

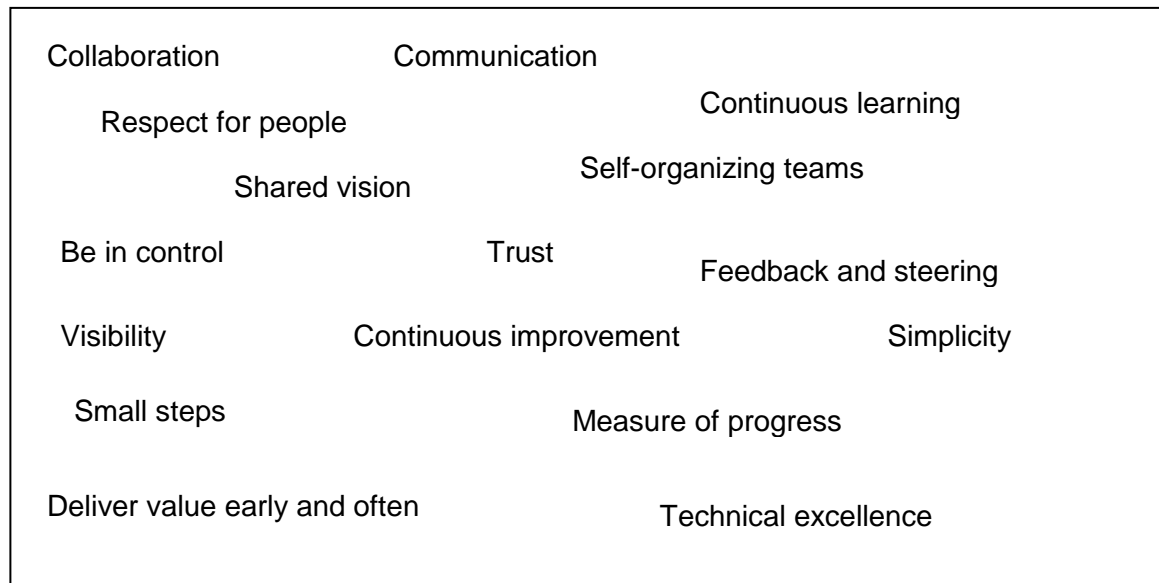


## XP loops: A normal day in an Agile team

<p><b>Ask yourself:</b></p> <ul style="list-style-type: none"> <li>• What can my team improve?</li> <li>• What can we learn?</li> <li>• What can we use?</li> <li>• How can my team change?</li> <li>• What would the result be?</li> </ul>	<p><b>On-site customer</b></p> <ul style="list-style-type: none"> <li>• “Customer” = interface for developers</li> <li>• Explain requirements</li> <li>• Plan, follow up, steer</li> <li>• Make business decisions</li> <li>• Evaluate output</li> <li>• 100% available</li> </ul> <p style="text-align: right;"><i>= Product Owner</i></p>
<p><b>Stories</b></p> <ul style="list-style-type: none"> <li>• Functional requirements on index card</li> <li>• Index cards <ul style="list-style-type: none"> <li>◊ Are cheap</li> <li>◊ Are easy to manipulate</li> <li>◊ Encourage brevity</li> </ul> </li> <li>• Story != specification <ul style="list-style-type: none"> <li>◊ Acceptance test == specification</li> </ul> </li> </ul>	<p><b>Small releases</b></p> <ul style="list-style-type: none"> <li>• 2 weeks – 1 month, <i>fixed</i> time</li> <li>• Real feedback to steer planning</li> <li>• Real progress indicator</li> <li>• Team learns to release</li> <li>• There’s always a working system</li> <li>• Time to reflect and improve</li> </ul> <p style="text-align: right;"><i>= Sprint</i></p>
<p><b>Planning Game &amp; Velocity</b></p> <ul style="list-style-type: none"> <li>• Developers estimate story effort</li> <li>• Customer chooses &amp; prioritizes <ul style="list-style-type: none"> <li>◊ Based on business value</li> </ul> </li> <li>• Effort / Release = Team “Velocity” <ul style="list-style-type: none"> <li>◊ Velocity is measured</li> </ul> </li> </ul>	<p><b>Sustainable Pace</b></p> <ul style="list-style-type: none"> <li>• No overtime, stress, pressure =&gt; <ul style="list-style-type: none"> <li>◊ Better quality</li> <li>◊ More productive</li> <li>◊ More learning, improving</li> <li>◊ More fun</li> </ul> </li> </ul>
<p><b>Daily standup</b></p> <ul style="list-style-type: none"> <li>• Stand up to keep it short</li> <li>• Everybody <ul style="list-style-type: none"> <li>◊ Agrees what they will work on</li> <li>◊ Raises problems &amp; difficulties</li> <li>◊ Knows what’s going on</li> </ul> </li> <li>• Initial pairing</li> </ul> <p style="text-align: right;"><i>= Daily Scrum</i></p>	<p><b>Pair Programming</b></p> <ul style="list-style-type: none"> <li>• 2 people working together +</li> <li>• Switch pairs regularly</li> <li>• Result = <ul style="list-style-type: none"> <li>◊ Better quality, continuous review</li> <li>◊ Problem solving, don’t get stuck</li> <li>◊ Concentrated on task (tiring !)</li> <li>◊ Knowledge diffusion</li> </ul> </li> <li>• Some people just can’t/won’t do it...</li> </ul>
<p><b>Unit Tests</b></p> <ul style="list-style-type: none"> <li>• Each unit has a set of automated tests</li> <li>• Succeeds / fails</li> <li>• Used as <ul style="list-style-type: none"> <li>◊ Regression test</li> <li>◊ Documentation</li> <li>◊ Examples</li> </ul> </li> <li>• Good unit tests are hard to write</li> </ul>	<p><b>Test Driven Design</b></p> <ul style="list-style-type: none"> <li>• Test first Design = <ul style="list-style-type: none"> <li>◊ Write a bit of the test. It Fails.</li> <li>◊ Write a bit of implementation. It works.</li> <li>◊ And so on... Going forward in tiny steps</li> </ul> </li> <li>• Simple design <ul style="list-style-type: none"> <li>◊ Easy to understand, change, get right</li> <li>◊ Solve today’s problems</li> </ul> </li> </ul>

