Practical Guide to Scrum

by Pavel Dabrytski

IQbusiness
consulting | research | contracting

Huge thank you to Ayanda Mkize, Biase De Gregorio and Hannah Ward for reviewing the guide and leaving me hundreds of review notes!

version 2.0. 7 April 2014.

Agile

Agile Manifesto and Scrum Values are the heart of any agile implementation

Agile Manifesto

Individuals and interactions	over	Process and tools
Working software	over	Comprehensive documentation
Customer collaboration	over	Contract negotiation
Responding to change	over	Following a plan

Keys to a successful Scrum team

- ◊ Roles of scrum master and product owner are performed by the right people
- ◊ Scrum master and product owner have enough time to perform their duties
- ◊ Development team is co-located
- ◊ Members of the development team are dedicated to that team (100% allocated)
- ◊ Development team is cross-functional
- ◊ Development team is self-organised

Definition of Agility

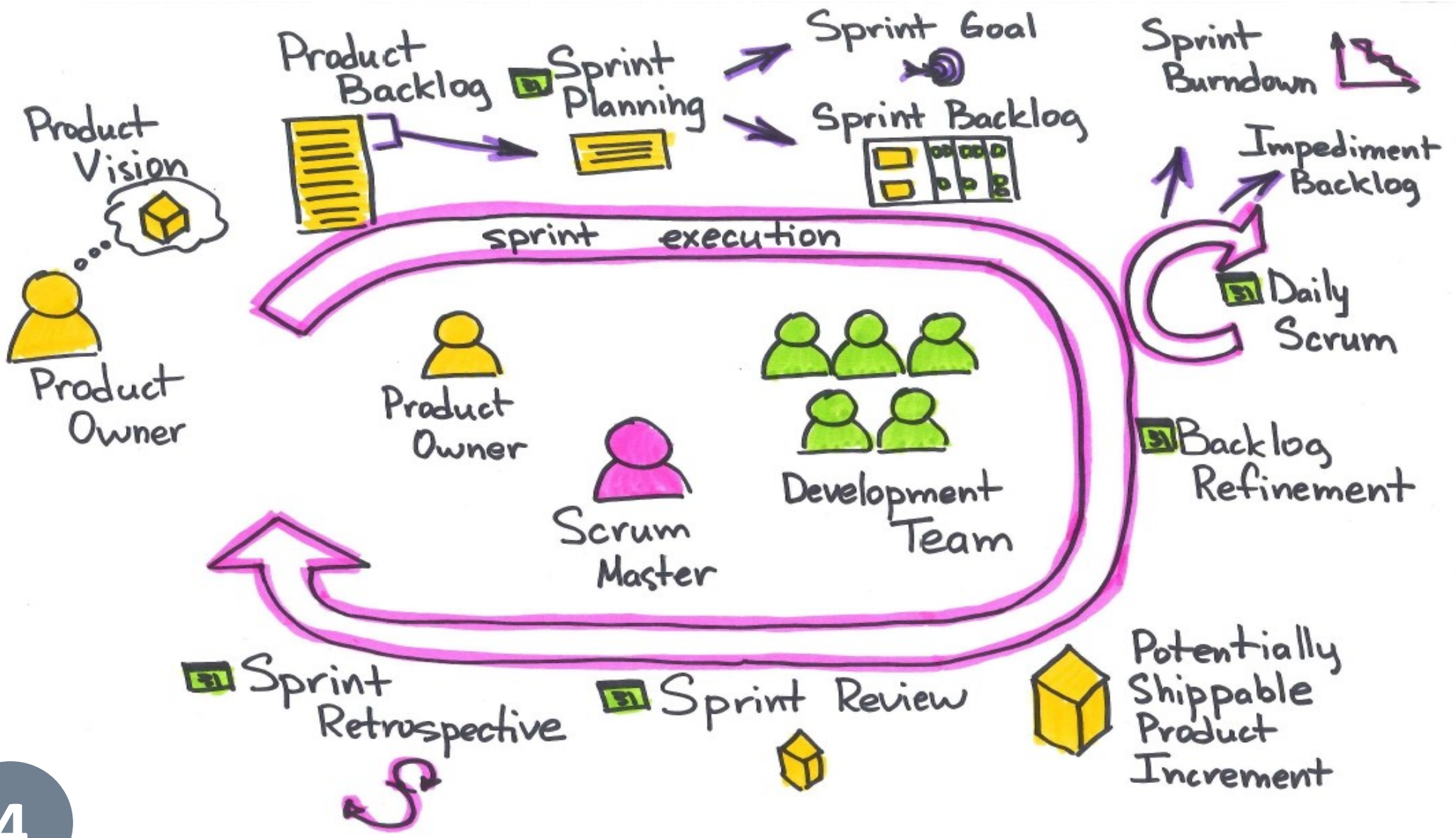
"Agility is the continued readiness "to rapidly or inherently create change, proactively or reactively embrace change, and learn from change while contributing to perceived customer value (economy, quality, and simplicity), through its collective components and relationships with its environment." Conby 2009.

Agile principles

1. Early and continuous delivery of valuable software
2. Welcome changing requirements
3. Deliver working software frequently
4. Business people and developers must work together
5. Motivated individuals produce the best results
6. Face-to-face conversation is the most valuable
7. Progress is measured by working software
8. Enforce a sustainable development pace
9. Technical excellence enhances agility (and quality)
10. Simplicity
11. Self-organising teams produce the best solutions
12. At regular intervals, the team adjusts its behavior to become more effective

Scrum Framework

This picture represent the full Scrum process



Product Vision

Product vision is a short statement which describes end goals, objectives and benefits of the product

WHEN First thing. Before product Backlog.

WHY Product vision is needed to ensure the product is moving in the right direction, strategies are aligned and that the development team spends its time creating the right product.

! Elevator test

"Can you explain your product in the time it takes to ride up in an elevator?" Moore (2006, p. 152). Passing this test ensures that your product vision is clear, engaging, and brief.

Moore's product vision model

FOR: «target customer»

WHO: «needs»

THE: «product name»

IS A: «product category»

THAT: «product benefit. Reason to buy»

UNLIKE: «competitors»

OUR PRODUCT: «differentiation or value proposition».

