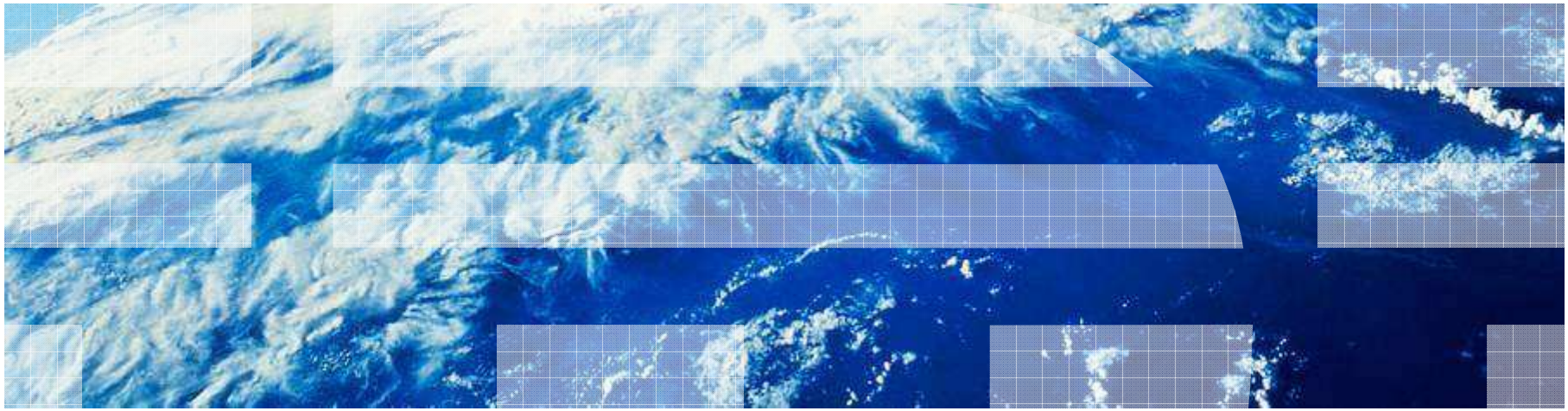
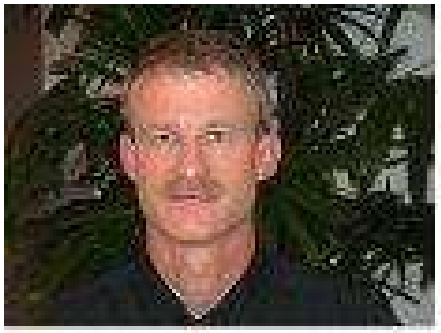
The Economics of Systems and Software Delivery: Reducing Risk and Improving Governance

Brian T. Nolan, Ph.D., *Go to Market Manager, Aerospace & Defense*
Rational Systems Marketing

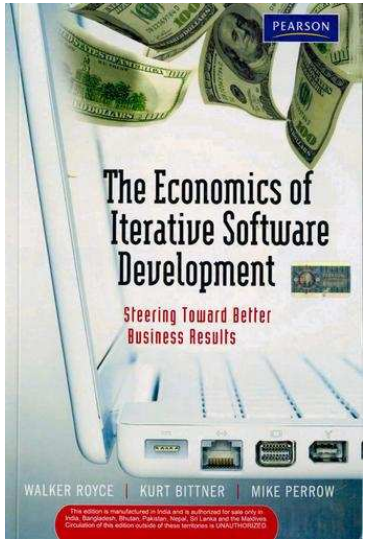


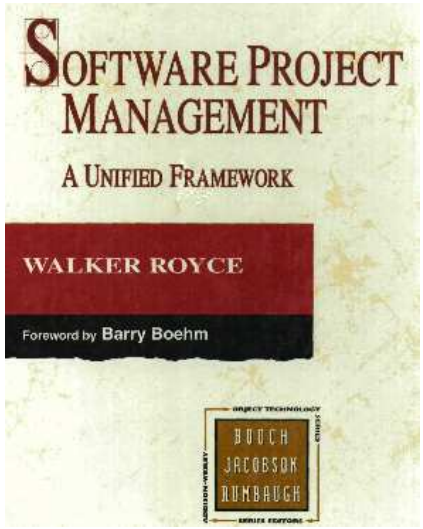
2011

I am not Walker Royce

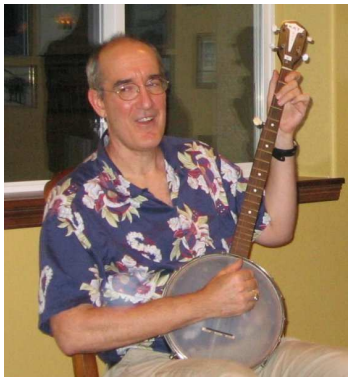


- Google his presentation at Innovate 2011: “Walker Royce Innovate 2011 keynote”
- Several excellent books and articles on this subject

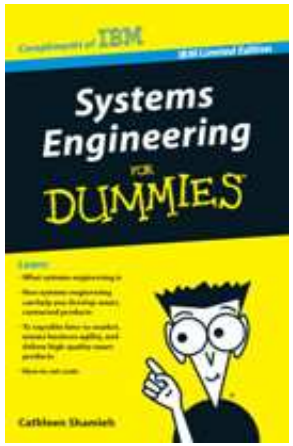


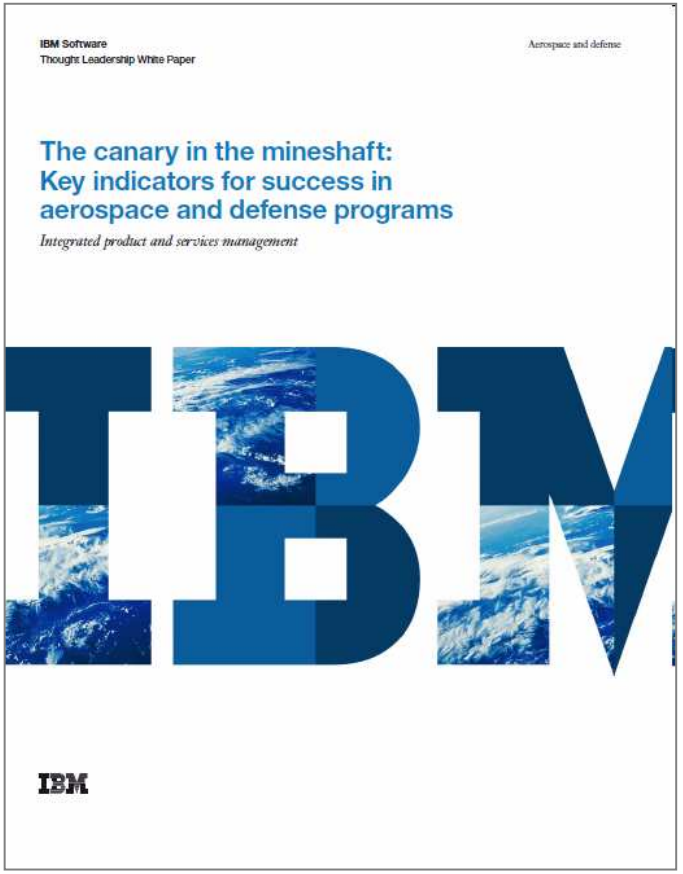


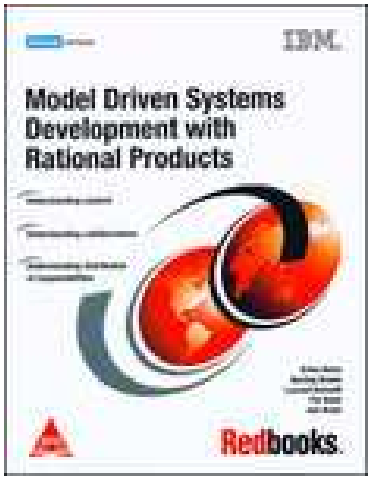
Brian Nolan, Ph.D.



- But I have been working with Walker, Murray Cantor, and others, on the topic of Systems and Software Econometrics, among other things







Global Aerospace and Defense Market Environment

Drivers of Change (Mega-Trends)

Key A&D Industry Challenges

New Business and Operating models

Systems of systems and software
Product to services
Global integrated enterprise
Risk and compliance
Agility and efficiencies
Visibility and control

Segment health	2009	2010	2011	2012
Commercial/ Regional Aircraft	↓	↔	↑	↑
General Aviation Shipments	↓	↓	↔	↑
Defense Budgets	↑	↔	↓	↓

A&D companies are seeking growth through new business opportunities across multi-industry ecosystems ....

smarter products

- Aircraft & Air Vehicles
- Launch Vehicles
- Missiles / Weapons
- Railcars / Trains
- Satellites
- Ships / Submarines

enabling

smarter systems

- Energy Systems
- Environmental Systems
- Information Systems
- Security Systems
- Transportation Systems
- Water Systems
- Weapon Systems

4

Do more with less!

Meet increasingly unique customer needs

Make better investment decisions

React to market shifts

Deliver innovation to differentiate products

Enable business agility while doing more with less

Exploit globalization

Manage regulatory requirements

Maximize asset reuse

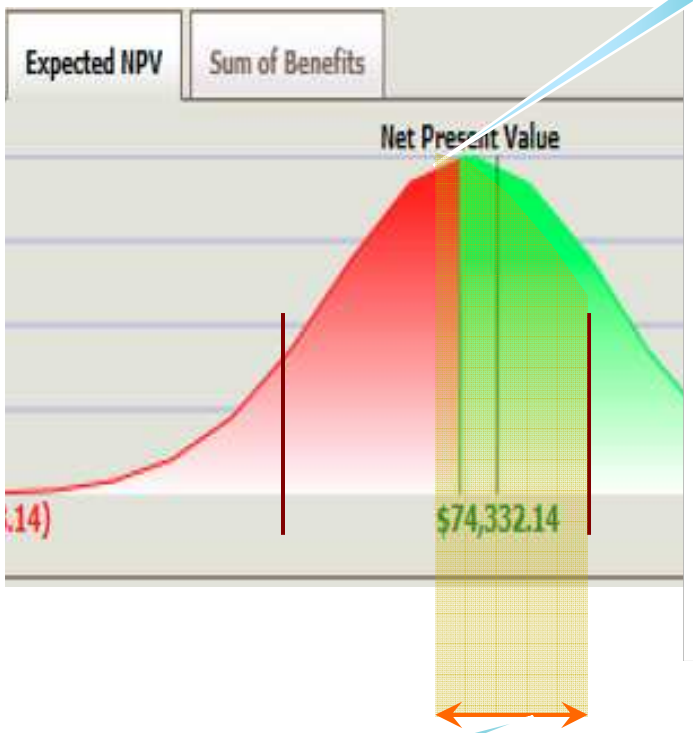
Improve quality

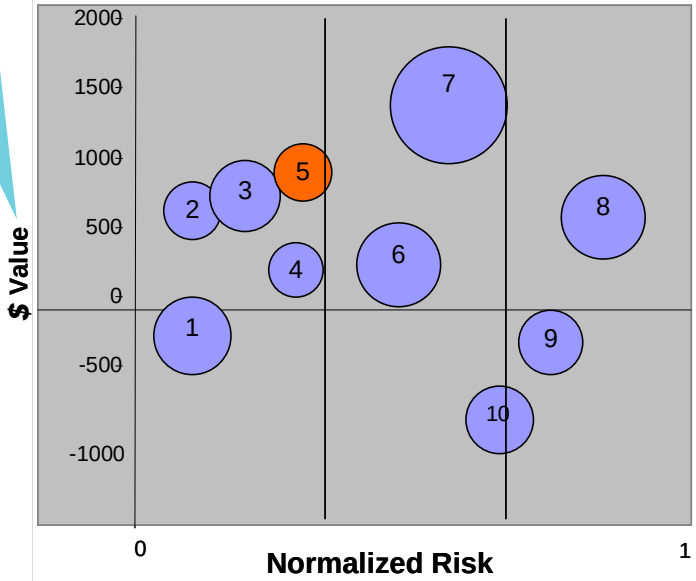
***Successful businesses** will be those that effectively deliver innovation while controlling cost and risk*

