# **HOW TO SLICE A FEATURE**

## **PREPARE THE INPUT FEATURE**

#### WARNING - Don't slice Features unless something is needed in the next PI

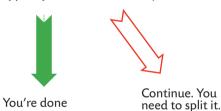
Does the Feature satisfy INVEST\* (Except, perhaps sized appropriately)





Reformulate the Feature to clearly communicate the benefit or slice off one or more smaller Features which do satisfy INVEST and carry clear benefit.

Is the Feature size less than 1/10th of your program velocity?\* (or typically medium or smaller).



\* INVEST Features should be: Independent . Negotiable Valuable Estimable Sized Appropriately Testable.

\*\* Velocity varies between programs but as a rule of thumb a program should be tackling at least the 'Top 10' Features hence the no greater than 1/10th of the program velocity guideline.

**EVALUATE** 

#### **FIND A STORY GROUP**

Could you find the set of most valuable Stories and develop and release them as their own Feature?

> Do many Features rely on the same underlying system behaviors (often making the first of them selected to be very large and complex)?

(KEEP IT SIMPLE)

Could you slice the Feature to do that

simple core first and build on it later

with further Features?

Does the Feature have a simple core that provides most of the

benefit and / or learning? This is

often the happy path with some

basic error handling.

Does 80% of the

value come from

20% of the Stories?

#### **BREAK OUT COMMON ENABLERS**

Could you break out the common enablers into their own 'Architectural' Features? Delivering the enablers can significantly de-risk, simplify and reduce the estimates for the other, related Features.

#### **DEFER OPTIONAL BEHAVIORS**

Could you make the optional behaviors separate Features to be done once the core functionality / most popular option is in



of optional behavior (for example different ways to achieve the same goal)?

> **APPLY THE SLICING**

**PATTERNS** 

Does the Feature include Special Variations?

Is there an obvious Feature to start

#### **ISOLATE SPECIAL** VARIATIONS

Could you focus on the most popular / highest volumé cases first and treat the more specialized corner cases as separate Features? You may find that their value / cost ratio is very small and they are never needed.

incrementally to different areas of the business?

Does the Feature lend itself

to being released

Does the Feature need to be delivered over different channels, different mediums or different routes to the customer?

#### SEPARATE DIFFERENT **CHANNELS**

Could you deliver it one technology / one channel at a time? Could you start with the channel or most value to the business and add the other channels over time?

Does the Feature involve different user groups with different goals?



ADDRESS DIFFERENT **USER GROUPS INDIVIDUALLY** 

Could you give each User Group their own Feature? This can help you to better understand the benefits to each group. See also Break Out Common Enablers.

Does the Feature involve lots of data from many sources?

Could you deliver benefit with a CONSIDER sub-set of the data? Could the data be consumed incrementally or sourced from

**SEPARATE BUSINESS** 

**VARIATIONS** 

Could you deliver it one business at a time? Could you start with the simplest

business variant to generate quick wins and fast feedback?

**INCREMENTALLY** existing secondary source. **SOURCING DATA** 

### **BREAK OUT A SPIKE**

Are you still baffled about how to slice the Feature?

you back?

Can you define the 1..3 questions most holding





Do the minimum to answer the questions and then start again at the top of this process.





Do each of the new Features readily fit into a PI (< 1/10 of the program velocity)?





with that gets you early benefit, learning, risk reduction etc? Try another pattern to see if you can get this.

You're done, though you could try another pattern to see if it gets better results.



#### **WARNING DON'T:**

- Defer non-functional requirements
- Slice too early Over slice
- Slice by component
- Forget the Feature testing





