

# Agile 2008

Agile Development in Globally Distributed Team

# Agility Experts



# Overview

- Globally Dispersed Teams and their Challenges
- Communication
- Suitability of Agile Development
- Introducing an Agile Methodology
- Trust
- Iteration Planning and Evaluation
- Project Visibility and Release Planning
- Distributed Meetings
- Coping with Fatigue
- Essential Development Tools
- Agile-Trac
- Questions



# Overview

- **Globally Dispersed Teams and their Challenges**
- Communication
- Suitability of Agile Development
- Introducing an Agile Methodology
- Trust
- Iteration Planning and Evaluation
- Project Visibility and Release Planning
- Distributed Meetings
- Coping with Fatigue
- Essential Development Tools
- Agile-Trac
- Questions

# What is a Globally Distributed Team?

- A single team on a single project with a common goal
  - Team boundaries vary across organisations
    - For example, some teams may or may not include QA and Technical Writers
- Most of the team is dispersed geographically
  - Typically the team members are in different countries
  - Typically there are strong cultural differences
  - Typically member's native languages may differ
  - Even if native languages are the same, cultural differences are likely to have an impact
- Typically team members are in different time zones
  - The common time window is likely to be small possibly only a couple of hours a day, if that

# Challenges faced by a Dispersed Team

- Communication burden is almost overbearing
- Clear communication is compounded by cultural and language differences
- Difficult to build trust among team members
- Hard to share and maintain tacit knowledge
- Synchronisation is often problematic
- Ensuring fair and accessible participation to all team members
- Maintaining a shared vision across all members often a challenge
- Providing visibility into progress for stakeholders both inside and outside of the organisation
- Creating and sustaining a team identity, both within the team and throughout the organisation



# Most Challenges relate to Communication and Trust

- Any methodology that addresses these issues will be useful in a distributed environment
  - Agile methodologies recognise the importance of these
- The basis of trust is honest, respectful and clear communication
  - Once gained it must be nurtured
- Identify and utilise as many forms of communication as is practical
  - Know what the default mediums of communication should be in common situations
- Recognise that you will not have face to face communication and identify a de-facto replacement
  - This will almost always be the telephone
  - Invest in headsets

# Overview

- Globally Dispersed Teams and their Challenges
- **Communication**
- Suitability of Agile Development
- Introducing an Agile Methodology
- Trust
- Iteration Planning and Evaluation
- Project Visibility and Release Planning
- Distributed Meetings
- Coping with Fatigue
- Essential Development Tools
- Agile-Trac
- Questions



# Communication, Communication, Communication

- There are a number of communication mediums at your disposal
  - e.g. Email, IM, Telephone, Wikis, Web Meetings, Video ...
- Each of the mediums contributes in a different way to the quality and usefulness of the communication
  - Is it Retrievable, Facilitates Real-time Dialogue, Formal, Intimate, Interactive, Acknowledged, Involving, Far reaching ... ?
- Essentially most synchronous communication will be a meeting
  - Disciplined approach to meetings is therefore essential
- The golden rule for all communication is that all participants must use a medium of the highest **common** bandwidth
  - In other words if some team members can meet in the same room and one must dial into the meeting then ALL members should dial in and none should meet in the room

# Learn to make the most of the Basic Communication Tools

- Five basic tools every dispersed team should have
  - Telephone
  - Email
  - Instant Messaging
  - Web-based Workspace Sharing Tool (like Webex)
  - Wiki
- By combining these tools in a variety of ways it is possible to create a rich communication environment
  - Telephone (for real-time dialogue)
    - + Web-based Workspace Sharing (for focus and involvement)
    - + Instant Messaging (to maintain a shared view and sidebar with)
    - + real-time Wiki updating (for retrievable data storage and reach)
  - Can be a very effective combination of mediums, and superior than a round-table face-to-face meeting in some situations

# Asynchronous Communication

## Email is Not Dead

- You will be interfacing with stakeholders and one of the few mediums you are guaranteed is email
  - Don't just discount it
  - Learn to make best use of it
- Use a group email address like [team@company.com](mailto:team@company.com)
  - Emphasises identity
  - Makes threads easier to track
- Encourage all team emails to be sent to this address
  - Fosters open communication
  - Builds trust
- Consider alternatives like newsgroups but ensure they are used
  - If you can use a web searchable archive do so



# Email is Not Dead

## Try to Keep it That Way

- When referencing project pages (such as a wiki) always use direct links
- Make sure links are not broken across multiple lines
  - Use <> but beware of sanitizing in clients
- Define an etiquette for email and at least ensure the team adopts it
- For example, adopt something similar to newsgroup use
  - No top-posting
  - Plain text only
  - Separate email for different topics
- May be hard to achieve with stakeholders as many have esoteric email habits
  - But worth the effort to try

# Semi-Synchronous Communication

## Instant Messaging

- Instant Messaging has several important uses
  - Basic semi-synchronous communication
  - A side-bar channel during meetings
  - An indicator of presence
- An IM client as an indicator of presence is very important in a dispersed team
- For it to be useful it must be used with some degree of honesty
  - Don't stay logged in all the time or always set your status to busy
- Instant Messaging helps create a feeling of closeness
  - When you want to wander over to someone's desk you typically use an instant message as a lightweight alternative
- Team members should be approachable through this medium
  - Define an etiquette if you must but remember it's a chat medium

# The Humble Telephone and Web Meetings

- Invest in headsets
  - Long meetings will be very fatiguing otherwise
  - Headsets keep hands free for interaction
- Invest in calling solutions that allow unlimited phone use
- Always check line volumes in meetings
- Supplement telephone calls with visual synchronisation
- Web meetings provide opportunities to improve meetings
  - They provide a common focus to help maintain attention
  - Facilitate co-operative working
  - Make virtual white-boarding possible (to some degree)
- Invest in unlimited web meeting capabilities
  - There are many options, ensure desktop sharing is supported



# Invest in Good Communication Infrastructure and Know its Limits

- No matter how effective you may be at utilising your available communication mediums you will be defeated easily by poor infrastructure
  - Poor quality telephone lines
  - Dropped calls
  - Lengthy set up times when joining web meetings
  - Intermittent availability
  - Tennis elbow from lack of a headset
  - Restricted bandwidth
- When an issue is identified that can't be addressed immediately make addressing it a priority at the next retrospective
- Develop strategies to work around problems
  - Use dry-runs, early preparation, or other methods

# Overview

- Globally Dispersed Teams and their Challenges
- Communication
- **Suitability of Agile Development**
- Introducing an Agile Methodology
- Trust
- Iteration Planning and Evaluation
- Project Visibility and Release Planning
- Distributed Meetings
- Coping with Fatigue
- Essential Development Tools
- Agile-Trac
- Questions

# What do I mean by Agile Development?

- Agile Development is development that supports the Agile Value System. The Agile Value System is based on a set of Agile Principles.
  - I am not focusing on any particular agile methodology
- In essence Agile Development seeks to cope Deterministically with Change
  - Progress is measured against business value delivered to the customer
  - Change is welcomed as a sign of better understanding of the business need
  - Effort is only spent as needed to deliver business value - minimising effort wastage in the face of change
  - Past observed progress is used to predict future progress
- Emphasis is on delivering Business Value in a sustainable way



# Agile Value System

## Left valued more than Right

- Individuals and interactions
  - over processes and tools
- Working software
  - over comprehensive documentation
- Customer Collaboration
  - over contract negotiation
- Responding to change
  - over following a plan

# Agile Principles

- The Value System is based on the following principles:
  - Early and continuous delivery of valuable software
  - Welcome changing requirements
  - Deliver working software frequently
  - Business people and developers work together
  - Trust motivated individuals
  - Working software is the primary measure of progress
  - Promote sustainable development
  - Technical excellence and good design
  - Simplicity is essential
  - Self-organising teams
  - Team reflection and adjustment

# Suitability of Agile Development with Dispersed Teams

- Myth: Agile methodologies require a large communication bandwidth. This is not true. Agile methodologies try to maximise use of the available bandwidth
- Agile places an emphasis on communication and trust building
  - Individuals and Interactions over processes and tools
  - Trust motivated individuals
  - Self-organising teams
  - Team reflection and adjustment
  - The goal of many agile practices is to enhance the quality of communication and create trust
- Contrast traditional development methodologies which emphasise plan following, process adherence and peripheral documentation to show work is being done.
  - Little is said of trust or quality communication



