

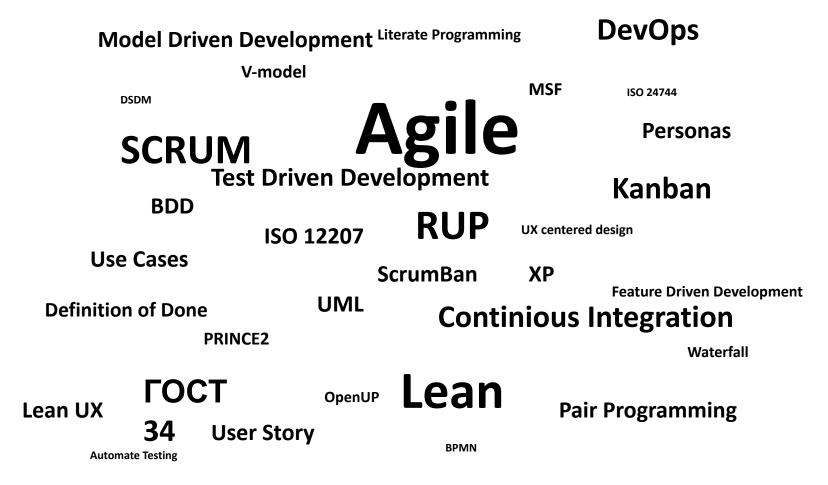


OMG! Essence: единая теория программной инженерии?

ЮРИЙ КУПРИЯНОВ SECON'2016

22/04/2016

В мире много методов приемов практик идеологий разработки ПО





Ивар Якобсон

UML, RUP, аспектноориентированное программирование



Бертран Мейер

Eiffel, **ООП**, контрактное программирование



Ричард Солей омб, **имL**, соява, мра



Software Engineering Method And Theory

"Программная инженерия

сегодня серьезно страдает от незрелых практик.

Основные проблемы:

- •Погоня за модой.
- •Отсутствие прочной, признанной теоретической базы.
- •Огромное число методов и их вариаций, различия которых искусственно преувеличены.
- •Отсутствие надежной экспериментальной оценки и проверки.
- •**Разрыв между** индустриальной **практикой и** академическими исследованиями.

Мы поддерживаем процесс нового основания программной инженерии – на твердой теории, проверенных принципах и лучших практиках, которые:

- **Включают** в себя **ядро** согласованных элементов, расширяемых для конкретных применений.
- Решают как технологические, так и человеческие вопросы.
- Поддержаны индустрией, наукой и пользователями.
- Поддерживают расширение в условиях меняющихся требований и технологий.



Pekka Abrahamsson

Подписанты

призыва

ctor Basili lean Bézivin

Robert V. Binder Dines Bjorner Barry Boehm Alan W. Brown

Larry Constantine Steve Cook Bill Curtis Donald Firesmith

Erich Gamma Carlo Ghezzi Tom Gilb Robert L. Glass

Ellen Gottesdiener Martin Griss Sam Guckenheimer David Harel

Brian Henderson-Sellers Ivar Jacobson Capers Jones Philippe Kruchten

Dean Leffingwell Harold "Bud" Lawson Robert Martin Bertrand Meyer

Paul Nielsen James Odell Meilir Page-Jones Dieter Rombach

Ken Schwaber Alec Sharp Richard Soley Ian Sommerville

Andrey Terekhov Fuqing Yang Edward Yourdon



Подписанты

THITLIPS

ABB

Martin Naedele Switzerland Ericsson

Anders Caspár Sweden Fujitsu UK

Duncan Tait

Huawei

Yanlin Hao China

IBM

Walker E. Royce USA Microsoft

Julio Fernandez-Gayoso Spain Munich RE

Burkhard Perkens-Golomb Germany SAAB

Göran Backlund Sweden

SICS

Jakob Axelsson Sweden SINTEF

Arne Berre Norway Software Engineering Institute (SEI)

