Лекция 7. Вопросы управления жизненным циклом разработки сложных систем. Вопросы обеспечения качества ПО.

ПЛАН ЗАНЯТИЕ

- Процессы жизненного цикла информационных систем в соответствии с ГОСТ Р ИСО/МЭК 12207-99 и их обобщение процессной модель жизненного цикла стандарта ГОСТ Р ИСО/МЭК 15288. Взаимосвязь стандартов.
- Вопросы применения стандарта ISO/IEC/IEEE 15288.
- Понятие архитектуры информационных систем. Типы архитектур. Архитектурный подход к проектированию ИС. Стандарт ISO/IEC 42010. Архитектурные стили. Архитектурные методологии (фреймворки).
- Характеристики качества программного обеспечения. Основные стандарты качества ПО.

ПРОЦЕССЫ И МОДЕЛИ ЖИЗНЕННОГО ЦИКЛА ИНФОРМАЦИОННЫХ СИСТЕМ

В соответствии с **ГОСТ Р ИСО/МЭК 12207-99** процессы жизненного цикла включают себя работы, которые могут выполняться в жизненном цикле программных средств, распределены по пяти основным, восьми вспомогательным и четырем организационным процессам.

ГОСТ Р ИСО/МЭК 12207-99 охватывает жизненный цикл программных средств от концепции замыслов через определение и объединение процессов для заказа и поставки программных продуктов и услуг. Он также используется для контроля и модернизации данных процессов. Процессы, определенные в стандарте, образуют множество общего назначения. Конкретная организация, в зависимости от своих целей, может выбрать соответствующее подмножество процессов для выполнения своих конкретных задач. Поэтому, стандарт следует адаптировать для конкретной организации, проекта или приложения.

Стандарт предназначен для использования как в случае отдельно поставляемых программных средств, так и для программных средств, встраиваемых или интегрируемых в общую систему.

ОСНОВНЫЕ ПРОЦЕССЫ ЖИЗНЕННОГО ЦИКЛА

Основные процессы жизненного цикла состоят из пяти процессов, которые реализуются под управлением основных сторон, вовлеченных в жизненный цикл программных средств. Под основной стороной понимают одну из тех организаций, которые инициируют или выполняют разработку, эксплуатацию или сопровождение программных продуктов. Основными сторонами являются заказчик, поставщик, разработчик, оператор и персонал сопровождения программных продуктов.

ОСНОВНЫЕ ПРОЦЕССЫ ЖИЗНЕННОГО ЦИКЛА. ПРОДОЛЖЕНИЕ

Основными процессами жизненного цикла являются:

1. **Процесс заказа**. Определяет работы заказчика, то есть организации, которая приобретает систему, программный продукт или программную услугу.
2. **Процесс поставки**. Определяет работы поставщика, то есть организации, которая поставляет систему, программный продукт или программную услугу заказчику.
3. **Процесс разработки**. Определяет работы разработчика, то есть организации, которая проектирует и разрабатывает программный продукт.
4. **Процесс эксплуатации**. Определяет работы оператора, то есть организации, которая обеспечивает эксплуатационное обслуживание вычислительной системы в заданных условиях в интересах пользователей.
5. **Процесс сопровождения**. Определяет работы персонала сопровождения, то есть организации, которая предоставляет услуги по сопровождению программного продукта, состоящие в контролируемом изменении программного продукта с целью сохранения его исходного состояния и функциональных возможностей. Данный процесс охватывает перенос и снятие с эксплуатации программного продукта.

ВСПОМОГАТЕЛЬНЫЕ ПРОЦЕССЫ ЖИЗНЕННОГО ЦИКЛА

Вспомогательные процессы жизненного цикла состоят из восьми процессов. Вспомогательный процесс является целенаправленной составной частью другого процесса, обеспечивающей успешную реализацию и качество выполнения программного проекта. Вспомогательный процесс, при необходимости, инициируется и используется другим процессом.

Вспомогательными процессами являются:

1. **Процесс документирования**. Определяет работы по описанию информации, выдаваемой в процессе жизненного цикла.
2. **Процесс управления конфигурацией.** Определяет работы по управлению конфигурацией.
3. **Процесс обеспечения качества**. Определяет работы по объективному обеспечению того, чтобы программные продукты и процессы соответствовали требованиям, установленным для них, и реализовывались в рамках утвержденных планов. Совместные анализы, аудиторские проверки, верификация и аттестация могут использоваться в качестве методов обеспечения качества.

ВСПОМОГАТЕЛЬНЫЕ ПРОЦЕССЫ ЖИЗНЕННОГО ЦИКЛА. ПРОДОЛЖЕНИЕ

4. **Процесс верификации.** Определяет работы (заказчика, поставщика или независимой стороны) по верификации программных продуктов по мере реализации программного проекта.
5. **Процесс аттестации**. Определяет работы (заказчика, поставщика или независимой стороны) по аттестации программных продуктов программного проекта.
6. **Процесс совместного анализа.** Определяет работы по оценке состояния и результатов какой-либо работы. Данный процесс может использоваться двумя любыми сторонами, когда одна из сторон (проверяющая) проверяет другую сторону (проверяемую) на совместном совещании.
7. **Процесс аудита.** Определяет работы по определению соответствия требованиям, планам и договору. Данный процесс может использоваться двумя сторонами, когда одна из сторон (проверяющая) контролирует программные продукты или работы другой стороны (проверяемой).
8. **Процесс решения проблемы**. Определяет процесс анализа и устранения проблем (включая несоответствия), независимо от их характера и источника, которые были обнаружены во время осуществления разработки, эксплуатации, сопровождения или других процессов.

ОРГАНИЗАЦИОННЫЕ ПРОЦЕССЫ ЖИЗНЕННОГО ЦИКЛА

Организационные процессы жизненного цикла состоят из четырех процессов. Они применяются в какой-либо организации для создания и реализации основной структуры, охватывающей взаимосвязанные процессы жизненного цикла и соответствующий персонал, а также для постоянного совершенствования данной структуры и процессов. Эти процессы, как правило, являются типовыми, независимо от области реализации конкретных проектов и договоров; однако уроки, извлеченные из таких проектов и договоров, способствуют совершенствованию организационных вопросов.

Организационными процессами являются:

1. **Процесс управления**. Определяет основные работы по управлению, включая управление проектом, при реализации процессов жизненного цикла.
2. **Процесс создания инфраструктуры**. Определяет основные работы по созданию основной структуры процесса жизненного цикла.
3. **Процесс усовершенствования**. Определяет основные работы, которые организация (заказчика, поставщика, разработчика, оператора, персонала сопровождения или администратора другого процесса) выполняет при создании, оценке, контроле и усовершенствовании выбранных процессов жизненного цикла.
4. **Процесс обучения**. Определяет работы по соответствующему обучению персонала.

ПРОЦЕССЫ ЖИЗНЕННОГО ЦИКЛА СИСТЕМЫ. ГОСТ Р ИСО/МЭК 15288-2005

Процессы жизненного цикла определенны стандартом **ГОСТ Р ИСО/МЭК 15288-2005** "Процессы жизненного цикла систем" и могут применяться любой организацией при приобретении и использовании или создании и поставке системы. Они распространяются на любой уровень системной иерархии и на любую стадию жизненного цикла.

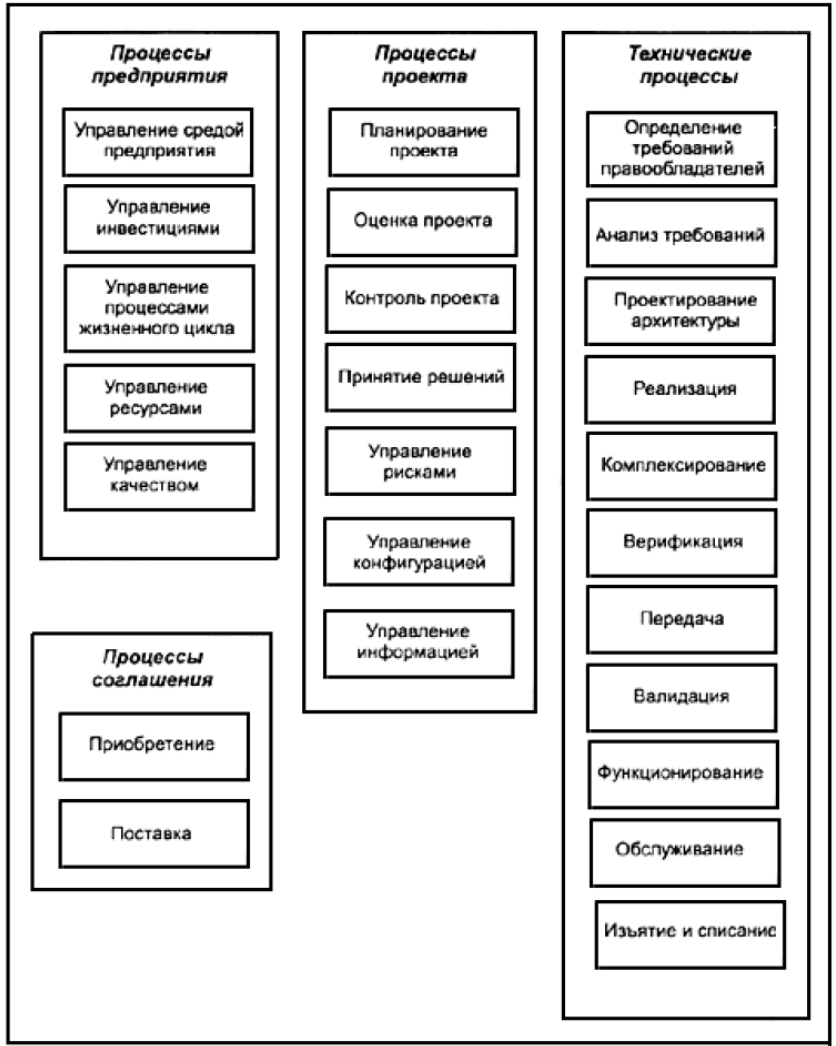
ГОСТ Р ИСО/МЭК 15288-2005 - объемный (почти 60 страниц) и сложный документ, расширяющий ГОСТ Р ИСО/МЭК 12207 на системы, "которые созданы человеком и состоят из одного или нескольких следующих элементов: технические средства, программные средства, люди, процессы (например, процесс оценки), процедуры (например, инструкции оператора), основные средства и природные ресурсы (например, вода, объекты живой природы, минералы)". Его отличают точность и продуманность формулировок, структурированность изложения. Структурно он следует ГОСТ Р ИСО/МЭК 12207 - в частности, здесь, как и там, процессы разбиваются на группы, хотя принцип разбиения совершенно другой.

Стандарт выделяет следующие группы процессов:

- процессы соглашения;
- процессы предприятия;
- процессы проекта;
- технические процессы.

ПРОЦЕССЫ ЖЦ

Процессы жизненного цикла основываются на принципах модульности (максимальная слаженность функций процесса и минимальная связь между процессами) и собственности (процесс связывается с ответственностью). Функции, которые осуществляются данными процессами, определяются в зависимости от конкретных целей, результатов и набора действий, составляющих данный процесс. Процессы, описанные в стандарте, не препятствуют и не исключают использование дополнительных процессов, которые организация посчитает необходимыми.



ПРОЦЕССЫ ЖЦ. ПРОДОЛЖЕНИЕ

Обычно организации различают разные области управленческой ответственности и действий. Взятые вместе, эти области вносят вклад в способность организации продвигать свою продукцию на рынок. В стандарте используется модель процессов, основанная на **трех основных областях (или уровнях) ответственности**: ответственность предприятия, ответственность проекта и техническая ответственность. В рамках каждой организации скоординированное множество процессов предприятия, проектных и технических процессов способствует эффективному созданию и использованию систем, содействуя, таким образом, достижению целей организации. Различные организации и различные области ответственности внутри организации устанавливают между собой рабочие взаимоотношения и подтверждают свою ответственность путем заключения соглашений. Эти соглашения унифицируют и координируют вклады, сделанные различными областями ответственности с целью достижения общих бизнес-целей.

ВЗАИМОСВЯЗЬ ГОСТ Р ИСО/МЭК 15288 И ИСО/МЭК 12207

На рисунке стандарт ИСО/МЭК 12207 представлен своей последней версией (Изменение № 1 к ИСО/МЭК 12207). С точки зрения структуры ГОСТ Р ИСО/МЭК 15288 различия между ГОСТ ИСО/МЭК 12207 и Изменением № 1 к ИСО/МЭК 12207 несущественны. ИСО/МЭК 12207 детализирует процесс реализации элементов системы, которые могут включать и программные средства. В Приложении С к стандарту точно определяется, процессам какого из двух стандартов (т. е. ГОСТ Р ИСО/МЭК 15288 и ГОСТ Р ИСО/МЭК 12207) следует отдавать предпочтение при построении программной системы.

Уровень абстракции описания процессов ГОСТ Р ИСО/МЭК 15288 выше, чем в ГОСТ Р ИСО/МЭК 12207.