# Overview

- Globally Dispersed Teams and their Challenges
- Communication
- Suitability of Agile Development
- **Introducing an Agile Methodology**
- Trust
- Iteration Planning and Evaluation
- Project Visibility and Release Planning
- Distributed Meetings
- Coping with Fatigue
- Essential Development Tools
- Agile-Trac
- Questions

# Create, Capture and Share a Common View of the Methodology

- There must exist a common language amongst the team and stakeholders that captures the methodology
- This will evolve as the project evolves and retrospectives help identify areas for improvement
- An important part of team building should be the team jointly identifying a basic methodology that each member feels he can live with
- Be wary of mandated or canned methodologies
  - Learn about existing methodologies and choose elements that everyone on the team feels comfortable with
  - If a canned methodology fits, by all means start with that, but do evolve it to suit the team
- There are certain fundamentals of any methodology that should almost always be adhered to

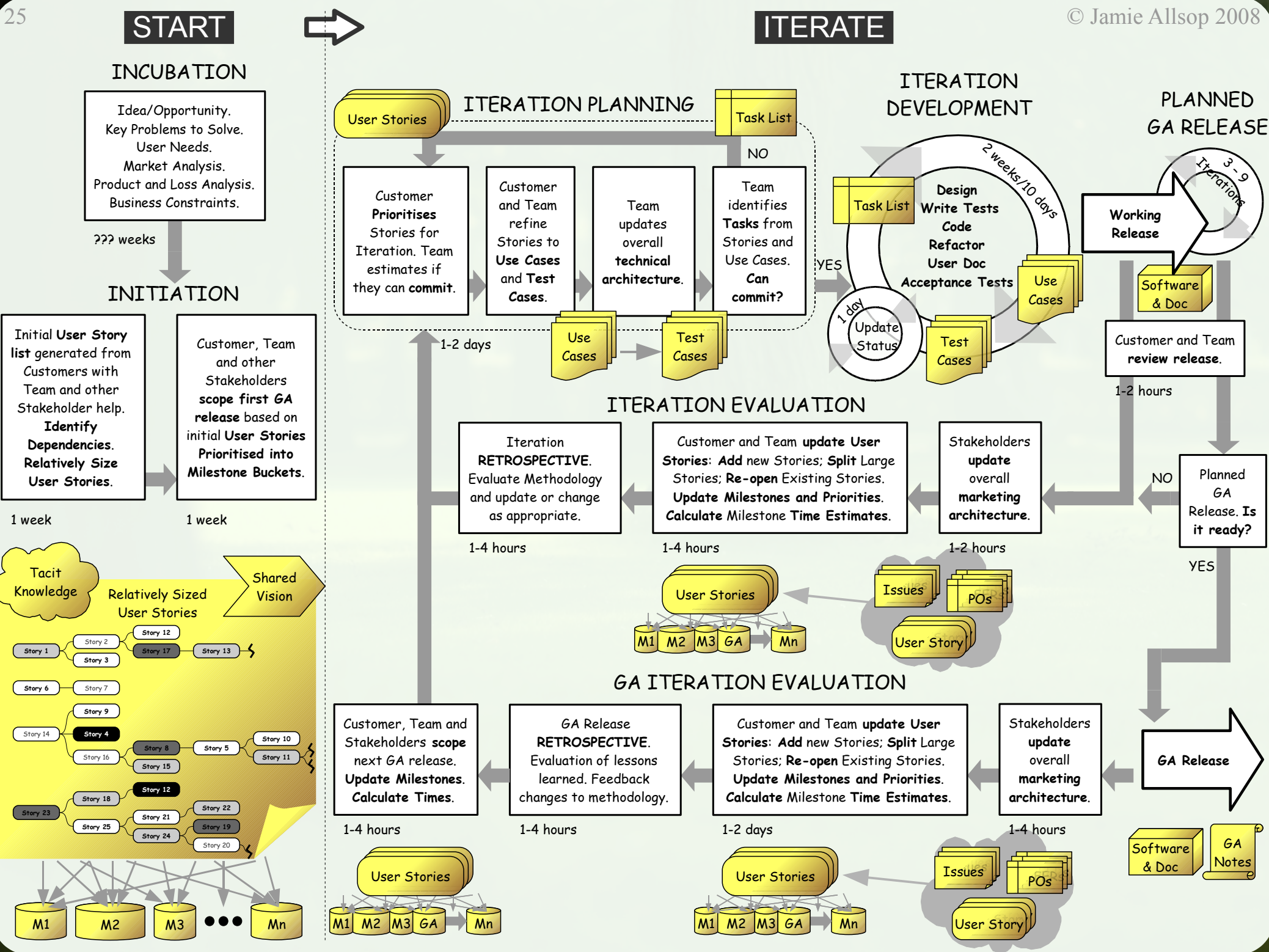
# Methodology Fundamentals

- Develop in iterations
  - Iterations should be time boxes of a fixed length
    - This allows planning and estimation
    - Creates a rhythm
  - Iterations should start with a planning phase
  - Iterations should end with a retrospective
- The methodology should support release planning and iteration planning
  - There should be a uniform way of representing features in both iteration planning and release planning, such as user stories
  - Size your stories (features) to allow estimation of future progress
- Releases should finish with a retrospective



# Share the Methodology and its Vocabulary with Stakeholders

- Make sure all your stakeholders can visualise your methodology
  - Draw or write it out and make it available
- It is important that stakeholders know where in the methodology you are at any given time.
  - This is an anchor point into the teams day-to-day progress
- Ensure all stakeholders are clear on vocabulary that you use
  - For example, don't overload on common terms like release. Use milestone instead of release and reserve release for the classical business use
- Spend some time doing Q&A with stakeholders
  - This provides an early chance for the team to introduce its identity to the stakeholders
- For example...



# Creating a Team Identity

- There needs to be somewhere people can 'go' to see what the team is doing
  - A website is the best way to do this
    - Wiki Webs are ideal
    - Tools like Trac (<http://trac.edgewall.org/>) are designed for this
- Ensure all meetings with stakeholders have all team members
  - It is important that the team is seen as a team
- The interface to the team should be the whole team, not an individual member of the team
  - Avoid having a project manager who acts as a communication proxy as this simply adds unnecessary indirection
- It may be useful to have a team page on the team site introducing the team members



# Creating a Project Identity

## The Whole is the Sum of its Parts

- A Shared Vision is not the same thing as a Project Vision
- A shared vision becomes more concrete as the project progresses and stakeholders understand better the product being developed – it is a Shared Understanding
- A project vision can be a great way to anchor the shared vision, but it must evolve
  - At the beginning it is unlikely that we know exactly what we need
  - A project vision therefore becomes vague
- Try to define the project as the sum of its stories and the dependencies between them
  - Stories can capture both functional and non-functional requirements
- Provide launch pages on your team site that offer these as up-to-date views

# Overview

- Globally Dispersed Teams and their Challenges
- Communication
- Suitability of Agile Development
- Introducing an Agile Methodology
- **Trust**
- Iteration Planning and Evaluation
- Project Visibility and Release Planning
- Distributed Meetings
- Coping with Fatigue
- Essential Development Tools
- Agile-Trac
- Questions

# Meet First Then Work From Afar

- It is almost always better to meet first before working together remotely
  - The longer the initial meeting the stronger the foundation of trust
- However, with many dispersed teams this is not an option
  - We had a 1 week meeting in 2 and half years, after 1 year
- You can still build a trusting, mutually respectful relationship without meeting face to face
  - But it requires a lot more time and effort
  - Important to have ground rules for behaviour
  - Important to humanise the relationships between team members and to encourage one-on-one time (like paired working) to develop closer relationships
  - Sometimes the personalities that people project through constrained mediums are easier to get along with