5

IBM's Investment Value Model *monetizes* value and risk

Investment Value = Mean



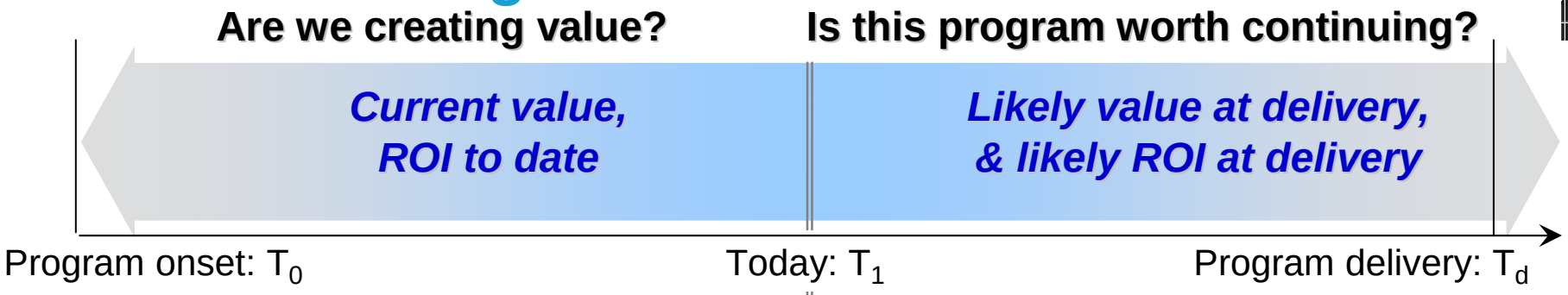


Standard Deviation

Normalized Risk = (Scaled) Standard Deviation

These two key questions support value-based decision making

- How do you compare routine and innovative efforts to each other?
- How do you manage project risk?
- How do you motivate architectural robustness and reuse?



Management Decisions Supported:

Monitoring
- **Is program healthy?**
- **Intervene?**
- **Cut losses?**

Investment
- **Is program still needed?**
- **Should we adjust content?**
- **Should we continue to invest?**

$$\text{To date ROI} = \frac{\text{mean(IV(today) – IV(onset))}}{\text{actual-costs-to-date}}$$

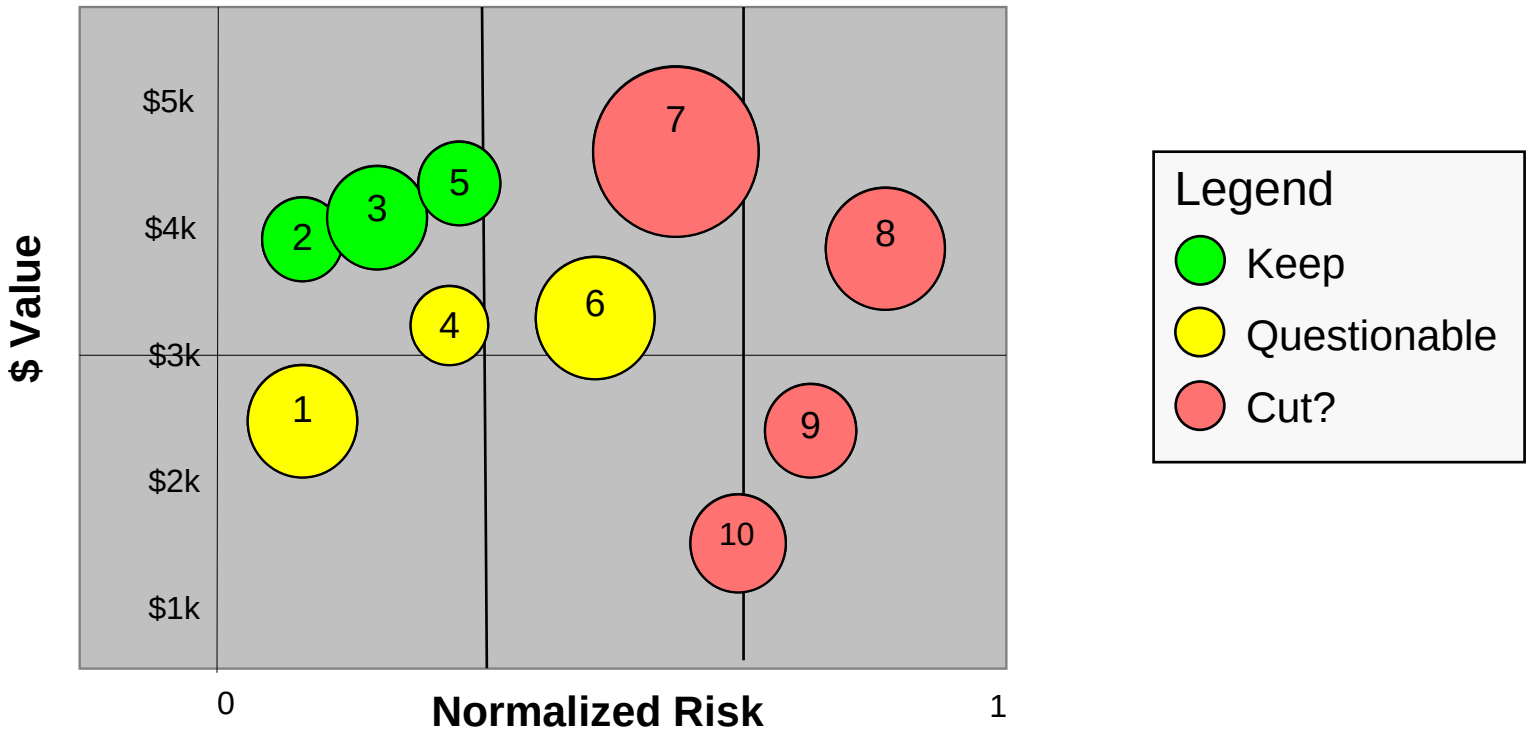
(a single value)

$$\text{To go ROI} = \frac{\text{IV(Deliv) – IV(today)}}{\text{PV(Costs to Deliv)}}$$

(a random variable)

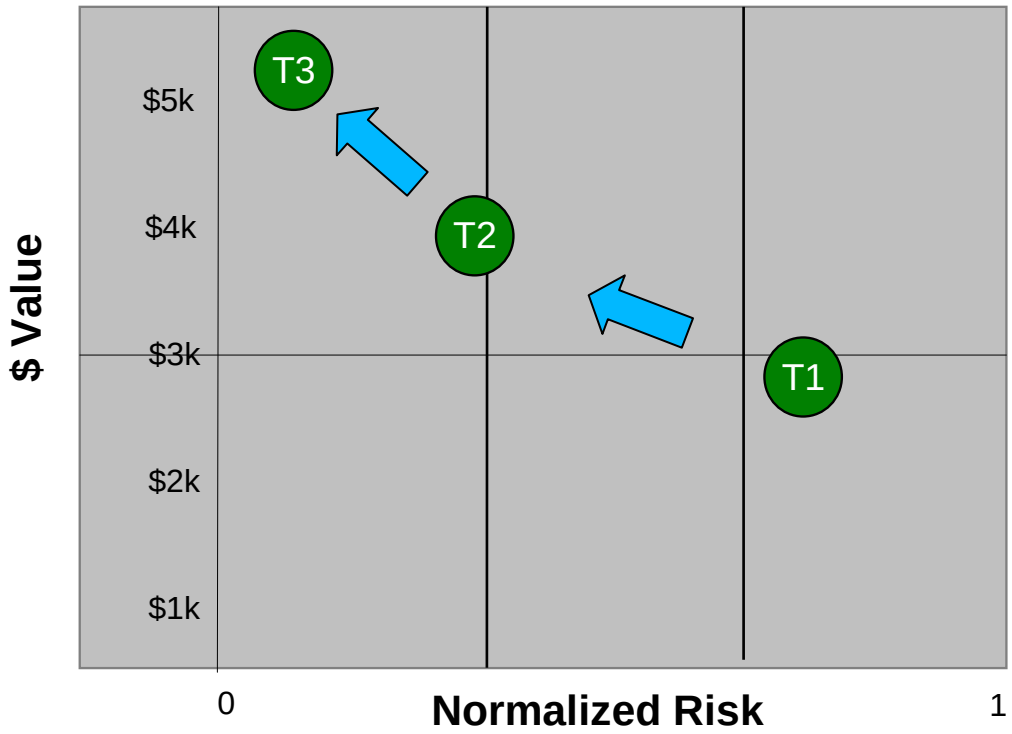
Software. Everyware.

The Model permits more objective management of the portfolio in the usual *resource-constrained* environment



Track improvement in value, and reduction of risk, throughout the project lifecycle

- T1 is project onset; T2 and T3 are later times in the lifecycle
- Movement from lower right to upper left shows that the investment (development) is delivering value



How do you keep your program from heading off a cliff?

