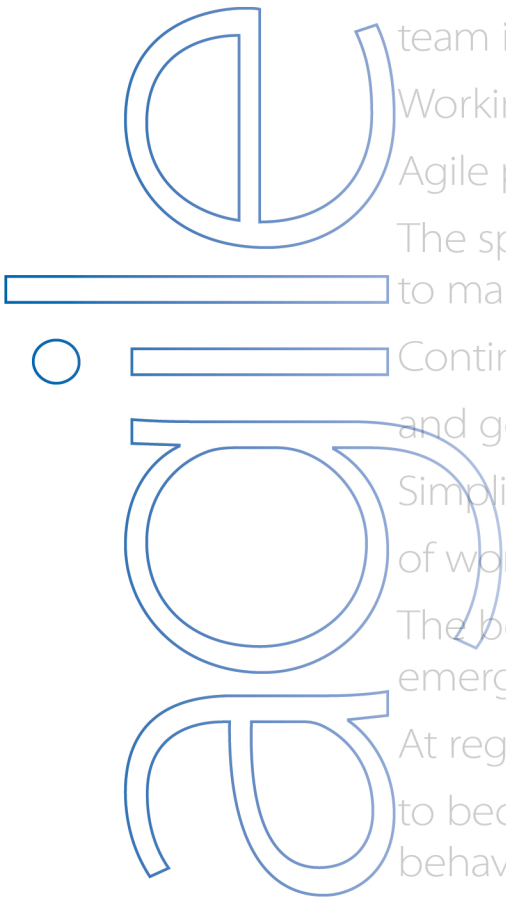
COLLABNET

Certified ScrumMaster Course

Workbook



Name: _____

COURSE AGENDA

This is what you can expect to see during these two days (the times are approximate):

	Day 1	Day 2
9 – 12	Agile thinking Agile principles Complexity and its effect to processes	Project kick-off / planning exercise - Vision - Writing user stories - Collaborative exercises to analyze and prioritize the user stories in the product backlog
12 – 13	Lunch	
13 – 15	Scrum framework - Meetings, roles, artifacts	Initial release plan Sprint planning in more detail
15 – 17	Scrum roles in detail	Elective topics, Q&A

Breaks will be held roughly every 60-90 minutes (for 10-15 minutes each). If you have requests for particular break times, come talk to me about.

CERTIFICATION

According to Scrum Alliance, CSM certification requires approved participation on a CSM course run by a Certified Scrum Trainer (CST). The CST has the authority to decide who gets the certificate or (starting April 1st, see below) who can take the CSM test.

For me to send your name forth to Scrum Alliance, you need to participate actively on this course for the full two days. If you have something that absolutely requires you be absent for a portion of the class, talk to me and we'll agree what to do.

Assuming full participation, we will send your name and email address to Scrum Alliance typically within two weeks of the course.

You will then receive an email from Scrum Alliance containing an Internet link to your personalized online CSM test. Until the end of March, 2012, the test will be pass only, but a completion is required for certification. Starting April 1 (unless things change), the test can be failed. There is currently no information about how many of the questions need to be correct for passing the test. You have 90 days from the reception of the email to do the test. The test will take approximately 20-40 minutes to complete, and you can do it at your own pace.

If you, for some reason, happen to fail the test, there will be an opportunity to redo it.

Once you've passed the test, you need to accept some simple license terms, and then you are Certified ScrumMaster. **Congratulations!!**

QUOTES

In your table group, please discuss these quotes. How do you interpret them? How is what they comment related to Agile or Scrum? Or to ourselves?

Write down key notes below each quote. Be prepared to comment on your insights and opinions.

"There are those who look at things the way they are, and ask why... I dream of things that never were, and ask why not?"

– Robert Kennedy

"Everything should be made as simple as possible, but not simpler."

– Albert Einstein

"Often detail adds no more usefulness – only a false appearance of validity."

– Edward de Bono

"The best bang-per-buck risk mitigation strategy we know is incremental delivery."

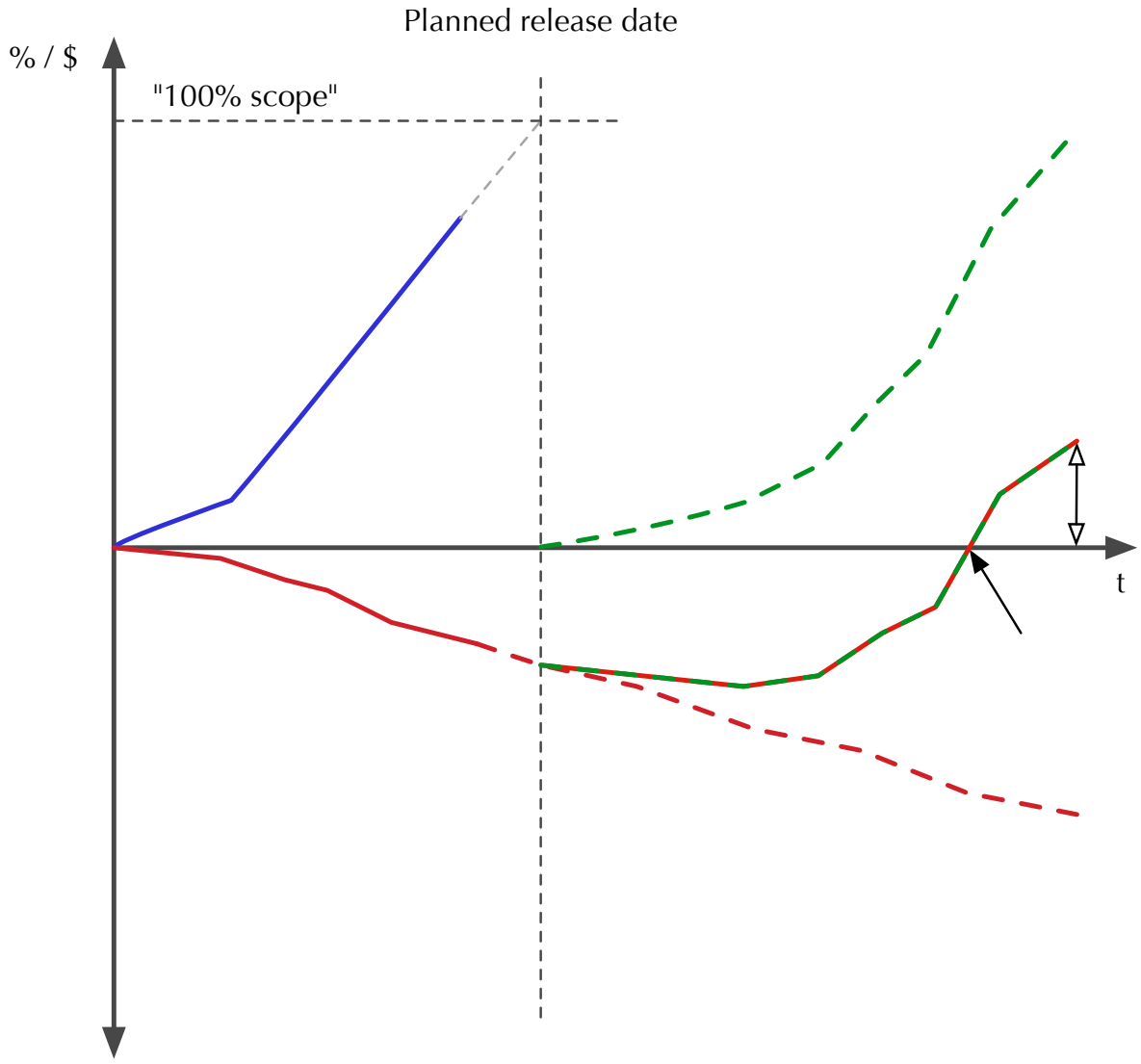
– Tom DeMarco, 2003

"It is not the strongest or most intelligent of species that survives, but the most adaptable."