# Trust, Mutual Respect and Communication

- Building a trusting relationship requires time, effort and patience
- In the beginning you have to trust each other
  - But trust on blind faith is too easily broken
- You must use effective communication to help build a foundation for the trust
  - Many of the ideas presented here are designed to strengthen the foundation of trust
- Mutual respect is a key to helping build trust
- Iteration retrospectives are essential
- Consider internal (team) and external (stakeholder) retrospectives
  - Encourages very honest discussion within the team
  - Presents a unified message to stakeholders when required

# Ground Rules for Trust

- Remember broken trust in a distributed environment is very hard to fix
  - The damage is often permanent
- Therefore identify grounds for minimising the opportunity to break trust carelessly
  - Never allow participants to sneak into meetings
  - Don't allow team members to blame each other
  - Always accept responsibility as a whole team
  - Never allow members to make or change team decisions individually
  - Pay attention to unhappy team members and address the issue as early as possible
  - Remember to give each other the benefit of the doubt

# Overview

- Globally Dispersed Teams and their Challenges
- Communication
- Suitability of Agile Development
- Introducing an Agile Methodology
- Trust
- **Iteration Planning and Evaluation**
- Project Visibility and Release Planning
- Distributed Meetings
- Coping with Fatigue
- Essential Development Tools
- Agile-Trac
- Questions



# Iteration Planning and Evaluation

- Expect these to take longer than in a co-located environment
  - Communication bandwidth is lower so progress is slower
  - Bandwidth constrained communication is fatiguing so meetings cannot last for too long
  - Sometimes stakeholders will have limited time. That coupled with the restricted common time window means sometimes multiple meetings are required where a single meeting may have sufficed
- Try your best to book meetings in advance and ensure required stakeholders will be present
  - With small time windows you cannot afford to miss meeting slots otherwise you could lose a day or more of time
- Establish a set rhythm of meetings so that stakeholders know what to expect
- Allow time for reflection between meetings

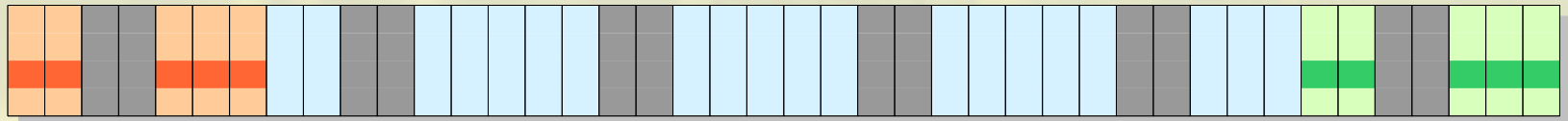
# Iteration Length

- Length should be fixed, though you may need to change to a new fixed length based on retrospectives
- Planning and evaluation duration should not be disproportionate to development duration
- Planning and Evaluation can take longer in a distributed setting so very short iterations, say 1 week, may not be practical
- Very long iterations, say 4 to 6 weeks have too much silence and set expectations too high
- Early in a project slightly longer iterations may make sense with more architectural work being done
- We found iterations of 2 to 3 weeks were appropriate
- Generally dispersed teams favour shorter iterations
- Shorter iterations provide a higher resolution on project visibility

# Planning & Evaluation : Development Ratios for Iteration Length

Lower  
Resolution

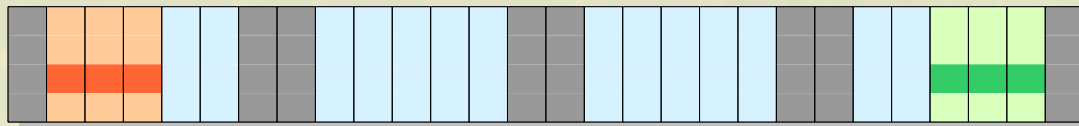
6 weeks  
42 days



Planning and Evaluation = 33%

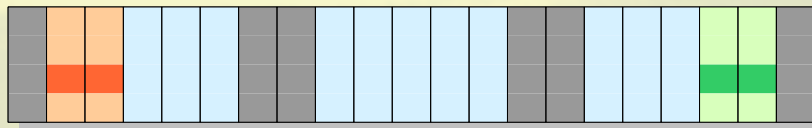
Note: Development never finishes on a Friday

4 weeks  
28 days



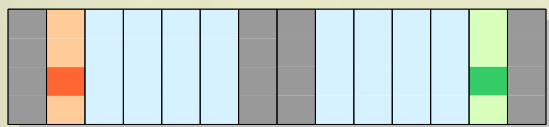
Planning and Evaluation = 30%

3 weeks  
21 days



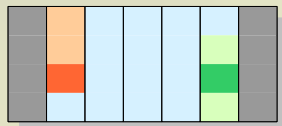
Planning and Evaluation = 27%

2 weeks  
14 days



Planning and Evaluation = 20%

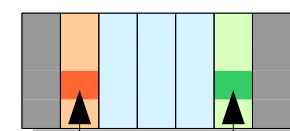
1 week  
7 days



Planning and Evaluation = 30%

Higher  
Resolution

Su M Tu W Th F Sa



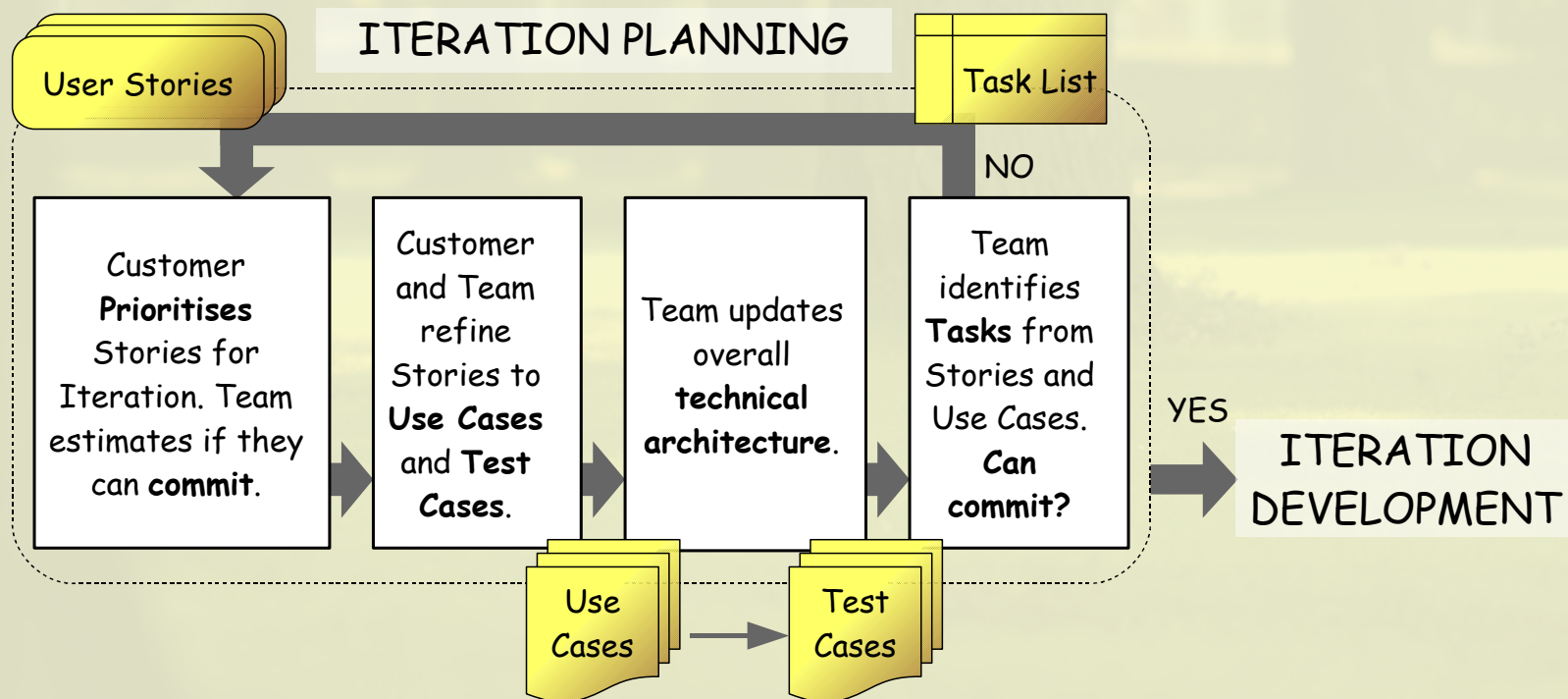
Planning  
Development  
Evaluation

Common Time



# Iteration Planning

- Adopt a workflow with development entry criteria to help guide the planning, for example:



- Use evolving templates to help drive meetings and the expected outcomes
  - Wiki pages are great for this

# An Iteration Planning Page provides the Iteration Details

## ■ Typically this will display

- Iteration Dates
- Iteration Title
- A summary of the iteration
- A list of User Stories the team commits to complete during the iteration
- A list of any other work that will impact the team during the iteration

### Iteration 21 Planning page

#### Proposed dates for the Iteration

- ♦ Start: September 6, 2007
- ♦ End: September 19, 2007

#### Iteration Title

Switch Focus to Drilldown from Exceptions

#### Iteration Summary

The purpose of this iteration is to lay the groundwork for providing drill-down for the released version of NIS as well as on the current development version.