Avoid the gotchas

- Pretending you know when you don’t
- Doing the easy things first
- Expecting life to be static, certain, and predictable
- Don’t mislead yourself with metrics

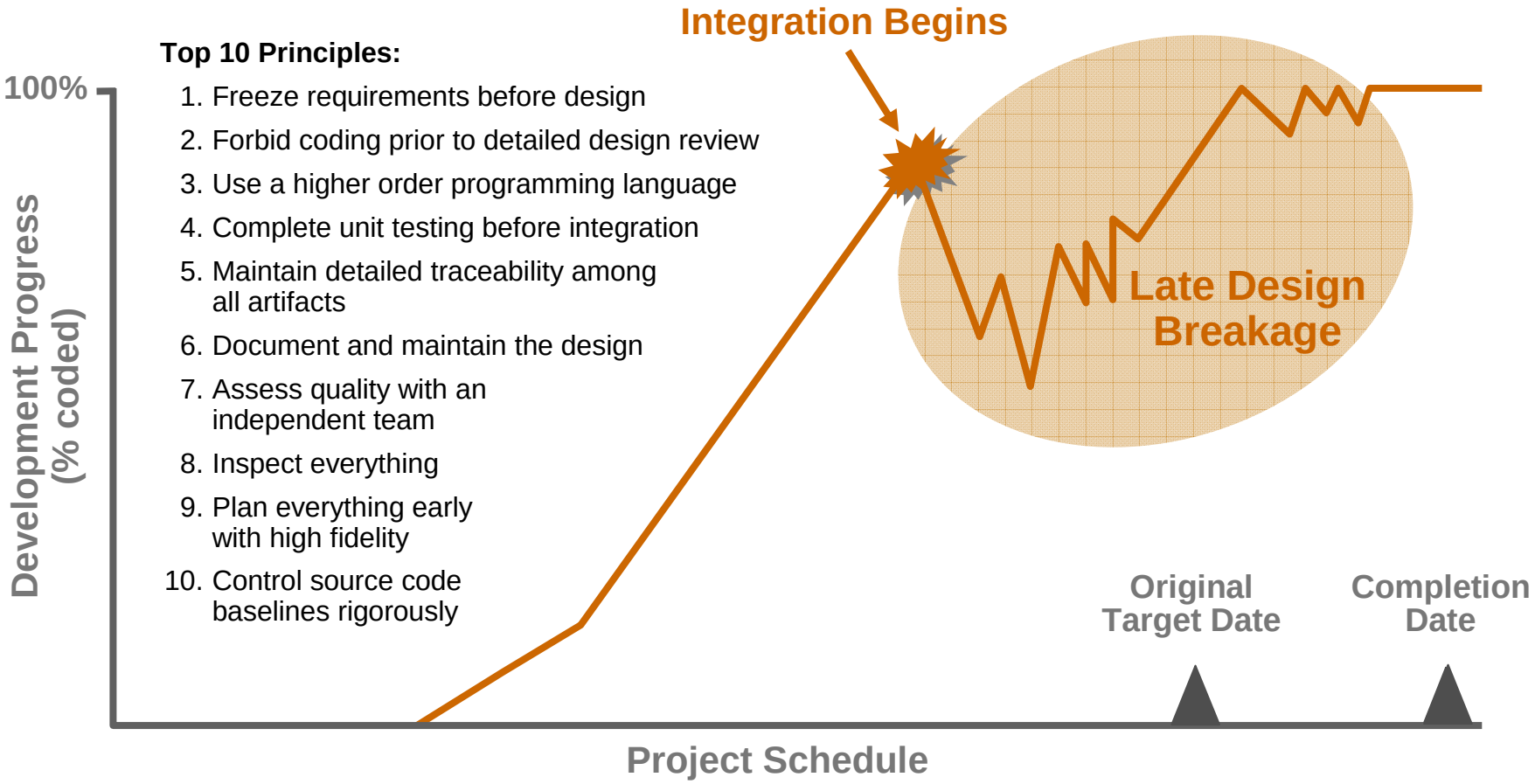
The predominant “as-is” state: Plan and track mentality

Sequential activities:

Requirements → Design → Code → Integration → Test

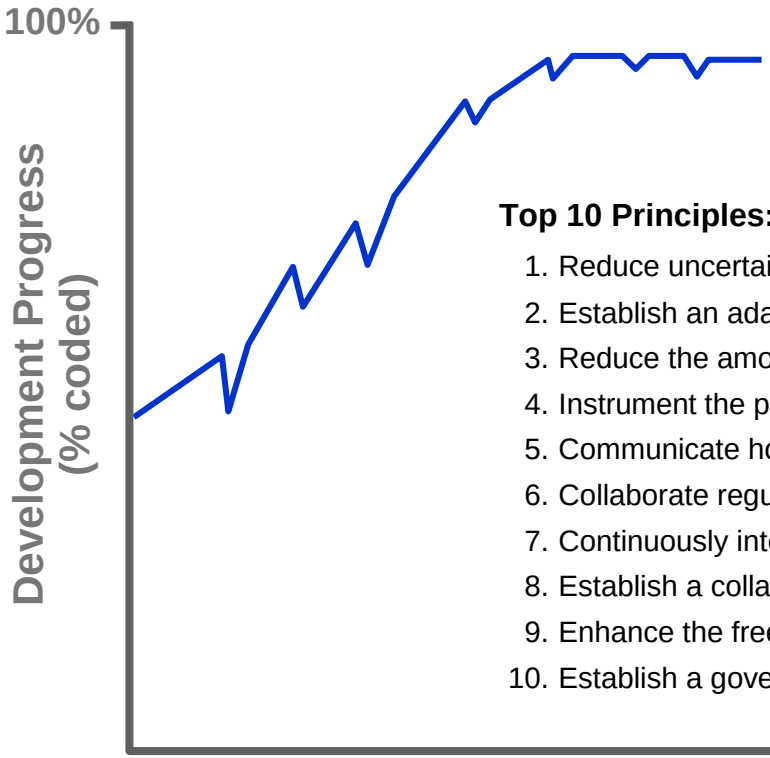
Top 10 Principles:

1. Freeze requirements before design
2. Forbid coding prior to detailed design review
3. Use a higher order programming language
4. Complete unit testing before integration
5. Maintain detailed traceability among all artifacts
6. Document and maintain the design
7. Assess quality with an independent team
8. Inspect everything
9. Plan everything early with high fidelity
10. Control source code baselines rigorously



Target state: Agile product delivery

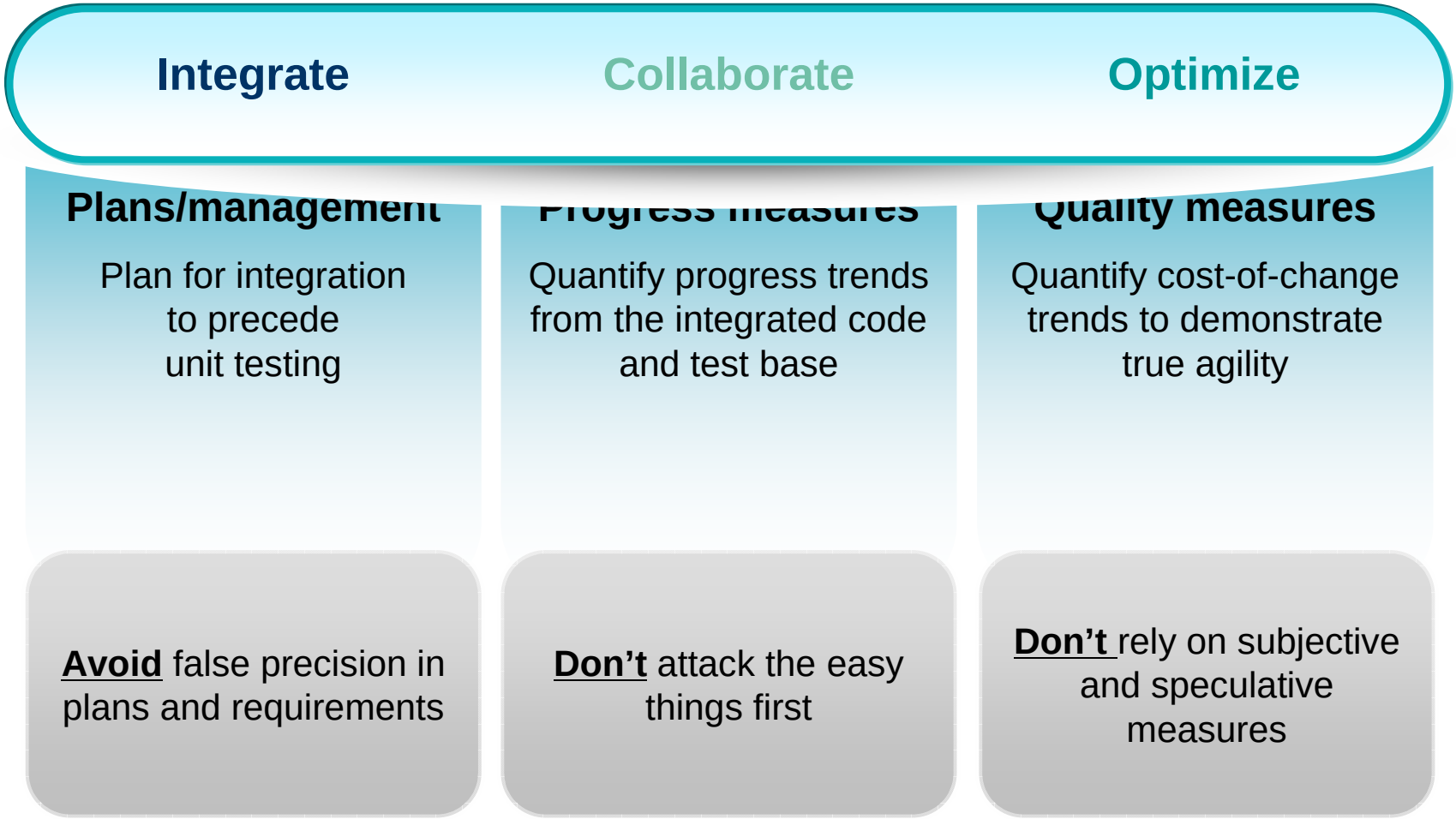
Assets → **Architecture Releases** → **Functional Releases** → **Product Releases**



Top 10 Principles:

1. Reduce uncertainties by addressing architecturally significant decisions first
2. Establish an adaptive life-cycle process that accelerates variance reduction
3. Reduce the amount of custom development through asset reuse and middleware
4. Instrument the process to measure cost of change, quality trends, and progress trends
5. Communicate honest progressions and digressions with all stakeholders
6. Collaborate regularly with stakeholders to renegotiate priorities, scope, resources, and plans
7. Continuously integrate releases and test usage scenarios with evolving breadth and depth
8. Establish a collaboration platform that enhances teamwork among potentially distributed teams
9. Enhance the freedom to change plans, scope and code releases through automation
10. Establish a governance model that guarantees creative freedoms to practitioners

Pivotal Culture Shifts



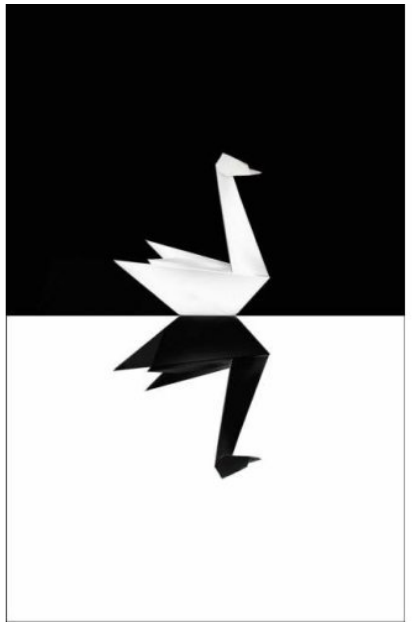
How do you reduce risk?

- Admit that you don’t know
- Do the hard things first

Socrates

- ἓν οἶδα ὅτι οὐδὲν οἶδα :
- I know that I don’t know

- Nicholas Taleb—The Black Swan



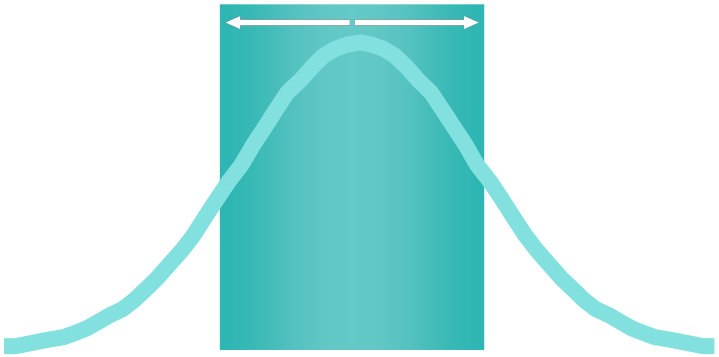
- Royce—”One of the most common failure patterns in the software industry is to develop a five-digits-of-precision version of a requirement specification (or plan) when you have only a one-digit-of-precision understanding of the problem”

But can we measure what we don’t know? Or estimate it?