Paul Nielsen USA **Tata Consulting Services**

Sumeet Malhotra

USA

Telecom Italia

Marco Forneris Italy Toronto, Ontario

Dave W. Wallace Canada WellPoint

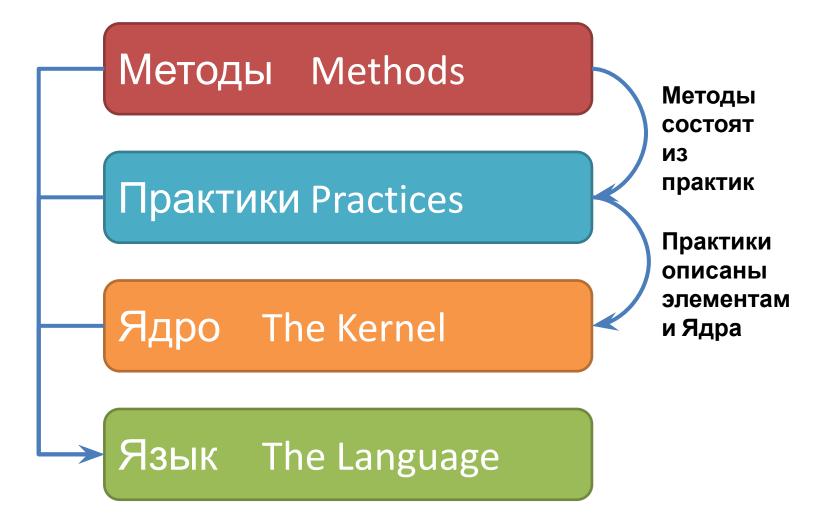
Andrew J. Lang USA







Методы, Практики и Ядро определен ы в терминах Языка





Области

интереса

Потребитель Customer

Решение Solution

Деятельность Endeavor



Внутри областей

интереса

Альфы *ALPHA*

Abstract-Level Progress Health Attribute



Поле деятельности Activity Space

Компетенции Competence



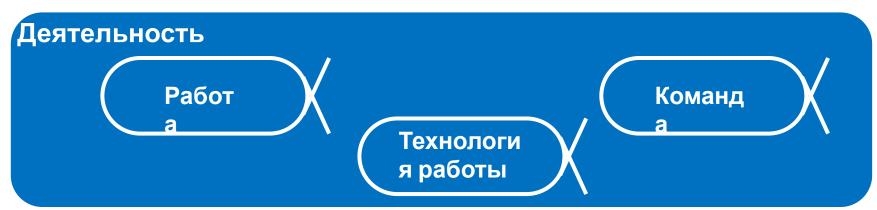














Связи





Поле

Потребитель

<u>деятельности</u>

Исследовать возможность

Понять нужды Убедиться в удовлетворени и

Изучать использован иесистемы

Решение

Понять требов а-ния

Спроектироват ь систему

Реализовать систему Протестироват ь систему

Развернуть систему

Обслуживать систему

Деятельность

Приготовить ся выполнять работу

Координировать дела

Поддерживат ь команду Отслеживат **ь** прогресс

Прекратит ь работу



Компетенци

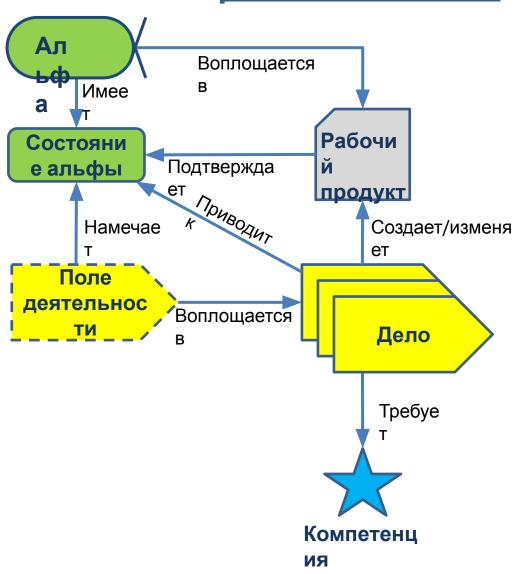








<u>Как это</u> работает:





Состояние альфы

Стейкхолдеры выявлены.

