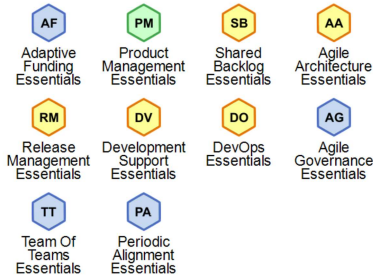


Agile at Scale Essentials

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

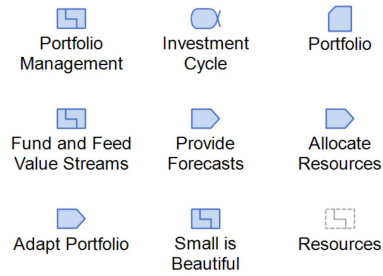


IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09

Adaptive Funding Essentials

Acquire and allocate funding to support continuous value flow and maximize return on investment.



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09

Portfolio Management

Single point of responsibility for the allocation of portfolio investments and resources to maximize overall return-on-investment and balance short and longer term considerations.



This is typically a group of senior executives that together own and drive the overall product strategy.

They collaborate with Product and Release Management to dynamically adapt investment and resource allocations over time.

Responsible For:

- Allocate Resources and
- Adapt Portfolio



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09

Investment Cycle

A time period in which the resources for a portfolio are forecasted, allocated and managed.



The shorter the cycle, the more responsive and adaptive the allocation of funds and resources.

Forecasted

Resourced

Balanced

Relates to: Work



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Portfolio

A collection of portfolio investments, each with strategic objectives, resource allocations and funding requirements.



Items Listed

Allocations Justified

Trend Data Analyzed

Describes: Work



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Fund and Feed Value Streams

“Stop-Go” funding of short-lived projects, forming and disbanding delivery capability each time, delays value and reduces productivity.

To avoid this, establish and fund value streams as long-lived delivery channels, and feed funded initiatives into these in priority order. Adjust funding regularly (e.g. quarterly) based on performance and strategy.

Devolve resourcing decisions to delivery streams, where local expertise can allocate resources to maximize value throughput.

Guides: Allocate Resources



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Provide Forecasts

Provide regular, rolling forecasts on the resources needed to meet demand for new and improved products and services, and to deliver the organization’s strategic vision.

Prepare to do the Work



Stakeholder Representation



Leadership



Management

Work: Prepared (contributes to)

Portfolio: Items Listed

Investment Cycle: Forecasted



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Allocate Resources

Allocate resources across the portfolio of work in order to steer the development work in line with strategic goals.

Work

Investment Cycle: Forecasted

Coordinate Activity



Stakeholder Representation



Management

Work: Under Control (contributes to)

Portfolio: Allocations Justified

Investment Cycle: Resourced



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09

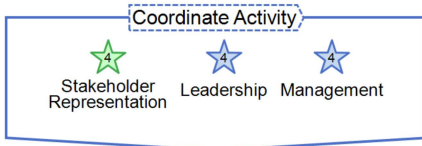


Adapt Portfolio

Continuously adapt the portfolio according to data and feedback from the market, and the overall performance of the portfolio.

Work

Investment Cycle: Resourced



Work: Under Control (contributes to)

Portfolio: Trend Data Analyzed

Investment Cycle: Balanced



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Small is Beautiful

Avoid large and long initiatives that defer value. Break big investments into smaller ones that deliver value early and often. Aim for a continuous flow of value return over time, as this delivers a better return-on-investment profile, and enables risk and exposure to be minimized.

If prioritization uses using a Weighted Shortest Job First (WSJF) formula, then smaller items will be prioritized over larger items, thus motivating the breaking up of large investment items.

Guides: Adapt Portfolio

Ref: Small is Beautiful



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Resources

- Essence Standard (OMG), refer to <http://www.omg.org/spec/Essence/1.1/>
- **Small is Beautiful**: Donald Reinertsen describes the economics and science of this in detail in "Reducing Batch Size" - ch. 5 of his book The Principles of Product Development Flow (Celeritas Publishing 2009).
- The Essence of Software Engineering: Applying the SEMAT Kernel, by Ivar Jacobson, Pan-Wei Ng, Paul E. McMahon, Ian Spence and Svante Lidman, Addison-Wesley 2013.
- This practice draws on various recognized industry practices, including **Beyond Budgeting** (see for example <http://bbri.org/about/what-is-beyond-budgeting/>), **Lean Accounting** (see for example https://en.wikipedia.org/wiki/Lean_accounting) and **Driver-Based Planning** (see for example <http://whatistechtarget.com/definition/driver-based-planning>).

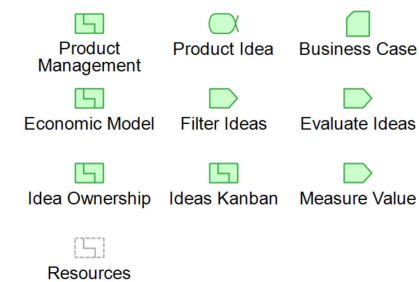


IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09

Product Management Essentials

Manage the progress of product ideas to maximize value, given limited development capacity.



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™



2018.09



Product Management

A single point of responsibility for direction and prioritization above and across many related teams, products, Product Owners or Stakeholder Networks, and for overseeing related work and progress. Depending on size and complexity, these responsibilities may rest with a single empowered individual and a supporting team, or with a product management group. The key for agility is that, in either case, they are highly available, engaged and responsive.



Evaluate and Prioritize:  Product Idea
Ref:  Product Management



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Product Idea

A proposal as to how some new value could be realized.



Value Agreed

Viability Assessed

Prioritized for Delivery

In Development

Realized

Value Measured

Relates to:  Opportunity



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Business Case

Articulates the anticipated value, feasibility, costs and the return-on-investment (ROI) case associated with a Product Idea.



Value Described

Viable Solution Outlined

ROI Case Established

Measurement Approach Designed

Describes:  Product Idea
Ref:  Business Case



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Economic Model

A framework to objectively assess the relative or absolute value of proposed investments in the product. Typically includes assessments of value-return as well as time-criticality and risk-reduction. Common approaches include Weighted Shortest Job First (WSJF), also known as Cost of Delay Divided by Duration (CD3).