Example

FOR a mid-sized company's marketing and sales departments

WHO need basic CRM functionality,

THE CRM-Innovator

IS A web-based service

THAT provides sales tracking, lead generation, and sales representative support features that improve customer relationships at critical touch points.

UNLIKE other services or package software products,

OUR PRODUCT provides very capable services at a moderate cost.

? Who owns vision?

Product owner. However everyone contributes towards the product vision.

? Can vision be updated?

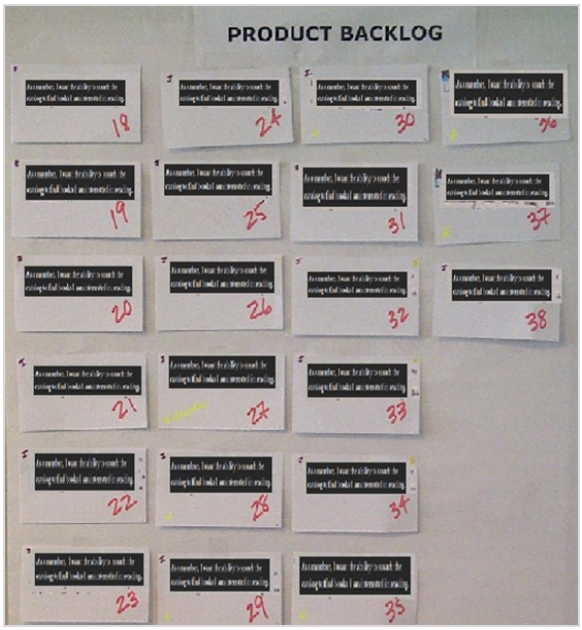
Absolutely! Product vision should reflect current business conditions (market, budget, capacity etc.). However, constantly changing vision is an indication of a problem.

? Who updates vision?

Product owner together with stakeholders and the team.

Product Backlog

Product backlog is a single list of features/requirements for the product, prioritised by value

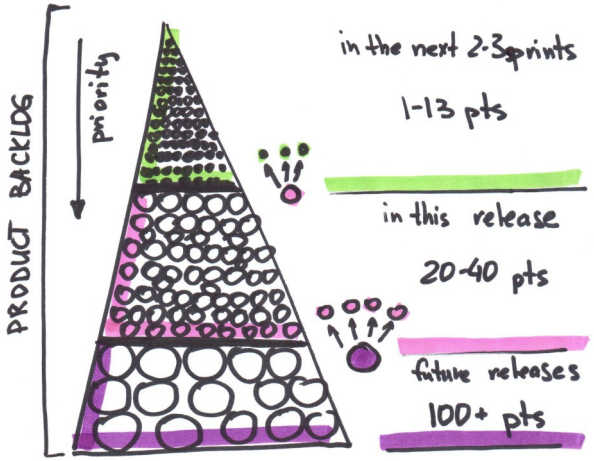


Product backlog

- Contains features, defects, technical work, knowledge acquisition
- Ordered by priority
- Prioritised by product owner with help from the development team and stakeholders

Backlog iceberg

Backlog Iceberg is an Agile *just-in-time* technique. User stories in Product Backlog are stored in different levels of detail where highest priority (top of the iceberg) stories are most detailed.



MoSCoW prioritization

Must have—all "must have" stories form the minimal viable product (MVP) and good enough for the first release

Should have—all "should have" stories make this product competitive

Could Have—all "could have" stories delight the customer

Won't Have—All "won't have" stories aren't worth doing. The majority of stories (65%) should fall into the "won't have" pile

? Multiple projects

If team works on multiple projects or products at the same time, we suggest creating a single team backlog. The product owner must prioritize work for multiple projects within this backlog. It will help the team to keep focused and will save time on scrum meetings.

! Tools for product backlog

- Physical wall, white board, index cards and post-it notes (recommended)
- Excel (most widely used)
- Any other electronic tools such as TFS, VersionOne, Pivotal Tracker, etc.

! Prioritization techniques

MoSCow is only one of many. You may look at Kano, Buy-a-feature, The Product tree and others.

User Story & Acceptance Criteria

User story is an agile technique to facilitate requirement management

Acceptance criteria—checklist which defines acceptance testing for this particular story

Story Formats

[Beginner] - Simple statement of functionality which needs to be build.

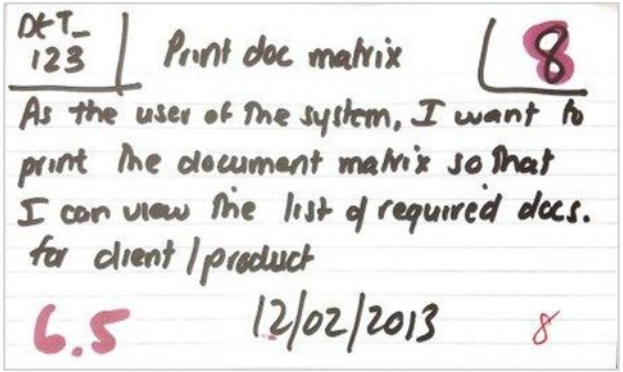
example: 'View email history.'

[Classic] - Connextra format: As a <user role>, I want <goal/desire> so that <benefit>.

example: 'As a senior support agent I want to view the emails send to the customer, so I know which communication took place.'

[Advanced] - In order to <receive benefit> as a <user role> I want <goal/desire>.

example: 'In order to be able to assist customer promptly, as a senior support agent I want to view previous customer email communication on my agent dashboard.'



! Wrong story format

'As a BA I will document price history'

- This user story is missing the actual user role. It is impossible to identify the person or role who will benefit from the user story
- This user story doesn't explain the reason why completing it is valuable and beneficial.

Acceptance Criteria Format

[Beginner] - Simple description on how functionality should work.

example: 'Only show last 2 emails sent to the customer' 'When pressing "resend email", show example of email sent'

[Advanced] - Given <precondition> When <scenario> Then <expected result>.

example: 'Given that customer didn't receive any emails yet, when agent opens dashboard, then 'no previous emails' message is displayed.'

