

By Ewald

Behind the scenes: the Backlog Priority or Stack Rank field



Ewald

May 14th, 2014

The backlog in TFS is an ordered list of work items, such as the product backlog. In Scrum you get the ordered list of PBIs and Bugs, in Agile you will see User Stories and CMMI contains the Requirements and Change Requests. The order of the backlog items is dictated by the the “Order” field as defined in [the process configuration](#). The process templates that we ship use different fields for this “Order” field. The Scrum template uses the field called “Backlog Priority”, while the Agile and CMMI templates use the field called “Stack Rank”.

In the screenshot below (which is taken from Visual Studio Online) you see a product backlog that is based on the Scrum template. As you can see in the Backlog Priority field, the values are not from 1..N. In this blog post, I want to explain why it is not consecutive and what is changed in TFS 2013 Update 2.

Backlog items

BacklogBoard

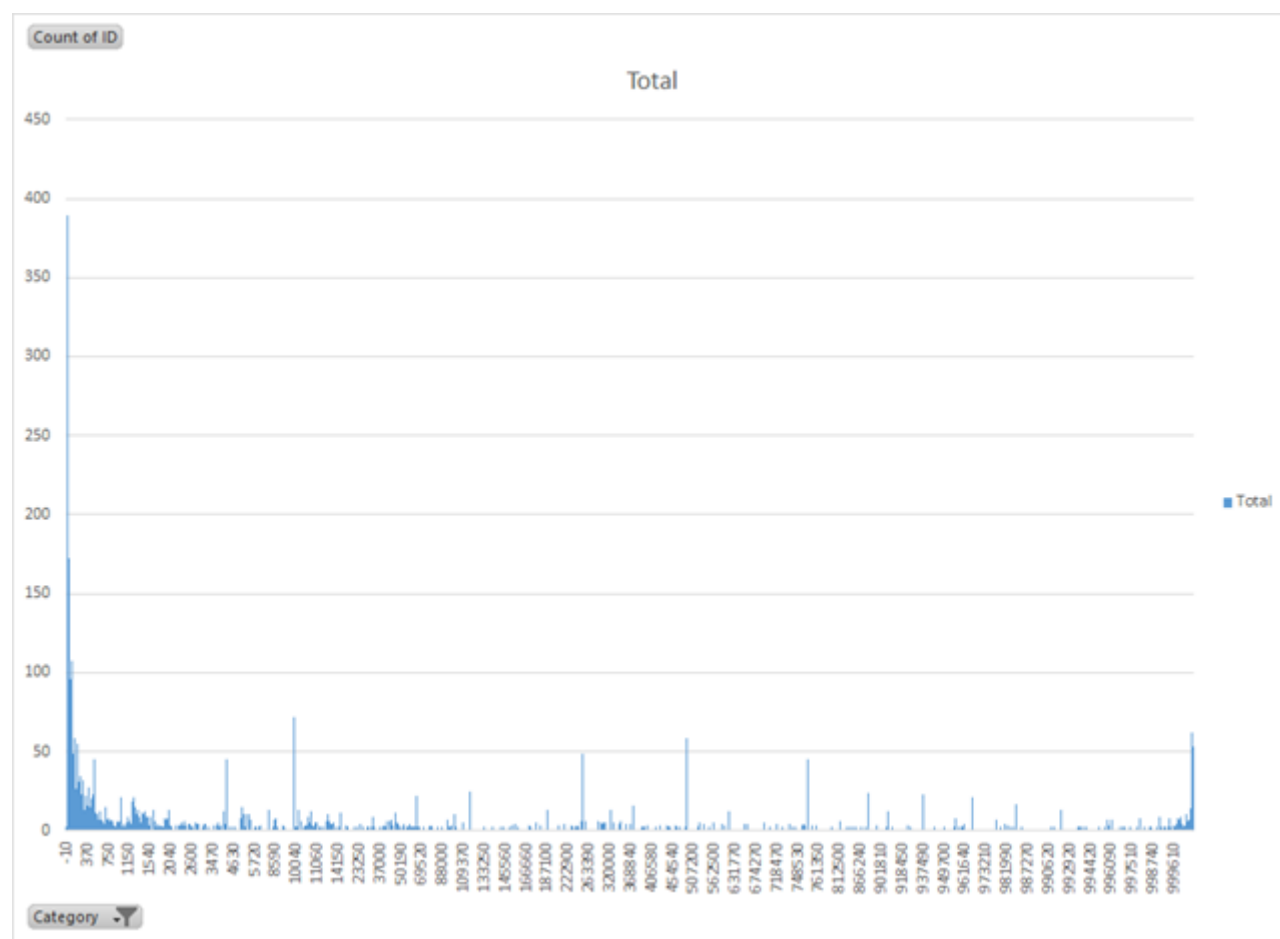
NewAddRemoveCreate queryColumn optionsEmail