Механизмы вовлечения стейкхолдеров согласованы, представители **Бредгавиц**ели стейкхолдеров вовлечены в работу и выполняют свои обязанности. Представители стейкхолдеров находятся в согласии.

Минимальные ожидания представителей стейкхолдеров удовлетворены.

Система удовлетворяет или превышает минимальные ожидания стейкхолдеров.





Выбрана архитектура, адресующая технические риски и удовлетворяющая организационным ограничениям.

Работающая версия системы готова для демонстрации соответствия архитектуры и возможности тестирования.

Система готова к использованию и демонстрирует заданные характеристики качества.

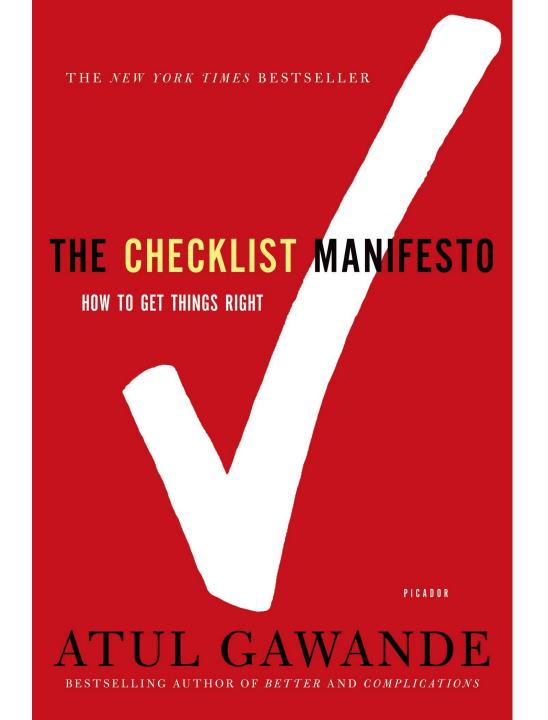
Система была принята к развертыванию и запуску.

Система используется в операционном окружении.

Система больше не поддерживается.



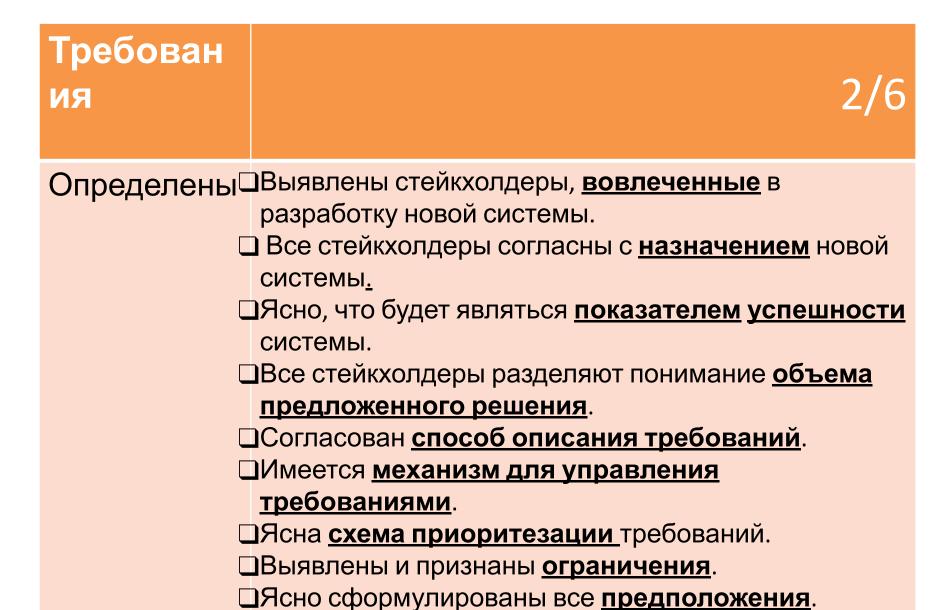




Чек-листы для состояний

Требован ия	1/6
	Стейкхолдеры согласны, что система должна быть создана. Выявлены стейкхолдеры, которые будут пользоваться системой. Выявлены стейкхолдеры, которые будут финансировать создание системы. Ясно, какую возможность будет использовать будущая система.

