

# Risk Priority Number

## Table of Contents

Carnegie Mellon University - Notices .....	3
Risk Priority Number: A Method for Defect Report Analysis .....	4
Agenda .....	7
A Generic Example – Comparing Four Defects .....	8
How do we judge importance? .....	9
RPN General Explanation -1 .....	10
RPN General Explanation -2 .....	12
RPN General Explanation -3 .....	13
Polling Question .....	14
Risk Priority Number: A Method for Defect Report Analysis .....	15
Expected Range of Application .....	16
Example Usage – scenario .....	18
Example Usage 1 .....	20
Example Usage – 2 .....	21
Risk Priority Number: A Method for Defect Report Analysis .....	23
Sample Scales .....	24
Rating Scales – Severity – System Function .....	25
Rating Scales – Severity - Operational Impact .....	27
Rating Scales – Detection .....	28
Rating Scales – Occurrence .....	30
Polling Question 2 .....	31

Using Proportional Scales .....	33
RPN – An Example – Weighted Average .....	35
Polling Question 3 .....	36
Resource Available .....	39
Sample Data Description.....	40
One way to look at the sample data .....	41
Four Analysis Methods.....	42
Analysis Method - Functionality .....	43
Draft Analysis Method - Functionality .....	44
Second Analysis Method – System Risk.....	45
Top 10 RPN DRs.....	46
Third Analysis Method – User Ranking.....	47
Top User Ranked DRs.....	48
Hybrid Method – Start with User Ranking.....	49
Hybrid Method – Then Consider Functionality.....	50
Hybrid Method – Determine What Else To Include.....	51
Hybrid Method – Final Listing.....	52
Other uses .....	54
Suggestions for DoD Usage.....	56
Questions? .....	58
Contact Information.....	71

## Carnegie Mellon University - Notices

# Carnegie Mellon University

This video and all related information and materials (“materials”) are owned by Carnegie Mellon University. These materials are provided on an “as-is” “as available” basis without any warranties and solely for your personal viewing and use.

You agree that Carnegie Mellon is not liable with respect to any materials received by you as a result of viewing the video, or using referenced websites, and/or for any consequences or the use by you of such materials.

By viewing, downloading, and/or using this video and related materials, you agree that you have read and agree to our terms of use ([www.sei.cmu.edu/legal/](http://www.sei.cmu.edu/legal/)).

© 2014 Carnegie Mellon University.

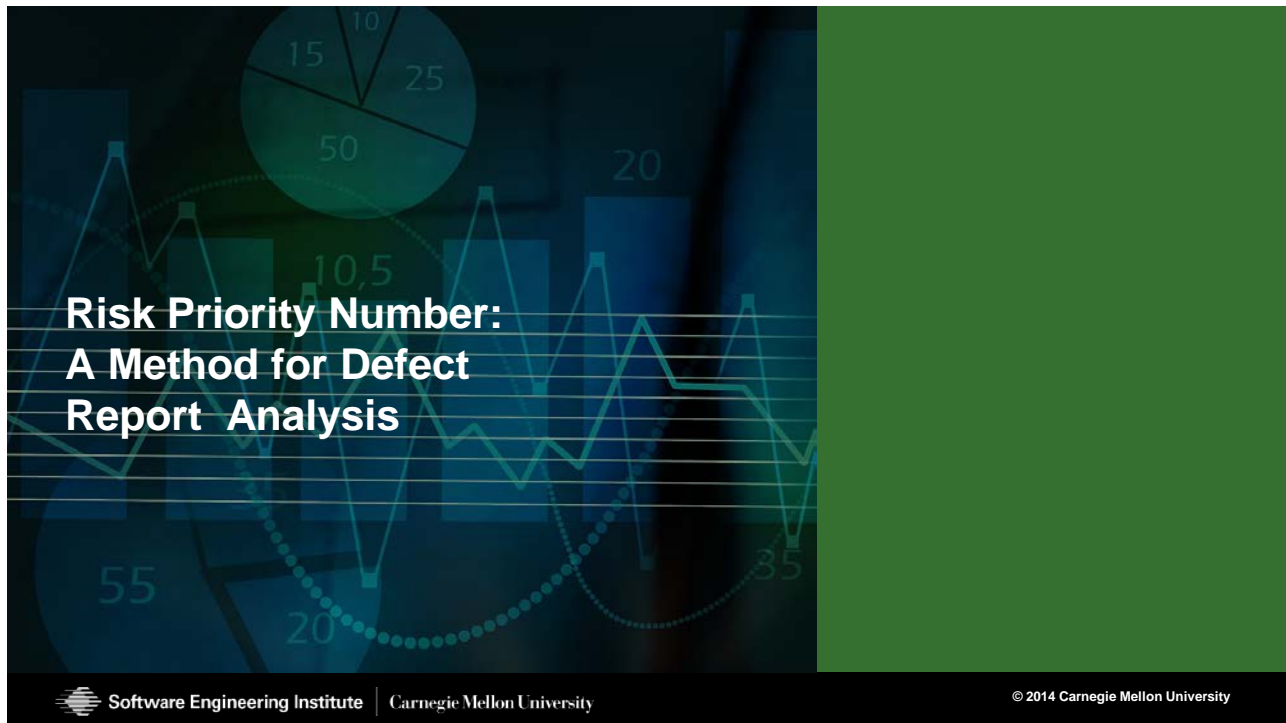


Software Engineering Institute

Carnegie Mellon University

© 2014 Carnegie Mellon University

## Risk Priority Number: A Method for Defect Report Analysis



\*\*003 Shane McGraw: And hello from the campus of Carnegie Mellon University in Pittsburgh, Pennsylvania. We welcome you to the Software Engineering Institute's Webinar Series.

Our presentation today is Risk Priority Number, or RPN.

Depending on your location, we wish you a good morning, a good afternoon or good evening.

My name is Shane McGraw. I'll be your moderator for the presentation; and I'd like to thank you for attending.

You can-- we want today to be as interactive as possible. So we will address questions throughout the

presentation, and again at the end of the presentation.

To a log a question-- you'll see the Questions tab on your console. Simply click that tab and send in your question; and we will address as many as we can throughout the presentation.

We will also ask a number of polling questions throughout the presentation, and they will appear as a popup window on your screen. These will help us drive the flow of the presentation. So please participate in those.

Another few tabs I'd like for you- to point out to you are the Materials tab, the Twitter tab and our Survey tab.

The Resources or Materials tab has a PDF copy of the presentation slides there now; and along with a one-page document that reviews our topic for today.

For those of you using Twitter, be sure to follow @SEInews and use the hashtag SEIrp; once again it's the hashtag of SEIrp.

And lastly that Survey tab will appear at the end of the presentation; again as a popup window. We request that you fill that out as your feedback is always greatly appreciated.

Now I'd like to introduce our presenters for today.

Our first presenter is Miss Julie B. Cohen, and she has been the SEI for 10 years.

She's currently involved in activities to support and improve acquisition practices in Air Force and civil communities.

She leads the SEI support for the Advanced Extremely High Frequency satellite ground systems software effort.

In addition, she's a major contributor to the Veteran's Health Administration support, to include long-term support of efforts to modernize the VHA's scheduling system. She's also a contributor on other customer efforts and on internal research projects.

Our next presenter will be Will Hayes; and he's a Senior Member of the Technical Staff at the SEI. He directs Lifecycle Management Support to major defense- Department of Defense programs.

Throughout his 23-year career at the SEI, Will has supported numerous commercial, government and defense organizations, providing consultation and coaching for a wide range of roles.

And now I'd like to turn it over to Julie Cohen. Julie, all yours.

Julie Cohen: Thank you Sean.

## Agenda

### Agenda



**General Explanation of Risk Priority Number (RPN)**

**Suggestions for RPN for DoD Systems Usage**

**Examples**

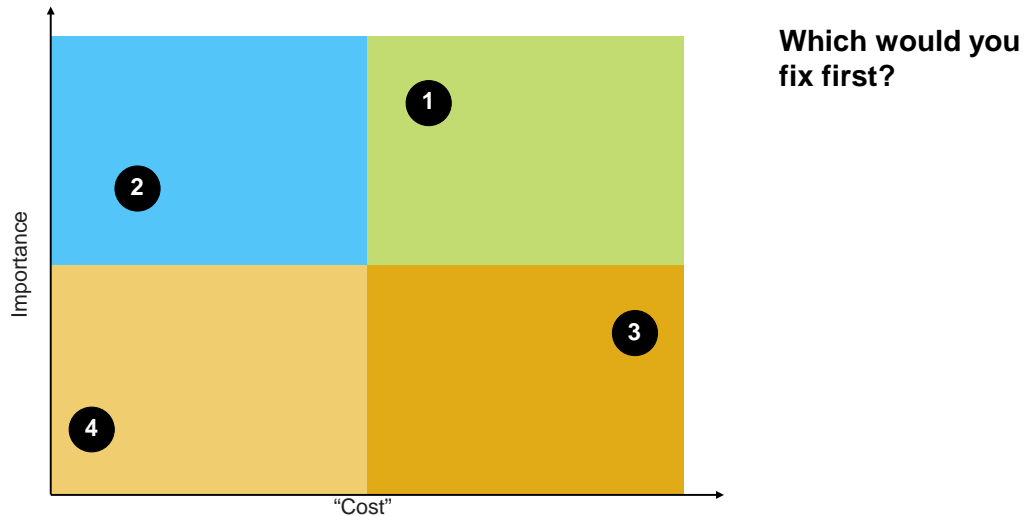


\*\*004 Welcome to this briefing on Risk Priority Number.

