Achieving Agility at Scale
Improving Software Economics

Walker Royce
Vice President, Chief Software Economist
### Report Documentation Page

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

| 1. REPORT DATE | APR 2010 |
| 2. REPORT TYPE | |
| 3. DATES COVERED | 00-00-2010 to 00-00-2010 |

| 4. TITLE AND SUBTITLE | Achieving Agility at Scale Improving Software Economics |

| 5a. CONTRACT NUMBER | |
| 5b. GRANT NUMBER | |
| 5c. PROGRAM ELEMENT NUMBER | |
| 5d. PROJECT NUMBER | |
| 5e. TASK NUMBER | |
| 5f. WORK UNIT NUMBER | |

| 6. AUTHOR(S) | IBM Corporation, IBM Software Group, 1 New Orchard Road, Armonk, NY, 10504-1722 |

| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) | |
| 8. PERFORMING ORGANIZATION REPORT NUMBER | |

| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) | |
| 10. SPONSOR/MONITOR’S ACRONYM(S) | |
| 11. SPONSOR/MONITOR’S REPORT NUMBER(S) | |

| 12. DISTRIBUTION/AVAILABILITY STATEMENT | Approved for public release; distribution unlimited |

| 13. SUPPLEMENTARY NOTES | Presented at the 22nd Systems and Software Technology Conference (SSTC), 26-29 April 2010, Salt Lake City, UT. Sponsored in part by the USAF. U.S. Government or Federal Rights License |

| 14. ABSTRACT | |

| 15. SUBJECT TERMS | |

| 16. SECURITY CLASSIFICATION OF: | a. REPORT unclassified |
| b. ABSTRACT unclassified |
| c. THIS PAGE unclassified |

| 17. LIMITATION OF ABSTRACT | Same as Report (SAR) |

| 18. NUMBER OF PAGES | 21 |
| 19a. NAME OF RESPONSIBLE PERSON | |

Standard Form 298 (Rev. 8-98)  
Prescribed by ANSI Std Z39-18
Software development obsolesced by software delivery

Software Development

- Distinct development phase
- Distinct handoff to maintenance
- Requirements-design-code-test sequence
- Phase and role specific tools
- Collocated teams
- Standard engineering governance
- Engineering practitioner led

Software Delivery

- Continuously evolving systems
- No distinct boundary between development and maintenance
- Sequence of released capabilities with ever increasing value
- Common platform of integrated process / tooling
- Distributed, web based collaboration
- Economic governance tailored to risk / reward profiles
- Business value and outcome led
Software cost models

From George Stark, Paul Oman, “A comparison of parametric Software Estimation Models using real project data”, in press
Improving software economics

- Empirical software cost estimation models for:
  - Enterprise modernization, software maintenance
  - New developments, new releases, early prototypes
  - Packaged applications, systems engineering

Time or Cost
To Build  = (Complexity) \( \times \) (Process) \( \times \) (Team) \( \times \) (Tools)

**Complexity**
- Volume of human generated stuff
  - KSLOC, FPs, UCs
- Quality/performance
- Scope

**Process**
- Methods
- Maturity
- Agility
- Precedence

**Team**
- Skills/Experience
- Collaboration
- Motivation

**Tools**
- Automation
- Process enactment
Schedule risk: Imagine you have 12 months to deliver a business critical system

- Your estimators tell you it will be done in 11 months

- What do you do with the information?
  - Rest easy, believing there is no risk?
Maybe you realize that program parameters (cost, schedule, effort, quality, …) are random variables

- Area under curve describes probability of measurement falling in range

Likelihood of actual value falling in range is area under curve
Imagine you have 12 months to deliver a business critical systems

- So you ask for the distribution and discover there is some uncertainty
Imagine you have 12 months to deliver a business critical systems

- In fact there is less than 50% chance of making the date
Then what?

- Move out the date to improve likelihood of shipping?
Then what?

- Or move in the estimate by sacrificing quality or content?
Managing variances in scope, solution, plans: The real key to improving software economics

- **Sources of uncertainty and variance**
  - Lack of knowledge
  - Lack of confidence
  - Lack of agreement

- **Reduction of variance reflects**
  - Increased predictability of outcome
  - Increased knowledge about
    - Client needs
    - Technology capability
    - Team capability
  - Good decisions
Then what?

- Determine the source of the variance

- Over the project lifecycle, reduce the variance to improve likelihood of shipping
Then what?

