What the Agile Toolbox Contains

Dr. Alistair Cockburn
Humans and Technology

The agile development community is noted for scorning Computer-Aided Software Engineering modeling and Gantt project scheduling tools (among others), but what has it replaced them with? Conducting a survey of agile teams for tools they say help produce better software quicker, this author found they used a cross-disciplinary set of mental, social, environmental, mechanical, and process tools, in addition to a carefully selected set of software-based tools. This list of tools can help your organization prepare for the tools – human resource, facilities, software, and non-software – that will be requested and used by the team starting to adopt the agile approach.

The word *tool* usually brings to mind a physical or software device. However, agile software development teams have removed much of the usually mentioned hi-tech development tools from their repertoire. Thus, in conducting a survey of tools agile teams say produce better software sooner, I had to be more general in considering what might be regarded as a *tool* when asking, "What does the agile toolbox contain?"

The set of tools that agile development teams consider part of their toolbox is very broad, ranging in purpose across *hiring*, *collaborating*, *communicating*, *managing*, *developing*, etc. Their tools also range in form across *environmental* (such as office layout), *social*, *physical*, *process*, *thinking*, and *computer-based*.

For the survey, I seeded a discussion about tools and posted requests for input to four agile development discussion groups [1, 2, 3, 4]. Originally, I intended to describe a few of the more unusual items on the resulting list. However, the toolset that arrived back was so interesting when considered as a whole that I chose to show it in its entirety.

When people are deciding whether to use an agile development approach on an upcoming project, they can work through this list together, considering the implications of each item on their budget and work habits. Then it will not be such a surprise when the team starts to rearrange the cubicles and request different furniture, post bits of paper all over the wall, or ask to have job applicants co-program with them for a morning.

This article is arranged in the following sections:

- A brief description of agile development with references for further reading and a short description of terms that will show up in the tool lists.
- The tools grouped by the purpose they support.
- The tools itemized by form.
- Reflection on this list as a whole.

Agile Development Acronyms and Key Words

Generally speaking, teams using the agile development approach focus strongly on collaboration and rapid feedback from running code.

> *"Although physical proximity, whiteboards, poster sheets, index cards, and sticky notes are still the dominant tools used in collaboration, people started finding and inventing online collaboration tools as agile development moved into distributed development."*

Collaboration is expected not only within the development team but also across organizational boundaries, with expert users and project sponsors. Collaboration involves group workshop techniques for project planning, requirements gathering and design, and programming in pairs or in close proximity such as in a war-room setting.

Collaboration also involves using *information radiators* [5] – large displays showing up-to-date information placed in public for people to see whenever they pass by. Information radiators are used in workshops, in the war-room setting, and between continents to keep people in sync on their goal and their state.

Rapid feedback is based on running, tested, integrated system features, or RTF [6]. The project plan is constructed, and progress is measured in terms of the steadily increasing set of integrated features. The team seeks early and frequent integration of features to get feedback about the team, the process it is using, and how the requirements fit the actual needs of the user base.

Attending to collaboration and feedback through RTF drives the selection of many of the tools listed in this article. The agile team cares that the following occurs:

- The right *roles* are established for the team.
- The people who show up *fit* with the rest of the team.
- The people develop particular *skills*.
- The *environment* is effective to the development task.
- They use selected *process* elements.
- *Collaboration* and *communication* are facilitated.
- The *mechanical*, *hardware*, and *software* tools used are easy to use, see, and update; are effective; and support the agile approach.

Teams debate items in all of these categories and will feel endangered or strengthened when the various items are removed or included. It is on this basis that I consider such a broad range of items *tools*.

There is not space here to undertake a longer description of agile development; it has been heavily described in books and articles. Perhaps the best introduction to the thinking and practices involved is found in "Agile Software Development

Ecosystems" [7] and the articles "The New Methodology" [8], "The Business of Innovation" [9], and "The People Factor" [10]. The AgileAlliance [11] and the Agile Project Management Group [12] offer much more information.