We'll start with a general explanation of what the Risk Priority Number process is; some suggestions for how you can use it-- and the suggestions will come from the Department of Defense world but certainly they're transferable to pretty much any other application-- and then we'll work an example.

## A Generic Example – Comparing Four Defects

### A Generic Example – Comparing Four Defects



\*\*005 So just starting with a very generic example. If we look at a very simple plot of four defects showing importance and cost, which one would you work first?

Well it's pretty obvious that you would work 1 before you would work number 3 because it's more important and it costs less.

But would you necessarily fix 1 before 2? And what about number 4? Well it's the least important. It's also the least cost.

So even in a very simple example you can see that sometimes it's not obvious which defects you might want to want fix first...



## How do we judge importance?

### How do we judge importance?

#### Using “severity” alone has issues

- People are tempted to negotiate a severity rating to account for the importance they perceive
- Without a way to discuss what makes things important, the conversation may become a competition among advocates

#### RPN focuses on risk exposure

- Allows the team to assess the priority of fixes
- Can relate priority to the understanding of risk

#### Risk can be perceived from different viewpoints

- User, developer, cost, time
- May need multiple views to make the best decision



\*\*006 ...in a given situation.

Oftentimes we use "severity" as a way to define what the defects are. And there are some problems with using severity alone. It often causes users to inflate the severity rating because they really think it's important to get their defect fixed; and the only way they can get it fixed is if they inflate the severity.

And so using severity alone can often cause problems with how people rate the severity. The severity may not actually reflect the risk based on the defect but how much somebody wants to get it fixed.

And so RPN focuses on risk exposure. So it allows the team to assess priority using a different way other than just the

severity; and it allows risk to be shown from different viewpoints.

The user might have one viewpoint; a developer might have another. Different users have different viewpoints. And so RPN allows those viewpoints to be part of the process for assessing the defects and which ones to fix first.

You may need multiple views; and it's very important usually to get multiple views because people do have different ways of looking at these defects. And you'll see- as we talk about the scales and as we work through the example you'll see how these multiple views work.

## RPN General Explanation -1

### RPN General Explanation -1

**Generally based on processes that were developed from reliability and cost methods**

- **Severity:** a rating of the adverse impact of the defect – a measure that reflects the negative consequence to the users or developers
- **Occurrence:** how often the defect is encountered and/or how long it takes to recover functionality – a measure that reflects a different element of the impact of the defect
- **Detection:** how easy it is to spot the defect is when it occurs – a measure that reflects the risk of unmitigated consequences if the defect is not remedied



\*\*007 So we'll start with just a

general explanation. The Risk Priority Number method comes from the failure modes and effects analysis world. And a colleague, Bob Ferguson, also helped us develop this methodology.

And at its bare minimum it's pretty simple. There are three basic areas which you'll look at to determine the overall risk.

The first one is Severity. How bad is the impact of the defect? Just what does it actually cause to happen with respect to the person who's writing up that defect? If it's a user or if it's a developer, what's the impact?

The second one is Occurrence. How often is it encountered? And as well, how long will it take to restore usage of the system when it's encountered? And those combine to form an Occurrence rating.

And then finally Detection. How easy is it to spot it when it occurs? And I liken this to your car. If you didn't have an engine check light, if you ran out of oil and you didn't know you ran out of oil it's a totally different situation than when that engine check light comes on; and it gives you a chance to know that you've run out of oil before severe engine damage occurs. So that's what the detection element is about.

## RPN General Explanation -2

### RPN General Explanation -2

**For weapon systems these may equate to:**

- **Severity** = Threat to mission success (Operational and System)
- **Occurrence** = How often it happens, how much time to recover
- **Detection** = Ability to detect that the problem has occurred



\*\*008 For a weapons system these three values may equate to, in severity, what's the threat to mission success? Can I go ahead and perform the mission? Might I have to cancel or only do a partial mission? Or can I go ahead and do it as planned?

The occurrence, just as I said before, how often it happens, how much time to recover. Detection generally becomes more important when I'm talking about the areas of security and safety, in a missions system.

## RPN General Explanation -3

### RPN General Explanation -3

#### RPN includes:

- Rating scales characterizing elements of:
  - Severity,
  - Occurrence
  - Detection
- Scaling values for the ratings
- (Optional) Weighting for each rating scale to emphasize what matters most/least in a given system

$$\text{RPN} = \text{Severity} \times \text{Occurrence} \times \text{Detection}$$

- A weighted sum, rather than multiplying the numbers together, can be included as an option



\*\*009 Finally RPN includes rating scales that look at the severity, occurrence and detection; and my colleague Will Hayes will describe these in much more detail in the next section.

And you also need scaling values for those ratings; and he'll explain those as well.

And then a way to combine these. In the original application of RPN it was a simple multiplication: Severity X Occurrence X Detection.

But you can use a weighted sum rather than just a simple multiplication, if that's easier to help explain to users of this system how these factors combine to get the

overall Risk Priority Number or the overall risk to the system.

And now we have a polling question for you.

## Polling Question

### Polling Question

Would you like us to explain the basic premise of RPN in greater detail?

- Yes
- No



\*\*010 Shane McGraw: Okay folks. As I mentioned during the intro, we'll ask a couple of polling questions just throughout the day. It's going to help us with the flow of the presentation.

So based on Julie's overview would you like us to explain the basic premise of what RPN is in greater detail? Or is what she provided so far sufficient to proceed?

So we'll give you about 10 or 15 seconds to vote there. We'll take a

look at the results. And based on the feedback, we'll go on from there.  
Great.

And I'm going to stop the poll. So I hope everybody had a chance to vote there.

And our results are 68 percent No and 32 percent Yes. So we have a sufficient understanding to proceed.

## Risk Priority Number: A Method for Defect Report Analysis

Risk Priority Number:  
A Method for Defect Report Analysis

Software Engineering Institute | Carnegie Mellon University

Risk Priority Number  
October, 2014  
© 2014 Carnegie Mellon University

© 2014 Carnegie Mellon University

\*\*011 Julie Cohen: So I'll turn it over to my colleague, Will Hayes, who will continue with the explanation.

## Expected Range of Application

### Expected Range of Application

Development, operation, and sustainment contexts are all candidates for adapting RPN to support decision making on which defects to fix first

Keys to successful usage

- Custom rating scales developed with appropriate personnel
- Socializing draft materials with stakeholders
- Buy-in from participants in existing defect review processes



\*\*012 Will Hayes: Great. So there's a range of applications that we expect to consider with this methodology. And our colleague in the LA office, Bruce Grant, has helped us to think more clearly about some of the development applications.

Our initial experience has been with a system that is in early fielding; and so the kinds of decision points that apply there may differ from an organization that's working on brand new products that are about to be fielded or will be fielded in the future.

In a development context you might be thinking about prioritizing a backlog of defects that need to be resolved in anticipation of your first release; and the Risk Priority Number



might help you to decide which defects if removed would contribute most to the value proposition your new product offers.

As well you might have a sorting rule for defects that occur during a system test event to help you identify which sets of defects discovered require more immediate, more concentrated attention, and which defects might be postponed for later examination because of the need to continue in the testing regime.

So as we think about applying these concepts in these variety of applications, there are a number of key points that we've uncovered through our experience that contribute to success.

One of the most important things in applying something like this is to adequately harness and understand the local knowledge that exists. As we think about attributes of the system, knowledge of that kind of information is essential to being able to triage or to make reliable judgments about the impact a defect might have; and so these scales help you to quantify that and make that a much more explicit thing. But that wouldn't be possible without involving people who have a deep knowledge of the system.

Also the communicating of preliminary results is one of the most important aspects of maintaining the momentum that you establish by involving the local staff; and then

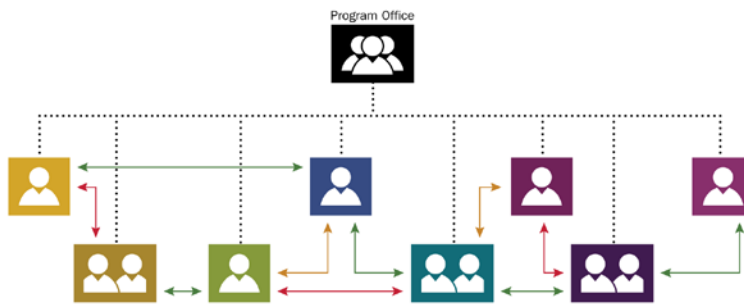
finally as you come to preliminary results if you're able to show the progression of evolution that you've seen in the methodology, that helps a great deal in getting people to understand and buy in to the activity that you're engaged in.