#### Proposed work for the Iteration

Refined User Story	User Story	Relative Size in Points	Title	Visibility	Notes	Status
<a href="#">NMSUserStory26pt4</a>	<a href="#">NMSUserStory26</a>	8	<a href="#">DrillDownUser</a> needs to be able to specify their host credentials	<a href="#">NisRelease1Pt1</a>		■
<a href="#">NMSUserStory26pt5</a>	<a href="#">NMSUserStory26</a>	8	<a href="#">DrillDownUser</a> needs to be validated against the mainframe user database	<a href="#">NisRelease1Pt1</a>		■
<a href="#">NMSUserStory74pt4</a>	<a href="#">NMSUserStory74</a>	5	<a href="#">SystemAdmin</a> can view the version of NIS and specific components (ie, CE)	<a href="#">NisRelease1Pt1</a>		■
<a href="#">NMSUserStory81pt1</a>	<a href="#">NMSUserStory81</a>	5	<a href="#">SystemAdmin</a> needs the Postgresql repository to have VACUUM run periodically.	<a href="#">NisRelease1Pt1</a>		■
SubTotal		26				

#### Maintenance work

Task description	Size
Naviplex maintenance	8
Navigraph maintenance	0
NIS maintenance	5

#### NIS 1.1 - Build and source control management

Task description	Size	Notes
Get testing moved to another machine	13	Continued from Iteration 20
Determine the work necessary for moving TRUNK under BRANCH	3	

#### Notes

# Splitting Stories

- Only allow user stories below a certain relative size to be considered for an iteration
  - For example on the scale, 1, 3, 5, 8, 13, 20, 40, 100, you might only allow user stories with a relative size of 13 or below
- If necessary split user stories into two or more stories of a smaller size
  - This is not always easy but is an important step and builds shared a shared view of the project
  - Always size user stories as a team
  - Can be time-consuming
- Maintain the original user story as a reference so that a hierarchical view of the capabilities of the project can be built
  - This is valuable when trying to communicate the big picture



# Completion Statements

## Improve User Story Clarity

- Provide a Completion Statement for any user story committed to in an iteration
  - A completion statement details what it means for a user story to be DONE, for example, “This is done when the EndUser can...”
  - Completion Statements provide important clarification of a user story and help build a shared view with stakeholders
  - Compensates for not being able to wander over to someone's desk to seek clarification

**User Story 26.5 : DrillDownUser needs to be validated against the mainframe user database**

**Relative Size : 8**

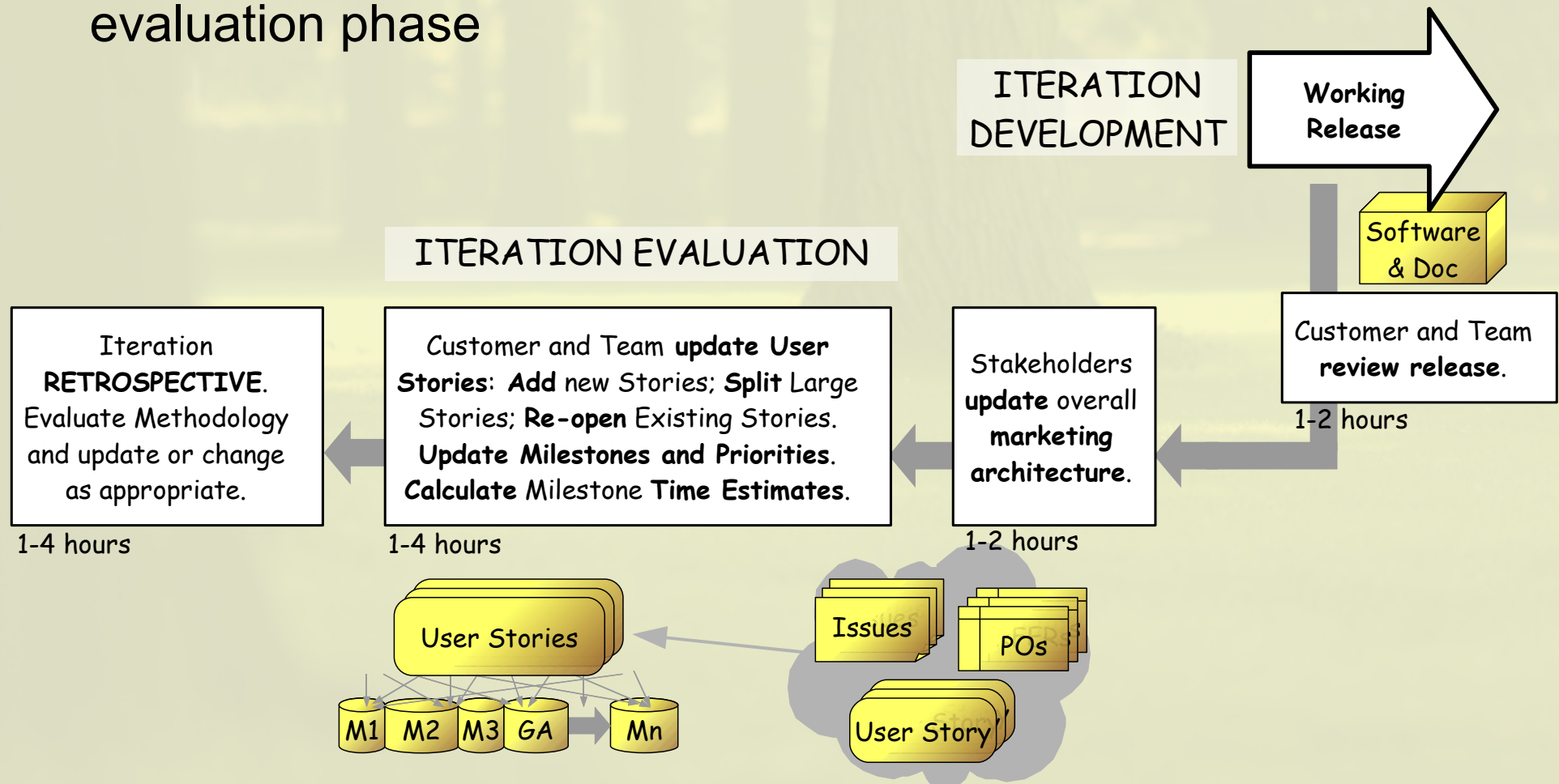
### **Completion Statement**

This user story is done when the user, having entered their credentials in the interface provided by [NMSUserStory26pt4](#), receives feedback that their credentials:

1. were accepted by the mainframe as valid
2. were rejected by the mainframe as invalid
3. could not be validated by the mainframe due to some unexpected circumstance

# Iteration Evaluation

- As with iteration planning have a workflow to help guide the evaluation phase



# Iteration Evaluation

## Retrospectives are Essential

- Use an evolving check-list to help with Iteration Retrospectives
  - But make sure ALL topics are on the table
- Synchronise the team first before including stakeholders
- Trust is easily broken and respect easily lost in a dispersed setting so always remember Kerth's Prime Directive
  - Regardless of what we discover, we must understand and truly believe that everyone did the best job he or she could, given what was known at the time, his or her skills and abilities, the resources available, and the situation at hand
- Hurt feelings and bruised egos are harder to mend in a dispersed team so avoid it
- Gap time between finishing an iteration and having the retrospective can help produce a more considered discussion
  - A day of gap time should be sufficient



