IVAR JACOBSON INTERNATIONAL

Creating winning teams.

What you need for IoT: Smarter Methods

Ivar Jacobson

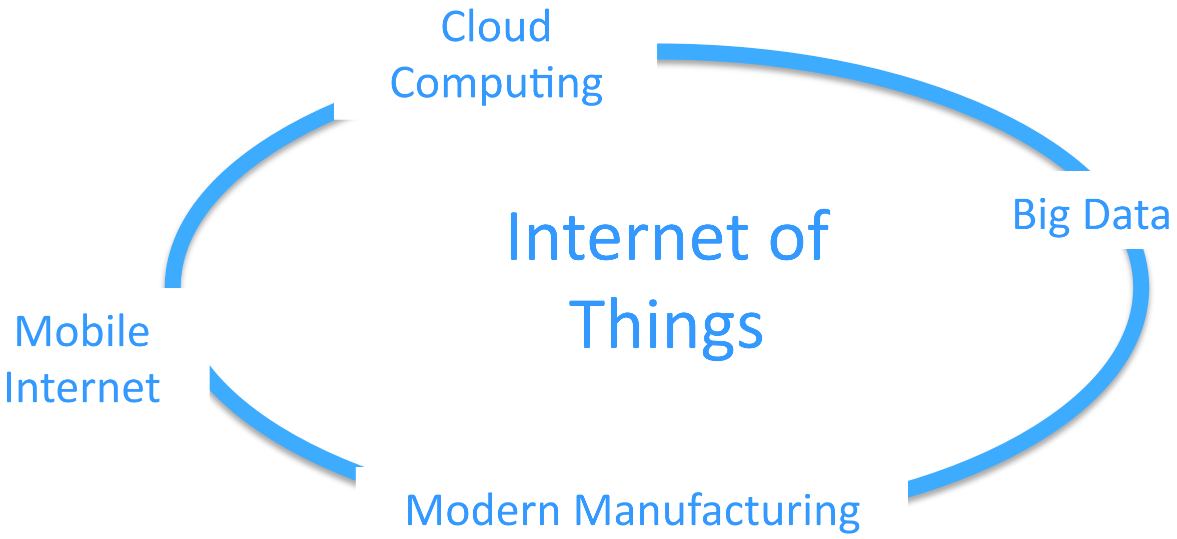
www.ivarjacobson.com

Agenda

1. IoT and Methods
2. Existing Methods puts you in Method Prisons
3. How to get out of your Method Prison?
4. Essentialization of Existing Methods
5. Essentialization of Ignite
6. What is the Value Proposition?
7. Next: It is Futurized
8. The Expectation Today

What’s so special about the Internet of Things?

- Internet of Things touches all kinds of products, services, functionality
- All levels of complexity
  - from very simple software running on basic sensors and other simple devices through to
  - the high-performance, highly reliable, highly governed, secure, resilient, scalable systems needed to process, analyze and respond to the vast amounts of data they produce and
  - everything else in between.



Internet of Things

Cloud Computing

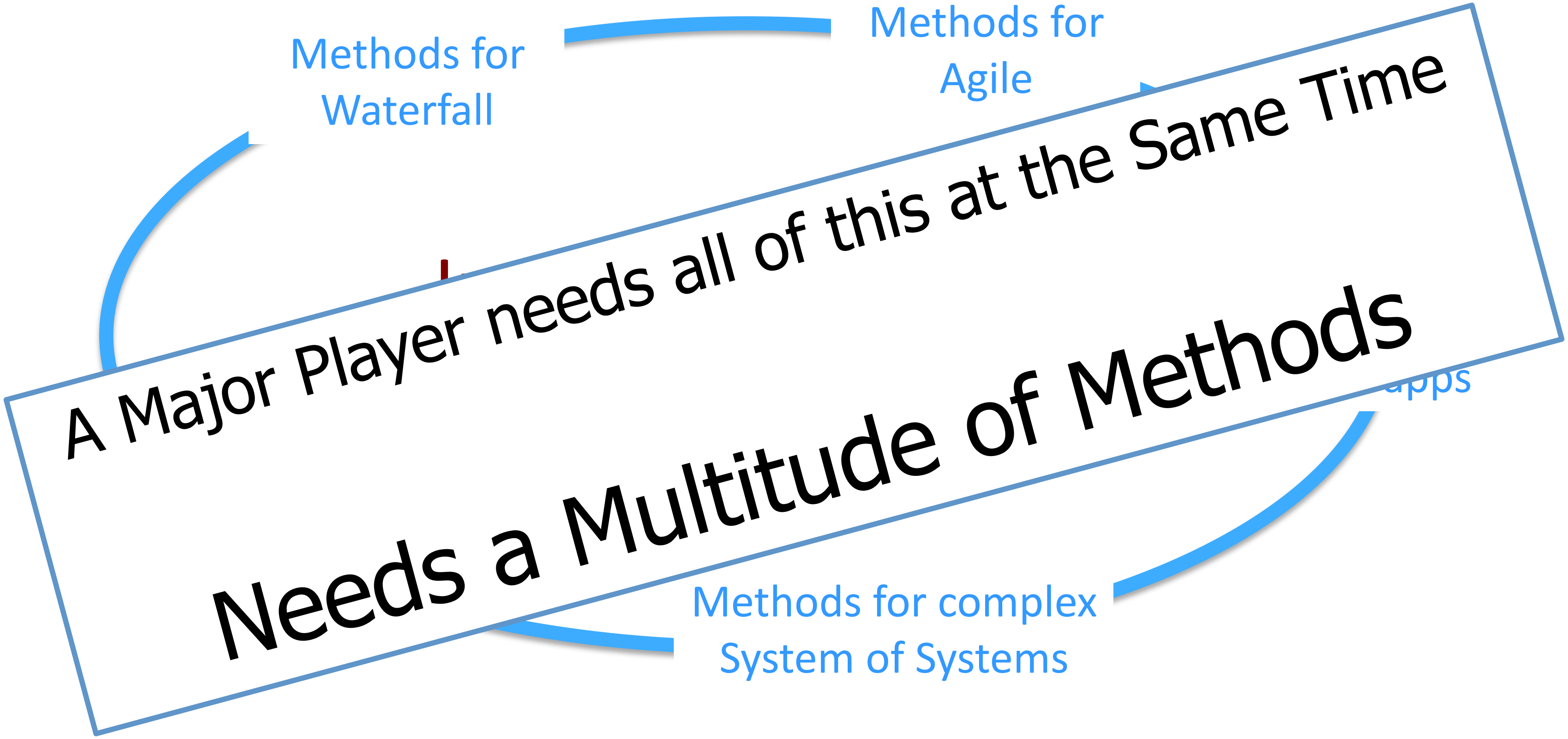
Big Data

Modern Manufacturing

Mobile Internet

The Internet of Things needs everything

- The Internet of Things doesn't lack methods.



Methods for Waterfall

Methods for Agile

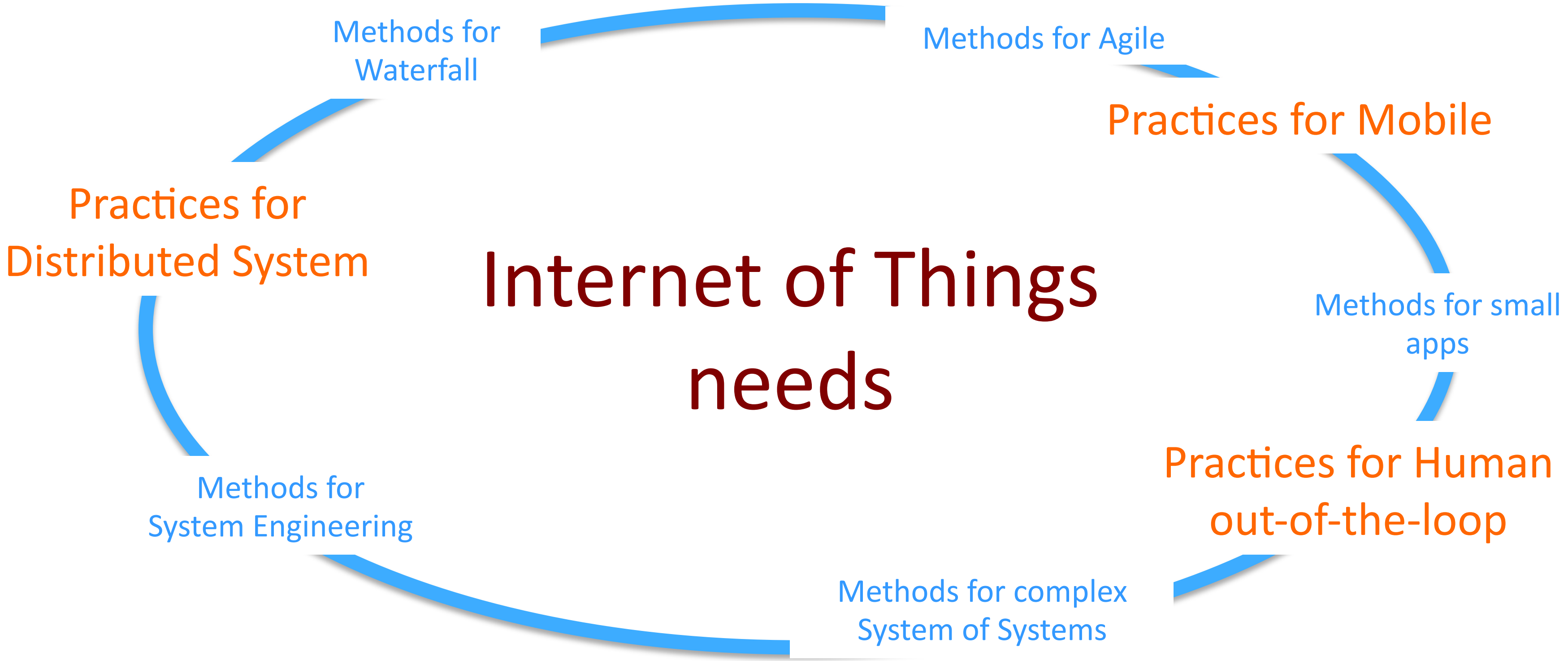
apps

Methods for complex System of Systems

A Major Player needs all of this at the Same Time

Needs a Multitude of Methods

And New Domain-Specific Practices are needed



Methods for Waterfall

Methods for Agile

Practices for Mobile

Methods for small apps

Practices for Human out-of-the-loop

Methods for complex System of Systems

Methods for System Engineering

Practices for Distributed System

Internet of Things needs

What aren’t needed are new management practices.

Summary of IoT needs

A single vendor needs a multitude of methods

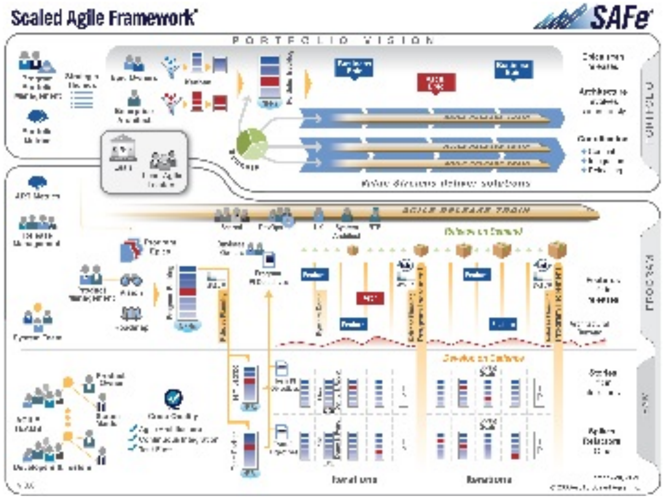
- Teams need to be able to select their own method from a library of practices
- Methods need a new user-experience – developers don't read books
- Methods need to focus on the essence – 5% of what an expert knows
- Methods need to guide in every day usage, not guide by reading a book
  - The method needs to help you monitor progress & health of the project

Agenda

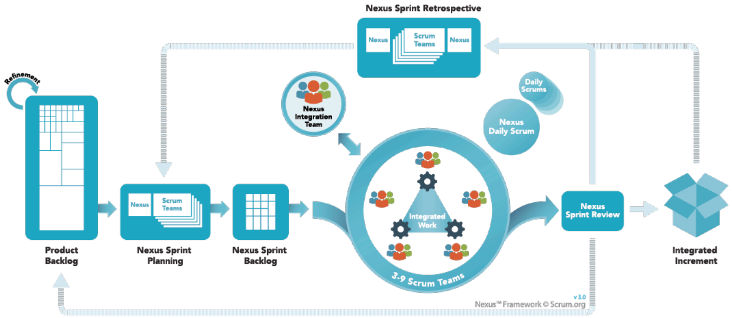
1. IoT and Methods
2. Existing Methods puts you in Method Prisons
3. How to get out of your Method Prison?
4. Essentialization of Existing Methods
5. Essentialization of Ignite
6. What is the Value Proposition?
7. Next: It is Futurized
8. The Expectation Today