– attributed to Charles Darwin

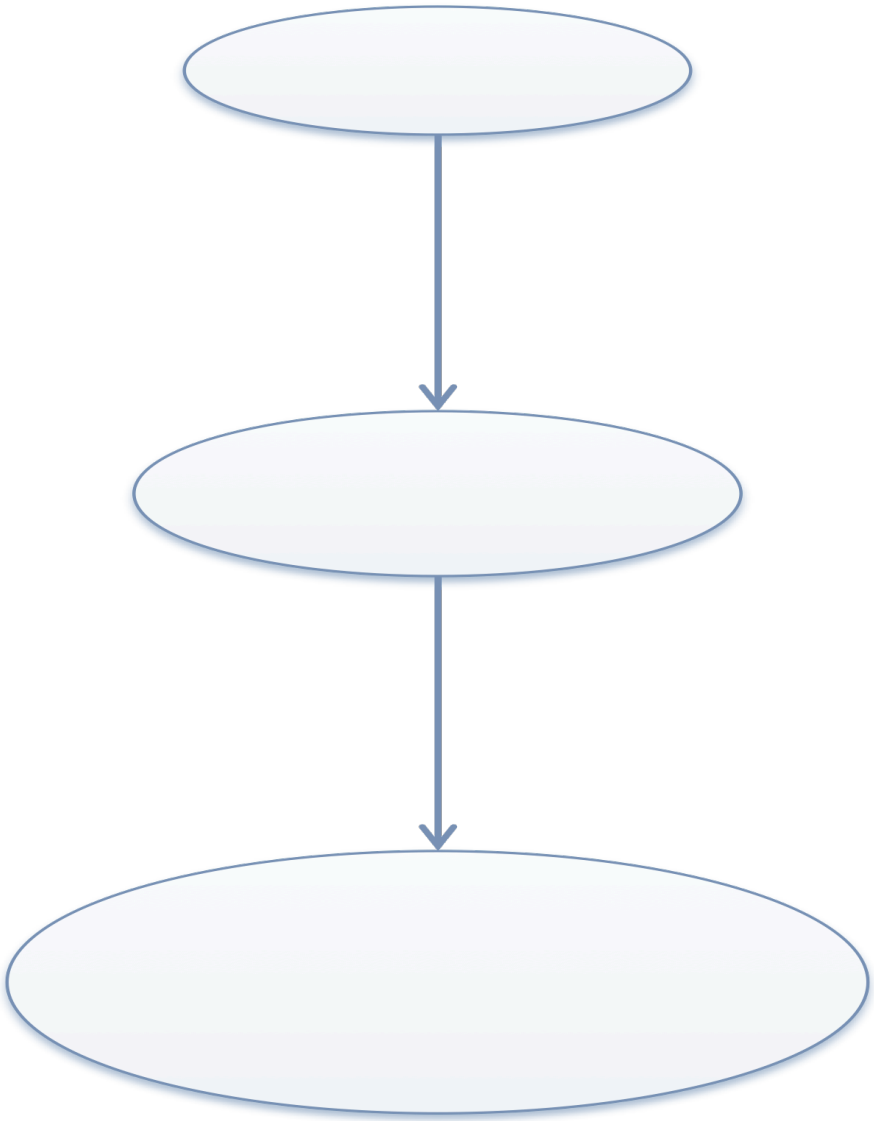
WHY AGILE FOR BUSINESS?

Please fill up the diagram as we go through it and write your notes to it.



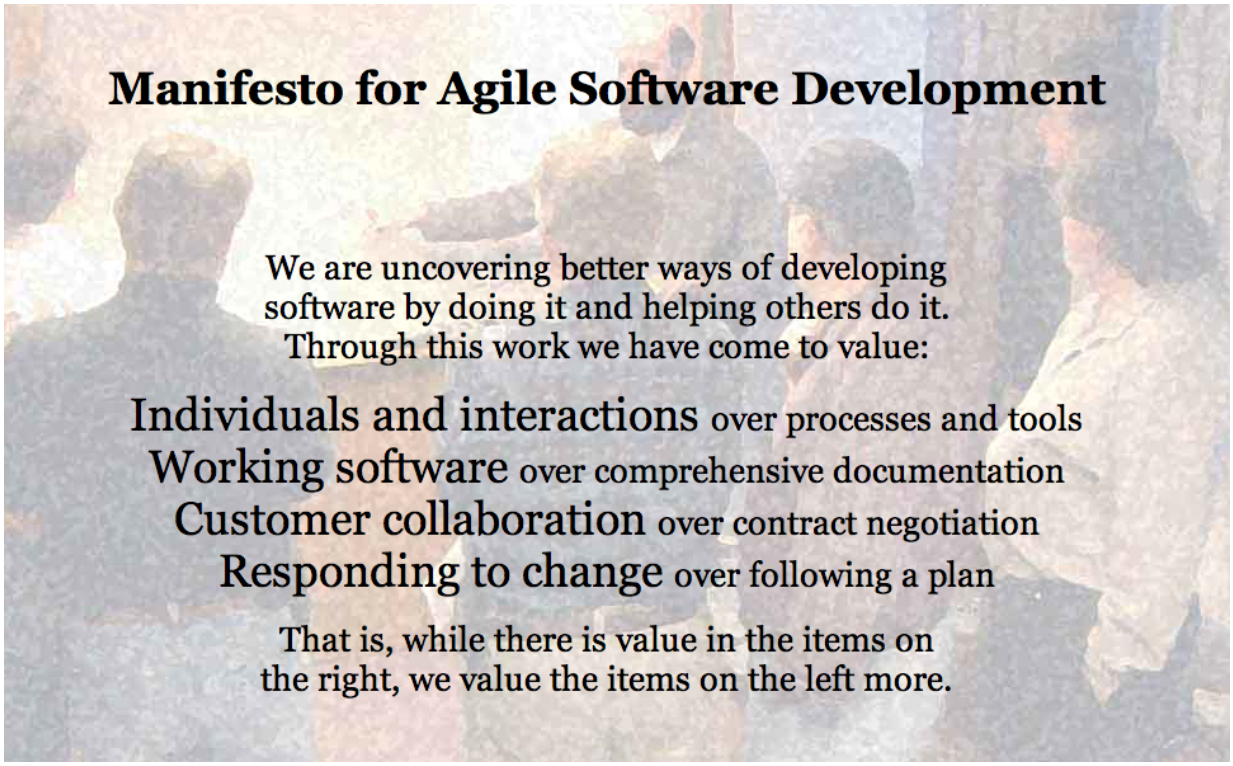
THE BIG PICTURE

As this topic is discussed, update the diagram below with conceptual levels and elements that are related to each level.



AGILE MANIFESTO

Here is the Agile Manifesto (for software development). In the space below the manifesto, you can create a mind map of your insights and notes about it and the discussions.



Below are the 12 principles that are part of the Agile Manifesto. In your table group, create "pocket-sized principles" by collaboratively extracting the essence out of each one and condensing that to three words or less.

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

Business people and developers must work together daily throughout the project.

Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Working software is the primary measure of progress.

Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

Continuous attention to technical excellence and good design enhances agility.

Simplicity--the art of maximizing the amount of work not done--is essential.

The best architectures, requirements, and designs emerge from self-organizing teams.

At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

COMPLEXITY AND CAUSALITY

Quoting from http://www.noop.nl/2008/08/simple-vs-complicated-vs-complex-vs-chaotic.html,

My car key is simple.

It took me about three seconds to understand how my car key works. OK, maybe that's not quite correct. Mine has a battery in it. If I take it apart it might take me another three hours to understand its details. But yeah, I'm smart, I'll manage.

My car is complicated.

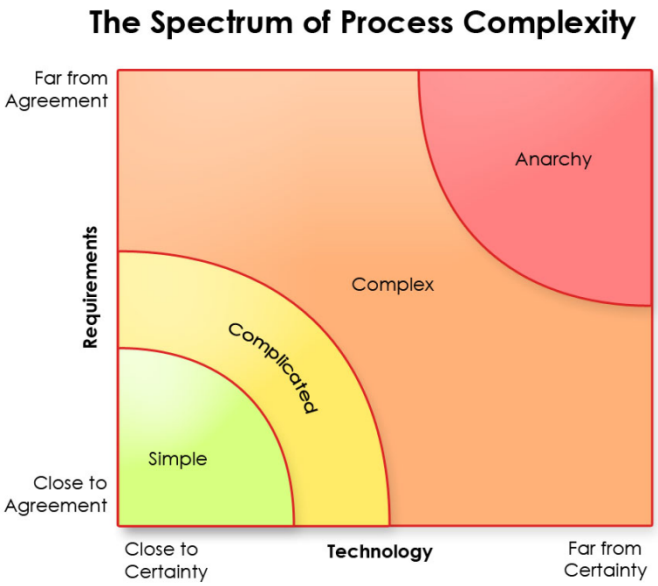
It would take me years to understand how my car works. And I don't intend to. But if I did, then some day in the far future I would know with certainty the purpose of each mechanism and each electrical circuit. I would fully understand how to control it, and I would be able to take my car apart and reassemble it, driving it exactly as I did before. In theory, of course. In practice, only real men do things like that.

Car traffic is complex.

I can travel up and down the same street for twenty years, and things would be different every time. There is no way to fully understand and know what happens around me on the road when I drive, how other drivers operate their vehicles, and how the people in the streets interact. I can make guesses, and I can gain experience in predicting outcomes. But I will never know for sure.

Car traffic in Lagos (Nigeria) is chaotic.

When things get too complex, they easily become chaotic. Traffic in Lagos is so bad, it is not even predictable. Poor infrastructure and planning, heaps of waste, pollution, lack of security, floods, and many more problems make it one of the worst places in the world to be, as a simple car driver.



Discuss in 3-4 person groups what each of these categories mean

Then, brainstorm in 3 minutes as many examples to each of the categories as you can. Try to find examples that are related to your working environment.