Listed below are some terms that may not be familiar to the reader:

- **Dynamic System Development Method (DSDM):** A founding agile methodology created in the United Kingdom in the late 1990s [13].
- **Scrum:** A founding agile methodology created in the mid-1990s [14].
- **Scrum master:** In the Scrum methodology, a form of team leader who specializes in getting people to talk together and in removing obstacles to progress.
- **Gold cards:** A token allowing a developer to work on something other than scheduled features.
- **Class-Responsibility-Collaborator (CRC) cards:** An object-oriented design technique in which designers write class names on index cards and role-play the design with the cards [15].
- **Java 2 Enterprise Edition (J2EE):** A widely used component library marketed by Sun Microsystems.
- **Unified Modeling Language (UML):** A widely used graphical design documentation notation.

If there are other terms you find unfamiliar, a quick Web search is sure to turn up descriptions and discussions of them.

Tools by Purpose

Included here are entries only for hiring, collaboration, communication, and management purposes. The entries for other activities should be fairly obvious when reading the list grouped by form later in this article.

Hiring

To hire the appropriate people for the team, you must first identify the roles needed and the people to fit those roles. *Process* and *social* tools are used here.

To avoid the standard mistakes in hiring, the tool most often used is a few hours of pair programming with the team members. Teams report being able to tell a lot more about how an applicant thinks, designs, communicates, and fits with the team from this experience. Even without pair programming, interviewers focus on discovering not only an applicant's technical abilities, but also their personal fit with the organization.

Two new roles or skills are sought: facilitators and coaches. In the late 1990s, the founders of the DSDM felt so strongly that their project teams needed proper facilitation expertise that they helped develop an internationally recognized facilitator training and certification program [16]. An increasing number of software people are becoming certified public facilitators, and more are taking basic facilitator courses.

Coach (from eXtreme Programming [XP]) and *scrum master* (from Scrum) are job titles designed to change the power relationship and interaction dynamics from the traditional team lead or project manager. The coach or scrum master is a lead person whose job typically is to keep desired practices in place and remove obstacles for the group, but not to create schedules for the developers or construct their end-of-year performance reviews. Therefore, the group perceives them as a *leading colleague* rather than a *boss*.

Collaboration

Although physical proximity, whiteboards, poster sheets, index cards, and sticky notes are still the dominant tools used in collaboration, people started finding and inventing online collaboration tools as agile development moved into distributed development. These tools will be listed separately in the computer-based category. They generally include WikiWiki and thread-based discussion group technologies, instant messaging technologies with group and recording variants, and distributed brainstorming technologies.

"Whether collocated or distributed, the two prevalent process tools for collaboration include workshops and short daily status meetings."

Whether collocated or distributed, the two prevalent process tools for collaboration include workshops and short daily status meetings. Workshops are used to gather requirements, understand usage patterns, plan the project, and design the software. To support the workshops, specific office facilities are required, including group work areas with lots of wall space, speakerphones, and videoconferencing.

Communication

Active and passive communication remains a dominant trait of agile development, whether the team is collocated or distributed.

Active communication involves two or more people working on the same task, whether at a whiteboard, sitting side-by-side looking at the same screen, or using shared workspace technology to look at the same screen from different sites.

Passive communication involves information radiators. These are most often on paper or whiteboard. When the information changes on a minute-by-minute basis, information radiators are sometimes driven online. Information radiators include the following:

- A flat monitor hung over the cubicle wall [17].
- A real traffic light hung in the development area that is controlled by an automated build machine [18].
- An ambient orb reporting the same as the traffic light, but using a nationally broadcast signal so teams in all locations can see the same information [19, 20].
- The build status maintained on a Web page so the developers can see what happened to the code they just entered.