Supports:  Evaluate Ideas
Ref:  Economic Model



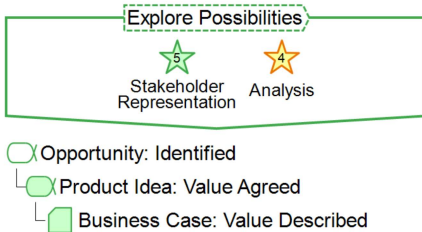
IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Filter Ideas

Ensure Product Ideas have sufficient relevance and possible value before investing in the further analysis needed to prioritize them for development.



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Evaluate Ideas

Analyze Product Ideas to determine what should be done in response, and what priority attaches to each possible response approach based on return-on-investment analysis.



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Idea Ownership

A single point of ownership for a Product Idea.



Idea Owners are responsible for the Business Case for an idea. They own the realization of the idea in the form of the backlog items and acceptance criteria that will realize it, and marshal the achievement of the business case by ensuring that the returns being achieved for the investments made is tracked and optimized over time.

Owns: Product Idea



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Ideas Kanban

Use a Kanban system to make Product Ideas visible, and to track progress in responding to ideas and prioritizing future value delivery.



Apply Work-In-Process (WIP) limits to ensure a smooth flow of work through the evaluation and prioritization process and into the delivery pipeline. The key is to focus on progressing high-value ideas, rather than working on all Ideas at once.

Tracks Progress Of: Product Idea

Ref: Ideas Kanban



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Measure Value

Measure the value of the Product Idea when it is in use by users, to validate against the business case and guide the development of future Product Ideas.

○ Opportunity: Addressed

○ Product Idea: Realized

Use the System



○ Opportunity: Benefit Accrued (contributes to)

○ Product Idea: Value Measured



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09

(Card 1 of 2)



Resources

- **Business Case:** An approach to quantifying ROI for incremental delivery is described in Software by Numbers by Mark Denne and Jane Cleland-Huang (Prentice Hall 2004).
- **Economic Model:** Using an economic model to ensure that priority and sequencing decisions are made consistently at all levels to maximize ROI across an organization is emphasised by Don Reinertsen in The Principles of Product Development Flow (Celeritas Publishing 2009).
- **Essence Standard (OMG),** refer to <http://www.omg.org/spec/Essence/1.1/>.
- **Ideas Kanban:** The use of a Kanban system for progressing high-level expressions of customer need is described by Dean Leffingwell in Agile Software Requirements (Addison-Wesley 2011).



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Resources

- **Product Management:** See p.42 of Dean Leffingwell's Agile Software Requirements (Addison-Wesley 2011) for a discussion of this role in large-scale agile versus, for example, a Scrum Product Owner.
- The Essence of Software Engineering: Applying the SEMAT Kernel, by Ivar Jacobson, Pan-Wei Ng, Paul E. McMahon, Ian Spence and Svante Lidman, Addison-Wesley 2013.



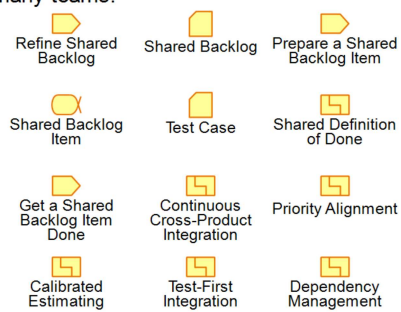
IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09

(Card 2 of 2)

Shared Backlog Essentials

Prioritize and marshal the delivery of product value requiring the collaboration of many teams.



Resources



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Refine Shared Backlog

Get and keep the Shared Backlog visible, up-to-date and in good working order, with high priority items agreed and well understood.

- Requirements: Conceived
- └─ Shared Backlog: (any level)

Understand the Requirements



- Requirements: Bounded or beyond
- └─ Shared Backlog: Items Prioritized
- └─ Shared Backlog Item: Identified (New Shared Product Backlog Items)



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Shared Backlog

An ordered list of valuable outcomes that requires the coordinated effort of many development teams.



Items Gathered

Items Prioritized

Cost-Benefit Quantified

Describes: ○ Requirements

Ref: Shared Backlog



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09

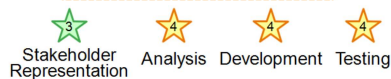


Prepare a Shared Backlog Item

Get a Shared Backlog Item ready for development by defining its constituent backlog items and how they will be integrated and tested.

- Shared Backlog Item: Identified

Understand the Requirements



- └─ Test Case: Test Ideas Captured (1 or more)
- └─ Shared Backlog Item: Subordinate Items Defined



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Shared Backlog Item

An independently releasable and valuable outcome that requires development contributions from many teams.



Identified

Subordinate Items Defined

Ready for Release

Relates to: ○ Requirements

Ref: Shared Backlog



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Test Case

Defines test inputs and expected results to help evaluate whether a specific aspect of the system works correctly.



Test Ideas Captured

Scripted

Automated

Describes: Requirements and Shared Backlog Item



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Shared Definition of Done

Have a clear definition of what it means for any Shared Backlog Item to be fully completed and ready for release. This should include the completion of each and every constituent backlog item according to the Definition of Done for these items, but it will also include completion of additional integration and quality assurance activities such as “*end-to-end*” testing and “*nearest neighbor*” testing.

Guides:

Get a Shared Backlog Item Done

Informs: Prepare a Shared Backlog Item

Relates To: Shared Backlog Item



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Get a Shared Backlog Item Done

Constituent backlog items are built, tested, integrated and it is verified that the Shared Backlog Item is correctly implemented.

Requirements: Bounded or beyond

Shared Backlog Item: Subordinate Items Defined

Implement the System Test the System



Stakeholder Representation



Analysis



Development



Testing

Test Case: Scripted

Shared Backlog Item: Ready for Release

Software System: Usable or beyond



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Continuous Cross-Product Integration

Code from different component products is continuously integrated, so that there is always a tested, working integrated build of the composite product that is being evolved to implement the Shared Backlog Items.

Demonstrate the integrated product as often as possible, e.g. at least once every development timebox and/or completion of each Shared Backlog Item.

Part Of:

Get a Shared Backlog Item Done



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Priority Alignment

To achieve the value associated with a Shared Backlog Item, all the constituent Product Backlog Items must be completed. The priority of doing this may need to be balanced with other items on the Product Backlogs of the different teams. This is an ongoing process of negotiation, e.g. between product owners or within a product management group. This is best managed within the context of an agreed economic prioritization model.