Чек-листы для состояний





Software System

Work

Team



- . The need for a new system is
- · Users are identified
- · Initial sponsors are identified

Requirements

Bounded

- . The purpose and extent of the system are agreed
- Success criteria are clear
- Mechanisms for handling requirements are agreed
- Constraints and assumptions

2/6

Requirements

Coherent

- . The big picture is clear and shared by all involved
- · Important usage scenarios explained
- · Priorities are clear
- Conflicts are addressed Impact is understood

3/6

Requirements

Acceptable

- · Requirements describe a solution acceptable to the stakeholders
- The rate of change to agreed requirements is low
- Value is clear

4/6

Requirements

Addressed

- Enough requirements are implemented for the system to be acceptable
- Stakeholders agree the system is worth making operational

5/6



6/6

Software System

Architecture

- Selected · Architecture selected that
- address key technical risks · Criteria for selecting architecture
- agreed · Platforms, technologies,
- languages selected Buy, build, reuse decisions

1/6

Software System

Demonstrable

- Key architecture
- characteristics demonstrated · Relevant stakeholders agree architecture is appropriate
- Critical interface and system configurations exercised

2/6

Software System

- Usable System is usable and has
- desired quality characteristics
- System can be operated by users
- Functionality and performance have been tested and accented
- Defect levels acceptable
- Release content known

Work

Software System

Ready

- · User documentation available
- · Stakeholder representatives accept system
- Stakeholder representatives want to make system operational

4/6

Software System

Operational

- · System in use in operational
- environment System available to intended
- users At least one example of system
- is fully operational System supported to agreed service levels



Retired

- · System no longer supported
- · Updates to system will no longer be produced
- System has been replaced or discontinued.

Work Initiated

- · Work initiator known
- Sponsorship and funding model clear
- · Priority of work clear

1/6

Work

Prepared

- . Cost & effort estimated
- · Funding and resources to start work in place
- Acceptance criteria understood · Governance procedures agreed
- · Risk exposure understood
- Dependencies clear

2/6

Work progress is monitored

· Development work has started · Work broken down into actionable items with clear definition of done Team members are accepting

Started

and progressing work items 3/6

Work

Under Control

- Work going well, risks being
- managed Unplanned work & re-work under control
- Work items completed within estimates

Measures tracked

4/6

Work

Concluded

- · Work to produce results have been finished
- · Work results are being achieved
- The client has accepted the resulting software system

5/6

Work

- Closed
- · All remaining housekeeping tasks completed, and work
- officially closed · Everything has been archived

· Lessons learned and metrics made available

6/6

Team

Seeded

- Team's mission is clear
- Team knows how to grow to
- achieve mission Required competencies are
- Team size is determined

1/5

Team

- Formed Team has enough resources to
- start the mission Team organization & individual
- responsibilities understood · Members know how to perform

2/5

Team

Collaborating

- · Members working as one unit
- Communication is open and
- honest Members focused on team
- mission Success of team ahead of personal objectives

3/5

Team

Performing

- Team working efficiently and
- effectively · Adapts to changing context
- · Produce high quality output · Minimal backtracking and re-

