Measuring Agile Development

Vijay G Arcot
British Telecom
602-B, RMZ Infinity,
Old Madras Road,
Bangalore, INDIA

vijay.arcot@bt.com
+91 98459 11668

Objectives

- Understand how measuring productivity in Agile development is different in terms of accounting the Function Points transactions and cost involved in the process
- Additional complexities and transaction types to collect relevant productivity metrics
- Understand the challenges involved in collection of costs of various phases and sprints of Agile

Agenda

- Productivity Measurement
- Overview of Agile Methodology
- Function Points
  - Multiple Sprints
  - Enhanced Complexity for higher FTR/DET
  - Reuse, Configuration and Test Points
  - Adjustments
- Cost
  - What is included
  - Challenges
- Conclusion
- Questions

Productivity Measurement

$$\text{Productivity} = \frac{\text{Cost (\$,£,€)}}{\text{Function Points}}$$

Agile development being different in the way requirements (stories) are organized and phases of development cycle, there is a need for a different approach to record FPs and costs.

Introduction to Agile

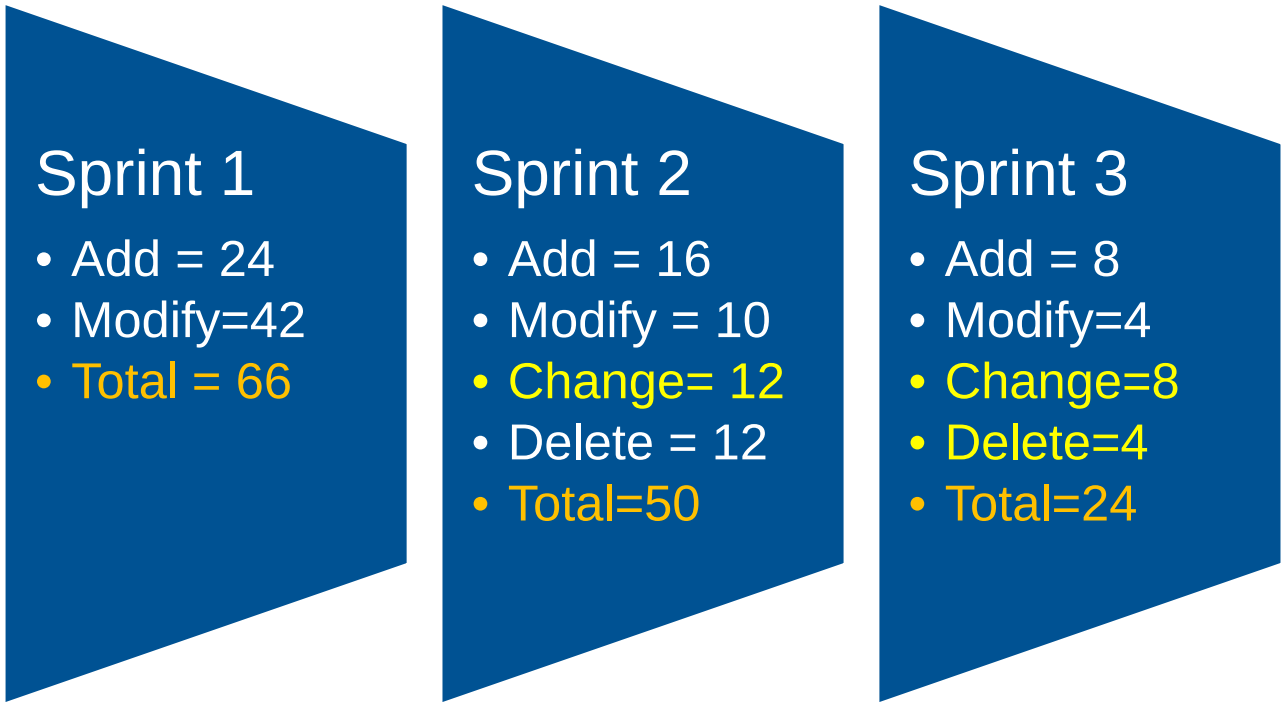
Agile software development is a model where in more emphasis is given on process and requirement adaptability throughout the life-cycle of the project. Agile methods break tasks into small iterations with minimal planning. Iterations are short time frames that typically last from one to four weeks. Each iteration involves a team working through a full software development cycle including planning, requirements analysis, design, coding, unit testing, and acceptance testing when a working product is demonstrated to stakeholders. An iteration may not add enough functionality to warrant a market release, but the goal is to have an available release (with minimal bugs) at the end of each iteration. Multiple iterations may be required to release a product or new features.

Source: Wikipedia

Multiple Sprints

- Requirements induced in each sprint
  - Count the FPs added
- Features modified in each sprint
  - Count the FPs associated
  - Count a transaction even if it was counted in previous sprint
- Deleted features in each sprint
  - Count all deleted transactions
  - If a transaction induced in one of previous sprint was deleted, that will get counted as well

Multiple Sprints… Example



Sprint-wise Counting = **140 FPs**

Traditional way of Counting (End to End) = 116 FPs

Enhanced Complexities

- Increase in usage of COTS products – Buy v/s Build
- Workflow and Reporting applications contain very high number for DETs and FTRs
- Most of the cases would only produce “High” Complexity transactions without being able to differentiate the sizes
- Enhanced complexity matrix to address “Higher” Complexity Transaction
- Split of transaction to further level not possible without sacrificing EP rules.
- Caution! Rule out possibility of not breaking transaction to lowest level