Management

Agile teams have replaced Gantt charts with earned value and burn-down charts [21], graphs of tests created versus passed, and similar charts. To report these to upper management, collocated teams still like the effects of posters taped to the wall or spreadsheet graphs. A fresh set of online project management tools is entering the market, including Rally, VersionOne, and XPlanner.

Whether online or on paper, these tools report status with respect to RTF, not planning, design, or documentation tasks.

Tools by Form

Here are the tools clustered by their form: environmental, social, physical, process, mental, and computer-based.

Environmental

You are likely to find that the agile team will either request a different office layout or will simply rearrange their given space to enhance collaboration. The following are common desires:

- Common design and programming areas.
- Lots of wall space for posting information radiators.
- Convex or straight desks so people can cluster around the monitor.
- A common couch area with a white-

board (recording type, preferably).
- Kitchen, for social discussions during breaks.

Social

The top social tools are collocating teams and attacking problems in workshop sessions. Other social tools revolve around increasing the tolerance or amicability of people toward each other, giving them a chance to alternate high-pressure work with decompression periods, and allowing them to feel good about their work and their contributions. The following are desired social tools:

- Social roles such as coach, facilitator, and scrum master.
- Collocated teams (for fast communication and also the ability to learn about each other).
- Personal interaction (within and across specialties).
- Facilitated workshop sessions.
- Daily stand-up status meetings.
- Retrospectives and reflection activities.
- Assisted learning provided by lunch-and-learn sessions, pair programming sessions, and having a coach on the project.
- Pair programming (to provide peer pressure).
- A shared kitchen.
- Toys (to allow humor and reduce stress).
- Celebrations of success and acknowledgment of defeat.
- Gold cards issued at an established rate (to allow programmers to investigate other technical topics for a day or two).
- Off-work get togethers (typically a Friday evening visit to a nearby pub, wine-and-cheese party, even volleyball, foosball, or Doom competitions).
- Posting information radiators in unusual places to attract attention (the most unique I have seen is the number of open defects being posted in the bathroom [22].)

Physical Devices

The best physical devices augment individual thinking, group thinking, and social interaction. The following are some of the preferred ones:

- Index cards and Post-it notes (in any gathering of agile developers, someone is likely to have a pack of index cards with them).
- Butcher paper lining walls and halls.
- Whiteboards (standard or moveable, printing, recording, or with a camera).
- Poster sheets (plain paper, 3M sticky, or plastic cling sheets).

Process

Preferred process tools include short, time-boxed iterations, frequent integration, and frequent delivery. Next to these are workshops for various purposes. Some of the tools are both process and social in form, so I risk listing them twice. They are as follows:

- Project planning jam session (XP's planning game [23], Crystal Clear's blitz planning [17], or Scrum's sprint planning).
- Requirements workshop.
- Group design workshop.
- Reflection or retrospective workshop.
- Pair programming session.
- Refactoring code.
- Growing the system (creating a very small but functional implementation, adding both infrastructure and functionality).
- Time boxing.
- Spike prototyping (throwaway prototyping lasting not more than a day or two).
- Early integration.
- Frequent delivery.
- Programmers writing unit tests.
- Customer writing acceptance tests.
- Tracking by earned value, burn-down, or backlog.

Thinking

Agile developers may or may not model the domain with UML, but they do have tools for helping them decide what and how to code, starting with using the brain. Thinking tools include the following:

- Brain-engaged common sense [24].
- Test-first design (assertion-driven design).
- CRC cards.
- KISS (keep it simple, stupid).
- Once-and-only-once code (do not repeat yourself in your code).

Computer-Based

Jeff Patton writes:

> Of course, agile developers have a long history of tool building – I think that started with chimpanzees using sticks to get bugs out of stumps. Today we use xDoclet to generate J2EE interfaces and classes, which is a lot like getting bugs out of stumps [25].