SCRUM SIMULATION

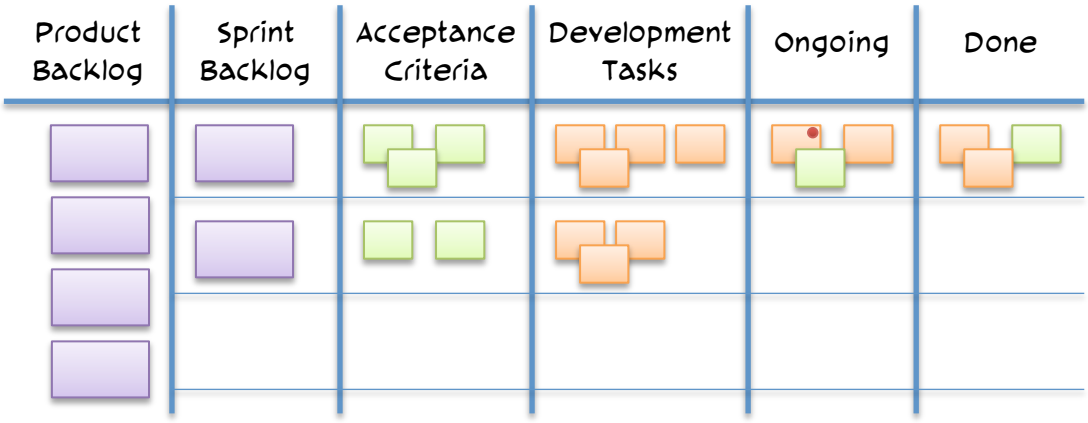
In this simulation, we explore the Scrum framework through practice.

The goal is to create a Scrum ”playbook” brochure for those not yet familiar with, or with an unclear understanding,of, the Scrum framework. There are seven stories in the backlog (in separate cards), each with their own acceptance criteria. There will be two Sprints, each of which is to end with a potentially shippable version of the brochure.

The simulation will have the following phases:

- Initial design workshop – 15 minuts
  - In this workshop, the team takes a look at the stories and creates their first ”product design” of the brochure, with a look at contents in general.
- Sprint 1 planning – 15 minutes
  - In Sprint planning, the team looks at the product backlog together with the PO, and select the product backlog items they can commit to delivering. Also, the team designs the sprint outcome in more detail and plans the needed tasks. Tasks are written down on post-it notes and placed on team’s Scrum board.
- Day 1 of Sprint 1 – 15 minutes
  - Daily Scrum – 3 minutes
  - Day 1 – 12 minutes
- Days 2 and 3, following the structure of day 1
- Sprint Review for Sprint 1 – 5 mins per team
  - Teams demonstrate and receive stakeholder feedback for their ”potentially shippable product increment”.
- Sprint Retrospective for Sprint 1 – 5 mins
  - Teams review their process and agree how to work better in Sprint 2.
- Sprint 2 with Sprint Planning, 3 development days and Sprint Review

The teams will plan and track their progress using a Scrum board with the following general appearance:



BUILD YOUR OWN SCRUM

With your table group, illustrate the Scrum framework in a way that makes most sense to you. You will do this part using cut-out pictures given by the instructor.

Below, you can draw your diagram at the end of the exercise.

If you want to do this exercise in your own training sessions, you can find the latest version from http://weisbart.com/buildyourownscrum/

SCRUM ROLES VS. TRADITIONAL PROJECT MANAGER

In small groups, discuss the different kinds of responsibilities and activities traditional project managers typically have. Then discuss how those responsibilities and activities would map to Scrum roles. Please write your mapping to the diagram below.

Product Owner responsibilities

ScrumMaster responsibilities

Development Team responsibilities

Responsibilities not related to Scrum

PRODUCT VISION

As examples of how a vision can be effectively created and communicated, here are three different approaches/tools:

Geoffrey Moore's template from Crossing the Chasm:

For *(target customer)*
Who *(statement of the need or opportunity)*
The *(product name)* is a *(product category)*
That *(key benefit, compelling reason to buy)*
Unlike *(primary competitive alternative)*
Our product *(statement of primary differentiation)*

Product Box

Design the cover (or the whole box) of an imaginary product box, in which the product would be shipped out in. For services or products without a box, just imagine it would be delivered in one. The box cover should have:

- The name of the product
- An image or logo of the product
- 3-4 key features (and no more than that)

User Testimonials

Write three user testimonials that you would like to hear from the actual users after the launch of the product or system. Each testimonial should have:

- One key feature or an important aspect of a central feature
- Description of the way in which the feature was useful to the user
- Name of the user and some description of the user's type or background

In your table groups, spend a few minutes discussing e.g. the following questions:

- Did the exercise work in clarifying the vision?
- What were the most important and useful part of the exercise?
- Why do exercises like these work? What is important in the exercise itself?

WRITING USER STORIES

Typical templates used:

- ***As a*** *<user role>**, I can*** *<do something>* ***in order to*** *<some benefit or purpose>*
- *<User role>* ***can*** *<do something>* ***so that*** *<some benefit or purpose>*

The "can" work can be replaced by "must", "should", or any appropriate verb.

User stories consist of three parts (three C's):

- _____
- _____
- _____

When you use index cards, it helpful to leave some space to the top and bottom of the card for different attributes, e.g. MoSCoW priority, value, size estimate*.

* These will be explained later as we do the planning exercise.

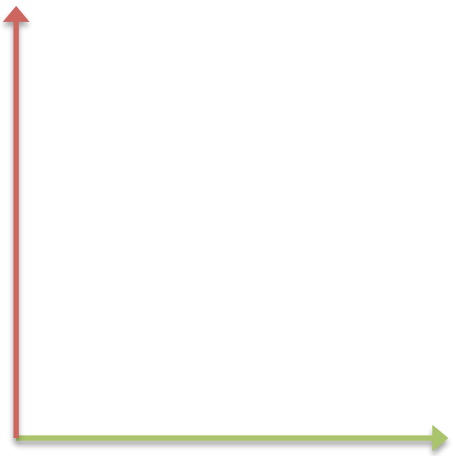
Remember, Scrum talks only about "product backlog items" (PBI's); user stories (a practice) is just one way of writing a PBI. Also other types of items can be placed in the Product Backlog.

SHAPING THE INITIAL PRODUCT BACKLOG & RELEASE PLAN

To get a project started, it needs a product backlog that has been sufficiently prioritized to allow the selection of valuable work for the first sprint. One tested sequence of analysis and planning activities, preceded by user story writing, is the following (write your comments and notes to the space below the names of the practices):

MoSCoW

Risk / Value Estimation



Dependency Mapping

Initial Release Plan

"READY" STORIES AND PRODUCT & SPRINT BACKLOGS

The first version of a user story can be virtually anything, but as the actual development iteration approaches, it is recommendable to groom them to meet the INVEST criteria:

I = _____

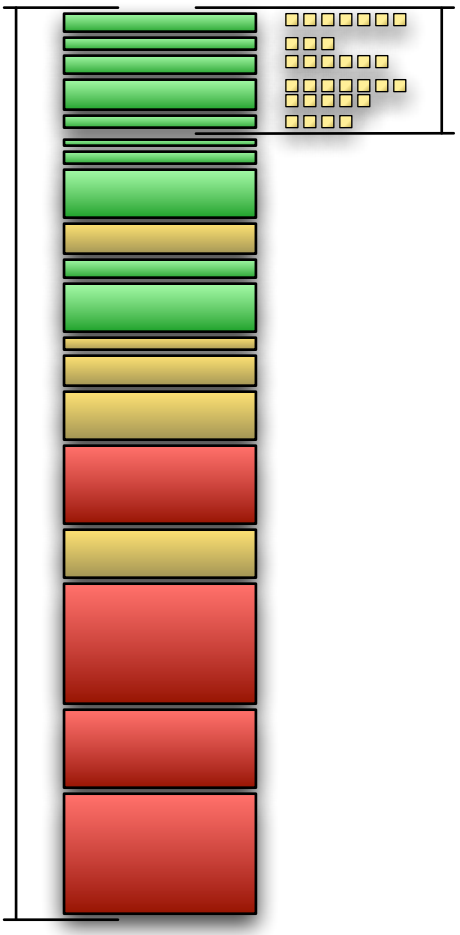
N = _____

V = _____

E = _____

S = _____

T = _____



SPLITTING USER STORIES

Extract smaller story...

- … by focusing on a particular user role or persona ("Prioritize your users first, then your user stories." -- Jeff Patton), e.g. "first time user", "social networker", "my mom", "Jack"
- … by substituting basic utility for usability (first make it work, *then* make it pretty)
- … by splitting on CRUD (Create, Read, Update, Delete) boundaries
- … by focusing on distinct scenarios, such as "happy day" and exception flows
- … by focusing on a simplified algorithm
- … by buying some component(s) instead of building everything yourself
- … by discarding technologies that increase hassle, dependency, and vendor lock
- … by substituting some manual processes instead of full automation
- … by substituting batch processing instead of online processing
- … by substituting generic instead of custom
- … by reducing supported hardware/OS/client platforms
- … from the acceptance criteria of a another story
- … by substituting "one" instead of "many"
- … by scanning for keywords like "and", "or", periods, and other kinds of separators
- … by separating an implied subfeature from the main story

This is not an exhaustive list*.

Use the above to extract smaller stories from the following stories:

As a User, I can login to the service so that...

_____

_____

_____

As an Author, I can manage my book submission page, in order to...

_____

_____

_____

As a User, I can use the service using any of the major browsers, so that...