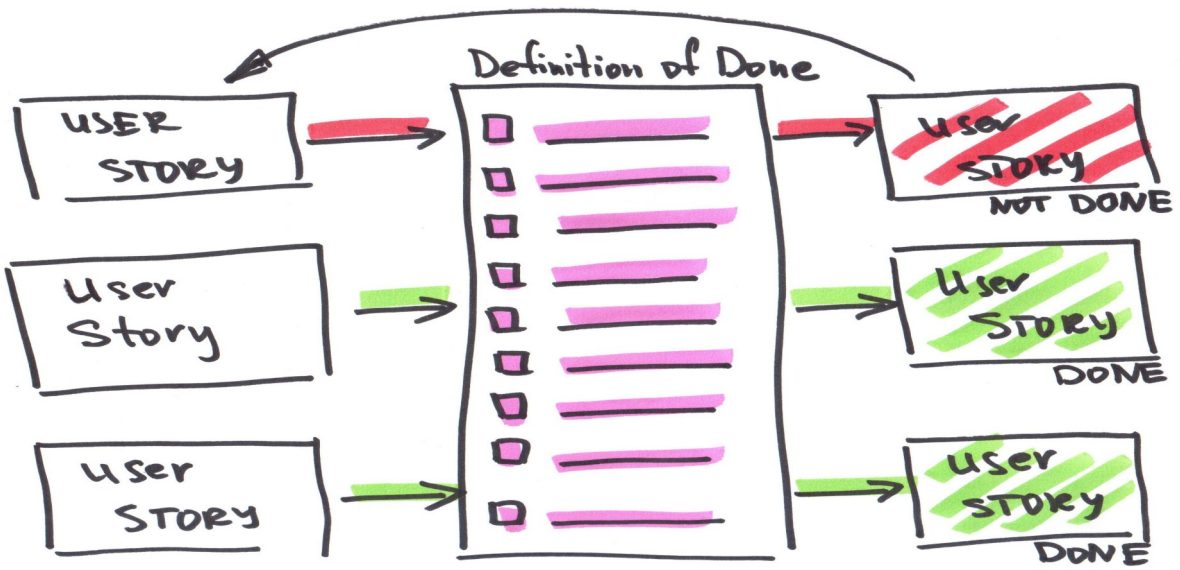
! Definition of Done vs Acceptance criteria

Definition of done is generic and applicable to all stories.

Acceptance criteria is specific and is different for different user stories.

Definition of Done

Definition of done is a checklist of valuable activities required to produce complete software



Example

- ◊ All development has been completed
- ◊ Functionality has been tested by developer
- ◊ Unit tests have been completed
- ◊ New business functionality satisfies acceptance criteria in TFS*
- ◊ All features have been tested in IE8** and IE9**
- ◊ Regression tested in IE8** & IE9** test environment
- ◊ Code has been reviewed by another developer
- ◊ Story has been reviewed by product owner, and product owner has accepted all open issues, if any

! DoD per team

Definition of done is unique per team, although it might contain some elements which are required by the department, organisation or industry.

? Can DoD change?

Definition of done may change over time as the team continues to build the product and learn from the process. Usually definition of done is reviewed by the whole team at one of the sprint events (i.e. sprint retrospective or sprint planning).

! Definition of done vs acceptance criteria

Definition of done is generic and applicable to all stories.

Acceptance criteria is specific and is different for different user stories.

*- Team Foundation Server

** - Internet Explorer

Relative Estimation

Relative estimation is a forecasting technique with a story point as a unit of measure

Planning poker

Planning poker is a consensus-based technique for estimation. It helps avoiding the influence of the other participants. Here is the procedure:

- ◊ User story is selected
- ◊ Product owner (or scrum master) briefly explains the story and answers any related questions (2mins)
- ◊ Voting occurs (1st round)
- ◊ If the estimates show a large variation, a further 2 mins should be used to explain why the estimates should be higher or lower
- ◊ Voting occurs (2nd round)
- ◊ If the estimates still show a large variation, a further 2 mins should be used to re-explain/justify the estimates
- ◊ Voting occurs (3rd round)
- ◊ If, after the 3rd round of voting, no consensus can be reached regarding the size, the story is placed on hold

Baseline

When estimating for the first time, select the smallest story and assign it 3 points. It will become your baseline.

Why relative estimation?

- ◊ Estimation in relative points has proven to be quicker
- ◊ It removes link between estimating effort and committing to timelines
- ◊ It helps to remove false accuracy around estimates

Time vs points

Estimation in relative points has proven to be quicker. Also estimation in hours is prone to further inaccuracies because it is based on ideal hours.

How much is 1 point?

A story point cannot be directly converted into hours. It only shows how much smaller or bigger the item is compared to other items in the backlog.
It can be calculated of course, but such information should be used for release planning only (i.e. velocity), and not per story or task .

Modified Fibonacci

Scale is **1 2 3 5 8 13 20 40 100**

What NOT to do

- ◊ Don't allow anyone to shout estimation aloud before the team estimates together
- ◊ Don't allow anyone in the team or any stakeholder to override teams decisions (i.e. 'come on guys, it is not 40, it is a 3')
- ◊ Don't take an average of estimates without having discussions
- ◊ Don't allow people not to vote. Asking everyone to vote will improve the team's understanding of each other's work
- ◊ Don't assign story points by yourself. It is always a team discussion

Comparing teams

Velocities and estimates of different teams are not directly comparable due to different baselines/team norms.

Sprint

Sprint is a time-boxed event with a goal of producing working product at the end of it

DOs

- Consistent duration throughout life cycle of the project
- New sprint starts immediately after the conclusion of the previous sprint
- Schedule recurring meetings for all scrum events in team diaries
- Create sprint goals for each sprint

DONTs

- Extend/shorten sprint after it commences
- Have a couple of days gap between sprints
- Add changes/stories which might endanger sprint goal
- Decrease quality of work in order to finish stories in the sprint
- Split testing and other quality checks into another sprint
- Start a new sprint on Monday as it will end on Friday. People are not focused enough on Fridays
- Don't create "mini-waterfalls" during your sprints. In other words 1 sprint of analysis, 1 sprint of development and 1 sprint of testing

Popular sprint length

By order of popularity:

- **2 weeks**
- **3 weeks**
- **1 week**
- **4 weeks**

Sprints should not be longer than 1 month.

Scrum events

- Daily scrum (standup)
- Sprint planning
- Backlog refinement (grooming)
- Sprint review
- Sprint retrospective

Iteration vs sprint

Often sprints are called iterations. Iteration and sprint are the same concept.

