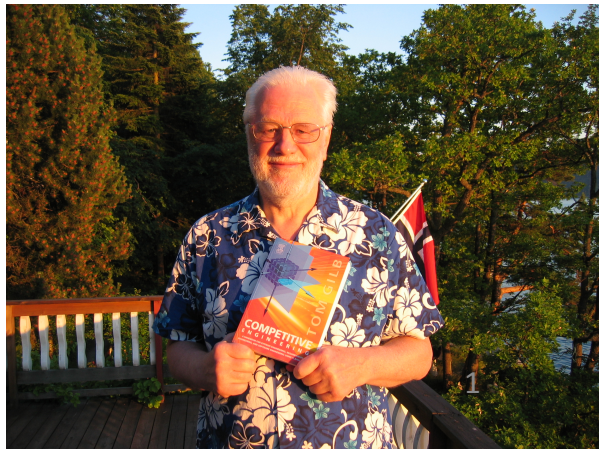
The *'Real* Quality Assurance' Manifesto: *are you ready for the next generation QA?*

Presenter: Tom Gilb, Gilb.com

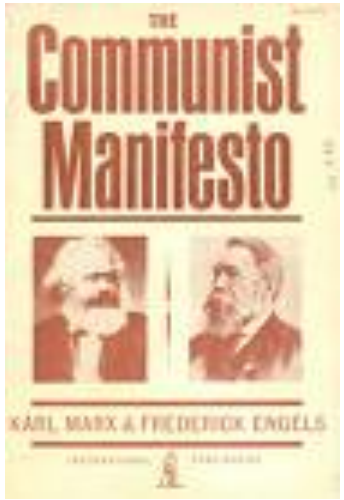
BCS SIGIST Conference - 'Motivating Testers' –
Thursday 10th December 2010
Keynote 09.30 – 10.30
London



Quality Manifesto/Declaration

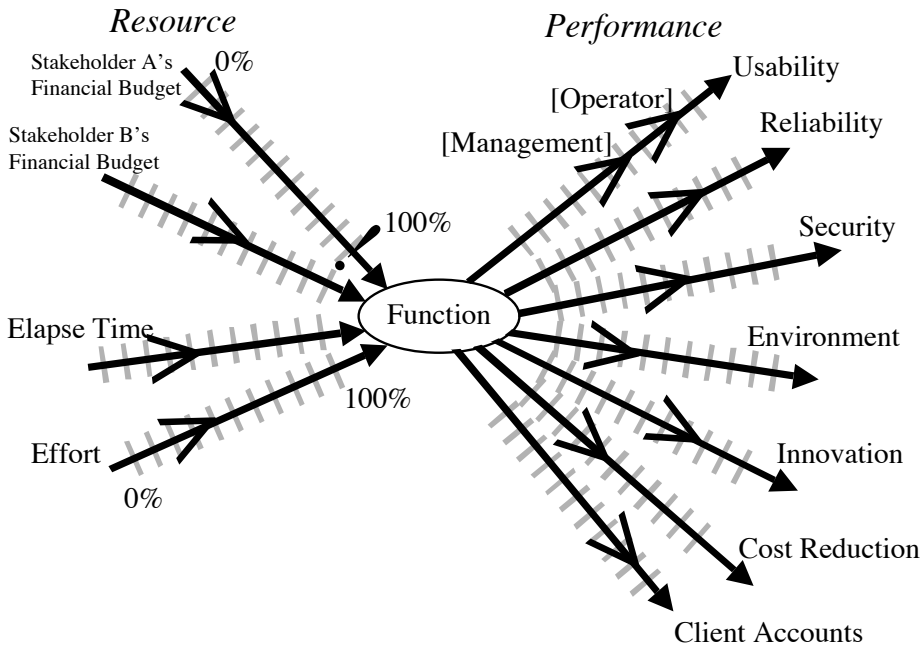
(1 page overview, see *following* slides for **detail**)

- **System Quality can be viewed as a set of quantifiable performance attributes, that describe how well a system performs for stakeholders, under defined conditions, and at a given time.**
- **System Stakeholders judge past, present, and future quality levels; in relationship to own their perceived needs/values.**
- **System Engineers can analyze necessary, and desirable, quality levels; and plan, and manage to deliver, a set of those quality levels, within given constraints, and available resources.**
- **Quality Management is responsible for prioritizing the use of resources, to give a satisfactory fit, for the prioritized levels of quality: and for trying to manage the delivery of a set of qualities - that maximize value for cost - to defined stakeholders.**

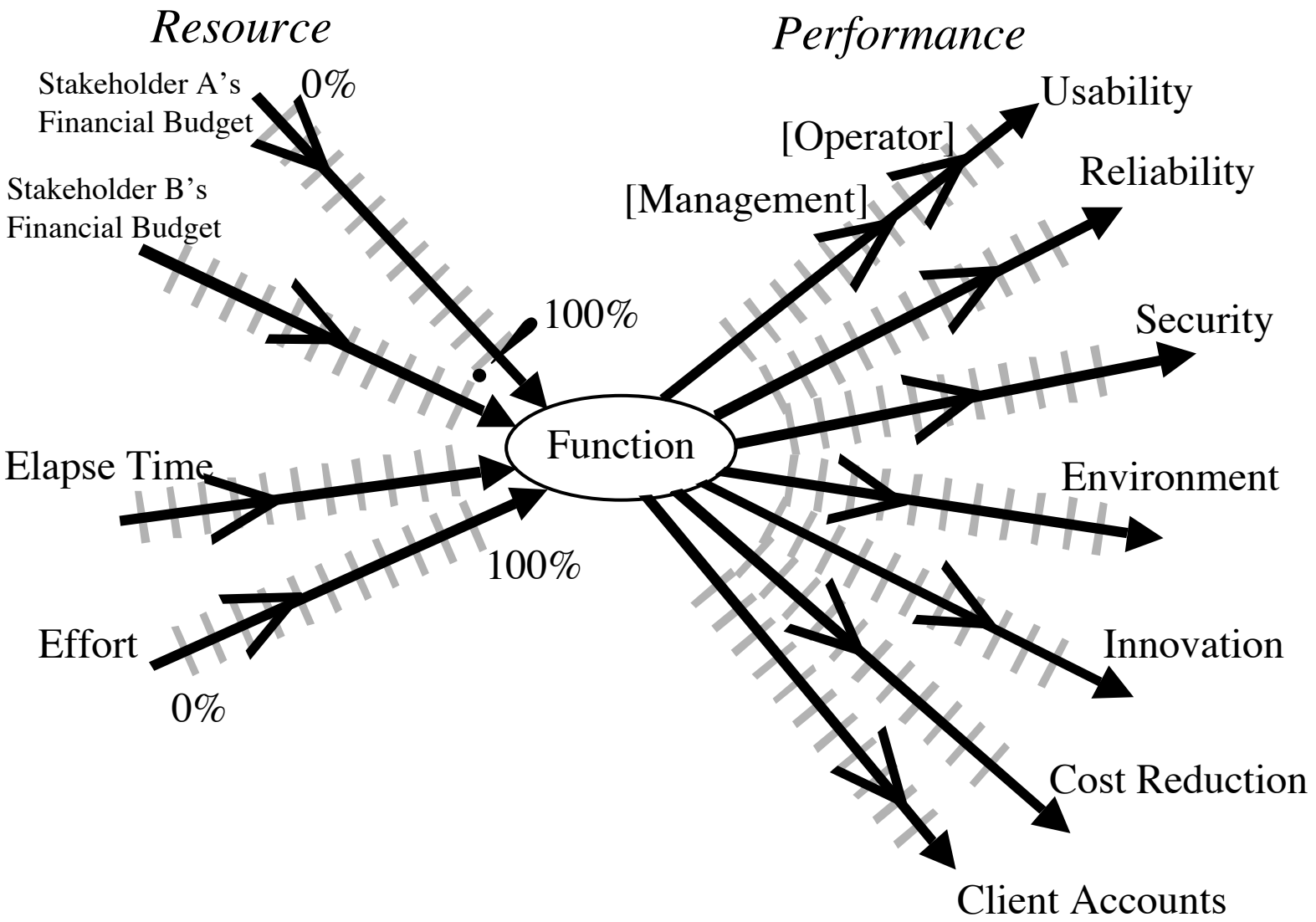


1. System Quality

- can be viewed as a set of *quantifiable* 'performance attributes',
- that describe *how well* a system performs for stakeholders,
  - under defined conditions, and at a given time.