Order	Work Item Type	Title	Backlog Priority
1	Product Backlo...	Customer should see weather-related outages on portal	400000000
2	Product Backlo...	Technician can send GPS location from Windows Phone	800000000
3	Bug	Customer with Canadian addresses not displaying properly	1200000000
4	Product Backlo...	Technician can edit customer contact details on Windows Phone	1600000000
5	Product Backlo...	Service rep can view service ticket details from the dashboard	1644444444
6	Product Backlo...	Technician can check on parts orders on Windows Phone	1688888888
7	Product Backlo...	Technician can look for closest hardware store from Windows Phone	1733333332
8	Bug	Customer is unable to logout from the application	1777777776
9	Product Backlo...	Technician can submit invoices on Windows Phone	1822222220
10	Product Backlo...	Customer can view service invoices online	1866666664
11	Product Backlo...	Customer can pay invoices online	1911111108
12	Product Backlo...	Customer can opt-in/opt-out of paper billing	1955555556

Imagine we had used a consecutive range of values of 1..N, then moving Item #1 to position 4 would mean that not only #1 needs to be updated, but #2, #3 and #4 as well. Roughly the reorder operation gets 4 times as slow. And imagine that you move #1 to position #200 or even further down.

When we shipped TFS 2012, the product would set the Backlog Priority field to a value to create big gaps, so many items can be added to the backlog (or reordered) without updating any other backlog item. Prior to TFS 2013 Update 2 we used the upper limit of the range of 1,000,000, and used an halving logic to create the big gaps. The first item on the backlog would get the value 500,000, the second item (assume it goes to the top of the backlog) 250,000 and so on. When you continue adding items to the top of the backlog, you would get to a Backlog Priority of 1 for the 20th item. When the user now adds or moves an item to the top of the backlog, multiple items need to be updated to create unique Backlog Priority field values (we only use whole numbers). We anticipated this scenario, and included a feature we call “sparsification”. This sparsification logic starts with the current item and walks down until it finds a gap. All the items in that range are then redistributed to create gaps for new backlog items without affecting others.

When we reviewed the sparsification logic with metrics that we capture on Visual Studio Online and our internal servers, it became clear that the logic was not very effective with how people interact with the backlog. The picture below shows the distribution of the items on our internal Pioneer server (VSO showed similar results). On the chart you can clearly see the spikes on the ‘halved’ numbers such as 500,000 and 250,000. But what is even more important is the ‘dust in the corners’. The top and the bottom of the backlog it is very dense. The result is that the sparsification runs often causing slower performance while working with the backlog. We have had reports where every reorder on the backlogs took over 10 seconds due to this behavior.



In Update 2 we have made modifications to fix this:

1. Increased the upper bound of the range to 2 billion
2. Optimized the scanning for gaps when the sparsification runs, which results in less runs of the sparsification
3. Optimized the logic to generate the new number (as opposed to the halving logic) to make better use of the available gaps


After we shipped the change, we have received multiple reports from customers who were confused with the change. If the customer had set the Backlog Priority field via Excel to values 1..N, the old sparsification logic kept the range values pretty much intact (especially the top items). With Update 2 however, once you add or reorder items on the backlog, the sparsification logic sets the field values to big numbers to introduce big gaps, and give you good performance. This change is intentionally, and by design. The Backlog Priority field was always intended to be a system field, and our recommendation is to not update field manually.

You can still use Excel to quickly set the order of the backlog with the values 1..N, but be aware that the sparsification will kick in. And although the values are different, the order of the items on the backlog are still the same!


Hopefully I have been able to give you more insight in the inner workings of the backlog, and you understand better what is going on.

Ewald Hofman


TFS Program Manager



Ewald Hofman

Follow Ewald 

Posted in [Uncategorized](#)



Log in to comment

Log In

No Comments.

Relevant Links


[Learn more about Azure DevOps](#)

[Documentation](#)


[DevOps at Microsoft](#)

[Visual Studio blog](#)


Top bloggers




[Edward Thomson](#)
Principal Program Manager




[Pratap Lakshman](#)
Senior Program Manager



[Erin Dormier](#)
Release Manager




[Anisha Pindoria](#)
Senior Program Manager



[Joe Bourne](#)
Senior Program Manager

Twitter Feed

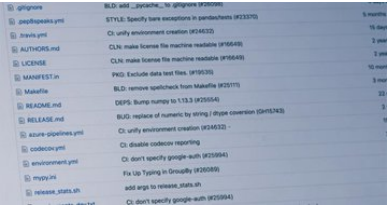
Tweets by @AzureDevOps



Azure DevOps

@AzureDevOps

New from [@martinwoodward](#)... 88 years of free open source builds in 8 months with Azure Pipelines - [aka.ms/build19/azured...](#) [#AzureDevOps](#)



Embed

View on Twitter

Stay informed



What's new

NEW Surface Pro 6

NEW Surface Laptop 2

NEW Surface Go

Xbox One X

Xbox One S

VR & mixed reality

Windows 10 apps

Office apps

Microsoft Store

Account profile

Download Center

Microsoft Store support

Returns

Order tracking

Store locations

Buy online, pick up in store

Education

Microsoft in education

Office for students

Office 365 for schools

Deals for students & parents

Microsoft Azure in education

Enterprise

Microsoft Azure

Microsoft Industry

Data platform

Find a solution provider

Microsoft partner resources

Microsoft AppSource

Health

Financial services

Developer

Microsoft Visual Studio

Windows Dev Center

Developer Network

TechNet

Microsoft developer program

Channel 9

Office Dev Center

Microsoft Garage

Company

Careers

About Microsoft

Company news

Privacy at Microsoft

Investors

Diversity and inclusion

Accessibility

Security