Cancelling sprint

- Only product owner can cancel a sprint
- All events (sprint review, sprint retrospective & sprint planning) should take place in order to commence new sprint
- There should be no gap between cancelled and new sprints
- Try to avoid sprint cancellation

Sprint Goal

Sprint Goal is a short (1-2 sentences) description of what the team plans to achieve during sprint

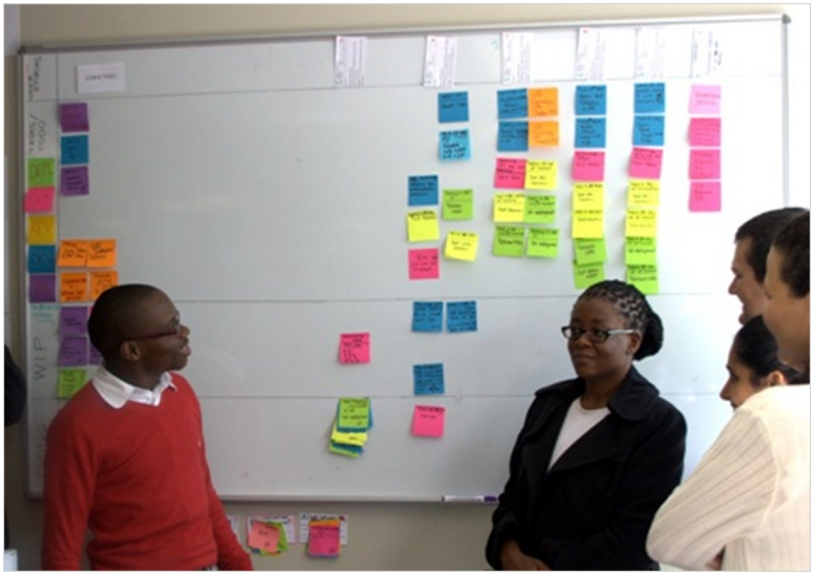
- **Whole team decides on sprint goal together with product owner, scrum master and stakeholders**
- **Sprint goal must be <u>realistic and SMART</u> (specific, measurable, achievable, relevant and timely)**
- **Sprint goal must not be forced onto the team**
- **Sprint goal is decided during sprint planning event**

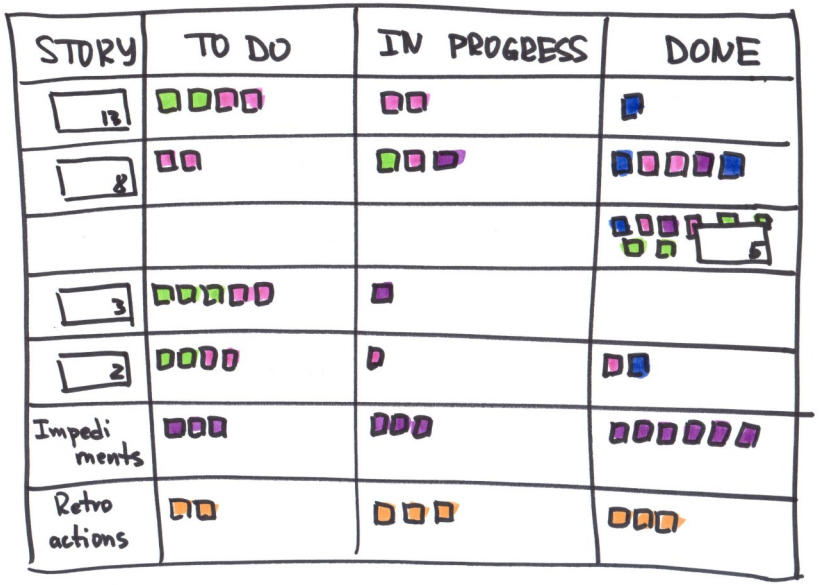
! Examples of sprint goal

- In this sprint we will allow users to log-in to the site, retrieve a forgotten password, and manage their own profile
- In this sprint we will implement basic shopping cart functionality including add, remove and update features
- In this sprint we will integrate VISA payment gateway into our billing module and investigate MasterCard gateway

Sprint Backlog

List of user stories, tasks and other activities team commits to deliver in order to achieve sprint goal





Tips

- Use different colours for different types of tasks (i.e. development—blue, impediments—purple, business analysis—pink), but each team can define their own standards at visualizing sprint backlog
- Limit number of user stories in progress—it will reduce amount of user stories carried over into next sprints
- Limit number of impediments in progress—focus on most serious blockers first
- Agree on maximum period of tome to resolve an impediment, before it is escalated to broader audience
- Use index cards for user stories and post-it notes for tasks
- A good user story can be done within 3 days and a good task should take no more than 1 day
- Add your retrospective actions to sprint backlog as well
- Use 'Super Sticky' post-it notes

Impediments

Impediment is anything which is standing in team's way towards achieving the print goal

! Problems with impediments

- If the impediment backlog lives in the mysterious black book of the scrum master, you have a problem
- If your impediment backlog does not change you have a problem
- If your impediment backlog is empty, you have a problem
- If you have an impediment backlog with a growing number of active impediments, you have a problem
- If the scrum master resolves all impediments himself/herself, you have a problem

? 4th standup question

One of the techniques to identify more impediments is to ask a 4th question at standup.

'How confident are you that you will achieve sprint goal?'.

If it is anything less than 100%, by asking why, you will discover an impediment.

'How can we go faster?'

Is another question to bring out impediments to the surface.

! Tips for dealing with impediments

- Make the impediments visible—use special colour post-it notes for them
- Search for impediments. Look out for impediment words ('still waiting', 'not available', 'hopefully', 'wish', 'guess', expected', I thought', 'try')
- Limit the number of impediments. Select 3 biggest ones and put a big red dot on each of them. It will help the team focus
- Help the team to resolve impediments

! Global vs local impediments

Global impediments need attention of your stakeholders and relevant stakeholders should be made aware of them as soon as possible.

Local impediments are within team's capability to resolve.

Sprint Burndown/Burnup

Sprint burndown/burnup are charts which represent progress of work during the sprint

? What do you need to create one?

