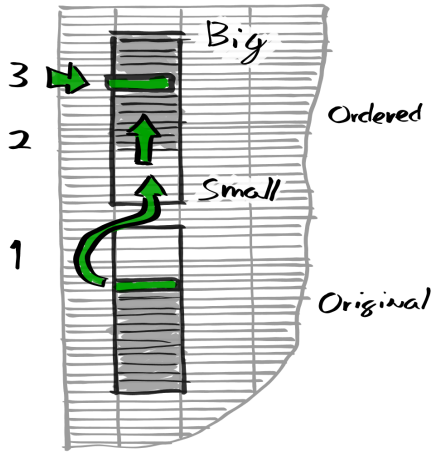
CrumbScale

Step 1 - Size Ordering

By only asking ourselves the simple question “Is this Story bigger than that other one?”, we will take the Backlog elements, one by one, and drop them into an ordered list until all the items are transferred from the initial *unordered* list into the new *ordered* list. This new list will be ordered by effort size where the size is not based on the elapsed time but on the amount of work (effort) needed, whether this amount results from quantity or from complexity. For example, a very simple but repetitive task may be the same *effort size* as a single but complex task (e.g. the creation of a single complex algorithm might take as much effort as a repetitive refactoring job).

Now that we are ready to start, we will take the elements from the unordered bottom list, one-by-one and drop them into the top list while sorting them by size. As you may have guessed, there are a few rules for this to work properly. The top (ordered) list will be ordered from big to small from the top to the bottom (big stories at the top, smaller ones at the bottom. See the following Figure). 1) Take a new Story from the Unordered list and insert it into the Ordered list by entering from the bottom, going up. 2) As you are moving up in the Ordered list, for each Story you encounter ask the question “Is this new Story bigger than that other one?” If the answer is *yes*, move the Story up by one position and ask the question again. Repeat until you have either found a Story that is bigger or reached the top of the list. 3) Once the answer is *no* (or you have reached the top), you have found the right spot to drop the Story. This process should never take more than 2-4 minutes per Story.



Take the next Story in the bottom list and repeat the process until all the Unordered Stories are inserted into the Ordered list. Each new Story is inserted this way, from the bottom up, from the smallest to the biggest, and the size comparison must be done for each Story encountered. This will create a smooth graduation of ever-increasing effort size. This smooth

graduation is an essential part of the CrumbScale method. Yet, even if this first step sounds simple, everybody makes the same mistake. After a while, the second list will begin to grow and you will be tempted to start dropping things directly into the list instead of inserting them from the bottom and moving up one Story at a time. This is a sure way to create errors in the sorting. Our brains are easily tricked and this method, even if it feels babyish, is designed to keep us from making mistakes. Here is another important point: as you insert the items into the top list, you will sometimes find an ordering mistake. Most mistakes are created by not following the previous rule but honest mistakes will also happen. You must keep the size progression smooth. The bottom of the ordered list is populated by the tiny elements with the smaller size differences and therefore a miscalculation can easily happen. Fix it right away. Remember that you are only doing this process once for the whole project.

Step 2 - Points Distribution

This second half of the CrumbScale points distribution goes much faster than the first one. Working with our newly created ordered list, we will distribute the effort points starting at the bottom of the list and moving up. We will use the following measuring scale:

½, 1, 2, 3, 5, 8, 13, 20, 40, 100

At the bottom of the ordered Stories list, you will almost certainly have really small tasks falling in the easy work category. If you had to include such a task in your workday, it would have no impact on your schedule. I always call such *easy* work "crumbs" hence the name of the method. Crumbs are the small things that must be done but are so small that it is annoying to even list them. Examples of crumbs are: installing software, configuring a system, or gathering some test data. Give the value of ½ to every crumb as you go up the list until you find the first Story that doesn't fit the 'crumbs' definition. This Story is the first sizable Story and you can give it the value of 1.

As you now go up the list, ask the question "Is this next Story the same size or [almost] twice as big as the last Story?". If this next Story is clearly bigger (almost twice the size), switch to the next value of our CrumbScale measure and keep going. In other words, if you were currently assigning values of 1, you switch to the next value of 2. If you were currently assigning values of 2, you switch to the next value of 3*. Keep assigning values and incrementing to the next level of the scale when needed until you have reached the top of the list. This process should never take more than 1-2 minutes per Story.

> **The step between 2 and 3, being only an increase of 50%, is smaller than the other. Because we are working at macro level estimation and for the sake of simplicity, we will still ask the same question when passing from 2 to 3; "Is this next Story the same size or [almost] twice as big as the last Story?"*

More Info

GitHub

Here's a 10 minutes video explaining CrumbScale

Get more information by reading Rally-Point Backlog,
available on Amazon (eBook or paperback)