Part Of:

Get a Shared Backlog Item Done



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Calibrated Estimating

Where multiple teams collaborate to deliver a Shared Backlog Item, to estimate size and velocity we need to be able to compare like-with-like across different team estimates. This means we need a shared, calibrated measurement unit.

One way this can be achieved is by having a shared “Gold Standard” backlog item, with an agreed size, that all teams use to calibrate all their relative estimates.

One Way To Size: Get a Shared Backlog Item Done

Ref: Calibrated Estimating



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Test-First Integration

The constituent Product Backlog Items that need to be developed, tested and integrated in order to fully implement a Shared Backlog Item are specified by defining the tests that each one needs to pass for it to be considered “done”.

When these Product Backlog Item tests are passing it means the item is ready for integration and testing with the other constituent items.

Supports:

Get a Shared Backlog Item Done



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Dependency Management

When many teams are working to evolve a complex product, dependencies between teams need to be managed. For example, to complete a Product Backlog Item, a team may need another team’s component product to be extended. This can either be managed via the other team’s Product Backlog, or using a more open model where teams can change other teams’ products with guidance and review support from the “owning” team.

Part Of:

Get a Shared Backlog Item Done



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Resources

- **Calibrated Estimating:** Dean Leffingwell talks about the challenge of normalizing velocity in ch. 8 of his book Agile Software Requirements (Pearson Education Inc. 2011). Leffingwell suggests a "Hybrid Model" estimating approach for managing this challenge, as opposed to the Gold Standard model suggested in this practice.
- Essence Standard (OMG), refer to <http://www.omg.org/spec/Essence/1.1/>.



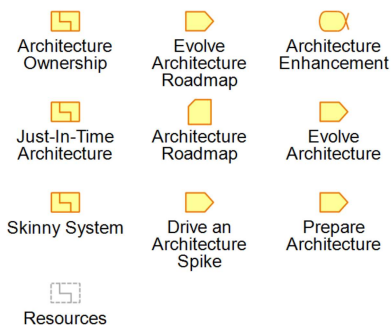
Resources

- **Shared Backlog:** The concept of a Product Backlog that is shared across multiple teams is a standard part of many scaled agile models and frameworks. Dean Leffingwell calls this a "Program Backlog" in his book Agile Software Requirements (Pearson Education Inc. 2011). Large-Scale Scrum simply calls it "the Product Backlog", as its role remains unchanged irrespective of how many teams are collaborating to deliver the product. (See for example Practices for Scaling Lead and Agile Development by Craig Larman and Bas Vodde (Addison-Wesley 2010)).
- The Essence of Software Engineering: Applying the SEMAT Kernel, by Ivar Jacobson, Pan-Wei Ng, Paul E. McMahon, Ian Spence and Svante Lidman, Addison-Wesley 2013.



Agile Architecture Essentials

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



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 individual architect or architecture team.

For big, complex or high-risk technical endeavors, dedicated Architecture Owners may well be needed to guide the evolution of the right technical solution, but they should work collaboratively with teams to reinforce collective responsibility and learning.

Owns: Architecture Roadmap

Ref: Architecture Ownership





Evolve Architecture Roadmap

Map out how the Software System will need to evolve over time, and continuously refine this thinking based on best current information.

○ Requirements

Shape the System



○ Software System: Architecture Selected

□ Architecture Roadmap: Outlined

○ Architecture Enhancement: Goals Articulated



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Architecture Enhancement

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



Goals Articulated

Approach Verified

Ready for Use

Relates to: ○ Software System



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Just-In-Time Architecture

Agile architecture specifically does *not* mean never thinking ahead beyond the current needs and targeted release.



It *does* mean keeping the current structure and approach as simple and cost effective as possible to achieve the next release goals, and then upgrade as needed to support future releases. This improves overall ROI and increases learning based on experience.

Guides: □ Evolve Architecture Roadmap

Shapes: □ Architecture Roadmap



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Architecture Roadmap

Communicates the planned enhancement of the architecture over time to support key releases and other milestones in the evolution of a Software System.



Outlined

Detailed

Describes: ○ Software System



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Evolve Architecture

The team and the stakeholders continuously think about, debate, question and evolve the architecture approach, based on learning.

☐ Software System: Architecture Selected



☐ Software System: Demonstrable (contributes to)



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Skinny System

Being agile means frequent demonstration and release of functionality and responding to feedback and learning.



A good architecture approach is to build first a deployable system that does close-to-nothing (aka "Hello World") and then extend this working Skinny System one Architecture Enhancement at a time, one backlog item at a time, and one test at a time.

Enables:

☐ Evolve Architecture Roadmap and

☐ Prepare Architecture

Ref: ☐ Skinny System



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

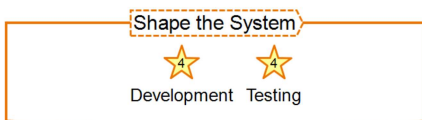
2018.09



Drive an Architecture Spike

An Architecture Enhancement is evaluated by testing it out.

☐ Software System: (any state)



☐ Software System: Architecture Selected (contributes to)

☐ Architecture Enhancement: Approach Verified



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09

Ref: ☐ Architecture Spike



Prepare Architecture

Implement an Architecture Enhancement to enable a Software System to respond to current or future needs.

☐ Software System: Architecture Selected



☐ Software System: Demonstrable (contributes to)

☐ Architecture Enhancement: Ready for Use



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Resources

- **Architecture Enhancement:** These are similar to Architecture Elements in the Incremental Funding Methodology described by Mark Denne and Jane Cleland-Huang in their book *Software by Numbers* (Prentice Hall 2004), and also to Architectural Epics as described in Dean Leffingwell's *Agile Software Requirements* (Addison-Wesley 2011).
- **Architecture Ownership:** See for example the Architecture Owner section on p.76-78 of Scott Ambler's book *Disciplined Agile Delivery* (IBM Press 2012).
- **Architecture Spike:** This concept is part of Extreme Programming (XP) – see for example <http://www.extremeprogramming.org/introduction.html>.
- **Essence Standard (OMG),** refer to <http://www.omg.org/spec/Essence/1.1/>
- **Skinny System:** This is the name given to an early version of the system that establishes an architectural baseline by Ivar Jacobson and Pan Wei Ng in their book *Aspect-Oriented Software Development with Use Cases* (Addison-Wesley 2004).