· Waste continually eliminated 4/5

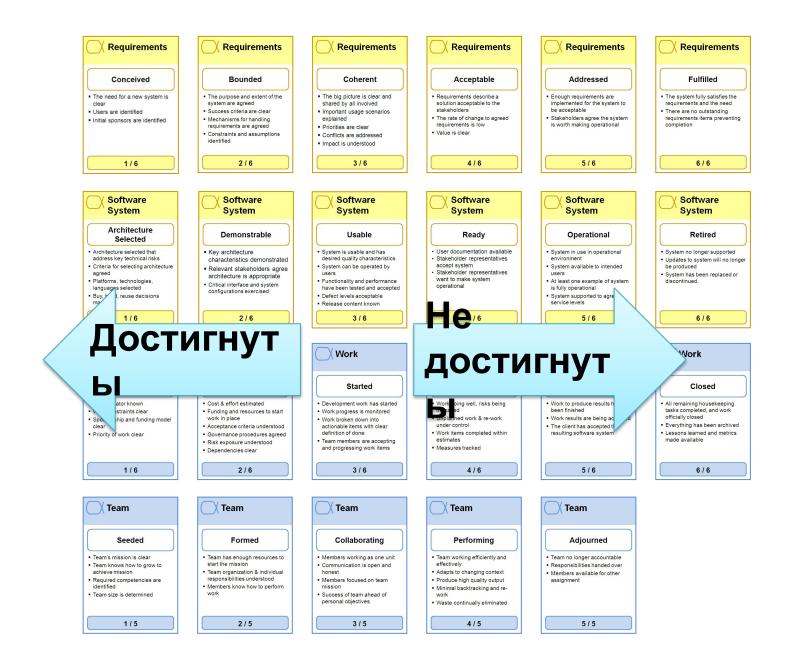
Team

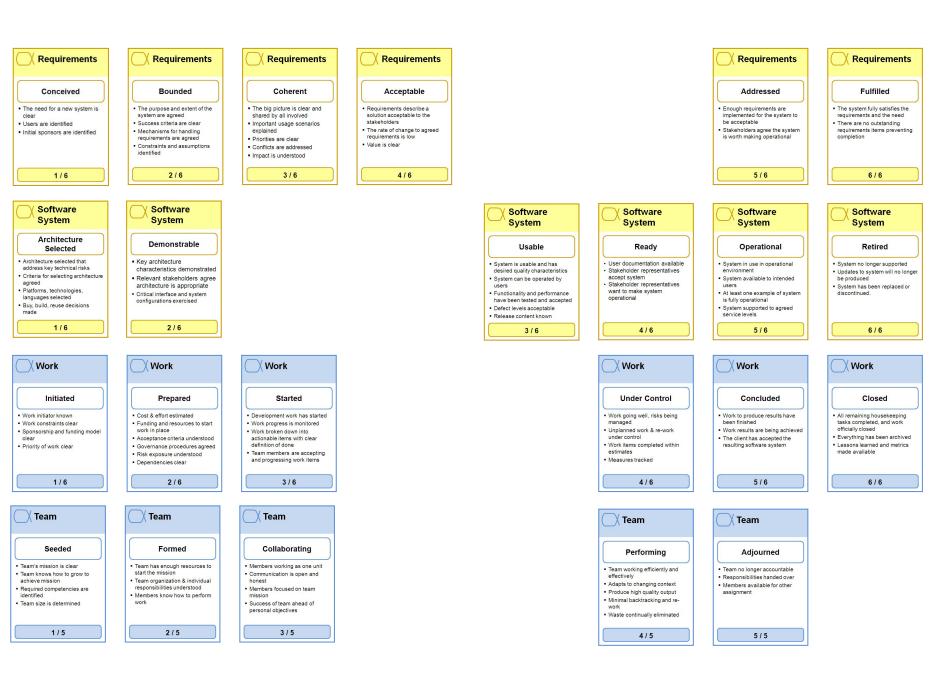
Adjourned

- · Team no longer accountable
- Responsibilities handed over

 Members available for other assignment

5/5







<u>Что дает Ядро?</u>

- Фокусирует внимание на состоянии альф;
- Разделяет роли по областям интереса;
- Предоставляет высокоуровневые последовательности состояний ключевых *альф* и чек-листы для их диагностики;
- Задает базовые элементы для описания практик и методов.



<u>Для чего использовать</u> Ядро?

- Для **оценки состояния** проекта (без привязки к конкретной методологии);
- Для планирования;
- Для сравнения двух методологий и проектирования процессов изменения;
- Для обучения;
- Для масштабирования;
- Для оптимизации методов;
- Для подбора людей.