- Have user stories
- Estimate each user story together with your team in story points (1 2 3 5 8 13 20 40 100)
- Calculate total number of story points in sprint
- Calculate total number of working days in sprint
- Should be updated by the team at the same time every day

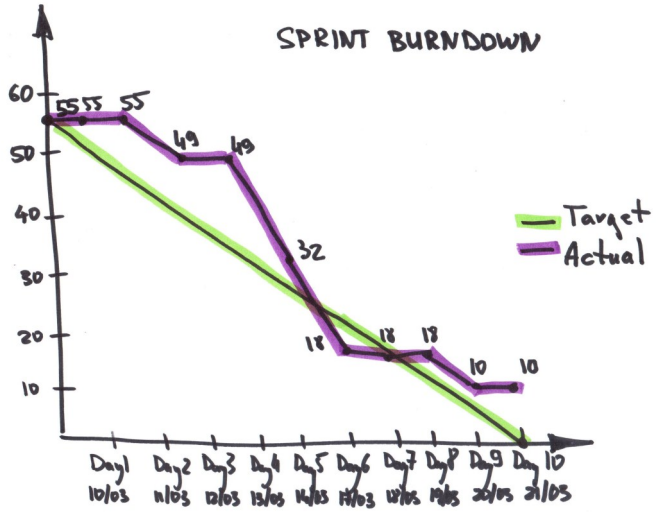
! Story points

Points are reflected in the chart once a whole story is complete as per definition of done. In other words, if a story is not complete, it is calculated as 0.

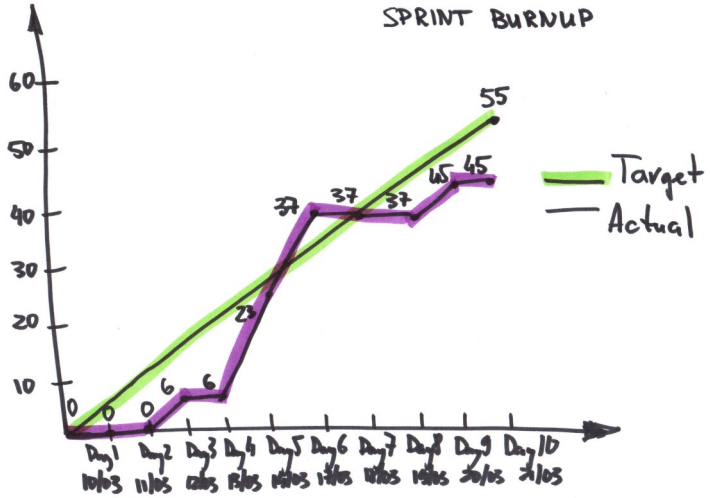
No points are assigned to tasks as an individual task doesn't usually carry business value on its own.

Both charts represent the same information.

Burndown displays work outstanding



Burnup displays work done



Daily Scrum [Standup]

Daily Scrum is an event to synchronise the team and plan for the next 24hrs

WHY Synchronise activities and create a plan for the next 24 hours

WHEN Every day

HOW LONG 15 minutes

WHO Mandatory: development team

Optional: scrum master, product owner, stakeholders

? Meet after

If you feel that a conversation is going on for too long and other team members start to lose interest, you can add the issue to the parking lot and ask those involved to discuss it right after the standup.

3 Standup Questions

Each team member should answer these 3 questions:

1. **What did I do yesterday that helped the development team meet the sprint goal?**
2. **What will I do today to help the development team to meet the sprint goal?**
3. **Are there any impediments that prevent me or the development team from meeting the sprint goal?**

! What else to do at standup/after standup

- ◊ Review impediments
- ◊ Update burndown/burnup chart
- ◊ Make decisions
- ◊ Update sprint backlog (i.e. move post-it notes, add more tasks)

Sprint Planning

Sprint Planning is an event to plan work for the sprint

WHY

Plan work; obtain commitment for the sprint; eliminate other meetings; manage stakeholder expectations; mitigate risks

WHEN

First day of each sprint

HOW LONG

2 hours for every week of sprint

WHO

Mandatory: development team, scrum master, product owner

Optional: stakeholders

Typical agenda

PART ONE: WHAT

1. Discuss team capacity
2. Identify sprint goal
3. Identify user stories needed to achieve sprint goal
4. Clarify requirements for these user stories
5. Update product backlog: split, delete, combine, move stories
6. Estimate user stories
7. Commit to sprint goal and user stories

PART TWO: HOW

1. Decide how to achieve the sprint goal (design)
2. Clarify requirements further
3. Negotiate trade-offs
4. Break down stories into tasks
5. Update Scrum board

? Product owner

Product owner should come to sprint planning prepared to talk about 2 sprints worth of work.

Prerequisites

- Product backlog and its complexity
- Stationery (post-it notes, index cards, markers)
- Team capacity (how much time will team have to do the work): leave, public holidays, training etc.
- Business conditions

Sprint Review

Sprint Review is an event for the team to show what they accomplished during the print and reflect on current state of the product

WHY

Demonstrate results of sprint; discuss product and adjust product backlog; collaborate on the next goals; review timeline, budget, release plan

WHEN

Last day of each sprint before sprint retrospective

HOW LONG

1 hour for every week of sprint

WHO

Mandatory: development team, scrum master, product owner, stakeholders

Typical agenda

- Review meeting agenda and guidelines
- Team walkthrough of completed functionality with product owner
- Team demonstrates working software to stakeholders
- Team discusses incomplete user stories
- Product owner moves/splits/re-prioritizes the backlog
- Product owner closes the sprint and accepts relevant functionality (if it wasn't accepted during the sprint)
- Open actions/impediments are noted

Say NO to…

- Say NO to slides, show working software instead
- Say NO to showing incomplete stories
- Say NO to scrum master or product owner doing demo, it is better for development team to demo their own work

Prerequisites

- List of complete/incomplete stories
- Big boardroom with projector screen
- Laptop
- Team needs to prepare for this event (~ 1 hour)

Sprint Retrospective

Sprint retrospective is an event for the team to reflect on current progress and find improvements

WHY Reflect on current progress; identify improvements

WHEN Last day of each sprint after sprint review

HOW LONG 1-1.5 hours

WHO Mandatory: development team, scrum master

Optional: retrospective should be safe place for the team to be truly open. Beware of inviting anyone outside of the development team as it might change dynamics of the retrospective.

