Implementing the Rock Crusher

contributed by Steve Adolph

Contents

> “The hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the detailed technical requirements. No other part of the work so cripples the resulting system if done wrong.”

No Silver Bullet: Essence and Accidents of Software Engineering. Computer
FP Brooks Jr - IEEE Computer Society, Washington, DC, 1987

Working Around Tooling Limitations

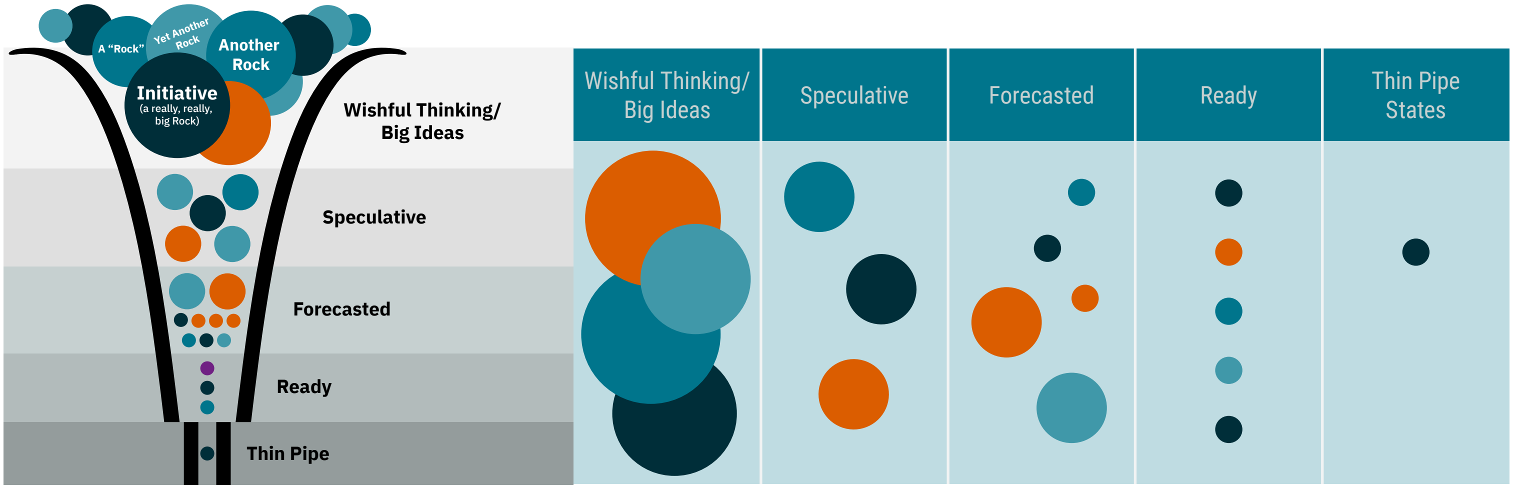
A major challenge for the Rock Crusher is that most tools use the forced-rank stacked-plates model to represent their backlog. Few tools, if any, let you visually cluster work to reflect the uncertainty associated with the ranking of that work. This approach can encourage a premature forced ranking of work items.

Most tools limit the generations of parent-child relationships between backlog items.

- For example, “out-of-the-box” Jira is limited to an Epic->User Story hierarchy, while a few others may support an Initiative->Feature->User story hierarchy. This limits the ability to visualize the ambiguity inherent in progressive elaboration as Rocks are crushed and refined into smaller, more concrete work items.

While most tools cannot directly implement a Rock Crusher model, there are many ways to approximate it from a simple “cutline” to a complex set of linked Kanban boards, for example:

- Simple cutline – reasonable for a small independent team;
- Simple Kanban board – applicable to a small team; and
- Linked Kanban boards – appropriate for medium to large teams.



Small Close-Knit Team: The Cutline

For an individual or small team, the simplest technique for approximating the Rock Crusher is a “cutline.” A cutline is simply a “dummy” work item in the backlog which demarcates Ready work from work that is “not ready.”

