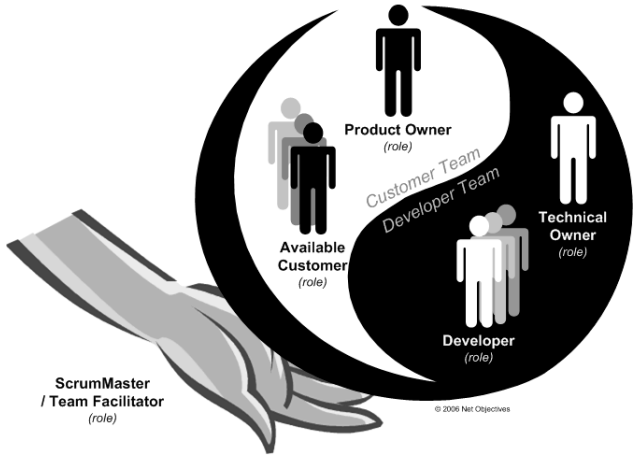
# Are Parametric Techniques Relevant for Agile Development Projects?

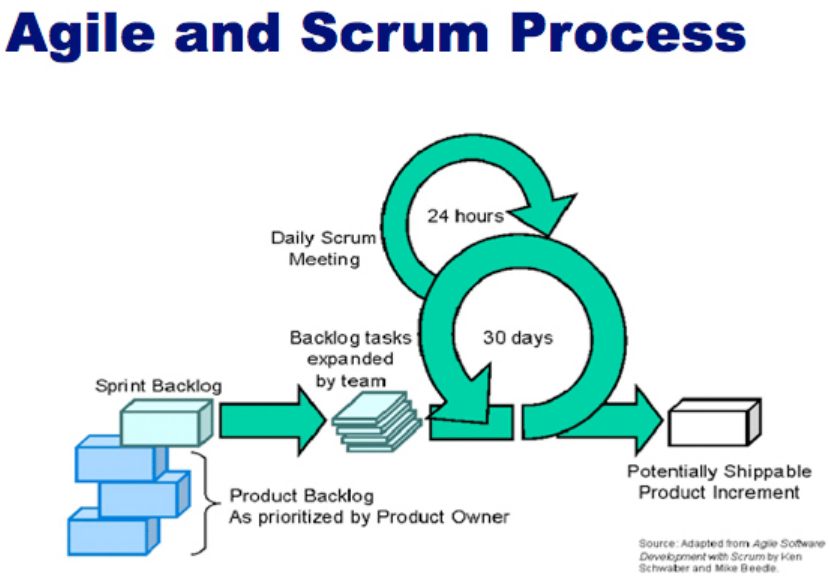



Arlene Minkiewicz, Chief Scientist
PRICE Systems, LLC
arlene.minkiewicz@pricesystems.com

TruePlanning®
by PRICE® Systems

# Agenda

- Introduction
- Agile Software Development
- Agile Estimation
  - Agile Size Estimation
  - Agile Cost Drivers
- Conclusions

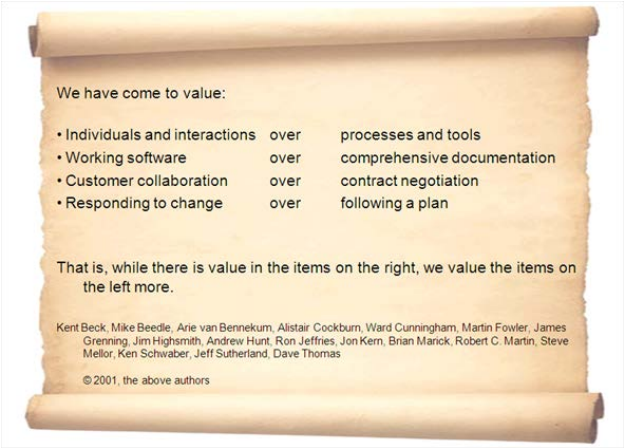


# Introduction

- Back in the day, complexity of applications was overshadowed by the logistics of implementation
  - Punched cards
  - Batch processing
- Technology improved, software solves increasingly complex problems
  - Logistics demurred to the solution as the bottleneck
- The so called ‘software crises’ (mid 60’s to mid 80’s) resulted in many ‘silver bullet’ type solutions
  - Structured programming, formal methods, CMMI, programming standards
  - They helped but is the software industry out of crisis mode?
- Lots of smart software development professionals began looking for more lightweight methods that address complexity in achievable chunks

# Agile Manifesto – February 2001

- We are discovering better ways of developing software by doing it and helping others to do it. Through this we have come to value
  - Individuals and interactions over processes and tools
  - Working software over comprehensive documentation
  - Customer collaboration over contract negotiation
  - Responding to change over following a plan



We have come to value:

| | | |
|---|---|---|
| • Individuals and interactions | over | processes and tools |
| • Working software | over | comprehensive documentation |
| • Customer collaboration | over | contract negotiation |
| • Responding to change | over | following a plan |

That is, while there is value in the items on the right, we value the items on the left more.

Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, Dave Thomas



- All agile projects adhere to this manifesto
- All agile projects share a common set of principles
- Each agile project uses a set of agile practices to implement these principles.
- Successful estimation for an agile project is like software estimation for any project – you need to understand the project properties and the practices employed

# Traditional Software Development

- Traditional software development
  - Requirements are analyzed
  - Architecture and Design are created
  - Requirements are implemented, tested and delivered
  - Months (or longer) occur before usable software for customer to evaluate

# Agile Software Development

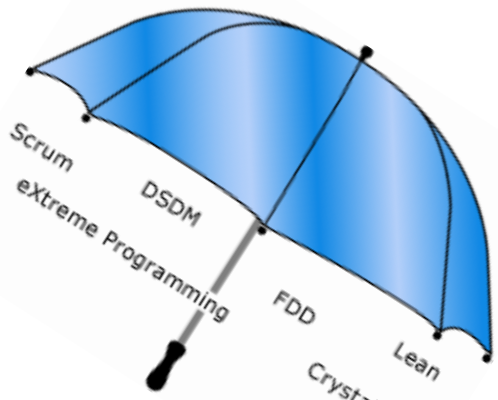

- Agile Software Development
  - Usable chunks of software are developed in short periods of time (sprints, iterations, etc.)
  - Requirements are translated into User Stories and become the project backlog
  - User stories deliver business value and are small enough to be completed in an iteration
  - Customer works with the team and reviews software regularly
  - Each iteration focuses on the User Stories that are currently highest priority of the customer
  - Priorities may shift from iteration to iteration
  - Agile teams expect and embrace change

# Agile Principles

- Customer satisfaction through frequent deliveries and customer involvement
- Focus on business need and business value
- Time boxed development
- Constant collaboration and communication
- Sustainable place
- Adaptation to change
- Self Organizing Teams
- Collective ownership and responsibilities
- Commitment to measurement

# Common Agile Practices

- Pair programming
- Continuous integration with automated testing
- Test Driven Development
- Daily Stand-up meetings
- Co-located teams
- Code refactoring
- Small releases
- Customer on team
- Simple Design



# Agile Estimation

- Frequently Asked Questions
  - How do I estimate size for an agile project when the team is working with Story Points?
  - What other cost driver changes are indicated for agile development?

# Agile Size Estimation

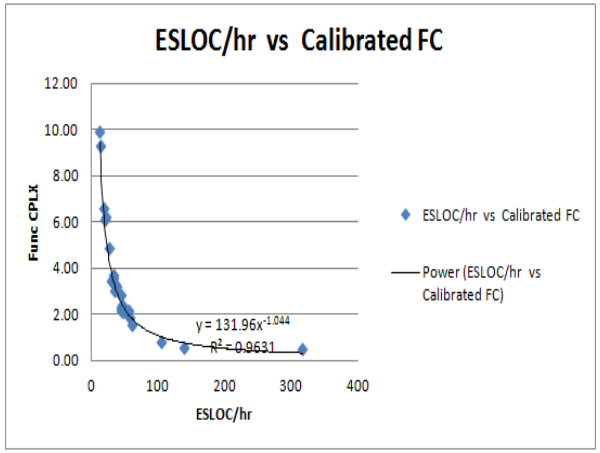
- Agile teams do lots of their own estimating
  - At the start of each iteration, the team breaks the current User Stories into tasks and estimates effort for each task
  - For each Story the team determines the number of Story Points using an arbitrary system agreed upon within the team
    - Planning Poker, Fibonacci numbers (0,1,2,3,5,8,13….)
    - Story Points are relative ( a story of 2 points will take twice the time as one with 1)
  - Estimation at the release level is more challenging since less detail is available
    - Some teams use t-shirt sizes to communicate scope to non-technical folks