## Example Usage - scenario

### Example Usage – scenario

A major weapon system in early fielding is looking for a way to plan the contents of releases comprised of DR fixes

- Diverse user community with legitimate competing priorities
- Limited funding for future work (many DRs will never be fixed)
- Program office motivated to maximize system utility/value



\*\*013 So we'll dive into a very specific example here. And Julie will follow with a quantitative set of information that will allow you to see how the numbers play out. But what we want to do here is kind of lay the foundation.

So the system that we've got initial experience in, they were planning upcoming releases using a set of Discrepancy Reports or Defect

Reports that exist against a system that is in the early process of fielding.

It has a broad set of missions that this system needs to support. There are a variety of different user communities, each community with a potentially unique usage profile for the system. And so the kinds of defects we encounter will be sensitive to the usage profile.

And so if this defect affects something in my usage profile, I'm going to be very motivated to see that it gets resolved. But if this defect doesn't apply to something that reflects my workflow, then I might be less concerned or I might have a difficult time seeing how severe its impact is somewhere else.

And the idea here is to find a way to make that an even playing field with people that come from different perspectives and have different priorities.

And one of the important aspects of our initial experience was the Program Office's engagement. The Program Office in major contracts or major systems of the type we're talking about has a long and broad view of investment and utility that you really need to have in applying something like this; because the long-term decision-making process is affected by how you prioritize what's worked on today.

## Example Usage 1

### Example Usage 1



1. A small working group was formed
  - Representatives familiar with existing DRs for this system
  - A member of the program office staff who understands the vision for the system
  - Measurement coach who can help navigate the process of constructing measurement scales
  - Draft rating scales were developed as well as computation procedures



\*\*014 Going on to the next slide; speaking about now the sequence of events that we orchestrated as we piloted these concepts.

The first thing-- I will repeat again-- the importance of local knowledge. So getting representatives from the diverse user communities, from the different stakeholder groups, to participate as a member of the working group was an essential ingredient to success.

Again the Program Office's long and broad view to be able to help people understand beyond the next release or beyond the next profile of utility; what's the long-term evolution in mind for the system?

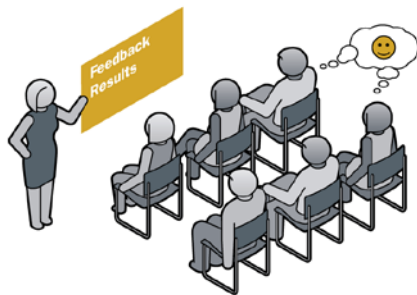
And then a coach; someone who helps to kind of listen carefully to the local knowledge and really channel that into a set of measurement scales that can result in consistent judgments being made and good decision making being supported.

And finally establishing agreement at the working level with your initial take on how to approach this is really essential; because you want people to go forward with agreement that we've captured a set of knowledge that is good to build on as a starting point.

Let me go to the next progression then.

## Example Usage - 2

### Example Usage – 2



3. Draft materials were reviewed with user communities

- The reasons for using RPN were explained and tied to the current decision processes
- The rating scales were explained to people who write DRs or who champion DRs to be included in releases
- Worked examples of real defects to discuss how ratings are assigned

4. Rating scales and procedures were updated based on feedback



\*\*015 With that initial set of draft materials, we were working for a very

smooth transition. We wanted folks to be involved and to offer adjustments to the methodology; to propose alternative ways of weighting different components or to experiment with different scales.

And the development of spreadsheet tools and quick ways of using forms to get feedback was really essential to letting people make adjustments.

And then telling people about those adjustments in subsequent meetings and letting them see the kinds of influence the user community or other stakeholders could have was important.

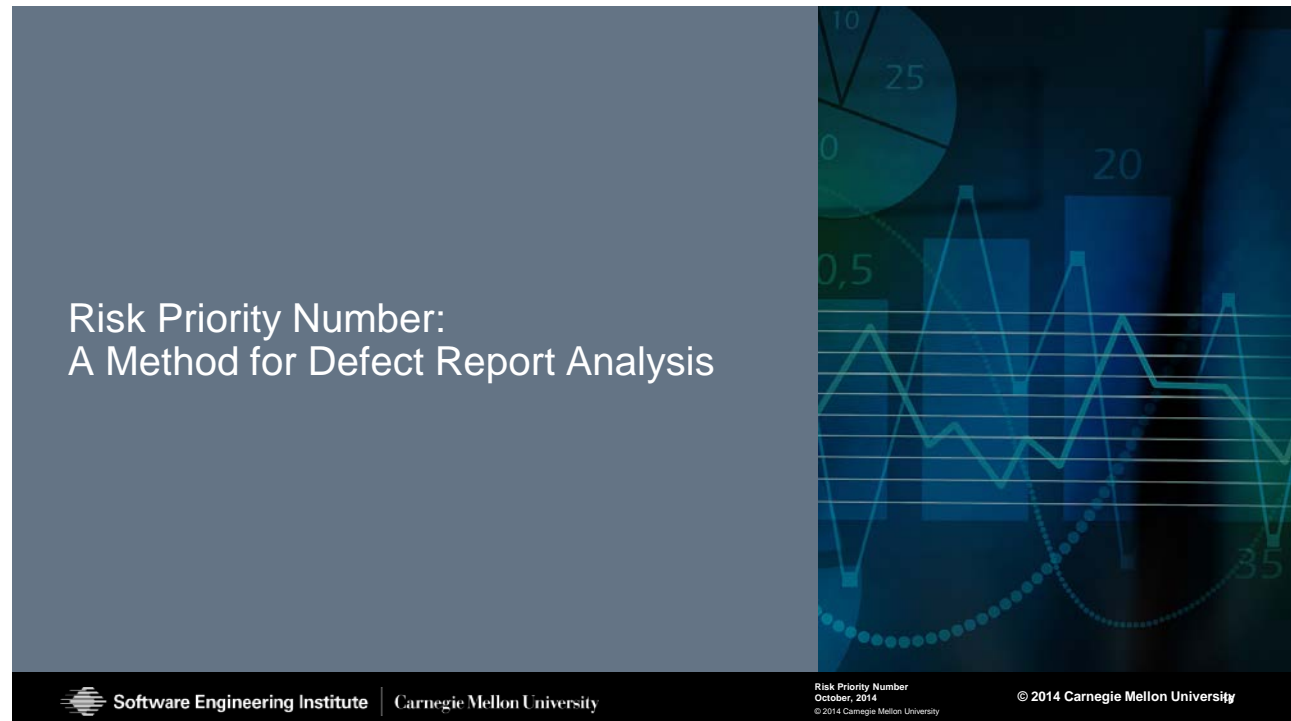
And this showed- demonstrated - a respect for local knowledge and the need to use that knowledge to implement what we were talking about.

And then finally there's nothing more powerful than actual data. So looking at past defects that have been removed and resolved and applying the proposed algorithms with that data allows us to then feed back: Here's what it would look like had we used this methodology at that time.

The experience of the people from that past event allows them to make good judgments about whether or not the methodology is serving them well going forward.

And so those are some of the keys that we found to having a successful pilot and rollout.

## Risk Priority Number: A Method for Defect Report Analysis



The slide features a dark blue background with a light blue sidebar on the left containing the title. The right side of the slide is decorated with a semi-transparent data visualization consisting of a pie chart, a bar chart, and a line graph. The pie chart has segments labeled 10, 25, and 0. The bar chart has a bar labeled 20. The line graph has a data point labeled 35. At the bottom left, there is a logo for the Software Engineering Institute and Carnegie Mellon University. At the bottom right, there is a copyright notice for 2014 Carnegie Mellon University.

Risk Priority Number:  
A Method for Defect Report Analysis

Software Engineering Institute | Carnegie Mellon University

Risk Priority Number  
October, 2014  
© 2014 Carnegie Mellon University

© 2014 Carnegie Mellon University

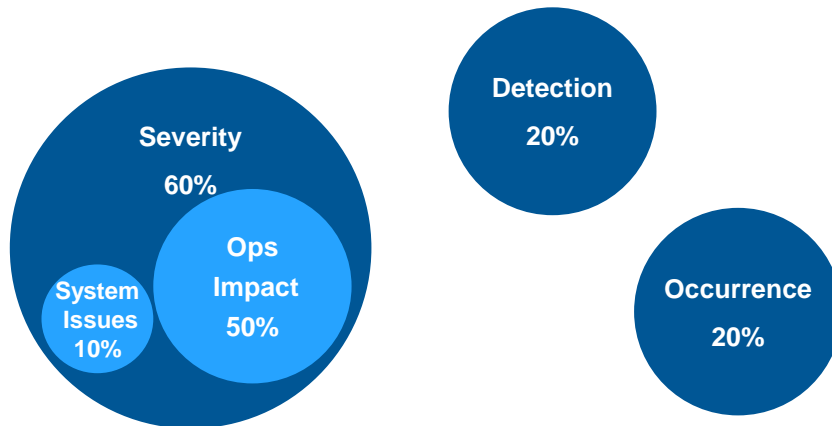
\*\*016 Now I want to talk a little bit in more detail about the content of the scales in the methodology itself.