Ready

The intent is that backlog items below this line are forced-rank prioritized and Ready – they meet the INVEST criteria, while those backlog items above the cutline shouldn’t be assumed as either Ready or forced-rank prioritized. There is a high probability that items below the cutline will soon be pulled into an iteration. A good guideline is to have one to two iterations worth of items that are Ready. Anything above the cutline still requires work to get “Ready.”

Backlog items above the cutline may:

- Not be sized beyond t-shirt size;
- Not be sized at all;
- Be large, perhaps classical epics – too big to complete in an iteration and need to be split;
- Not yet have acceptance criteria;
- Still need additional learning to refine them; and
- Appear “forced ranked” in the tool, they really are in loose clusters of priority.

A team may like to clarify their ability to visualize the “readiness” of work in their backlog by adding a second cutline to differentiate forecasted from speculative.

Forecast

Backlog items between Forecasted and Ready are being actively refined, the team is working together to Crush these backlog items.

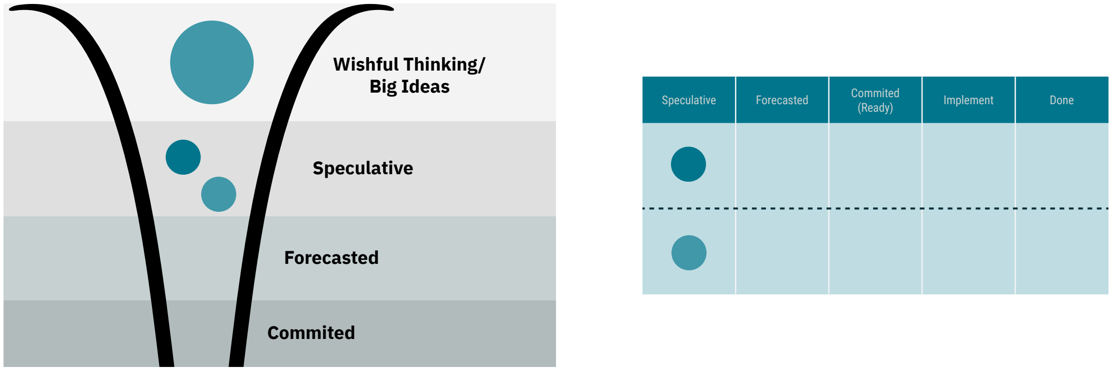
If there are more than two cutlines, look for a more sophisticated technique.

The advantage of this technique is its simplicity. For a small, tight-knit development team, this may be more than adequate. The disadvantage of this technique is its limited ability to visualize the parent -> child relationship between larger abstract Rocks and their progeny. For a small team this may not be a concern because the team can easily share knowledge tacitly, but in a larger or more complex situation this may not be enough.

Simple Kanban Board

For a small team with a need for greater visibility than offered by the cutline approach, a simple Kanban board offers a more sophisticated technique for implementing the Rock Crusher. The use of Kanban boards is the canonical solution for implementing the Rock Crusher.

Most tools, like Jira, allow teams to create and map additional workflow states to whatever states administrators have defined in the tool (usually most systems default to something like Backlog, In Progress, In-Review and Accepted). Creating a team-level Kanban board with additional states prior to Backlog enables a fair approximation of the Rock Crusher.