КОНСТРУКЦИИ ПРОЦЕССОВ В ИСО/МЭК 12207 и ИСО/МЭК 15288

Процессы требуют цели и результатов. Все процессы имеют, как минимум, один вид деятельности. Процессы с их сформулированными целями и выходами составляют эталонную модель процессов (ЭМП). ЭМП представлена в приложении В

Виды деятельности являются структурным компонентом для группирования связанных задач. Виды деятельности предоставляют средства для рассмотрения связанных задач в пределах процесса с целью улучшения понимания и взаимосвязей процессов. Если деятельность достаточно согласована, то она может быть преобразована в процесс более низкого уровня посредством определения цели и совокупности выходов

Задача является детализированным условием реализации процесса. Она может служить требованием (должно»), рекомендацией («следует») или разрешением («может»)

Примечания используются, если появляется необходимость в поясняющей информации для лучшего описания содержания или структуры процесса. Примечания обеспечивают понимание, относящееся к реализации или области применяемости, такой как списки, примеры и другие представления

ВЗАИМОСВЯЗЬ МЕЖДУ ИСО/МЭК 15288 И ИСО/МЭК 12207 (ИЗМ. №1)

ИСО/МЭК 15288	Степень отображения специфики процесса			Изменение № 1 к ИСО/МЭК 12207
	←	—	→	
1	2	3	4	5
Процесс управления средой предприятия		✓		Процесс управления, процесс усовершенствования
Процесс управления инвестициями	✓			Процесс инфраструктуры
Процесс управления процессами жизненного цикла системы	✓			Процесс поставки
Процесс управления процессами жизненного цикла системы, процесс управления средой предприятия			✓	Процесс управления, процесс усовершенствования
Процесс приобретения		✓		Процесс приобретения
Процесс определения требований правообладателей	✓			Процесс разработки, процесс применимости
Процесс поставки		✓		Процесс поставки
Процесс управления рисками	✓			Процесс приобретения, процесс поставки, процесс управления
Процесс управления информацией		✓		Процесс документирования, процесс управления активами
Процесс анализа требований		✓		Процесс разработки
Процесс проектирования архитектуры		✓		Процесс разработки, процесс применимости

ВЗАИМОСВЯЗЬ МЕЖДУ ИСО/МЭК 15288 И ИСО/МЭК 12207 (ИЗМ. №1). ПРОДОЛЖЕНИЕ

ИСО/МЭК 15288	Степень отображения специфики процесса			Изменение № 1 к ИСО/МЭК 12207
	←	—	→	
1	2	3	4	5
Процесс реализации элементов системы		✓		Процесс разработки
Процесс комплексирования		✓		Процесс разработки
Процесс передачи	✓			Процесс разработки
Процесс передачи		✓		Процесс обучения
Процесс функционирования		✓		Процесс функционирования
Процесс обслуживания		✓		Процесс сопровождения
Процесс изъятия и списания	✓			Процесс сопровождения
Процесс управления конфигурацией		✓		Процесс управления конфигурацией
Процесс оценки проекта		✓		Процесс обеспечения гарантии качества
Процесс управления качеством		✓		Процесс управления
Процесс верификации		✓		Процесс верификации
Процесс валидации		✓		Процесс валидации, процесс применимости
Процесс оценки проекта			✓	Процесс совместного анализа
Процесс управления средой предприятия			✓	Процесс аудита
Процесс оценки проекта			✓	Процесс аудита
Процесс принятия решения		✓		Процесс решения проблем, процесс разработки, процесс управления повторным использованием программ
Процесс оценки проекта			✓	Процесс оценки продукта
Процесс планирования проекта		✓		Процесс управления, процесс поставки, процесс разработки
Процесс оценки проекта		✓		Процесс управления
Процесс контроля проекта		✓		Процесс управления, процесс решения проблем
Процесс управления ресурсами		✓		Процесс инфраструктуры
Процесс управления ресурсами		✓		Процесс управления человеческими ресурсами
Изготовление			✓	Область инженерии
Процесс адаптации		✓		Процесс адаптации

ПРОЦЕССЫ ЖЦ. ЦЕЛИ И РЕЗУЛЬТАТЫ ПРОЦЕССОВ

В ГОСТ Р ИСО/МЭК 15288 для каждого процесса определены цели и результаты, а также приведено описание соответствующей деятельности.

"Организация осуществляет процессы жизненного цикла избирательно, чтобы достичь целей и результатов стадий жизненного цикла".

"Каждый процесс жизненного цикла при необходимости может быть начат в любой момент жизненного цикла, при этом нет определенного порядка в их использовании. Любой процесс может выполняться одновременно с любыми другими процессами жизненного цикла и может быть реализован на любом уровне иерархии структуры системы. Таким образом … порядок, в котором представлены используемые процессы и группы процессов, не подразумевает предшествования или последовательности их применения в течение жизненного цикла системы. Однако группы процессов отражают концепции, лежащие в основе настоящего стандарта".

Жизненный цикл системы не подразумевает жесткой последовательности действий. Порядок действий зависит от ситуации и может меняться, хотя, конечно, "любой процесс" не может выполняться "одновременно с любыми другими процессами".

УПРАВЛЕНИЕ ПРОЦЕССАМИ ЖИЗНЕННОГО ЦИКЛА

Согласно **ГОСТ Р ИСО/МЭК 15288** , **цель управления процессами жизненного цикла системы** заключается в гарантировании доступности эффективных процессов жизненного цикла для использования организацией. Данный процесс обеспечивает процессы жизненного цикла системы, которые согласованы с целями и политикой организации, определены, адаптированы и поддержаны соответствующим образом для учета особенностей отдельных проектов и способны реализовываться с помощью эффективных проверенных методов и инструментальных средств.

В результате эффективного управления процессами жизненного цикла системы:

a) определяются процессы жизненного цикла системы, которые будут использоваться организацией;

b) определяется политика применения процессов жизненного цикла системы;

c) определяется политика адаптации процессов жизненного цикла системы для удовлетворения потребностей отдельных проектов;

d) определяются критерии оценки результатов применения процессов жизненного цикла системы;

e) предпринимаются действия по совершенствованию способов определения и применения процессов жизненного цикла системы.

ДЕЯТЕЛЬНОСТЬ В ПРОЦЕССЕ УПРАВЛЕНИЯ ПРОЦЕССАМИ ЖИЗНЕННОГО ЦИКЛА СИСТЕМЫ

При **реализации процессов управления процессами жизненного цикла системы организация должна осуществлять следующие действия** в соответствии с принятой политикой и процедурами:

a) устанавливать стандартные наборы процессов жизненного цикла систем для соответствующих стадий жизненного цикла системы;

b) определять приемлемые политику и процедуры адаптации и требования к их утверждению;

c) определять методы и инструментальные средства, которые поддерживают выполнение процессов жизненного цикла системы;

d) по возможности устанавливать показатели, которые позволяют определять характеристики выполненных стандартных процессов;

e) контролировать выполнение процесса, сохранять и анализировать показатели процесса и определять тенденции по отношению к критериям предприятия;

f) определять возможности для усовершенствования стандартных процессов жизненного цикла систем;

g) совершенствовать имеющиеся процессы, методы и инструментальные средства, используя найденные возможности.

ПУТИ ДОСТИЖЕНИЯ РЕЗУЛЬТАТОВ ДЕЯТЕЛЬНОСТИ В ПРОЦЕССЕ УПРАВЛЕНИЯ ЖЦ

В **результате реализации процесса управления процессами жизненного цикла** должны быть определены "процессы жизненного цикла системы, которые будут использоваться организацией". В ГОСТ Р ИСО/МЭК 12207 такой или аналогичный процесс отсутствовал - технология внедрения стандарта оставалась за кадром.

Содержательно процесс подразумевает, что для (каждой) разрабатываемой или внедряемой системы X необходимо разработать, согласовать и утвердить у руководства документ "Процессы жизненного цикла системы X", причем нужно принять решение о том, будут ли эти процессы такими же, как и для систем Y, Z…, или для них придется разрабатывать свои документы.

Должны быть определены "политика применения процессов жизненного цикла системы" и "политика адаптации процессов жизненного цикла системы для удовлетворения потребностей отдельных проектов". То есть нужно создать документ (или совокупность документов для отдельных проектов), описывающий, какие процессы, как и почему (а возможно, и кем) должны быть адаптированы к требованиям организации и/или проекта, а также определяющий механизм(ы) модификации процессов при необходимости. В стандарте есть Приложение А, посвященное процессу такой адаптации.

Необходимо утвердить "критерии оценки результатов применения процессов жизненного цикла системы", т.е. придумать набор параметров, по которым можно будет судить о том, как работает каждый внедренный процесс жизненного цикла и что можно и нужно сделать для повышения его эффективности.

Также необходимо разработать и начать выполнять "действия по совершенствованию способов определения и применения процессов жизненного цикла системы".

ПУТИ ДОСТИЖЕНИЯ РЕЗУЛЬТАТОВ ДЕЯТЕЛЬНОСТИ В ПРОЦЕССЕ УПРАВЛЕНИЯ ЖЦ. ПРОДОЛЖЕНИЕ

Для того чтобы выполнить требования стандарта целесообразно создавать специальную структуру, или подразделение, или временную рабочую группу, распределять ответственности, вводить формы отчетности и т. д. и т. п. Возможно, потребуются дополнительные средства автоматизации (например, корпоративный портал или база знаний). Поскольку речь идет о долгосрочной деятельности, необходимо запланировать соответствующие инвестиции. Значит, нужно понять, кто в организации заинтересован в результате, выявить стейкхолдеров этого процесса, определить его владельца, согласовать требования к результатам.

Стандарт не уточняет ни ролей, ни форматов документов, ни детального состава работ, оставляя все это на усмотрение конкретных организаций. Таким образом, успешность или не успешность внедрения рассмотренного процесса в большой степени определяется готовностью организации взяться за такую сложную работу, возможностью ее контролировать и желанием довести до конца.

КОНЦЕПЦИЯ СТАДИЙ ЖИЗНЕННОГО ЦИКЛА

Приложение В ГОСТ Р ИСО/МЭК 15288-2005 предлагает концепцию стадий жизненного цикла, которая развивает идеи, содержащиеся в ГОСТ Р ИСО/МЭК 15271-2002.

Предлагаются следующие стадии жизненного цикла систем:

a) стадия замысла;

b) стадия разработки;

c) стадия производства;

d) стадия применения;

e) стадия поддержки применения;

f) стадия прекращения применения и списания.

Стадии "образуют структуру работ для детализированного моделирования жизненных циклов системы". Это означает, что стадии являются теми кирпичиками, из которых в ходе процесса адаптации (Приложение А данного стандарта) строится жизненный цикл конкретной системы. Стадии могут перекрываться или повторяться; каждая стадия содержит процессы, отобранные и настроенные (также во время адаптации) так, чтобы реализовать цели этой стадии.

Для каждой стадии определяются цель и результаты.

ПРИМЕР. ОПРЕДЕЛЕНИЕ ЦЕЛЕЙ НА СТАДИИ ЗАМЫСЛА

Стадия замысла выполняется для оценки новых возможностей в деловой сфере, разработки предварительных системных требований и осуществимых проектных решений.
Эта стадия часто является наиболее важной.
Результаты выполнения стадии замысла по ГОСТ Р ИСО/МЭК 15288:
a) установление новых замыслов, в которых предлагаются новые возможности, увеличение производительности или снижение общей стоимости собственности правообладателей в течение жизненного цикла системы;
b) оценка осуществимости замысла и решений для рассматриваемой системы в течение жизненного цикла, включая обеспечивающие системы, с учетом как технических, так и деловых целей правообладателя;
c) подготовка и формирование базовой линии требований правообладателя и предварительных системных требований (технических спецификаций для выбранной рассматриваемой системы и пригодности спецификаций для предусмотренного способа взаимодействия между человеком и системой);
d) уточнение результатов стадий в модели жизненного цикла системы;
e) планы идентификации, оценки и уменьшения рисков для стадий модели жизненного цикла системы;

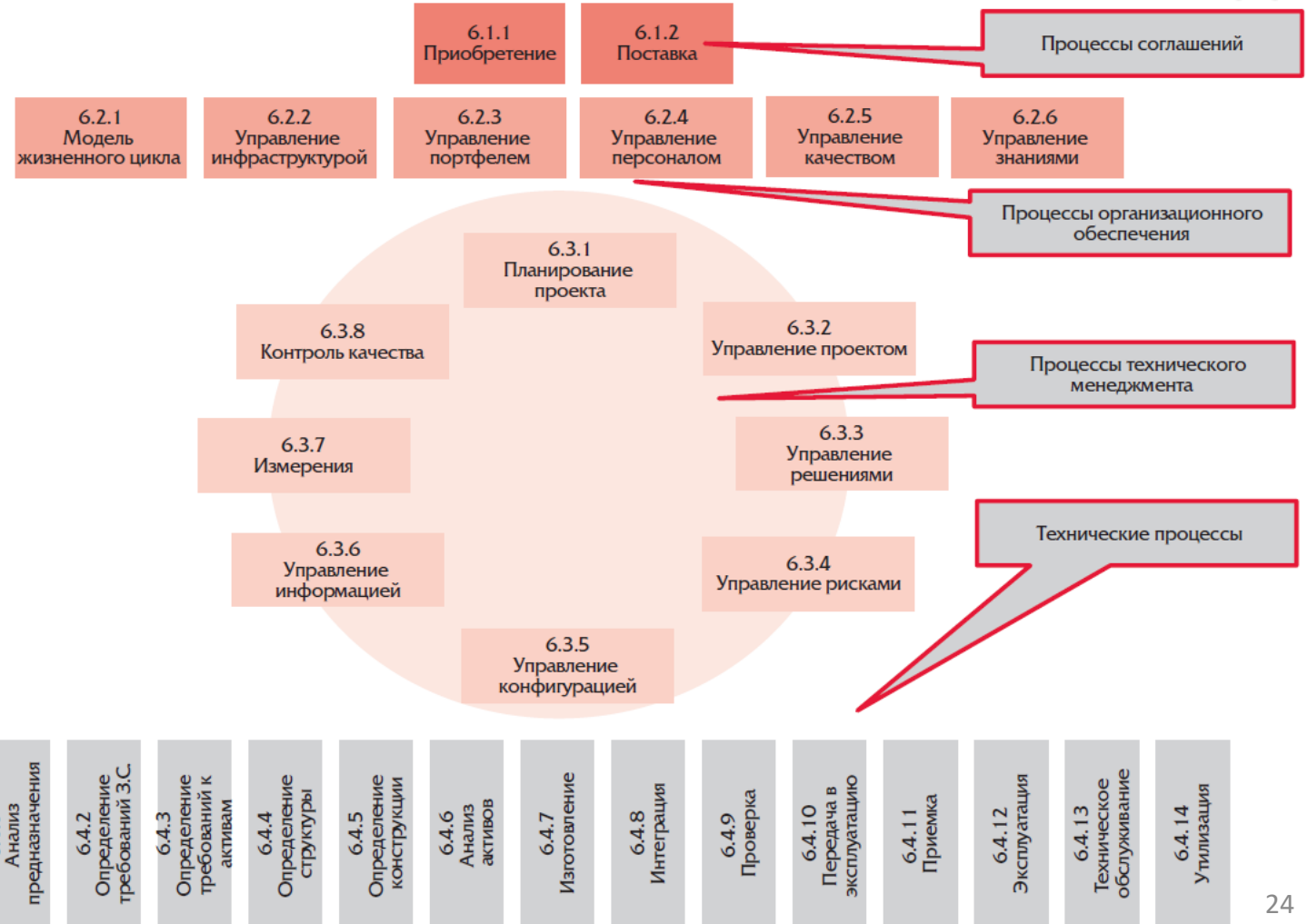
ПРИМЕР. ОПРЕДЕЛЕНИЕ ЦЕЛЕЙ НА СТАДИИ ЗАМЫСЛА. ПРОДОЛЖЕНИЕ

f) идентификация и предварительная спецификация услуг, которые необходимо получать от обеспечивающих систем в течение жизненного цикла рассматриваемой системы;
g) замыслы выполнения всех последующих стадий;
h) планы и критерии завершения стадии разработки;
i) планы идентификации, оценки и уменьшения рисков для данной и последующих стадий модели жизненного цикла системы;
j) удовлетворение критерием завершения данной стадии;
k) санкционирование перехода на стадию разработки.