So Julie is going to follow up with scenarios that have fictitious data against these scales. And so you'll be able to see further how this plays out when she begins her section next.

## Sample Scales

### Sample Scales

The following example covers scales developed to fit a specific context, with active involvement of stakeholders.



\*\*017 So the scales that we constructed with this pilot experience-- remember there are three primary components: severity, detection and occurrence.

In the group that we were working with, it seemed to make good sense-- and our piloting and our trial runs with the data supported this-- it made sense that severity was most heavily weighted; and detection and occurrence were equally weighted, each at 20 percent.

And in this case a weighted sum was chosen instead of a product multiplying these together. But that's a nuance, a detail you may not be as interested in.









The Severity scale now is made of up a number of different components. And the operational impact, given the nature of the system and the experience of its users, operational impact was viewed as the most important consideration in judging the risk inherent in allowing a defect to persist in the system.

And there are a number of other components that make up severity. But we'll speak in more detail about each of those now.

## Rating Scales – Severity – System Function

### Rating Scales – Severity – System Function

1		Minor System Malfunction
2		System Malfunctions or Fails to Execute Some Functions but work-around exists
3		Interruption in System Functionality Requiring operator intervention
4		Interruption in System Functionality Requiring contractor Intervention
5		Severely Constrained System Functionality—difficult work-arounds needed
6		No functionality is available and task cannot be performed by any method.
		N/A

\*\*018 So this first example scale-- and a shout out to Kurt Hess who is our graphics artist, who does a marvelous job with these things.

I think most of us can relate to the notion of an automobile and how well it serves us. If you've driven a car, if you've owned a car, you understand that first level displayed on the slide: A minor system malfunction.

You've experienced probably a flat tire or a backfire; something that perhaps doesn't completely prevent you from doing what you need to do with the car-- go shopping, go pick up your children or whatever it may be-- all the way to the highest severity scale.

You see there at the bottom a car with its doors off and the wheels have come off; and we've seen some of those vehicles as well. You can relate I suspect to what's going on here.







As well, you probably know the difference between a malfunction that you can cope with and a malfunction that requires intervention of a mechanic.

And so that sort of threshold and understanding the malfunction that's inherent in the system because of a defect, the knowledge of the system and the user profile is really essential for consistent judgments along these lines.

So here's an example of a scale for severity that we used in our pilot activity; and it's one of a number of different scales.

## Rating Scales – Severity - Operational Impact

### Rating Scales – Severity - Operational Impact

- 1  Increases operator workload slightly
  - 2  Increases operator workload significantly
  - 3  Could limit/delay mission operations
  - 4  Certain delay/limit to mission operations
  - 5  Could cause mission failure
  - 6  Certain mission failure
- N/A



\*\*019 Moving on now to a scale relating to severity. This one relates to operational impact; again, the most heavily weighted component of Risk Priority Number based on our pilot experience.






That may not be the case in your situation if you choose to use Risk Priority Number. But just want to caution you to be aware of the connection to the nature of the system here.

So let's talk you through: What is it that's causing the issue that this defect causes? And if you think of, if you've ever written a Discrepancy Report or a Defect Report yourself, there are a number of elements to the write-up that make it have weight or may let it be dismissed.

And so these kinds of differences, as you talk about: it may increase the workload of the operator slightly or it is certain to cause a mission failure. The conversation that instigates, that you have about the defect, is really the essential component here. Okay?

## Rating Scales - Detection

### Rating Scales – Detection

- 1**  There is an explicit alert or warning that there is a malfunction; or the system or application fails or crashes.  
System
- 2**  Users will always notice a visible malfunction, and only novices would fail to detect the unexpected system behavior.  
System
- 3**  Users will always notice a visible malfunction, but only after other functions or workflow steps have completed.  
System
- 4**  A user may detect subtle symptoms during normal operation, but may not immediately recognize the cause.  
System
- 5**  Issue not detectable during normal operation  
System



\*\*020 Moving on. I have two more scales.

Here the Detection scale. One of the things we want to alert you to here is that safety and security issues are particularly important to think about with respect to detection. If you're unable to see that there is an issue with the system-- so look at the bottom- the bottom point on this scale, number 5.

This shows-- the system is smiling at you, like everything's fine, but in fact behind the scenes there's something gone awry and you're not able to know it.

In contrast, the lowest point on this scale is there's an explicit alert or warning. So that is the light coming on the dashboard of your automobile saying: There's an issue you need to pull over and deal with. Or there's an issue; at some point you need to put more fluid in the windshield wiper reservoir. Those kinds of differences.

Again detection for security and safety kinds of issues becomes a very important opportunity to avoid being blindsided, if you're able to think through these kinds of concepts as you're looking at your Defect Reports.

Julie Cohen: And Will, people may be wondering now, the last two slides both had ratings 1 through 6, and this one only has 1 through 5. And so we should just point out that the number of levels that you choose will depend on your system and what you're looking at.

They don't all have to be the same. You could use 1 through 6, you could use 1 through 5; you could use 1 through 3, if that's easier for the group of users that you're dealing with.

Will Hayes: Great. Thanks for that Julie.

## Rating Scales - Occurrence

### Rating Scales – Occurrence

- 1  Under 10 hours to recover
- 2  Less than a week to recover
- 3  About a week to to recover
- 4  Weeks to months to recover
- 5  Up to 3 months to recover
- 6  More than 3 months to recover

*Note: Occurrence = Number of times the defect is encountered per year x the time restore functionality*



\*\*021 So we'll move on to the next scale, the Occurrence scale.

And here there's an interesting pairing of how long does the defect disrupt the operation?; as well as how frequently does the workflow supported by the system encounter that defect?

And so a small problem that occurs every other minute can be a very severe issue. A major problem that occurs only once but has an outage for a very long period of time could also be a major issue. In some contexts those two scenarios have an equivalent impact. In other contexts they may be different.

In the pilot experience we had, we found a good way to combine

frequency of occurrence with the amount of time required to restore the system back to the original point. And so there's an example scale here. Okay?

## Polling Question 2

### Polling Question 2

We discussed two scales that equated to Severity – you could use additional scales for other forms of severity and you could also use multiple scales for detection or occurrence.

Would you like to see more examples of these types of scales or continue on to how these scales are used?

- More examples
- Continue



\*\*022 So we next have another polling question. Shane?

Shane McGraw: Okay. So polling question number 2-- and I should've mentioned this earlier. These polling questions, obviously we have to go with majority rule for- you know, in regards to time. So that's how we'll dictate where we go.

So we have discussed two scales that equated to severity. You could use additional scales for other forms of severity; and you could use also

multiple scales for detection of occurrence. Would you like to see more examples of these type of scales or continue onto how these scales are used? So your options are more examples or continue on. And we'll give you about 15 seconds to vote.

While we're waiting there, how many current customers do you have using RPN or are you looking for more people to pilot this; or what's the status there?

Julie Cohen: So we're looking for people who are interested in piloting this. We're still working with an initial customer on using this. We have a couple of others who have expressed some interest. But we're always happy to work with other customers who think this might be useful in their context.

Will Hayes: In the whitepaper that's referenced in the slides, there's a URL for it. That'll give you a set of details that correspond well to the example that Julie's going to walk through, if you want to talk with others in your organization about what we're talking about.

Shane McGraw: Okay. So let's get to our results. And 79 percent would like us to continue on. So I will share those results with everybody.

Will Hayes: Okay great.

Shane McGraw: And you can proceed.



Will Hayes: And so in the backup slides, part of the package that you'll be able to download, there are other scales available. Those are the things we would've covered had we decided to divert to that.

## Using Proportional Scales

### Using Proportional Scales

RPN is based on the use of proportional scales

The ordinal discussed in the last few slides must be changed to a proportional rating



\*\*023 Okay. So moving on; talking a bit about scaling. And this is a place where the measurement coach can really be of great assistance to you.

It turns out that humans in trying to make consistent judgments comparing things have an easier time making those judgments consistently when the scale in which the differences are expressed is something perhaps like a geometric progression; similar to what's shown here.

It's easier for me to consistently judge if something is twice as long or twice as large as another thing than if I have to make some more fine-grained distinctions of is this 10 percent greater or 15 percent greater?

And for that reason what we've done with Risk Priority Number, and what's usually done with it, is to use a consistent proportional scale like the one displayed here; which is in contrast to the ordinal scale of 1 through 6 shown all the way on the right.

The idea is for us to decide on these anchor points based on things that differ by these amounts.

If you want to follow more on this, there's a great book by a gentleman named Douglas Hubbard called How to Measure Anything; which has inspired a great many folks to think along these lines.

## RPN – An Example – Weighted Average

### RPN – An Example – Weighted Average

Based on user input the final weighed average was:

Scaled System Behavior rating scale value \* 10% +

Scaled Operational Impact scale value \* 50% +

Scaled Detection rating scale value \* 20% +

Scaled Time scale value \* 20%

Resulted in a non-continuous rating scale from 0 to 2400

Note: The four values could also have just been multiplied together, using different scales to adjust for importance



\*\*024 Okay?