_____

_____

_____

* This list uses ideas by Bill Wake, Lasse Koskela, Mark Levison, and Jeff Patton. List originally composed by Michael James. More info:
http://xp123.com/articles/twenty-ways-to-split-stories/
http://radio.javaranch.com/lasse/2008/06/13/1213375107328.html
http://agilepainrelief.com/notesfromatooluser/2010/09/story-slicing-how-small-is-enough.html
http://www.amazon.com/User-Story-Mapping-Jeff-Patton/dp/1449304559

RELEASE PLANNING

Let us assume the following prioritized Product Backlog (105 story points in total size):

Story number	Size
Story 1	3
Story 2	2
Story 3	5
Story 4	5
Story 5	3
Story 6	8
Story 7	5
Story 8	13
Story 9	13
Story 10	20
Story 11	8
Story 12	20

You are the PO. The team has been working on the product for already several sprints and has established probable velocity between 7 and 10. The next release is 7 sprints into the future.

- What is the scope that you feel could commit to stakeholders with reasonable safety?
- Which stories you feel you should pretty much rule out of probable release scope?

Visualize the "thresholds" to the Product Backlog by e.g. drawing lines appropriately.

Map the stories (using story numbers) to the following simple release plan.

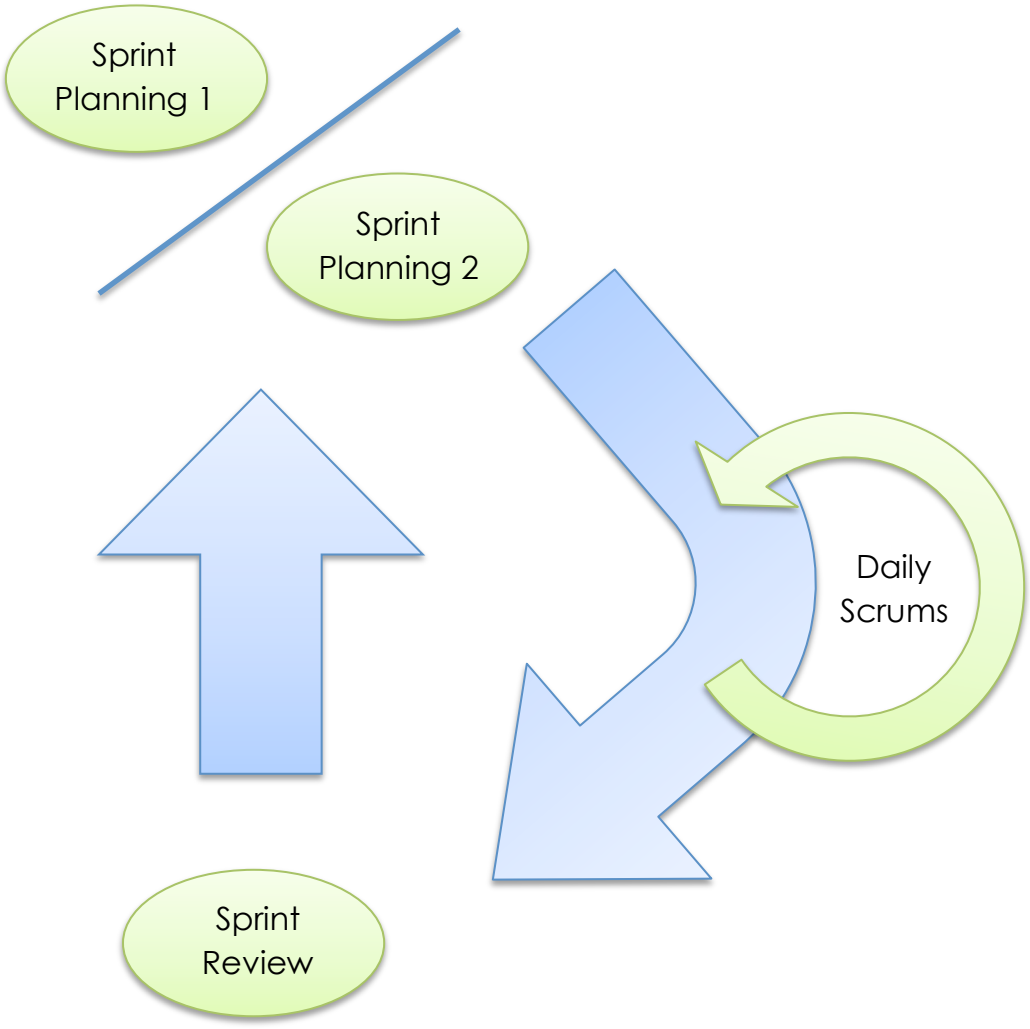
- How many points worth of stories can you allocate to each "slot" (note that multiple sprints are grouped together as you plan further out)?
- Is this simple method sufficient for release planning? If not, what other things need to be considered?

Consider the following scenarios:

- 2 Sprints forward, the team has delivered up to story 6 (26 more story points). They've updated their probable velocity to between 9 and 14. How does this change the release plan?
- User review at the end of Sprint 2 revealed new ideas and requirements that should be incorporated into the release. The team has estimated that their total size is 16. How does this change affect the release plan?

SPRINT PLANNING AND EXECUTION

The iterative and incremental product* development cycle in Scrum is managed in three key meetings – Sprint Planning, Daily Scrums, and Sprint Reviews. As we discuss them in more detail, please do write your key insights into the diagram below.



Compare your notes with a pair. What kind of differences do you notice? What do you make of those differences?

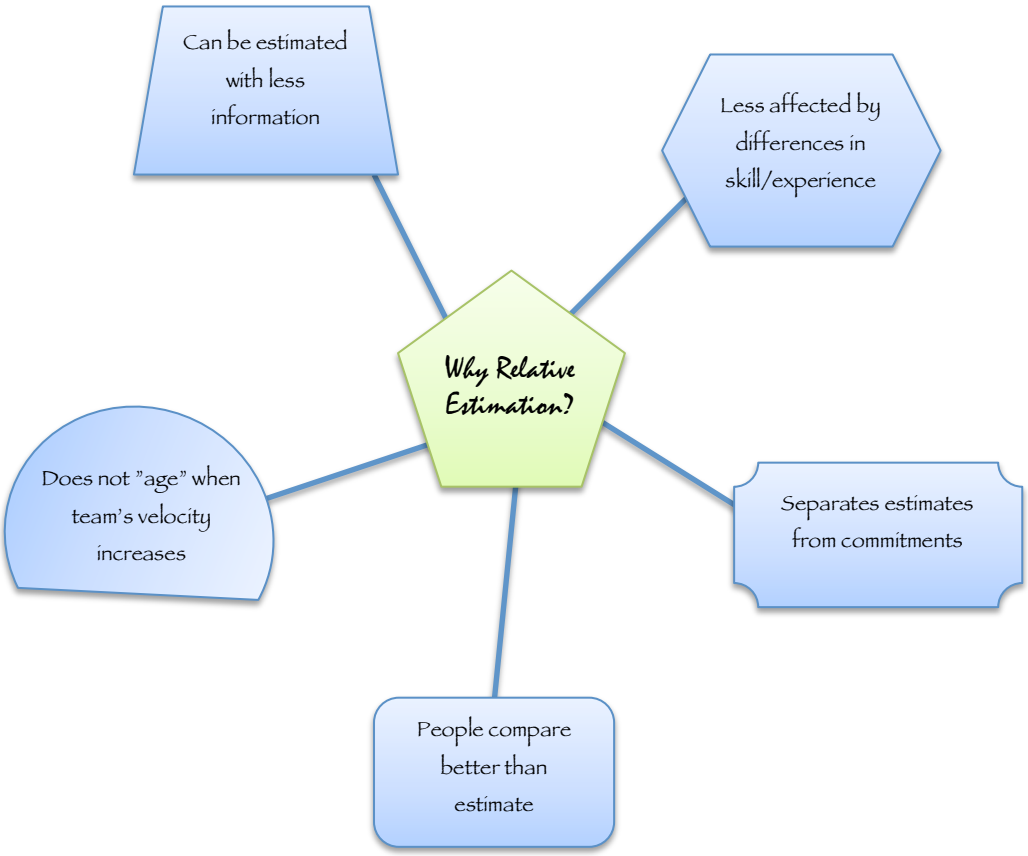
* The process improvement cycle consists of Sprint Planning, Daily Scrums and Sprint Retrospective. So there are two simultaneous cycles ongoing in Scrum at all times.

RELATIVE ESTIMATION

In relative estimation, items are compared against one another rather than against a certain metric. For example, one item maybe considered twice the size of another, rather than estimating how many hours or days it would take to do.

In the diagram below, add any additional notes you think are relevant to this topic.

In table groups, discuss each of the five benefits. What do each of them mean? Do you know some data or information regarding them?



When would time-based estimates be more useful? Where using them would make more sense than relative estimation?

TEAM FORMATION

From Wikipedia on Forming, Norming, Storming, Performing:

Forming

In the first stages of team building, the forming of the team takes place. The individual's behavior is driven by a desire to be accepted by the others, and avoid controversy or conflict. Serious issues and feelings are avoided, and people focus on being busy with routines, such as team organization, who does what, when to meet, etc. But individuals are also gathering information and impressions - about each other, and about the scope of the task and how to approach it. This is a comfortable stage to be in, but the avoidance of conflict and threat means that not much actually gets done.