Multiple Required Performance and Cost Attributes are the basis for architecture selection and evaluation



2. System Stakeholders

- judge
  - –past, present, and future quality levels;
  - –in relationship to own their perceived needs/values.

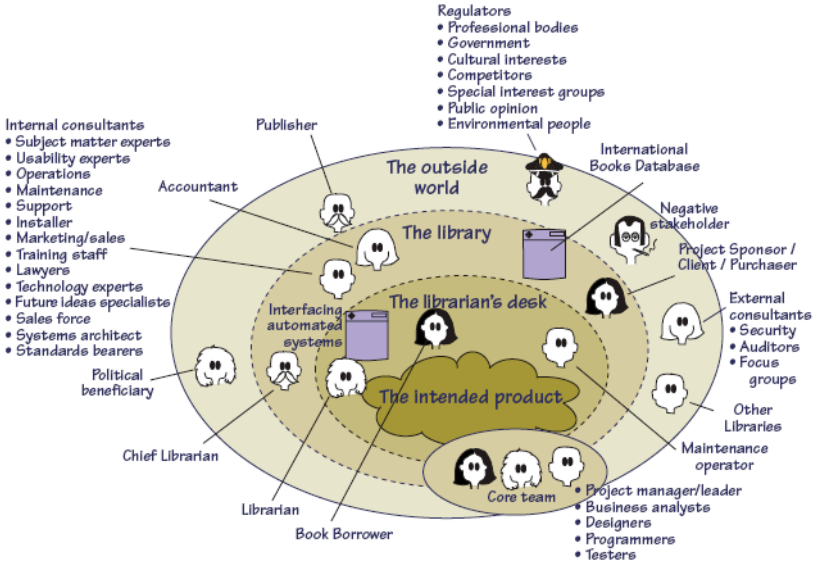


Figure 1: A Stakeholder Map for the Library Loans project

Stakeholder Map

Suzanne Robertson
& James Robertson

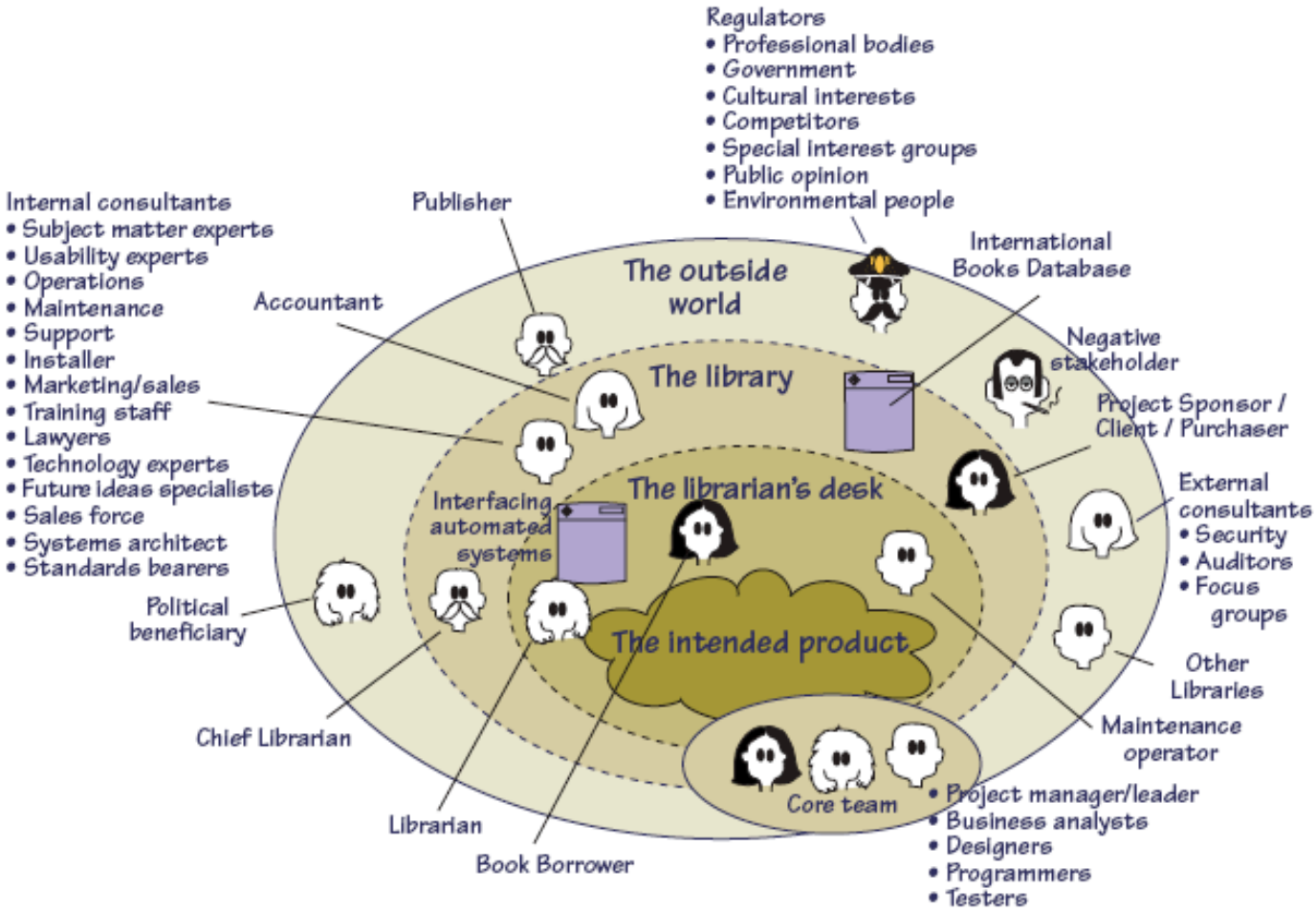


Figure 1: A Stakeholder Map for the Library Loans project

http://www.requirementsnetwork.com/sites/requirementsnetwork.com/files/Volere_Requirements-A_Socio_Technical_Discipline.pdf

3. System Engineers

- can analyze
  - necessary, and desirable, quality levels;
  - and plan, and manage to deliver,
    - a set of those quality levels,
    - within given constraints,
      - and available resources.

Example of an Impact Estimation table

Requirements \ Designs	An Organic Apple	An Organic Orange
- Performance Eater Acceptance From 50% to 80% of People	70%	85% (80%)
Pesticide Measurement Reduce from 0.005% to 0.001%	100%	100%
Shelf-Life Increase from 1 week to 1 month	70%	200% (100%)
Vitamin C from fruit Increase from 50mg to 100mg per day	20%	140% (100%)
Carbohydrates from fruit Increase from 50g to 100g per day	40%	30%
Sum of Performance	300%	410%
- Costs Relative Cost (Local currency)	0.33	0.40
Performance to Cost Ratio or 'Benefit to Cost Ratio'	909	1025

Lindsey Brodie

•Figure 1: Real (NON-CONFIDENTIAL version) example of an initial draft of setting the objectives that engineering processes must meet.