There are enough entries in this list that I need to group them by purpose. Obviously, I am not attempting a full listing of tool vendors, so I name only one or two sample entries of available online tools where that is relevant. I apologize to the other tool suppliers.

Computer-Based Tools By Purpose

Communication/Collaboration Tools

Here are communication or collaboration tools that require software:

- Group discussion technologies such as WikiWiki, Yahoo! eGroups, Lotus Notes, Starteam, NetMeeting, WebEx, phpBB, and blogs.
- Instant messaging, including group messaging, messaging with drawing, and messaging with discussion thread management. Examples include Yahoo! Messenger with Doodle Imvironment (so people can draw at each other as well as talk), Jabber, AIM, GAIM (group chat), Engage Thoughtware (thread management), and Trillian.
- Collaboration software packages such as Marratech, Raindance, Sparrow, Flywheel, Thoughtware, and Borland's Caliber.
- Video projectors for group coding, learning, and discussion sessions.

Documenting Tools

There is an overlap between collaboration tools and documentation tools. Increasingly, teams look for easy ways to put the results of a group workshop into archive format. Often that involves a camera, but sometimes it means using an online tool during collaboration, including the following:

- Recording whiteboards; scanners; and archiving message, discussion, and collaboration tools (the output is simply put or linked into the documentation).
- Generic drawing tools, PowerPoint, Visio, Dia, and ArgoUML (free) replace expensive computer-aided software engineering packages.

Project Tracking Tools

These are online alternatives to poster sheets posted on the wall, and are particularly useful for distributed teams and for projects whose requirements or plans change multiple times per week. They are as follows:

- Spreadsheets (used to hold project plan and status, and derive tracking graphs).
- Software for tracking the project against stories and tasks comes from XPlanner (free), Rally Software's Agile Release Management, Borland's CaliberRM, and VersionOne.

Designing-Programming Tools

Here are the essential tools requested by agile programming teams:

- Configuration management/version control (Concurrent Version System or your favorite).
- Automated unit test harness such as JUnit or any of the xUnit family.
- Automated acceptance test harness such as Fit or FitNesse.
- Automated build system, preferably a continuous build system such as augmented Another Neat Tool or CruiseControl.
- Refactoring development environment (safe refactoring built in) such as Intellij's IDEA, Eclipse, or ReSharper.
- Performance profiling tool such as Jmeter, Jprofiler, or Jprobe.
- Laptops on a wireless network for programming anywhere.

Other Resources

The Internet contains many discussions of social, process, physical, and computer-based tools for agile development. Ken Boucher has created the Web site <www.fairlygoodpractices.com> for collecting a number of social and process tool descriptions.

To discover your own set, simply hold a workshop with the people on your team and ask them what mental, social, environmental, and physical devices help them in their work. My experience is that they will be glad to share, and you will end up with an impressive list of your own.

Reflection on the Lists

I was surprised at the breadth of tools requested by agile teams, by how far back into the hiring cycle these tools extend, and by the number and importance of the social tools. I was surprised at how far the industry has come in supporting distributed teams with distance collaboration and automated build systems.

As I wrote in [5], understanding passes from person to person more rapidly when they are standing next to each other, as when they are discussing at a whiteboard. Agile teams stress using tools that permit the rapid flow of understanding. Some of those tools are social, starting even at the hiring stage. Some tools are technological, helping distributed teams simulate being physically present. Many tools are physical, allowing people to manipulate them in workshops.