Let’s look at Methods for Agile at Scale

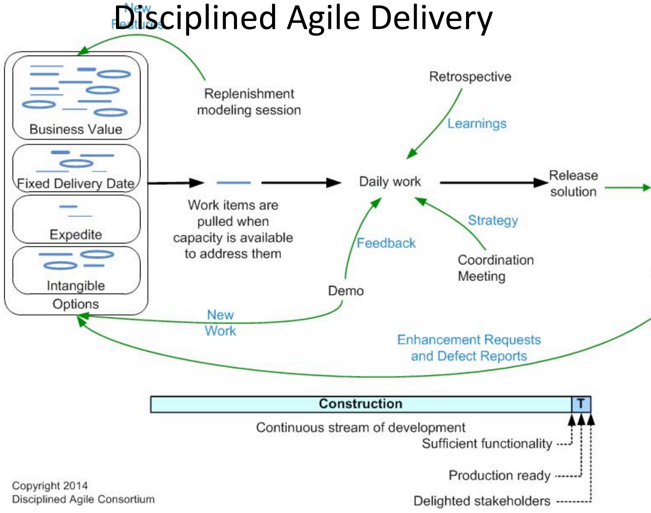
SAFe



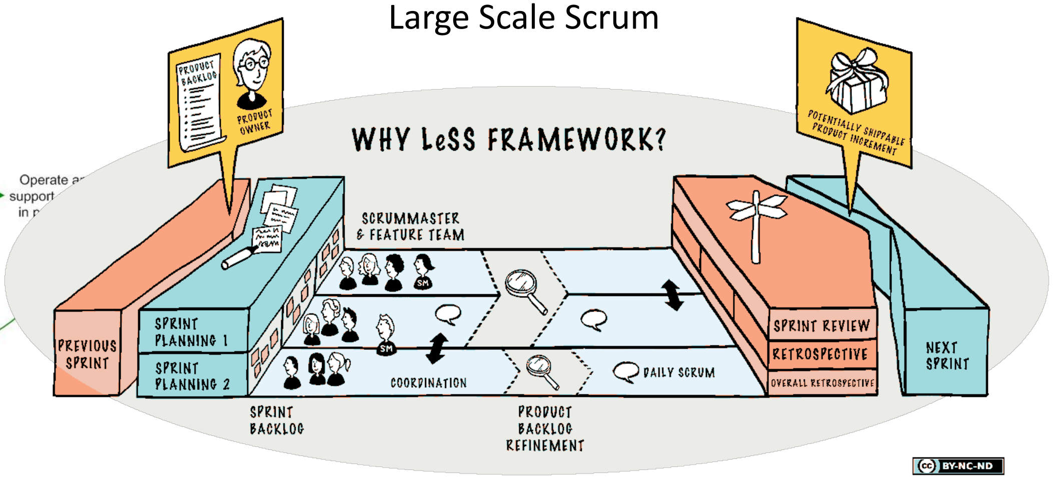
Scaled Professional Scrum



Disciplined Agile Delivery

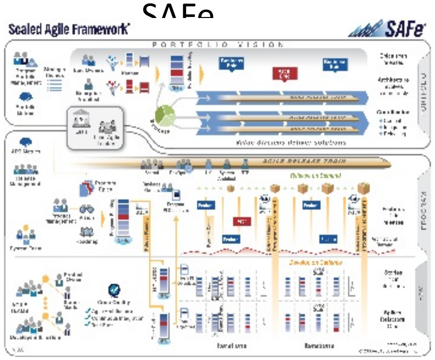


Large Scale Scrum

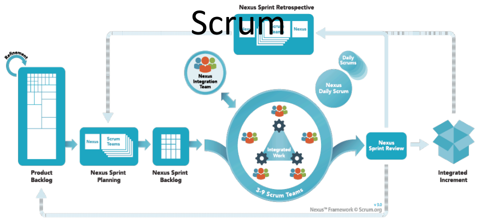


Let’s look at Methods for Agile at Scale

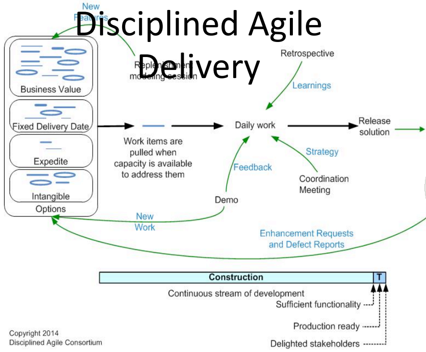
SAFe



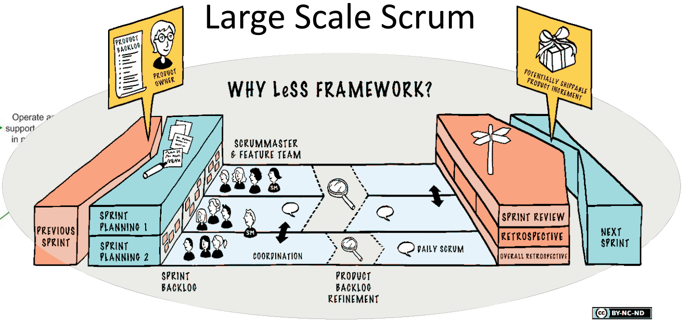
Scaled Professional Scrum



Disciplined Agile Delivery



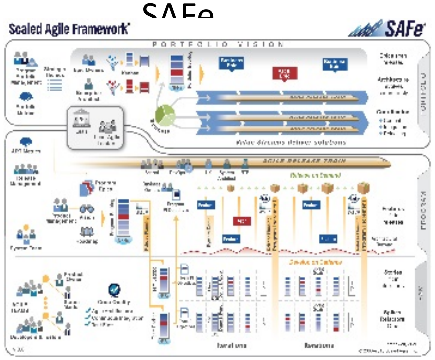
Large Scale Scrum



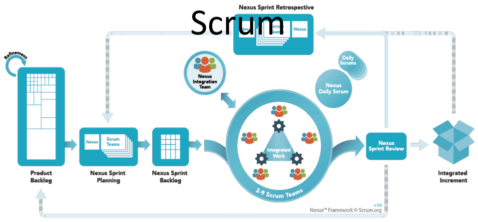
- They are all monolithic – non modular
- They have a lot in common (but you can’t easily see it)
- They all have unique own practices, but you cannot mix and match practices from them
- If you select one, you are in a “Method Prison” controlled by the guru of that Method
- And, there are many more other methods that also are monolit

Let’s look at Methods for Agile at Scale

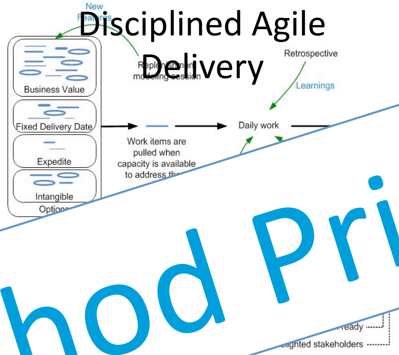
SAFe



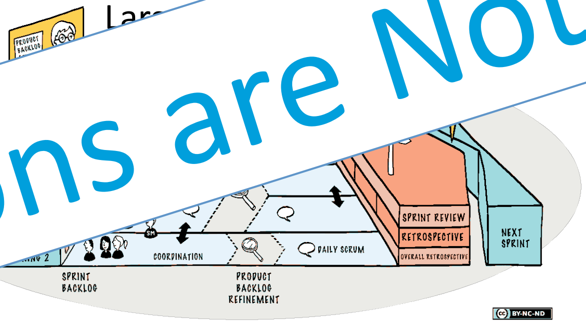
Scaled Professional Scrum



Disciplined Agile Delivery



Lar



Method Prisons are Not Smart

- monolithic – non modular
- They have a lot in common (but you can’t easily see it)
- They all have unique own practices, but you cannot mix and match practices from them
- If you select one, you are in a “Method Prison” controlled by the guru of that Method
- And, there are many more other methods that also are monolit

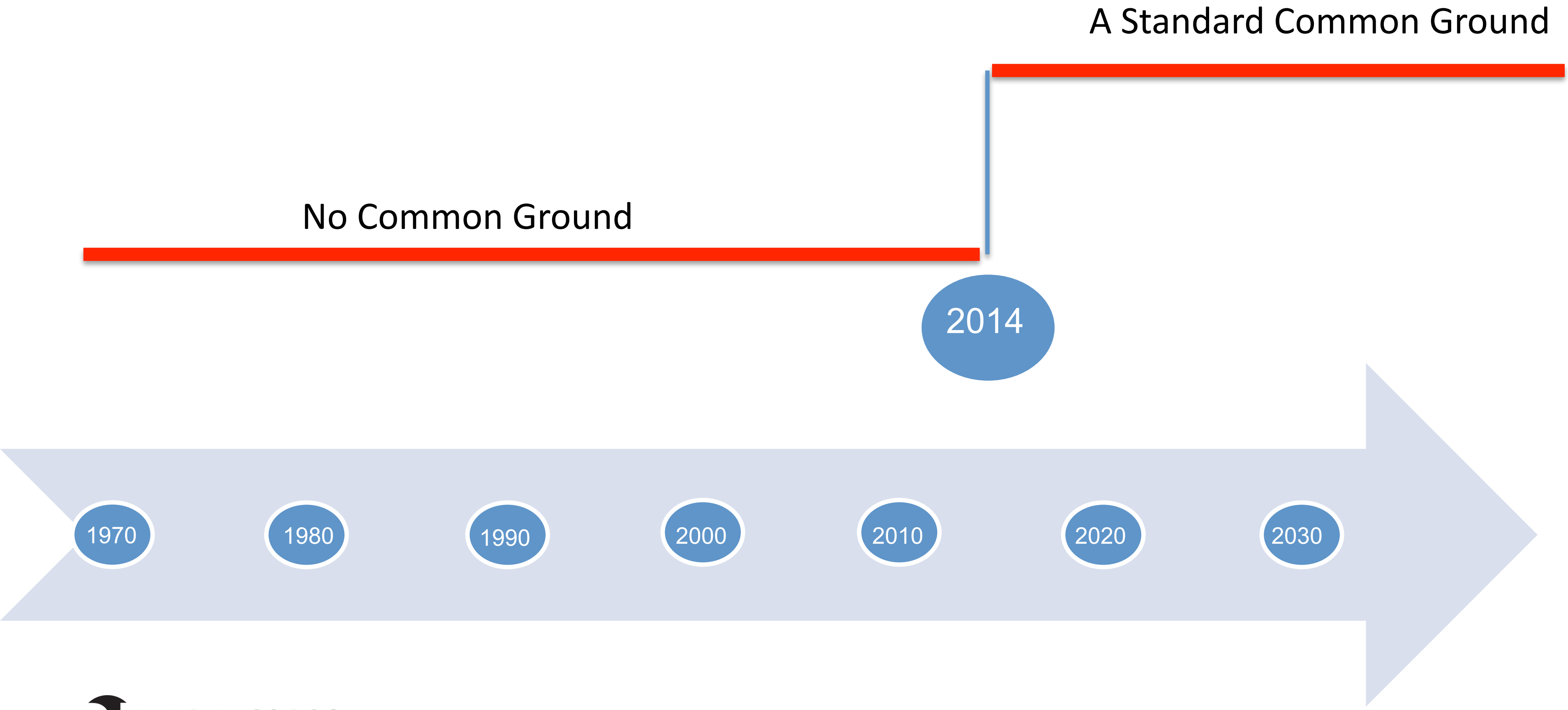
Agenda

1. IoT and Methods
2. Existing Methods puts you in Method Prisons
3. How to get out of your Method Prison?
4. Essentialization of Existing Methods
5. Essentialization of Ignite
6. What is the Value Proposition?
7. Next: It is Futurized
8. The Expectation Today

Start getting a Common Ground

What is a Common Ground?

- It includes elements that every method has, what every method produces, what you do always, etc.
- It is a starting point to understand software engineering



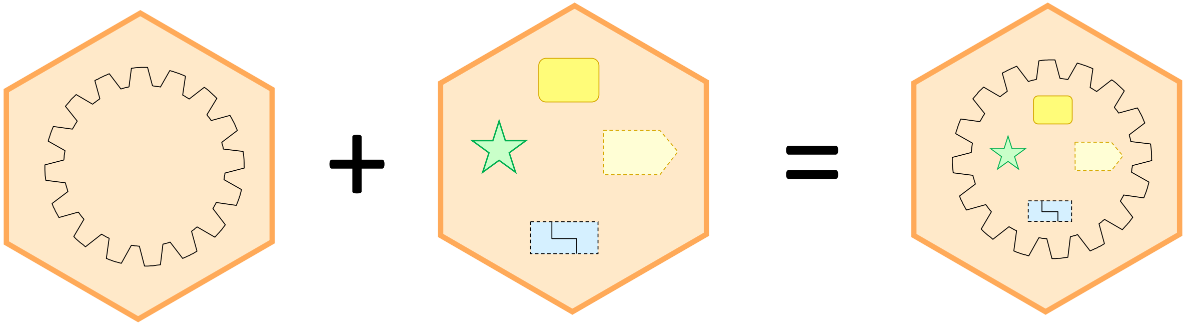
IVAR JACOBSON INTERNATIONAL

Creating winning teams.

There is a standard Common Ground: Essence

For the first time in the 60+ year history of software engineering, we have got a Common Ground

Kernel **Language** **Essence**



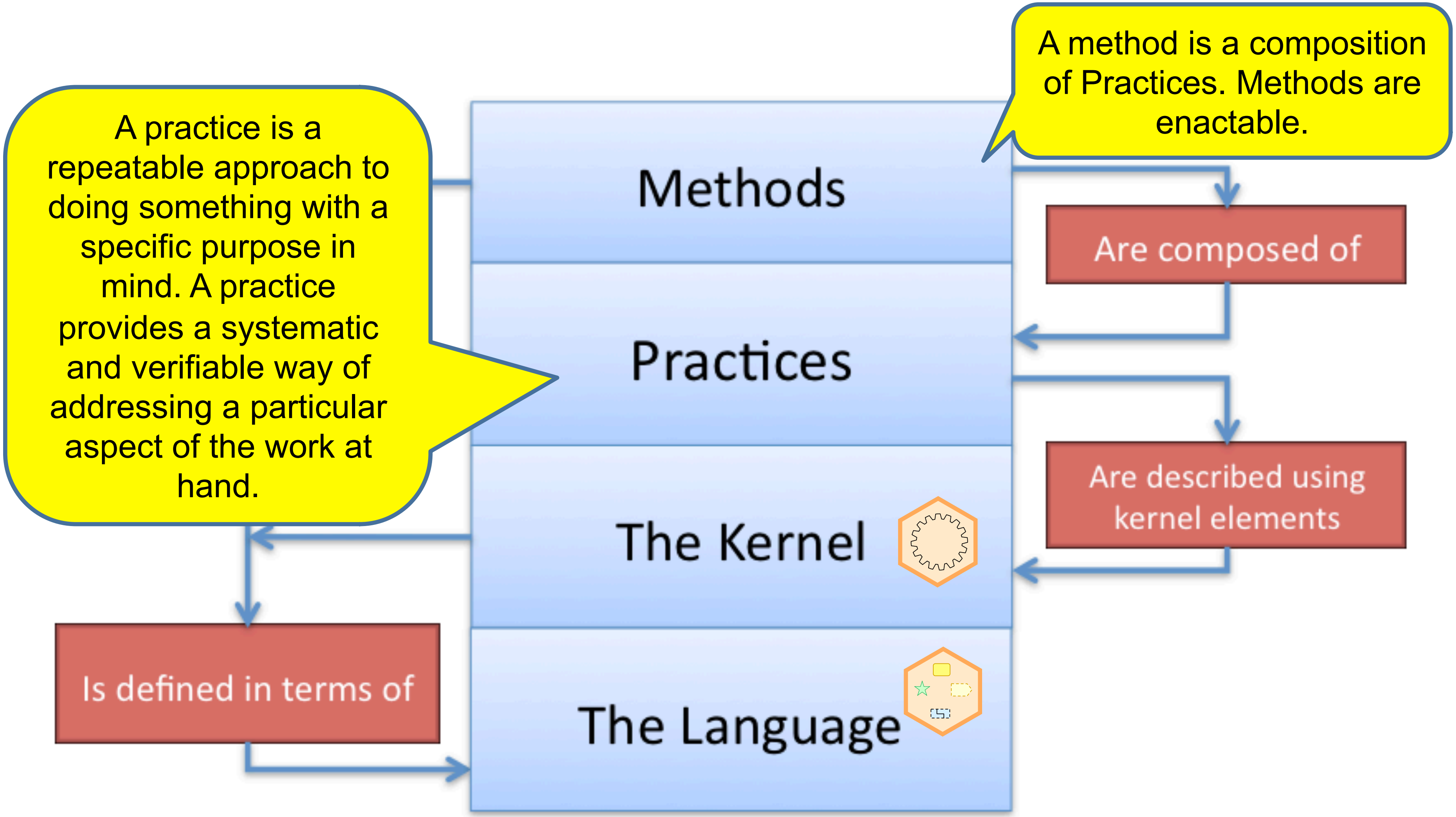
WE SET THE STANDARD®

- Essential Things to Work with
- Essential Things to Do
- The Essential Competencies you need