Business objective	Measure	Goal (200X)	Stretch goal ('0X)	Volume	Value	Profit	Cash
Time to market	Normal project time from GT to GT5	<9 mo.	<6 mo.	X		X	X
Mid-range	Min BoM for The Corp phone	<$90	<$30	X		X	X
Platformisation Technology	# of Technology 66 Lic. shipping > 3M/yr	4	6	X		X	X
Interface	Interface units	>11M	>13M	X		X	X
Operator preference	Top-3 operators issue RFQ spec The Corp	1	2	X		X	X
Productivity							X
Get Torden	Lyn goes for Technology 66 in Sep-04	Yes		X		X	X
Fragmentation	Share of components modified	<10%	<5%		X	X	X
Commoditisation	Switching cost for a UI to another System	>1yr	>2yrs	X		X	X
Duplication	The Corp share of 'in scope' code in best-selling device	>90%	>95%		X	X	X
Competitiveness	Major feature comparison with MX	Same	Better	X		X	X
User experience	Key use cases superior vs. competition	5	10	X	X	X	X
Downstream cost saving	Project ROI for Licensees	>33%	>66%	X	X	X	X
Platformisation IFace	Number of shipping Lic.	33	55	X		X	X
Japan	Share of of XXXX sales	>50%	>60%	X		X	X
	Numbers are intentionally changed from real ones						

Business Values Quantified

Strategy Impact Estimation:
for a $100,000,000 Organizational Improvement Investment

Objectives

Technical Strategies

Strategy Impacts "Benefits" on Objectives

Defined In earlier slide

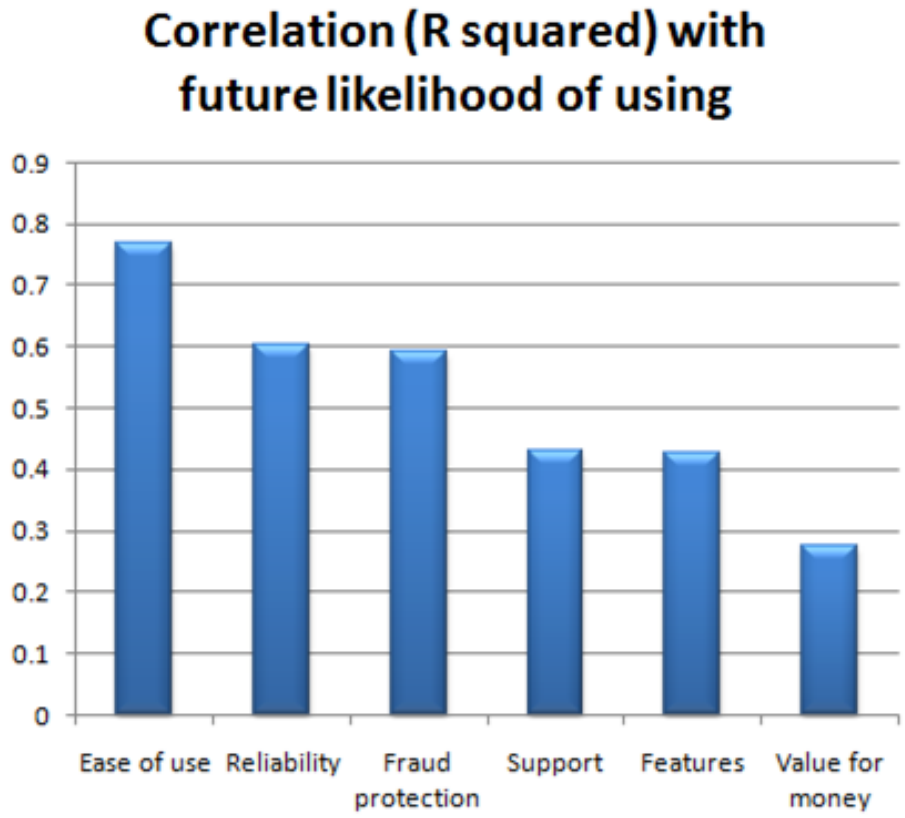
Business Objective	Viking Deliverables: hardware adaptation	Telephony	Reference designs	IFace	Modularity	Defend vs Technology 66	Tools	User Experience	GUI & Graphics	Security	Defend vs OCD	Enterprise
Time to market	20%	10%	30%	5%	10%	5%	15%	0%	0%	0%	5%	5%
Mid-range	15%	[illegible]	[illegible]	[illegible]	[illegible]	5%	5%	10%	5%	5%	0%	0%
Platformisation Technology	25%	10%	30%	0%	[illegible]	10%	0%	5%	0%	10%	0%	5%
Interface	5%	15%	15%	0%	5%	0%	5%	0%	0%	10%	0%	10%
Operator preference	0%	[illegible]	[illegible]	[illegible]	[illegible]	20%	5%	10%	10%	20%	5%	10%
Get Torden	25%	10%	10%	-10%	0%	20%	0%	10%	-20%	10%	10%	5%
Commoditisation	20%	10%	20%	10%	-20%	25%	15%	0%	0%	5%	10%	5%
Duplication	15%	0%	10%	0%	0%	40%	0%	0%	0%	5%	20%	5%
Competitiveness	10%	15%	20%	0%	10%	20%	10%	10%	20%	10%	10%	10%
User experience	5%	0%	0%	0%	[illegible]	0%	0%	30%	10%	0%	0%	0%
Downstream cost saving	15%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	10%	0%	0%	10%	5%
Platformisation IFace	10%	10%	20%	40%	0%	20%	5%	0%	0%	0%	0%	5%
Japan	10%	5%	20%	0%	10%	0%	0%	10%	5%	0%	0%	0%
Contribution to overall result	15%	9%	17%	4%	7%	15%	6%	6%	1%	6%	6%	5%
Cost (£M)	£ 2.85	£ 0.49	£ 3.21	£ 2.54	£ 1.92	£ 2.31	£ 0.81	£ 1.21	£ 2.68	£ 0.79	£ 0.62	£ 0.60
ROI Index (100=average)	106	358	109	33	78	137	148	107	10	152	202	174