Typical agenda

5 stages of retrospective

1. Set the stage
2. Gather & analyze data
3. Generate insights
4. Decide what to do
5. Close the retrospective

Practical example:

1. Each team member around the table says 3 words about previous sprint
2. Ask everyone to write 3 post-it notes for 'what went well' and 'what can be improved' and stick them on the wall
3. Group them into themes
4. Identify the most painful item
5. Brainstorm actions that can be taken in the next sprint to improve the item
6. Thumb vote on how retrospective went

More ideas

You can pick up more ideas for your sprint retrospective at

http://plans-for-retrospectives.com/

Or invite other scrum masters to facilitate a session for you.

Prerequisites

- White board
- Post-it notes

Actions

Make your actions SMART (specific, measurable, achievable, relevant and timely).

Assign owners to retrospective actions.

Put all retrospective actions on Scrum board (sprint backlog) and track them as any other work.

Backlog Refinement

Backlog refinement is an event to help the team to get user stories ready for future sprints

WHY

Get user stories ready; add detail to user stories; estimate user stories; identify risks and issues before sprint planning; update priority of user stories

WHEN

During sprint, perhaps weekly

HOW LONG

30 minutes—2 hours

WHO

Mandatory: development team, scrum master, product owner

Optional: stakeholders

Typical Agenda

1. Identify user stories to refine
2. Clarify requirements for these user stories
3. Identify open items/questions for these user stories and assign owners to them
4. Break down user stories which are too big
5. Discuss priority of these user stories and update if necessary
6. Estimate user stories

INVEST

A good user story is:

Independent / immediately actionable—ideally can be implemented in any order

Negotiable—and negotiated

Valuable—to the customer

Estimatable—enough to rank and schedule it

Small—and with short descriptions

Testable—I could write a test for it

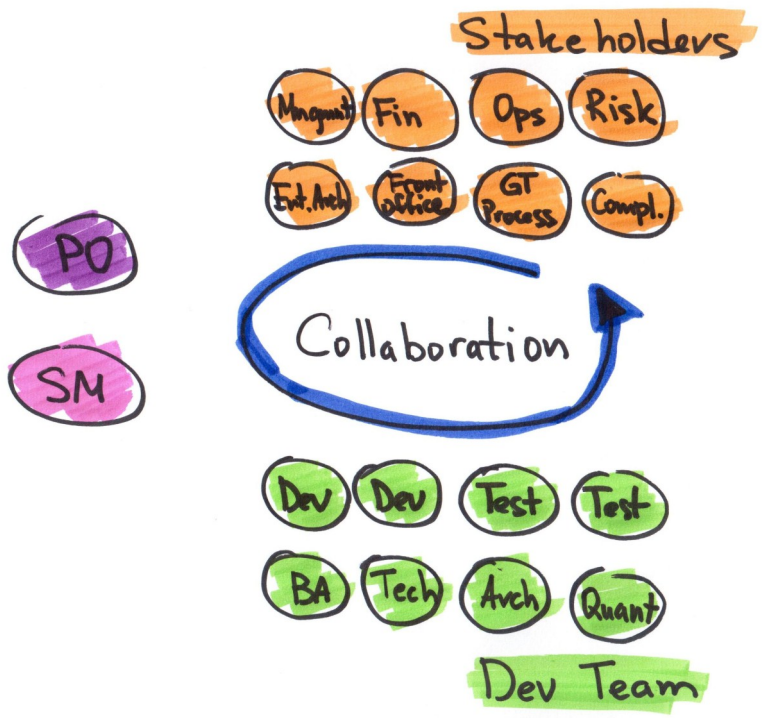
Prerequisites

◊ Product backlog

Scrum Roles

3 Roles in Scrum

- **Scrum Master**
- **Product Owner**
- **Development Team**



Development Team

- Cross functional
- Dedicated (allocated 100% to the team)
- Co-located
- Self-organized
- Ideal size for development team is 3 to 9

? Stakeholder or team member

If person works daily together with development team towards achieving sprint goal, then he/she forms part of the team.

Scrum Master

3 Main responsibilities

- **Facilitate Scrum events**
- **Identify and help to resolve impediments**
- **Be guardian of Scrum process**

? Scrum master vs product owner

Can product owner and scrum master be the same person?

No. The roles have different goals and responsibilities. Product owner protects interests of stakeholders and scrum master protects interests of their development team. Often they play offense-defense, which is not possible to achieve when one person combines both roles.

! Multiple teams

Good scrum master can work with 2-3 teams at the same time, great scrum master always works with a single team.

Full list of responsibilities

- Help the team to resolve impediments
- Facilitate Scrum events (prepare, lead, writeup)
- Help team to continuously improve their process
- Maintain Scrum tools (story board, backlogs, charts, etc.)
- Help team to create/maintain definition of done
- Help to create/refine user stories
- Coach team members, consult team members regarding everything agile
- Resolve conflicts within the team
- Help team to make decisions
- Encourage team self-organisation
- Mediate communication between development team and product owner
- Continue personal growth in Agile (user groups, conferences, reading books and articles, writing blogs)
- Interact constantly with other scrum masters within the organisation
- Help with release planning
- Be familiar with team's work/progress, feedback to stakeholders on team's progress
- Bring right people together for communication
- Keep in touch with stakeholders regularly
- Give learning opportunities to people within the organisation
- Arrange/capture team policies and agreements, remind the team about them
- Ask open questions, encourage team members to express their opinions
- Help the team to keep focus

Product Owner

3 Main responsibilities

- **Own product backlog**
- **Prioritise work**
- **Accept/reject work**

? How much time do I need

Product owner is a full time role in many organisations. We suggest to spend at least 20 hours per week for each team in order to be able to achieve desirable outcomes.

! Proxy Product Owner

Backlog owner or proxy product owner is usually an interim role during Agile implementation. Person in this role takes over product backlog management responsibilities.

In mature teams this roles considered an anti-pattern as proxy product owner becomes a middle man in communication between product owner and the team.