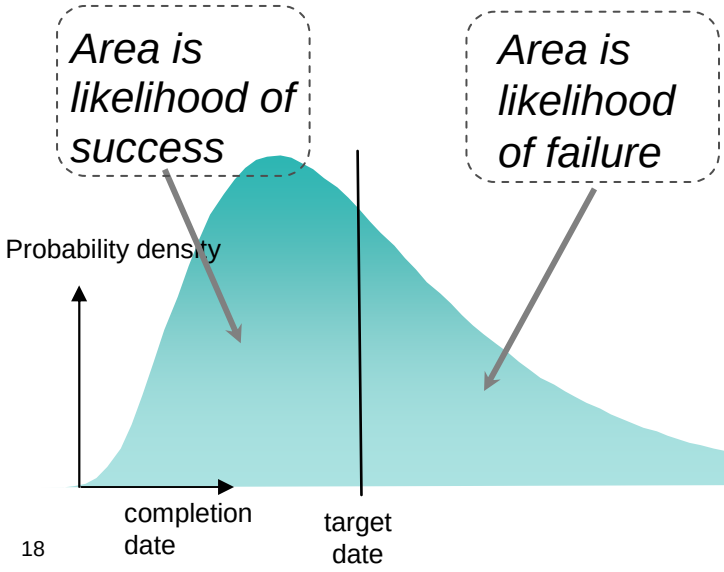
Variance as a measure of risk

Reducing the variance and improving the odds is the mathematics of agile development

Probability of measurement



Variance of measurement

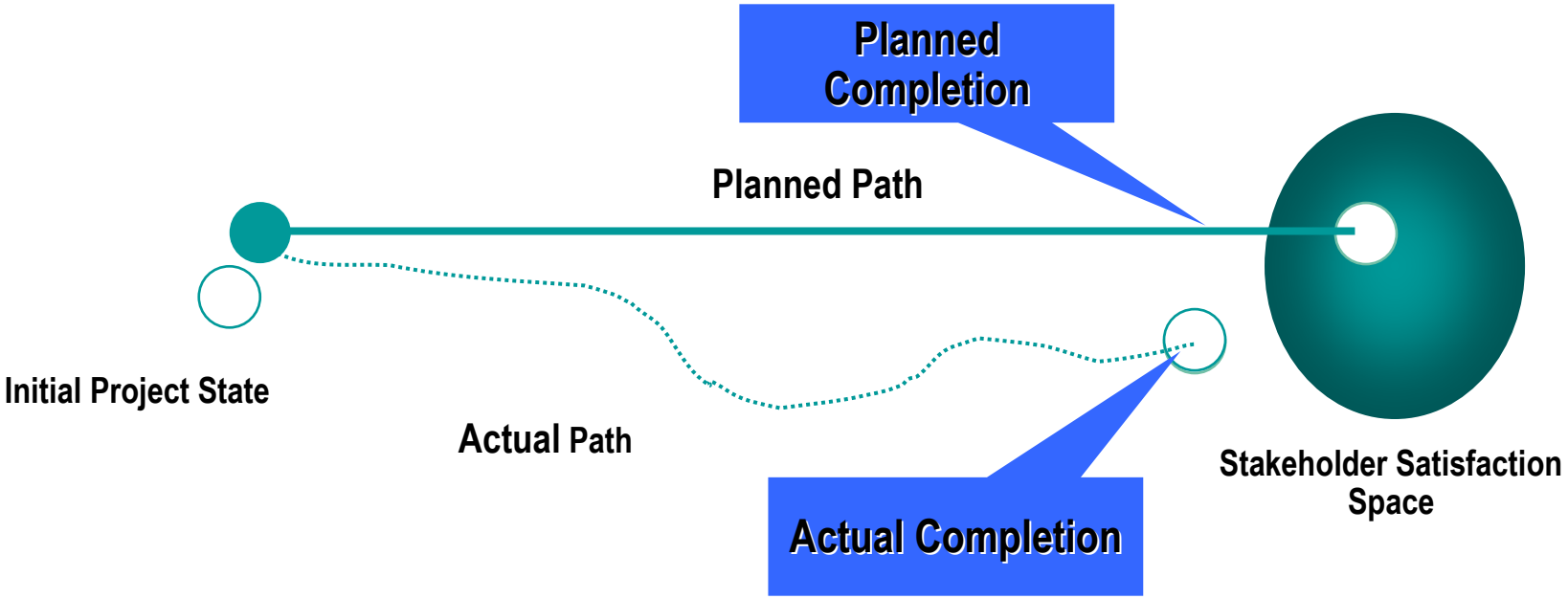


- Program parameters (cost, schedule, effort) are uncertain and so you would like to know the odds
- Specify each parameter as a random variable described by a mathematical distribution
- Area under distribution curve describes probability of measurement falling in range
- Reduction of variance reflects increased predictability of outcome

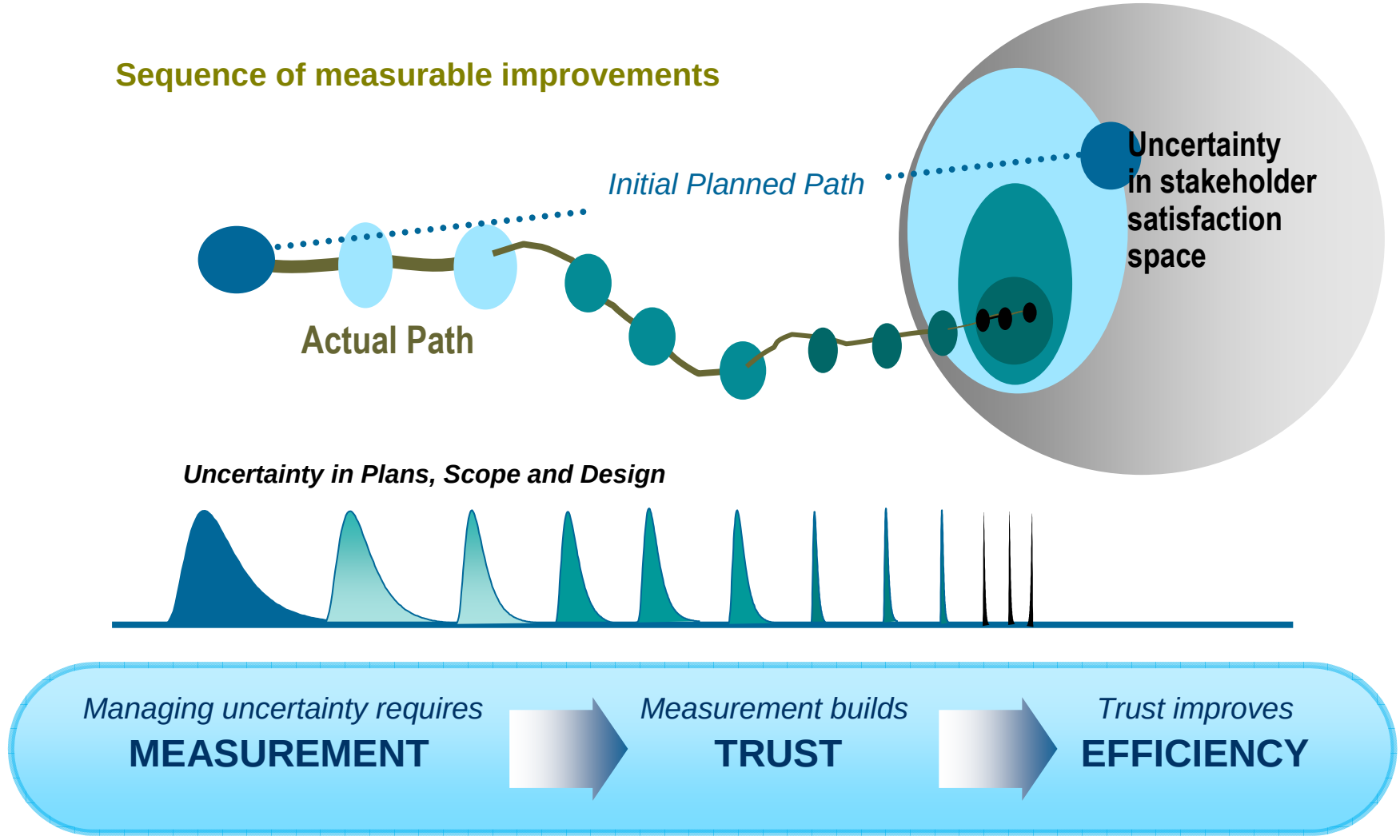
"I need to improve my overall development capability and predictability."

Old method emphasizes "Plan and Track"

- Plan and estimate the activities of the project for the entire life cycle and then track to the plan.
- Assess variances between actuals and plans



Economic Governance: Measurement and Steering



Do the hard things first

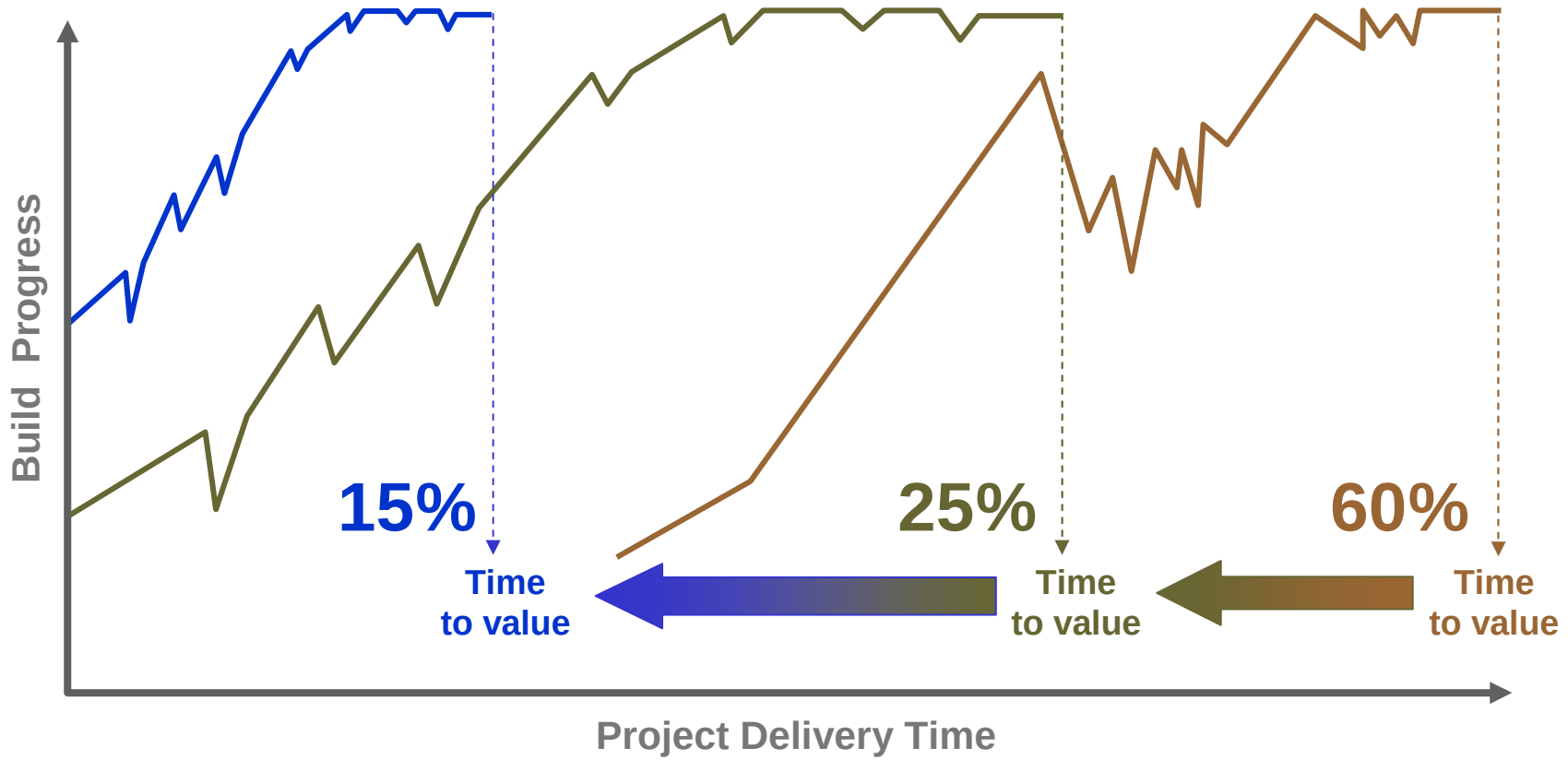
- Iterative development
- Integrate, then test