The team meets and learns about the opportunities and challenges, and then agrees on goals and begins to tackle the tasks. Team members tend to behave quite independently. They may be motivated but are usually relatively uninformed of the issues and objectives of the team. Team members are usually on their best behavior but very focused on themselves.

The forming stage of any team is important because, in this stage, the members of the team get to know one another, exchange some personal information, and make new friends. This is also a good opportunity to see how each member of the team works as an individual and how they respond to pressure.

Storming

Every group will then enter the storming stage in which different ideas compete for consideration. The team addresses issues such as what problems they are really supposed to solve, how they will function independently and together and what leadership model they will accept. Team members open up to each other and confront each other's ideas and perspectives. In some cases storming can be resolved quickly. In others, the team never leaves this stage. The maturity of some team members usually determines whether the team will ever move out of this stage. Some team members will focus on minutiae to evade real issues.

The storming stage is necessary to the growth of the team. It can be contentious, unpleasant and even painful to members of the team who are averse to conflict. Tolerance of each team member and their differences needs to be emphasized. Without tolerance and patience the team will fail. This phase can become destructive to the team and will lower motivation if allowed to get out of control. Some teams will never develop past this stage.

Norming

The team manages to have one goal and come to a mutual plan for the team at this stage. Some may have to give up their own ideas and agree with others in order to make the team work. In this stage, all the team members takes the responsibility and have the ambition to work for the success of the goals of the team.

Performing

It is possible for some teams to reach the performing stage. These high-performing teams are able to function as a unit as they find ways to get the job done smoothly and effectively without inappropriate conflict or the need for external supervision. Team members have become interdependent. By this time they are motivated and knowledgeable. The team members are now competent, autonomous and able to handle the decision-making process without supervision. Dissent is expected and allowed as long as it is channeled through means acceptable to the team.

CHALLENGING SITUATIONS WITH TEAMS

In table groups, discuss the following scenarios. What kind of issues draw your attention? What "smells" suspicious? What would you do to start resolving the situation?

SCENARIO 1

You are the ScrumMaster and are heading for the team room. The functional analyst runs past you crying and the lead engineer runs past you enraged, both on the way to their functional managers' offices.

You go into the team room. You can cut the tension with a knife it is so thick.

Apparently, the analyst has been writing specs and giving them to the engineers, who then change them as they see fit. Anger over this has been building for three weeks.

What do you do?

SCENARIO 2

Before becoming the ScrumMaster, you were a technical lead and respected for your design and programming skills.

In your new project, the team comes to you for advice on a challenging architectural choice. The team members have been arguing between two approaches, but could not agree which one to choose.

They want you to choose the approach.

What do you do?

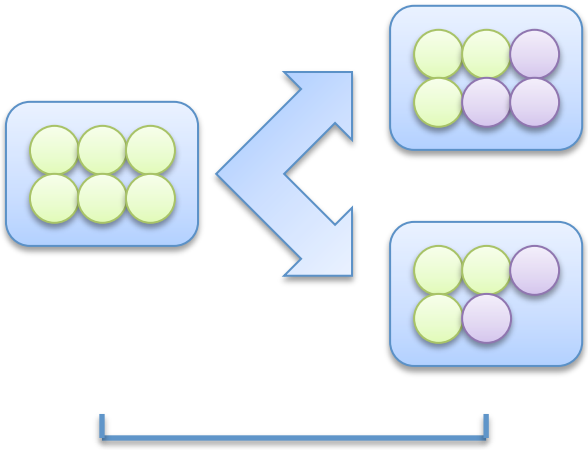
SCENARIO 3

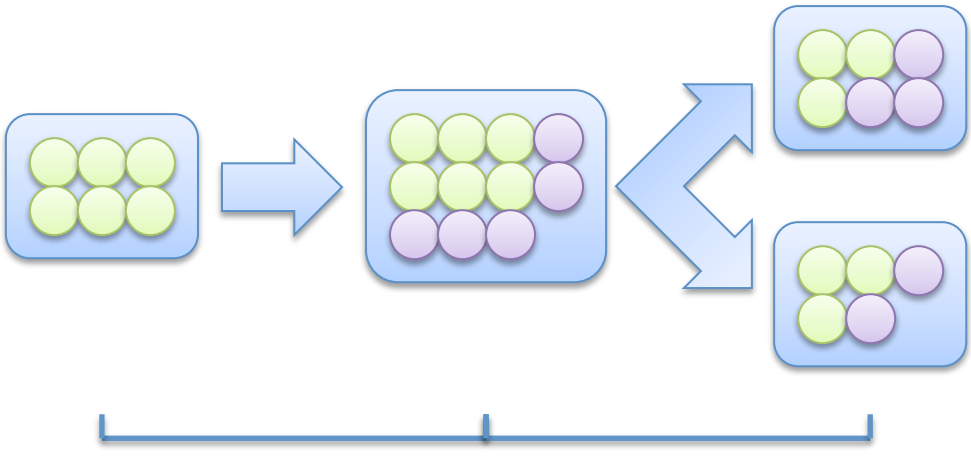
You are the ScrumMaster. Everyone on the team except John meets with you. They tell you that John is not doing his work, is offensive, is difficult to work with, and they want you to fix the problem.

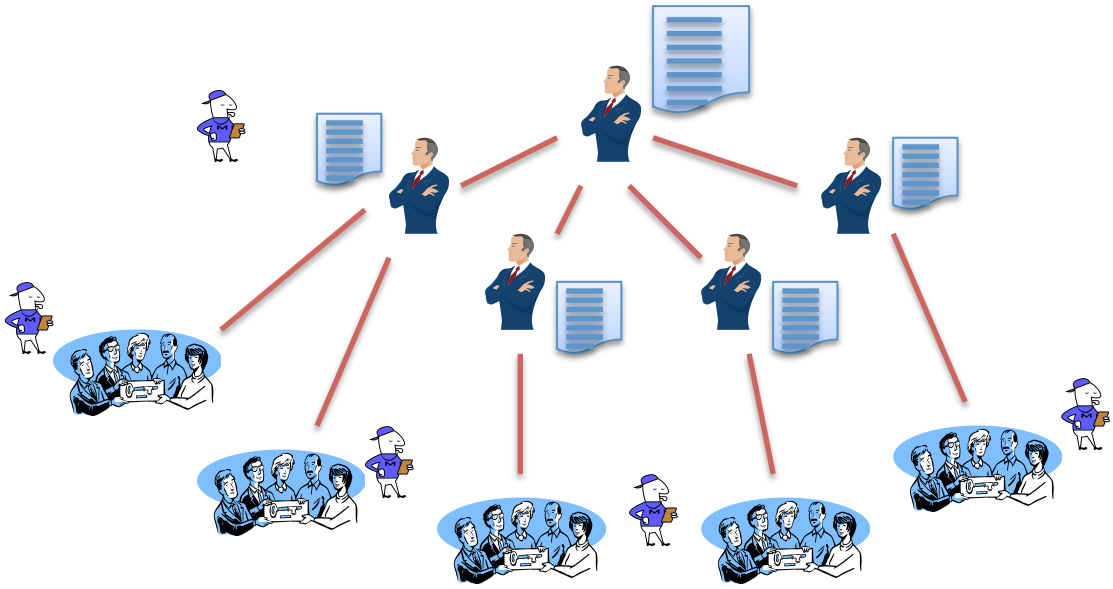
What do you do?