Приведенный перечень вполне может послужить основой **для корпоративного стандарта** на подобные документы. Стандарт не содержит никаких указаний на то, какие именно процессы включаются в стадию замысла (как и в другие стадии). Можно предположить, что в ходе этой стадии будут использоваться такие процессы, как процесс анализа требований или процессы планирования и управления ресурсами (если стадия замысла реализуется как проект), но это только одна из возможностей. Общая задача моделирования жизненного цикла выходит за рамки стандарта.

СОВОКУПНОСТЬ ПРОЦЕССОВ ЖЦ ПО СТАНДАРТУ ISO/IEC/IEEE 15288:2015(E)

В новой версии стандарта сформулирован новый перечень процессов, которые должны выполняться на всех стадиях жизни системы (подрядные отношения, управления проектами и портфелями, риск-менеджмент, управление человеческими ресурсами). Среди перечня технических практик появились практики анализа бизнеса, определения потребностей и требований стейкхолдеров, управления знаниями и другие.



СТАНДАРТЫ ПРОЦЕССОВ ЖЦ. КРАТКИЕ ВЫВОДЫ

- Принцип разделения процессов на группы для ГОСТ Р ИСО/МЭК 15288-2002 - абсолютной иной, нежели в ГОСТ Р ИСО/МЭК 12207. На первый план выдвигаются бизнес-цели, на достижение которых работают процессы.
- Идея привязки процессов к ответственностям представляется плодотворной. Это помогает привязать процессы к структуре организации, определить в ней владельцев процессов. Таким образом, становится более структурированной задача внедрения стандарта, возникают связи между бизнес-целями организации и результатами деятельности владельцев процессов.
- ГОСТ Р ИСО/МЭК 15288-2005 открывает ряд новых возможностей. Прежде всего, он позволяет связать задачу анализа процессов с бизнесом организации, переводя ее из разряда технических или технологических в разряд бизнес-задач. Например, посредством процессов соглашения и управления инвестициями реализуется связь процессов жизненного цикла с задачами управления корпоративными финансами, финансового планирования и бюджетирования, процесс управления ресурсами взаимодействует с управлением персоналом и т. д. Роль стандарта с этой точки зрения прежде всего в том, что он определяет место процессов управления жизненным циклом систем среди остальных процессов организации.
- Новый стандарт ISO/IEC/IEEE 15288:2015 расширяет перечень мероприятий и задач, необходимых для реализации процессов, однако не содержит подробного описания порядка их выполнения. Эта информация присутствует в других технических и финансовых стандартах, которые необходимо использовать совместно с этим стандартом.
- Предстоит еще огромная работа не только по расширению области охвата моделей, но и по осознанию и решению новых задач, возникших в связи с внедрением в практику управления процессного подхода. Например, необходимо научиться оценивать затраты на внедрение эталонных или адаптированных процессов управления ИТ в практику организации, а также связанные с этим выгоды и риски.

ПРИМЕР. РАЗРАБОТКА И РАЗВЕРТЫВАНИЕ ОНТОЛОГИИ С ИСПОЛЬЗОВАНИЕМ СТАНДАРТА ISO/IEC 15288

Из работы *Гарольд «Бад» Лоусон. Путешествие по системному ландшафту. - М.:ДМК Пресс. – 2013. – 323 с.*

Предпосылки разработки методологии.

- Хотя существует несколько методологий разработки онтологий, построение онтологии остается скорее искусством, чем понятным инженерным процессом, практический опыт применения онтологий явно недостаточен.
- Известные методологии разработки онтологий в значительной степени заимствуют терминологию программной инженерии и не учитывают требования стандартов управления ЖЦ ИС.
- Стандарт ISO/IEC 15288 предоставляет не связанный с конкретной предметной областью подход, удобный для понимания природы систем, создаваемых человеком и принципов их построения, а также учитывает особенности продвижения систем по жизненному циклу. Стандарт требует, чтобы модель жизненного цикла была разработана для каждой целевой системы, к которой этот стандарт будет применяться.
- Наиболее распространенных определений онтологии, данное Грубером: «спецификация концептуального представления в явном виде». Грубер позже уточнил: «Оглядываясь назад, я бы не стал менять это определение, а постарался бы подчеркнуть, что мы разрабатываем онтологии. Последствия такого представления заключаются в том, что мы можем применять инженерные методы для их разработки и оценки».
- Использование стандарта ISO/IEC 15288 может привнести ценный инженерный опыт в разработку онтологий.

ОНТОЛОГИИ КАК СИСТЕМЫ

Онтологии могут рассматриваться как системы, поскольку онтологии представляют собой соглашения, (которые можно рассматривать как разновидность описания системы) о том, что мы хотели бы описать. Онтология, по существу, является всего лишь описанием: это описание того, как считает группа людей или организация, необходимо описать, обсудить и совместно использовать информацию о какой-то сущности. Системы служат для удовлетворения потребностей. Потребности, которые можно удовлетворить с помощью онтологии, могут быть следующими:

- разделение общего представления о структуре информации между людьми и программными агентами;
- обеспечение возможности повторного использования знаний в конкретной предметной области;
- возможность выражения допущений в явном виде в конкретной предметной области;
- отделение предметного знания от операционного знания и возможность анализа знания в конкретной предметной области.

Элементом системы является либо онтология (состоящая из классов, свойств и ограничений), которая не поддается дальнейшей декомпозиции, либо другая онтология, которую мы используем повторно. Помимо того, что онтология может являться системным элементом другой онтологии, она также может быть системным элементом приложения, основанного на онтологии. В этом случае возможно применить стандарт ISO/IEC 15288 к уровню разработки программного обеспечения на основе онтологии.
Применительно к онтологии заинтересованными сторонами являются и люди и машины.
Рассматривая онтологии как системы, мы можем использовать методики системного мышления и системной инженерии для упорядочивания разработки и развертывания онтологий и для оказания помощи в управлении изменениями.

МЕТОДОЛОГИИ ДЛЯ РАЗРАБОТКИ ОНТОЛОГИЙ

Общие принципы.

В процессе построения онтология проходит стадии: спецификации, составления концепции, формализации, реализации и сопровождения.

На протяжении всего жизненного цикла должны выполняться следующие действия: приобретение знаний, оценка и документирование.

Отличие от разработки программного обеспечения заключается в том, для ПО приобретение знаний используется редко, в то время как применительно к процессу построения онтологии это действие является центральным. Другим отличием является разделение стадий составления концепции и формализации.

Метод Ашольда и Кинга.

1. Определение назначения и области применения.
2. Выявление ключевые концепций и взаимосвязей для выбранной предметной области, фиксация их однозначным образом в текстовой форме с использованием конкретной терминологии.
3. Кодирование на формальном языке представления знаний. Повторное использование подходящих знаний из существующих онтологий.
4. Оценка онтологии и оформление документации для ее последующего повторного использования и модификации.

МЕТОДОЛОГИИ ДЛЯ РАЗРАБОТКИ ОНТОЛОГИЙ. ПРОДОЛЖЕНИЕ

Метод Грюнингера и Фокса (методология, основанная на использовании логики первого порядка).

1. Определение мотивационных сценариев.
2. Подготовка неформальных вопросов компетенции для определения области применения онтологии.
3. Выделение основных концепций и их свойств, отношений и аксиом, основываясь на ответах на вопросы.
4. Подготовка полуформальных описаний концептов и отношений.
5. Формальные описание онтологии.
6. Оценка онтологии на основе вопросов компетенции.

Метод METHONTOLOGY.

Может быть использован для построения онтологий с нуля, повторного использования других онтологий в том виде, в котором они имеются, или для их модернизации. Включает процесс разработки онтологии, жизненный цикл, основанный на развивающихся прототипах, и конкретные методики для выполнения каждого действия. Процесс разработки состоит из составления графика работ, контроля, обеспечения качества, спецификации, приобретения знаний, составления концепции, комплексирования, формализации, реализации, оценки, сопровождения, документирования и управления конфигурацией. Жизненный цикл определяет стадии, через которые проходит онтология и где имеются взаимозависимости с жизненными циклами других онтологий.

ОСОБЕННОСТИ ЖЦ ОНТОЛОГИИ. ПОВТОРНОЕ ИСПОЛЬЗОВАНИЕ

Повторное использование онтологий представляет собой процесс использования онтологий, имеющихся в распоряжении, для порождения новых онтологий.

Выделяют два процесса повторного использования: слияние (fusion) и компоновка (composition).

При использовании слияния онтология создается посредством объединения двух или более различных онтологий, касающихся одного и того же предмета.

При использовании компоновки создается онтология, касающаяся одного предмета, и повторно использующая одну или несколько онтологий, касающихся различных предметов.

Исходные онтологии группируются, сочетаются, собираются вместе для образования итоговой онтологии.

При разработке метода управления ЖЦ онтологии как системы необходимо предусматривать возможности повторного ее использования, включая слияние и компоновку.

ОСОБЕННОСТИ ЖЦ ОНТОЛОГИИ. ОЦЕНКА ОНТОЛОГИИ

Оценка онтологии является важной частью процесса разработки онтологии.

Известные методы оценки:

- рассматривается как некий «золотой стандарт» и используется в приложении, результаты которого оцениваются;
- рассматривается как источник данных для предметной области;
- оценивается людьми, в соответствии с предварительно установленными критериями.

При разработке онтологий оценка может быть разделена на **техническую и пользовательскую оценку**.

При технической оценке онтология оценивается по отношению к некоторой системе взглядов. Используются два действия: верификация (правильность в соответствии с принятым представлением о предметной области) и валидация (соответствие тому, что, как предполагается, должно быть в соответствии с документом, содержащим технические условия).

При пользовательской оценке применимость и полезность онтологии и ее документации при (повторном) использовании или совместном использовании в приложениях оцениваются с точки зрения пользователя.

ОСОБЕННОСТИ ЖЦ ОНТОЛОГИИ. СОПРОВОЖДЕНИЕ ОНТОЛОГИЙ

Модели знаний почти неизбежно меняются в процессе построения и использования систем, основанных на знаниях.

Необходимость в формальном описании модели может заставить эксперта пересмотреть ее, особенно при наличии обратной связи, полученной в результате применения модели в реальном или имитированном мире.

Выявление всех требований на ранних этапах проектирования онтологии является проблематичным, так как потенциальным пользователям трудно оценить преимущества или возможные применения новой системы, а сама система после ее инсталляции изменяет рабочие процессы. Предположения относительно того, на чем основана модель, могут быть неверными, причиной этого отчасти являются трудности, с которыми эксперты в предметной области, сталкиваются при описании своей повседневной деятельности.

ОСОБЕННОСТИ ИСПОЛЬЗОВАНИЯ СТАНДАРТА ISO/IEC 15288 ДЛЯ РАЗРАБОТКИ И РАЗВЕРТЫВАНИЯ ОНТОЛОГИЙ

Стандарт требует, чтобы цели и выходы (результаты) были определены для каждой стадии жизненного цикла. Процессы и действия жизненного цикла выбираются, адаптируются и применяются на данной стадии для достижения ее цели и результатов. Для каждого процесса стандарт описывает цель, результаты успешной реализации и связанные с этим действия.
Различные процессы могут использоваться в течение жизненного цикла разработки и применение онтологии. Для каждой стадии жизненного цикла системы выбираются подходящие процессы из стандарта.
Выбор процессов-компонентов данного жизненного цикла зависит от того, какой вид онтологии будет разрабатываться. Онтологии, которые, как предполагается, будут крупномасштабными онтологиями, использующимися многими пользователями, сосредоточатся на многих аспектах и будут характеризоваться более высокими требованиями к строгости, чем онтологии, разрабатываемые в интересах небольших групп пользователей.
Способ построения онтологий, ручной или полуавтоматический, также влияет на то, какие действия должны быть выполнены в различных процессах.

СТАДИИ И ПРОЦЕССЫ РАЗРАБОТКИ ОНТОЛОГИЙ

1. Стадия концепции.

Процесс определения требований заинтересованных сторон. Выявление заинтересованных сторон, в частности специалистов в предметной области, разработчиков, пользователей и лиц, осуществляющих сопровождение онтологий. Установление мотивационных сценариев и их использование для определения назначения и области применения онтологии и для перечисления важных терминов.

2. Стадия обоснования.

Процесс определения требований заинтересованных сторон. Уточнение мотивационных сценариев, назначения и области применения, а также важных терминов.

Процесс приобретения. Идентификация онтологий, пригодных для повторного использования, с перечнем важных терминов в качестве отправной точки; оценка выявленных онтологий в соответствии с критериями, установленными в процессе определения требований заинтересованных сторон. Принятие решения о том, удовлетворяет ли какая-либо из внешних онтологий конкретным потребностям и в какой мере нужны внутренние адаптации. Рассмотрение в случае существования источников для полуавтоматического построения онтологии, какие их них могли бы быть использованы для начальной загрузки онтологии.

СТАДИИ И ПРОЦЕССЫ РАЗРАБОТКИ ОНТОЛОГИЙ. ПРОДОЛЖЕНИЕ

3. Стадия разработки. На этой стадии принимается решение о том, какие классы, свойства, ограничения и экземпляры должны быть включены в нашу онтологию. Это делается неформально. Для этого используются следующие процессы.

Процесс приобретения. Для онтологий, отобранных для повторного использования, выполняется более детальный анализ того, какие концепты будут использоваться повторно и какие концепты должны быть представлены во внутренней онтологии.

Процесс разработки архитектуры. Для определения архитектуры онтологии должно быть рассмотрено как ручное, так и полуавтоматическое обучение онтологии (Ontology Learning). Определяются классы и иерархия классов. Возможны следующие способы разработки иерархии классов: нисходящий, восходящий или сочетание и того, и другого. Определяются свойства классов, которые описывают внутреннюю структуру концептов. Определяются ограничения, которые описывают или ограничивают совокупность возможных значений свойств. Ограничения могут касаться количества элементов множества, типа значений, а также предметной области и диапазона свойств. Идентифицируются экземпляры классов. Определение конкретного экземпляра класса требует выбора класса, создания конкретного экземпляра этого класса и установления свойств. Возможно использование шаблонов проектирования онтологий.

Процесс реализации. Принимается решение о том, как осуществлять кодирование классов, свойств и ограничений, идентифицированных в процессе разработки архитектуры.