Cost

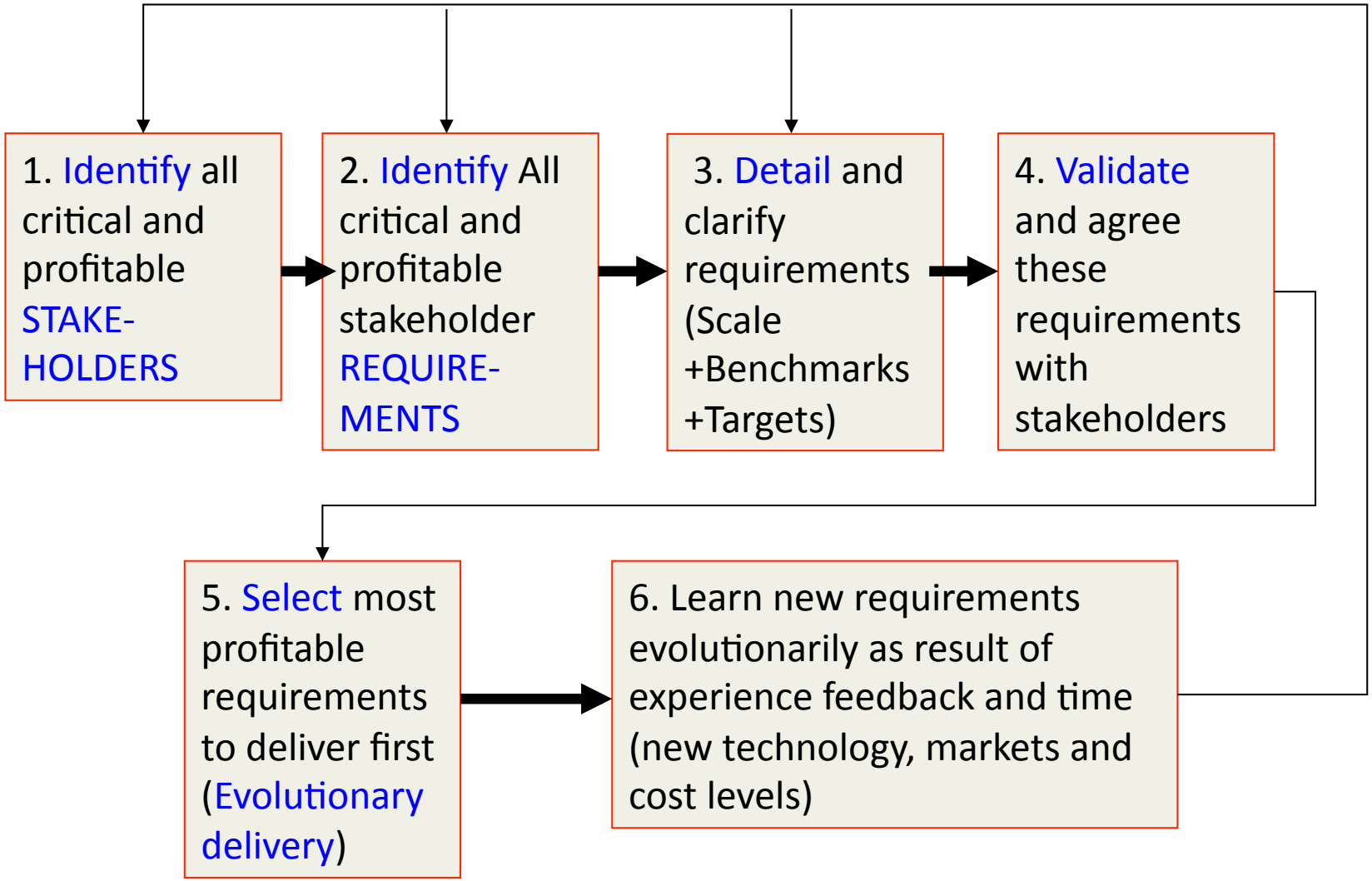
358 !

4. Quality Management

- is responsible for prioritizing the use of resources,
- to give a satisfactory fit,
- for the prioritized levels of quality:
- and for trying to manage the delivery of a set of qualities -
- that maximize value for cost - to defined stakeholders.



Stakeholders:
How to find out about, and confirm, their requirements
(a process of managing quality iteratively)



My Quality Principles (overview)

Quality Principles Heuristics for Action: An Overview.

1. Quality Design: Ambitious Quality Levels are designed in, not tested in. This applies to work processes and work products.
2. Software Environment: "Software" Quality is totally dependent on its resident system quality, and does not exist alone; 'software qualities' are dependent on a defined system's qualities – including stakeholder perceptions and values.
3. Quality Entropy: Existing or planned quality levels will deteriorate in time, under the pressure of other prioritized requirements, and through lack of persistent attention.
4. Quality Management: Quality levels can be systematically managed to support a given quality policy. Example : "Value for money first", or "Most competitive World Class Quality Levels".
5. Quality Engineering: A set of quality levels can be technically engineered, to meet stakeholder ambitions, within defined constraints, and priorities.
6. Quality Perception: Quality is in the eyes of the beholder: objective system quality levels may be simultaneously valued as great for some stakeholders, and terrible for others.
7. Design Impact on Quality: any system design component, whatever its intent, will likely have unpredictable main effects, and side effects, on many other quality levels, many constraints, and many resources.
8. Real Design Impacts: you cannot be sure of the totality of effects, of a design for quality, on a system, except by measuring them in practice; and even then, you cannot be sure the measure is general, or will persist.
9. Design Independence: Quality levels can be measured, and specified, independently of the means (or designs) needed to achieve them
10. Complex Qualities: many qualities are best defined as a subjective, but useful, set of elementary quality dimensions; this depends on the degree of control you want over the separate quality dimensions.[1]

1. **Quality Design** Principle

- *Ambitious* Quality Levels are **designed in**, not tested in.
- *This applies to work processes and work products.*

Current Status	Improvements		Survey Engine .NET		
Units	Units	%	Past	Tolerable	Goal
			Backwards.Compatibility (%)		
83,0	48,0	80,0	40	85	95
0,0	67,0	100,0	67	0	0
			Generate.WI.Time (small/medium/large seconds)		
4,0	59,0	100,0	63	8	4
10,0	397,0	100,0	407	100	10
94,0	2290,0	103,9	2384	500	180
			Testability (%)		
10,0	10,0	13,3	0	100	100
			Usability.Speed (seconds/user rating 1-10)		
774,0	507,0	51,7	1281	600	300
5,0	3,0	60,0	2	5	7
			Runtime.ResourceUsage.Memory		
0,0	0,0	0,0		?	?
			Runtime.ResourceUsage.CPU		
3,0	35,0	97,2	38	3	2
			Runtime.ResourceUsage.MemoryLeak		
0,0	800,0	100,0	800	0	0
			Runtime.Concurrency (number of users)		
1350,0	1100,0	146,7	150	500	1000
			Development resources		
64,0			0		84

Testers Note: Testing cannot deliver quality!

EVO Plan Confirmit 8.5

4 product areas were attacked in all: 25 Qualities concurrently, one quarter of a year. Total development staff = 13

9

8

3

3

Impact Estimation Table: Reportal codename "Hyggen"

Current Status	Improvements		Reportal - E-SAT features		
Units	Units	%	Past	Tolerable	Goal
			Usability.Intuitivness (%)		
75,0	25,0	62,5	50	75	90
			Usability.Consistency.Visual (Elements)		
14,0	14,0	100,0	0	11	14
			Usability.Consistency.Interaction (Components)		
15,0	15,0	107,1	0	11	14
			Usability.Productivity (minutes)		
5,0	75,0	96,2	80	5	2
5,0	45,0	95,7	50	5	1
			Usability.Flexibility.OfflineReport.ExportFormats		
3,0	2,0	66,7	1	3	4
			Usability.Robustness (errors)		
1,0	22,0	95,7	7	1	0
			Usability.Replacability (nr of features)		
4,0	5,0	100,0	8	5	3
			Usability.ResponseTime.ExportRep t (min es		
1,0	12,0	150,0	13	13	5
			Usability.ResponseTime.ViewRepo seco s)		
1,0	14,0	100,0	15		1
			Development resources		
203,0			0		91