Improving time to value

Economic governance
SOAs and assets
Collaborative platforms

Iterative processes
Middleware components
Mature commercial tools

Waterfall Governance
Stovepipe architectures
Proprietary tools/methods



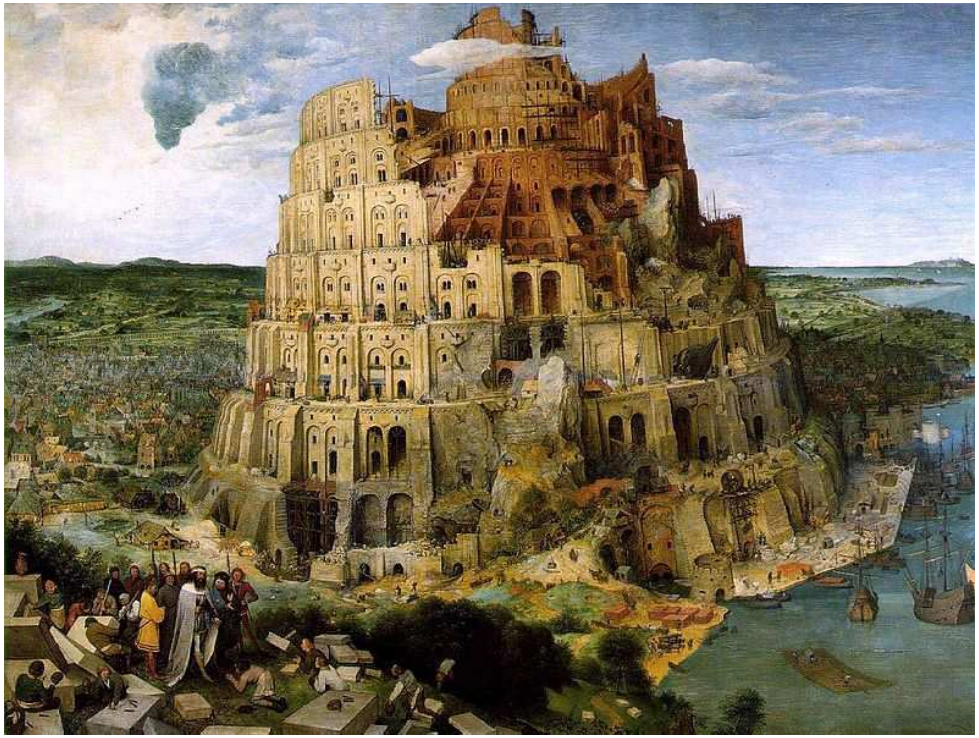
Integrate then test

- The sooner you integrate, the lower your risk
- Force integration between collaborating subsystems to reduce writing of emulation code, and surface problems earlier
- This also enforces better collaboration between teams
- Use models to determine integration schedules
- Use models to drive testing
- Execute models to test integration

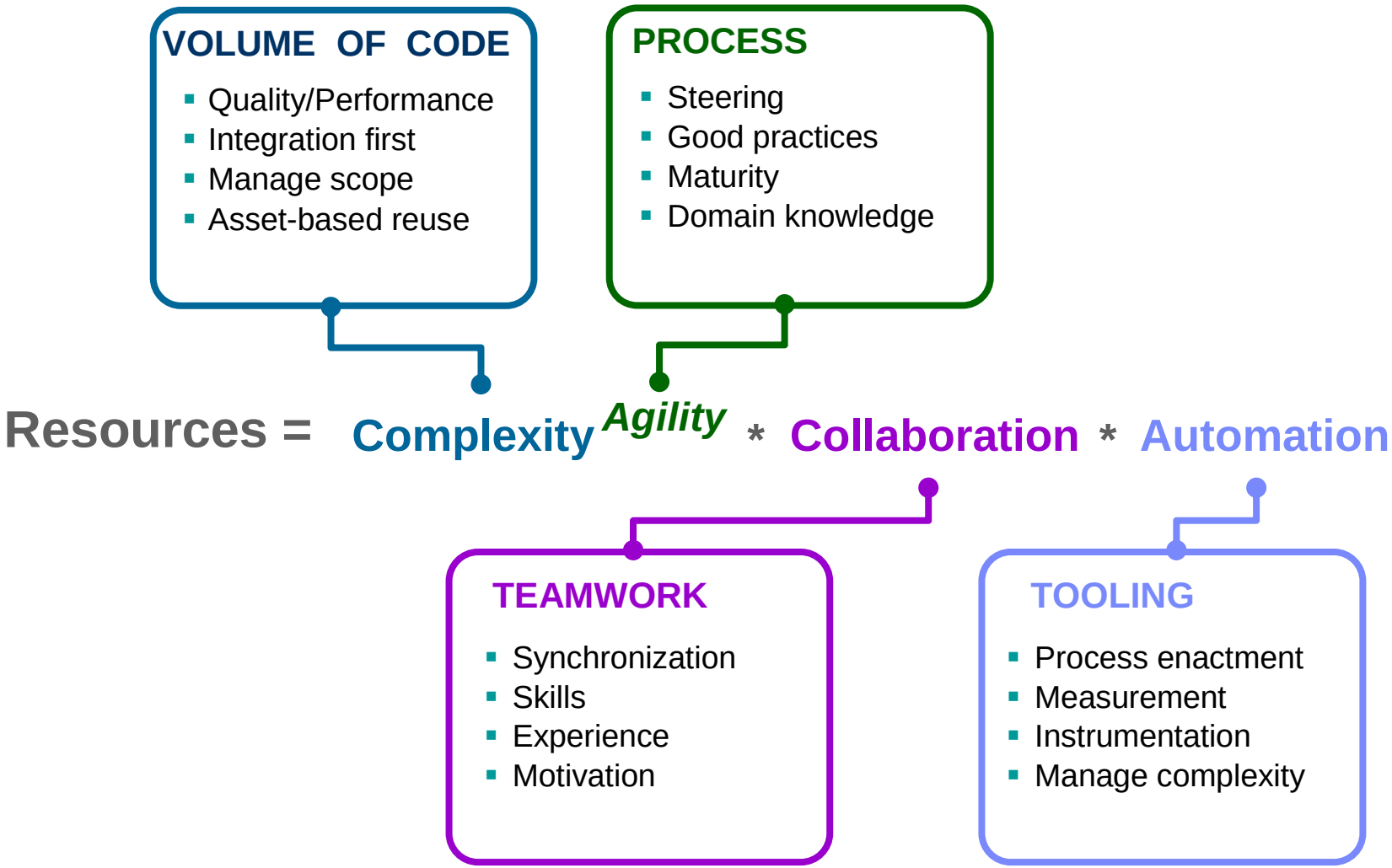
Do not expect life (programs, projects) to be static and predictable

- Manage change
- Manage complexity

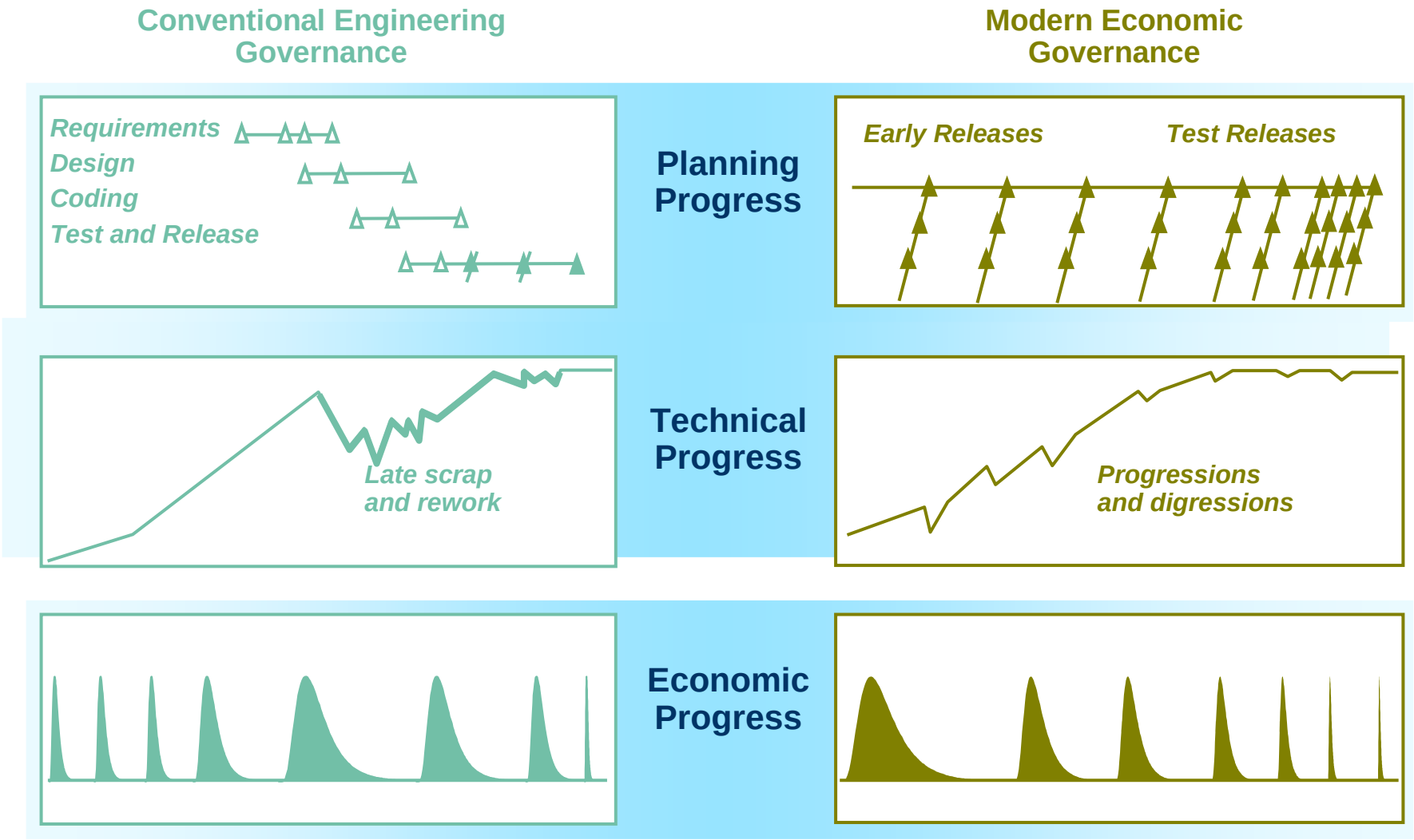
Why so complex?