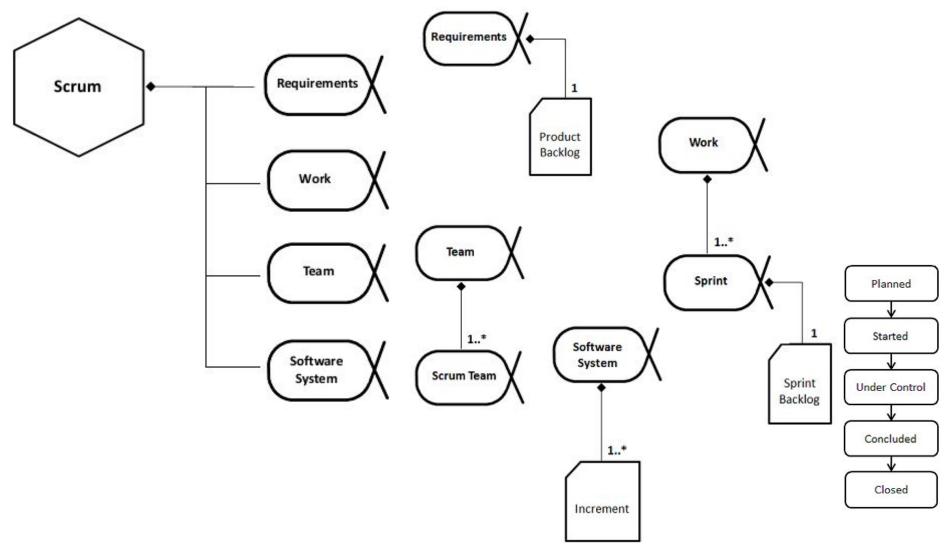
<u>Немедленная</u> польза

- 1. Применять чек-листы.
- 2. Раскладывать пасьянсы/покер.
- Включать пункты из чек-листов сразу в договоры и проектные документы.



<u>Дальнейшие</u>

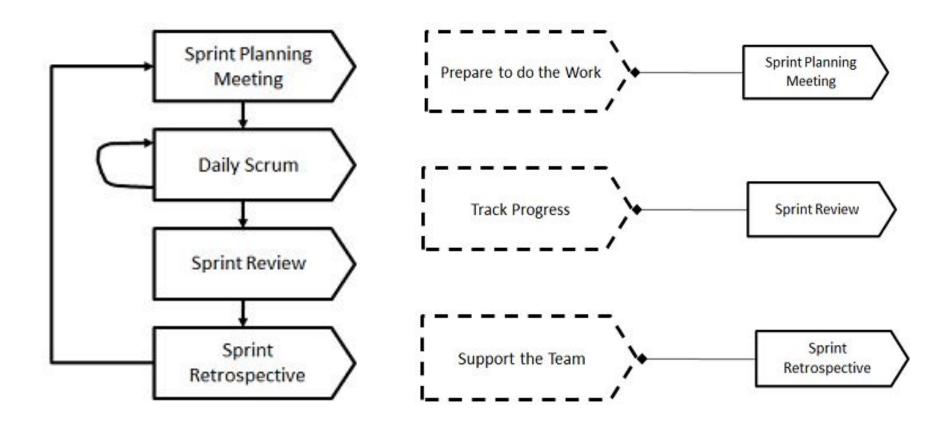
исследования Моделирование практик в терминах ядра:





<u>Дальнейшие</u>

исследования Моделирование практик в терминах ядра:





<u>Дальнейшие</u>

Сборка методов из практик:

		_	_
Оо Метод разработки платформы	Метод интеграции приложений	Разработка мобильных приложений	
DevOps	DevOps	DevOps	Улучшени я
RUP Architecture Centric T3 по ГОСТ 34	Kanban Emerging Architecture Use Cases	Scrum Emerging Architecture Lean UX	Специфически е практики
Defect/Issue Tracking Git Flow	Defect/Issue Tracking Git Flow	Defect/Issue Tracking Git Flow	Общие практики
	Kernel (





<u>U</u>

Страница стандарта на сайте OMG:

http://www.omg.org/spec/Essence/

Глоссарий на русском: http://goo.gl/zfyzjt

Инициатива SEMAT: http://www.semat.org/

Карты Essence на английском:

https://www.ivarjacobson.com/alphastatecards

Спасибо за внимание!

Давайте пробовать!

yksi12@gmail.com

http://facebook.com/yksi12

skype: yury.kupriyanov

8-903-617-4283