Enhanced Complexities

Uniform additional weights for transactions:

	1-4 DET	5-15 DET	16+ DET	**50-100 DET**	**101-200 DET**	**201+ DET**
0-1 FTR	L	L	A	**VH**	**VH**	**UH**
2-3 FTR	L	A	H	**VH**	**UH**	**SH**
4+ FTR	A	H	H	**VH**	**UH**	**SH**

Key	Complexity	FP
L	Low	3
A	Average	4
H	High	6
VH	**Very High**	**9**
UH	**Ultra High**	**12**
SH	**Super High**	**15**

Enhanced Complexities… Example

Story

"As a...ATB product manager I want to ...launch a new UAMP (unlimited anytime and mobile) call plan add-on with inclusive mobile minutes for bundles So that ...voice churn is reduced "

FP

Txn's	No.	FP (Now)	FP (New)
L	4	12	12
A	2	10	10
H	4	24	24
VH	2	12	18
UH	3	18	36
SH	1	6	15
Total		**82**	**115**

Productivity

Cost = $150k

Productivity:

Now - 150/82 = $1.8k per FP

*New - 150/115 = **$1.3k** per FP*

Tracking Reuse

- With increased focus on minimizing code changes during development an additional metric to record re-use is introduced
- Percentage of code reuse is recorded against each transaction – as a ratio existing lines of code to added/modified lines of code
- The average percentage of all transactions across a release is taken as a reuse percentage
- Acts as an indicator to identify correlation between increase in reuse and decrease in $/FP

Configuration

- Any change that is perceived by the user and is countable by Function Point rules as a valid transaction is counted
- Change may be facilitated through complete configuration. Example – Changing row values in XML file.
- All configuration is counted as 100% reuse
- Only transaction functions are included in the scope, no data functions

Test Points

- Applicable to an application that tests functionality already in place to support change in output behavior without any changes to code/configuration
- Requirements for testing the specific features should be in place
- Not applicable for Regression/System/Performance testing
- Only transaction functions are included in the scope, no data functions
- Tracked separately from normal Function points

Adjustments

- For Reuse, Function Points are adjusted by a factor varying between 0.75 to 1.0

$$\text{Reuse FP} = \text{AdjFP} * 1 - (0.25 * R\%)$$

Where $R = \sum r$

r = percentage reuse in a transaction

- Test Points are converted to equivalent Function points by multiplying the test points by flat factor of 0.4

Source of Values – In house metrics data

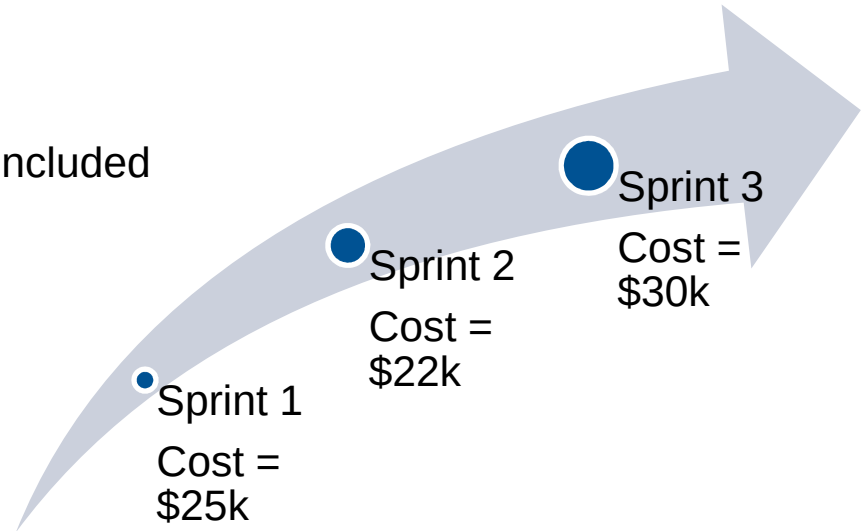
Cost – What is Included

- In a traditional development model not all phases of development are included in the cost for productivity
- Only Phases whose costs can be apportioned to the participating applications are included
- In Agile, where the demarcation between various phases are blurred due to overlapping activities between sprints, all costs are included.
- The additional metrics introduced viz. Reuse, Test Points contribute to like to like comparison with costs included

Cost – What is Included

Traditional Model: Highlighted phases included in cost

Agile Model: All Sprints included



Cost - Challenges

- Scope changes – In between sprints, requirements dropped due to scope changes should be treated as sunken costs.
- If a requirement is past testing phase, the cost and FPs need to be included – accurate tracking system is expected to be in place
- To establish a sprint-level metric, the overheads such as Architect costs, project management costs, etc need to apportioned to sprints. This may be a challenging exercise

Conclusion

- Accuracy of measuring productivity in Agile is enhanced by modifications to Function Point counting and Accounting Costs
- Consistency of the methods described need to be maintained across releases over a longer period of time to be able to produce a stable baseline that will enable meaningful trend lines
- The reuse and test points can be used in non-Agile development models also, with suitable modifications to adjustment factors.

References and Further Investigations

- www.wikipedia.org
- BT Internal Guidelines
- IFPUG Bulletin Board
- Software Engineering A Practitioners Approach – Roger Pressman

Questions?

Disclaimer: All BT confidential information has been masked suitably so as to not reveal the real figures