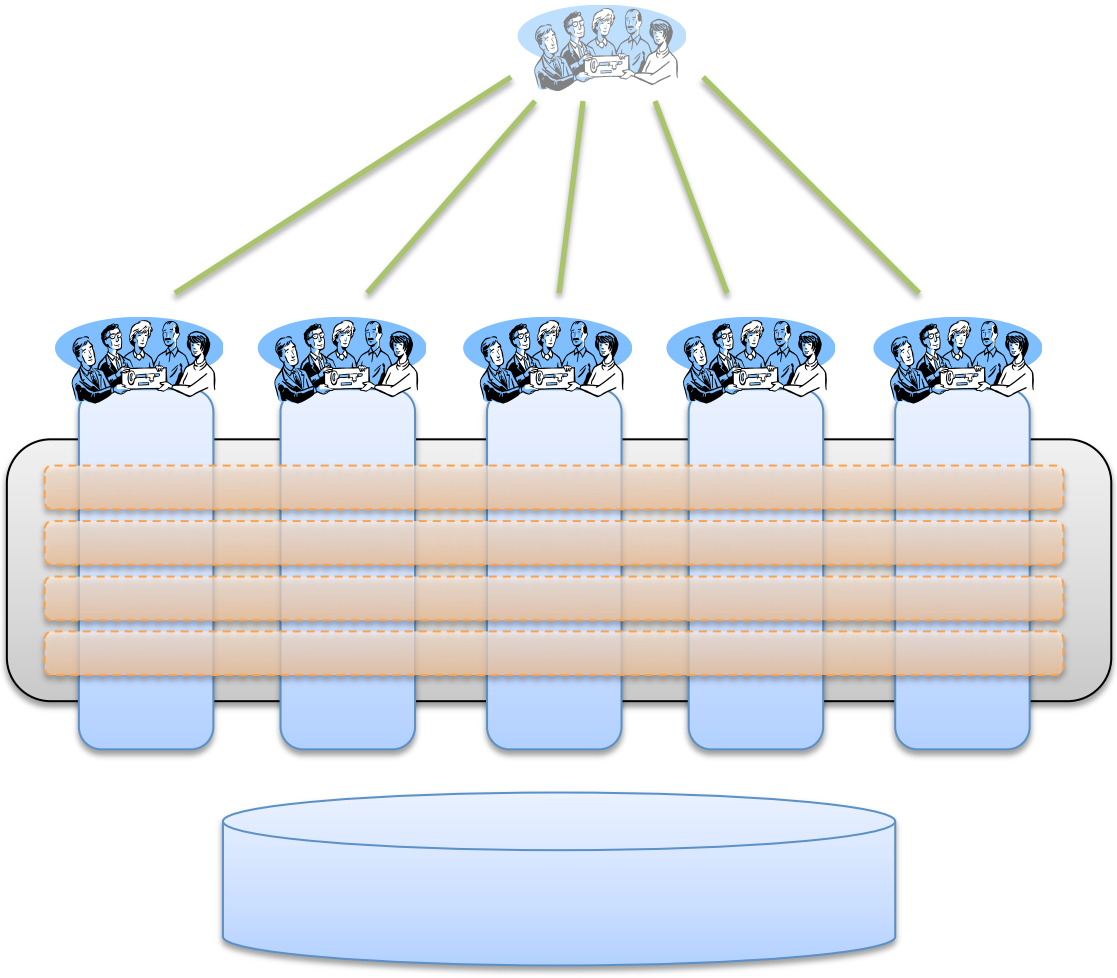
SCALING SCRUM

Please add your comments regarding scaling to the diagrams below.







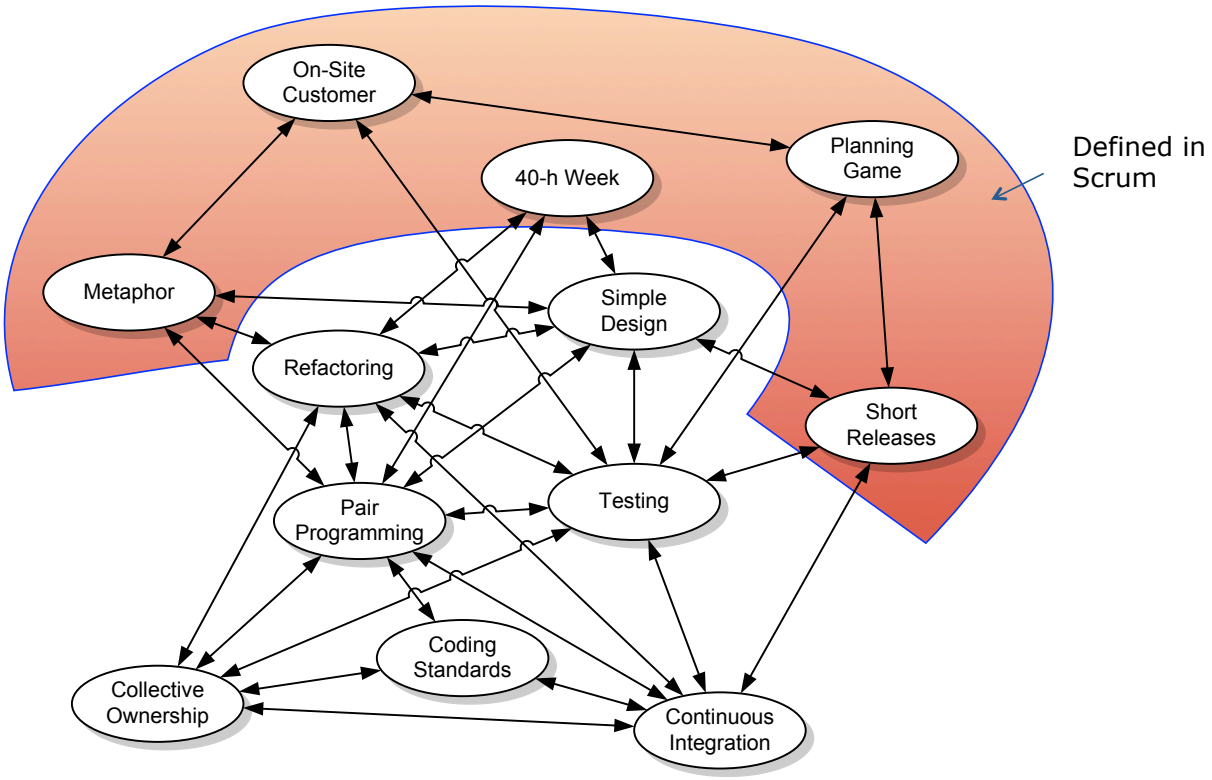


TECHNICAL PRACTICES (XP)

Extreme Programming (XP) forms a solid methodology for development of software systems. Unlike Scrum, it is therefore tied to software development domain. Given that it is assumed that a Scrum team will seek ways in which they can effectively deliver a new tested version of the system in every sprint, it is expected that the team adopt XP or similar technical practices through continuous inspect and adapt cycles. Unfortunately, it is not always the case.

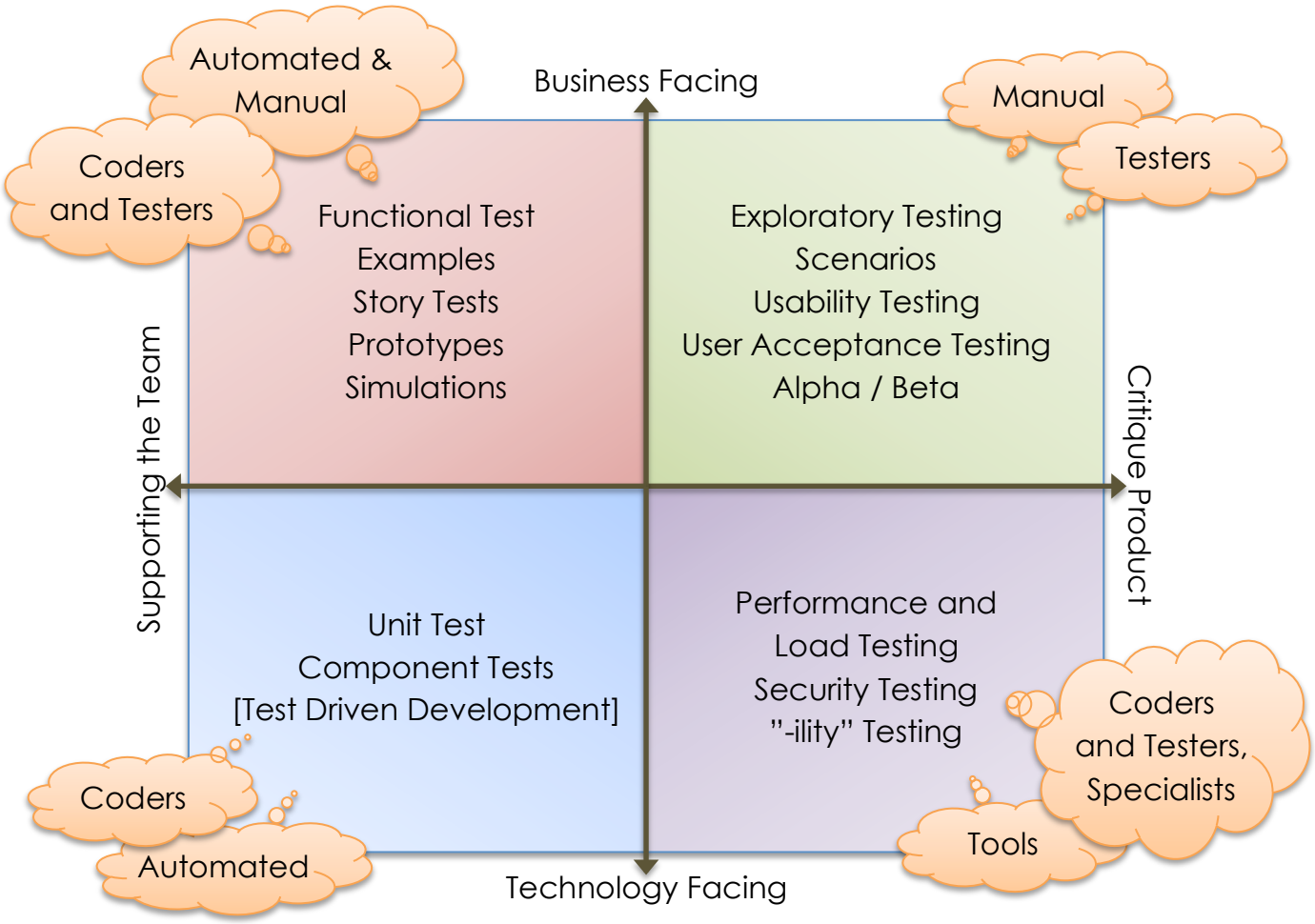
XP consists of a set of development practices, each of which is simple and insufficient in itself, but becomes very powerful when combined with and supported by other XP practices.