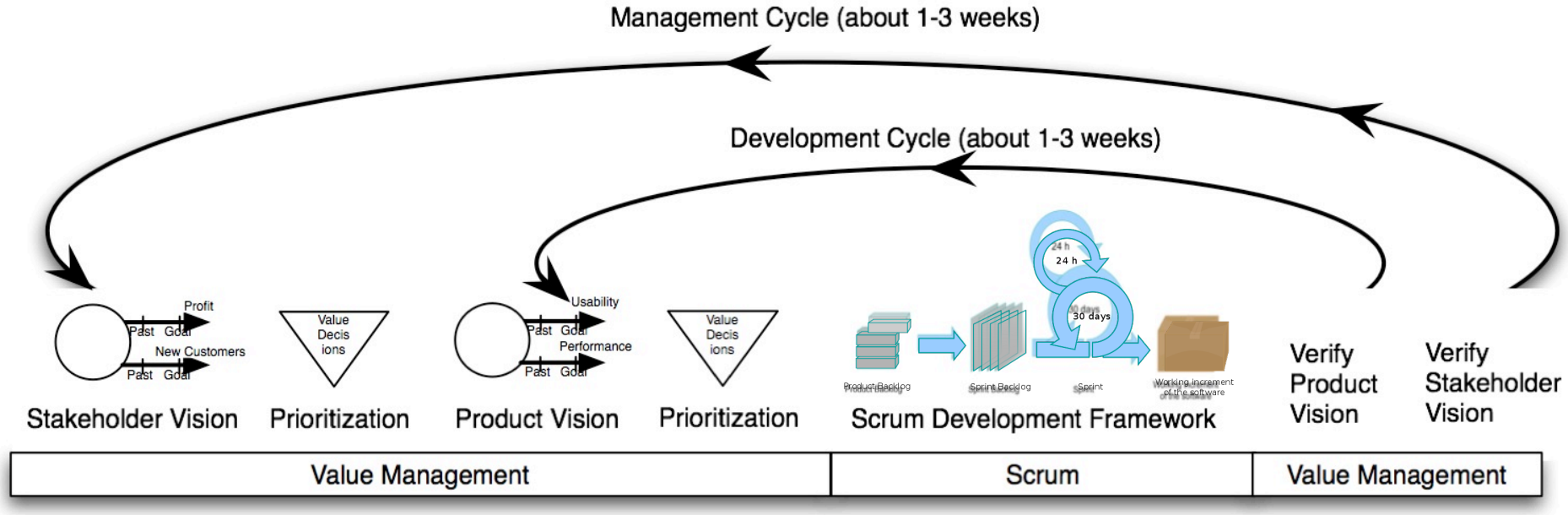
Current Status	Improvements		Survey Engine .NET		
Units	Units	%	Past	Tolerable	Goal
			Backwards.Compatibility (%)		
83,0	48,0	80,0	40	85	95
0,0	67,0	100,0	67	0	0
			Generate.WI.Time (small/medium/large seconds)		
4,0	59,0	100,0	63	8	4
10,0	397,0	100,0	407	100	10
94,0	2290,0	103,9	2384	500	180
			Testability (%)		
10,0	10,0	13,3	0	100	100
			Usability.Speed (seconds/user rating 1-10)		
774,0	507,0	51,7	1281	600	300
5,0	3,0	60,0	2	5	7
			Runtime.ResourceUsage.Memory		
0,0	0,0	0,0		?	?
			Runtime.ResourceUsage.CPU		
3,0	35	97,2	38	3	2
			Runtime.ResourceUsage.MemoryLeak		
0	800	100,0	800	0	0
			Runtime.Concurrency (number of users)		
350	1100	146,7	150	500	1000
			Development resources		
64			0		84

Current Status	Improvements		Reportal - MR Features		
Units	Units	%	Past	Tolerable	Goal
			Usability.Replacability (feature count)		
1,0	1,0	50,0	14	13	12
			Usability.Productivity (minutes)		
20,0	45,0	112,5	65	35	25
			Usability.ClientAcceptance (features count)		
4,4	4,4	36,7	0	4	12
			Development resources		
101,0			0		86

Current Status	Improvements		XML Web Services		
Units	Units	%	Past	Tolerable	Goal
			TransferDefinition.Usability.Efficiency		
7,0	9,0	81,8	16	10	5
17,0	8,0	53,3	25	15	10
			TransferDefinition.Usability.Response		
943,0	-186,0	######	170	60	30
			TransferDefinition.Usability.Intuitiveness		
5,0	10,0	95,2	15	7,5	4,5
			Development resources		
2,0			0		48

Trond Johansen

Value Management



15

Value Decision Tables

	Stakeholder Value 1	Stakeholder Value 2
Business Value 1	-10%	40%
Business Value 2	50%	10%
Resources	20%	10%

	Product Value 1	Find.Fast
Stakeholder Value 1	-10%	50 %
Stakeholder Value 2	10 %	10%
Resources	2 %	5 %

	Solution 1	Service Guide
Find.Fast	-10%	40%
Product Value 2	50%	80 %
Resources	1 %	2 %

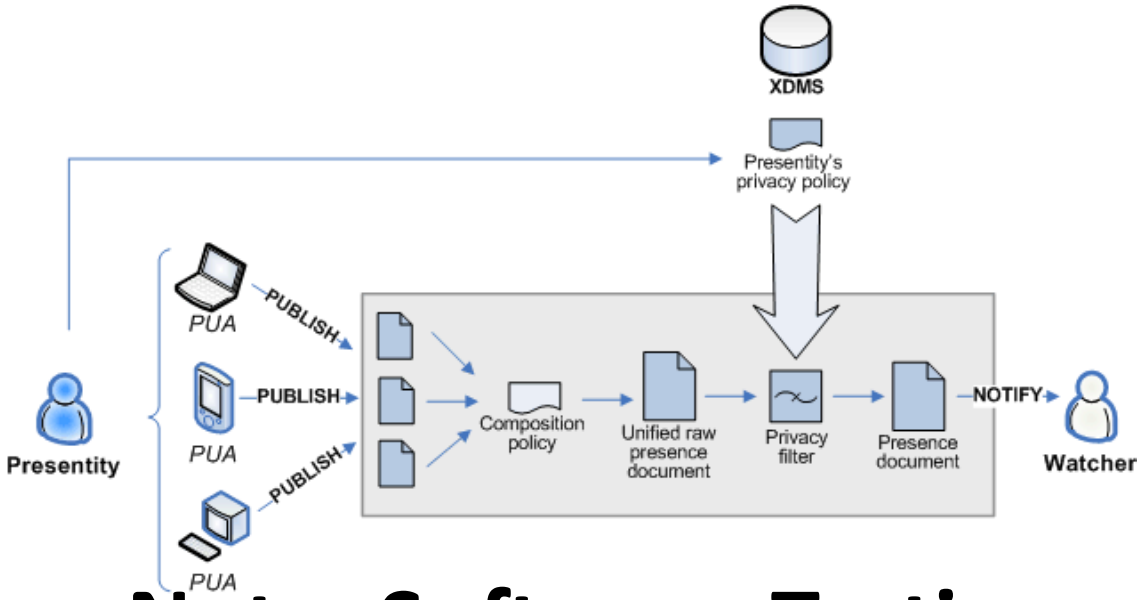
Prioritized List
1. Service Guide
2. Solution 9
3. Solution 7