- Visual language
- Simple
- Intuitive

THE COMMON GROUND

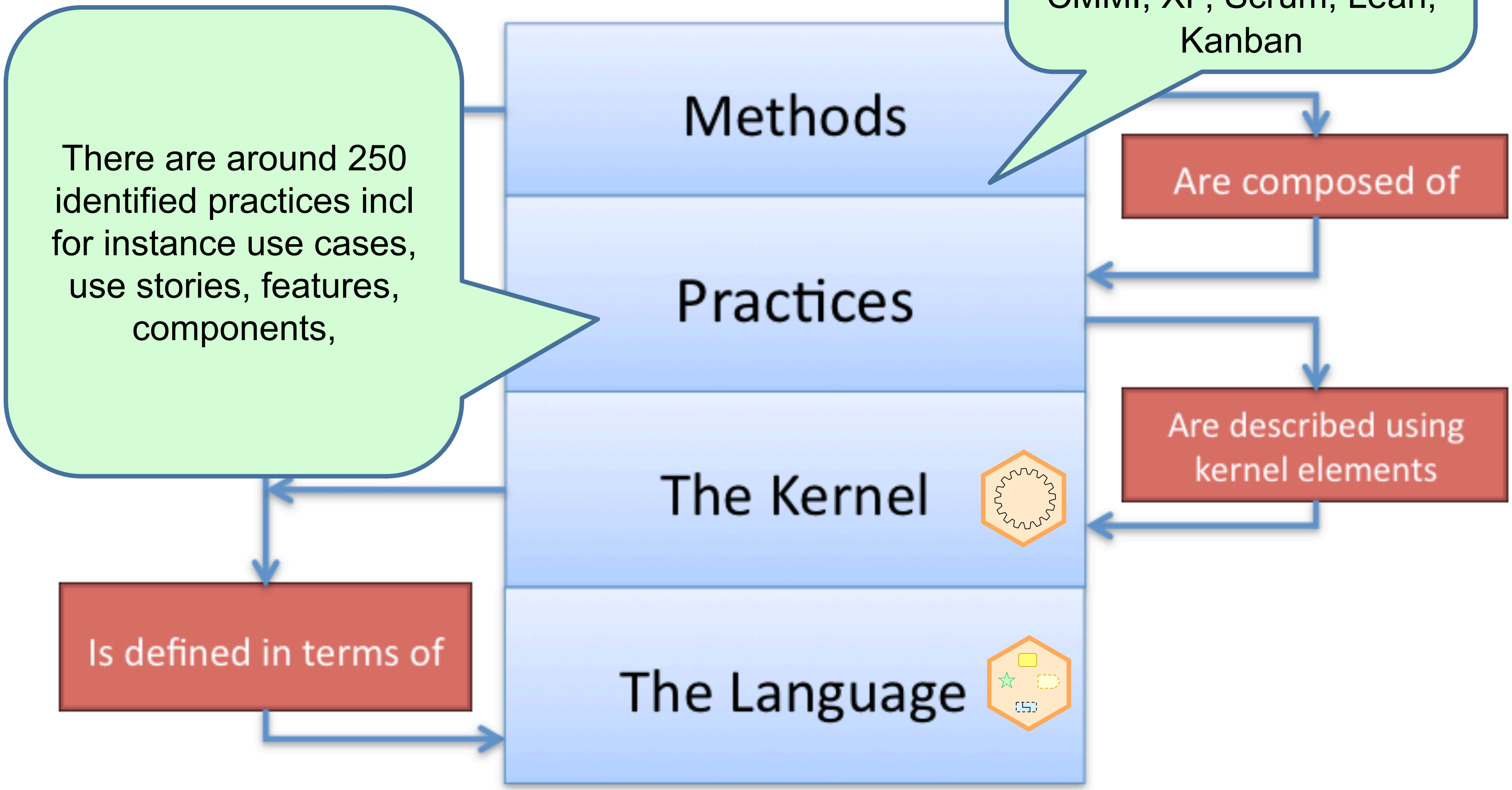
Essence is the common ground to build Practices and Methods upon


A method is a composition of Practices. Methods are enactable.
A practice is a repeatable approach to doing something with a specific purpose in mind. A practice provides a systematic and verifiable way of addressing a particular aspect of the work at hand.
Methods
Practices
The Kernel
The Language
Are composed of
Are described using kernel elements
Is defined in terms of

IVAR JACOBSON INTERNATIONAL

Creating winning teams.

The Method Architecture

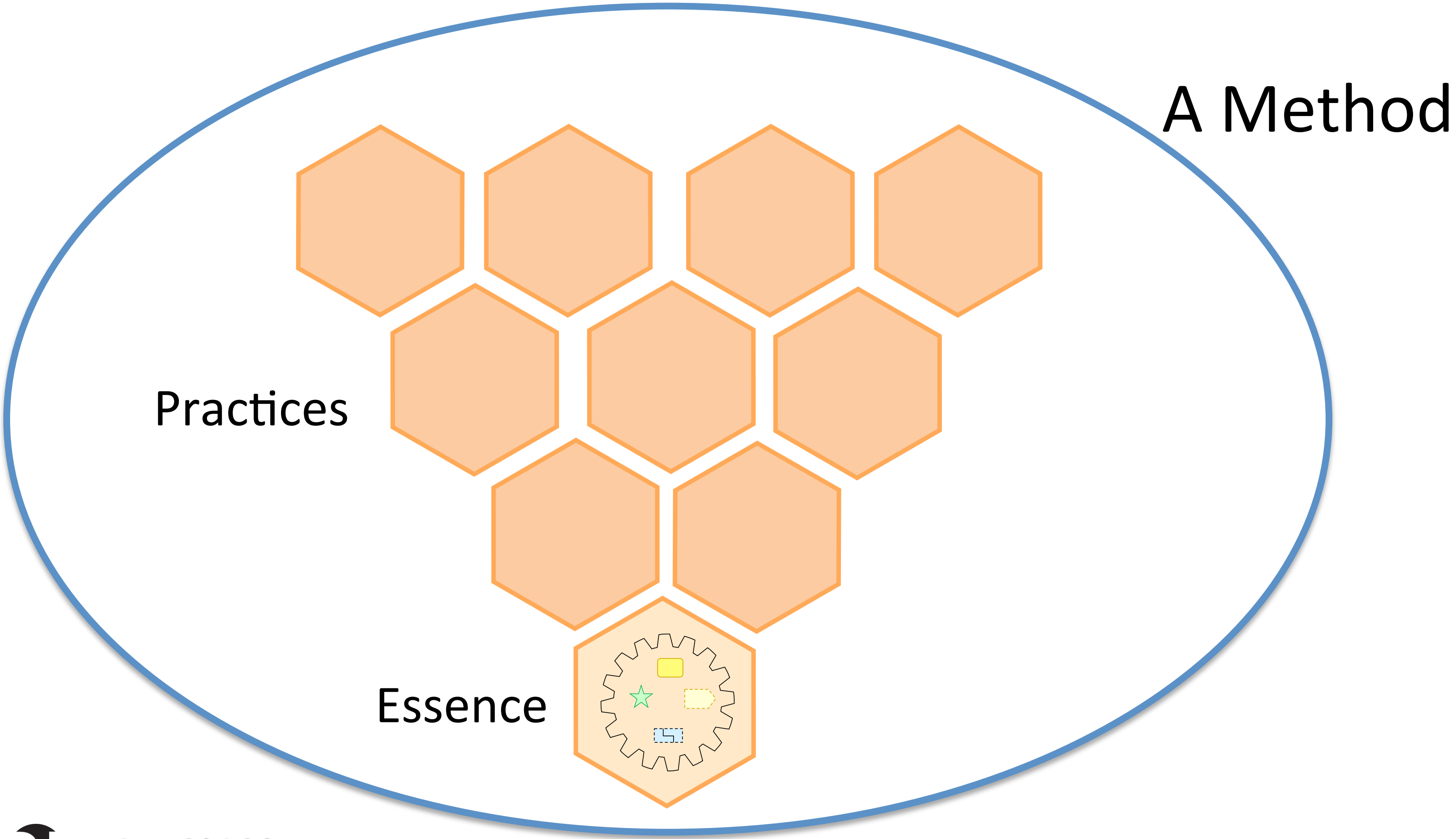


There are probably more than 100,000 methods incl. for instance SADT, Booch, OMT, RUP, CMMI, XP, Scrum, Lean, Kanban

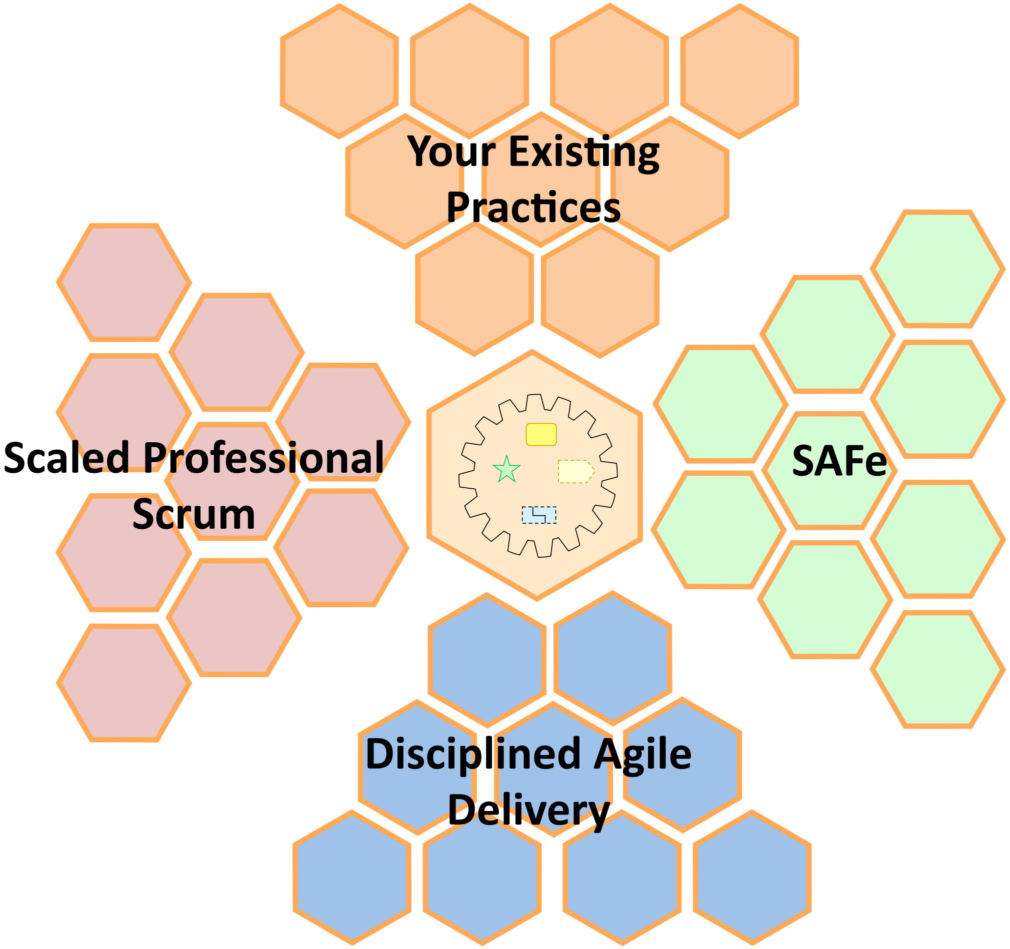
There are around 250 identified practices incl for instance use cases, use stories, features, components,

Then Add Practices on Top of Essence

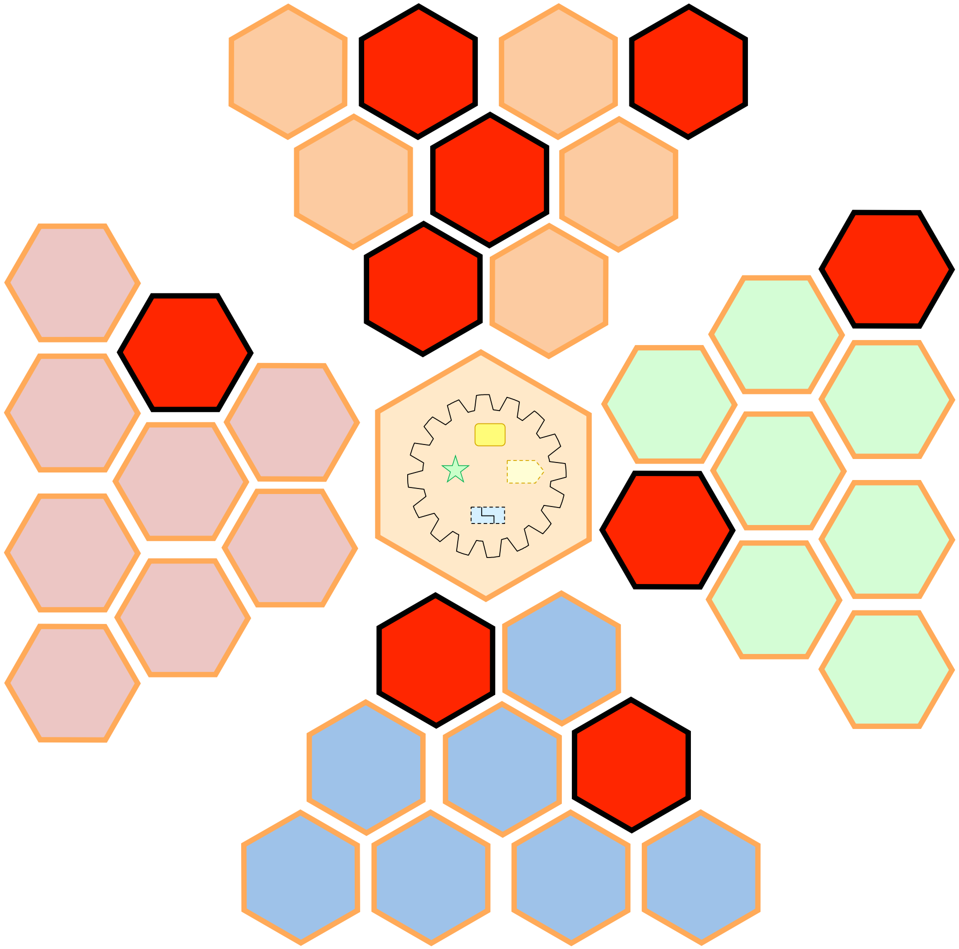
- Essence makes Methods Modular – not Monolithic



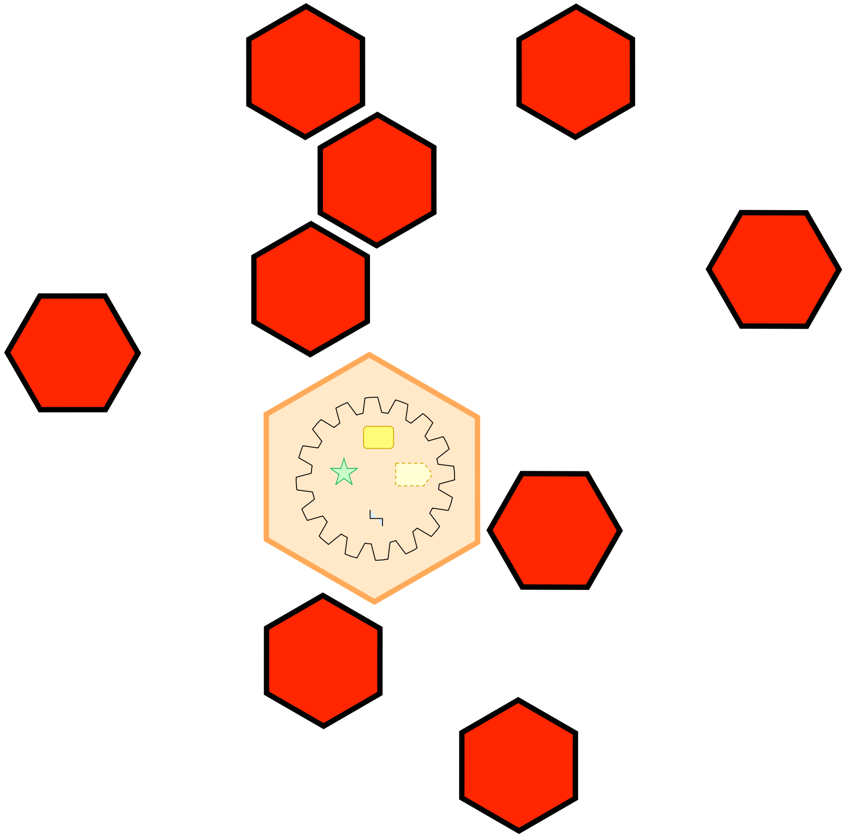
Imagine a Practice Library



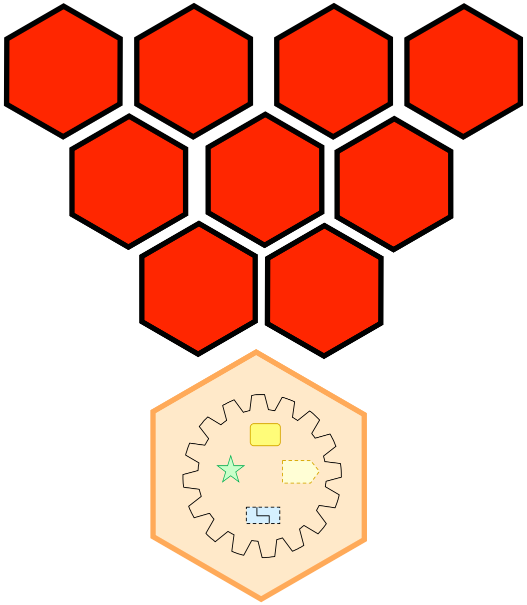
And you select the practices you like



And ignore the ones you don’t need



To create your own method



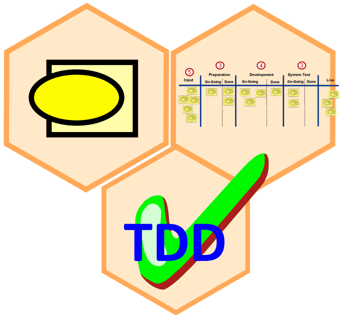
…to get flexibility

Mix and Match Practices to Empower your Teams

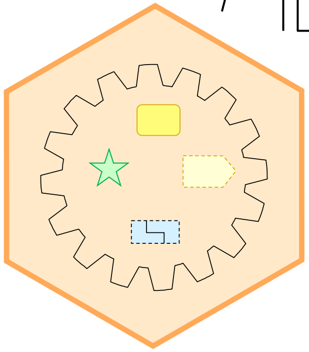
Team B

Use Case

Kanban



Test Driven Development

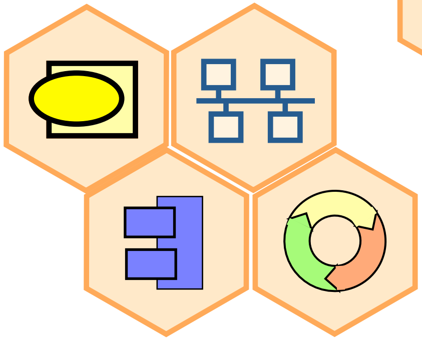


Team's build their way-of-working on top of pre-defined common ground and in-line with organizational principles and policies.