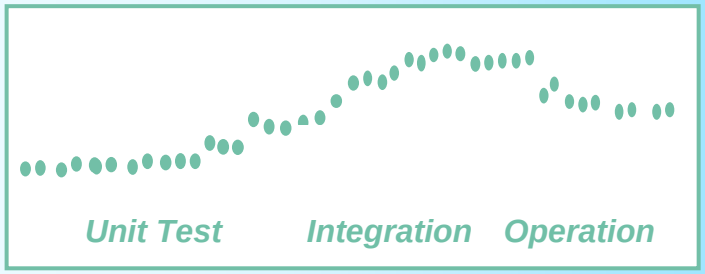
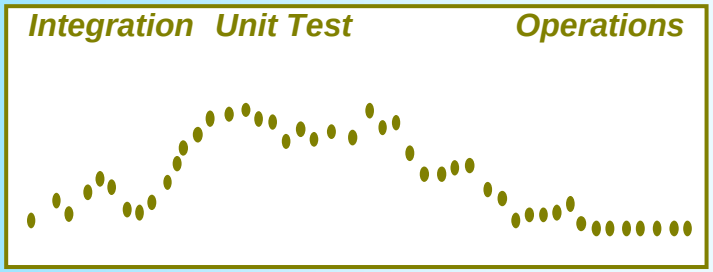
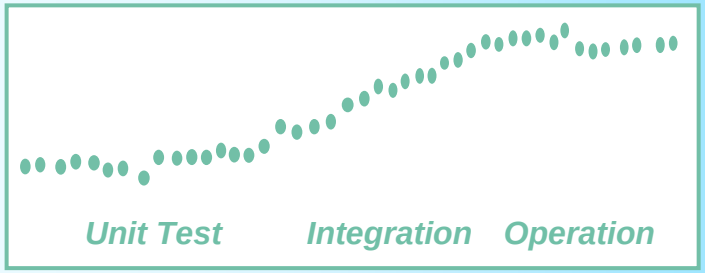
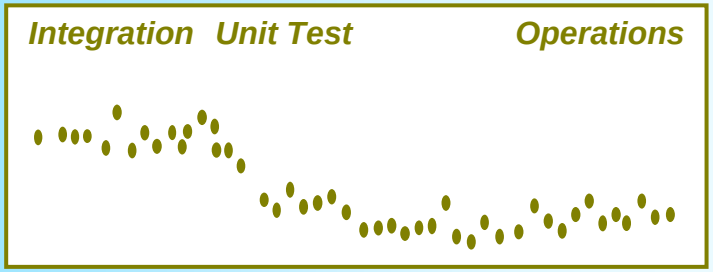
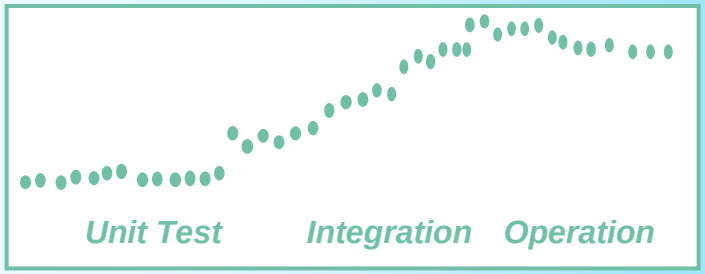
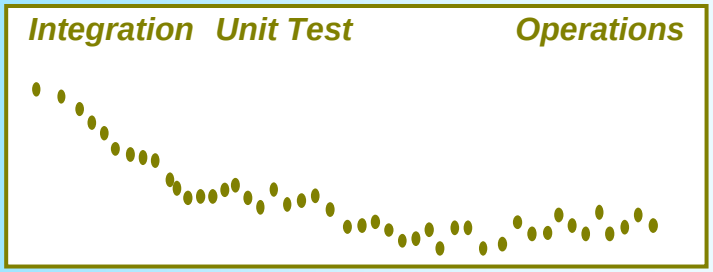
Improving Software Economics



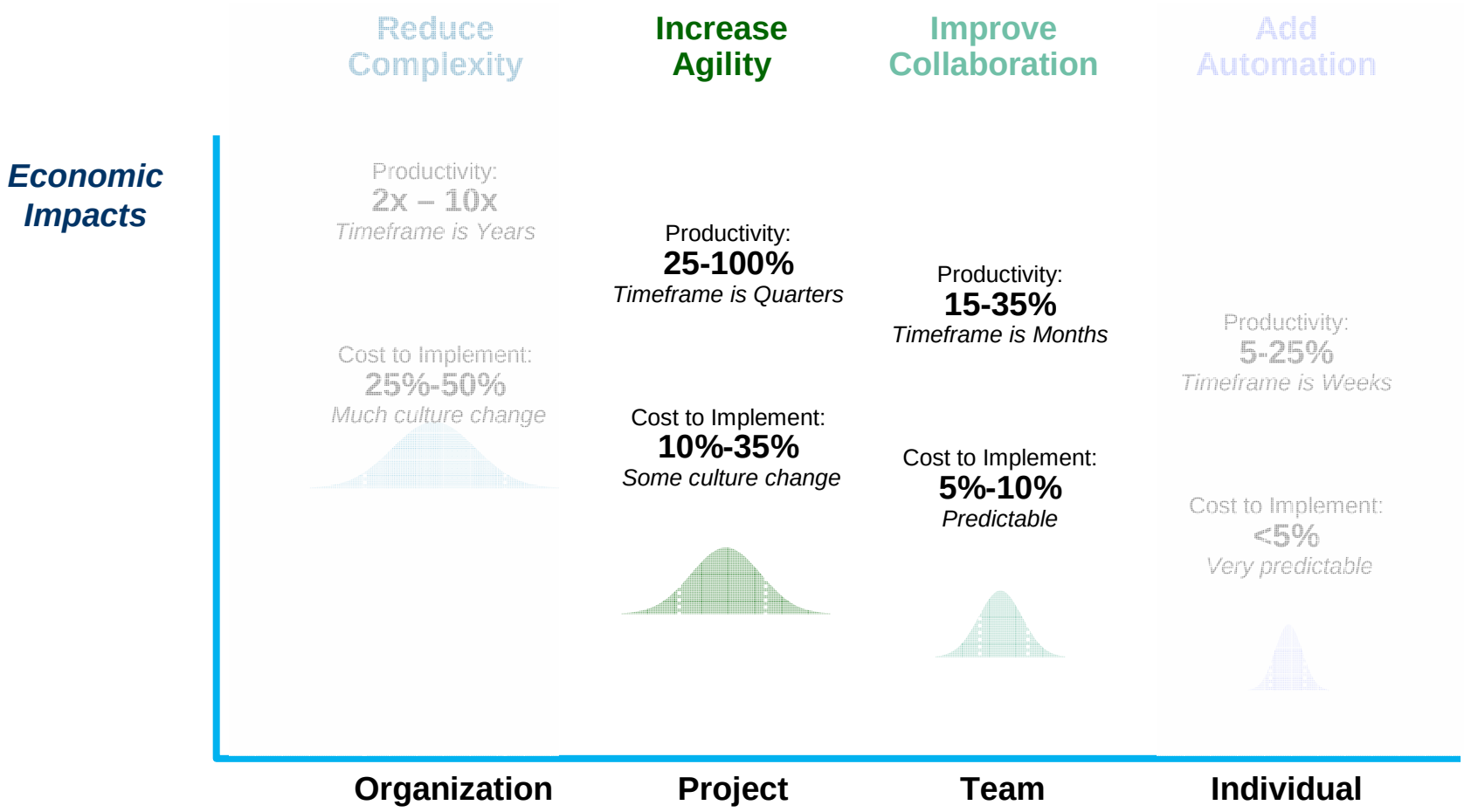
Measured Improvement: *Progress Econometrics*



Measured Improvement: *Quality Econometrics*

Conventional Engineering Governance		Modern Economic Governance
	**Maturity** ***Defect Trend***	
	**Modularity** ***Change Volume Trend***	
	**Adaptability** ***Cost of Change Trend***	

Productivity Improvement Leverage



Software development obsolesced by software delivery

Software Development	Software Delivery
Distinct development phase	*Continuously evolving systems*
Distinct handoff to maintenance	*No distinct boundary between development and maintenance*
Requirements-design-code-test sequence	*Sequence of released capabilities with ever increasing value*
Phase and role specific tools	*Common platform of integrated process / tooling*
Collocated teams	*Distributed, web based collaboration*
Standard engineering governance	*Economic governance tailored to risk / reward profiles*
Engineering practitioner led	*Business value and outcome led*

The Moral of This Story

- **Better software economics is a result of:**

- **1. Measured improvement for improved predictability**
  - The foundation of economic governance
  - Measurement helps you manage uncertainty

- **2. Agility for improved operational efficiency**
  - Best measured by cost of change trends
  - Best achieved by accelerating integration testing

If you play better defense you can play more offense!

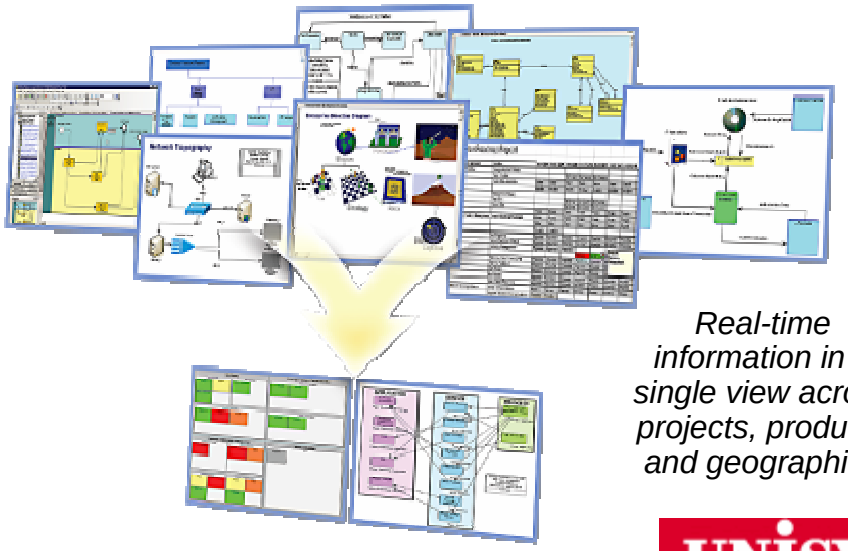
www.ibm.com/software/rational

www.ibm.com/software/rational

IBM Rational Insight

Automated project and process measurement can improve team productively by 15%

- Monitor and analyze project and process performance
  - ***Collaborate*** across disparate development artifacts and in the context of operational and business objectives
  - ***Automate*** measurement leveraging an open RESTful and Jazz compliant data integration architecture
  - ***Report*** on relevant data with a built-in library of industry best practice metrics and dashboards
- Drive business innovation and reduce costs through measures and continuous **process improvement**
- Built on best in class business intelligence platform from IBM **Cognos**



Real-time information in a single view across projects, products and geographies

UNISYS imagine it. done.