Scrum Develop
We measure improvements
Learn and Repeat

16

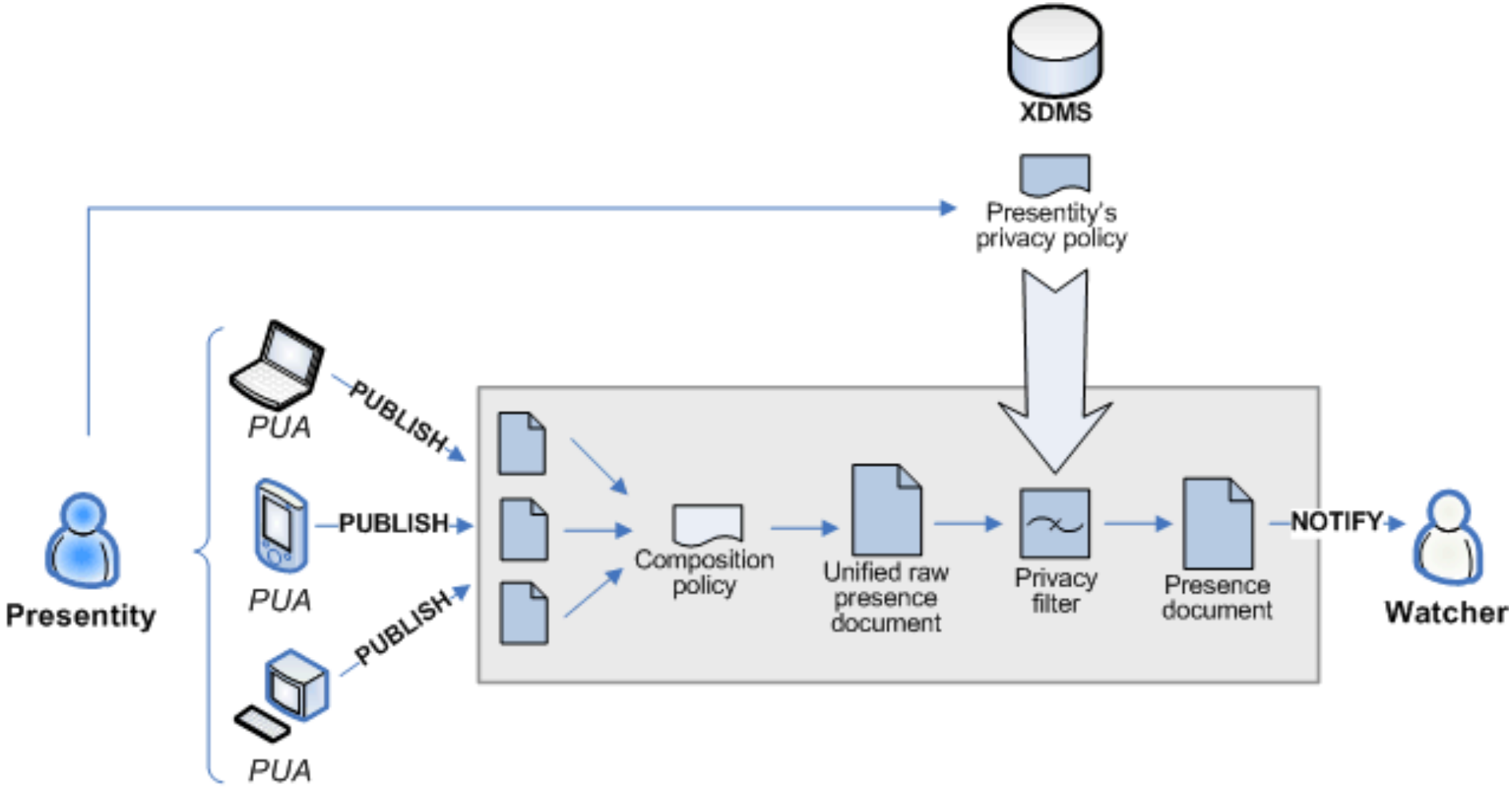
2. **Software Environment** Principle

- "Software" Quality is
  - *totally dependent on its resident system* quality,
    - and does not exist alone;
  - '***software*** qualities' are dependent
  - on a defined **system's** qualities – (think, hardware)
    - including stakeholder perceptions and values



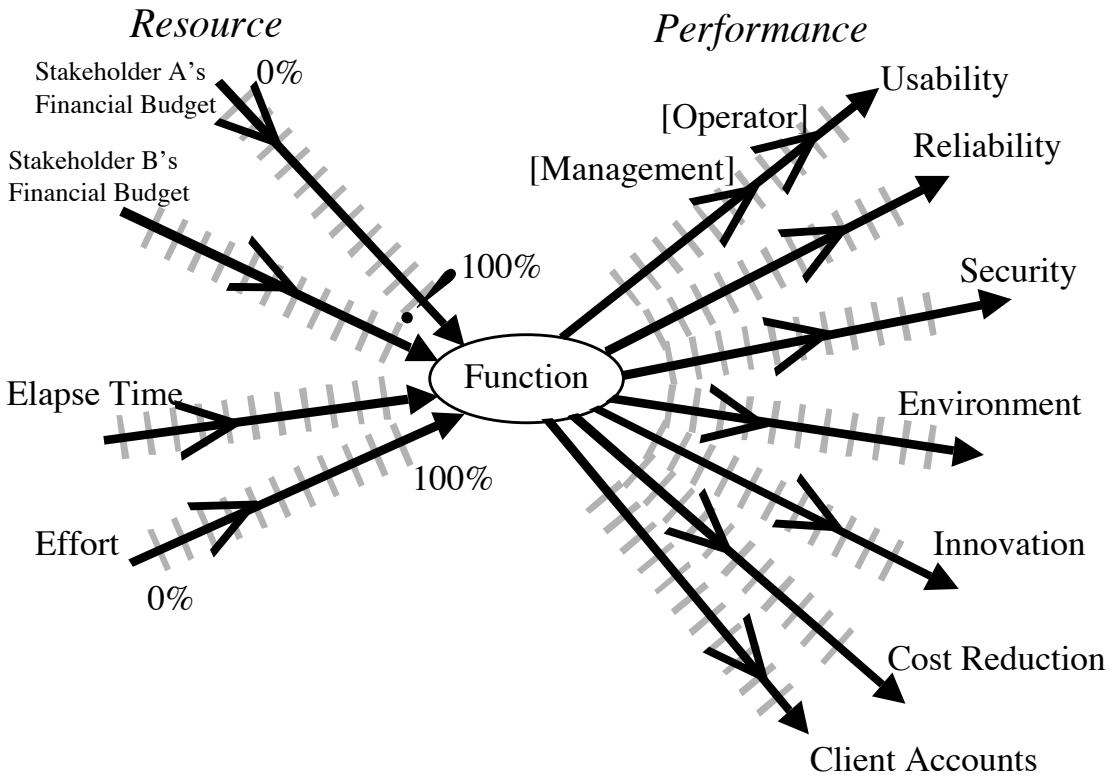
Testers Note: Software Testing cannot measure *system* qualities

Component Quality depends on Related Components



3. **Quality Entropy** Principle

- Existing or planned quality levels will
  - *deteriorate* in time,
  - under the pressure of other prioritized requirements,
  - and through lack of persistent attention



Testers Note: Software Testing cannot measure future long term qualities, they change

Engineering “Maintainability”: Green Week
Weekly ‘Refactoring’ at Confirmit

Current Status		Improvement	Goals			Step 6 (week 14)		Step 7 (week 15)	
	Units		Past	Tolerable	Goal	Estimated Impact	Actual Impact	Estimated Impact	Actual Impact
	100,0	100,0	0	80	100			100	100
Speed									
	100,0	100,0	0	80	100	100	100		
Maintainability.Doc.Code									
	100,0	100,0	0	80	100	100	100		
InterviewerConsole									
NUnitTests									
	0,0	0,0	0	90	100				
PeerTests									
	100,0	100,0	0	90	100			100	100
FxCop									
	0,0	10,0	10	0	0				
TestDirectorTests									
	100,0	100,0	0	90	100			100	100
Robustness.Correctness									
	2,0	2,0	0	1	2	2	2		
Robustness.BoundaryConditions									
	0,0	0,0	0	80	100				
Speed									
	0,0	0,0	0	80	100				
ResourceUsage.CPU									
	100,0	0,0	100	80	70	70			
Maintainability.Doc.Code									
	100,0	100,0	0	80	100	100	100		
SynchronizationStatus									
NUnitTests									