Team A

Use Case

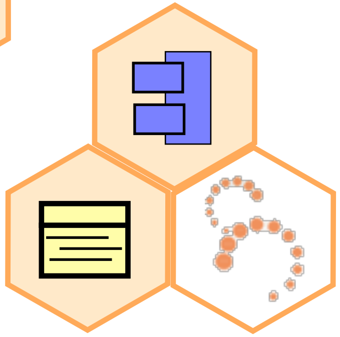
Architecture



Component

Iterative

Component



User Story

Scrum

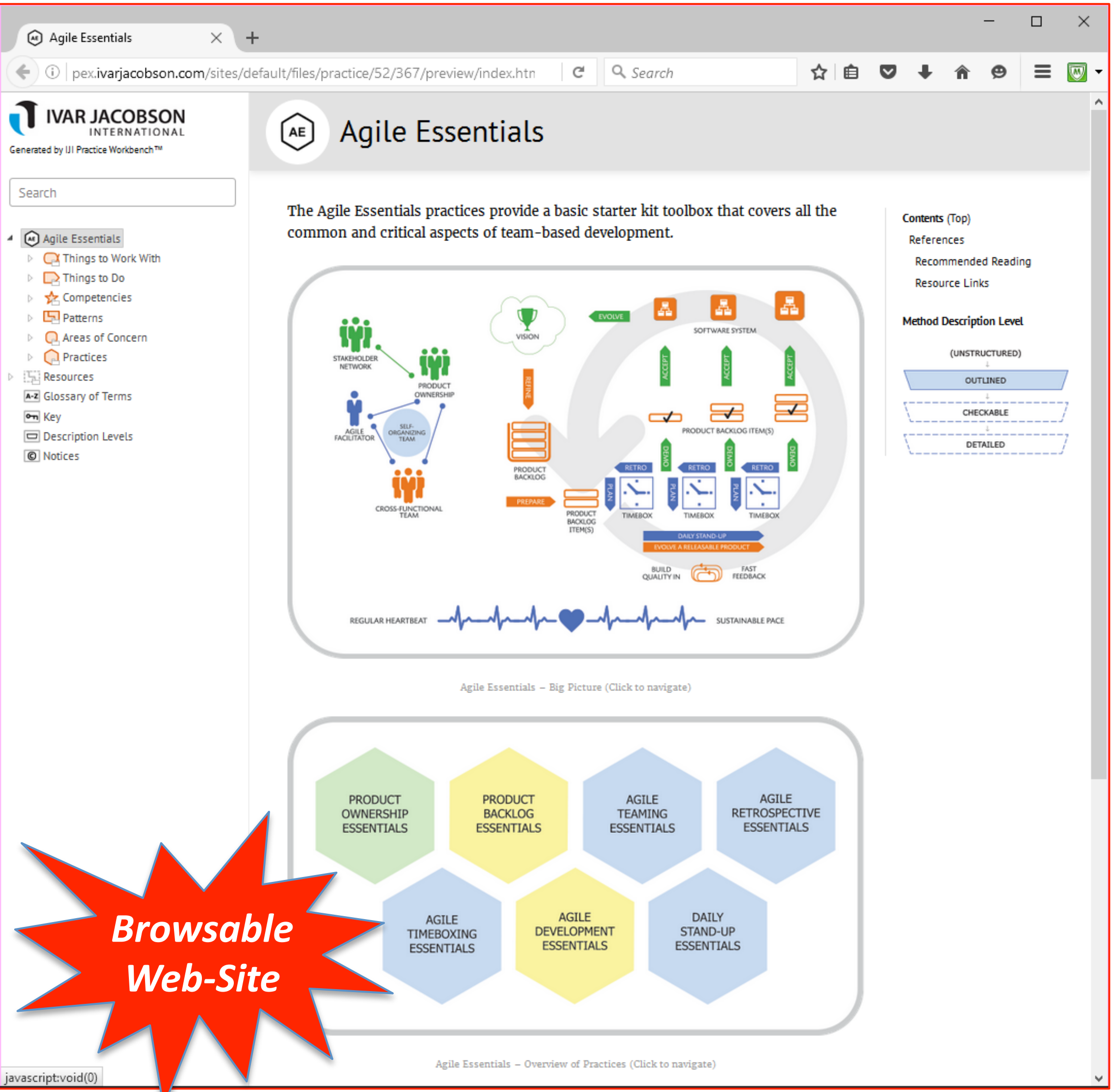
Team C

Practices enable teams to work the way that works for them

Agenda

1. IoT and Methods
2. Existing Methods puts you in Method Prisons
3. How to get out of your Method Prison?
4. Essentialization of Existing Methods
5. Essentialization of Ignite
6. What is the Value Proposition?
7. Next: It is Futurized
8. The Expectation Today

Agile Essentials



Agile Essentials

The Agile Essentials practices provide a basic starter kit toolbox that covers all the common and critical aspects of team-based development.

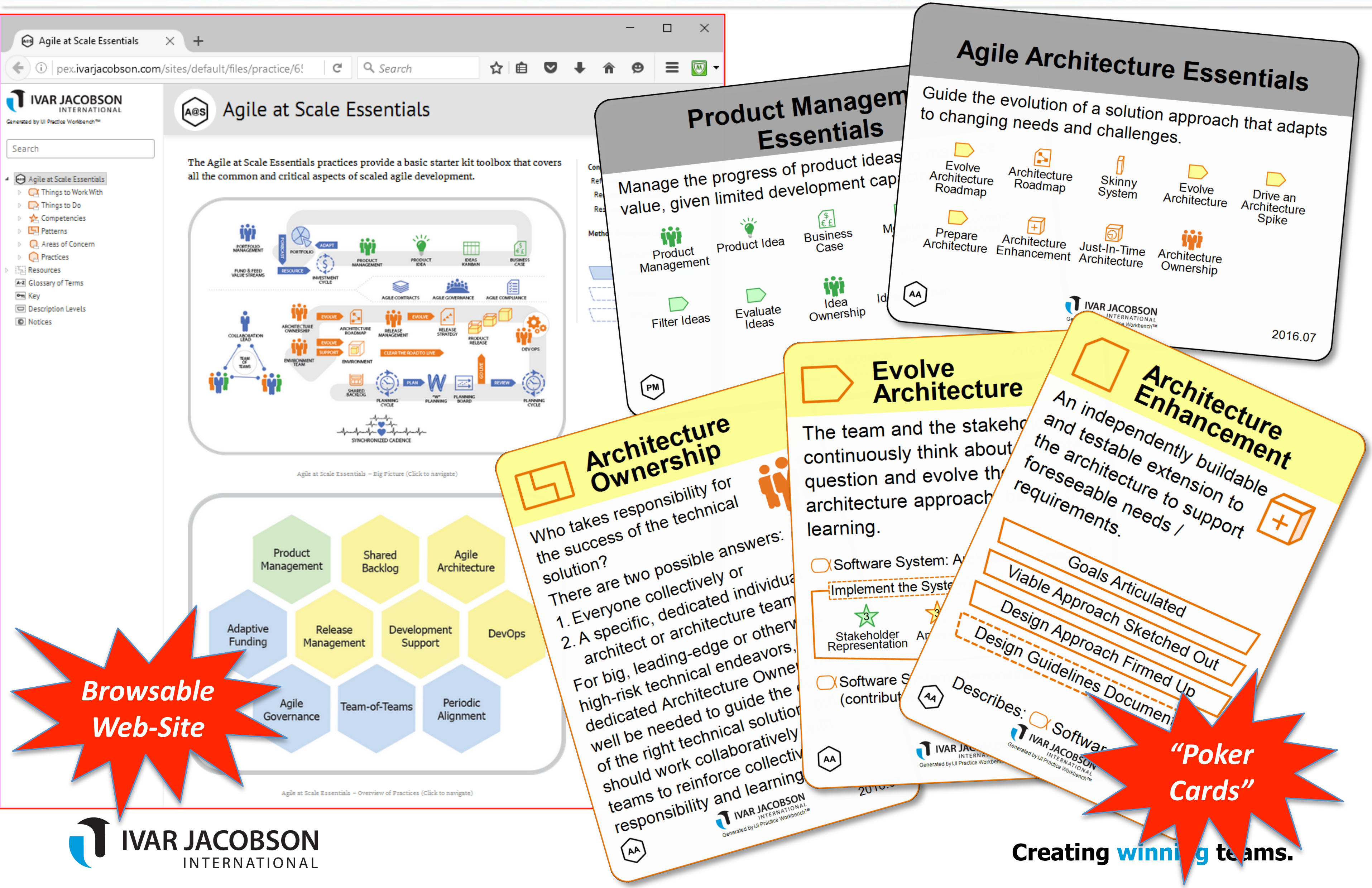
Agile Essentials – Big Picture (Click to navigate)

PRODUCT OWNERSHIP ESSENTIALS
PRODUCT BACKLOG ESSENTIALS
AGILE TEAMING ESSENTIALS
AGILE RETROSPECTIVE ESSENTIALS
AGILE TIMEBOXING ESSENTIALS
AGILE DEVELOPMENT ESSENTIALS
DAILY STAND-UP ESSENTIALS

Agile Essentials – Overview of Practices (Click to navigate)

Browsable Web-Site

Agile at Scale



Agile at Scale Essentials

pex.ivarjacobson.com/sites/default/files/practice/6!

IVAR JACOBSON INTERNATIONAL
Generated by IJI Practice Workbench™

Search

- Agile at Scale Essentials
  - Things to Work With
  - Things to Do
  - Competencies
  - Patterns
  - Areas of Concern
  - Practices
- Resources
- Glossary of Terms
- Key
- Description Levels
- Notices

Agile at Scale Essentials

The Agile at Scale Essentials practices provide a basic starter kit toolbox that covers all the common and critical aspects of scaled agile development.

Agile at Scale Essentials – Big Picture (Click to navigate)

Agile at Scale Essentials – Overview of Practices (Click to navigate)

Agile Architecture Essentials

Guide the evolution of a solution approach that adapts to changing needs and challenges.

IVAR JACOBSON INTERNATIONAL
2016.07

Product Management Essentials

Manage the progress of product ideas value, given limited development cap

Architecture Ownership

Who takes responsibility for the success of the technical solution?

There are two possible answers:

1. Everyone collectively or
2. A specific, dedicated individua architect or architecture team

For big, leading-edge or otherw high-risk technical endeavors, dedicated Architecture Owne well be needed to guide the of the right technical solution should work collaboratively teams to reinforce collectiv responsibility and learning

IVAR JACOBSON INTERNATIONAL
Generated by IJI Practice Workbench™

Evolve Architecture

The team and the stakeh continuously think about question and evolve th architecture approach learning.

Software System: A
Implement the Syste

Software S
(contribut

IVAR JAC
INTERNA
Generated by IJI Practice Workben

Architecture Enhancement

An independently buildable and testable extension to the architecture to support foreseeable needs / requirements.

Goals Articulated
Viable Approach Sketched Out
Design Approach Firmed Up
Design Guidelines Document

Describes: Softwar

IVAR JACOBSON INTERNATIONAL
Generated by IJI Practice Workbench™

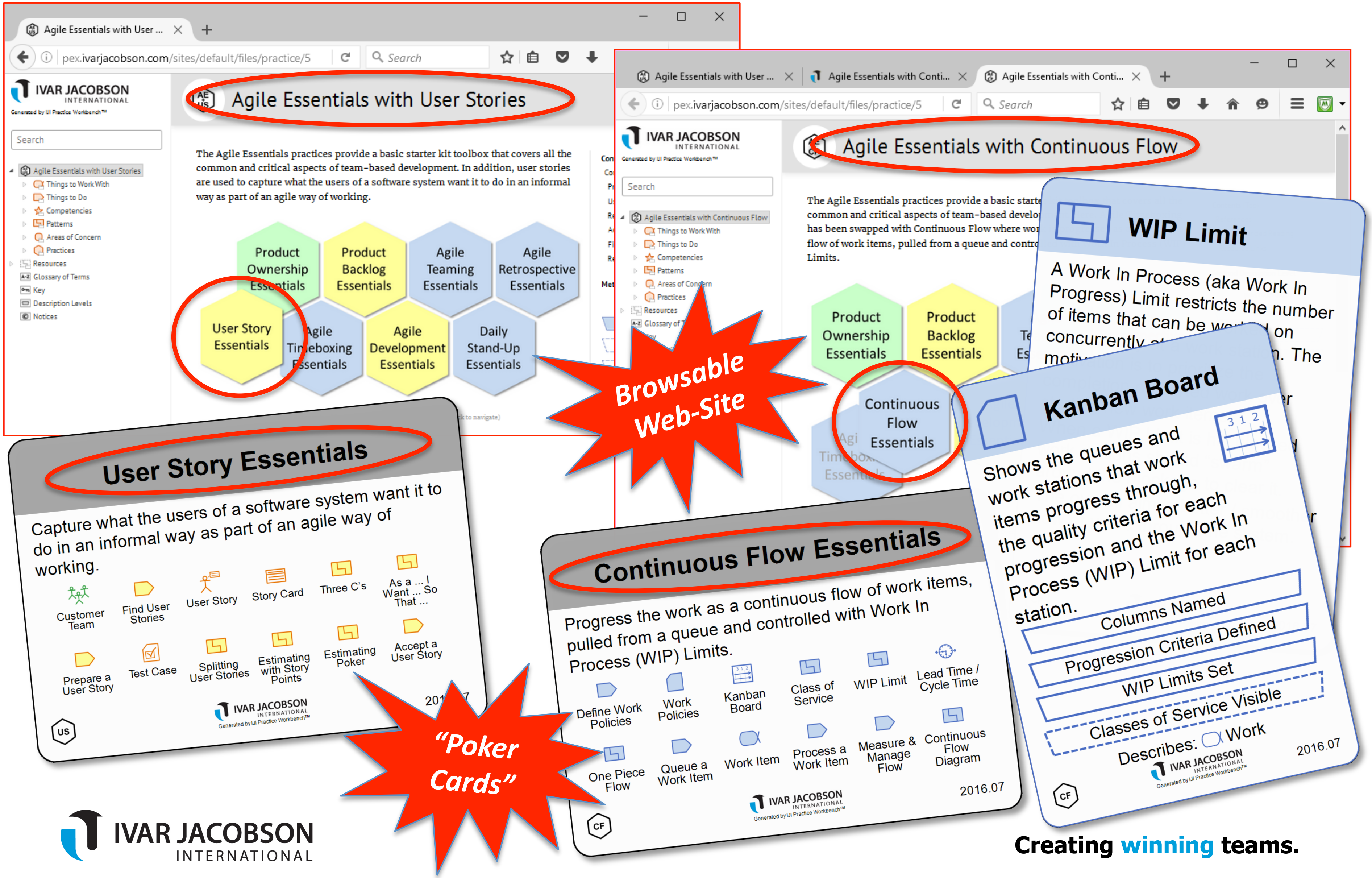
Browsable Web-Site

"Poker Cards"

IVAR JACOBSON INTERNATIONAL

Creating winning teams.