"Until now we were gathering metrics by manual methods. Rational Insight simplifies the presentation... Overall great tool to have for optimizing project efforts and giving management a birds eye view of the project performance at all stages."

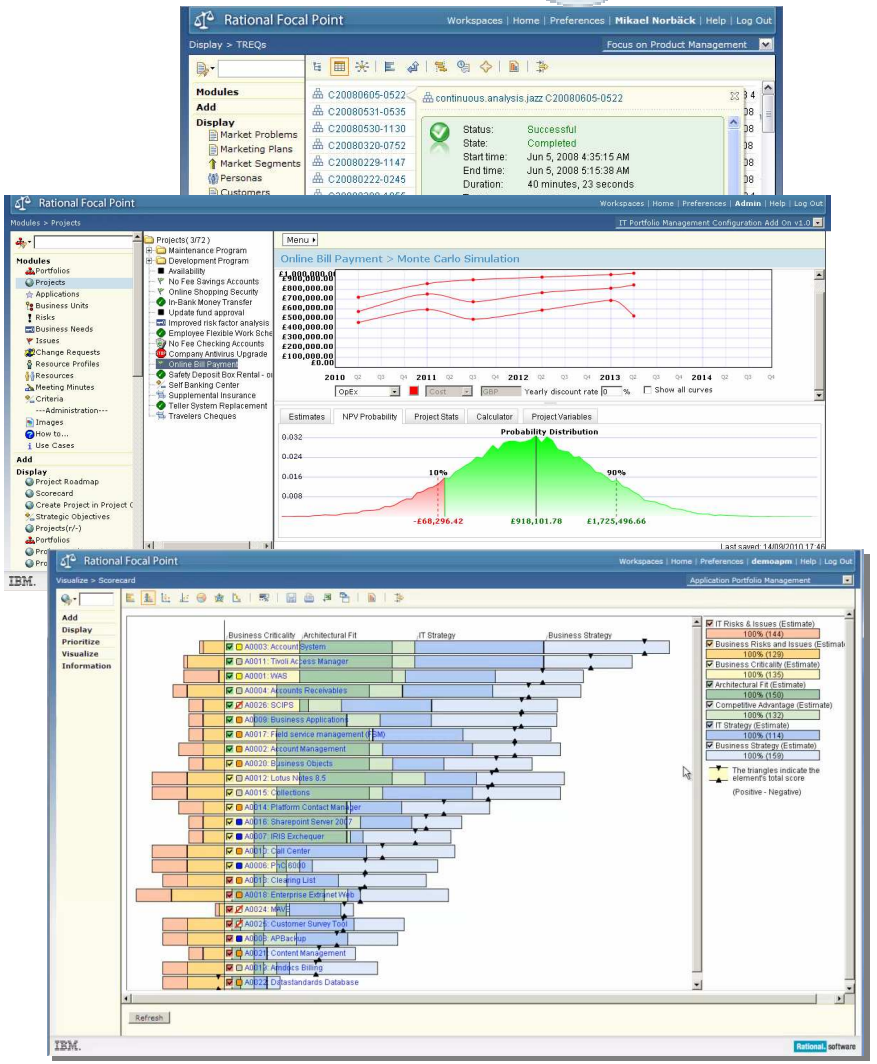
IBM

"Through the automation and dashboard capabilities in Insight, our team is saving over 40 hours a month on our reporting activities. We also can offer new capabilities such as "on-demand" dashboards which would have been very labor intensive to do without Insight"

Rational Focal Point 6.5

Hardwiring the linkage between strategy and execution

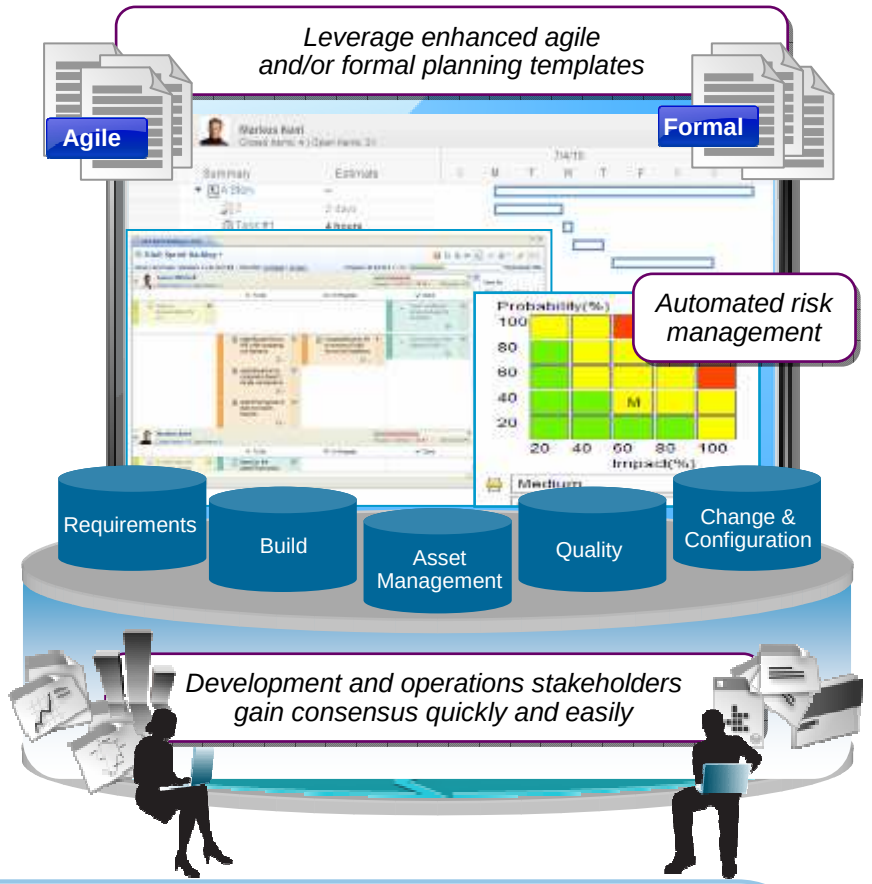
- **The Rational Team Concert integration allows users to prioritize and manage project scope and rollup project status**
- **The System Architect integration connects the Enterprise Architecture perspective to the portfolio management perspective in Focal Point**
- **FP's Investment Analysis component assists users with financial modeling and business case assessment**
- **Users can take advantage of advanced resource management allowing skill-based supply and demand tracking and balancing**
- **Configuration templates included in the product helps users get up and running quickly**



Rational Team Concert 3.0

Enhanced project planning capabilities, templates and risk tracking

- Users can leverage agile and formal planning methodologies – or a hybrid of the two
  - New "out-of-the-box" templates for both agile and formal project management
  - Maximize resource allocation and scheduling with integrated planning and execution
  - Quickly view dependencies and critical paths to avoid progress delays
- Stakeholders have increased visibility and insight into project risks
  - Weight and evaluate risk at each step of the development project plan



▪**48% of companies surveyed are using a hybrid of multiple development methodologies, 20% agile, 12% iterative and 11% waterfall. All of them could benefit from the new combined project planning capabilities.**