# Overview

- Globally Dispersed Teams and their Challenges
- Communication
- Suitability of Agile Development
- Introducing an Agile Methodology
- Trust
- Iteration Planning and Evaluation
- **Project Visibility and Release Planning**
- Distributed Meetings
- Coping with Fatigue
- Essential Development Tools
- Agile-Trac
- Questions

# Project Visibility and Release Planning

- Important to have an easily accessible 'window' into the project
- The most common questions that need answered are:
  - What does the project do? Give us the 'big picture'
    - A list of user stories on their own can be overwhelming, resulting in the classic problem of not being able to see the proverbial wood for the trees
  - What is currently done? In other words, where are we now?
  - What are the future milestones for the project and when are they likely to be complete?
- Time zone differences mean most communication will be asynchronous
- An auto-updating wiki page is a good place to show progress
  - In effect you want a virtual story board

# Communicating what the Project Does

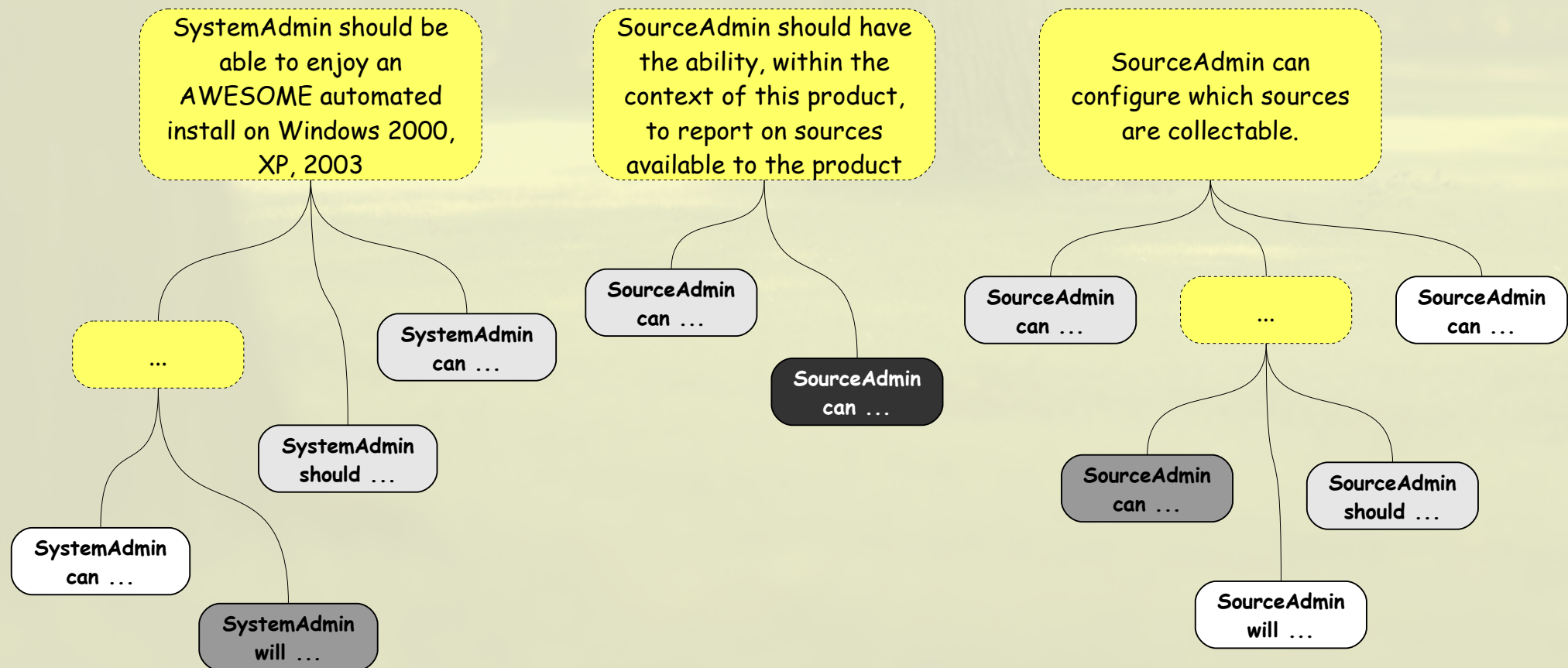
- We have basically two macro views
  - The dependency view of user stories
  - The split tree view of user stories where leaves are actual user stories
- The dependency view shows us how user stories (features) relate to one another with an implicit implementation order
- The split tree view shows us a top-down breakdown of capabilities of the project providing a mapping from the 'big picture' down to individual user stories
- The detailed view of what the project does is captured by user stories and use cases
- User stories can capture both functional and non-functional requirements
- Use a pre-defined set of users to clarify the main actors



# What the Project Does

## User Story Split Tree View

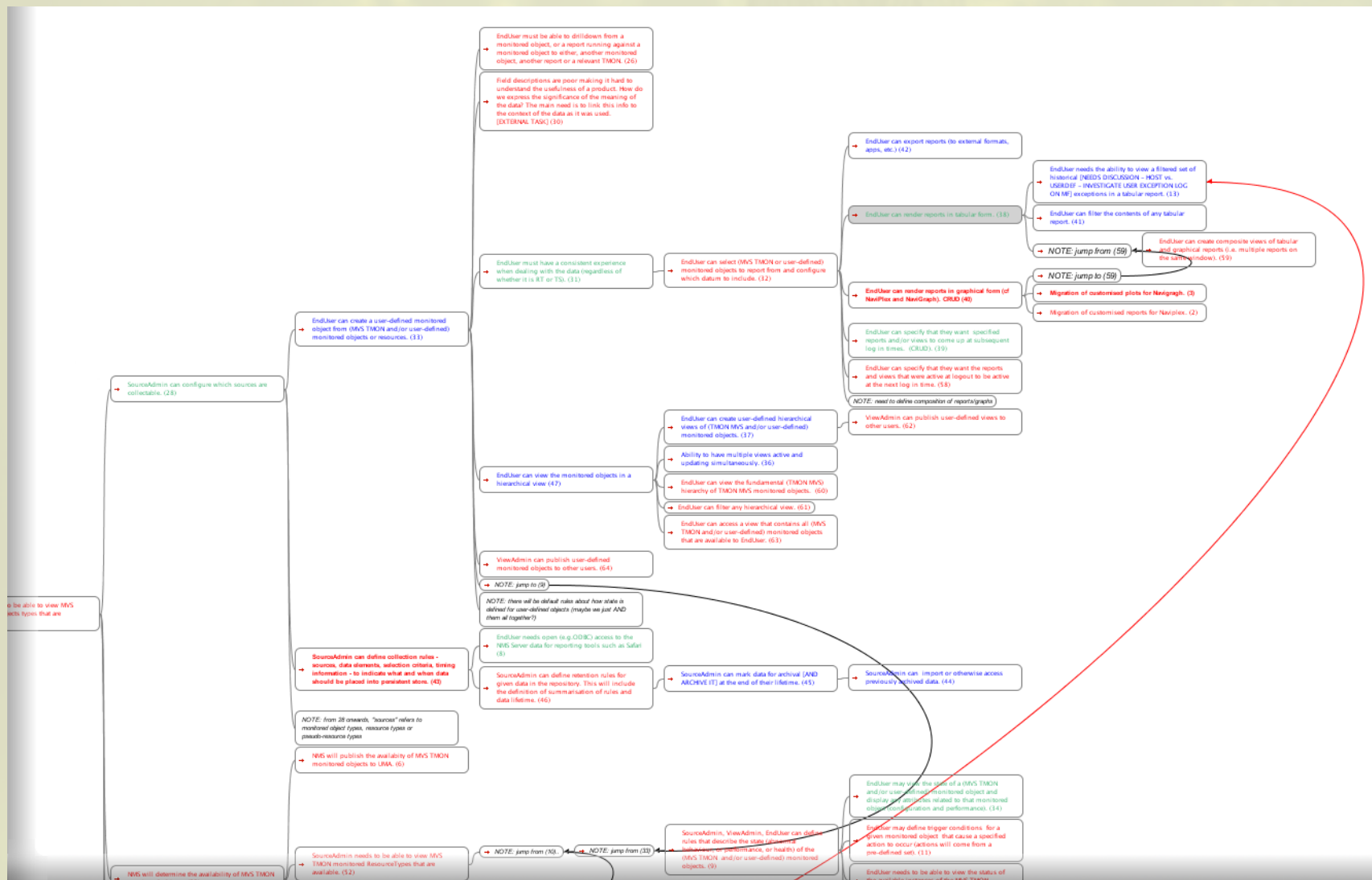
- As user stories are split the tree gets broader and deeper
- The fewer root user stories provide a useful overview
- The high level user stories can be grouped into capabilities