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Resources

- **The Essence of Software Engineering: Applying the SEMAT Kernel,** by Ivar Jacobson, Pan-Wei Ng, Paul E. McMahon, Ian Spence and Svante Lidman, Addison-Wesley 2013.

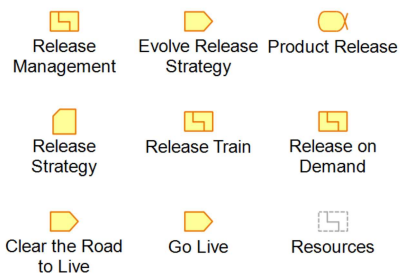


IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09

Release Management Essentials

Evolve and enact a strategy to enable releases to be made in a frequent, fast, safe and timely manner.



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Release Management

A single point of responsibility for ensuring that an optimized release strategy is evolved and executed.



Responsibilities of this individual or team include ensuring that:

- Good release governance checks are agreed, in line with the vision and release strategy
- The right investments are made to decrease delays and risks
- Each release is achieved in a smooth and timely fashion.

Responsible For:

- Evolve Release Strategy and
- Clear the Road to Live

Ref: Release Management



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09

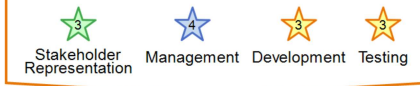


Evolve Release Strategy

Evolve a release strategy to maximize responsiveness to business need, enable value to be delivered as frequently as business constraints allow, and minimize delay and risk.

○ Software System

Shape the System



■ Release Strategy: Checkpoints Specified

■ Product Release: Release Checks Agreed



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Product Release

An increment of the product that is usable and adds new value.



Goals Established

Release Checks Agreed

Ready to Release

Released

Relates to: ○ Software System



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Release Strategy

Describes the planned releases (including internal, limited, full etc.), covering goals, timing and release process, including quality, security and other checks that are needed.



Goals and Targets Clear

Checkpoints Specified

Release Process Documented

Describes: ○ Software System



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Release Train

The Release Train metaphor involves frequent, regular releases, e.g. daily, or even many times daily.

New features that are ready can "catch the train", the rest are left behind, but can always "catch the next train".

The more often trains depart the better, so that business value spends less time on average queued up to wait for the next release.

Alternative To: ■ Release on Demand

Shapes: ■ Release Strategy

Ref: ■ Release Train



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Release on Demand

Make releases in response to business demand.

This might mean targeting a release within specified timescales, to support specific business events, legal deadlines, or other time-critical market or user needs.

This in turn might involve agreeing the Minimum Viable Product (MVP) needed to meet the time-critical needs, and tracking progress towards achieving it, while negotiating on scope to meet the release timescales.

Alternative To: Release Train

Helps To Shape: Release Strategy

Ref: Minimum Viable Product (MVP)



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Clear the Road to Live

Work to ensure that what is being developed can be released quickly and safely as soon as it is ready for release.



Software System: Ready

Product Release: Ready to Release



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09

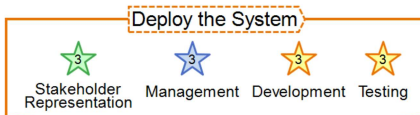


Go Live

A new live release is made, to give some or all users access to new or changed functionality, or to fix an issue with a previous release.

Software System: Ready

Product Release: Ready to Release



Software System: Operational

Product Release: Released



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Resources

(Card 1 of 2)

- Essence Standard (OMG), refer to <http://www.omg.org/spec/Essence/1.1/>
- **Minimum Viable Product (MVP)**: Defined by Frank Robinson, and popularized by Steve Blank and Eric Ries – see The Lean Startup by Eric Reis (Penguin 2011) and https://en.wikipedia.org/wiki/Minimum_viable_product.
- **Release Management**: A concept with a long history. Dean Leffingwell describes an agile approach to release management with a Release Management Team in p.73-74 of Agile Software Requirements (Addison-Wesley 2011).
- **Release Train**: This term for a series of regular releases is part of Henrik Kniberg's description of the Spotify Engineering Culture – <https://labs.spotify.com/2014/03/27/spotify-engineering-culture-part-1/>.



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Resources

- The Essence of Software Engineering: Applying the SEMAT Kernel, by Ivar Jacobson, Pan-Wei Ng, Paul E. McMahon, Ian Spence and Svante Lidman, Addison-Wesley 2013.



Development Support Essentials

Automate and operate environments to optimize the end-to-end delivery process.



Environment Team



Environment



Automate the Pipeline



Evolve the Environment



Manage Testing Debt



Support the Environment



Environment as Code



Resources



Environment Team

In a large-scale agile delivery endeavor, involving many independent development teams, significant investment is needed to ensure that the required tools, environments, etc. are available to the teams to enable them to effectively collaborate to evolve a large-scale, complex product.



The Environment Team is focused on ensuring that the development teams have access to these capabilities, tools and environments.

Responsible For:

- Evolve the Environment and
- Support the Environment

Ref: Environment Team



Environment

The environment includes all the “boxes”, tools, configurations, workspaces etc. needed to support code production and promotion (e.g. dev., test, staging, prod. etc.).



Specified

In Place

Working Well

Decommissioned

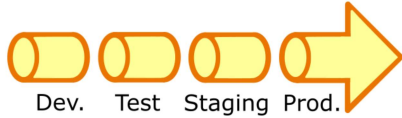
Relates to: Software System





Automate the Pipeline

To reduce lead-time, automate the development pipeline as much as possible. Define, exercise and refine the steps from making a single software change to the deployment of the software system into production. Seek to automate every single step through the environments.



Part Of: Evolve the Environment

Ref: Automate the Pipeline



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Evolve the Environment

Evolve the environment to support the development, test, and deployment of the software system, including specifying, setting up, making it available to teams, and ensuring they can access and use it.

Software System

Environment



Software System: Ready (contributes to)

Environment: In Place



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Manage Testing Debt

Each team's "Definition of Done" should, ideally, cover all types of testing. In practice teams simply may not have the resources to run tests such as stress, performance and reliability as often as ideal.

