Software Engineering Toolbox

What is a Toolbox?

Construx finds that the toolbox is very strong metaphor for software development. A software engineering toolbox provides a central knowledge base for acquiring, defining, and disseminating guidance about the software engineering processes and practices. Although particular software tools (e.g., developer environments, revision control systems, automated test suites, project management software) will play a role in the toolbox, **the term "tool" is used in a broader sense to encompass techniques, best practices, templates, checklists, examples, etc**. The techniques employed often have a greater impact than any particular piece of software.

The toolbox provides a mechanism for sharing knowledge about the organization's process assets and how they are used. It accelerates project work by reducing duplication, promoting common usage, and leveraging reuse. The toolbox allows the organization to add new tools to their existing way of working without changing everything or throwing away the tools that they already know.

Growing your Toolbox

The best toolbox is one that you grow over time. Just like the toolbox you have in the garage, you shouldn't buy a new tool until you need it. A high-level process for growing your toolbox would consist of:

- **Define a simple taxonomy**. This defines how the tools are organized, for example the directory structure in a common repository. The taxonomy allows team members to quickly find tools and easily discuss software engineering issues. A good toolbox taxonomy becomes the framework for your organization to plan, discuss, and execute software project. One trap to avoid is spending too much time trying to define taxonomy. It is more important to have a common, well-understood organization than to have one that is "perfect."
- **Bootstrap the toolbox.** Populate the toolbox with an initial set of tools. Bootstrapping consists of **harvesting** tools that the teams already have defined and identifying and **filling in critical gaps**. Harvesting leverages investments your company has already made and promotes buy-in. It also sends a clear message that management is not trying to impose a one-size-fits-all set of practices. You should then identify and fill critical gaps by creating or acquiring the necessary tools.
- **Continuous improvement**. The toolbox should be the first place they look for a tool. In many cases, the team will be able select the appropriate tool from the toolbox. However, the team will face situations where existing tools do not meet their needs. Teams should approach this by first seeing if a tool could be modified for the use. If not, they can create a new tool. In both cases, the team should put the tool back in the toolbox for future consideration.

Using the Toolbox

The toolbox approach does not mandate a particular lifecycle or methodology. It specifically recognizes that the one-size-fits-all approach is fundamentally flawed. Instead, it acknowledges that the team is ultimately responsible for identifying the tools that they will use on their projects.

The toolbox provides the project members the framework in which to select the right-sized tool and optimize it for their particular needs. The tools can be applied separately or in concert with one another. They can be mixed and matched with other tools to support different teams and objectives.

Construx's Toolbox

Most of the examples in this document are based on artifacts in Construx's CxOne toolbox. CxOne is an agile framework of templates, checklists, patterns, and other tools that provide relief from software development pain. CxOne provides practical software engineering support to project teams. This vision is realized through efficient execution of proven techniques.

CxOne is based on industry best practices, Steve McConnell's works, Construx's real-world project experience, and lessons learned while working with Construx's clients. CxOne does not prescribe a single approach to software development; it provides a foundation that allows projects and organizations to quickly improve how they develop software. CxOne does not replace your existing practices; it builds on them. CxOne's document-based approach provides maximum flexibility for teams to deploy new and improved practices.

For more information, see Construx's CxOne web page.