Процесс верификации. Собирает сценарии проверки рассуждений которые следует использовать для обнаружения ошибок при кодировании онтологии.

Процесс валидации. Проверяет определенные компоненты онтологии с заинтересованными сторонами, прежде всего со специалистами в предметной области.

СТАДИИ И ПРОЦЕССЫ РАЗРАБОТКИ ОНТОЛОГИЙ. ПРОДОЛЖЕНИЕ

4. Стадия производства. На данной стадии выполняется кодирование классов, свойств, ограничений и экземпляров, идентифицированных на стадии разработки с использованием языка представлений онтологий. По сравнению с результатами стадии разработки результаты стадии производства лучше формализованы, однако степень детализации и формализации здесь по-прежнему сильно зависит от назначения онтологии.

На этой стадии используются следующие процессы.

Процесс реализации. Осуществляет кодирование классов, свойств и ограничений, идентифицированных на стадии разработки в репрезентативной форме в соответствии с принятым решением. В этом процессе должен быть использован отождествленный передовой опыт. Для OWL, например, документы Рабочей группы по передовому опыту и использованию семантической паутины – W3C Semantic Web Best Practices and Deployment Working Group (*http://www.w3.org/2001/sw/BestPractices/*).

Процесс верификации. Проверяет, что кодирование выполнено в соответствии со спецификацией требований. Кроме того, сценарии проверки рассуждений, определенные на стадии разработки, могут быть использованы для выявления проблем и противоречий.

Процесс валидации. Проверяет, может ли быть использована построенная онтология при выполнении мотивационных сценариев. Осуществляет проверку правильности в соответствии с принятым представлением о предметной области.

Процесс сопровождения. Устанавливает критерии для последующих изменений онтологии и адаптирует политику управления версиями онтологии.

СТАДИИ И ПРОЦЕССЫ РАЗРАБОТКИ ОНТОЛОГИЙ. ПРОДОЛЖЕНИЕ

5. Стадия эксплуатации.

Процесс функционирования. В процессе функционирования онтология используется как часть приложения.

Процесс поставки. Делает онтологию доступной и известной другим лицам, например, добавляет ее в онтологический репозиторий.

6. Стадия поддержки.

Процесс сопровождения. В соответствии с политикой управления версиями вносит коррективы в онтологию при обнаружении возможных ошибок или с учетом изменений в знаниях, связанных с конкретной предметной областью.

7. Стадия изъятия из обращения.

Процесс прекращения использования и изъятия. Когда онтология больше не применима, а изменения, которые нужно внести, очень велики, или когда другая онтология удовлетворяет тем же потребностям лучше, то можно изъять онтологию из обращения. Это требует создания стратегии изъятия из обращения для решения ряда вопросов, в том числе, как обеспечить преемственность при переходе от используемой онтологии к новой онтологии.

Многие из рассмотренных стадий имеют итеративный характер. Например, работа по неформальному определению классов может вызвать изменения в определении требований или при формальном определении классов мы можем осознать, что нужно внести изменения в неформальные определения. Требования заинтересованных сторон распределяются между разными стадиями.

ПРИМЕР РАЗРАБОТКИ И РАЗВЕРТЫВАНИЕ ОНТОЛОГИИ ДЛЯ СТОМАТОЛОГИИ. SWEDISH ORAL MEDICINE WEB (SOMWEB)

В проекте Swedish Oral Medicine Web (SOMWeb) разрабатывалась онтология для представления результатов стоматологических обследований. Были выделены следующие стадии и процессы.

1. Стадия концепции.

Процесс определения требований заинтересованных сторон. Известными заинтересованными сторонами были практикующие врачи и ИТ-персонал в стоматологической клинике (специалисты в предметной области, пользователи и лица, осуществляющие сопровождение), а также представители информационных департаментов (разработчики онтологий и лица, осуществляющие их сопровождение). Мотивирующим сценарием было использование онтологии в качестве схемы для представления результатов обследований в стоматологии с использованием технологии RDF для сетевого сообщества SOMWeb. Назначение онтологии заключается в представлении концептов, имеющих отношение к представлению результатов стоматологических обследований, а область применения онтологии – получение возможности представлять хотя бы то, что может быть представлено при помощи существовавшего ранее инструмента для представления знаний MedView. Использовались термины, уже перечисленные в предыдущем представлении, различные составляющие части стоматологических обследований и связанные с ними свойства, а также списки значений для свойств.

2. Стадия обоснования.

Процесс приобретения. Выполнялся поиск онтологий, связанных с конкретной предметной областью, на шведском языке, который не привел к успеху. Поэтому для представления метаданных был принят стандарт Dublin Core.

SWEDISH ORAL MEDICINE WEB. ПРОДОЛЖЕНИЕ

3. Стадия разработки.

Процесс приобретения. Было решено, что перевод на OWL более крупных медицинских онтологий, написанных не на OWL, выходит за рамки данного проекта. Онтология, принятая для повторного использования, Dublin Core, небольшая и написана на OWL, поэтому никакие адаптации не производились.

Процесс разработки архитектуры. Список, составленный на стадии концепции в рамках процесса определения требований заинтересованных сторон, использовался для принятия решения о том, что должно быть представлено в виде классов, свойств и экземпляров. Было решено, что различные этапы врачебного осмотра, такие, как общий анамнез и диагноз, должны быть представлены в виде классов. С каждым из них были связаны такие свойства, как «*Имеет Аллергию*» и «*Имеет Предварительный Диагноз*». Эти свойства могут получать значения из экземпляров соответствующих классов значений, таких, как «Аллергия» и «*Диагноз*» соответственно.

Процесс реализации. Было принято решение использовать разновидность языка OWL - OWL DL. Ожидалось, что это облегчит объединение данных, собранных SOMWeb, с данными из других источников и даст возможность для использования более широкого диапазона инструментов, разработанных в соответствии с этими рекомендациями.

Процесс верификации. Для разработки онтологий SOMWeb сценарии проверки рассуждений не использовались, поскольку логическая сложность онтологий была невысока.

Процесс валидации. Разработчики онтологии обсуждали проект онтологии со специалистами в предметной области. Поскольку большая часть знаний приобреталась в проекте MedView, такая валидация уже была осуществлена ранее.

SWEDISH ORAL MEDICINE WEB. ПРОДОЛЖЕНИЕ

4. Стадия производства.

Процесс реализации. Определенные классы, свойства и экземпляры были закодированы на OWL при помощи Protégé и Jena. Прототип был создан главным образом вручную при помощи Protégé. Конечная онтология создана программным путем при помощи Jena посредством чтения шаблонов результатов стоматологических обследований в старом формате MedView и создания классов, свойств и экземпляров на OWL в соответствии с ранее принятыми проектными решениями.

Процесс верификации. Поскольку предыдущий формат MedView использовался в качестве спецификации для онтологий SOMWeb, которые переводились автоматически, было установлено, что онтологии SOMWeb соответствуют спецификации.

Процесс валидации. В прототипе, созданном при помощи Protégé, выяснили, что онтология SOMWeb может быть использована для выполнения мотивационного сценария предоставления шаблонов для протоколов стоматологических обследований.

Процесс сопровождения. Важность разработки конечным пользователем означает, что у пользователей должна быть возможность добавлять экземпляры к онтологии, однако метаданные должны быть добавлены для того, чтобы показать, кто создал концепт и с какой целью. В SOMWeb целью добавления экземпляров является потребность в протоколе стоматологического обследования. Политика управления версиями онтологии не была установлена.

SWEDISH ORAL MEDICINE WEB. ПРОДОЛЖЕНИЕ

5. Стадия эксплуатации.

Процесс функционирования. Онтология используется в качестве схемы для представления результатов стоматологических обследований в системе SOMWeb. Другие части системы, такие, как данные о членах, встречах, новостях, метаданные истории болезни, также представлены с использованием OWL и RDF.

Процесс поставки. Онтологии доступны по адресу: *http://www.somweb.se/ontologies/diagnosisOntology.owl*. Однако результаты конкретных обследований не могут быть общедоступными. Экземпляры обследований, имеющиеся в сетевом сообществе, доступны в интерактивном режиме через логин члена сообщества, причем, когда практикующие врачи будут просматривать истории болезни, они увидят данные представленные на естественном языке, а не на RDF.

Стадии сопровождения и изъятия из обращения не включены, поскольку данная разработка еще не достигла этих стадий.

ВЫОДЫ О ОБ ИСПОЛЬЗОВАНИИ СТАНДАРТА ISO/IEC 15288 ДЛЯ РАЗРАБОТКИ И РАЗВЕРТЫВАНИЯ ОНТОЛОГИЙ

- Использование стандарта ISO/IEC 15288 для разработки онтологий имеет сходства с другими методологиями построения онтологий, например с подходом, реализованным в Methontology.
- Применимость стандарта к широкому диапазону действий, связанных с системами, дает возможность добиться необходимого уровня способности к адаптации к различным конфигурациям разработки онтологий.
- Имеет смысл применять стандарт ISO/IEC 15288 на уровне онтологий.
- Рассмотрение онтологии как системы обеспечивает доступ к интересной и полезной концептуальной структуре, характеризующейся целостным представлением о программной и системной инженерии.
- Стандарт процессов жизненного цикла предоставляет основу для моделей жизненного цикла, формируемых на основе стадий, и адаптируемый базовый набор процессов, которые обеспечивают отправные точки для связи и координации. Способность стандарта к адаптации позволяет исключить процессы, которые не имеют непосредственного отношения к делу.
- Стандарт ISO/IEC 15288 обеспечивает поддержку технических систем, нетехнических систем и систем, состоящих из технических и нетехнических системных элементов. Это необходимо в случае онтологий, в составе которых нетехнические системные элементы, например познавательная способность человека и соглашение о концептах предметной области, весьма уместны.
- Стандарт и его процессы могут быть применены к разработке программного обеспечения на основе онтологий.

ОБЩЕЕ ПОНЯТИЕ АРХИТЕКТУРЫ

Изначально термин архитектура применялся в проектировании и постройке различных сооружений. Он определял их структуру, взаимосвязи между составными частями, базовые принципы их организации и дальнейшего развития.

Считается, что впервые концепцию арихитектуры предложил Витрувий (*«Десять книг об архитектуре» / Пер. Ф. А. Петровского. Т.1. М., Изд-во Академии архитектуры. (Серия «Классики теории архитектуры»). 1936*).

Витрувия называют первым онтологом в проектировании.

ПРАВИЛА ПОСТРОЙКИ ГОРОДА ПО ВИТРУВИЮ

Описания постройки городов и зданий у Витрувия – это реализация правил продукций в формате «If-Then-Else»:

Если <условие…>, То <решение 1…>, Иначе <решение 2…>, <пояснение решения>

При постройке городов или военных постов наши предки, принося в жертву пасшихся в этой местности овец, рассматривали их печень, и **если** в первый раз она оказывалась синеватой и больной, **то** приносили в жертву других, для выяснения, страдает ли скот от болезни или от дурного пастбища. И где после повторных наблюдений они удостоверялись, что печень животных здорова и не страдает от воды и пастбища, там они строили укрепления.

Если же они находили печень больной, **то** заключали отсюда, что и для людей будут вредоносны и вода, и пища, происходящие из этой местности, и потому уходили оттуда и переселялись в другие области в поисках, прежде всего, здоровых условий жизни.

Если город будет основан на болотистом месте, **то** при условии, что эти болота будут у моря, а город обращён на север или на северо-восток, болота же расположены выше морского берега, можно счесть, что город основан разумно. **Ибо** путём проведённых канав вода отводится на берег, а море, загоняемое бурями на болота, благодаря сильному прибою волн и своим морским примесям, не допускает разводиться там болотным тварям, а те из них, которые из вышележащих мест подплывают к самому берегу, уничтожаются непривычной для них соленостью.

Наоборот, там, где болота стоячие и не имеют проточного выхода ни по рекам, ни по канавам, они, застаиваясь, загнивают и испускают тяжёлые и заразные испарения на всю округу.

ТРЕБОВАНИЯ К АРХИТЕКТОРУ СИСТЕМЫ ПО ВИТРУВИЮ

- **Грамотность** необходима архитектору, чтобы поддерживать память записями.
- **Уметь рисовать** он должен для изображения при помощи рисунков задуманное им произведение.
- **Геометрия** облегчает составление планов зданий и правильное применение уровней и отвесов.
- Посредством **арифметики** составляют смету постройки, вычисляют её размеры и путём применения геометрических законов и выкладок разрешают сложные вопросы соразмерности.
- **Знакомство с историей** необходимо потому, что архитекторы часто намечают в своих произведениях многочисленные украшения, в значении которых они должны уметь дать отчёт тем, кто этого потребует.
- **Философия** возвышает дух архитектора и, искореняя в нём самонадеянность, делает его более обходительным, справедливым и честным.
- **Музыку** архитектор должен знать для того, чтобы быть осведомлённым в канонической и математической теории, а кроме того, быть в состоянии рассчитывать напряжение баллист, катапульт и скорпионов...
- **Медицину** надо знать для определения воздуха, здоровых местностей и пригодности воды...

Требуются компетенции в разных дисциплинах, т.е. **междисциплинарный подход**!

ПОНЯТИЕ АРХИТЕКТУРЫ ИНФОРМАЦИОННЫХ СИСТЕМ

С течением времени понимание термина «архитектура», применительно к техническим системам, несколько изменилось. В техническом аспекте можно рассматривать архитектуру, как высоко абстрактную модель, в которой отсутствуют подробности реализации.

Общепринятого определения понятия «архитектура информационной системы» не существует.

Архитектуру информационной системы можно описать как концепцию, определяющую модель, структуру, выполняемые функции и взаимосвязь компонентов информационной системы.

Процедура выбора архитектуры для проектируемой информационной системы, в рыночных условиях, сводится к определению стоимости владения ею. Стоимость владения информационной системой складывается из плановых затрат и стоимости рисков.

Концепция архитектуры информационной системы должна формироваться ещё на этапе технико-экономического обоснования и выбираться такой, чтобы стоимость владения ею была минимальной.

Архитектуру информационной системы можно рассматривать как модель, определяющую стоимость владения через имеющуюся в данной системе инфраструктуру.

Архитектура системы — принципиальная организация системы, воплощенная в её элементах, их взаимоотношениях друг с другом и со средой, а также принципы, направляющие её проектирование и эволюцию (*Википедия*).

ТИПЫ АРХИТЕКТУР

Для архитектуры крупных организаций принято использовать понятие «корпоративная архитектура», которую можно представить в виде совокупности нескольких типов архитектур:

- бизнес архитектура (Business architecture);
- ИТ-архитектура (Information Technology architecture);
- архитектура данных (Data architecture);
- программная архитектура (Software architecture);
- техническая архитектура (Hardware architecture).