The running of deferred tests needs to be planned for and supported, so that they can be run as frequently as possible. Deferred tests are technical debt, and should be reduced over time so that they do become part of the "Definition of Done".

Part Of: Evolve the Environment

Relates To: Software System



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Support the Environment

Keep the environment running, help teams to use it well, and optimize and refine its configuration and usage, based on experience and feedback.

Software System

Environment: In Place



Software System: Operational (contributes to)

Environment: Working Well or beyond



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Environment as Code

Scripts and configuration parameters used in automation should be treated like code. They need to be version-controlled like any other piece of code in the software system. This ensures that promotion and deployment is deterministic and repeatable.

Also look to parameterize database content to ensure all test conditions are re-creatable and all tests are repeatable.

Part Of: Evolve the Environment and

Support the Environment

Ref: Environment as Code



2018.09

(Card 1 of 2)



Resources

- **Automate the Pipeline:** See the "Automate Almost Everything" principle on p.25 of Continuous Delivery by Jez Humble and David Farley (Addison-Wesley 2011).
- **Environment as Code:** See the "Keep Everything in Version Control" principle on p.26 of Continuous Delivery by Jez Humble and David Farley (Addison-Wesley 2011).
- **Environment Team:** This concept is similar to what Henrik Kniberg calls an **Infrastructure Squad** (see for example <https://labs.spotify.com/2014/03/27/spotify-engineering-culture-part-1/>), and what Dean Leffingwell calls a **System Team**, see p.71-73 of Agile Software Requirements (Addison-Wesley 2011).
- Essence Standard (OMG), refer to <http://www.omg.org/spec/Essence/1.1/>



2018.09



Resources

- The Essence of Software Engineering: Applying the SEMAT Kernel, by Ivar Jacobson, Pan-Wei Ng, Paul E. McMahon, Ian Spence and Svante Lidman, Addison-Wesley 2013.



2018.09

DevOps Essentials

Dev and Ops work together to respond to customer needs and transition make live releases frequently, smoothly and safely.

Joint Rapid Response Team

Monitor and Respond

Operations

Production Issue

Design for DevOps

End-to-End Optimization

Master Deployment through Repetition

Blameless Post-Mortem

Resources



2018.09



Joint Rapid Response Team

Dev and Ops team members work together as a rapid response team to fix Production Issues.



The team is responsible for all aspects of the end to end process, from diagnosis, through solution design, development and test to deployment of the code that fixes the issue.

Such a team can either be formed on an as-needs basis, or as a permanently on-call team, optionally with rotating membership.

Performs: Monitor and Respond



IVAR JACOBSON
INTERNATIONAL
Generated by IUI Practice Workbench™

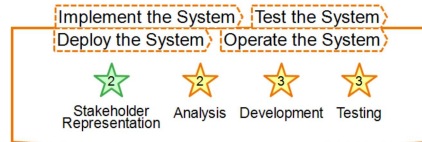
2018.09



Monitor and Respond

Focus on the rapid detection and resolution of any and all sources of user dissatisfaction with the operational system.

Software System: Operational



Software System: Operational

Production Issue: Resolved



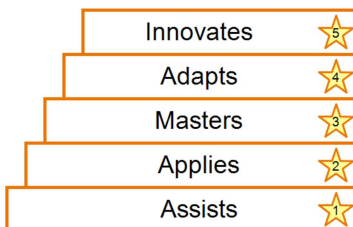
IVAR JACOBSON
INTERNATIONAL
Generated by IUI Practice Workbench™

2018.09



Operations

The ability to operate software systems successfully, including monitoring and maintaining systems to maximize availability and performance, and to make new deployments safely and without operational disruption.



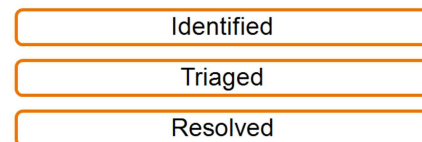
IVAR JACOBSON
INTERNATIONAL
Generated by IUI Practice Workbench™

2018.09



Production Issue

Any production system issue or potential enhancement that is detected through operational use, monitoring or end-user feedback.



Relates to: Software System



IVAR JACOBSON
INTERNATIONAL
Generated by IUI Practice Workbench™


2018.09



Design for DevOps

Deployability and operability are made key architectural concerns:

- Involve Ops in the design process
- Design-in hot-deploy capability
- Reduce the blast area through modular design to limit and localize the impact of a failure
- Design for progressive release, e.g. internal then progressively to ever-larger user populations
- Layer the system (platform, services, apps), and increase the scope of DevOps progressively over time, layer by layer.

Enables:  Monitor and Respond



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09





End-to-End Optimization

Use “*systems thinking*” to measure and optimize the full “*end-to-end*” process, e.g. from detection of need, to deployed change that meets the need.

Avoid local measures and blinkered change initiatives, e.g. focused on reducing effort to develop without considering deployment and operations.

Ensure failure demand (including responding to production issues) is measured, and *systems thinking* employed to minimize it.

Part Of:  Monitor and Respond

Ref:  End-to-End Optimization



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09





Master Deployment through Repetition

If deployment is costly and painful it is done less frequently, which makes it more costly and more painful.

To turn this vicious circle into a virtuous one, set hard goals to deploy more and more frequently - e.g. from once per week to multiple times per day.

This can only be achieved by investing in automation, version control etc., and through close collaboration between Dev and Ops.

Part Of:  Monitor and Respond

Ref:  Master Deployment through Repetition



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™


2018.09




Blameless Post-Mortem

Sometimes things inevitably do go wrong. “Things going wrong” therefore is not a matter for blame. Failure to learn from things going wrong, on the other hand, should be a matter of universal shame and embarrassment.

The key is to have an open, all-inclusive, post-mortem after every incident or deployment, and agree specific changes to improve the process in future and reduce the risk of repetition of similar issues.