# What the Project Does

## User Story Dependency View

- Part of a user story dependency view - colour indicates size



# What is Done? The Sum of its Parts: Milestones are composed of Stories

- Not all milestones are GA Releases
- Using Named Milestones makes it easy to insert new ones

Milestone **UmaPmoDemo** 26 Points COMPLETED

100%

[Toggle User Stories](#)

Milestone **DexiaBankDemo** 103 Points COMPLETED

100%

[Toggle User Stories](#)

Milestone **NmsUmaPmoMARElease** 219 Points COMPLETED

100%

[Toggle User Stories](#)

Milestone **NisRelease1Pt1** 498 Points

33%

[Toggle User Stories](#)

- ◆ Size completed by team : 168 points
- ◆ Points Remaining: 330 pts ( ~ 12.09 iterations @ 27.3 pts/iteration )
- ◆ Estimated weeks remaining: 31 to 50 weeks.

Refined User Story	Unrefined User Story	Title	Visibility	Size	Test Complete	Need TCOM and QA?	Doc Complete	Acceptance Complete
	004	SystemAdmin should be able to enjoy an AWESOME automated install on Windows 2000, XP, 2003.						
<a href="#">004pt35</a>	<a href="#">004</a>	<a href="#">SystemAdmin</a> will be given the option of installing an <b>external</b> JDK if a suitable JDK is not found.	<a href="#">NisRelease1Pt1</a>	3	<div></div>	Yes	<div></div>	<div></div>
<a href="#">004pt38</a>	<a href="#">004</a>	<a href="#">SystemAdmin</a> will be able to upgrade NIS without having to uninstall.	<a href="#">NisRelease1Pt1</a>	8	<div></div>	Yes	<div></div>	<div></div>



# Virtual Story board: Story Completion is an Immediate Indicator of Progress

## Milestone NisRelease1Pt1 498 Points

33%

### Toggle User Stories

- Size completed by team : 168 points
- Points Remaining: 330 pts ( ~ 12.09 iterations @ 27.3 pts/iteration )
- Estimated weeks remaining: 31 to 50 weeks.

Refined User Story	Unrefined User Story	Title	Visibility	Size	Test Complete	Need TCOM and QA?	Doc Complete	Acceptance Complete
	<b>010</b>	<b>NMS will evaluate exceptions on a continuous basis in the background. There should be no need to run, and attach, a client.</b>						
<a href="#">010pt4</a>	<a href="#">010</a>	<a href="#">SystemAdmin</a> can define active exception timeout (ones not killed by a TMON)	<a href="#">NisRelease1Pt1</a>	1	■	Yes	■	■
<a href="#">010pt5</a>	<a href="#">010</a>	<a href="#">SystemAdmin</a> can define in-active exception lifetime	<a href="#">NisRelease1Pt1</a>	1	■	Yes	■	■
<a href="#">010pt13</a>	<a href="#">010</a>	<a href="#">EndUser</a> will observe that the <a href="#">TmonException</a> state is updated on a timed basis regardless of whether or not new <a href="#">TmonExceptions</a> are being received from the host, that is, <a href="#">TmonException</a> states will age as expected.	<a href="#">NisRelease1Pt1</a>	5	■	Yes	■	■
	<b>026</b>	<b><a href="#">EndUser</a> must be able to drilldown from a monitored object, or a report running against a monitored object to either, another monitored object, another report or a relevant TMON.</b>						
<a href="#">026pt2?</a>	<a href="#">026</a>	<a href="#">UmaUser</a> needs drilldown data from a monitored object (i.e. APPLID) or something to define where to drilldown to. Ensure NIS can identify and persist necessary data, for each TMON, to facilitate drilldown.	<a href="#">NisRelease1Pt1</a>	13	■	No	■	■
<a href="#">026pt3?</a>	<a href="#">026</a>	<a href="#">UmaUser</a> and <a href="#">OdbcUser</a> needs access to a URL to facilitate the drilldown	<a href="#">NisRelease1Pt1</a>	8	■	No	■	■
<a href="#">026pt4</a>	<a href="#">026</a>	<a href="#">DrillDownUser</a> needs to be validated against the mainframe user database	<a href="#">NisRelease1Pt1</a>	8	■	No	■	■
<a href="#">026pt5</a>	<a href="#">026</a>	<a href="#">EndUser</a> must be able to drilldown from TMON exceptions	<a href="#">NisRelease1Pt1</a>	5	■	No	■	■
<a href="#">026pt6?</a>	<a href="#">026</a>	<a href="#">OdbcUser</a> must be able to drilldown from Time Series Data (i.e. APPLID)	<a href="#">NisRelease1Pt1</a>	13	■	No	■	■
	<b>027</b>	<b>SourceAdmin should have the ability, within the context of this product, to report on sources [aggregate instances] available to the product.</b>						
<a href="#">027pt1</a>	<a href="#">027</a>	<a href="#">SourceAdmin</a> can view an up-to-date list of sources	<a href="#">NisRelease1Pt1</a>	13	■	Yes	■	■
<a href="#">027pt9</a>	<a href="#">027</a>	<a href="#">EndUser</a> will observe that the NIS Server will update the Source state at regular intervals so that even if no Sources are received state remains current.	<a href="#">NisRelease1Pt1</a>	5	■	Yes	■	■
<a href="#">027pt10</a>	<a href="#">027</a>	<a href="#">SourceAdmin</a> will expect Source state to update dynamically in the user interface	<a href="#">NisRelease1Pt1</a>	8	■	Yes	■	■
	<b>028</b>	<b>SourceAdmin can configure which sources are collectable.</b>						
<a href="#">028pt1</a>	<a href="#">028</a>	The <a href="#">SourceAdmin</a> can mark sources (aggregate instances) collectible (i.e. making query target more specific).	<a href="#">NisRelease1Pt1</a>	13	■	Yes	■	■
<a href="#">028pt13</a>	<a href="#">028</a>	<a href="#">ProductSupport</a> will have access to an initial set of sensible default collection rules for each aggregate	<a href="#">NisRelease1Pt1</a>	13	■	No	■	■
<a href="#">028pt14</a>	<a href="#">028</a>	<a href="#">SourceAdmin</a> will have access to pre-configured default collection rules	<a href="#">NisRelease1Pt1</a>	13	■	Yes	■	■
<a href="#">028pt15</a>	<a href="#">028</a>	<a href="#">SourceAdmin</a> needs to be able to view a list of aggregates for a given tmon product instance	<a href="#">NisRelease1Pt1</a>	8	■	Yes	■	■
<a href="#">028pt16</a>	<a href="#">028</a>	<a href="#">SourceAdmin</a> can change the collectability attribute of an aggregate (which has default collectable element list)	<a href="#">NisRelease1Pt1</a>	8	■	Yes	■	■
<a href="#">028pt17</a>	<a href="#">028</a>	<a href="#">ProductSupport</a> can view the results of a TMON LFS <a href="#">TimeSeries</a> query for a specified TMON Product LFS Time Series Aggregate in the log file	<a href="#">NisRelease1Pt1</a>	8	■	No	■	■
<a href="#">028pt18</a>	<a href="#">028</a>	<a href="#">OdbcUser</a> can view the contents of a TMON LFS <a href="#">TimeSeries</a> aggregate in the database for a specified TMON Product LFS Time Series Aggregate	<a href="#">NisRelease1Pt1</a>	13	■	Yes	■	■
<a href="#">028pt19</a>	<a href="#">028</a>	<a href="#">TmonDev?</a> can validate <a href="#">TimeSeries</a> representation in the NMS Schema	<a href="#">NisRelease1Pt1</a>	13	■	No	■	■
<a href="#">028pt20</a>	<a href="#">028</a>	<a href="#">OdbcUser</a> can view the contents of any TMON LFS <a href="#">TimeSeries</a> aggregate that is found in the NMS Schema validated in <a href="#">NMSUserStory28pt19</a>	<a href="#">NisRelease1Pt1</a>	13	■	No	■	■
<a href="#">028pt21</a>	<a href="#">028</a>	<a href="#">OdbcUser</a> can expect a consistent mapping between TMON LFS <a href="#">TimeSeries</a> Element Types and NMS Database Types	<a href="#">NisRelease1Pt1</a>	13	■	No	■	■