Moving on now. This is the expression of the formula that we used reflected in the chart we saw earlier with the different size bubbles.

So the result, the Risk Priority Number, is computed using this written out equation; and the scale, in this case, ranges from 0 to 2400. And so a Risk Priority Number could be as high as 2,400; could be 0.

There are issues relating to the scale. If you want to pursue that further, we have another follow-up polling question to ask.

## Polling Question 3

### Polling Question 3

Would you like us to discuss the use of proportional scales and ways to combine the scales or continue with a discussion of how to use the RPN numbers

- More discussion of scales
- Continue with how to use the RPN numbers



\*\*025 And just go right to that.

Shane McGraw: Okay our next polling question, which I will pose now, asking: Would you like to discuss the use of proportional scales and ways to combine the scales; or continue with a discussion of how to use RPN numbers. And our options again are: More discussion on scales or continue with how to use RPN numbers. We'll take about 15 seconds to vote for that.

While we're voting, let's get an audience question here. And this was from Tom for Julie and Will asking: Could you please address what I call the "swarm of flies" problem where a large number of minor low RPN defects that accrue over time can have a large aggregate

impact on a project that is much larger than any single defect impact? Often out of control projects must address the minor defects first in order to really see the state of the project. I don't see how the RPN approach would help this situation. How would we address that?

Julie Cohen: So one way to address it-- you'll actually see in the example where we look at grouping things by functionality. Or you could group them by a CSC; or by a CSCI, if that's appropriate-- so if you're in development.

So you can group things in many different ways. And if a group of defects ends up with a very high RPN number as a group, then you may want to say: We need to address this group.

Or if there are defects that are blocking you from continuing development or from continuing tests, they may have a low RPN, depending on how you have your RPN. But if they're actually blocking you moving forward, then maybe you need to reconsider your RPN scales and make that a little bit more important.

But certainly you can look at blocks of defects together instead of single defects one at a time, if that's more appropriate for your situation.

Shane McGraw: Great. Will, anything to add?

Will Hayes: So I think Tom's question is a good one. It shows that perhaps the number of lines of code required to address a defect or the length of the paragraphs required to explain it isn't always going to be directly tied to the risk inherent in allowing that defect to persist.

And so it may be that a group of, quote/unquote, small defects could conspire together to create a bigger problem than one large defect that's on your list. So great question. Thank you.

Shane McGraw: Yes it's great. Please keep the questions coming.

The audience has spoken: 62 percent would like to continue with how to use the RPN numbers.

Will Hayes: Great.

Shane McGraw: And I'll share those results now.

## Resource Available

### Resource Available

For a more complete discussion of the examples presented here, please download the white paper available at the following URL:

[http://resources.sei.cmu.edu/asset\\_files/whitepaper/2013\\_019\\_001\\_70276.pdf](http://resources.sei.cmu.edu/asset_files/whitepaper/2013_019_001_70276.pdf)



\*\*027 Will Hayes: And so pass it back to Julie.

Julie Cohen: All right. So the first slide shows a resource. There is a whitepaper that goes over an example in much more detail than we can cover in this webinar; and you can feel free to look at that if you need more information.

## Sample Data Description

### Sample Data Description

For the sample data we have:

Three users – A, B, and C with 10 DRs each

Five Functions

- Communications
- Navigation
- Planning
- Propulsion
- Security

Assume DRs will be fixed in increments of 3,000 Source Lines Of Code (SLOC) each  
(Note: SLOC is used as a proxy for cost)

Even with this small sample there are hundreds of combinations!



\*\*028 So for the sample I'm going to go through right now, we have three users, A, B and C, with 10 DRs each; and we've looked at five functions that we've called Communications, Navigation, Planning, Propulsion and Security.

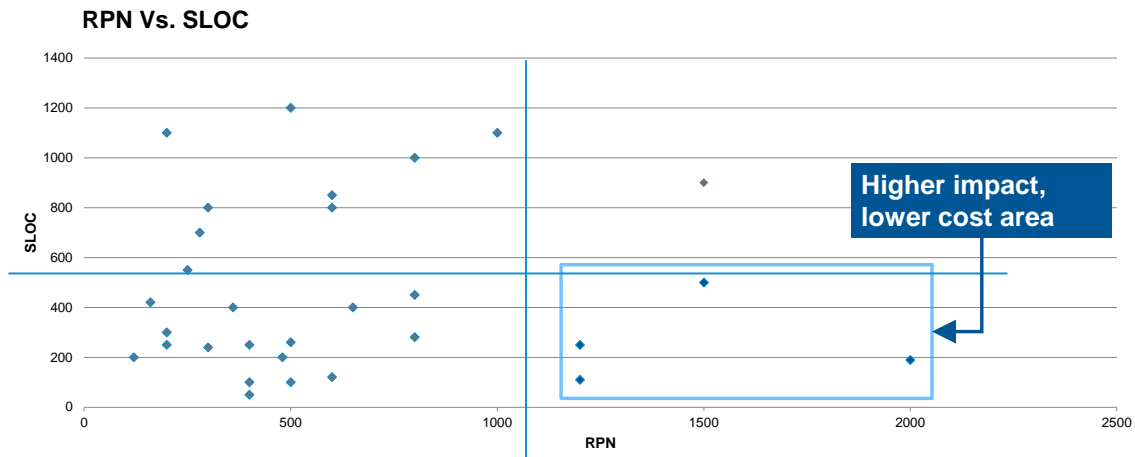
We're going to assume that the DRs will be fixed in increments of 3000 Source Lines of Code or SLOC. And in this case SLOC really is just a proxy for cost.

So if you don't use SLOC, don't think that this doesn't apply to you. Any term of effort measurement that you use, any term of cost effort that you use, that can be substituted for SLOC. In this example that was just an easy way to express this. And even with this small sample there can be hundreds of combinations.



## One way to look at the sample data

### One way to look at the sample data



**Note: In this example, SLOC is being used as a proxy for cost**

\*\*029 So first just a very quick look at the same data plotted out. And you can see in the lower right-hand quadrant, these are the defects with higher impact and lower cost; and that might be a place to start.

But it isn't always that simple. And so we've developed four analysis methods. And don't think that these four are the end-all and the be-all. There could be other ways of doing this analysis, based on your specific situation and what's important in your specific project.

## Four Analysis Methods

### Four Analysis Methods

Method	Brief Description	Pros	Cons
Functionality	Group DRs by system function using RPN and SLOC to select order	<ul style="list-style-type: none"><li>- Easier to test specific functional areas</li><li>- Should see improvements in specific areas addressed</li></ul>	<ul style="list-style-type: none"><li>- May not address top user ranked DRs</li><li>- Some functional areas will not be addressed in every increment</li><li>- Some functional areas may still need to be split due to SLOC constraints</li></ul>
System Risk	List DRs by RPN and draw a line at the 3000 SLOC; Best used for pure maintenance (regression testing only)	<ul style="list-style-type: none"><li>- Addresses system level risk first</li><li>- Fairly easy to use</li></ul>	<ul style="list-style-type: none"><li>- Doesn't specifically address functionality groups</li><li>- Doesn't specifically address user rankings</li></ul>
User rankings	List DRs by user rankings and draw a line at 3000 SLOC;	<ul style="list-style-type: none"><li>- Addresses user rankings</li><li>- Fairly easy to use</li></ul>	<ul style="list-style-type: none"><li>- May fix DRs with lower overall system risk earlier; Doesn't address system value</li><li>- Doesn't specifically address functionality groups</li><li>- Need to address differences between users</li></ul>
Hybrid	Combinations of the methods above	Depends on method	Depends on method

\*\*030 But the four analysis methods we came up with was to look first at Functionality.

Another way to do this is to look first at System Risk.

You could start with User Ratings; if your users are extremely important and you want to make sure that they know that their needs and wishes are being thought about, that could be the way you start.

Or you could use a Hybrid method.

So I'll walk through each one of these in a very simple example. And I know that the numbers have been simplified. But hopefully this will give you a taste for how this methodology can be used and what you might be

able to get from following this process.

## Analysis Method - Functionality

### Analysis Method - Functionality

Look at top level data in a summary format  
(30 DRs from 3 Users)

Functional Area	DRs	Total SLOC	Total RPN
Communications	7	2200	5240
Navigation	7	1700	4210
Planning	8	4700	3620
Security	5	3550	2720
Propulsion	3	1450	2100
		13600	

Highest RPN areas are Communications and Navigation

Assuming 3000 SLOC per build you could close all the DRs in Communications, but you will need to do a partial fix in the Navigation Area



\*\*031 So first off, the Functionality method.

So we just looked at the five areas of functionality, how many DRs there were in each one, the total SLOC for each one and the total RPN.

So the highest RPN areas are Communication and Navigation. And again if we assume a 3000 SLOC per build, you could close all the DRs in the Communications area, in the first build; but you would need to do a partial fix in Navigation.

And in fact, you couldn't even do a full build of any of the other areas as

well. So you might as well go to Navigation next.