ТИПЫ АРХИТЕКТУР. ПРОДОЛЖЕНИЕ

Техническая архитектура является первым уровнем архитектуры информационной системы. Она описывает все аппаратные средства, использующиеся при выполнении заявленного набора функций, а также включает средства обеспечения сетевого взаимодействия и надёжности.

Программная архитектура представляет собой совокупность компьютерных программ, предназначенных для решения конкретных задач. Данный тип архитектуры необходим для описания приложений, входящих в состав информационной системы. На данном уровне описывают программные интерфейсы, компоненты и поведение.

Архитектура данных объединяет в себе как физические хранилища данных, так и средства управления данными. Кроме того, в неё входят логические хранилища данных, а при ориентированности рассматриваемой компании на работу со знаниями, может быть выделен отдельный уровень – архитектура знаний (Knowledge architecture). На этом уровне описываются логические и физические модели данных, определяются правила целостности, составляются ограничения для данных.

ИТ-архитектура является связующим уровнем. Здесь формируется базовый набор сервисов, которые используются как на уровне программной архитектуры, так и на уровне архитектуры данных. Основной функцией ИТ-архитектуры является обеспечение функционирования важных бизнес-приложений для достижения обозначенных бизнес-целей. Если некоторая функция требуется сразу в нескольких приложениях, то её следует перенести на уровень ИТ-архитектуры, тем самым повысив интеграцию системы и снизить сложность архитектуры приложений.

Бизнес-архитектуры или архитектуры бизнес-процессов. На этом уровне определяются стратегии ведения бизнеса, способы управления, принципы организации и ключевые процессы, представляющие наибольшую важность.

ТИПЫ АРХИТЕКТУР. МИКРОАРХИТЕКТУРА И МАКРОАРХИТЕКТУРА

Термины **микроархитектура и макроархитектура** в большей степени применяются для описания программных систем.

В соответствие с рассмотренной моделью уровней архитектур корпоративных информационных систем, микроархитектуру можно отнести к уровням программной архитектуры и архитектуры данных, а макроархитектуру – к уровню ИТ-архитектуры.

Микроархитектура описывает внутреннее устройство конкретного компонента или подсистемы, а **макроархитектура** описывает устройство всей ИС, как совокупности её компонент или подсистем.

Без применения **архитектурного подхода** при построении сложных систем, их создание, обслуживание и модификация, в конце концов, станут нерентабельными для бизнеса.

АРХИТЕКТУРНЫЙ ПОДХОД К ПРОЕКТИРОВАНИЮ ИС

Процесс проектирования информационной системы тесным образом связан с её архитектурным описанием, что отражено в некоторых определениях термина «архитектура».

Можно выделить пять различных **подходов к проектированию**:

- Календарный подход.
- Подход, на основе процесса управления требованиями.
- Подход, основанный на процессе разработки документации.
- Подход на основе системы управления качеством.
- **Архитектурный подход**.

Архитектурный подход к проектированию информационных систем можно считать наиболее зрелым. Его ключевым аспектом является создание или выбор фреймворка, адаптация которого под нужды конкретной системы будет легко осуществима. В соответствии с этим, задача проектирования разбивается на две: разработка многократно используемого каркаса и создание системы на его основе. При использовании каркасов появляется возможность довольно быстро изменять функциональность системы за счёт итеративности процесса проектирования.

АРХИТЕКТУРНОЕ ОПИСАНИЕ. СТАНДАРТ ISO/IEC 42010

Согласно ISO/IEC 42010 IEEE Std 1471-2000 (ГОСТР 57100) System and software engineering – Recommended practice for architectural description of software-intensive systems архитектура может иметь несколько представлений, отражающих структуру ПО с разных точек зрения. Стандарт рекомендует для каждого представления фиксировать отраженные в нем взгляды и интересы, причины, обуславливающие необходимость такого рассмотрения системы, несоответствия между элементами одного представления или между различными представлениями, а также различную служебную информацию.

Подход к архитектурному описанию (ISO 42010 -- architecture framework: conventions, principles and practices for the description of architectures established within a specific domain of application and/or community of stakeholders) включает в себя:

а) описание функций системы как отдельную тематическую группу описаний (ISO 42010 -- architecture view: work product expressing the architecture of a system from the perspective of specific system concerns), отражающее необходимость целевой системы для объемлющей надсистемы;

б) указания на метод архитектурной работы (из ISO 42010: architecting -- process of conceiving, defining, expressing, documenting, communicating, certifying proper implementation of, maintaining and improving an architecture throughout a system's life cycle), описание функций системы обычно становится первым же разрабатываемым артефактом.

АРХИТЕКТУРНОЕ ОПИСАНИЕ. СТАНДАРТ ISO/IEC 42010. ПРОДОЛЖЕНИЕ

Для более подробного описания принципов построения архитектуры стандарт ISO/IEC/IEEE 42010-2011 вводит следующие понятия:

Архитектурная группа описаний (англ. architectural view) — представление системы в целом с точки зрения связанного набора интересов. Каждая группа описаний относится к одному или более стейкхолдеру. Термин «группа описаний» употребляется для выражения архитектуры системы при некотором методе описания.

Архитектурное описание (англ. architectural description) — рабочий продукт, использующийся для выражения архитектуры.

Архитектурный подход (англ. architectural framework) — соглашения, принципы и практики для описания архитектуры, установленные для конкретной области применения и/или конкретным сообществом стейкхолдеров.

АРХИТЕКТУРНОЕ ОПИСАНИЕ. СТАНДАРТ ISO/IEC 42010. ПРОДОЛЖЕНИЕ

Архитектурный метод описания (англ. architectural viewpoint) — спецификация соглашений для конструирования и применения группы описаний. Шаблон или образец, по которому разрабатываются отдельные группы описаний посредством установления назначений и аудитории для группы описаний, а также приемы их создания и анализа. Метод описания устанавливает соглашения, по которым группа описаний создается, отображается и анализируется. Тем самым метод описания определяет языки (включая нотации, описания или типы продуктов), применяемые для определения группы описаний, а также все связанные методы моделирования или приемы анализа, применяемые к данным представлениям группы описаний. Данные языки и приемы применяются для получения результатов, имеющих отношение к адресуемым интересам.

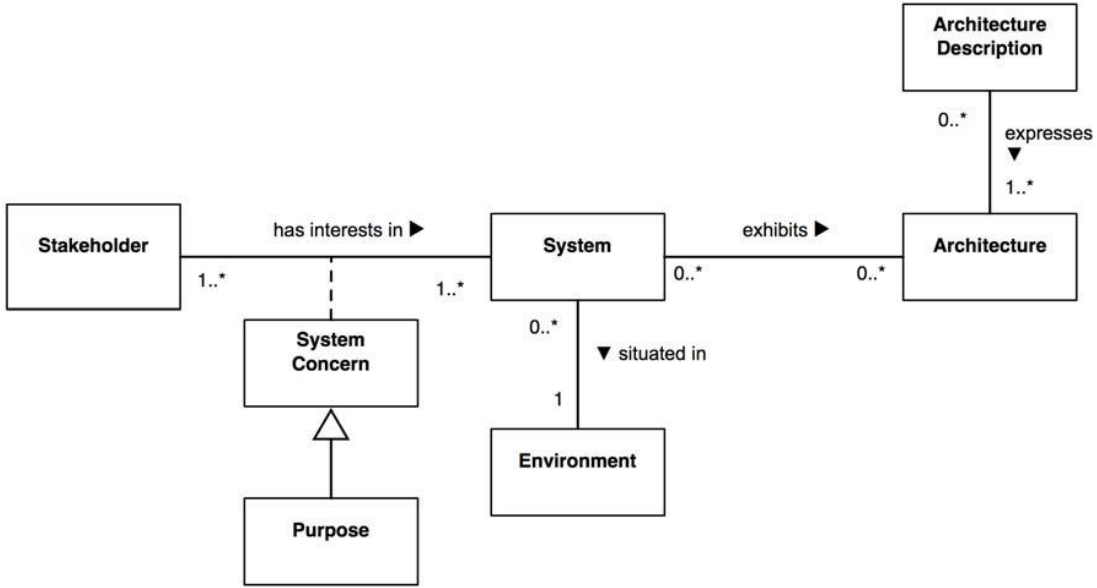
Понятие модели. M является моделью системы S, если M может использоваться для ответа на вопросы о S. M является моделью относительно множества вопросов O, если и только если M может использоваться, чтобы ответить на вопросы об объекте A в O в пределах приемлемости Г.

Вид модели (англ. model kind) — соглашения по средствам моделирования (например, сети Петри, диаграммы классов, организационные диаграммы и т. д.).

КОНТЕКСТ АРХИТЕКТУРЫ СОГЛАСНО ISO 42010

Архитектура какой-либо системы представляет собой то, что является существенным относительно рассматриваемой системы в ее окружающей среде. Не существует единственной характеристики того, что является существенным или основным для системы. Такая характеристика может принадлежать к следующим категориям:

- системным компонентам или элементам;
- тому, как системные элементы устроены или взаимосвязаны;
- принципам организации системы или проекта;
- принципам, управляющим развитием системы е ее жизненном цикле.

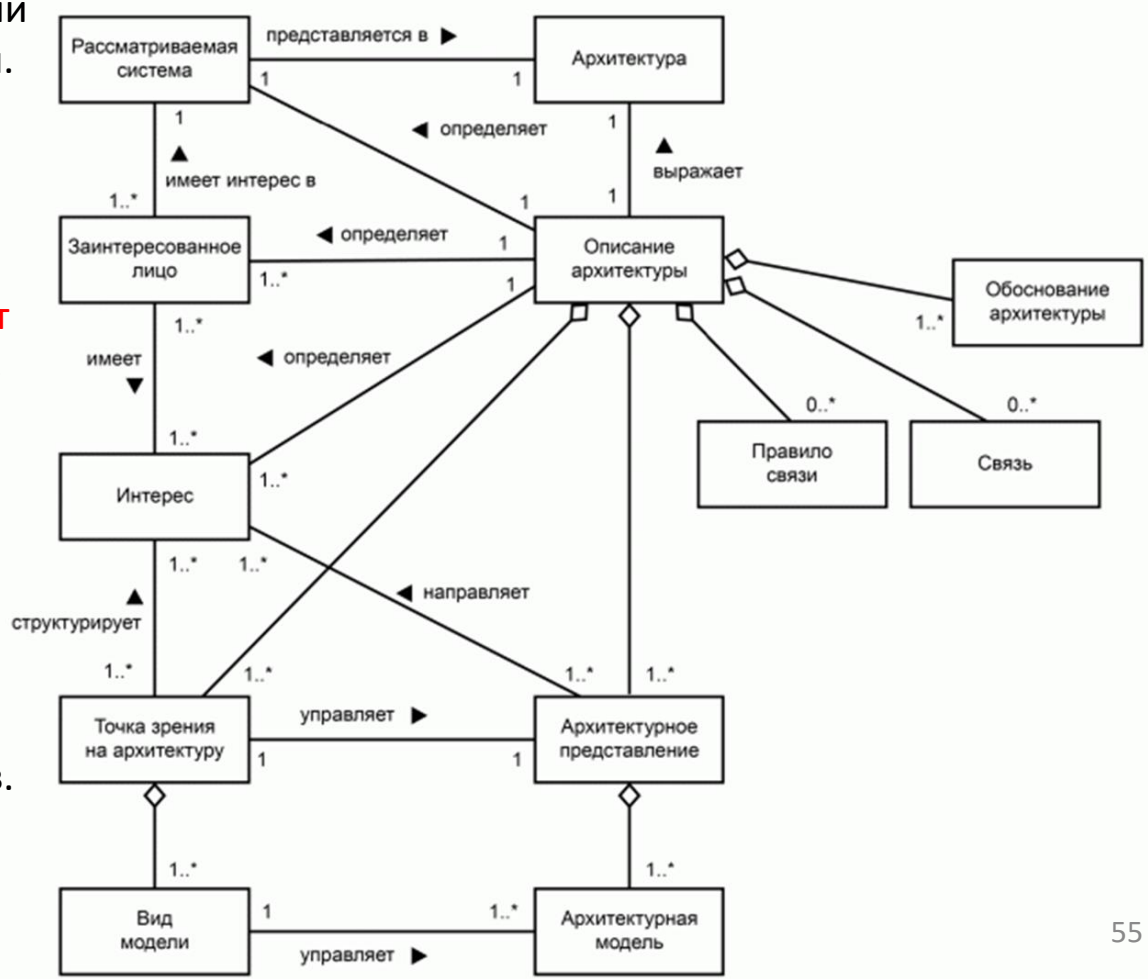


КОНЦЕПТУАЛЬНАЯ МОДЕЛЬ ОПИСАНИЯ АРХИТЕКТУРЫ

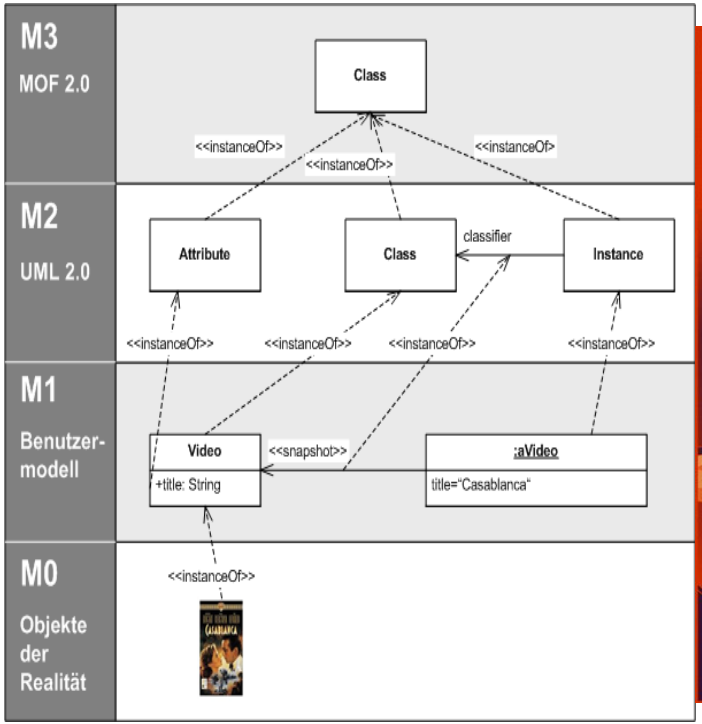
Описание архитектуры включает одно или несколько архитектурных представлений.