Add-Ins and Swap-Ins


Agile Essentials with User ...
pex.ivarjacobson.com/sites/default/files/practice/5
Search
IVAR JACOBSON INTERNATIONAL
Generated by IJI Practice Workbench™
Agile Essentials with User Stories
Things to Work With
Things to Do
Competencies
Patterns
Areas of Concern
Practices
Resources
Glossary of Terms
Key
Description Levels
Notices
Agile Essentials with Conti...
Agile Essentials with Continuous Flow
Browsable Web-Site
“Poker Cards”

Agile Essentials with User Stories

The Agile Essentials practices provide a basic starter kit toolbox that covers all the common and critical aspects of team-based development. In addition, user stories are used to capture what the users of a software system want it to do in an informal way as part of an agile way of working.

Product Ownership Essentials
Product Backlog Essentials
Agile Teaming Essentials
Agile Retrospective Essentials
User Story Essentials
Agile Timeboxing Essentials
Agile Development Essentials
Daily Stand-Up Essentials

Agile Essentials with Continuous Flow

The Agile Essentials practices provide a basic starte
common and critical aspects of team-based develo
has been swapped with Continuous Flow where wo
flow of work items, pulled from a queue and contro
Limits.

Product Ownership Essentials
Product Backlog Essentials
Continuous Flow Essentials
Agile Timeboxing Essentials

User Story Essentials

Capture what the users of a software system want it to do in an informal way as part of an agile way of working.

Customer Team
Find User Stories
User Story
Story Card
Three C's
As a ... I Want ... So That ...
Prepare a User Story
Test Case
Splitting User Stories
Estimating with Story Points
Estimating Poker
Accept a User Story

US
IVAR JACOBSON INTERNATIONAL
Generated by IJI Practice Workbench™
2016.07

Continuous Flow Essentials

Progress the work as a continuous flow of work items, pulled from a queue and controlled with Work In Process (WIP) Limits.

Define Work Policies
Work Policies
Kanban Board
Class of Service
WIP Limit
Lead Time / Cycle Time
One Piece Flow
Queue a Work Item
Work Item
Process a Work Item
Measure & Manage Flow
Continuous Flow Diagram

CF
IVAR JACOBSON INTERNATIONAL
Generated by IJI Practice Workbench™
2016.07

WIP Limit

A Work In Process (aka Work In Progress) Limit restricts the number of items that can be wo on concurrently a moti n. The

Kanban Board

Shows the queues and work stations that work items progress through, the quality criteria for each progression and the Work In Process (WIP) Limit for each station.

Columns Named
Progression Criteria Defined
WIP Limits Set
Classes of Service Visible

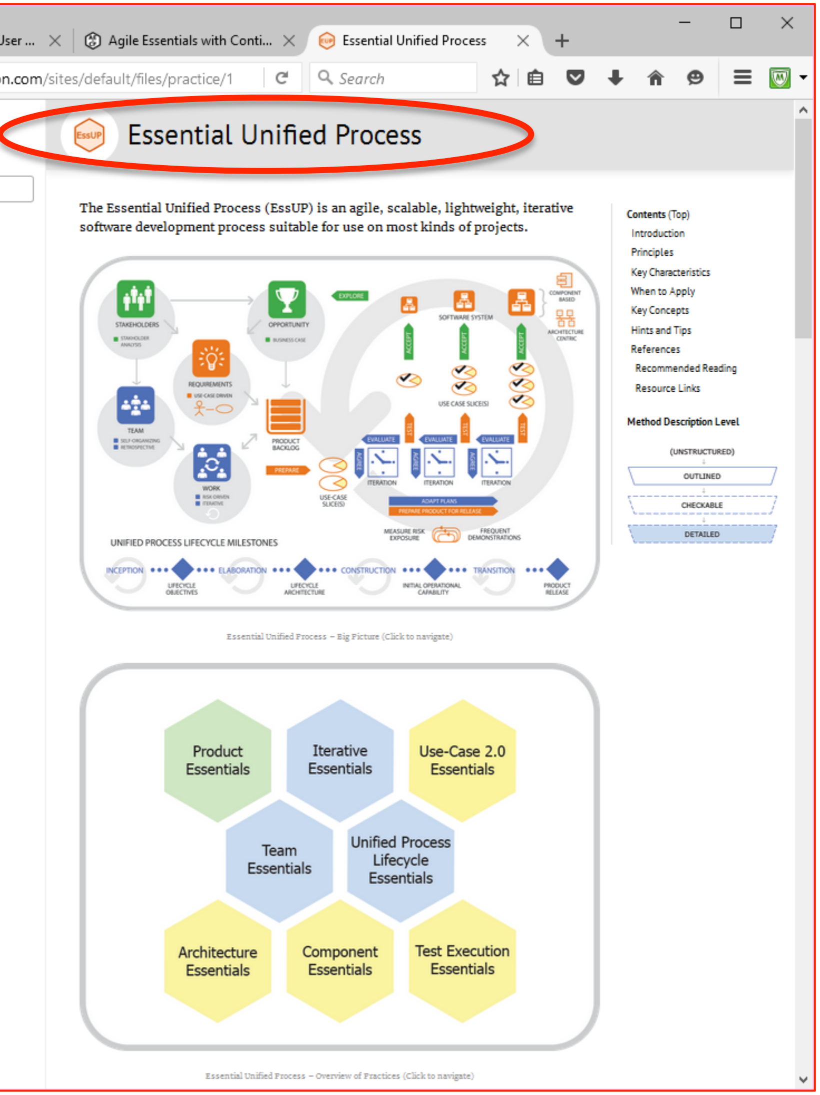
Describes: Work

CF
IVAR JACOBSON INTERNATIONAL
Generated by IJI Practice Workbench™
2016.07

IVAR JACOBSON INTERNATIONAL

Creating **winning** teams.

Essentialized Practices/Methods (more coming …)



Essential Unified Process

The Essential Unified Process (EssUP) is an agile, scalable, lightweight, iterative software development process suitable for use on most kinds of projects.

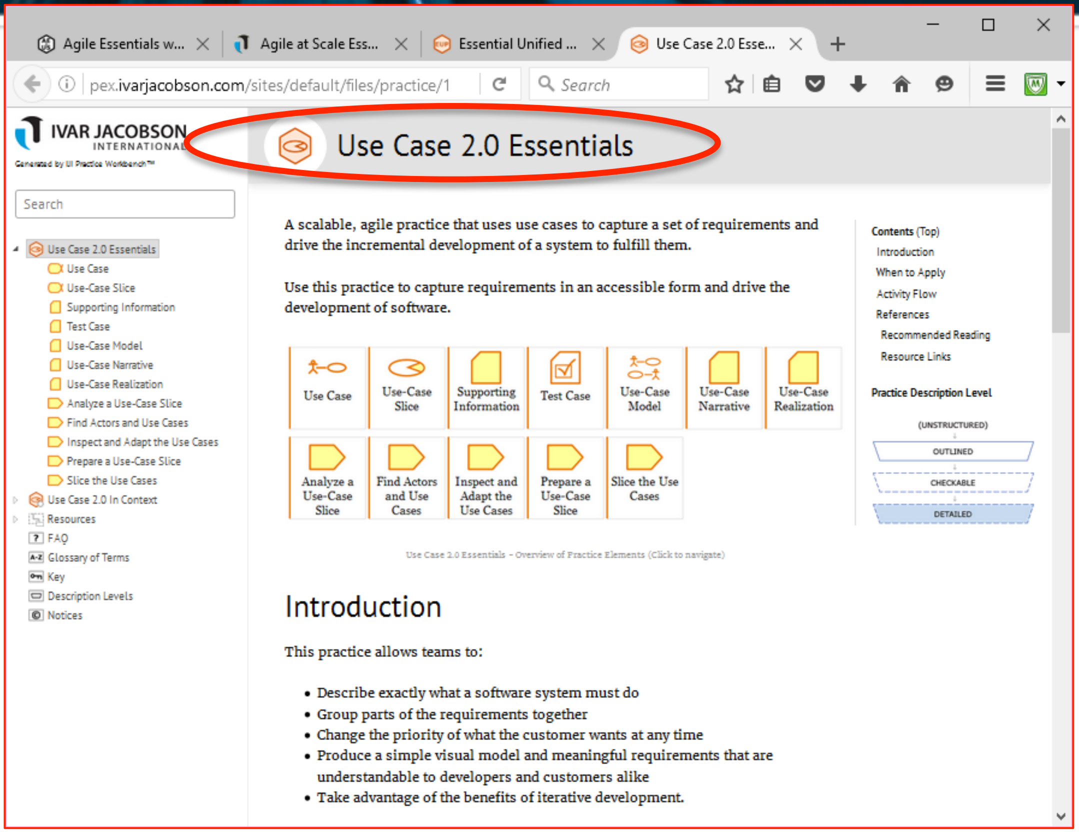
Essential Unified Process – Big Picture (Click to navigate)

Essential Unified Process – Overview of Practices (Click to navigate)

Contents (Top)
- Introduction
- Principles
- Key Characteristics
- When to Apply
- Key Concepts
- Hints and Tips
- References
  - Recommended Reading
  - Resource Links

Method Description Level

(UNSTRUCTURED)
OUTLINED
CHECKABLE
DETAILED



Use Case 2.0 Essentials
- Use Case
- Use-Case Slice
- Supporting Information
- Test Case
- Use-Case Model
- Use-Case Narrative
- Use-Case Realization
- Analyze a Use-Case Slice
- Find Actors and Use Cases
- Inspect and Adapt the Use Cases
- Prepare a Use-Case Slice
- Slice the Use Cases

Use Case 2.0 In Context
Resources
FAQ
Glossary of Terms
Key
Description Levels
Notices

Use Case 2.0 Essentials

A scalable, agile practice that uses use cases to capture a set of requirements and drive the incremental development of a system to fulfill them.

Use this practice to capture requirements in an accessible form and drive the development of software.

Use Case 2.0 Essentials – Overview of Practice Elements (Click to navigate)

Introduction

This practice allows teams to:

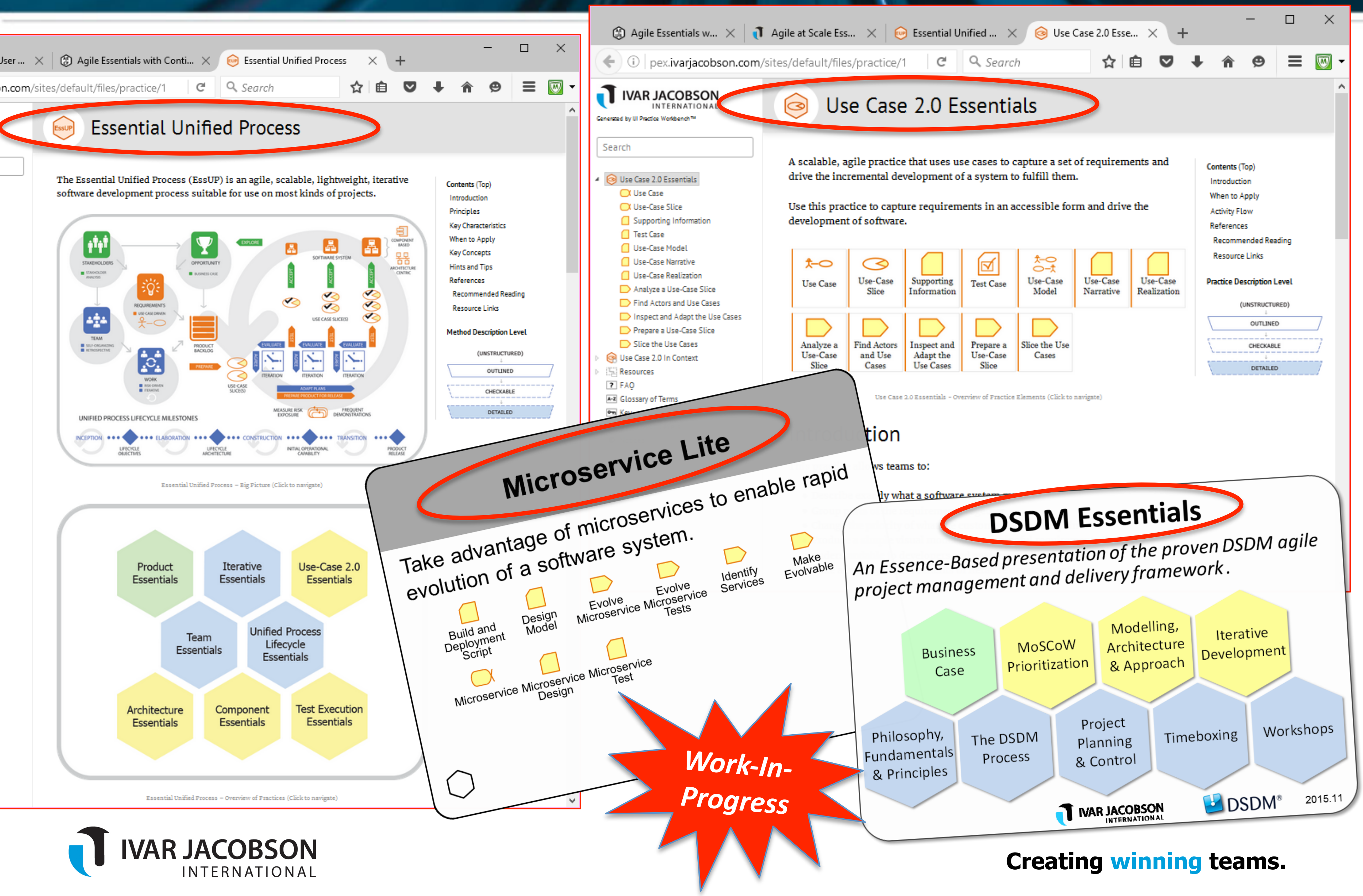
- Describe exactly what a software system must do
- Group parts of the requirements together
- Change the priority of what the customer wants at any time
- Produce a simple visual model and meaningful requirements that are understandable to developers and customers alike
- Take advantage of the benefits of iterative development.

Contents (Top)
- Introduction
- When to Apply
- Activity Flow
- References
  - Recommended Reading
  - Resource Links

Practice Description Level

(UNSTRUCTURED)
OUTLINED
CHECKABLE
DETAILED

Essentialized Practices/Methods (more coming …)



Essential Unified Process

The Essential Unified Process (EssUP) is an agile, scalable, lightweight, iterative software development process suitable for use on most kinds of projects.

Contents (Top)
- Introduction
- Principles
- Key Characteristics
- When to Apply
- Key Concepts
- Hints and Tips
- References
  - Recommended Reading
  - Resource Links

Method Description Level

Essential Unified Process – Big Picture (Click to navigate)

Essential Unified Process – Overview of Practices (Click to navigate)

Use Case 2.0 Essentials

A scalable, agile practice that uses use cases to capture a set of requirements and drive the incremental development of a system to fulfill them.

Use this practice to capture requirements in an accessible form and drive the development of software.