<p><b>Refactoring</b></p> <ul style="list-style-type: none"> <li>• How do you keep code simple &amp; clean? <ul style="list-style-type: none"> <li>◊ How do you keep your house clean?</li> </ul> </li> <li>• Change structure, not the behavior <ul style="list-style-type: none"> <li>◊ Tests will verify that it still works</li> </ul> </li> <li>• Do it all the time <ul style="list-style-type: none"> <li>◊ Small, simple steps</li> </ul> </li> </ul>	<p><b>Continuous Integration</b></p> <ul style="list-style-type: none"> <li>• When the code works and is clean...</li> <li>• ... integrate with the rest of the code</li> <li>• ... several times a day</li> <li>• Fast feedback, test the whole system</li> <li>• Give up code to collective ownership</li> </ul>
<p><b>Collective Code Ownership</b></p> <ul style="list-style-type: none"> <li>• The TEAM makes the product</li> <li>• Increase the “Truck Number”</li> <li>• It works...</li> <li>• ... in disciplined XP teams</li> </ul>	<p><b>Acceptance Tests</b></p> <ul style="list-style-type: none"> <li>• Automated tests serve as...</li> <li>• ... Specification</li> <li>• ... Acceptance criterion</li> <li>• ... Regression test</li> <li>• ... Progress indicator</li> <li>• ... Bug tracking tool</li> </ul>
<p><b>Open Workspace</b></p> <ul style="list-style-type: none"> <li>• ONE team, including customer</li> <li>• Lots of communication</li> <li>• With how many people do you want to work in one room?</li> </ul>	<p><b>Metaphor</b></p> <ul style="list-style-type: none"> <li>• Stories about the way it works +</li> <li>• Definition of terms</li> <li>• Shared Vision among developers</li> <li>• Common vocabulary, incl. Customer</li> <li>• Bring newcomers &amp; outsiders up to speed</li> </ul>
<p><b>Retrospective</b></p> <ul style="list-style-type: none"> <li>• At the end of iteration/release/project</li> <li>• What happened?</li> <li>• What can we do better next time?</li> <li>• It's not about the previous project...</li> <li>• It's about the next project</li> </ul>	<p><b>The end?</b></p> <ul style="list-style-type: none"> <li>• XP is simple but hard (to do it right)</li> <li>• XP is not the end, it's a beginning</li> <li>• Define your own process</li> <li>• Solve your worst problem...</li> <li>• ... and then the next worst problem</li> <li>• ... and keep improving</li> </ul>
<p><b>Agile Project Management</b></p> <ul style="list-style-type: none"> <li>• Project Vision</li> <li>• Self Organizing teams</li> <li>• 4 Phases of a team (forming, storming, norming, performing)</li> <li>• Ask for help</li> <li>• Product Backlog</li> <li>• Information Radiators</li> <li>• Burn down charts</li> </ul>	<p><b>Scaling Agile</b></p> <ul style="list-style-type: none"> <li>• Scaling agile: Enterprise Agile &amp; Distributed agility</li> <li>• Cultural differences</li> <li>• Scrum of Scrums</li> <li>• Local visits</li> <li>• Tools for better communication between distributed members</li> <li>• Reading groups</li> </ul>

## Values and Principles



### See also:

<http://www.xp.be> Belgian XP/Agile User Group  
<http://www.xpday.net> XP Days Benelux conference  
<http://www.agilealliance.org> Agile Alliance

“Extreme Programming Explained: Embrace Change” by Kent Beck  
“Refactoring: Improving the Design of Existing Code“ by Martin Fowler  
“Test Driven Development: by Example” by Kent Beck  
“The Pragmatic Programmer” by Andrew Hunt and David Thomas  
“Fearless Change” by Linda Rising and Mary Lynn Manns  
“Project Retrospectives“ by Norman L. Kerth  
"Implementing Lean Software Development" by Mary & Tom Poppendieck  
“Agile Management” by David Anderson  
“Agile Software Development“ by Alistair Cockburn  
“Agile Software Development with SCRUM” by Ken Schwaber & Mike Beedle  
“DSDM: Business Focussed Development“ by DSDM Consortium  
“Agile Software Development in the Large: Diving Into the Deep” by Jutta Eckstein  
“Better Software Faster“ by Andy Carmichael

More books on XP, Lean, Theory of Constraints, Systems Thinking and Agile at

- <http://wiki.systemsthinking.net/Systemsthinking/BookList.html>
- <http://www.librarything.com/catalog.php?view=YvesHanouille>

Pascal Van Cauwenberghe  
[www.nayima.be](http://www.nayima.be)

Vera Peeters  
[www.tryx.com](http://www.tryx.com)