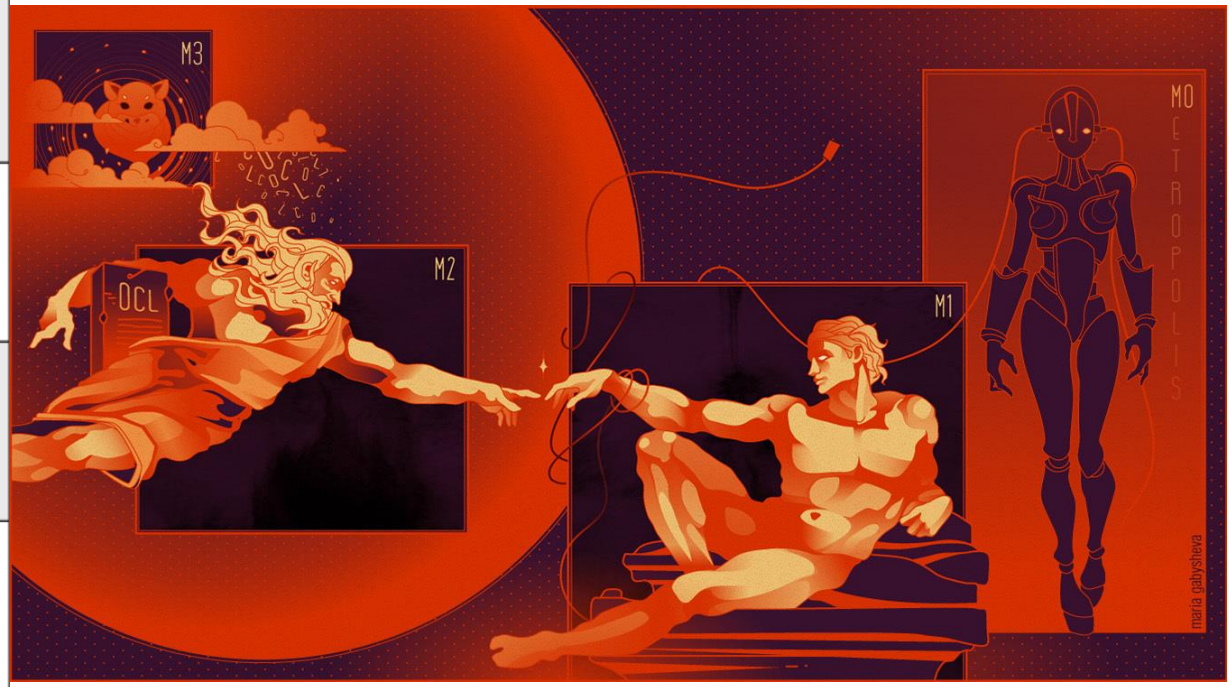
Архитектурное представление обращается к одному или более интересам заинтересованных сторон системы.

Архитектурное представление выражает архитектуру рассматриваемой системы в соответствии с архитектурной точкой зрения. Точка зрения имеет два аспекта: интересы, которые структурно представляются для заинтересованных сторон, и условности, которые устанавливаются в представлениях. Какая-либо архитектурная точка зрения структурирует один или более интересов. Интерес может быть структурирован более, чем одной точкой зрения.



ЯЗЫКИ АРХИТЕКТУРНОГО МОДЕЛИРОВАНИЯ НА ОСНОВЕ UML





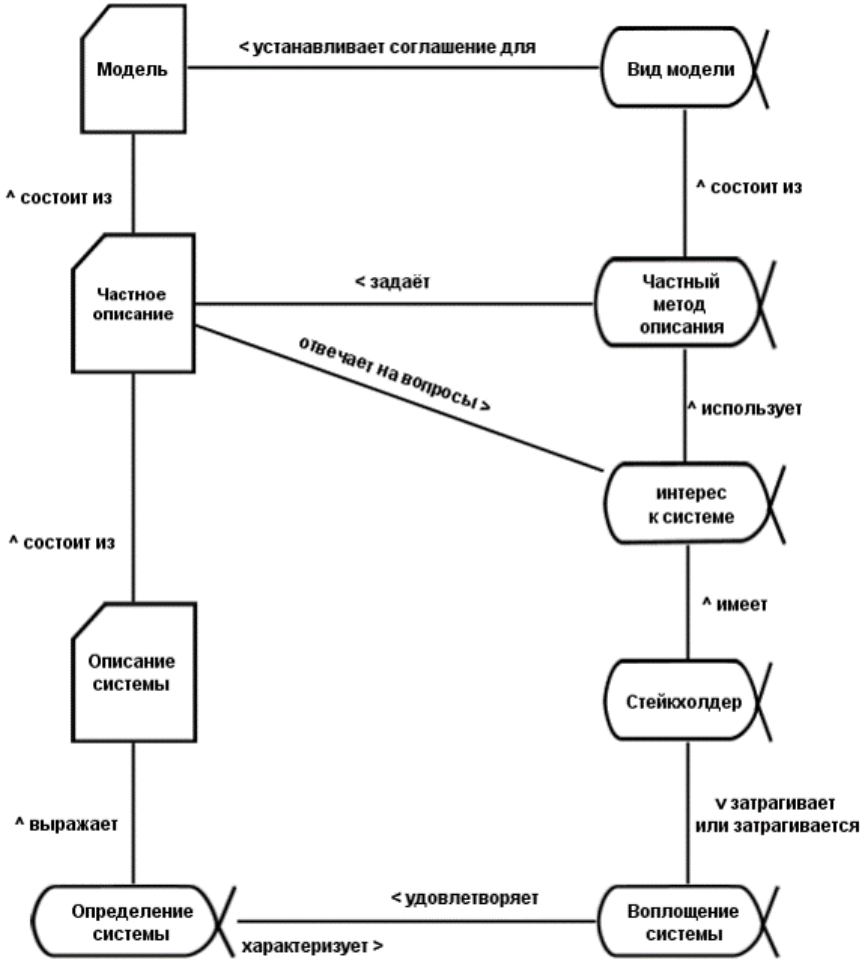
ВОЗМОЖНЫЕ ОБЛАСТИ ПРИМЕНЕНИЯ ОПИСАНИЙ АРХИТЕКТУРЫ

- в качестве основы системного проекта и действий по разработке системы;
- в качестве основы анализа и оценки альтернативных реализаций архитектуры;
- в качестве документации по разработке и сопровождении;
- для обеспечения документирования существенных аспектов системы;
- для использования автоматизированных средств моделирования, системной имитации и анализа;
- для обеспечения связи между сторонами, вовлеченными в разработку, производство, развертывание, функционирование и сопровождение системы;
- для обеспечения связи между заказчиками, приобретающими сторонами, поставщиками и разработчиками как части контрактных переговоров;
- для обеспечения документирования характеристик, свойств и проекта системы для потенциальных заказчиков, приобретающих сторон, собственников, операторов и интеграторов;
- при планировании перехода от устаревшей архитектуры к новой;
- в качестве руководства по эксплуатационной и инфраструктурной поддержке и управлению конфигурацией;
- для поддержки системного планирования и действий, связанных со сроками и бюджетом;
- для установления критериев при сертификации реализаций на соответствие архитектуре;
- в качестве основы для ревизий, анализа и оценки системы в ее жизненном цикле.

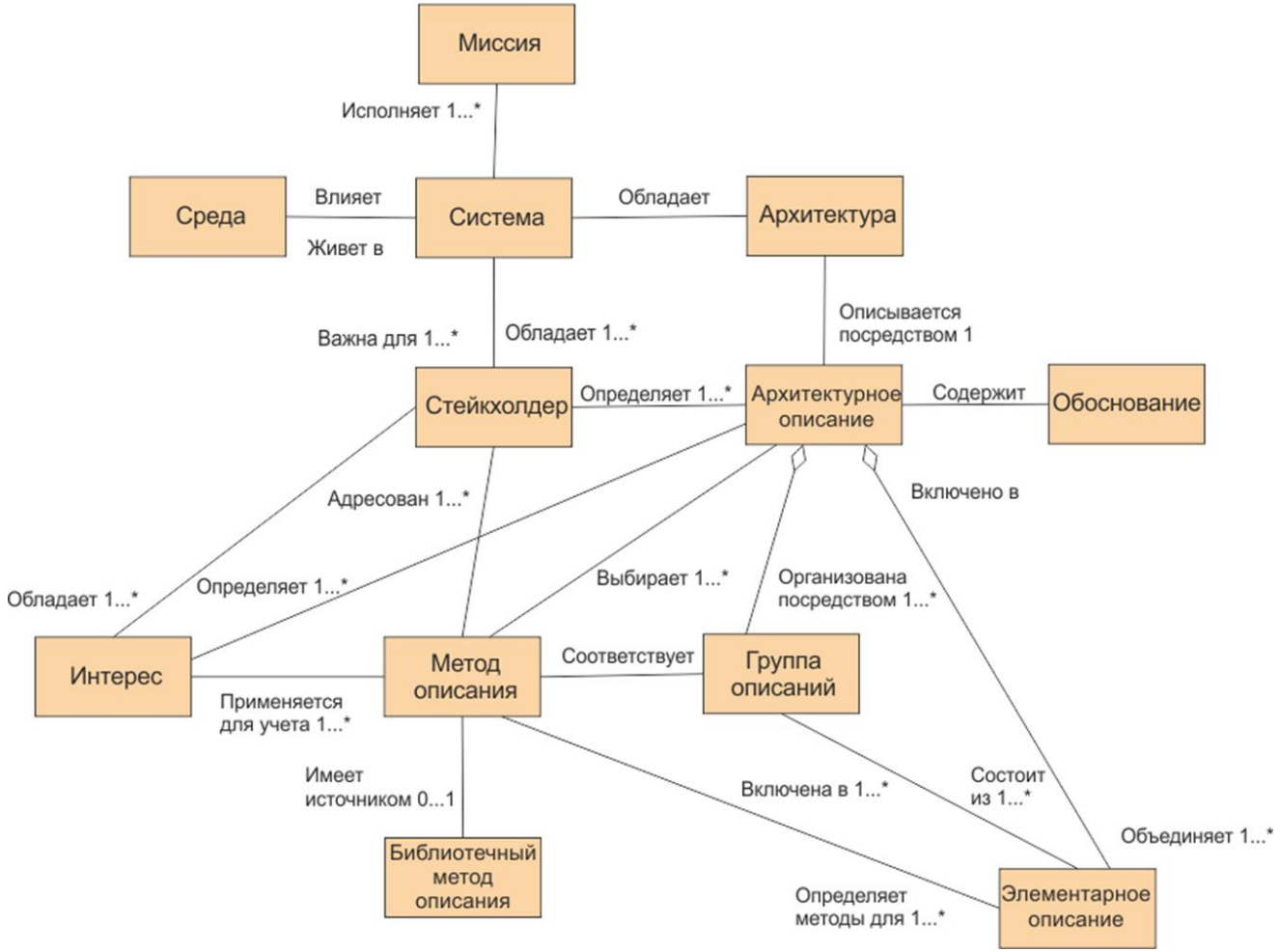
ВЗАИМОСВЯЗЬ ОПИСАНИЙ АРХИТЕКТУРЫ СО СТАНДАРТАМИ ЖЦ

<table>
<tr><th>Стандарт / Процессы ЖЦ</th><th>Пример архитектурной точки зрения</th></tr>
<tr><td>ИСО/МЭК 12207:2010. Проектирование архитектуры системы и проектирование архитектуры программных средств .</td><td rowspan="2">Точка зрения декомпозиции и распределения структурирует следующие интересы:
• определение системных требований;
• декомпозицию системы на объекты;
• распределение требований по объектам;
• верификацию того, что все требования распределены по объектам.</td></tr>
<tr><td>ИСО/МЭК 15288:2008. Проектирование архитектуры.</td></tr>
</table>

ВАРИАНТ СОВМЕЩЕНИЯ ДИАГРАММ ISO 42010 И OMG ESSENCE



ПРИМЕР АРХИТЕКТУРНОГО ОПИСАНИЯ



АРХИТЕКТУРНЫЕ ФРЕЙМВОРКИ

В соответствии со стандартом ISO/IEC 42010 **архитектурный фреймворк** определяется как «совокупность соглашений, принципов и практик, используемых для описания архитектур и принятых применительно к некоторому предметному домену и (или) в сообществе специалистов (заинтересованных лиц)». Архитектурный фреймворк включает в себя описание заинтересованных лиц, типовые проблемы предметной области, архитектурные точки зрения и методы их интеграции.

В качестве примеров можно выделить пять наиболее известных фреймворков :

- Фреймворк Захмана.
- TOGAF.
- DoDAF.
- FEA.
- Gartner.

АРХИТЕКТУРНЫЙ ФРЕЙМВОРК DoDAF

Фреймворк миистерства обороны США DoDAF (Department of Defense Architecture Framework) состоит из трёх основных элементов: **модели** (models), **виды** (views) и **точки зрения** (viewpoints). Этот набор позволяется отражать взгляды всех заинтересованных сторон. Несмотря на военное назначение, DoDAF является свободно распространяемым. На его базе были сформированы фреймворки НАТО (NATO Architecture Framework – NAF), фреймворк Министерства Обороны Великобритании (Ministry of Defense Architecture Framework – MODAF) и множество других.

Главная особенность DoDAF – ориентация на данные, что подразумевает повышенную важность сохранности данных и повторного их использования. Основным классом систем, которые проектируются при помощи этого фреймворка, являются системы сбора, хранения и анализа данных для поддержки принятия решений (СППР).

DoDAF предназначен для составления **архитектурного описания системы**. Результатом его применения будет являться набор документов, а не информационная система.

МОДЕЛИ ФРЕЙМВОРКА DoDAF

DoDAF определяет **модели**, как шаблоны для сбора данных и подразделяет их на классы:

- таблицы;
- графические изображения структурных аспектов архитектурного решения;
- графические изображения поведенческих аспектов архитектурного решения;
- отображения, определяющие взаимосвязь между типами информации;
- онтологии;
- картинки в свободном формате;
- временные диаграммы.

Виды определяются способами представления для пользователя связанного набора данных. К видам можно отнести документы, таблицы, графики, диаграммы и т.д.

Точки зрения представляют собой **упорядоченное множество видов**.