# When will we be done?





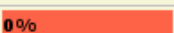
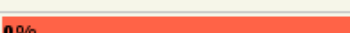
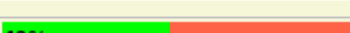

## The Release Plan is the Roadmap

- Estimated completion date is a Calculated Date Range
- Estimates change as stories are completed, time progresses and the rate of completion changes

### NMS Milestones Roadmap

UPDATED ON Mon 20-Aug-2007: Based on 27.3 points per iteration, 3 weeks per iteration

#### Milestone Summary

MileStone	Stories	Points To Do	Total Size	Estimated Completion Date	Progress
<a href="#">UmaPmoDemo</a>	9	N/A	26	COMPLETED	
<a href="#">DexiaBankDemo</a>	13	N/A	103	COMPLETED	
<a href="#">NmsUmaPmoMARelease</a>	40	N/A	219	COMPLETED	
<a href="#">NisRelease1Pt1</a>	56	330	498	Mon 14-Apr-2008 -> Mon 25-Aug-2008	
<a href="#">NisRelease1Pt1ValueAdded</a>	15	100	100	Mon 4-Aug-2008 -> Mon 15-Sep-2008	
<a href="#">NisRelease1Pt1Bling</a>	9	200	200	Mon 22-Dec-2008 -> Mon 23-Mar-2009	
<a href="#">NearTerm</a>	27	103	200	Mon 23-Mar-2009 -> Mon 11-May-2009	
<a href="#">LongTerm</a>	44	343	343	Mon 23-Nov-2009 -> Mon 5-Apr-2010	

# Story Points per Iteration is based on a Moving Average

- It is possible to use more than one average to offer both long and short terms estimates of progress

Iteration	Points	Avg	Stories Completed	Retro-spective	Development Phase	
					Start	End
<a href="#">6</a>	26	26.00	<a href="#">NMSUserStory4pt1</a> <a href="#">NMSUserStory27pt6</a> <a href="#">NMSUserStory65pt8</a>	<a href="#">6</a>	2006-05-17	2006-05-31
<a href="#">7</a>	32	29.00	<a href="#">NMSUserStory4pt2</a> <a href="#">NMSUserStory65pt11</a> <a href="#">NMSUserStory65pt13</a> <a href="#">NMSUserStory65pt14</a>	<a href="#">7</a>	2006-06-07	2006-06-21
<a href="#">8</a>	24	27.33	<a href="#">NMSUserStory10pt9</a> <a href="#">NMSUserStory56pt1</a> <a href="#">NMSUserStory65pt17</a>	<a href="#">8</a>	2006-06-29	2006-07-13
<a href="#">9</a>	50	33.00	<a href="#">NMSUserStory56pt1</a> <a href="#">NMSUserStory10pt2</a> <a href="#">NMSUserStory10pt10</a> <a href="#">NMSUserStory10pt8</a> <a href="#">NMSUserStory56pt2</a>	<a href="#">9</a>	2006-07-24	2006-08-11
<a href="#">10</a>	24	31.20	<a href="#">NMSUserStory4pt11</a> <a href="#">NMSUserStory10pt11</a> <a href="#">NMSUserStory53pt2?</a> <a href="#">NMSUserStory71pt1</a>	<a href="#">10</a>	2006-09-21	2006-10-04
<a href="#">11</a>	29	30.83	<a href="#">NMSUserStory10pt3</a> <a href="#">NMSUserStory10pt12?</a> <a href="#">NMSUserStory4pt12</a> <a href="#">NMSUserStory8pt5</a> <a href="#">NMSUserStory23pt1</a>	<a href="#">11</a>	2006-10-19	2006-11-02
<a href="#">12</a>	0	26.42		<a href="#">12</a>	2006-11-10	2006-11-27
<a href="#">13</a>	13	24.75	<a href="#">NMSUserStory4pt3</a> <a href="#">NMSUserStory4pt2?</a>	<a href="#">13</a>	2006-12-11	2006-12-22
<a href="#">14</a>	11	23.22	<a href="#">NMSUserStory6pt4</a> <a href="#">NMSUserStory4pt14?</a>	<a href="#">14</a>	2007-01-09	2007-01-22
<a href="#">15</a>	39	24.8	<a href="#">NMSUserStory4pt28</a> <a href="#">NMSUserStory4pt29</a> <a href="#">NMSUserStory28pt17</a> <a href="#">NMSUserStory28pt19</a> <a href="#">NMSUserStory71pt2</a> <a href="#">NMSUserStory71pt3?</a>	<a href="#">15</a>	2007-02-06	2007-02-19
<a href="#">16</a>	39	26.1	<a href="#">NMSUserStory4pt7</a> <a href="#">NMSUserStory27pt1</a> <a href="#">NMSUserStory51pt2</a> <a href="#">NMSUserStory68pt1</a>	<a href="#">16</a>	2007-03-03	2007-03-16
<a href="#">17a</a>	34	26.75	<a href="#">NMSUserStory4pt36</a> <a href="#">NMSUserStory56pt4?</a> <a href="#">NMSUserStory73pt2</a> <a href="#">NMSUserStory73pt3?</a> <a href="#">NMSUserStory73pt4?</a>	<a href="#">17a?</a>	2007-03-23	2007-04-05
<a href="#">17b</a>	34	27.31	<a href="#">NMSUserStory4pt19</a> <a href="#">NMSUserStory22pt2?</a> <a href="#">NMSUserStory50pt3</a> <a href="#">NMSUserStory50pt8?</a> <a href="#">NMSUserStory50pt9</a> <a href="#">NMSUserStory51pt5</a> <a href="#">NMSUserStory73pt11?</a>	<a href="#">17b?</a>	2007-04-06	2007-04-19
<a href="#">18?</a>			<a href="#">NMSUserStory1nt1?</a> <a href="#">NMSUserStory1nt4?</a> <a href="#">NMSUserStory73nt6?</a>	<a href="#">18?</a>	2007-04-24	2007-05-07



# Overview

- Globally Dispersed Teams and their Challenges
- Communication
- Suitability of Agile Development
- Introducing an Agile Methodology
- Trust
- Iteration Planning and Evaluation
- Project Visibility and Release Planning
- **Distributed Meetings**
- Coping with Fatigue
- Essential Development Tools
- Agile-Trac
- Questions

# Synchronisation: Keep Meetings Useful

- Should be no longer than necessary
- Should remain true to the expected duration
  - Do not overrun meetings, though some team meetings may be an exception
- Should be reserved for
  - Interactive discussion
  - Dissemination of important information requiring feedback
- Interactive meetings should be structured to enhance interactivity and maximise participation
  - Slides are not very interactive. Workspace sharing is better.
- Dissemination meetings must articulate the message clearly and provide a reference point for the information
- All meetings longer than a stand-up should have a focal point

# Distributed Meetings

- Meetings with dispersed teams generally take longer than those with co-located teams to produce the same value
  - For planning and evaluation dispersed meetings are often required to be longer
- Dispersed meetings also tend to be more demanding on participants
  - The assumption being that participants actually do participate
- Due to time differences the available meeting windows are often small
- It is often difficult getting all stakeholders (likely also dispersed) together for a suitable time
  - Important to plan planning and evaluation meetings in advance
- Calender discipline is very important