Full list of responsibilities

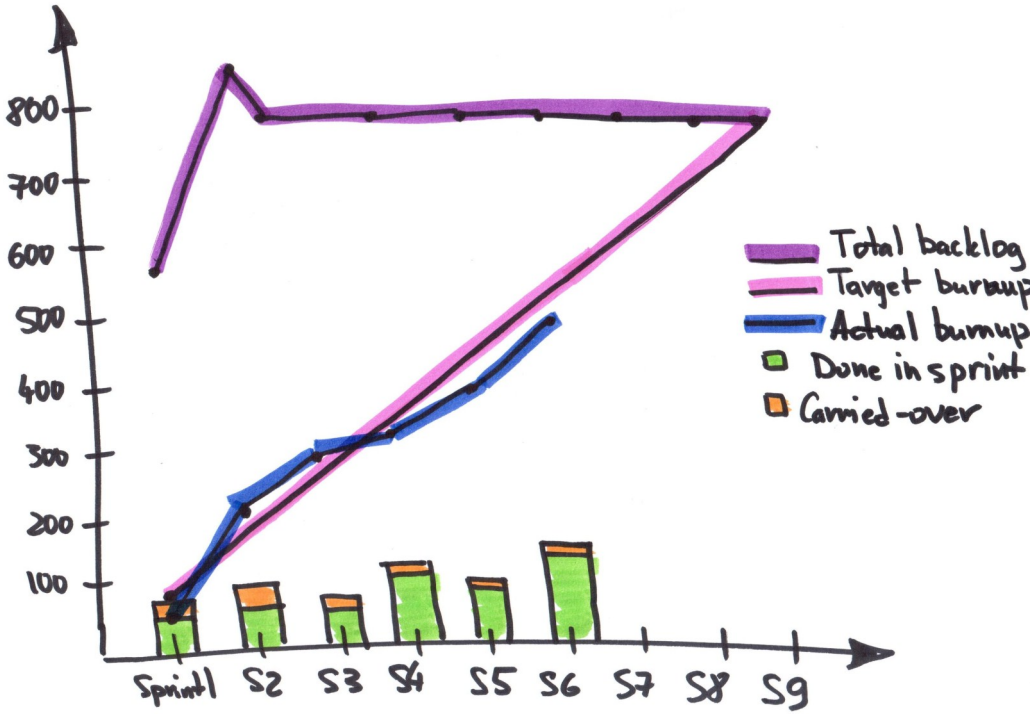
- Ensure that the team builds the right product
- Manage ROI and make sure to deliver business benefits
- Responsible for the budget constraints to be met
- Ensure that what the team is asked to build is aligned with what the sponsor, stakeholders and users want
- Provide a vision for the product
- Provide boundaries to describe the realities within which the vision must be realised (e.g. time frames, external quality)
- Make sure management, stakeholders, sponsors are informed and the vision is aligned with their wishes
- Communicate the project vision to the team and motivate the team to subscribe to the product vision
- Manage stakeholder expectations regarding requirements and project boundaries (time frames, quality, technical constraints etc.)
- Create and maintain the product backlog (the product owner can delegate some of the work of writing user stories to a BA but the Product Owner is still responsible for ensuring that the work is being done and is being done properly)
- Continuously refine the product backlog
- Prioritise user stories within the product backlog
- Define releases, their goals and sprint goals
- Continuously answer questions to add detail to requirements/user stories
- Accept/reject developed user stories at the end of the sprint (or during the sprint)
- Communicate about the project within the organisation (e.g. demo attendance and invites, forecasting, management reporting, sponsor liaison)

Product Burnup

Product burnup is a chart which represents the progress towards completing the product backlog

? What do you need to create one?

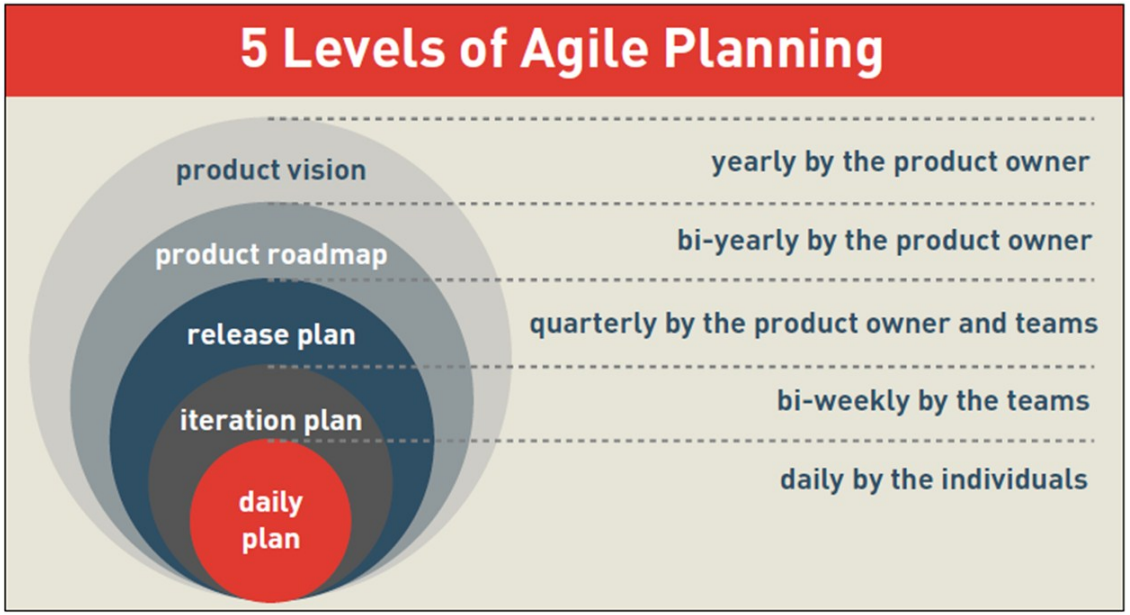
- Have all requirements included in product backlog (initially these requirements will be epics/high level features. With time, product owner and team will refine these epics into more detailed stories)
- All stories and epics are estimated in story points



! Valuable information

- Project delivery date, based on historical velocity
- Average velocity required to finish work by specific date
- Is project ahead of behind schedule (comparing ideal and actual burning rates)?
- Amount of carry over stories in each sprint and its trend
- Velocity per sprint and its trend

Release Plan



Release plan shows days of release and epics/high level features to be realized in this specific release.

	Date	Sprints	Status	Description
Release 1	28/01/2014	2-3	**Delivered**	Feature A Feature B Integration with system X Feature C
Patch release 1.2	05/02/2014	4	**Delivered**	Fixes for Feature C
Patch release 1.3	09/02/2014	5	**Delivered**	Fixes for Feature C and A
Release 2	28/04/2014	6-8	**Off track**	Feature D Migration of data Y
Release 3	15/05/2014	9-10		Feature E

? How often to release

Release as often as possible. It will allow for early feedback from users and stakeholders.