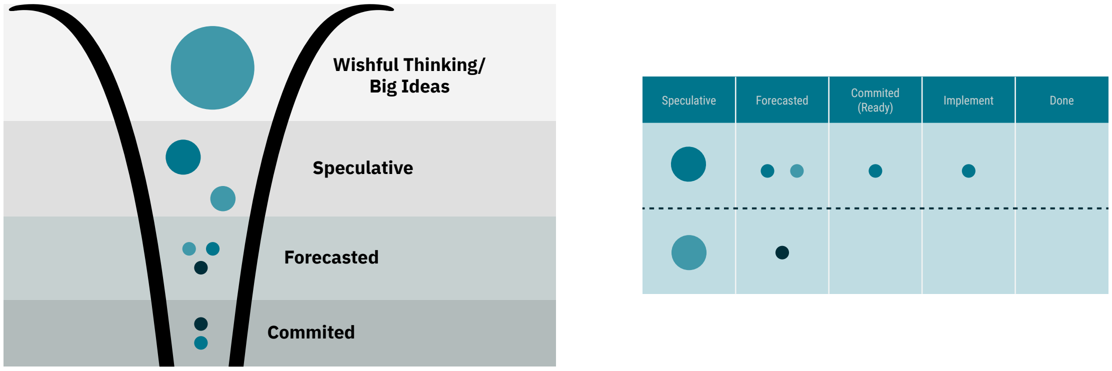


Simple Mapping Example Between the Rock Crusher and a Kanban Board

Like any good Kanban board, there should be work-in-progress (WIP) limits at each step. Real Kanban super stars will have explicit exit criteria for each state. Ideally, that exit criteria expresses how a backlog item passes some kind of test description, whether code or analysis.

Depending on how much visibility the team requires, swim lanes, or even strings, can be used to capture the rock crushing processes and the parent->child links between the “big Rocks” and the “small Rocks.”

Many tools have a limited ability to maintain parent->child relationships between Rocks. Ideally with the Rock Crusher, a more arbitrary parent->child relationship could be maintained when “big chunks” of work are crushed into smaller pieces of work.



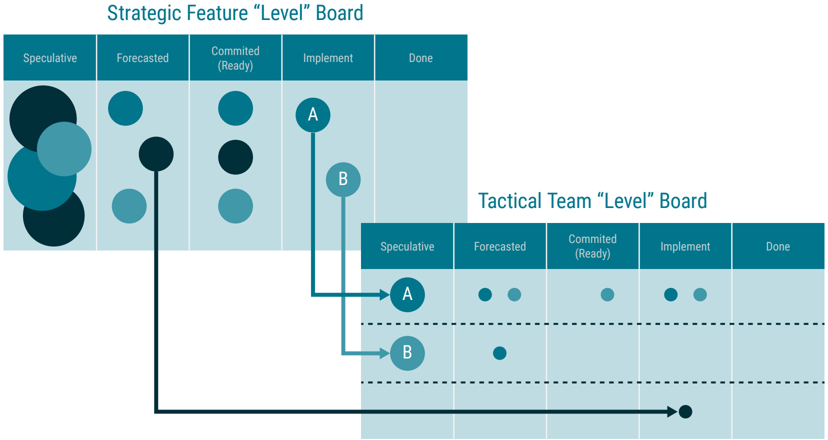
Kanban Board with Swim Lanes Reflecting the Flow of Work Through the Rock Crusher

Linked Hierarchy of Kanban Boards – Single Team

There are many situations which require greater visibility and reporting capabilities than are possible with just a simple Kanban board representation of a Rock Crusher. Big Rocks may represent features that are important to stakeholders, and those stakeholders are very interested in the lifecycle of those features, but they don't really care about how the team gets the job done. There may be multiple teams operating the Rock Crusher.

The roles of Solution Manager and Backlog Owner may be carried out by different individuals. Linking Kanban boards in a hierarchy provides a more useful representation of the Rock Crusher.