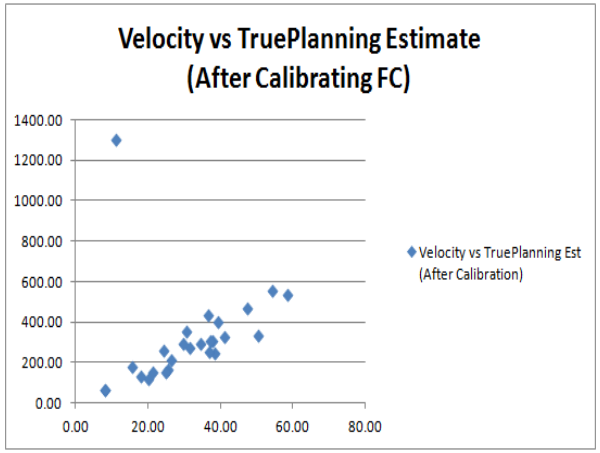
# Agile Size Estimation

- Story Points are Notional and don't trend with effort nicely
- In the context of a parametric model – Story Points really combine two typical drivers
  - Software Size
  - Complexity of the functionality

# Agile Size Estimation

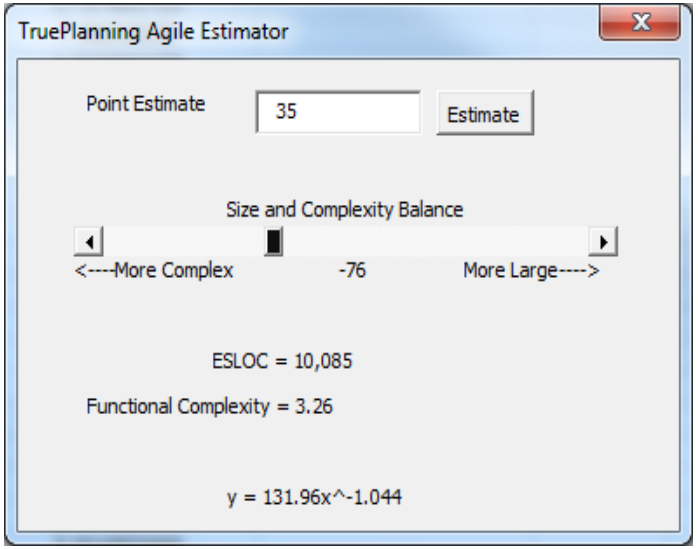
- Study of PRICE's agile development data found no correlation between story points and software size (SLOC) or effort
- Did find a significant relationship between Software Size and Complexity pairs and effort/hr.
- Calibration of Functional Complexity to agile data resulted in interesting size/complexity relationship



# Agile Size Estimation

- From PRICE Data we developed a methodology to estimate size complexity pairs from Story points and the teams assessment of complexity relative to the data set.
- This methodology transcends tool but needs to be informed by data from the agile team delivering story point estimates
- This methodology appeals to agile thinking folks but supports parametric estimates with relevant cost drivers



# Agile Cost Drivers

- The fact that a project is agile is not a cost driver.
- There are common agile practices that, if employed by an agile team should have cost/effort impacts
- It is important that the estimation team determine which agile practices apply
- For teams early in agile adoption, there are likely to be productivity issues as new practices are adopted

# Agile Cost Drivers

- Agile teams tend to be highly skilled
  - Hard to be a slacker in an agile environment
  - Working closely with high skilled team members, new members of the team are quickly brought up to speed
  - Input parameters indicating team experience would be affected
- Agile teams tend to have tool sets that are more sophisticated than the average team
  - Overhead associated with being agile – backlog maintenance, agile metrics collection, maintain user stories and tasks, etc.
  - Input parameters around tools or automation would be affected


# Agile Cost Drivers

- Co-location of teams should impact productivity positively
  - Culture of interruption – but in a good way
  - Questions are answered in real time
  - Red tests get fixed right away
  - Co-location tends to increase the cohesion of the team
  - Co-located stakeholders and Subject Matter Experts (SMEs) make the team function more like an IPT
  - Well run daily stand up meetings also impact productivity
  - Input parameters indicating distribution of team locations, communication, IPT or team cohesion would be affected

# Agile Cost Drivers

- Continuous integration with automated testing should increase the productivity for test and integration
  - Code is checked in frequently and builds are run and tested either on code change or in regular increments (hours, not days)
  - Red tests raise red flags and someone on the team steps up
  - Since little code was changed problem is easy to isolate
  - Since the change was recent developer more likely to remember context
  - Fix should occur quickly
  - Input parameters focused on integration testing complexity, issues would be affected.

# Conclusion

- Agile development is still software development and common estimating principles apply to it
- Estimating from Story Points requires an understanding of the multiple dimensions of a Story Point and how they line up with estimation parameters
- Agile is a paradigm, not a methodology

- Agile methodologies recommend many practices that should be considered as potential cost drivers
- Estimation requires an understanding of the software being developed and the environment it's being developed in.

# Questions


Arlene.minkiewicz@pricesystems.com


TruePlanning®
by PRICE® Systems