First release should be your minimum viable product (MVP).

! Flight Plan

Flight Plan (Story Mapping, Critical Path) is a 2-dimentional view of product backlog which highlights timelines and dependencies.

Notes

Notes

References

1. Schwaber, K. & Sutherland, J. (2013, July). Scrum Guide. Retrieved from https://www.scrum.org/Scrum-Guide
2. Scrum Alliance. (2014). Agile Atlas. Retrieved from http://agileatlas.org/
3. Cohn, M. (n.d.). Retrieved from http://www.mountaingoatsoftware.com/agile/scrum
4. Greaves, K. & Laing, S. (2013). Growing Agile: A Coach's Guide to Training Scrum. Lean Pub.
5. Greaves, K. & Laing, S. (2014). Impediments [Video]. Youtube.
6. De Gregorio, B. (2014). Agile Boot Camp. Using the Scrum Framework [Presentation].
7. Watts, G. (2013). Scrum Mastery: From Good To Great Servant-Leadership. Inspect & Adapt Ltd.

Further Reading

Title: Succeeding with Agile: Software Development Using Scrum
Author(s): Mike Cohn
Publisher: Addison-Wesley
Publication Date: 2010

Title: Agile Estimating and Planning
Author(s): Mike Cohn
Publisher: Prentice Hall
Publication Date: June 2010

Title: Agile Product Management with Scrum
Author(s): Roman Pichler
Publisher: Addison Wesley
Publication Date: 2010

Title: Agile Retrospectives
Author(s): Esther Derby and Diana Larsen
Publisher: Pragmatic Programmers
Publication Date: 2006

Title: Agile Software Development with Scrum
Author(s): Ken Schwaber, Mike Beedle
Publisher: Prentice Hall
Publication Date: 2002

Title: Agile Testing: A Practical Guide for Testers and Agile Teams
Author(s): Lisa Crispin and Janet Gregory
Publisher: Addison-Wesley
Publication Date: 2009

Title: Clean Code
Author(s): Martin
Publisher: Prentice Hall
Publication Date: 2009

Title: Continuous Integration
Author(s): Paul Duvall
Publisher: Addison Wesley
Publication Date: 2007

Title: Extreme Programming Explained
Author(s): Kent Beck
Publisher: Addison Wesley
Publication Date: 2007

Title: Extreme Programming Installed
Author(s): Jeffries, Anderson, and Hendrickson
Publisher: Addison-Wesley
Publication Date: 2001

Title: How Do We Know When We Are Done?
Author(s): Mitch Lacey
Publisher: Scrum Alliance Website Article
Publication Date: 2008

Title: Implementing Lean Software Development
Author(s): Mary Poppendieck, Tom Poppendieck
Publisher: Addison Wesley
Publication Date: July 2007

Title: Planning Extreme Programming
Author(s): Kent Beck, Martin Fowler
Publisher: Addison Wesley
Publication Date: December 2004

Title: Pragmatic Project Automation
Author(s): Clark
Publisher: Pragmatic Books
Publication Date: 2004

Title: Project Retrospectives: A Handbook for Team Reviews
Author(s): Norman L. Kerth
Publisher: Dorset House Publishing
Publication Date: 2001

Title: Promiscuous Pairing and Beginner's Mind: Embrace Inexperience
Author(s): Arlo Belshee
Publisher: IEEE
Publication Date: 2006

Title: Refactoring: Improving the Design of Existing Code
Author(s): Fowler
Publisher: Addison-Wesley
Publication Date: 1999

Title: Retrospectives – The Missing Practice
Author(s): Tim Mackinnon
Publisher: ThoughtWorks Company
Publication Date: 2003

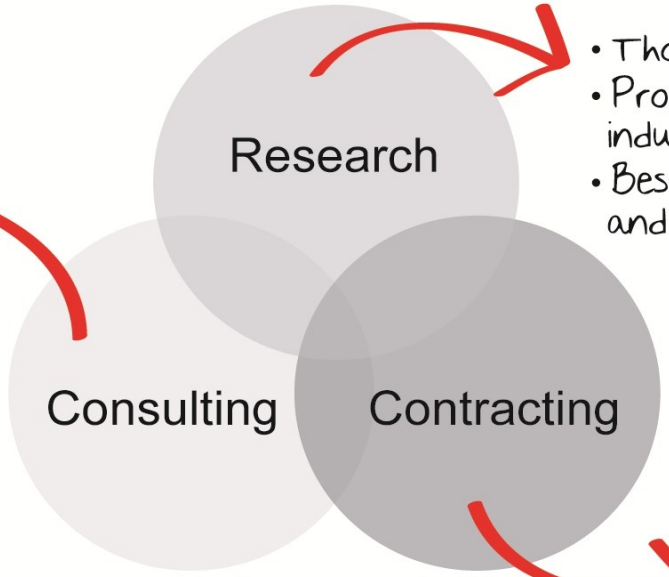
Title: Scrum Primer
Author(s): Pete Deemer, Gabrielle Benefield,
Craig Larman and Bas Vodde
Publisher:
Publication Date: 2010

Title: User Stories Applied
Author(s): Mike Cohn
Publisher: Addison Wesley
Publication Date: 2004

IQbusiness

consulting | research | contracting

- Agile consulting
- Agile coaching
- Agile training
- Performance Support tool
- Scaling Agile to portfolio, programmes and teams through best practice frameworks



- Thought leadership
- Provision of tailored research in Agile industry
- Best practice frameworks, methodologies and approaches

- Agile coaches
- SAFe Program consultants
- Certified Scrum Masters
- Certified Product owners
- Certified Scrum Professionals
- Agile Certified Practitioner
- Professional Scrum Masters

agility@iq clients

OUTSURANCE YOU ALWAYS GET SOMETHING OUT

SHOPRITE

Investec

BARCLAYS

ABSA

METROPOLITAN HEALTH

NEDBANK

Sanlam

TransUnion

Follow up on Twitter

@AgilityIQ

Join our LinkedIn group

Agility@IQ

IQ Business Park
Third Avenue
Rivonia
Johannesburg
www.iqbusiness.co.za
agility@iqgroup.net