## Draft Analysis Method - Functionality

### Draft Analysis Method - Functionality

DR#	User Priority	Area	SLOC	RPN	User Top 3 Priority
120	A2	Communications	250	1200	
114	A3	Communications	1000	800	RPN >1000
116	B5	Communications	200	2000	
121	A6	Communications	100	200	RPN <500
100	A8	Communications	400	160	SLOC > 500
123	B8	Communications	50	400	
115	C9	Communications	200	480	
102	B1	Navigation	500	1500	
106	B2	Navigation	100	600	
107	B3	Navigation	250	200	
108	B6	Navigation	100	250	
122	B7	Navigation	100	500	
101	B9	Navigation	400	360	
117	B10	Navigation	250	800	
			3900		

↑ 3,000 SLOC Cut-Off

**First Build - 4 of 9 Top 3 User Rankings, All Comm DRs, First 2 Navigation DRs ; All 3 Users have at least 1 DR fixed**

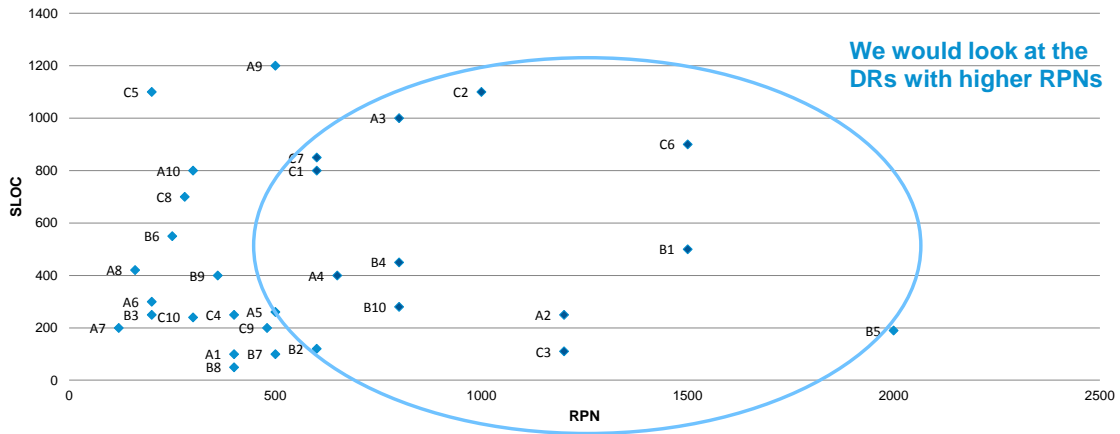
\*\*032 And this looks at the specific DRs; and if you're very quick with your math you'll see that where I've drawn the line is actually 2800 SLOC. And so yes, you could-- I think I've actually done the math wrong. It's not exactly 3000 SLOC and you could certainly add more-- add some smaller DRs to fill out your 3000 SLOC.

But you get the idea. The idea here is you start with all the communication DRs. You then would go to the navigation DRs, until you run out of room; and that would be your first build.

## Second Analysis Method – System Risk

### Second Analysis Method – System Risk

RPN vs. SLOC



\*\*033 All right. The second analysis method is System Risk. And in this one it's pretty simple. You look at the DRs with the higher RPNs first. And this is often used when you're in a maintenance mode. You really want to try and burn down the high-risk DRs first.

And this is a case where you might want to group. As the question asked: What if I have a bunch of small ones that together form a higher risk? If you've got an awful lot of DRs, you may want to group them and pick the groupings with the higher RPN.

## Top 10 RPN DRs

### Top 10 RPN DRs

DR #	User Priority	Area	SLOC	RPN	User Top 3 Priority
116	B5	Communications	200	2000	
102	B1	Navigation	500	1500	RPN >1000
113	C6	Security	900	1500	RPN <500
120	A2	Communications	250	1200	
103	C3	Propulsion	400	1200	SLOC > 500
114	A3	Communications	1000	800	
117	B10	Navigation	250	800	
125	B4	Security	450	800	
118	C2	Planning	1100	800	
106	B2	Navigation	100	600	
			5150		

3,000 SLOC Cut-Off

**First Build - 3 of 9 Top 3 Priority DRs, 4 of 5 functions, burns down ~40% of total system risk**

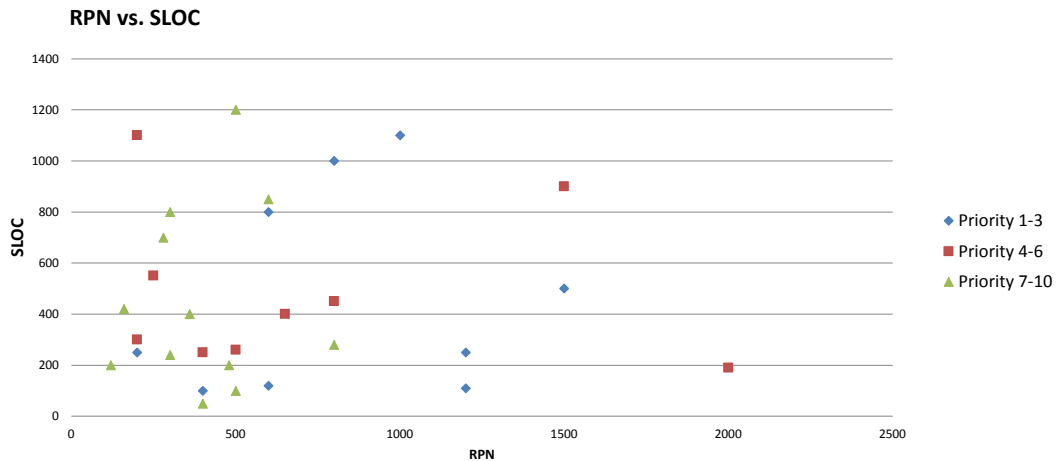


\*\*034 In this case though the top DRs, just looking at RPN first, you could do the first five DRs; and this includes some of the top three priority ones, it includes a lot of different functionality.

It burns down about 40 percent of the system risk. And again, it doesn't add up to exactly 3000 SLOC. You could add some lower priority ones to fill out that 3000 SLOC if you wanted to.

## Third Analysis Method – User Ranking

### Third Analysis Method – User Ranking



Concentrate on the blue diamonds first



\*\*035 Okay the third analysis method starts with the User Ranking. And in this one the blue diamonds show the top three rankings. The red boxes show Priorities 4 through 6. And then finally the green triangles show the lower three priority ratings, 7 through 10.

So we'd want to concentrate on the blue diamonds first. We would want the users to know that we cared about the way that they had ranked and prioritized their defects, and that we were going to start with their highest rated defects.

## Top User Ranked DRs

### Top User Ranked DRs

DR #	User Priority	Area	SLOC	RPN	User Top 3 Priority
124	A1	Planning	100	400	RPN >1000
102	B1	Navigation	500	1500	RPN <500
127	C1	Propulsion	800	600	SLOC > 500
120	A2	Communications	250	1200	
106	B2	Navigation	100	600	
118	C2	Planning	1100	800	
114	A3	Communications	1000	800	
107	B3	Navigation	250	200	
103	C3	Propulsion	400	1200	

3,000 SLOC Cut-Off

First Build - 6 of 9 Top 3 Priority DRs, 4 of 5 functions



\*\*036 So again if we look at the list, we could actually close the top two defects for each user and still stay under our 3000 SLOC cutoff. I believe we're at about 2805. And so we could find 100 SLOC DRs somewhere to help finish off the 3000 SLOC limit if we wanted to.

But again in this case you can see there are a lot of different areas that we're touching. And in some cases you may not want to do this because that touches a lot of code, it creates a lot of testing. And so because of those impacts, we developed the Hybrid method; which I'm going to show you next.



## Hybrid Method – Start with User Ranking

### Hybrid Method – Start with User Ranking

DR #	User Priority	Area	SLOC	RPN	User Top 3 Priority
124	A1	Planning	100	400	
102	B1	Navigation	500	1500	RPN >1000
127	C1	Propulsion	800	600	RPN <500
120	A2	Communications	250	1200	
106	B2	Navigation	100	600	SLOC > 500
118	C2	Planning	1100	800	
114	A3	Communications	1000	800	
107	B3	Navigation	250	200	
103	C3	Propulsion	400	1200	
126	A4	Security	400	100	
125	B4	Security	450	800	
129	C4	Planning	250	400	

Based solely on User Rankings you would fix all the users' top 2 DRs - BUT



\*\*037 So in the Hybrid method you start just like we did before. We start with the User Ranking; in this Hybrid method again. You could start with anything you wanted. In our case we were looking at a system where the user input was very important. And so we started with the user ranking.

And we looked at this and we said: Well okay we've got a lot of different areas and we really want to try and be a bit more efficient. So what could we do?

And when we look at the entire list here you can see that there are two Propulsion DRs in this list.