ТОЧКИ ЗРЕНИЯ DoDAF

Вторая версия DoDAF содержит **восемь точек зрения**:

- обобщенная (All Viewpoint): интегрирует все точки зрения для создания архитектурного контекста;
- определяющая потенциальные возможности (Capability Viewpoint): обучение персонала, сроки поставок и т.д.;
- определяющая данные и информацию (Data and Information Viewpoint): определяет способы представления и структуры данных;
- операционная (Operational Viewpoint): рассматривает сценарии работы и активности системы;
- проектная (Project Viewpoint): рассматривает требуемые характеристики и возможности системы;
- сервисная (Service Viewpoint): рассматривает совокупность сервисов;
- учитывающая стандарты (Standards Viewpoint): рассматривает технические стандарты, ограничения, методики, руководства и т.д.;
- системная (System Viewpoint): рассматривает совокупность взаимодействующих систем и их взаимодействие.

МЕТА-МОДЕЛЬ ДАННЫХ DoDAF

DoDAF использует **мета-модель данных** (Data Meta-Model – DM2), которая является онтологией, составленной из уровней, отражающих особенности представления информации для конкретных групп пользователей. DM2 может быть расширена.

В ней определены три уровня:

1. Концептуальная модель данных (Conceptual Data Model) – описывает архитектуру в нетехнические терминах.
2. Логическая модель данных (Logical Data Model) – расширение концептуальной модели путём добавления атрибутов.
3. Спецификация обмена данными на физическом уровне (Physical Exchange Specification) – средство, обеспечивающее обмен информации между моделями.

БАЗОВЫЕ ПРИНЦИПЫ ПРИМЕНЕНИЯ DoDAF

Существует **восемь базовых принципов**, руководствуясь которыми, можно успешно применять DoDAF:

1. Архитектурное описание должно быть чётко ориентировано на провозглашённые цели.
2. Архитектурное описание должно быть по возможности простым и понятным, но не упрощённым.
3. Архитектурное описание должно облегчать, а не затруднять процесс принятия решений.
4. Архитектурное описание должно быть составлено таким образом, чтобы его можно было использовать для сравнения различных архитектур.
5. При составлении архитектурного описания должны в максимальной степени использоваться стандартные типы данных, определяемые в мета-модели данных DM2.
6. Архитектурное описание должно выполняться в терминах самих данных, а не инструментальных средств работы с данными.
7. Архитектурные данные должны быть организованы в виде, удобном для групповой работы.
8. Архитектурное описание должно быть построено таким образом, чтобы его можно было использовать в сетевой среде.

КАЧЕСТВО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Понятие качественного ПО соответствует представлению о том, что программа достаточно успешно справляется со всеми возложенными на нее задачами и не приносит проблем ни конечным пользователям, ни службе поддержки, ни специалистам по продажам. На создание качественного ПО существенное влияние оказывает качество технологических процессов его разработки. Общие принципы обеспечения качества процессов производства во всех отраслях экономики регулируются набором стандартов серии ISO 9000.

НАИБОЛЕЕ ВАЖНЫЕ СТАНДАРТЫ КАЧЕСТВА ПО

- **ISO 9001:2000** Quality management systems – Requirements. Системы управления качеством – Требования. Этот стандарт определяет общие правила обеспечения качества результатов во всех процессах жизненного цикла изделия.
- **ISO/IEC 90003:2004** Software engineering – Guidelines for the application of ISO 9001:2000 to computer software. Разработка программного обеспечения – Руководящие положения по применению стандарта ISO 9001:2000 к программному обеспечению. Этот стандарт конкретизирует положения ISO 9001 для разработки программного обеспечения с упором на обеспечение качества при процессе проектирования. Он также определяет некоторый набор техник и процедур, которые рекомендуется применять для контроля и обеспечения качества разрабатываемых программ.
- **ISO/IEC TR 90005:2008** Software engineering – Guidelines for the application of ISO 9001:2000 to system life cycle processes. Разработка программного обеспечения – Руководящие положения по применению стандарта ISO 9001:2000 к процессам жизненного цикла программных систем. Этот стандарт конкретизирует положения по применению ISO 9001:2000 для приобретения, поставки, разработки, применения и сопровождения программных систем. Он не добавляет и не изменяет требования ISO 9001:2000. ISO/IEC TR 90005:2008 использует ISO/IEC 15288 как отправную точку для формирования требований обеспечения качества при проектировании программных систем.

НАИБОЛЕЕ ВАЖНЫЕ СТАНДАРТЫ КАЧЕСТВА ПО. ПРОДОЛЖЕНИЕ

- **Стандарт ISO 9126** предлагает использовать для описания качества ПО многоуровневую модель показателей качества. На верхнем уровне выделено 6 основных характеристик качества ПО. Каждая характеристика описывается при помощи набора атрибутов, позволяющих оценить эту характеристику. Для каждого атрибута определяется набор метрик, позволяющих оценить этот атрибут. Для каждой метрики определяется набор оценочных элементов, позволяющих оценить эту метрику. Построение модели качества позволяет системно описать требования к программному обеспечению, определяя, какие свойства ПО по данной характеристике хотят видеть заинтересованные стороны. Показатели качества, закрепленные в стандартах, не исчерпывают полностью понятие качества ПО. В зависимости от назначения программного обеспечения перечень показателей качества может быть расширен или сужен в рамках проекта по разработке конкретного ПО.
- **ISO/IEC 9126-1:2001** Software engineering – Product quality – Part 1: Quality model . Определяет набор характеристик и атрибутов качества программного обеспечения.
- **ISO/IEC 9126-2:2003** Software engineering – Product quality – Part 2: External metrics .
- **ISO/IEC 9126-3:2003** Software engineering – Product quality – Part 3: Internal metrics .
- **ISO/IEC 9126-4:2004** Software engineering – Product quality – Part 4: Quality in use metrics.

СТАНДАРТЫ ТЕСТИРОВАНИЯ ПО

- **ISO/IEC 25051:2006** Software engineering – Software product Quality Requirements and Evaluation (SQuaRE) – Requirements for quality of Commercial Off-The-Shelf (COTS) software product and instructions for testing. Разработка программного обеспечения – Оценка и требования качества к программному обеспечению – Требования для качества готового коммерческого программного продукта и инструкций для испытания. Определяет требования качества к программным продуктам и к документации по тестированию. Регламентирует содержание документации по тестированию, которая должна включать план тестирования, описание используемых методов тестирования, описание наборов тестов и результатов тестирования. Этот стандарт в 2006 году пришел на смену ISO/IEC 12119:1994.
- **IEEE 829-1998** Standard for Software Test Documentation . Описывает базовый набор документов для тестирования программного обеспечения. Стандарт также определяет форму и содержание тестовых документов.
- **IEEE 829-2008** Standard for Software and System Test Documentation . Стандарт применяется к программным системам.
- **IEEE 1008-1987** (R1993, R2002) Standard for Software Unit Testing . Описывает организацию модульного тестирования.

ПРОЦЕСС ОЦЕНКИ ХАРАКТЕРИСТИК КАЧЕСТВА ГОТОВОГО ПО

Процессу оценки характеристик качества готовых программных средств и их компонентов (программных продуктов) на различных этапах жизненного цикла посвящен международный стандарт **ISO 14598**, состоящий из шести частей. Для каждой характеристики качества рекомендуется формировать меры и шкалу измерений с выделением требуемых, допустимых и неудовлетворительных значений. Реализация процессов оценки должна коррелировать с этапами жизненного цикла конкретного проекта программного средства в соответствии с применяемой адаптированной версией стандарта **ISO 12207**.

- **ISO/IEC 14598-1:1999** Information technology – Software product evaluation – Part 1: General overview .
- **ISO/IEC 14598-2:2000** Software engineering – Product evaluation – Part 2: Planning and management .
- **ISO/IEC 14598-3:2000** Software engineering – Product evaluation – Part 3: Process for developers.
- **ISO/IEC 14598-4:1999** Software engineering – Product evaluation – Part 4: Process for acquirers.
- **ISO/IEC 14598-5:1998** Information technology – Software product evaluation – Part 5: Process for evaluators .
- **ISO/IEC 14598-6:2001** Software engineering – Product evaluation – Part 6: Documentation of evaluation modules.

АСПЕКТЫ КАЧЕСТВА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Качеством программного обеспечения можно считать совокупность его характеристик, относящихся к возможности удовлетворять высказанные или подразумеваемые потребности всех заинтересованных лиц.

Данное определение включено в стандарт ISO 9126, в котором также определены и сами характеристики.

Три **аспекта качества**:

1. **Внутреннее качество** (характеристики самого программного обеспечения).
2. **Внешнее качество** (поведенческие характеристики программного обеспечения).
3. **Контекстное качество** (ощущения пользователей при различных контекстах использования).

ХАРАКТЕРИСТИКИ КАЧЕСТВА ПО

Руководствуясь этими аспектами, **стандарт ISO 9126 выделяет шесть характеристик качества программного обеспечения**:

1. Функциональность.
2. Надёжность.
3. Производительность.
4. Удобство использования.
5. Удобство сопровождения.
6. Переносимость.

Данные характеристики описывают внутренние и внешние аспекты качества программного обеспечения.

ХАРАКТЕРИСТИКИ КАЧЕСТВА ПО. ФУНКЦИОНАЛЬНОСТЬ

Функциональность подразумевает способность ПО решать задачи в определённых условиях и подразделяется на следующие подхарактеристики:

- функциональная пригодность (suitability) – способность решать нужный набор задач;
- точность (accuracy) – способность получать требуемые результаты;
- способность к взаимодействию (interoperability) – способность взаимодействия с требуемым набором иных систем;
- защищённость (security) – способность предотвращать неавторизованный доступ к данным и программам;
- соответствие стандартам и правилам (compliance) – соответствие программного обеспечения различным регламентирующим нормам.

ХАРАКТЕРИСТИКИ КАЧЕСТВА ПО. НАДЁЖНОСТЬ

Надёжность (reliability) характеризуется способностью программного обеспечения удерживать функциональность в заданных рамках при определённых условиях и подразделяется на следующие подхарактеристики:

- зрелость (maturity) – величина, обратная частоте отказов программного обеспечения;
- устойчивость к отказам (fault tolerance) – способность удерживать определённый уровень работоспособности при различных отказах и нарушениях правил взаимодействия с окружением;
- способность к восстановлению (recoverability) – способность восстанавливать требуемый уровень работоспособности после отказа;
- соответствие стандартам надёжности (reliability compliance).

ХАРАКТЕРИСТИКИ КАЧЕСТВА ПО. ПРОИЗВОДИТЕЛЬНОСТЬ

Производительность (efficiency) определяется способностью программного обеспечения при определённых условиях гарантировать требуемую работоспособность по отношению к выделяемым для этого ресурсам. Это - отношение получаемых результатов к затраченным ресурсам. Данная характеристика подразделяется на следующие подхарактеристики:

- временная эффективность (time behavior) – способность программного обеспечения получать требуемые результаты и обеспечивать передачу необходимого объёма данных за определённое время;
- эффективность использования ресурсов (resource utilization) – способность программного обеспечение решать требуемые задачи и использованием заданных объёмов определённых видов ресурсов;
- соответствие стандартам производительности (efficiency compliance).

ХАРАКТЕРИСТИКИ КАЧЕСТВА ПО. УДОБСТВО ИСПОЛЬЗОВАНИЯ

Удобство использования (usability) характеризуется привлекательностью для пользователей, удобством в обучении и использовании программного обеспечения. В своём составе имеет ряд подхарактеристик:

- понятность (understandability) – величина обратная усилиям, затраченным пользователями, по осознанию применимости программного обеспечения для решения требуемых задач;
- удобство работы (operability) – величина обратная усилиям, затраченным пользователями, для решения требуемых задач при помощи программного обеспечения;
- удобство обучения (learnability) – величина обратная усилиям, затраченным пользователями, на процесс обучения работе с программным обеспечением;
- привлекательность (attractiveness) – способность программного обеспечения быть привлекательным для пользователей;
- соответствие стандартам удобства использования (usability compliance).

ХАРАКТЕРИСТИКИ КАЧЕСТВА ПО. УДОБСТВО СОПРОВОЖДЕНИЯ

Удобство сопровождения (maintainability) характеризуется удобством сопровождения программного обеспечения. Данная характеристика включает ряд подхарактеристик:

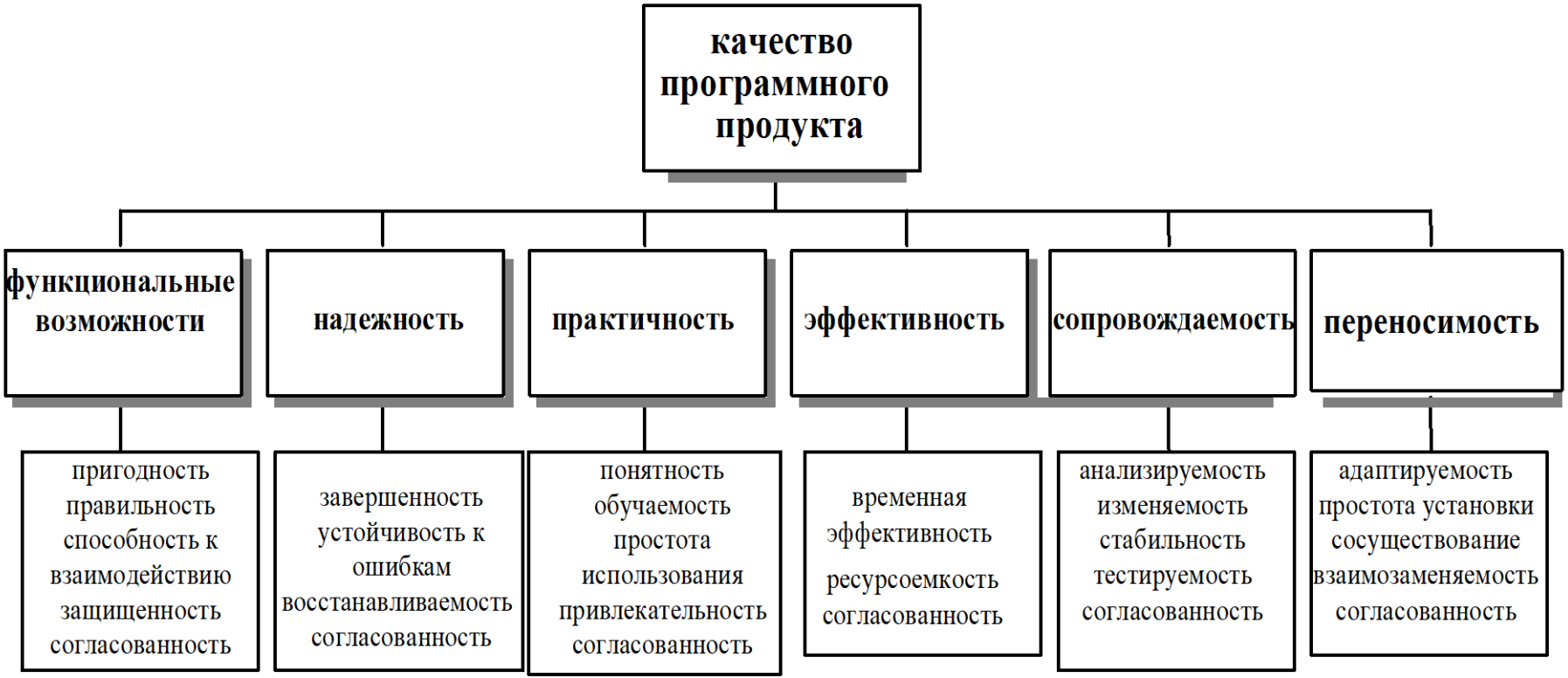
- анализируемость (analyzability) характеризуется удобством проведения анализа ошибок, дефектов, недостатков, необходимостей внесения изменений и их возможных последствий;
- удобство внесения изменений (changeability) – величина обратная трудозатратам на выполнение требуемых изменений;
- стабильность (stability) – величина обратная риску появления непредусмотренных последствий при внесении требуемых изменений;
- удобство проверки (testability) – величина обратная требуемым трудозатратам на тестирование и другие виды проверок достижения предусмотренных результатов при внесении изменений;
- соответствие стандартам удобства сопровождения (maintainability compliance).