Part Of:  Monitor and Respond

Ref:  Blameless Post-Mortem



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Resources

- **Blameless Post-Mortem:** See "DevOps Patterns Area 2: Create Production feedback into Development" in The Top 11 Things You Need To Know About DevOps v1.0 by Gene Kim (itrevolution.com).
- **DevOps:** The term "DevOps" was popularized through a series of "DevOps Days" starting in 2009 in Belgium. (<https://en.wikipedia.org/wiki/DevOps>).
- **End-to-End Optimization:** See for example "The First Way" in Section 5: "Systems Thinking" in The Top 11 Things You Need To Know About DevOps v1.0 by Gene Kim (itrevolution.com).
- **Essence Standard (OMG),** refer to <http://www.omg.org/spec/Essence/1.1/>.
- **Master Deployment through Repetition:** See for example Principles of Software Delivery: "Create a Repeatable, Reliable Process for Releasing Software" and "If it Hurts, Do It More Frequently and Bring the Pain Forward" p.24-27 of Continuous Delivery by Jez Humble and David Farley (Addison-Wesley 2011).



2018.09



Resources

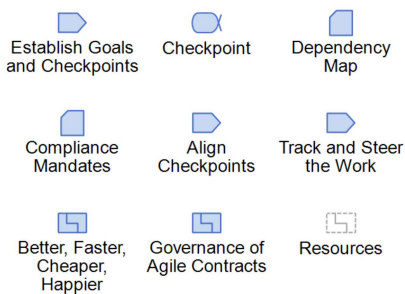
- The Essence of Software Engineering: Applying the SEMAT Kernel, by Ivar Jacobson, Pan-Wei Ng, Paul E. McMahon, Ian Spence and Svante Lidman, Addison-Wesley 2013.



2018.09

Agile Governance Essentials

Targeted and account for funding and ensure compliance with regulations, processes and procedures.

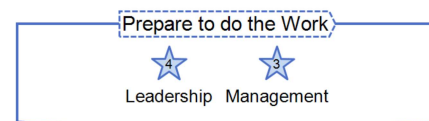


2018.09



Establish Goals and Checkpoints

Goals and checkpoints are agreed, focusing on valuable outcomes and critical compliance needs and constraints.



- Work: Prepared (contributes to)
 - Dependency Map: Checkpoints Defined
 - Checkpoint: Defined (1 or more)
- Way of Working: Principles Established (contributes to)
 - Compliance Mandates: Policies Defined



2018.09



Checkpoint

A point in time at which a meaningful and measurable progression is targeted to have been achieved, including value release points and quality gates.



Defined

Aligned

Scheduled

Achieved

Relates to: Work



IVAR JACOBSON
INTERNATIONAL
Generated by IUI Practice Workbench™

2018.09



Dependency Map

Shows key time-critical checkpoints and milestones that must be achieved for the work to be on track, and shows dependencies that exist between them.



Checkpoints Defined

Dependencies Identified

Timelines Captured

Describes: Work



IVAR JACOBSON
INTERNATIONAL
Generated by IUI Practice Workbench™

2018.09



Compliance Mandates

What compliance rules and constraints must be met, and how the work must be conducted in order to ensure compliance.



Checks Defined

Policies Defined

Processes Documented

Describes: Way of Working



IVAR JACOBSON
INTERNATIONAL
Generated by IUI Practice Workbench™

2018.09



Align Checkpoints

Milestones and other checkpoints are aligned and the goals and dependencies are identified.

Dependency Map: Checkpoints Defined

Checkpoint: Defined (1 or more)

Way of Working: Principles Established

Compliance Mandates: Policies Defined

Coordinate Activity



Leadership Management

Work: Under Control (contributes to)

Dependency Map: Dependencies Identified

Checkpoint: Aligned (1 or more)



IVAR JACOBSON
INTERNATIONAL
Generated by IUI Practice Workbench™

2018.09



Track and Steer the Work

Track progress towards achieving checkpoints and goals. Adjust approach, plans and goals based on metrics data, feedback and learning.

Work: Started

Dependency Map: Dependencies Identified

Checkpoint: Aligned (1 or more)



Work: Under Control

Checkpoint: Achieved (1 or more)



IVAR JACOBSON
INTERNATIONAL
Generated by IJI Practice Workbench™

2018.09



Better, Faster, Cheaper, Happier

Have clear and simple metrics that measure what really matters, including value delivered / accrued for investments made.

For agile delivery organizations a powerful balanced scorecard covers:

- Better – e.g. improvements in service or product quality
- Faster – e.g. lead time to value
- Cheaper – e.g. cost of ownership
- Happier – e.g. improved levels of satisfaction of customers, employees and stakeholders.

Influences:

Establish Goals and Checkpoints

Supports: Track and Steer the Work

Ref: Better, Faster, Cheaper, Happier



IVAR JACOBSON
INTERNATIONAL
Generated by IJI Practice Workbench™

2018.09



Governance of Agile Contracts

When work is outsourced, the contract must enable agile delivery, and governance should focus on:



- Ensuring frequent delivery and acceptance of releasable product (e.g. every 2–4 weeks)
- Tracking achievement of meaningful and valuable outcomes and milestones, not delivery of fixed product scope
- Responsiveness to change to maximize ROI
- Stopping the work when goals are met and ROI is diminishing.

Impacts: Track and Steer the Work and

Establish Goals and Checkpoints

Ref: Governance of Agile Contracts



IVAR JACOBSON
INTERNATIONAL
Generated by IJI Practice Workbench™

2018.09



Resources

- **Better, Faster, Cheaper, Happier:** See http://www.ivarjacobson.com/sustainable_change/ and <https://www.ivarjacobson.com/publications/case-studies/kpn-case-study>.
- Essence Standard (OMG), refer to <http://www.omg.org/spec/Essence/1.1/>
- **Governance of Agile Contracts:** See for example Agile Contracts by Andreas Opelt et al. (Wiley-Blackwell 2013).
- The Essence of Software Engineering: Applying the SEMAT Kernel, by Ivar Jacobson, Pan-Wei Ng, Paul E. McMahon, Ian Spence and Svante Lidman, Addison-Wesley 2013.



IVAR JACOBSON
INTERNATIONAL
Generated by IJI Practice Workbench™

2018.09

Team Of Teams Essentials

Achieve a collectively high-performing team of collaborating teams.



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Collaboration Lead

An experienced agile coaching capability that is focused on ensuring that teams-of-teams can and do collaborate and communicate successfully to achieve shared goals, and that effective practices and capabilities are cross-pollinated through capability sharing. A Collaboration Lead will typically facilitate all team-of-teams events, meetings and communications to ensure that they happen, are well prepared for, and are successful.