## Hybrid Method – Then Consider Functionality

### Hybrid Method – Then Consider Functionality

Look at top level data in a summary format  
(30 DRs from 3 Users)

Functional Area	DRs	Total SLOC	Total RPN
Communications	7	2200	5240
Navigation	7	1700	4210
Planning	8	4700	3620
Security	5	3550	2720
Propulsion	3	1450	2100
		13600	

Based solely on User Rankings you would fix all the users' top 2 DRs - BUT

**There are only 3 Propulsion DRs total and 2 were top-3 priority list – the total SLOC for all three is 1450 so you might consider doing those first**



\*\*038 But there were only three Propulsion DRs in total. So if I go back to the original breakout by area, there are only three Propulsion DRs; and since two of those made it into that top list, you may just want to start with the Propulsion DRs. Excuse me. And the SLOC for that is only 1450. So you might consider doing those first.

## Hybrid Method – Determine What Else To Include

### Hybrid Method – Determine What Else To Include

Based solely on User Rankings you would fix all the users top 2 DRs - BUT

There are only 3 Propulsion DRs total and 2 are in this list – the total SLOC for all three is 1450 so you might consider doing those first

You could then add in 6 of the 7 Navigation DRs and still be under the 3000 SLOC budget



\*\*039 You could then add in some of the Navigation DRs and still be under the 3000 SLOC budget; and I'll show you that next.

## Hybrid Method – Final Listing

### Hybrid Method – Final Listing

DR #	User Priority	Area	SLOC	RPN	User Top 3 Priority
127	C1	Propulsion	800	600	
103	C3	Propulsion	400	1200	RPN >1000
112	C10	Propulsion	250	300	RPN <500
102	B1	Navigation	500	1500	
106	B2	Navigation	100	600	SLOC > 500
107	B3	Navigation	250	200	
108	B6	Navigation	100	250	
122	B7	Navigation	100	500	
117	B10	Navigation	250	800	

Based solely on User Rankings you would fix all the users top 2 DRs - BUT

There are only 3 Propulsion DRs total and 2 are in this list – the total SLOC for all three is 1450 so you might consider doing those first

You could then add in 6 Navigation DRs and 1300 SLOC (2750 total SLOC)

Note: You could add additional DRs to get to 3000 SLOC; or you could have considered adding Communication DRs next instead of Navigation



\*\*040 So when we add in the Navigation DRs, we can still do a fair amount of work. But we've limited it to Propulsion and Navigation instead of trying to do all the areas in the original User Ranking List.

You may ask why did I add a Navigation and not Communication; because the Communication DRs had the highest RPN. Well the reason was because Navigation had more DRs on the top ten list. So we wanted to look at those next.

Could you have gone to Communication next? Absolutely, you certainly could have.

Will Hayes: So if I could jump in here. This is a nice illustration how-- the idea is to provide a measurement

strategy that supplements and aids in the decision process; not one that replaces it.

So those local priorities and the business decisions to be made-- if you think of in a new development about to be fielded kind of system, understanding what you're going to have for initial operating capability based on these kinds of functionality tradeoffs may be what you're focused on.

If we focus on this set of DRs it will allow us to stand up the full capability of this component of the system or serve this set of users more completely. And our strategy for deployment of this system then can be tuned according to what sorts of functionality is going to be ready in what sort of schedule.

Julie Cohen: Thank you Will.

## Other uses

### Other uses

Can be used in a development environment:

- Severity can be related to test blockers or number of interfaces to other units, to key requirements or to operational impacts (if known)
- Detection still based on ability to know the defect has occurred
- Time can be based on the effort needed to correct the defect
- RPN can still be compared to functionality and to total cost to fix

Can be used in a maintenance environments

- Rating scale development would be very similar to the example
- Would tend to try to fix the highest RPN defects first, but may still group by functionality or users depending on the situation



\*\*041 So that was a very quick run through the example. And if folks have questions, you can certainly type those in and we'll answer them.

Some other uses. This certainly can be used in a development environment, as Will mentioned at the beginning.

Severity might be related to the DRs that are blocking test; or the number of interfaces to other units.

Detection would still be based on the ability to know that it's occurred.

The time element could actually be based on the effort needed to correct the defect.

And RPN can still be looked at with respect to functionality and total cost to fix. So in a development environment you could still use this. You would need to tailor your rating scales. You would probably need to tailor your weightings. But you could still use the same methodology.

It can also be used very effectively in maintenance environments. The Rating Scale development could be very similar if you're in maintenance. You have operational users; and that might be a big part of what you're looking for as far as the impact.

It could also be though the cost to maintain. If you have certain parts of the code that are more costly to open up and to test and to fix, then that could be another area that you might want to look at, trying to fix defects in those areas; and try and fix those early.

So you can use this in other areas other than just when you're in an operational use.

## Suggestions for DoD Usage

### Suggestions for DoD Usage

#### **Develop a team to put together the structure for RPN use**

- Include the program office, using command, users, contractors, etc. as needed

#### **Need to develop:**

- Definitions for severity which may include different categories
- Definitions for detection which may include different categories
- Methods for dealing with occurrence measures
- Scaling factors
- Computation methods
- Data collection methods
- Process for using RPN values



\*\*042 Some suggestions for DoD usage. Again this is a lot of what Will already talked about.

Very important to develop a team; to put together the structure for RPN usage.

You really want to be sure to have the proper stakeholders on this team.

Certainly if there's a program office that's running the program; if there's a using command.

If there are different users, you want to be sure that you have all those users as stakeholders. Each user may bring a different perspective to the table.



If contractors are involved, you would want them to understand what you're doing and why you're doing it as well.

And as we talk about what you would need to do is you would need to develop those definitions for severity. And you could have different categories for severity. We showed you two. As Will pointed out in the backup, there are several other ways to look at severity that you may want to consider.

You would need methods for dealing with occurrence. Do you just do time to restore? Are there recovery things that people can do? Are there workarounds that people can do that might impact that? And how do you factor those things into an occurrence measure? You would need to think about that. And then you'd also need detection.

You would want to think about your scaling factors. And in relation to the scaling factors you might want to think about your computation methods.

You can use the scaling factors to weight things differently. But that's often harder to explain to your user community. And so you need to- you need to sort of balance how easily you want the user community to understand versus how much math you want to try and put in.

Finally, you have to think about your data collection methods. Most people

already have some sort of a form that they use to collect defect data. This could be added to that form. If you're doing it online it could be made even easier.

And then finally your process for looking at those RPN values and how you're actually going to do the analysis. Who will provide that analytical support and how those things will be used.

**Questions?**

**Questions?**



\*\*043 And I think we've reached the end of the main presentation and we're ready to--

Shane McGraw: Yes so we're going to open it up to questions here. So feel free- if you got questions feel free to type them in now.

We're getting a number of questions on if the slides are available and if an archive will be available of the presentation? A PDF copy of the slides are in the Materials tab now that you can walk away with today.

And this event of course is being archived. Most likely it'll be available at some point tomorrow; and there will be an email that goes out when it is up and ready, and it'll be the same login that you used today.

So let's just jump right in to our questions. From Ed asking: Can you explain how user priority can be high but RPN low?

Will Hayes: Oh so a user may have a particular profile for usage of the system. And that profile may be fairly unique to that user. And so what is very important to them may represent a small amount of the total space in which defects occur. And so because they are a specialized user, their needs may rise higher in their view than the rest of the user base that might be a more diverse- a more shared set of functionality than that user community has in mind.

Is that-- anything to add?

Julie Cohen: Yes also there can be times when there is a defect that takes a lot of time for that particular user to fix when it happens. But maybe it only happens once a quarter.

So if you think about security updates; and every time you do a

security update you have to reboot the system because a security update doesn't install correctly. So for the user that's actually installing that update-- right?-- that might be their highest priority; because every time they do it they have to reboot their system and it's a pain for them to have to do that and it takes them a half a day to do it.

But if you look at the fact that it only happens quarterly-- you know exactly when it happens so it's easily detectable. The system has crashed, you have to reboot it. Then the RPN may be low.

But for them-- it takes them- it takes a person time to actually reboot. The system goes down and they may think that it's important to them to get that fixed. But the total risk to the system may be lower. You know when it happens. You can predict that it's going to happen. You can plan for it; and so total system risk may not be very high.

Shane McGraw: Okay.

Will Hayes: Great.

Shane McGraw: Next question from Siva asking: RPN itself as it suggests is based on the risk and priority. How user priority would be different from RPN method? Let me repeat that one more time. RPN itself, as it suggests, is based on the risk and priority. How can user priority be different from the RPN method?

Julie Cohen: So I'll start with that one Shane. Many times users aren't really thinking about the overall risk to the system. They're either thinking about their own workload; they're thinking about the risk to their particular mission and not necessarily to the entire system. And so they have a very specialized view.

And so just as I talked about before, if it's somebody's security update, to them that is a high user priority. But it may not be as high an overall risk to the system as another user's defect where you can't tell when it's happened and you would actually have to cancel an operational mission if that risk were to occur.

And so oftentimes how it impacts a specific user may cause them to say: This is a really high priority for me. But the overall risk to the system may be much smaller.

Will Hayes: So there may be risk associated with the workflow of the particular user community. And I think Julie's alluding to this as well.