Speed

Maintainability

Nunit Tests

PeerTests

TestDirectorTests

Robustness.Correctness

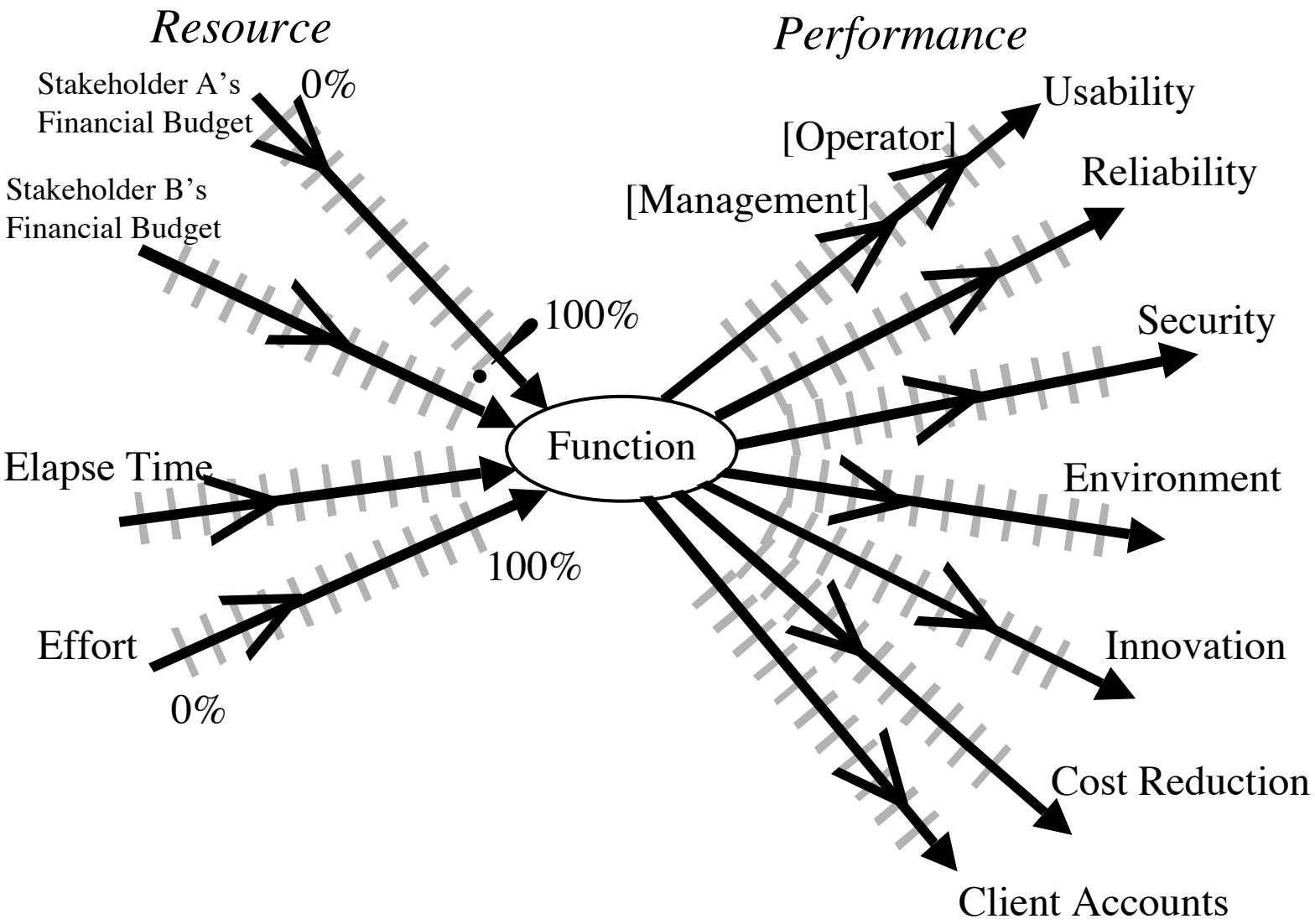
Robustness.Boundary Conditions

ResourceUsage.CPU

Maintainability.DocCode

SynchronizationStatus

Multiple Required Performance and Cost Attributes are the basis for architecture selection and evaluation



4. **Quality Management** Principle

- Quality levels can be systematically managed
  - to support a given quality policy.
- *Example :*
  - *"Value for money first", or*
  - *"Most competitive World Class Quality Levels".*

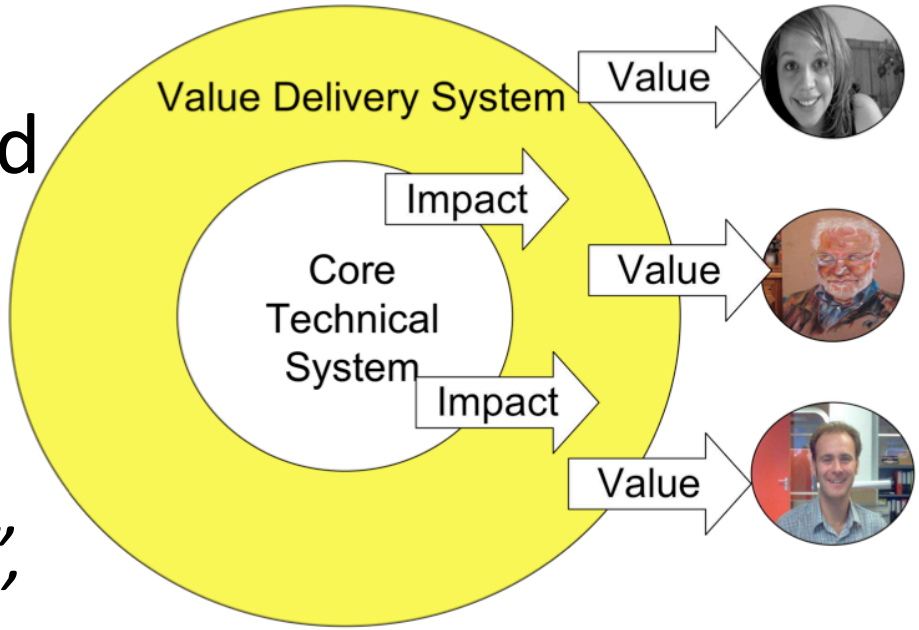
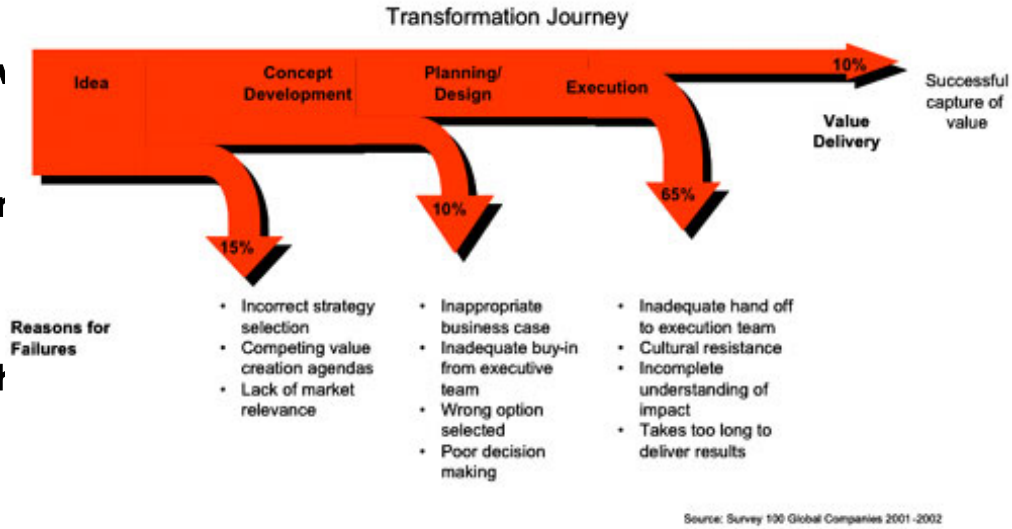