The practices, and their key dependencies, are:



"I've never seen or heard of a hyperproductive team that wasn't doing the eXtreme Programming practices (as described by Kent Beck, Ron Jeffries, etc.)."

Michael James, http://danube.com/system/files/A_ScrumMaster's_Checklist_blog.pdf



Adapted from Janet Gregory's version of Brian Marick's original diagram

MANAGING MAINTENANCE

Scrum is fundamentally a batch-driven approach. It works best when work can be allotted into those batches. However, many environments have rapidly changing priorities and fast response times, and it's impossible to wait for the next Sprint to act on that in a managed way. Fortunately, there are excellent continuous flow frameworks available.

In the space below, please make notes as we discuss the issue and look at the Kanban framework. You can also draw yourself a simple Kanban board.

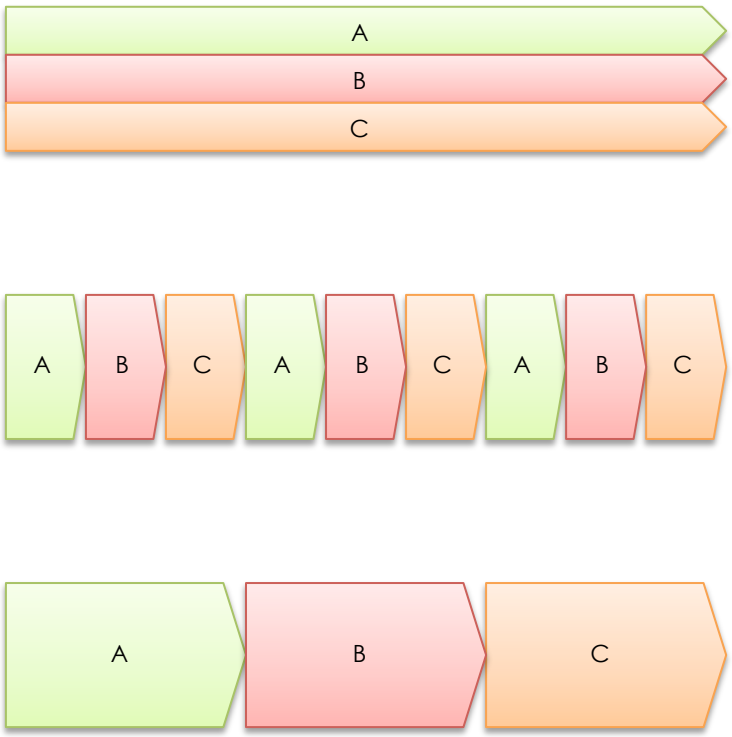
MULTI-SITE SCRUM

In your table group, discuss problems caused by distributing Scrum teams and projects across multiple sites and possibly across multiple time zones.

There are many ways these problems can be mitigated, but the reduced productivity from distribution is very difficult to remove entirely. As we discuss the various possible techniques, write down the key mechanisms below.

OPTIMIZING WORK

Assuming that the team has three features (A, B, and C) to do, each taking an equal amount of effort and providing equal value to customers, which of the following approaches is the (theoretically) most efficient and least risky way to deliver those features? Why?



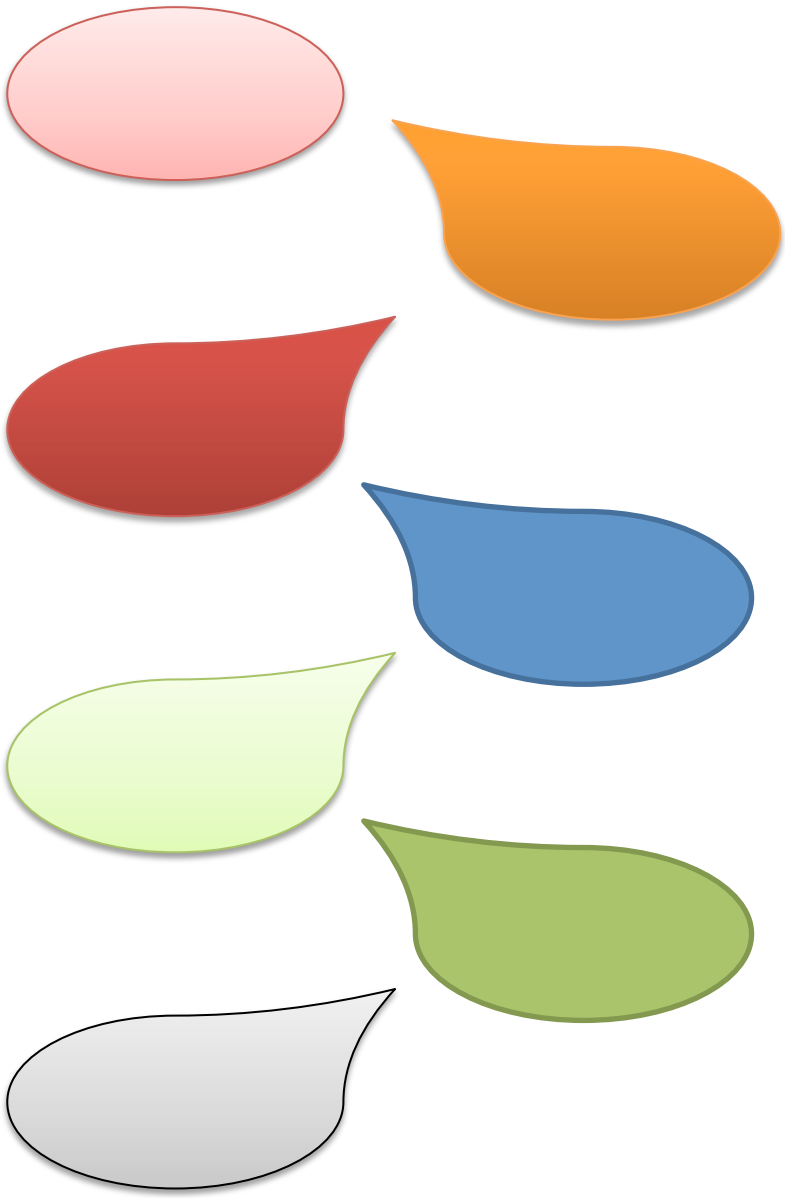
In each of the approaches, how would you calculate the amount of value delivered?

In your table group, discuss the key risks and challenges in each of the approaches. In what kind of situations would you select them?

How does this relate to the tasks the development team has planned for itself?

ELIMINATING WASTE

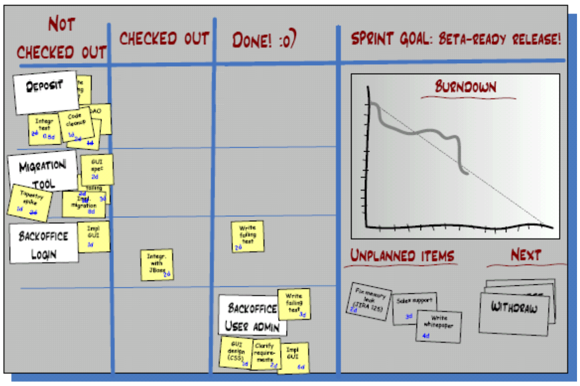
The seven categories of waste (as defined by Tom and Mary Poppendieck, in Lean Software Development):

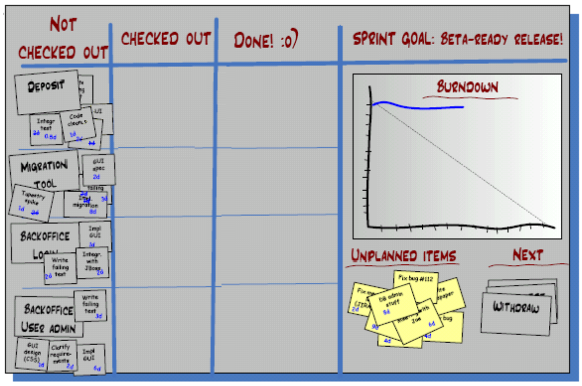


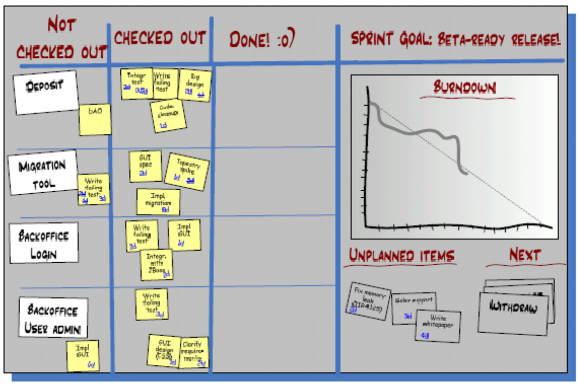
In your table group, discuss the seven forms of waste and identify at least one example for each category. Then discuss how those wastes could be eliminated (or at least significantly reduced).