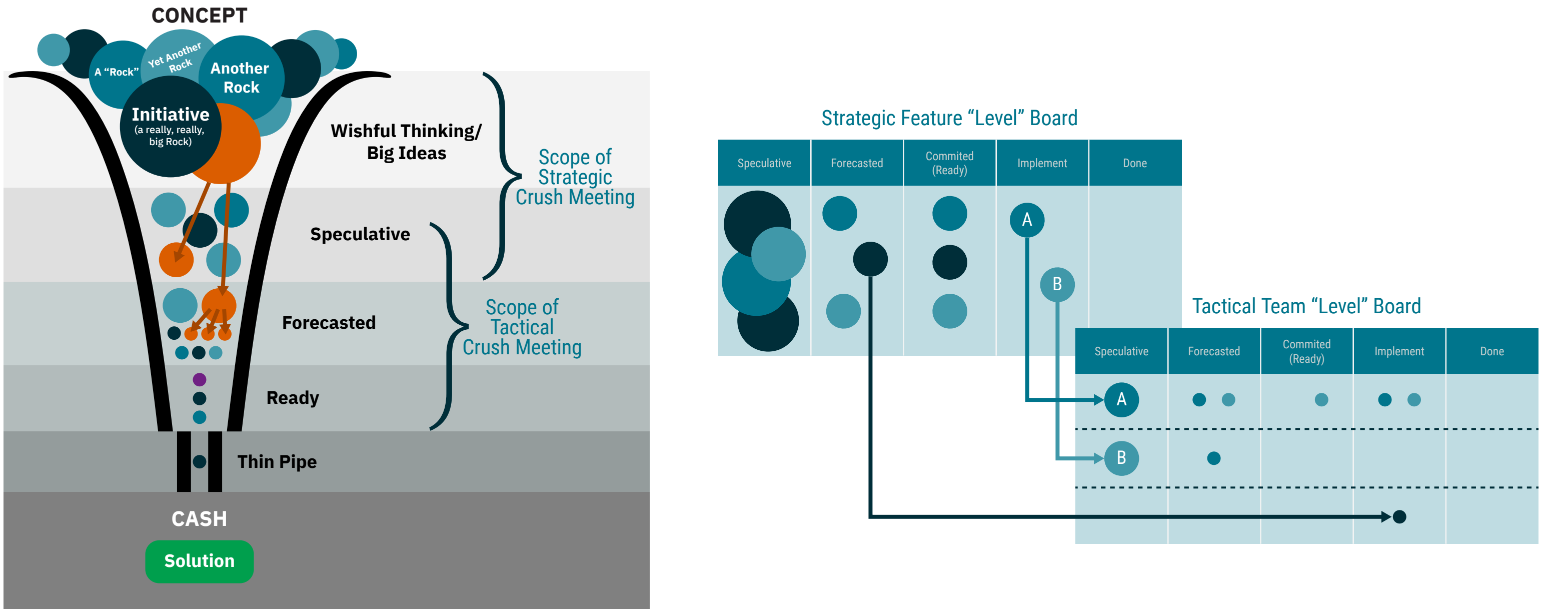
A "Feature Level" board represents the workflow of the big "feature size" Rocks. These "features" may represent the minimal collection of behaviour that can be released and are the unit of work that stakeholders are really interested in. The intent of the Feature-Level Kanban board is to get a feature to the "implementation" state, which means each Rock is well enough defined to start breaking it down into smaller chunks of work.



Linked Hierarchical Kanban Boards for a Single Team

In this example, features "A" and "B" are being implemented by the team. The Team-Level Kanban board is an example of the "simple Kanban board," with a swim lane for each feature. There is the link between a Forecasted feature and the implementation. In this case, this could represent work the team is doing to do a spike, or to use a Crusher to model and better understand the feature.

This implementation works well, especially when a team uses the "strategic-tactical" split for implementing the Rock Crusher.

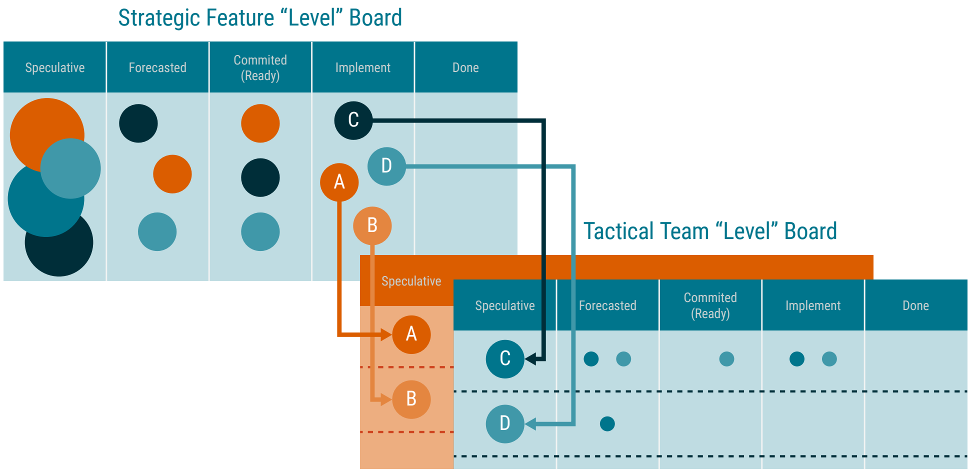


Using Kanban Boards to Implement the Strategic-Tactical Split

Linked Hierarchy of Kanban Boards – Multiple Teams

When multiple teams are collaborating to deliver a common solution and sharing a common backlog, then use the multiple Kanban boards to model the Rock Crusher. Like the single team case, there is a single higher-level Kanban board representing the lifecycle of “big Rocks,” and then multiple team Kanban boards representing the lifecycle of smaller Rocks that individual teams are crushing. This model can scale to support multiple teams.