Facilitates:

- Practice Cross-Pollination and
- Evolve Team Organization



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Evolve Team Organization

The teams agree how they should be organized to minimize dependencies and collaborate effectively when working on tasks that cut across teams.

☐ Team

Prepare to do the Work



Leadership Management

☐ Team: Formed or beyond (contributes to)

☐ Way of Working: Principles Established (contributes to)



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09



Feature Team

Teams are aligned to customer value, and every team can and does build, test and make release-ready new user-facing value. In the ideal model, every team has the capability and knowledge required to make changes anywhere within the overall system architecture as needed to implement new customer value.

Mature configuration management and integration capabilities are needed to enable many such teams to operate in parallel safely and effectively.

Guides: ☐ Evolve Team Organization

Ref: ☐ Feature Teams



IVAR JACOBSON
INTERNATIONAL
Generated by UI Practice Workbench™

2018.09




Component Team

Teams are organized primarily around top-level components of the overall system architecture. This works well when a software system is relatively new to build up the core capabilities of each component.

The down-side with this approach is that the work of the different teams typically needs to be integrated and tested before value can be released, which can increase time-to-value, and means that delivery teams can lose direct contact with customer value realization.

Guides:  Evolve Team Organization

Ref:  Component Teams




2018.09




Practice Cross-Pollination


Teams share lessons-learned and other knowledge, including good and bad experiences with tools and practices, and aim to cross-skill using events such as coding dojos.

 Team

 Way of Working



 Team: Performing (contributes to)

 Way of Working: Working Well (contributes to)

Ref:  Coding Dojo



2018.09




Team-of-Teams Get-Together


Have a regular, co-located event with as full attendance as possible from all the collaborating teams, e.g. 2 or 3 days in duration, every 2 or 3 months.

The event can be used for review, planning and related activities, such as team-of-team retrospectives. It can also include innovation activities such as "hackathons".

This enables all these activities to be prepared, organized and facilitated together in one place and on a regular basis.

One Approach To:

 Practice Cross-Pollination

Ref:  Team-of-teams Get-Together




2018.09




Scrum-of-Scrums

An effective way to ensure ongoing communication and collaboration across many teams is a short, frequent regular meeting of representatives from all the teams.

This is the team-of-teams equivalent of the team-level whole-team Daily Stand-Up meeting (also known as a "Daily Scrum"). It is less frequent (e.g. twice a week), and involves e.g. 1-3 people from each team – either team leads / "Scrum Masters" and/or a rolling roster of interested team members

Supports:  Practice Cross-Pollination

Ref:  Scrum-of-Scrums



2018.09



Resources

- **Coding Dojo:** An in-house variant of a CoderDojo - see <https://en.wikipedia.org/wiki/CoderDojo>.
- **Component Teams:** See "Conway's Law" pattern in Organizational Patterns of Agile Software Development by James Coplein and Neil Harrison (Prentice Hall 2007) and Ch.7 of Scaling Lean and Agile Development by Craig Larman and Bas Vodde (Addison-Wesley 2009).
- **Essence Standard (OMG),** refer to <http://www.omg.org/spec/Essence/1.1/>.
- **Feature Teams:** See Ch.7 of Scaling Lean and Agile Development by Craig Larman and Bas Vodde (Addison-Wesley 2009).
- **Scrum-of-Scrums:** Described by Jeff Sutherland in Agile Can Scale: Inventing and Reinventing Scrum in Five Companies - see <https://www.agilealliance.org/glossary/scrum-of-scrums/>.



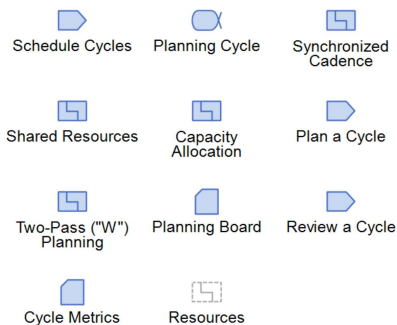
Resources

- **Team-of-teams Get-Together:** See Spotify "Hack Weeks" as described in Spotify Engineering Culture - Part 2 (<https://vimeo.com/94950270> Henrik Kniberg April 2014).
- **The Essence of Software Engineering: Applying the SEMAT Kernel,** by Ivar Jacobson, Pan-Wei Ng, Paul E. McMahon, Ian Spence and Svante Lidman, Addison-Wesley 2013.



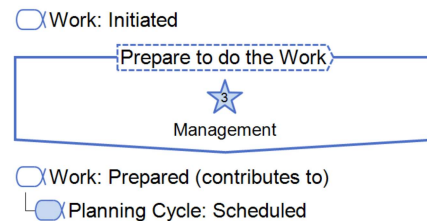
Periodic Alignment Essentials

Align work priorities and plans of many teams to overall goals using synchronized planning cadences.



Schedule Cycles

Book cycles and related events in the calendar. Cycles are typically longer than team-level timeboxes (e.g. 2 or 3 months rather than 2 or 3 weeks), and related meetings and events also longer and larger in equal measure.





Planning Cycle

A fixed time period focused on building and delivering value that is used to coordinate and align multiple, smaller team-level timeboxes.



Scheduled

Planned

Under Control

Reviewed

Relates to: Work

Ref: Planning Cycle



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Synchronized Cadence

An effective way to manage work across many interdependent teams is to synchronize team-level timebox "heartbeats" within a larger regular cross-team cadence "Cycle". For example, each team might have a 3-week synchronized timebox heartbeat, within a larger 12-week cadence cycle. This enables ongoing intra-team alignment (every team timebox) and periodic inter-team alignment around shared objectives (every larger Cycle).

Relates To: Planning Cycle

Ref: Synchronized Cadence



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Shared Resources

Decentralizing resources (placing all people and capability into separate development teams) does not always maximize the flow of value through a delivery channel.

Centralizing scarce resources can help smooth flow and reduce queues and lead time, provided there is enough capacity for them not to be bottlenecks during demand peaks.

Examples of specialist skills that might be centralized include user experience, security experts etc.