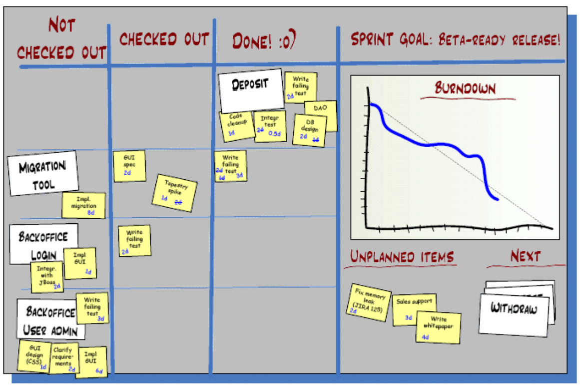
TEAM WALL CHARTS

As we go through the following examples of team wall charts together, we'll discuss what observations you can make in each and what possible problems they highlight.









Original copyright for the diagrams: Henrik Kniberg

AGILE CONTRACTS

In my experience, Agile software development can be done using largely the same contracts as traditional development. However, there are certain things that is recommended to be different. The following additions can be added to virtually any contract.

DEFINITIONS

We mutually agree on working together, and thereby build trust in each others expertise.

Customer Participation in Scrum Team:

The Customer is expected to be active in the project. The role of the Customer includes the following:

- Prioritize features by business value and have them implemented in order of maximum value
- Mutually agreed estimates for all work items. The official representatives of both parties need to agree. This needs to be noted in a signed addendum to the contract for each change.
- Participate in each Sprint planning meeting by discussing the selected features with Company team, including answering questions to provide clarification to the team.
- Participate in writing the conditions of satisfaction for each feature, so the team and client have a shared definition of when a feature is done. These conditions should be completed as part of a user story before any code is written.
- Participate in each Sprint review meeting, and provide timely feedback both for both work-in-progress and completed work.

***To company: Things to ensure are defined in the contract:

- Total value of the contract
- Rates for time and materials billing
- Scope of the contract

CLAUSE: EARLY TERMINATION (MONEY FOR NOTHING)

The Customer may terminate the contract at the end of any Sprint. The standard metric for termination is when the Customer perceives the cost of continuing the project is higher than the additional value received. The Customer will pay Company 20% of the remaining contract value to exercise early termination.

Company commits to delivering 80% of the project scope as high quality by the agreed upon delivery date. High quality is defined by the agreed upon Definition of Done.

This clause can only be enacted if the Customer maintains Participation in the Team Scrum during the project.

In the event that both parties cannot mutually agree on work item estimates or that the Customer does not maintain participation in the Scrum Team, the contract shall revert to a time and materials billing.

CLAUSE: CHANGE FOR FREE

If the Customer maintains Participation in Scrum Team during the entire project, Customer shall be able to make changes to the Scope without incurring any additional cost if total Scope of contracted work is not changed. New features may be added for free at Sprint boundaries if items of equal scope are removed from the contract.

http://www.coactivate.org/projects/agile-contracts/money-for-nothing-change-for-free

In your table group, discuss the impact these two clauses would have on a project contract.

RETROSPECTIVE

In your table group, hold a simple retrospective regarding this course. Please select one of the template structures below.

THE GOOD, THE BAD, THE IMPROVEMENT

In this template structure, there are three rounds around the table. On each round, each participant mentions one thing, and only one, that hasn't been mentioned before.

On the first round, mention something that went well on the course.

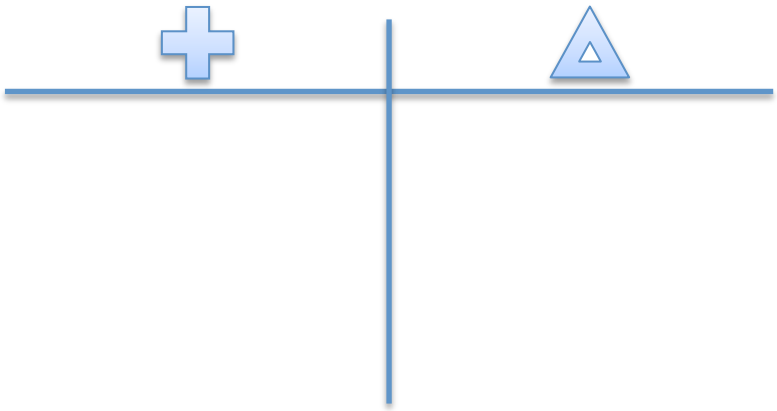
On the second round, mention something that didn't go so well.

On the third round, make a concrete suggestion as to how to improve the course.

Choose someone in the group to act as a record keeper and to write the items down.

PLUSES AND DELTAS

In this template structure, take a flipchart sheet and draw the figure shown below on it. Alternatively you can mark corresponding areas on your group table.



Using post-it notes, take a minute or two to silently write down two kinds of things, one item per note:

- Things to keep (plusses, i.e. things that were good about the course)
- Things to change (deltas, i.e. things to either modify, stop using, or start using)

Once written, group similar notes together on the diagram. Look at the diagram and discuss what do you see. Any surprises? What issues stand out? What are the key recommendations for modifications?

RECOMMENDED READING, BLOGS, ETC.

Books I like and that I have found useful in expanding my understanding of Agile:

- Lean Software Development – Mary and Tom Poppendieck
- Leading Lean Software Development – Mary and Tom Poppendieck
- Succeeding with Agile – Mike Cohn
- User Stories Applied – Mike Cohn
- Agile Estimation and Planning – Mike Cohn
- Agile Retrospectives – Diana Larsen & Esther Derby
- Agile Project Management with Scrum – Ken Schwaber
- Extreme Programming Explained – Kent Beck
- Scaling Lean & Agile Development: Thinking and Organizational Tools for Large-Scale Scrum – Bas Vodde & Craig Larman
- Agile and Iterative Development: A Manager's Guide – Craig Larman
- Management 3.0 – Jurgen Appelo

The above list IS NOT a comprehensive list of good Agile books, just what I've read and liked.

If you want to go outside Agile space to understand Agile organizational behavior (and what Agile organizations could possibly look like), here are some books I've liked:

- Maverick – Ricardo Semler
- Seven Day Weekend – Ricardo Semler
- Leading with LUV – Ken Blanchard, Colleen Barrett
- Delivering Happiness – Tony Hsieh

Petri Heiramo
CollabNet Certified Scrum Trainer (CST) and Agile Transformation Mentor

I'm particularly interested in looking at Scrum and Agile as a holistic approach to improving the competitiveness of businesses. I find myself often in discussions where I emphasize the need for businesses to focus on generating value (over managing costs) as a means to success. Be it in a project or in an organization, I frequently see dysfunctions where silly savings are made at the cost of significant value. And many times the decision makers are totally unaware of the impact of their decisions. It is this mismatch between intent (to work for the best of the company) and action that I seek to eliminate. When focusing on value, we need to focus on customer delight, job satisfaction and continuous improvement.

I've studied Agile methods actively since Fall 2005. In addition, to leading Scrum and Agile projects, I have been doing Scrum and Agility training and consulting since early 2006. I was certified as Scrum Practitioner in April 2007 and as CST in November 2008.

I have over 10 years of experience in developing software as a developer, project manager, QA manager, ScrumMaster, process developer, coach and trainer – and experience with both the traditional and Agile approaches to software development. I've seen a wide variety of projects, and as a result, have come to appreciate the positive effects of Scrum on the human side of software development. Seeing people enjoy their work again has been my greatest joy in Agile deployments. Of course, seeing a happy and engaged customer is great, too, and so is instilling a new kind of pride and common sense in the work people do.

Transform with Experience

CollabNet's ScrumCORE™ training division has helped hundreds of companies successfully adopt and scale Scrum and Agile. As the largest Scrum Alliance certified training facility, CollabNet's Certified Scrum Trainers possess deep experience leading organizations — from small businesses to multinational enterprises — through Agile transformations. With a full range of training services, CollabNet's ScrumCORE offerings can meet the needs of any organization.

The ScrumCORE team conducts regularly scheduled public courses in major markets throughout North America and Europe, offering ScrumMaster and Product Owner Certification courses. Additionally, CollabNet's ScrumCORE trainers are available for on-site private coaching engagements. These coaching sessions connect organizations with an experienced Scrum coach, who can lead them through a transformation or help them resolve specific challenges.

For more information, visit http://www.open.collab.net/training/scrummaster, or to speak with a training specialist contact scrumtraining@collab.net.

CONTACT US

Corporate Headquarters
8000 Marina Blvd,
Suite 600
Brisbane, CA 94005
United States
Phone: +1 (650) 228-2500
Toll Free: +1 (888) 778-9793

Chennai, India Office
Phone: +91 44 4220-3700

Shanghai, China Office
Phone: +86-21-61221082

Seoul, Korea Office
Phone:+82-2-722-8271

Tokyo, Japan Office
Phone:+81-3-5789-5366

London, UK Office
Phone:+44 (0) 207-397-8690

Munich,Germany Office
Phone: +49 (89) 24218-442