Teams — отличная альтернатива телефонным звонкам и электронной почте, а в ключевых точках проекта возможны и личные встречи команд.
7. **Работающий продукт — основной показатель прогресса**
   Смысл принципа, который называет работающий продукт основным показателем прогресса, в том, что главная цель команды всегда остается одна — предоставить клиенту как можно более высококачественный результат. Когда клиент доволен, это и есть главный показатель успеха проекта.
8. **Agile помогает наладить устойчивый процесс разработки. Инвесторы, разработчики и пользователи должны иметь возможность бесконечно поддерживать постоянный ритм Многие команды поначалу показывают бурный прогресс, который не получается сохранить до конца проекта.**
9. **Постоянное внимание к техническому совершенству и качеству проектирования повышает гибкость проекта**
   Agile не работает по принципу «раз — и готово». Каждый новый проект — это возможность для инноваций, а не для повтора одних и тех же идей.
10. **Простота как искусство сократить до минимума лишнюю работу крайне необходима**
    Команды Agile не занимаются переусложнением — они просто соблюдают проектные требования и хорошо выполняют свою работу, а затем переходят к следующему проекту.
11. **Самые лучшие требования, архитектурные и технические решения рождаются у самоорганизующихся команд**
    Лучшие команды — это те команды, у которых есть лидер, предоставляющий им свободу самовыражения. Микроменеджмент редко делает команды лучше или продуктивнее, и Agile-команды — отличный пример того, чего можно добиться без микроменеджмента.
12. **Команда должна систематически анализировать возможные способы улучшения эффективности и соответственно корректировать стиль своей работы**
    Непрерывное совершенствование — сама суть Agile, и регулярные проверки эффективности команды в целом могут помочь избавиться от вредных привычек и добиваться бóльшего.