- Use Case 2.0 Essentials
  - Use Case
  - Use-Case Slice
  - Supporting Information
  - Test Case
  - Use-Case Model
  - Use-Case Narrative
  - Use-Case Realization
  - Analyze a Use-Case Slice
  - Find Actors and Use Cases
  - Inspect and Adapt the Use Cases
  - Prepare a Use-Case Slice
  - Slice the Use Cases
- Use Case 2.0 In Context
- Resources
- FAQ
- Glossary of Terms
- Key

Contents (Top)
- Introduction
- When to Apply
- Activity Flow
- References
  - Recommended Reading
  - Resource Links

Practice Description Level

Use Case 2.0 Essentials – Overview of Practice Elements (Click to navigate)

tion

ws teams to:

ly what a software system

Microservice Lite

Take advantage of microservices to enable rapid evolution of a software system.

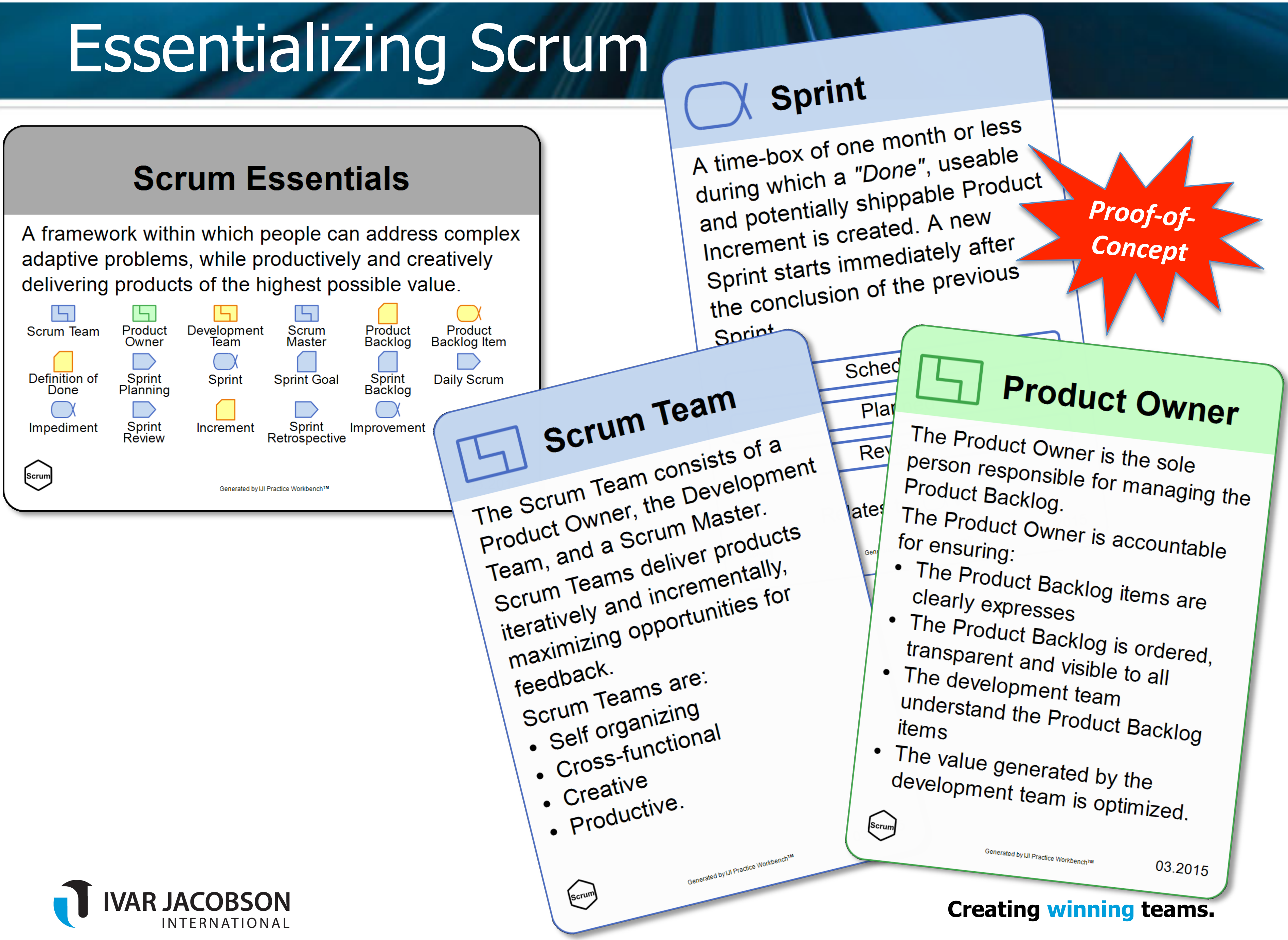
DSDM Essentials

An Essence-Based presentation of the proven DSDM agile project management and delivery framework.

IVAR JACOBSON INTERNATIONAL

Creating winning teams.

Essentializing Scrum



Scrum Essentials

A framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value.

Scrum Team · Product Owner · Development Team · Scrum Master · Product Backlog · Product Backlog Item

Definition of Done · Sprint Planning · Sprint · Sprint Goal · Sprint Backlog · Daily Scrum

Impediment · Sprint Review · Increment · Sprint Retrospective · Improvement

Scrum

Generated by IJI Practice Workbench™

Sprint

A time-box of one month or less during which a "Done", useable and potentially shippable Product Increment is created. A new Sprint starts immediately after the conclusion of the previous Sprint

Sched

Plar

Rev

lates

Gen

Proof-of-Concept

Scrum Team

The Scrum Team consists of a Product Owner, the Development Team, and a Scrum Master. Scrum Teams deliver products iteratively and incrementally, maximizing opportunities for feedback.

Scrum Teams are:

- Self organizing
- Cross-functional
- Creative
- Productive.

Scrum

Generated by IJI Practice Workbench™

Product Owner

The Product Owner is the sole person responsible for managing the Product Backlog.

The Product Owner is accountable for ensuring:

- The Product Backlog items are clearly expresses
- The Product Backlog is ordered, transparent and visible to all
- The development team understand the Product Backlog items
- The value generated by the development team is optimized.

Scrum

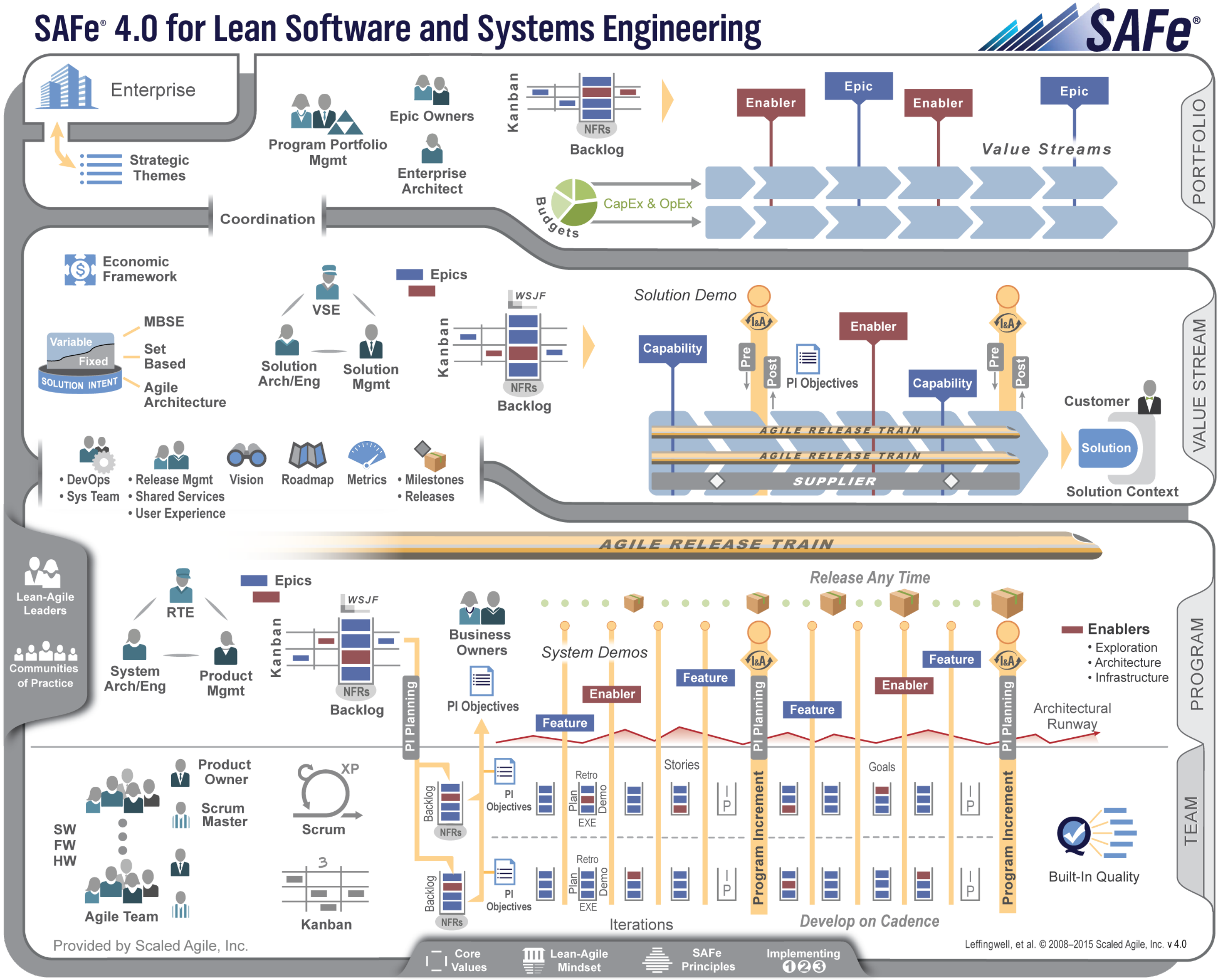
Generated by IJI Practice Workbench™

03.2015

IVAR JACOBSON INTERNATIONAL

Creating winning teams.

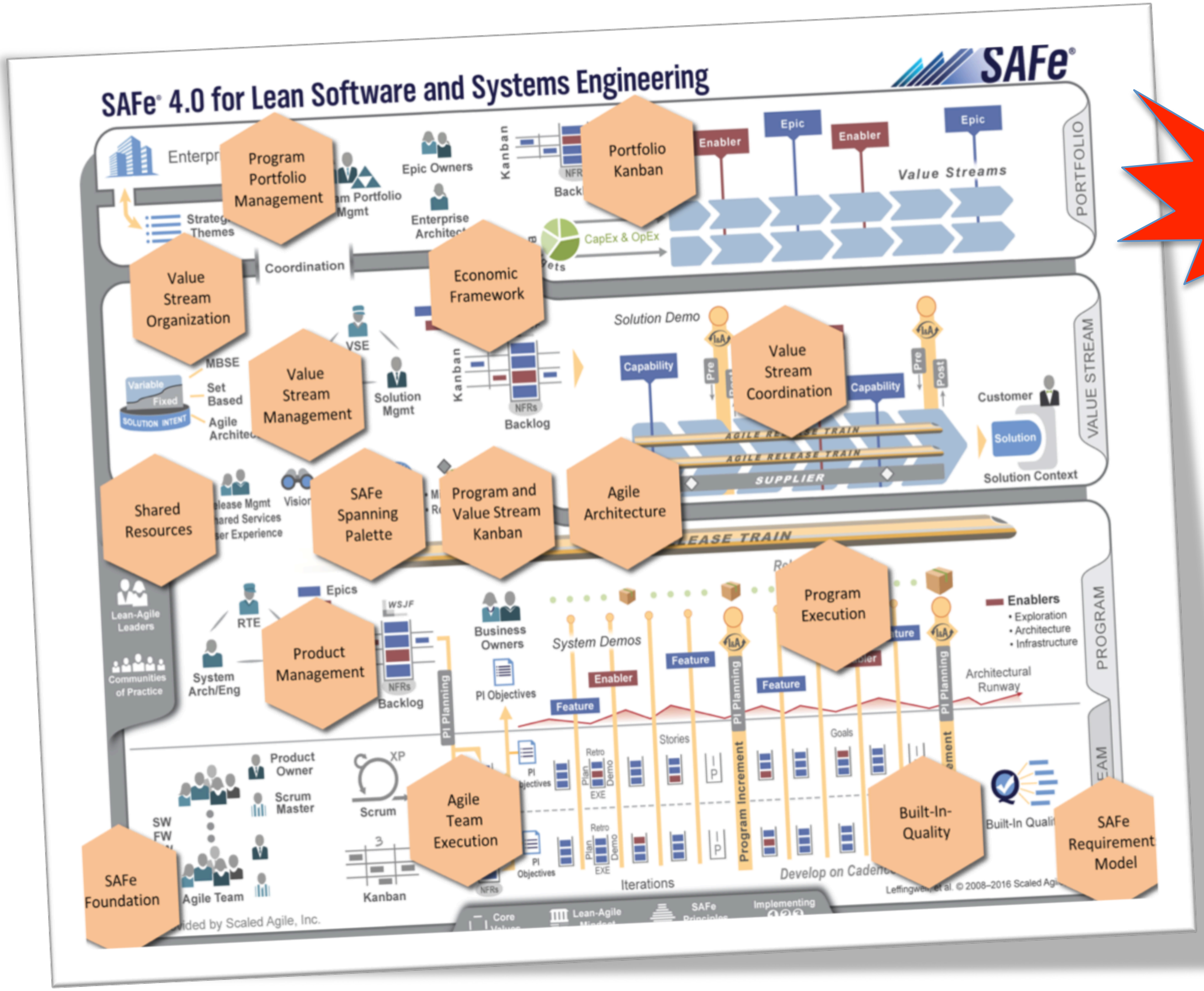
SAFe 4.0



IVAR JACOBSON INTERNATIONAL

Creating winning teams.

Essentializing SAFe 4.0

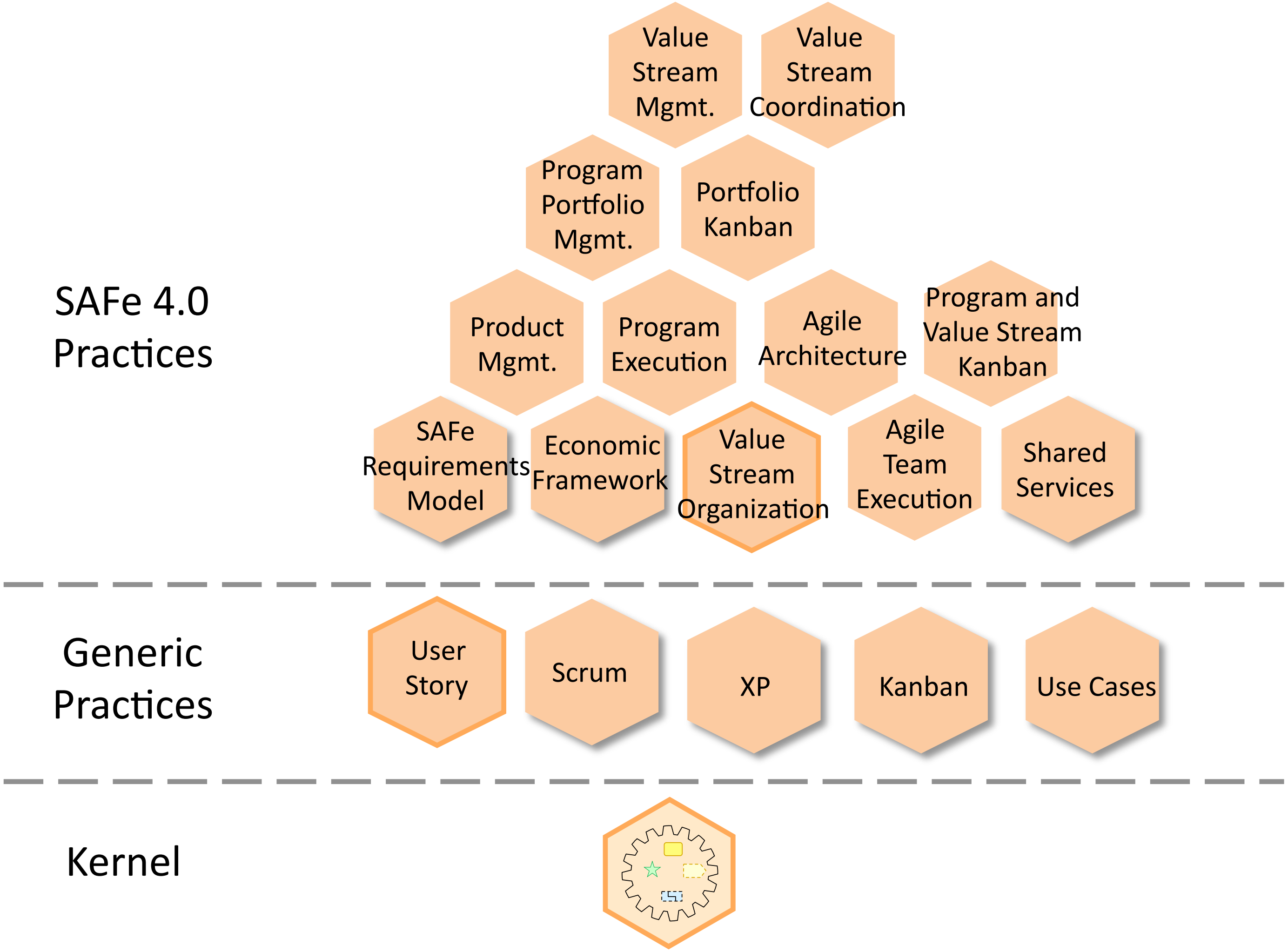


Proof-of-Concept

IVAR JACOBSON INTERNATIONAL

Creating winning teams.