If I have a step in the process that tends to be staffed by junior people or tends to have visibility to external parties, even though it is a minor glitch in the system, because that is a part of my workflow that I feel is a higher risk than other parts of my workflow, a defect that affects that support for my workflow from the system might be a higher importance to me to have resolved than it would be for someone who doesn't have my

workflow in mind as they're looking at this defect.

Shane McGraw: Okay. Next from- actually from Sean asking How would you apply RPN to measuring/monitoring engineering performance metrics such as defect density and defect closure rates? I'll repeat that one more time. How would you apply RPN to measuring or monitoring engineering performance metrics such as defect density and defect closure rates?

Will Hayes: So when I think of engineering performance metrics, I'm thinking of it slightly different than that. And this is an area where our colleague Bruce Grant has offered some examples as well.

You might have a performance requirement for a component of the system and a performance requirement for the system overall. And you might have a defect that causes a time lag or some performance degradation at the local level; but your budget for performance at the system level can still accommodate that.

So even though there's a communication delay on this step of the process, because the overall system performance with respect to communication is ahead of your performance, that may be a tradeoff you're able to make. And so the risk presented by that defect needs to be considered in light of risks associated

with the system of interacting performance measures.

In terms of defect density-- do you want to try to run with that one? Sorry.

Julie Cohen: So I think that really the only connection to defect density would be that as you have defects, if you have RPN you can tell what the overall risk to the system is. And as you burn down the higher RPN risks, you will burn down more of the overall system risk.

And so instead of just showing the defect density, you can now get a little bit more in depth as to-- instead of just trying to burn down overall defects, you could actually burn down higher priority defects first and therefore lower the overall risk to the system faster than just picking up defects one at a time by some other method. This will give you an insight into how much risk that particular defect poses to the system.

Shane McGraw: Okay. We got three more questions in the queue. So if you have questions, feel free to type them in now.

From Bradley asking: Is there an analog for opportunity priority number? Have you considered extending this from risk to opportunities as well?

And I can repeat that one more time: Is there an analog for opportunity

priority number? Have you considered extending this from risk to opportunity- opportunities as well?

Julie Cohen: So we haven't.

Julie Cohen: I wouldn't see why you couldn't. I wouldn't see why you couldn't do an opposite scale to say what the opportunity would give you; what advantage would it provide? So are you looking for a cost savings? Are you looking for a time advantage? Are you-- so you can have severity; instead of severity you would have opportunity.

Detection might be how visible is it to the users that you have advanced this opportunity?

The occurrence might be how often this opportunity could be taken advantage of.

And so yes I think you could. We haven't thought about it. But that's a very interesting idea. Thank you.

Will Hayes: That is a natural linkage. Thank you for making that. Because it really is about a value proposition in the original formulation. It's the value associated with improving quality by removing risk. And you're talking about value associated with deploying capability. Well that's the same thing in some sense.



If you think of concepts like weighted shortest job first, that we see in the Agile world, that's the same kind of thinking that we're doing here.

If you think about that disruption metric of how many times does it occur and how long does it disrupt our operations for, that's very much a similar kind of logic as you see in weighted shortest job first.

Shane McGraw: We got the smartest audience. So. Next one, from Jim asking: is there any consideration given to dependencies between DRs? From Jim asking: Is there any consideration given to dependencies between DRs?

Julie Cohen: So not on priority. Right? Each DR is written up separately. But in truth you're absolutely right; in any system there will be some dependencies between DRs.

And so again depending on your system. If you tend to have a lot of these connections, you may want to group the DRs together and do Risk Priority Numbers for interrelated groups of DRs.

If solving one DR is only going to solve half the problem, and by solving that DR you're really not going to reduce the risk as much as it looks like you're going to reduce it, then certainly you would need to combine those together and do a combined RPN number for several risks that naturally would fit together.

So good question, good point.

Will Hayes: And in thinking along those lines I would caution against trying to alter the formula or the weightings of RPN to account for that other compelling reason why things belong together. That compelling reason has merit on its own. And please, as you saw Julie work through in the examples, please include that in the analysis. Don't try to load everything into RPN.

Shane McGraw: Okay great. Next question from Manny asking: Are there examples of how RPN might be implemented using multiple criteria simultaneously to identify the DRs to work on? For example, user rating, functionality and risk.

Julie Cohen: So we haven't gone that far with the process yet. I think the closest is the hybrid example that we worked here and in the whitepaper. But we would love the opportunity to start to go down that path with an organization.

So if you're interested in trying to explore that with us, please-- there's a contacts slide at the end of the slide deck. Please let us know. We would love to explore that with you.

Shane McGraw: Okay. We're down to two questions in the queue. So feel free to type in. We still have got about 10 minutes left. Otherwise we can always wrap up a little early.

A two part question from Siva asking: In your experience have you come across a situation where the team might have possibly ignored most important risks while focusing so much on computing the RPN? That's the first question.

And number two: How do you decide the minimum number of risks to be addressed, based on RPN? In other words, how do you define the cut off number for RPN?

Julie Cohen: So as far as ignoring the risk while computing RPN, I think the idea is to make RPN simple enough that while you're filling out whatever the normal defect form is, that it doesn't take any longer than five or 10 minutes.

The other thing that we saw with the group that we were working with was that the discussions around the Severity rating could be very contentious.

It went through several rounds and people said: Oh I don't believe that it's really as severe as you thought; and why is it? And somebody else did analysis and came up with a different answer. And then the user felt like they had to defend themselves and--

We feel that this can really help that process. So if you have any type of a process where these defects are adjudicated that having this extra information on why you feel it's as important as it, why you feel the

severity is what it is, if you can't detect it, that automatically helps somebody understand well if you can't tell when it's happened and you have to walk through five more steps until you know that you have this defect back there; and so two more hours have gone by before you even realize that you had the defect five steps ago-- it can really help to explain to somebody how you came to that severity rating.

So we believe that this can actually help in the overall adjudication of DRs. But absolutely, if you make it so difficult that to do the RPN takes away from actually writing up the DR or fixing it, then that's not the way to go.

Will Hayes: And I think this is another opportunity to repeat something that Bob Ferguson has told us many times. It's about the conversation. It's about structuring that conversation so people can have objective criteria and know what it is that's leading to the decision outcome that's happening.

It is not about getting to a precisely accurate number to four significant decimals; as much as I might enjoy that as a statistician. The idea here is to structure the conversation to avoid the condition the questioner asked.

Julie Cohen: And what was the second part of the question?

Shane McGraw: Oh you know what? Let me go back to it. Okay part two was: How do you decide the minimum number of risks to be addressed, based on RPN? In other words, how do you define the cutoff number for RPN?

Julie Cohen: Right. So that generally is formed by either budget or schedule, in what we found.

Generally people want an increment delivered by a specific date; and so the amount of work that you could get done in that timeframe would be how you would base your cutoff. Or how much resources you can put in to the next build would determine your cutoff.

And so that's really not determined by the RPN number. Generally it's determined by a project constraint.

Shane McGraw: Okay great. Last question in the queue. So if you have any last minute questions, feel free to type them in. Otherwise we're going to wrap up on this one.

From Hemont asking: Where do you see being used in the SDLC and operational maintenance processes? Where do you see this being used in SDLC and operational maintenance processes? Will it be mostly for deciding which defects to fix in a system?

Julie Cohen: that's a very natural place to use it is deciding which defects to fix. But as

I just discussed, I think it can also help in the conversation as to why defects are important, why a user feels like this defect is a top defect.

So it can be used in the adjudication process; and then also it will be very useful in deciding which defects to fix first.

Again, it may not be as straight; you know, number them by RPN and go down one at a time, starting with the top one.

You may want to group them into functionalities to get some testing advantage or some development advantage; you're not opening the same code over and over again.

But it certainly is a good guidepost to start out with trying to reduce- trying to take the highest RPN defects first, thus reducing the greatest risk to your system.

Shane McGraw: Will?

Will Hayes: No that's great.

Shane McGraw: Okay Will and Julie you're off the hot seat. Thank you for your excellent presentation.

Again folks, thanks everyone for attending today.

## Contact Information

### Contact Information

#### Presenter / Point of Contact

Julie Cohen  
Client Technical Solutions Division  
Telephone: +1 412.268.3950  
Email: [jcohen@sei.cmu.edu](mailto:jcohen@sei.cmu.edu)

William Hayes  
Client Technical Solutions Divisions  
Telephone: + 1 412.268.6398  
Email: [wh@sei.cmu.edu](mailto:wh@sei.cmu.edu)

#### Customer Relations

Email: [info@sei.cmu.edu](mailto:info@sei.cmu.edu)  
Telephone: +1 412.268.5800  
**SEI Phone:** +1 412.268.5800  
**SEI Fax:** +1 412.268.6257



\*\*044 We do request that you fill out the survey upon exiting as your feedback is always greatly appreciated.

For any of you- those of you having some bandwidth issues today, this will be archived. We'll get an email out tomorrow or the next day when that is available; and you can catch up then.

But again thank you for attending. Have a great day everyone.