ХАРАКТЕРИСТИКИ КАЧЕСТВА ПО. ПЕРЕНОСИМОСТЬ

Переносимость (portability) характеризуется способностью программного обеспечения сохранять работоспособность при изменении организационных, аппаратных и программных аспектов окружения. Для этой характеристики выделяются следующие подхарактеристики:

- адаптируемость (adaptability) – способность программного обеспечения без совершения непредусмотренных действий приспосабливаться к изменениям окружения;
- удобство установки (installability) – способность программного обеспечения устанавливаться в заранее определённое окружение;
- способность к сосуществованию (coexistence) – способность программного обеспечения функционировать в общем окружении с другими программами, разделяя с ними ресурсы;
- удобство замены (replaceability) – возможность применения программного обеспечения вместо уже используемого для решения тех же задач, в том же окружении;
- соответствие стандартам переносимости (portability compliance).

МОДЕЛЬ КАЧЕСТВА ПО



ХАРАКТЕРИСТИКИ КОНТЕКСТНОГО АСПЕКТА КАЧЕСТВА ПО

С точки зрения **контекстного аспекта качества** выделяют следующий уменьшенный набор характеристик:

- эффективность (effectiveness) – способность программного обеспечения решать пользовательские задачи с заданной точностью и в заданном контексте;
- продуктивность (productivity) – способность программного обеспечения получать требуемые результаты при использовании заранее определённого количества ресурсов;
- безопасность (safety) – способность программного обеспечения поддерживать требуемый низкий уровень риска нанесения ущерба людям, бизнесу и окружающей среде;
- удовлетворённость пользователей (satisfaction) – способность программного обеспечения при использовании в определённом контексте приносить удовлетворение пользователям.

МЕТРИКИ КАЧЕСТВА ПО

Типы метрик. Существует три типа метрик:

1. метрики программного продукта, которые используются при измерении его характеристик свойств;
2. метрики процесса, которые используются при измерении свойства процесса ЖЦ создания продукта.
3. метрики использования.

Метрики программного продукта включают:

- внешние метрики, обозначающие свойства продукта, видимые пользователю;
- внутренние метрики, обозначающие свойства, видимые только команде разработчиков.

Стандарт ISO/IEC 9126-2 определяет следующие типы мер:

мера размера ПО в разных единицах измерения (число функций, строк в программе, размер дисковой памяти и др.);

мера времени (функционирования системы, выполнения компонента и др.);

мера усилий (производительность труда, трудоемкость и др.);

мера учета (количество ошибок, число отказов, ответов системы и др.).

МЕТРИКИ КАЧЕСТВА ПО. ПРОДОЛЖЕНИЕ

Специальной мерой может служить **уровень использования повторных компонентов** и измеряется как отношение размера продукта, изготовленного из готовых компонентов, к размеру системы в целом. Данная мера используется также при определении стоимости и качества ПО. Примеры метрик:

- общее число объектов и число повторно используемых;
- общее число операций, повторно используемых и новых операций;
- число классов, наследующих специфические операции;
- число классов, от которых зависит данный класс;
- число пользователей класса или операций и др.

Примером широко используемых внешних метрик программ являются метрики Холстеда - это характеристики программ, выявляемые на основе статической структуры программы на конкретном языке программирования: число вхождений наиболее часто встречающихся операндов и операторов; длина описания программы как сумма числа вхождений всех операндов и операторов и др.

В качестве **метрик процесса** могут быть время разработки, число ошибок, найденных на этапе тестирования и др.

Метрики использования служат для измерения степени удовлетворения потребностей пользователя при решении его задач.

СТАНДАРТНАЯ ОЦЕНКА ЗНАЧЕНИЙ ПОКАЗАТЕЛЕЙ КАЧЕСТВА

По определению стандарта ISO/IES 9126-2 **метрика качества ПО** представляет собой "модель измерения атрибута, связываемого с показателем его качества".

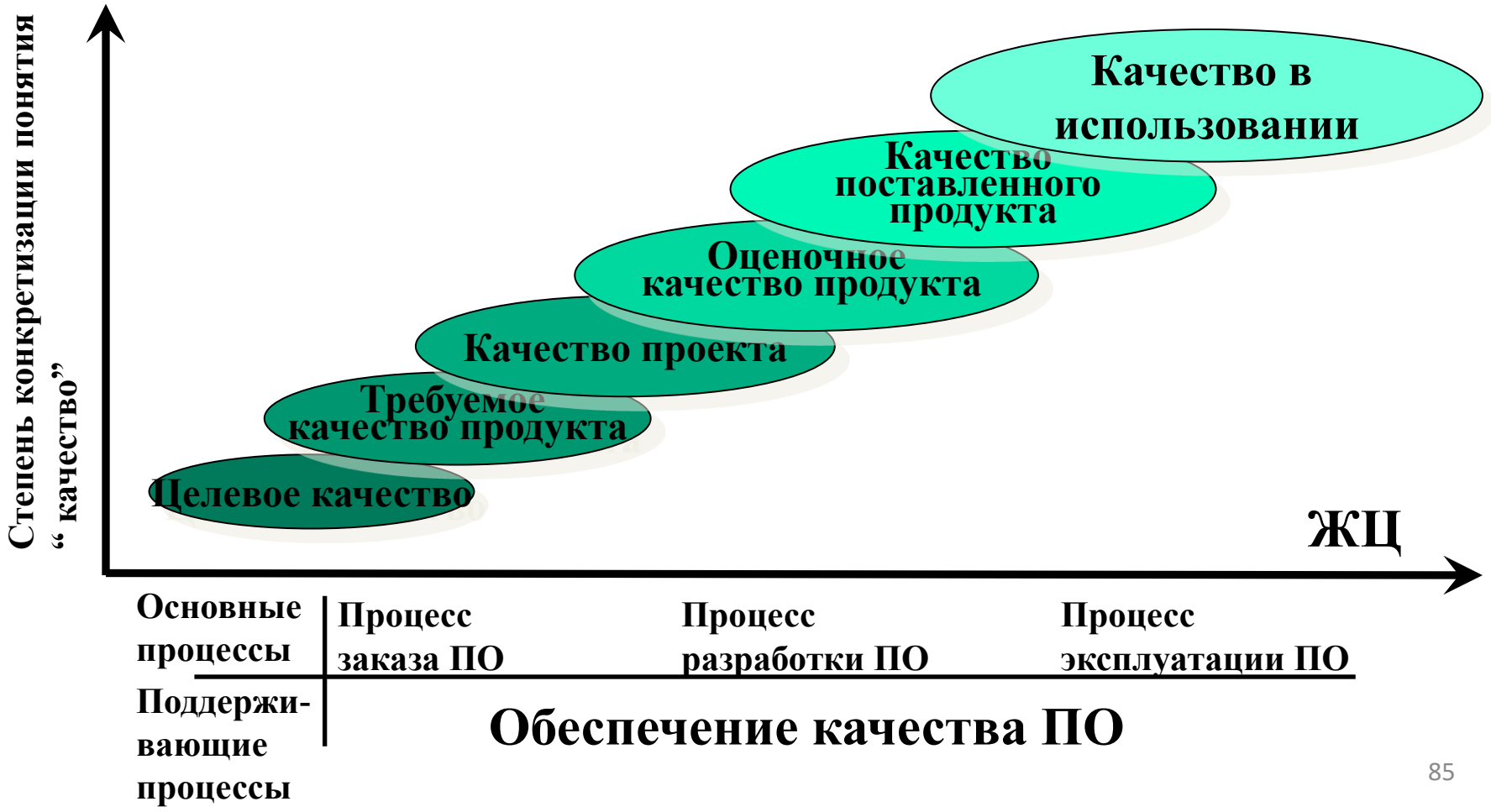
Метрики качества используются при оценке степени тестируемости с помощью данных (безотказная работа, выполнимость функций, удобство применения интерфейсов пользователей, БД и т.п.) после проведения испытаний ПО на множестве тестов.

Наработка на отказ как атрибут надежности определяет среднее время между появлением угроз, нарушающих безопасность, и обеспечивает трудноизмеримую оценку ущерба, которая наносится соответствующими угрозами. Очень часто оценка программы проводится по числу строк. При сопоставлении двух программ, реализующих одну прикладную задачу, предпочтение отдается короткой программе, так как её создает более квалифицированный персонал и в ней меньше скрытых ошибок и легче модифицировать. Длину программы можно использовать в качестве вспомогательного свойства для сравнения программ с учетом одинаковой квалификации разработчиков, единого стиля разработки и общей среды.

Если в требованиях к ПО было указано получить несколько показателей, то просчитанный после сбора данных показатель умножается на соответствующий весовой коэффициент, а затем суммируются все показатели для получения комплексной оценки уровня качества ПО.

В конечном итоге результат оценки качества является критерием эффективности и целесообразности применения методов проектирования, инструментальных средств и методик оценивания результатов создания программного продукта на стадиях ЖЦ.

КАЧЕСТВО И ЖИЗНЕННЫЙ ЦИКЛ (ИСО/МЭК 9126-1)



КАЧЕСТВО И ЖИЗНЕННЫЙ ЦИКЛ. ПРОДОЛЖЕНИЕ

Целевое Качество – Goal Quality (ЦК) означает необходимое и достаточное качество, которое отражает реальные потребности пользователя.

Требуемое Качество Продукта – Required Product Quality (ТКП) - это качество, фактически установленное в спецификации требований к качеству.

Качество Проекта - Design Quality (КП) – это качество, представленное в основных частях или основе проекта ПО, например, в архитектуре ПО, структуре программы и стратегии проектирования интерфейса пользователя.

Оценочное (или прогнозируемое) качество продукта – Estimated (or Predicted) Product Quality (ОКП) – это качество, оцененное или предсказанное для конечного потребителя на каждой стадии разработки и базирующееся на КП.

Качество поставленного продукта – Delivered Product Quality (КПП) - это качество поставленного продукта, обычно прошедшего испытания в смоделированной среде с имитированными данными.

Качество в использовании Quality in Use (КВИ) – это качество системы, содержащей ПО, которое воспринимается пользователями, и оно измеряется скорее в терминах результата использования ПО, чем свойств самого ПО.

ДОБРОТНОСТЬ ПО

Понятие добротности программы введено Игорем Васильевичем Поттосиным в 1997 году

И.В.Поттосин. О добротности программ // Системная информатика: Сб. науч. тр. – Новосибирск: Наука. Сибирское отделение РАН, 1998. – Вып. 6: Проблемы архитектуры, анализа и разработки программных систем. – с. 90 – 122.

«Добротность программы никак не связана со степенью уверенности в ее корректности – будем предполагать, что корректность достаточно убедительна, иначе не только о хорошей программе, но и о программе вообще не стоит говорить». Предполагается считать добротную программу подмножеством надежных программ.

- Комплексная мера технической реализации программы.
- Критерии добротности:
  - Количественные, основанные на некоторой количественной характеристике (или наборе характеристик), сопоставленной программе;
  - Генетические (определяются дисциплиной создания программы);
  - Структурные, отвечающие тому, насколько хорошо организована *программа как текст*;
  - Прагматические, которые возникают из сопоставления программы и ее цели – насколько эта цель может быть формально усмотрена в тексте.

ДОПОЛНИТЕЛЬНЫЕ СЛАЙДЫ

РУКОВОДСТВОМ ПО ПРОГРАММНОЙ ИНЖЕНЕРИИ SWEBOK

Методическим руководством по программной инженерии является появившийся в 2004 г. документ SWEBOK, полное английское название которого - Guide to the Software Engineering Body of Knowledge (SWEBOK), что можно перевести как "Руководство к методическому справочнику по программной инженерии" (IEEE Computer Society, 2004).

Книга представляет собой систематизированное и структурированное изложение основных понятий, связанных с разработкой ПО, объединяющее определения, концепции, методы из множества источников, среди которых стандарты занимают одно из центральных мест. Сам по себе SWEBOK не является ни стандартом, ни методикой, ни моделью. Это просто развернутый справочник, который состоит из десяти разделов, соответствующих десяти основным видам деятельности при разработке ПО. В каждом из разделов даются ссылки на стандарты, полностью или частично описывающие соответствующие процессы.

Одно из приложений к руководству (Приложение С) целиком посвящено стандартам, в котором показывается степень покрытия стандартами (их приведено свыше пятидесяти) десяти разделов SWEBOK.

Руководство дает достаточно полное представление о том, какое место занимают процессные модели в программной инженерии и как они развиваются. Очевидный вывод, который следует из анализа SWEBOK, состоит в том, что модели процессов, представленные разными стандартами, хотя и являются отчасти взаимодополняющими, не складываются в единую непротиворечивую модель. Как следствие, выбор подходящей процессной модели для внедрения на практике оказывается неизбежно субъективным, и для обоснования его приходится привлекать дополнительные соображения. В этой ситуации SWEBOK может служить хорошей отправной точкой для углубленного анализа ситуации и принятия решения о выборе процессной модели.

Руководство доступно для просмотра на https://www.computer.org/education/bodies-of-knowledge/software-engineering. Текущим релизом SWEBOK является версия 3.0, вышедшая в 2013г. и охватывающая пятнадцать областей программной инженерии.