The Method SAFe4.0 Essentialized



Value Stream Coordination

Value Stream Mgmt.

Value Stream Coordination

Portfolio Practices

Program Portfolio Mgmt.

Portfolio Kanban

SAFe 4.0 Practices

Program Practices

Product Mgmt.

Program Execution

Agile Architecture

Program and Value Stream Kanban

Infrastructure Practices

SAFe Requirements Model

Economic Framework

Value Stream Organization

Agile Team Execution

Shared Services

Generic Practices

User Story

Scrum

XP

Kanban

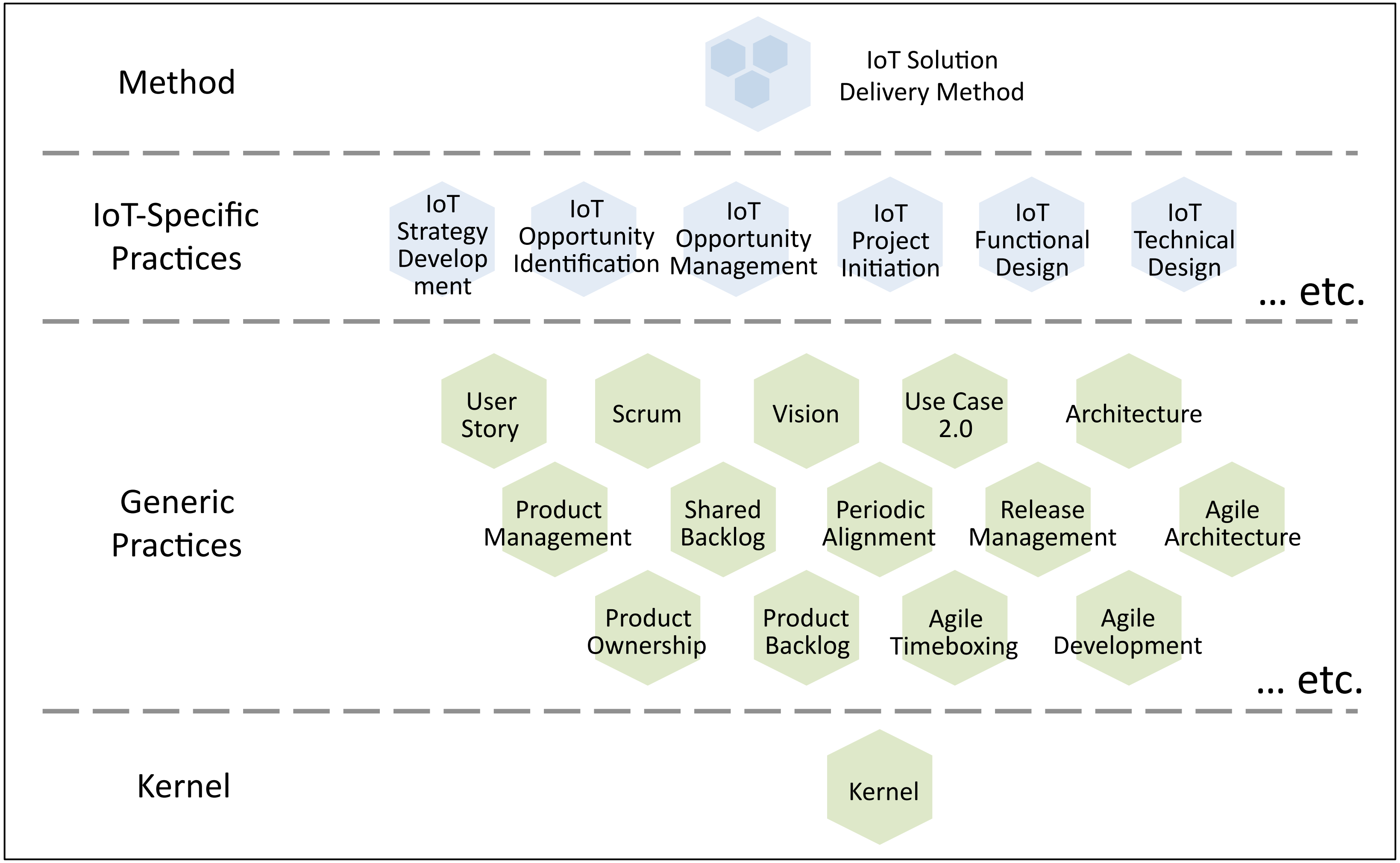
Use Cases

Kernel

Agenda

1. IoT and Methods
2. Existing Methods puts you in Method Prisons
3. How to get out of your Method Prison?
4. Essentialization of Existing Methods
5. Essentialization of Ignite
6. What is the Value Proposition?
7. Next: It is Futurized
8. The Expectation Today

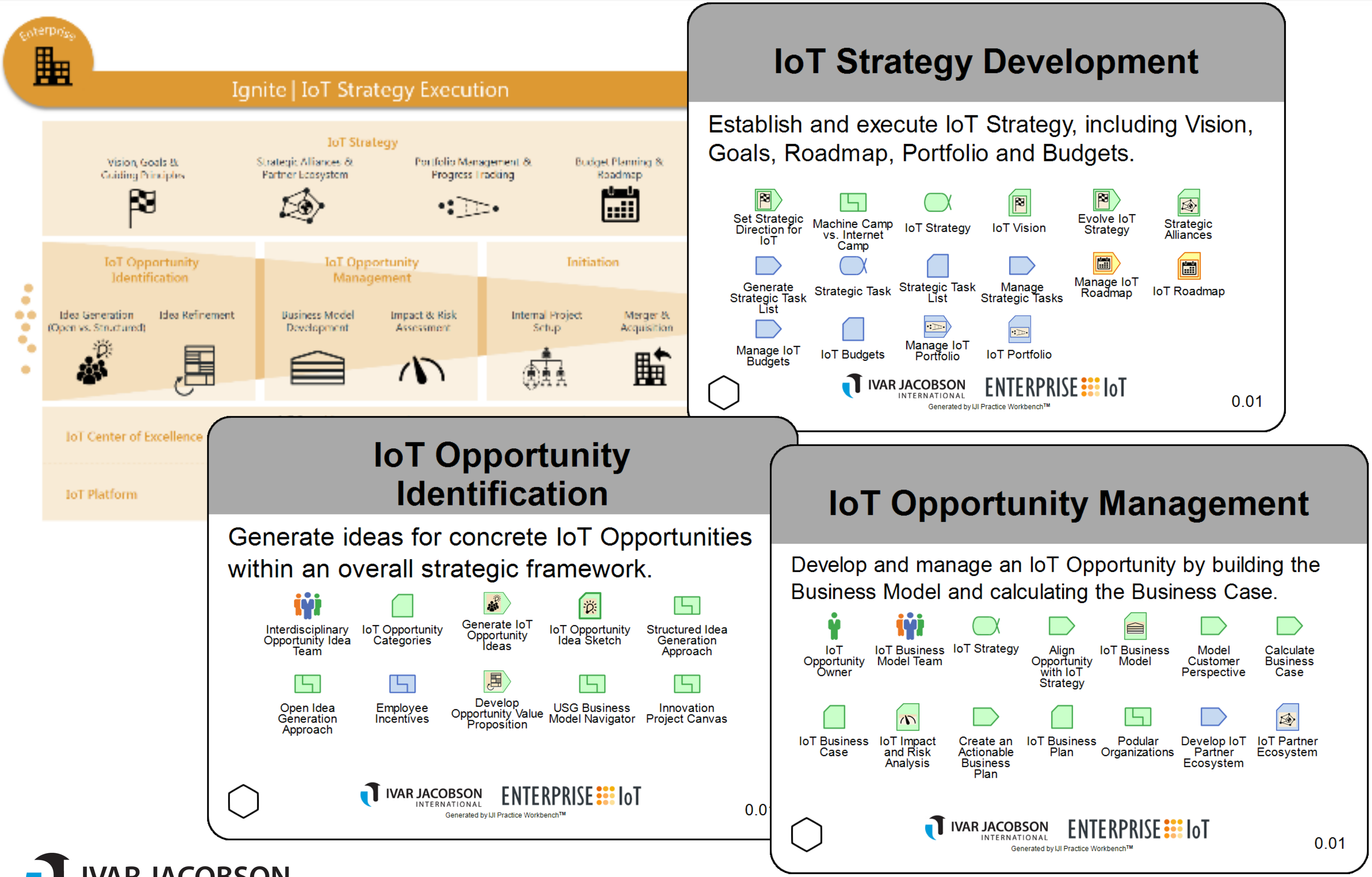
Ignite expressed as a set of Essence Practices



IVAR JACOBSON INTERNATIONAL

Creating winning teams.

IoT Strategy: Sample Practice Cards


Enterprise
Ignite | IoT Strategy Execution
IoT Strategy
Vision, Goals & Guiding Principles
Strategic Alliances & Partner Ecosystem
Portfolio Management & Progress Tracking
Budget Planning & Roadmap
IoT Opportunity Identification
Idea Generation (Open vs. Structured)
Idea Refinement
IoT Opportunity Management
Business Model Development
Impact & Risk Assessment
Initiation
Internal Project Setup
Merger & Acquisition
IoT Center of Excellence
IoT Platform

IoT Strategy Development

Establish and execute IoT Strategy, including Vision, Goals, Roadmap, Portfolio and Budgets.

- Set Strategic Direction for IoT
- Machine Camp vs. Internet Camp
- IoT Strategy
- IoT Vision
- Evolve IoT Strategy
- Strategic Alliances
- Generate Strategic Task List
- Strategic Task
- Strategic Task List
- Manage Strategic Tasks
- Manage IoT Roadmap
- IoT Roadmap
- Manage IoT Budgets
- IoT Budgets
- Manage IoT Portfolio
- IoT Portfolio

IVAR JACOBSON INTERNATIONAL ENTERPRISE IoT
Generated by IJI Practice Workbench™

0.01

IoT Opportunity Identification

Generate ideas for concrete IoT Opportunities within an overall strategic framework.

- Interdisciplinary Opportunity Idea Team
- IoT Opportunity Categories
- Generate IoT Opportunity Ideas
- IoT Opportunity Idea Sketch
- Structured Idea Generation Approach
- Open Idea Generation Approach
- Employee Incentives
- Develop Opportunity Value Proposition
- USG Business Model Navigator
- Innovation Project Canvas

IVAR JACOBSON INTERNATIONAL ENTERPRISE IoT
Generated by IJI Practice Workbench™

0.0

IoT Opportunity Management

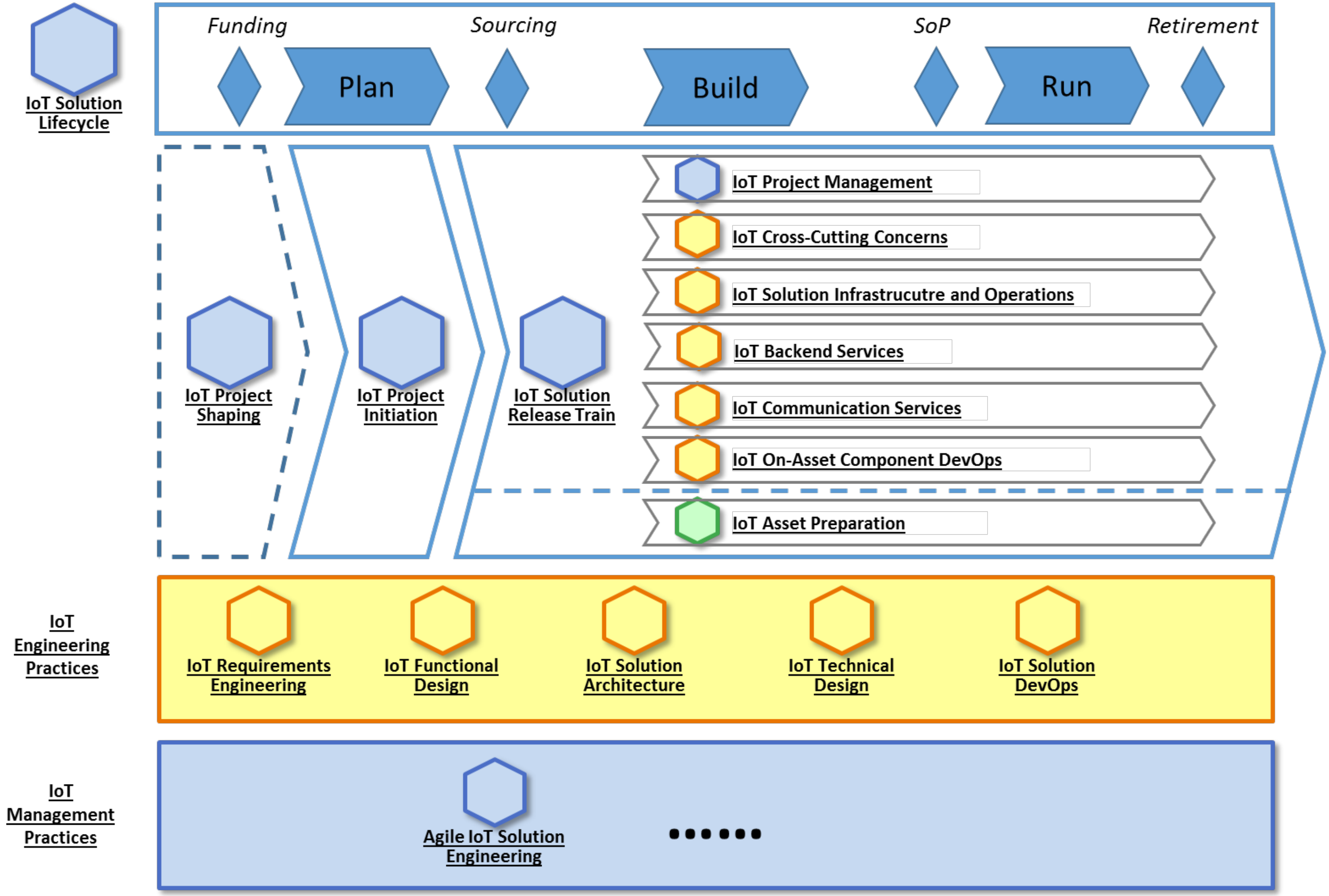
Develop and manage an IoT Opportunity by building the Business Model and calculating the Business Case.