If collaboration is one leg that agile development stands on, the other is rapid feedback from running code. Configuration management, automated testing, refactoring, and performance profiling tools are the dominant entries here. As Michael Vizdos reminds us, do not forget to keep brain and common sense engaged [24].◆

References

1. Yahoo Groups. eXtreme Programming <http://groups.yahoo.com/group/extremeprogramming/message/93430>.
2. Yahoo Groups. Scrum Users <http://groups.yahoo.com/group/scrumdevelopment/message/3652>.
3. Yahoo Finance Groups. Agile Project Management <http://finance.groups.yahoo.com/group/agileprojectmanagement/message/2424>.
4. Yahoo Groups. Salt Lake Agile Software Development <http://groups.yahoo.com/group/sl-agile/message/646>.
5. Cockburn, A. Agile Software Development. Addison-Wesley, 2002.
6. Jeffries, R. "A Metric Leading to Agility." XP Magazine 14 June 2004 <www.xprogramming.com/xpmag/jatRtsMetric.htm>.
7. Highsmith, J. Agile Software Development Ecosystems. Addison-Wesley, 2002.
8. Fowler, M. "The New Methodology." Apr. 2003 <www.martinfowler.com/articles/newMethodology.html>.
9. Highsmith, J., and A. Cockburn. "Agile Development 1: The Business of Innovation" <http://alistair.cockburn.us/crystal/articles/asdboi/asd1businessofinnovation.htm>.
10. Cockburn, A., and J. Highsmith. "Agile Development 2: The People Factor." <http://alistair.cockburn.us/crystal/articles/asdpf/asd2peoplefactor.htm>.
11. Agile Alliance <http://agilealliance. org>.
12. Agile Project Management <http://agileprojectmgt.com>.
13. DSDM Consortium <http://www.dsdm.org>.
14. Control Chaos.com. ADM, Inc. <http://controlchaos.com>.
15. Beck, K., and W.A. Cunningham. "A Laboratory for Teaching Object-Oriented Thinking." ACM SIGPLAN 24.10 (1989): 1-7.
16. Airth, Alan. GlobalFN. Personal communication to the author. 29 June 2004.
17. Cockburn, A. Crystal Clear. Addison-Wesley, 2004.
18. Freeman-Benson, B., and A. Borning. "YP and Urban Simulation: Applying an Agile Programming Methodology in a Politically Tempestuous Domain." Seattle, WA: University of Washington, 2003 <http://agiledevelopmentconference.com/2003/schedule/researchpapers.html#P1>.
19. Ambient Devices <www.ambientdevices.com>.
20. Savoia, Alberto. "eXtreme Feedback for Software Development." Agitar Software, Inc., 2003 <www.developertesting.com/managed_developer_testing/000036.html>.
21. Cockburn, A. "Earned-Value and Burn Charts." Humans and Technology, 22 June 2004. Extracted from Crystal Clear, Addison-Wesley, 2004 <http://alistair.cockburn.us/crystal/articles/evabc/earnedvalueandburncharts.htm>.
22. Developertesting.com <http://www.developertesting.com/images/entryimages/mdt-extreme-feedback_07_0001.jpg>.
23. Beck, K., and M. Fowler. Planning eXtreme Programming. Addison-Wesley, 2001.
24. Vizdos, Michael. Online Posting. Yahoo Groups. Scrum Users <http://groups.yahoo.com/group/scrumdevelopment/message/3661>.
25. Patton, Jeff. Online Posting. Yahoo Groups. Salt Lake Agile Software Development <http://groups/yahoo.com/group/sl-agile/message/648>.

About the Author

Alistair Cockburn, Ph.D., is an internationally respected expert on object-oriented design, software development methodologies, use cases, and project management. He is the author of two Jolt Productivity award winning books, "Agile Software Development" and "Writing Effective Use Cases," as well as author of "Surviving OO Projects." He was also one of the authors of the "Agile Development Manifesto." Cockburn defined an early agile methodology for the IBM Consulting Group in 1992, served as special advisor to the Central Bank of Norway in 1998, and has worked in companies from Scandinavia to South Africa, North America to China. Internationally, he is known for his seminal work on methodologies and use cases, as well as his lively presentations and interactive workshops. Many of his materials are available online at http://alistair.cockburn.us.

Humans and Technology
1814 Fort Douglas CIR
Salt Lake City, UT 84121
E-mail: acockburn@aol.com