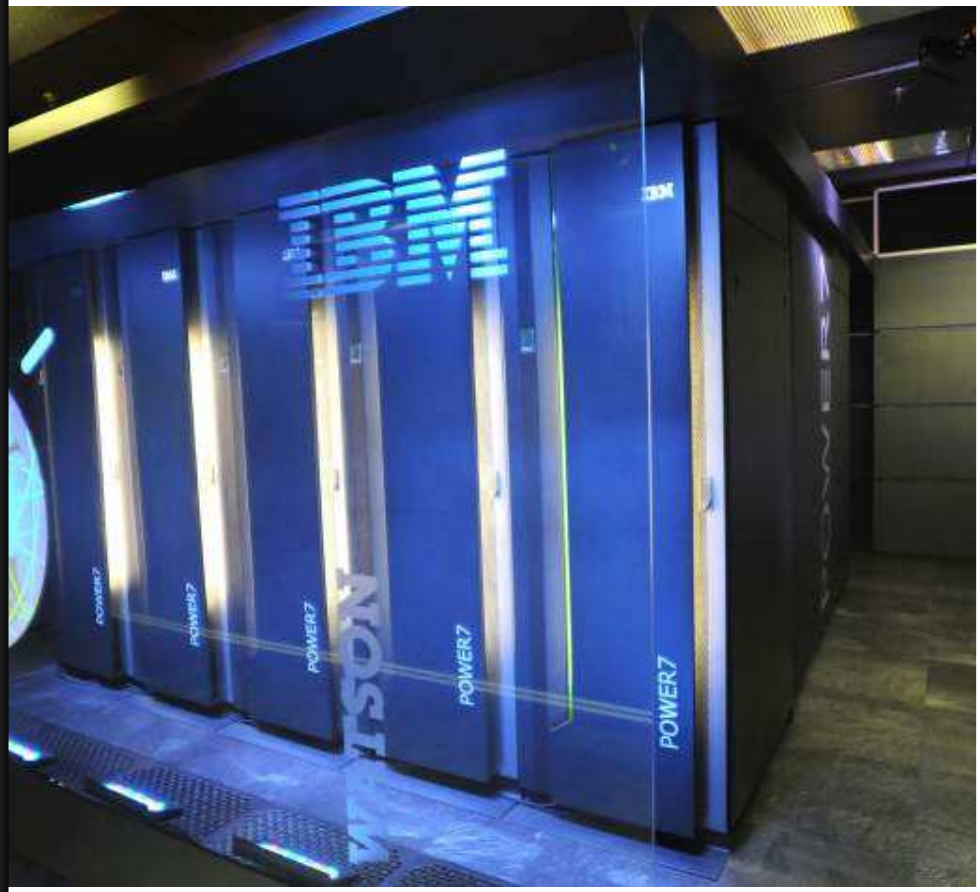
Learn More

IBM A&D Solutions

IBM Rational A&D solutions

ibm.com/Watson

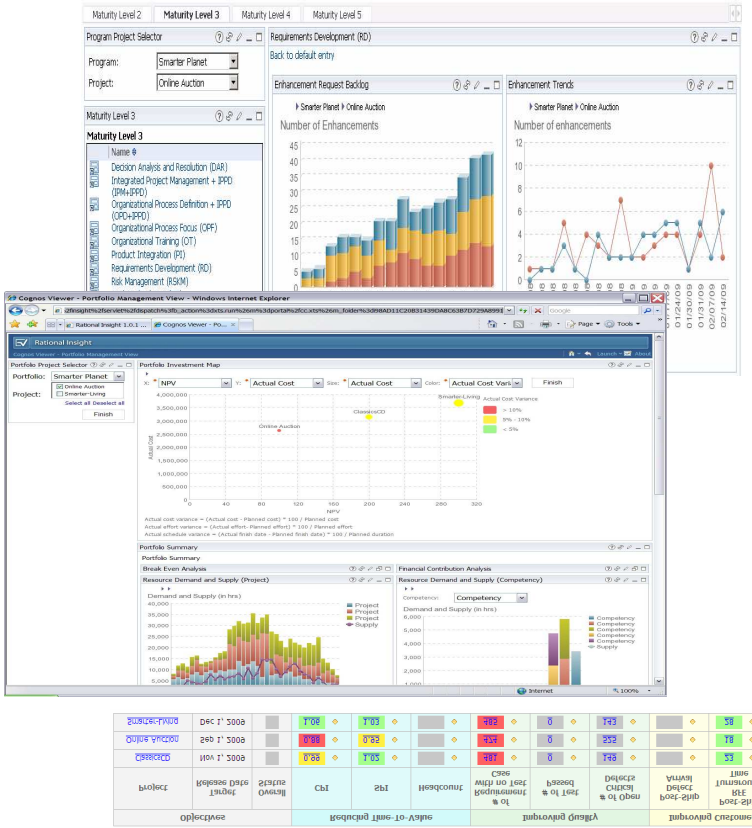
ibm.com/BAO



Backup

Rational Insight 1.0.1

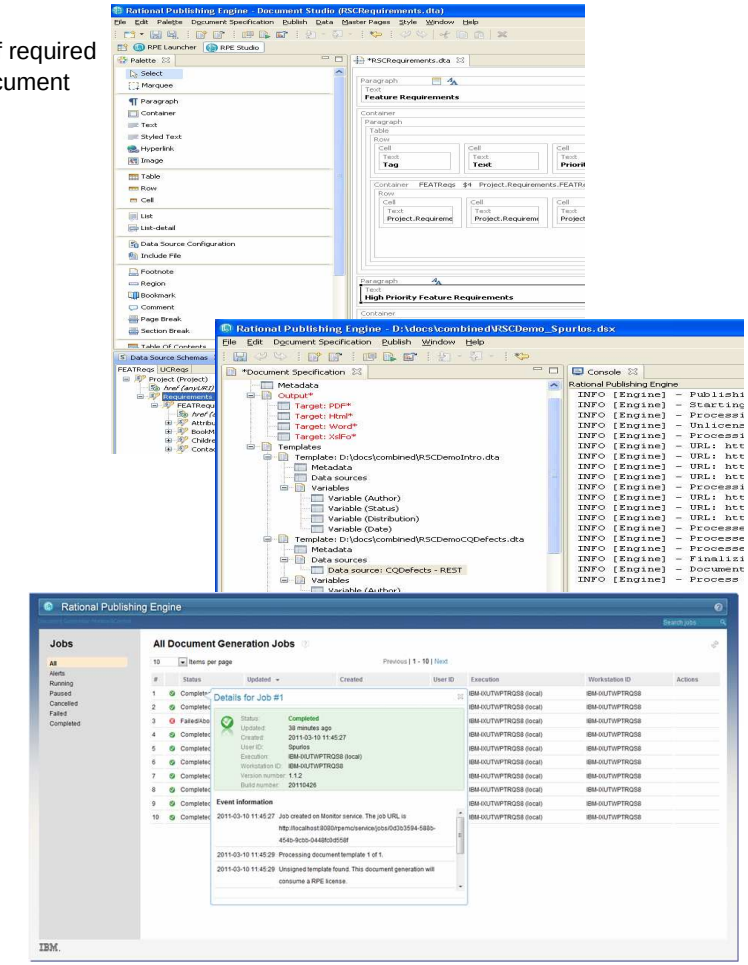
- **Enhanced Rational product integrations**
  - Support for Rational Focal Point
  - Support for RTC 3.0
- **Installation and Configuration Enhancements**
  - 64 bit Linux and Windows server support
  - Full support for IE and Firefox browsers
  - CLM Workbench compatible DW schema
  - Translation to many different languages
- **Usability Enhancements**
  - Simplified Data warehouse setup and configuration
  - Simplified Dynamic Schema configuration
- **OOTB Report and Dashboard Enhancements**
  - Exec DB style Project health scorecards
  - CMMI based dashboards / reports
  - Performance Management / RMC dashboards / reports
- **Event Studio for KPI and Event monitoring and notification**



Backup

Rational Publishing Engine 1.1.2

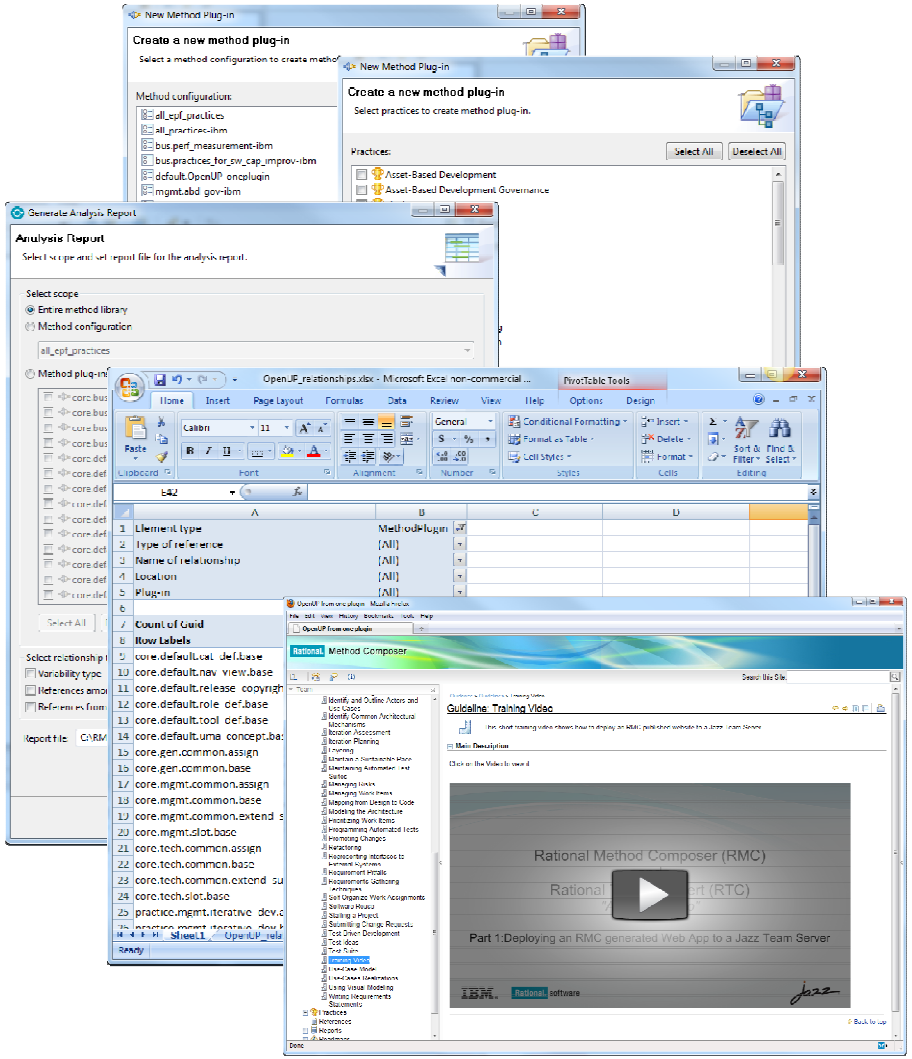
- **Distributed environment robustness and scalability**
  - Monitor and Control providing administrators with means to monitor jobs and cancel if required
  - Disconnection recovery enabling user mobility and workstation hibernation during document generation
  - Load balance improving concurrency support
- **Enhanced template performance and flexibility**
  - Dynamic variable support in ‘native’ filters enabling query variables to be assigned at generation time
  - Conditional repetition independent of data query
  - Post document generation execution enabling workflow integrations
- **Template translation facilitating multilingual re-use**
- **Casting enabling drilldown into nested models**
- **Migration assistants for SoDA and DocExpress**
- **Technically**
  - Install Manager replacing Installsheild technology
  - Data source support via OSLC Reporting Profile
  - Platforms: Windows 2008 R2 & WAS 6.1 / 7.0



Rational Method Composer 7.5.1.1

Simplified tailoring, relationship reports, rich content and work item export

- **Create a single plug-in based on a method configuration and/or selection of practices to simplify tailoring by non-experts**
- **New CSV report supports impact analysis and analysis of changes**
- **Improved element move across plug-ins that simplifies library management**
- **Enhanced work item template export that bridges the gap for enactment with RTC**
- **Ability to incorporate multi-media content and JavaScript in published pages**



Monte Carlo Simulation (Wikipedia[1] has a good article at http://en.wikipedia.org/wiki/Monte_Carlo_method)

Monte Carlo methods (or Monte Carlo experiments) are a class of computational algorithms that rely on repeated random sampling to compute their results. Monte Carlo methods are often used in simulating physical and mathematical systems. These methods are most suited to calculation by a computer and tend to be used when it is infeasible to compute an exact result with a deterministic algorithm. This method is also used to complement the theoretical derivations.

- Monte Carlo methods are especially useful for simulating systems with many coupled degrees of freedom, such as fluids, disordered materials, strongly coupled solids, and cellular structures (see cellular Potts model). They are used to model phenomena with significant uncertainty in inputs, such as the calculation of risk in business. They are widely used in mathematics, for example to evaluate multidimensional definite integrals with complicated boundary conditions. When Monte Carlo simulations have been applied in space exploration and oil exploration, their predictions of failures, cost overruns and schedule overruns are routinely better than human intuition or alternative "soft" methods.
- The Monte Carlo method was coined in the 1940s by John von Neumann, Stanislaw Ulam and Nicholas Metropolis, while they were working on nuclear weapon projects in the Los Alamos National Laboratory. It was named in homage to Monte Carlo casino, a famous casino, where Ulam's uncle would often gamble away his money.

Introduction and example: Monte Carlo method applied to approximating the value of π

- Monte Carlo methods vary, but tend to follow a particular pattern:
  - Define a domain of possible inputs.
  - Generate inputs randomly from a probability distribution over the domain.
  - Perform a deterministic computation on the inputs.
  - Aggregate the results.
- For example, given that a circle inscribed in a square and the square itself have a ratio of areas that is π/4, the value of π can be approximated using a Monte Carlo method:
  - Draw a square on the ground, then inscribe a circle within it.
  - *Uniformly* scatter some objects of uniform size (grains of rice or sand) over the square.
  - Count the number of objects inside the circle and the total number of objects.
  - The ratio of the two counts is an estimate of the ratio of the two areas, which is π/4. Multiply the result by 4 to estimate π.
- In this procedure the domain of inputs is the square that circumscribes our circle. We generate random inputs by scattering grains over the square then perform a computation on each input (test whether it falls within the circle). Finally, we aggregate the results to obtain our final result, the approximation of π.
- To get an accurate approximation for π this procedure should have two other common properties of Monte Carlo methods. First, the inputs should truly be random. If grains are purposefully dropped into only the center of the circle, they will not be uniformly distributed, and so our approximation will be poor. Second, there should be a large number of inputs. The approximation will generally be poor if only a few grains are randomly dropped into the whole square. On average, the approximation improves as more grains are dropped.