Impacts: Plan a Cycle

Ref: Shared Resources



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Capacity Allocation

Allocate capacities to different kinds of work (e.g. new features, architecture enhancements, refactoring, defect fixing, usability improvements etc.) to ensure that each one gets a reasonable level of attention.

This enables the software system to be evolved in a balanced way, without incurring excessive levels of technical debt, for example.

Allocations should be continuously monitored and periodically adjusted to respond to changing priorities.

Part Of: Plan a Cycle

Ref: Capacity Allocation



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Plan a Cycle

Agree on a viable plan for the cycle with the delivery teams. This is best achieved as a cross-team event, as it requires alignment of overall objectives with team-level plans.

- ☐ Work: Initiated or beyond
- ☐ Planning Cycle: Scheduled



- ☐ Work: Started
- ☐ Planning Cycle: Planned
- ☐ Planning Board: Dependencies Visible



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Two-Pass ("W") Planning

To Plan a Cycle use a two-pass approach:

1. From shared Objectives to team Plans, then ...
2. Align plans across teams, Adjust team plans, achieve shared Commitment.



An Approach To: ☐ Plan a Cycle

Ref: ☐ Two-Pass ("W") Planning



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Planning Board

Shows key milestones and timeboxes for the current Planning Cycle, contributing teams (e.g. as horizontal swim-lanes) and key cross-team dependencies.



Cycle Objectives Visible

Timebox Objectives Visible

Dependencies Visible

Team Work Plans Visible

Describes: ☐ Planning Cycle



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Review a Cycle

Review the outcomes of a Planning Cycle as inputs for adjusting plans and continuous improvement.

- ☐ Work: Started or beyond
- ☐ Planning Cycle: Under Control



- ☐ Work: Under Control or beyond (contributes to)
- ☐ Cycle Metrics: Results Analyzed
- ☐ Planning Cycle: Reviewed



IVAR JACOBSON
INTERNATIONAL
Generated by IJ Practice Workbench™

2018.09



Cycle Metrics

Communicates the effectiveness of each planning cycle as inputs for improvement.



Metrics Defined

Data Collected

Results Analyzed

Trends Analyzed

Describes: Work



IVAR JACOBSON
INTERNATIONAL
Generated by IJI Practice Workbench™

2018.09



Resources

- **Capacity Allocation:** See for example "Allocating Capacity According to Demand" section in David Andersen's book Kanban (Blue Hole Press 2010).
- **Essence Standard (OMG),** refer to <http://www.omg.org/spec/Essence/1.1/>
- **Planning Cycle:** See for example "Principles of the Agile Release Train" on p.303 of Agile Software Requirements by Dean Leffingwell (Addison-Wesley 2011).
- **Shared Resources:** See principle "F29: The Principle of Resource Centralization: Correctly managed, centralized resources can reduce queues" p.206-208 of The Principles of Product Development Flow by Donald Reinertsen (Celeritas Publishing 2009).
- **Synchronized Cadence:** See The Principles of Product Development Flow by Donald Reinertsen, sections on Cadence (p.177-186) and Synchronization (p.186-191) (Celeritas Publishing 2009).
- The Essence of Software Engineering: Applying the SEMAT Kernel, by Ivar Jacobson, Pan-Wei Ng, Paul E. McMahon, Ian Spence and Svante Lidman, Addison-Wesley 2013.



IVAR JACOBSON
INTERNATIONAL
Generated by IJI Practice Workbench™

2018.09



Resources

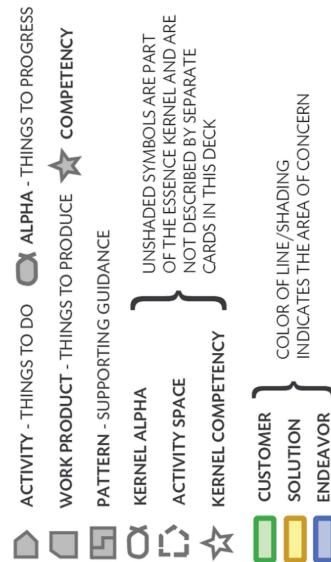
- **Two-Pass ("W") Planning:** See for example "Multiple Team Release Planning", p.153 of Scaling Software Agility by Dean Leffingwell (Addison-Wesley 2007) and "Release Planning" - ch.16 of Agile Software Requirements by Dean Leffingwell (Addison-Wesley 2011).



IVAR JACOBSON
INTERNATIONAL
Generated by IJI Practice Workbench™

2018.09

KEY OR LEGEND



Card Pack Contents Copyright Statements

© 2018 Ivar Jacobson International SA. All rights reserved.

Ivar Jacobson and IJI Practice Workbench are trademarks or registered trademarks of Ivar Jacobson International SA and/or its subsidiaries.

The OMG and Essence content included is provided under license by the following:

Copyright © 2013-2018 Data Access Technologies (Model Driven Solutions), Florida Atlantic University, Fujitsu, Fujitsu Services, Ivar Jacobson International AB, KTH Royal Institute of Technology, Metamaxim Ltd., PEM Systems, Stiftelsen SINTEF, and University of Duisburg-Essen and © 1997-2018 Object Management Group.

Use of Essence – Kernel and Language for Software Engineering Methods Specification is subject to the Terms, Conditions & Notices found at <https://www.omg.org/spec/Essence/1.1>

The alpha state checkpoint names included in the IJI Essence Kernel are provided under license by SEMAT Inc. under the Creative Commons Attribution CC International Public License:

<https://creativecommons.org/licenses/by/4.0/legalcode>
Copyright © SEMAT Inc. All rights reserved.

MOVE FROM PROCESS TO PRACTICES

- Get started with IJI's Library of Practices
- Ready to go practices to get you up and running quickly

[ivarjacobson.com/
practicelibrary](https://ivarjacobson.com/practicelibrary)

IJI PRACTICE WORKBENCH

- Compose and publish your own methods with IJI's Practice Workbench
- Create print-ready cards or export to HTML

[ivarjacobson.com/
practice-workbench](https://ivarjacobson.com/practice-workbench)

IJI SUSTAINABLE CHANGE FRAMEWORK

- Achieve consistent and lasting adoption of your agile techniques across your broad communities of practices!
- IJI's Framework and Consulting Services can help guide you.

[ivarjacobson.com/
sustainable-agile-transformation](https://ivarjacobson.com/sustainable-agile-transformation)