- Over the lifecycle, reduce the variance further to improve likelihood of shipping
Practices included as part of Rational Method Composer

- **Governance & Compliance**
  - Risk-Value Lifecycle
  - Practice Authoring & Tailoring

- **Agile Core**
  - Iterative Development
  - Two-Level Project Planning
  - Whole Team
  - Continuous Integration
  - Test Driven Development

- **Change & Release Management**
  - Team Change Management
  - Formal Change Management

- **Requirements Management**
  - Shared Vision
  - Business Process Sketching
  - User-Case Driven Development
  - Requirements Management

- **Architecture Management**
  - Evolutionary Architecture
  - Evolutionary Design
  - Component Based Software Architecture
  - Design Driven Implementation

- **Quality Management**
  - Concurrent Testing
  - Test Management
  - Independent Testing
  - Application Vulnerability Assessment
  - Performance Testing
Critical culture shifts in improving software economics

**Conventional Governance**

**Activity-based management**
- Mature processes, PMI/PMBOK
- Plan in detail, then track variances

**Adversarial relationships**
- Paper exchange, speculation

**Requirements first**
- Assumes certainty in desired product
- Avoid change

**Early false precision**
- “More detail = higher quality”

**Apply too much or too little process**
- Process is primary, blind adherence

---

**Agile Economic Governance**

**Results-based management**
- More art than engineering
- Plan/steer/plan/steer…

**Honest collaborative communication**
- Progressions/digressions, facts

**Architecture (risk mitigation) first**
- Admits uncertainties
- Manage change

**Evolving artifacts**
- Scope (Problem specs)
- Design (Solution specs)
- Constraints (Planning specs)

**Right-size the process**
- Desired results drive process
- Manage variances
Measure and steer

- At onset of program
  - **Report:** Establish estimates/variances of effort, cost, establish initial plan
  - **Collaborate:** Set initial scope and expectations with stakeholders
  - **Automate:** Establish a collaborative development environment

- At each iteration, improve estimates and report
  - **Report:** Values and variances of progress achieved, quality achieved, resources expended
  - **Collaborate:** With stakeholders to refine scope and plans
  - **Automate:** Manage changes to plans, baselines, test-beds
Agile Governance = Managing Uncertainty = Managing Variance

- A completion date is not a point in time, it is a probability distribution

- Scope is not a requirements document, it is a continuous negotiation

- A plan is not a prescription, it is an evolving, moving target

Plans/Resource estimates
Scope
Product features/quality

Uncertainty in Stakeholder Satisfaction Space

Initial State
Actual path and precision of Scope/Plan

Initial Plan
Four patterns of success in achieving Agility at Scale

1. Scope management ➔ Asset based development
   Solutions evolve from requirements AND requirements evolve from available assets
   As opposed to getting all the requirements right up front

2. Process management ➔ Rightsize the process
   Process and instrumentation rigor evolves from light to heavy
   As opposed to the entire project’s lifecycle process should be light or heavy depending on the character of the project

3. Progress management ➔ Honest assessments
   Healthy projects display a sequence of progressions and digressions
   As opposed to progressing to 100% earned value with monotonically increasing progress against a static plan

4. Quality management ➔ Incremental demonstrable results
   Testing needs to be a 1st class, full lifecycle activity
   As opposed to a subordinate, later lifecycle activity
Effective software delivery enabled by agility and measurement

Measures of increasing value

- More creative time, less overhead time
- Painless governance
- More automation support
- Fewer meetings
- Less scrap/rework
- Earlier defect detection
- Honest metrics
- More reusable assets, services, skills, practices and measures
- More predictable outcomes
- Higher ROI
- Optimized investments and supply chains
- Software development as an first class business process
- Business optimization
Invest across the spectrum of improvement to manage risks and optimize business outcomes

**Business Value**

- **Improve Automation**
  - Cost to Implement: <5%
    - Very predictable
  - Productivity: 5-25%
    - Timeframe = Weeks

- **Improve Collaboration**
  - Cost to Implement: 5%-10%
    - Predictable
  - Productivity: 15-35%
    - Timeframe = Months

- **Improve Process**
  - Cost to Implement: 10%-35%
    - Some culture change
  - Productivity: 25-100%
    - Timeframe = Quarters

- **Increase Flexibility & Investment Value**
  - Cost to Implement: 25%-50%
    - Much culture change
  - Productivity: 2x – 10x
    - Timeframe = Years

**Efficiency**

- **Individual**
- **Team**
- **Organization**
- **Business**

**Control**

Implementation costs are per person per year.
Some final thoughts

Agile Software delivery is a discipline of software economics

Strong measurement practices are necessary to manage uncertainty and achieving agility at scale

Economic governance requires a platform that is architected for automation, collaboration and reporting