This technique of using of multiple linked Kanban boards is used in SAFe (essential configuration) to link and create alignment between teams on a release train.



Multiple Team Kanbans

The "Thin Pipe"

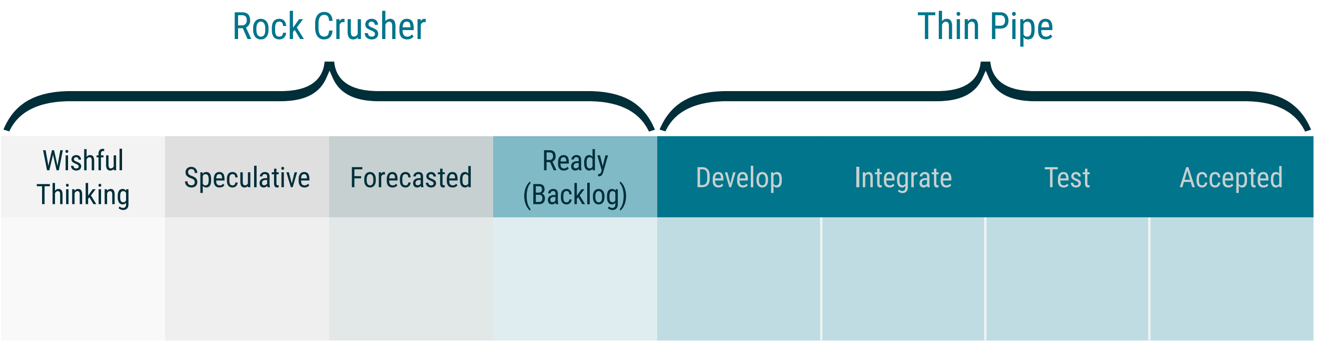
The Thin Pipe is the Rock Crusher metaphor for the steps required take a Rock from Ready (as defined by the team's "Definition of Ready") to "done" (as defined by the team's Definition of Done). The Thin Pipe metaphor is intended to evoke a mental model where the Rock Crusher must get Rocks to Ready so they are sufficiently small that they can flow through a Thin Pipe without clogging it.

Prior to the Rock Crusher, a typical team board for getting something from Ready to "done" may look like: Backlog->Develop->Integrate->Test->Accepted.

Backlog	Develop	Integrate	Test	Accepted

Typical Team Kanban Board

When the team board is organized like this, then how work arrives in the backlog is invisible and unmanaged. The intent of the Rock Crusher is to make the steps prior to "backlog" visible and encourage flow, rather than use the backlog as a reservoir. A complete Rock Crusher-styled Kanban board for a small team may look like:



Adding the Rock Crusher to a Team Kanban Board

The Rock Crusher expands the backlog into the Wishful Thinking, Speculative, Forecasted and Ready categories, which makes the backlog refinement work highly visible. The Thin Pipe is then the subsequent steps a team performs to take a Rock from Ready to "accepted."

Summary

Most tools don’t provide much support for backlog management beyond the stacked plates model. However, the Rock Crusher can be approximated using “cutlines” or Kanban boards.

An individual or small tight-knit team can simulate a minimal version of the Rock Crusher using one or two cutlines.

An individual team requiring more sophisticated management of the value stream should make use of a Kanban board.

Use WIP limits to manage the flow of work through the value stream represented by the Rock Crusher.

Use Linked Kanban boards for more sophisticated implementations of the Rock Crusher.

Learn More

Rock Crusher for Backlog Management

Read:

1. Introduction
2. Ceremonies
3. Implementing
4. Roles
5. Crushes/Backlog Items

Considered Harmful
Anti-Patterns
Re-acquainting
Work not Road Mapped

View:

Rock Crusher Infograph

User Story Infograph