# Stakeholder Meetings

- Ensure participants understand the agenda and are equipped to participate
  - Encourage preparation work to minimise time wasted in meetings
  - Try to reserve meeting time as much as possible for interactive discussion
- Teams usually need reflection time between stakeholder meetings and these eat into the daily common time window
  - Sometimes two days are required rather than one day to coalesce on a decision
  - Factor this time upfront into the iteration times, particularly planning and evaluation
- Always try to provide a visual focal point for the meeting
  - Web meeting tools are ideal for this

# Maintaining a Shared Team View

- A shared team view is critical to allow the team to maintain an identity both within the team and the organisation
- When meeting stakeholders for important meetings meeting first to ensure a consistent voice
- During meetings make use of tools like Instant Messaging to maintain in sync with each other
- Be careful with 'offline' tools like Instant Messaging
  - Use for synchronisation, not whispering
  - There is no point banning use of such tools
    - This would be a breakdown of trust
    - Impractical to enforce
  - So encourage use of tools and remain honest about how you want to use them for the teams benefit

# Encouraging Participation

## Share the Driving

- Like all journeys, if you share the driving you reach your destination a lot fresher and with a lot less pain
- In team meetings lack of participation can be a problem
  - Without visual cues to encourage engagement in the discussion some members simply 'go with the flow'
- Share control of the meeting among team members
  - Use a web meeting tool, like Webex as the meeting focal point
  - Rotate the 'driver' between meetings
  - Rotate the 'driver' within longer meetings
  - Allows each team member to adopt a position of responsibility and trust with only minimal risk attached
    - This builds confidence
  - Improves visibility with other team members



# Encouraging Participation Co-Pilots

- Team members will always display different levels of willingness to communicate
  - Some team members may prefer to be left alone to get on with own work
  - Others may be somewhat overbearing
- Pair programming can help engage members but with time-zone differences and distance this is often impractical
- User story 'co-pilots' can be used to encourage participation
  - At least one other team member is assigned to help with a user story
  - Acts as a peer-reviewer or sounding-board
  - Encourages important one-on-one time between team members
  - Strengthens relationships and builds trust and respect

# Encouraging Participation Code Presentations

- Code-presentations can then be shared with co-pilots to help build tacit knowledge within the team
  - Emphasis on sharing understanding of the code and building team spirit
  - Not judgemental
  - Like retrospectives we follow Kerth's Prime Directive
  - Basically we provide an opportunity for presenters to highlight areas that can be improved or explain difficulties encountered
- An opportunity to enjoy a good technical discussion that may or may not lead to changes
- Atmosphere is supportive
- Greater respect and trust
- Better understanding of the code and increased tacit knowledge

# Daily Stand-Ups

- Daily Stand-Ups are an essential heartbeat of a dispersed team
- Often end up as Sit-Downs on the phone
  - It's just more practical, but you could mandate standing if possible
- The usual topics are:
  - What I've done since last stand-up
  - What I'll do before the next stand-up
- Time synchronisation issues will mean an imprecise start as people join the call
- Aim then for a definite finish time
  - Example consider ending on the hour rather than starting on it
- However distributed stand-ups are often longer
- Often stand-ups bleed into other topics



# Daily Knowledge Transfer

- Stand-up time is typically a mutually convenient time for all team members
  - If there are topics to be discussed then this is a good time to do it
- Important to finish stand-up first
- Often the whole team will stay on for follow-up discussions
  - Good for building tacit knowledge
  - Bad in that sometimes it can waste team member's time
  - Some team members may feel obliged to remain on the call
  - Balancing act that will require honest discussion
- Clearly identify the end of stand-up and provide an opportunity for team members to leave (or not re-join) when they will no longer benefit from participating

# Cats and Dogs

- Find time to talk about 'Cats and Dogs'
- “Anyone got any cats and dogs stories they'd like to share?”
  - There is no obligation to talk if you don't want to
  - The question is simply there to open the door for non-work topics
  - This is a chance to learn more about each other's personal lives and culture
- The goal is to humanise the relationship between team members.
  - Gain an appreciation of each other's culture
  - Learn a little about each other's humour
  - Being too politically correct may not help here
  - Learn to appreciate and value differences

# Overview

- Globally Dispersed Teams and their Challenges
- Communication
- Suitability of Agile Development
- Introducing an Agile Methodology
- Trust
- Iteration Planning and Evaluation
- Project Visibility and Release Planning
- Distributed Meetings
- **Coping with Fatigue**
- Essential Development Tools
- Agile-Trac
- Questions



# Coping with Fatigue

- Working in a dispersed team can be very draining
  - On the phone every work day
  - Always trying to make sure your message is understood
- Find opportunities to allow team members to relax a little
  - Restrictions of the common time window during planning and evaluation mean there may be some 'free' time
    - Use this to explore ways to improve your tools
    - Catch up on other development interests that may help
  - Depending on team size consider giving one team member a 'personal' iteration once in a while
    - Work must be project related
    - May chose to learn more about a different area of the project
- One of the best refreshments is to meet up but avoid times of high stress

# Overview

- Globally Dispersed Teams and their Challenges
- Communication
- Suitability of Agile Development
- Introducing an Agile Methodology
- Trust
- Iteration Planning and Evaluation
- Project Visibility and Release Planning
- Distributed Meetings
- Coping with Fatigue
- **Essential Development Tools**
- Agile-Trac
- Questions

# Essential Development Tools for Productive Distributed Development

- Source Control Management (centralised or distributed)
- Automatic Tests and Builds (continuous, daily, other)
- Test and Build Results Reporting and Notification
  - Particularly important with time differences
- All source and build related tools should have a web interface
- There are many free open source options so there will likely be something to suit your needs
- Many of these tools will require customisation and maintenance effort
  - The team is the only group who fully appreciate what is needed
  - Do not ignore this
  - Size the effort and factor it into your iteration plans



# Typical Tool Use

## From User Stories to Milestones

- User Stories
  - Basically a Special kind of Issue (Ticket System)
  - Must be editable with rich content (Wiki)
  - Tickets can be raised against User Stories
- Use Cases
  - Template based for ease of completion and Editable (Wiki, Ticket System)
- Milestones (the Release Plan)
  - Should be automatically updated as Stories become Done (Wiki, Trac, Script Driven Web-page or similar)
  - Must be visible to Stakeholders (Web-page)
  - Must be clear – this is the main focus for understanding progress

# Typical Tool Use

## Planning and Evaluation

- Release Planning
  - May need to capture dependencies in a graphical form to ease understanding (Graphviz, Freemind)
- Iteration Planning
  - Must have a scratch pad for intent and to capture commitment (Wiki)
  - Links to User Stories (Wiki)
- Iteration Evaluation
  - Should communicate a summary of what was achieved (Wiki)
  - Should have place to capture the outcome of the Retrospective (Wiki)

# Overview

- Globally Dispersed Teams and their Challenges
- Communication
- Suitability of Agile Development
- Introducing an Agile Methodology
- Trust
- Iteration Planning and Evaluation
- Project Visibility and Release Planning
- Distributed Meetings
- Coping with Fatigue
- Essential Development Tools
- **Agile-Trac**
- Questions



# Agile-Trac

- Trac (<http://trac.edgewall.org/>) is
  - “an enhanced wiki and issue tracking system for software development projects. Trac uses a minimalistic approach to web-based software project management. Our mission is to help developers write great software while staying out of the way. Trac should impose as little as possible on a team's established development process and policies.”
- However it has some limitations when used for agile projects
  - Tickets aren't sized and can be clumsy when used as user stories
  - No concept of iterations
  - Milestones dates are pre-determined, not calculated
  - Milestones cannot be prioritised
- Agile-Trac (<http://agiletrac.devjavu.com/>) is a plugin and patch for Trac that aims to address these and other issues

A photograph of a park scene. In the foreground, a large, dark tree trunk is partially visible on the left. The ground is covered in green grass, with a bright, yellowish-green patch of light on the grass near the base of the tree. In the background, there are more trees and a fence, all slightly out of focus.

Questions?  
Comments?  
Concerns?