- IoT Opportunity Owner
- IoT Business Model Team
- IoT Strategy
- Align Opportunity with IoT Strategy
- IoT Business Model
- Model Customer Perspective
- Calculate Business Case
- IoT Business Case
- IoT Impact and Risk Analysis
- Create an Actionable Business Plan
- IoT Business Plan
- Podular Organizations
- Develop IoT Partner Ecosystem
- IoT Partner Ecosystem

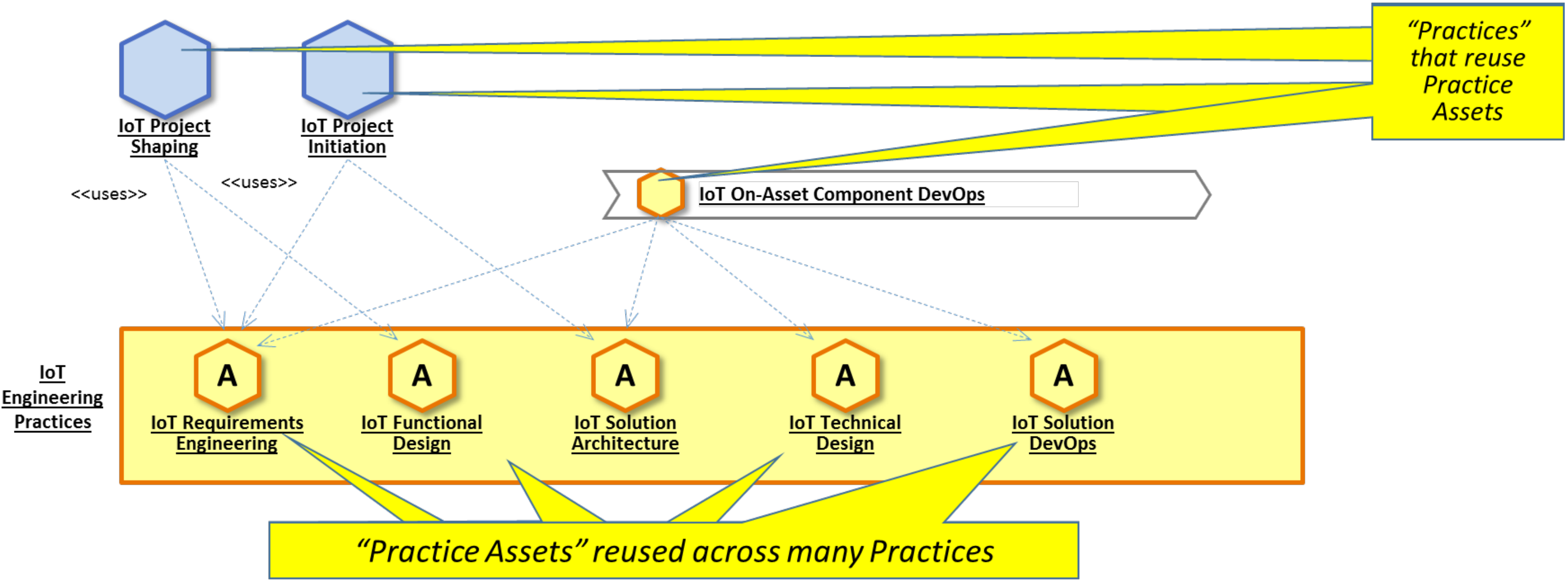
IVAR JACOBSON INTERNATIONAL ENTERPRISE IoT
Generated by IJI Practice Workbench™

0.01

IoT Solution Development "Big Picture"

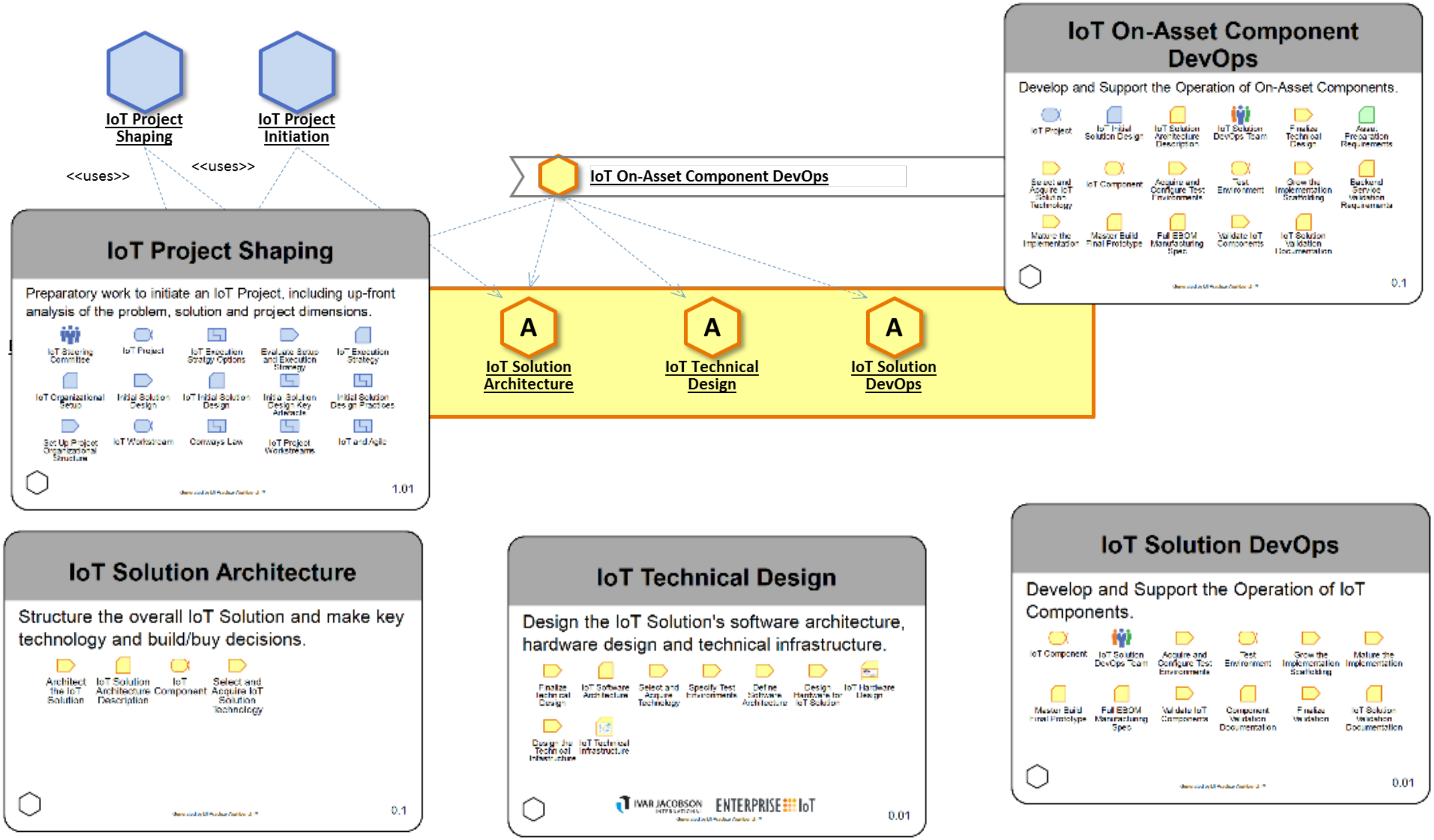

IoT Solution Lifecycle
Funding
Plan
Sourcing
Build
SoP
Run
Retirement
IoT Project Shaping
IoT Project Initiation
IoT Solution Release Train
IoT Project Management
IoT Cross-Cutting Concerns
IoT Solution Infrastrucutre and Operations
IoT Backend Services
IoT Communication Services
IoT On-Asset Component DevOps
IoT Asset Preparation
IoT Engineering Practices
IoT Requirements Engineering
IoT Functional Design
IoT Solution Architecture
IoT Technical Design
IoT Solution DevOps
IoT Management Practices
Agile IoT Solution Engineering
•••••••

Approach to Reuse Across Practices



"Practices" that reuse Practice Assets
IoT Project Shaping
IoT Project Initiation
<<uses>>
<<uses>>
IoT On-Asset Component DevOps
IoT Engineering Practices
A
IoT Requirements Engineering
A
IoT Functional Design
A
IoT Solution Architecture
A
IoT Technical Design
A
IoT Solution DevOps
"Practice Assets" reused across many Practices

IoT Solution: Sample Practice Cards


IoT Project Shaping
IoT Project Initiation
<<uses>>
<<uses>>
IoT On-Asset Component DevOps
A
IoT Solution Architecture
A
IoT Technical Design
A
IoT Solution DevOps

IoT On-Asset Component DevOps

Develop and Support the Operation of On-Asset Components.

IoT Project
IoT Initial Solution Design
IoT Solution Architecture Description
IoT Solution DevOps Team
Finalize Technical Design
Asset Preparation Requirements
Select and Acquire IoT Solution Technology
IoT Component
Acquire and Configure Test Environments
Test Environment
Grow the Implementation Scaffolding
Backend Service Validation Requirements
Mature the Implementation
Master Build Final Prototype
Full EBOM Manufacturing Spec
Validate IoT Components
IoT Solution Validation Documentation
0.1

IoT Project Shaping

Preparatory work to initiate an IoT Project, including up-front analysis of the problem, solution and project dimensions.

IoT Steering Committee
IoT Project
IoT Execution Stratgy Options
Evaluate Setup and Execution Strategy
IoT Execution Strategy
IoT Organizational Setup
Initial Solution Design
IoT Initial Solution Design
Initial Solution Design Key Artefacts
Initial Solution Design Practices
Set Up Project Organizational Structure
IoT Workstream
Conways Law
IoT Project Workstreams
IoT and Agile
1.01

IoT Solution Architecture

Structure the overall IoT Solution and make key technology and build/buy decisions.

Architect the IoT Solution
IoT Solution Architecture Description
IoT Component
Select and Acquire IoT Solution Technology
0.1

IoT Technical Design

Design the IoT Solution's software architecture, hardware design and technical infrastructure.

Finalize Technical Design
IoT Software Architecture
Select and Acquire Technology
Specify Test Environments
Define Software Architecture
Design Hardware for IoT Solution
IoT Hardware Design
Design the Technical Infrastructure
IoT Technical Infrastructure
IVAR JACOBSON INTERNATIONAL
ENTERPRISE IoT
0.01

IoT Solution DevOps

Develop and Support the Operation of IoT Components.

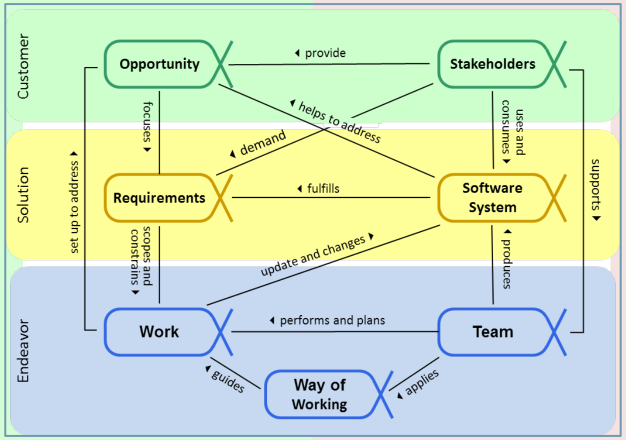
IoT Component
IoT Solution DevOps Team
Acquire and Configure Test Environments
Test Environment
Grow the Implementation Scaffolding
Mature the Implementation
Master Build Final Prototype
Full EBOM Manufacturing Spec
Validate IoT Components
Component Validation Documentation
Finalize Validation
IoT Solution Validation Documentation
0.01

Agenda

1. IoT and Methods
2. Existing Methods puts you in Method Prisons
3. How to get out of your Method Prison?
4. Essentialization of Existing Methods
5. Essentialization of Ignite
6. What is the Value Proposition?
7. Next: It is Futurized
8. The Expectation Today

Essence has Two Major Usages

Practice Descriptions

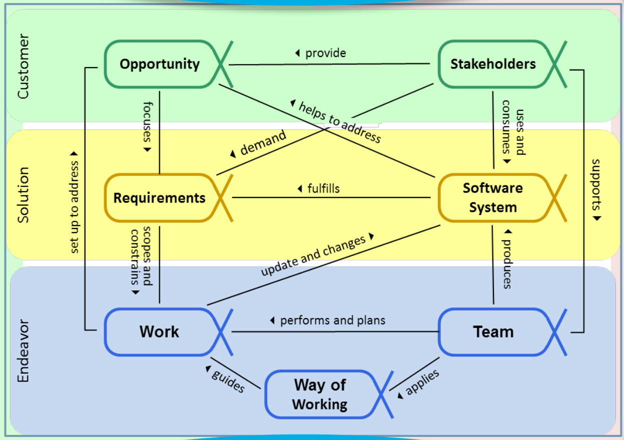


Practice Application

The Major Impact of Essence

Industrial Scale Engineering

Practice Descriptions



Practice Application

A Learning Organization

Essence Value Proposition

While preparing

- We have a Common Ground
- All methods can be modularized
- Build Practice Libraries with practices from many providers
- Safely Mix and Match practices best for you
- Easy to compose team's own ways-of-working
- Harvest and share the latest practices
- Just what you need – the Essence
- Easy to present and train, learn and apply
- Easy to compare and contrast
- Easy to get team's started
- Easy to continuously improve and keep up to date

While working

Active guidance and gamification

Monitor progress and health avoiding catastrophic failures

Understand where you are and where you're going

Practice independent governance practices

Grow your way-of-working as you grow your team

Essentialization moves us to….

Industrial Scale Engineering

- Systematically address the methods to allow for dramatic efficiency and quality improvements through tooling and techniques
- Right size the applied methods to fit the problems at hand with minimum overhead, which shortens time to market
- Application of many engineering practices for
  - requirements such as use cases, features, user stories
  - design and architecture patterns, for developing components and services
  - testing complex, distributed systems
  - encouraging systematic reuse
  - helping engineers code with confidence
  - architectural concerns such as concurrency, security, user experience, micro-services, and data protection
- Application of practices with broader architectural concerns such as enterprise architecture, product-line architecture, service-oriented architecture and the architecture of systems of systems
- Working systematically instead of relying on heroics

A Learning Organization

- Common language / common culture
  - Create your own kernel – if needed
  - Establish shared common ground for all teams
  - Exchange and share practices and experiences
- Increase the competency of every individual
  - Building practice libraries accessible to everyone
  - Continuously improve
  - Nurture communities of practices
  - Share practice
  - Directed coaching
  - Practice-based accreditation
- Create winning teams
  - Plug and play methods and practices
  - Track progress and health
  - Lightweight, practical governance
- More competent people will
  - develop better software faster and cheaper with happier customers
  - innovate more effectively

Agenda

1. IoT and Methods
2. Existing Methods puts you in Method Prisons
3. How to get out of your Method Prison?
4. Essentialization of Existing Methods
5. Essentialization of Ignite
6. What is the Value Proposition?
7. Next: It is Futurized
8. The Expectation Today

The Future is around the corner

- Less than 20% of software development work is really creative
- 80% is “no-brain” work following well-known patterns
- It is expected that at the least 50% of the “no-brain” work can be taken over by intelligent agents (expert systems)
- Developers can be liberated from some “no-brain” work to really innovate instead of solving the similar problem over and over again
- Practices are perfect candidates for intelligent agents
- We have extensive experience in designing intelligent agents through Waypointer

Virtual Pair

- Programmers
- Analysts
- Designer
- Tester
- Project Managers

Agenda

1. IoT and Methods
2. Existing Methods puts you in Method Prisons
3. How to get out of your Method Prison?
4. Essentialization of Existing Methods
5. Essentialization of Ignite
6. What is the Value Proposition?
7. Next: It is Futurized
8. The Expectation Today

Early Adopters' Expectations

- you can do twice as good a job…
- you can do the job twice as fast…
- you can do the job with half the people…
- you can do all that and still make your customer happier…

Early Adopters' Expectations

- you can do twice as good a job…
- you can do the job twice as fast
- you can do the job wi
- you can d ke your

Welcome to the Future!