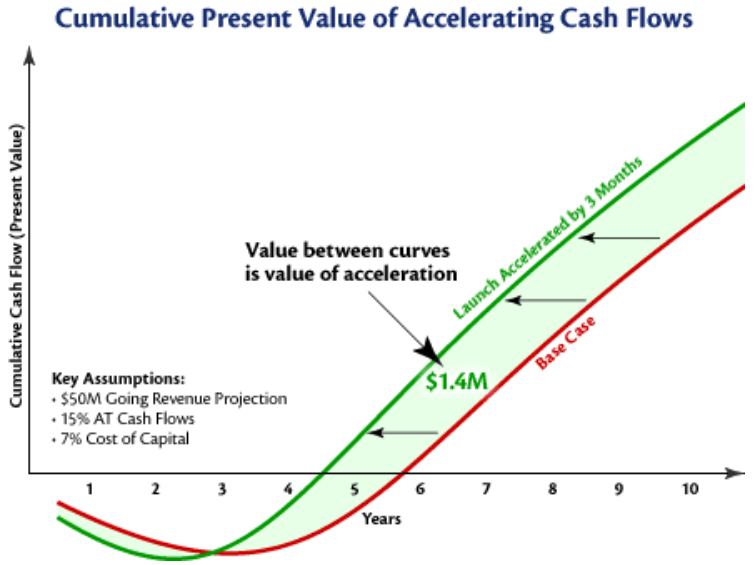
Figure 1: The Value Delivery System: some level of systems engineering has to take responsibility for final delivery of expected value to stakeholders.

Testers Note: Software Testing cannot
Manage multiple quality/performance/cost levels

Gilb's Value Manifesto: A Management Policy?

1. **Really useful value, for real stakeholders w be defined measurably.**
   No nice-sounding emotive words please.
2. **Value will be seen in light of total long terr costs**
   as a decent return on investment.
3. **Powerful management devices, like motivation and follow-up, will make sure th the value for money is really delivered –**
   or that the failure is punished, and the success is rewarded.
4. **The value will be delivered evolutionarily –**
   not all at the end.
5. **That is, we will create a stream of prioritized value delivery to stakeholders, at the *beginning* of our value delivery projects;**
   and continue as long as the real return on investment is suitably large.
6. **The CEO is primarily responsible for making all this happen effectively.**
   1. **The CFO will be charged with tracking all value to cost progress.**
   2. **The CTO and CIO will be charged with formulating all their efforts in terms of measurable value for resources.**





Source "Value Delivery in Systems Engineering" available at www.gilb.com
Unpublished paper http://www.gilb.com/tiki-download_file.php?fileId=137

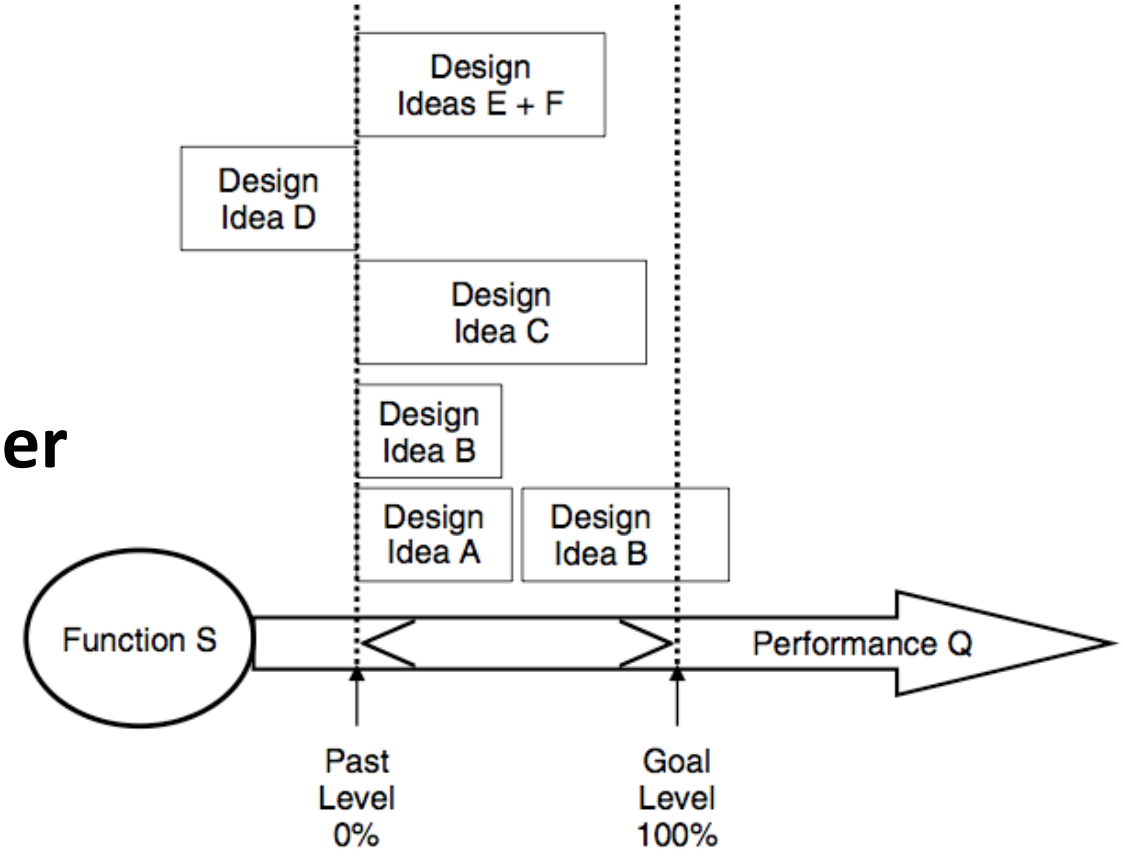
The Value Principles:

1. **Value can always be articulated quantitatively, so that we can understand it, agree to it, track it, contract for it and understand it in relation to costs.**
2. **Value is a result, delivered to a real set of stakeholders.**
3. **Value must be seen in light of lifetime total cost aspects, and must be as profitable as alternative investments.**
4. **Value occurs through time, as a stakeholder experience: it is not delivered when a system to enable it is delivered – only when that system is successfully used to extract the value.**
5. **Value can be delivered early, and for part of one stakeholder's domain. This proves the value potential, and actually improves the real organization.**
6. **There is never a really sufficient reason to put off value delivery until large-scale long-term investments are made. This is just a common excuse from the many weak, ignorant, cowards who would like to spend a lot of money before being held to account.**
7. **People who cannot deliver a little value early, in practice, cannot be entrusted to deliver a lot of value for a larger investment.**
8. **The top management must be primarily responsible for making value delivery happen in their organization. The specialist managers will never in practice take the responsibility, unless they are aiming to take over the top job.**
9. **Value is a multiplicity of improvements, and certainly not all related to money or savings – but we still need to quantify the value proposition in order to understand it, and manage it.**
10. **If we prioritize highest value for money first, then we should normally experience an immediate and continuous flow of dramatic results, that the entire organization can value and relate to. Be deeply suspicious of long-term visions with no short-term proof.**

http://www.gilb.com/tiki-download_file.php?fileId=137
"Value Delivery in Systems Engineering"

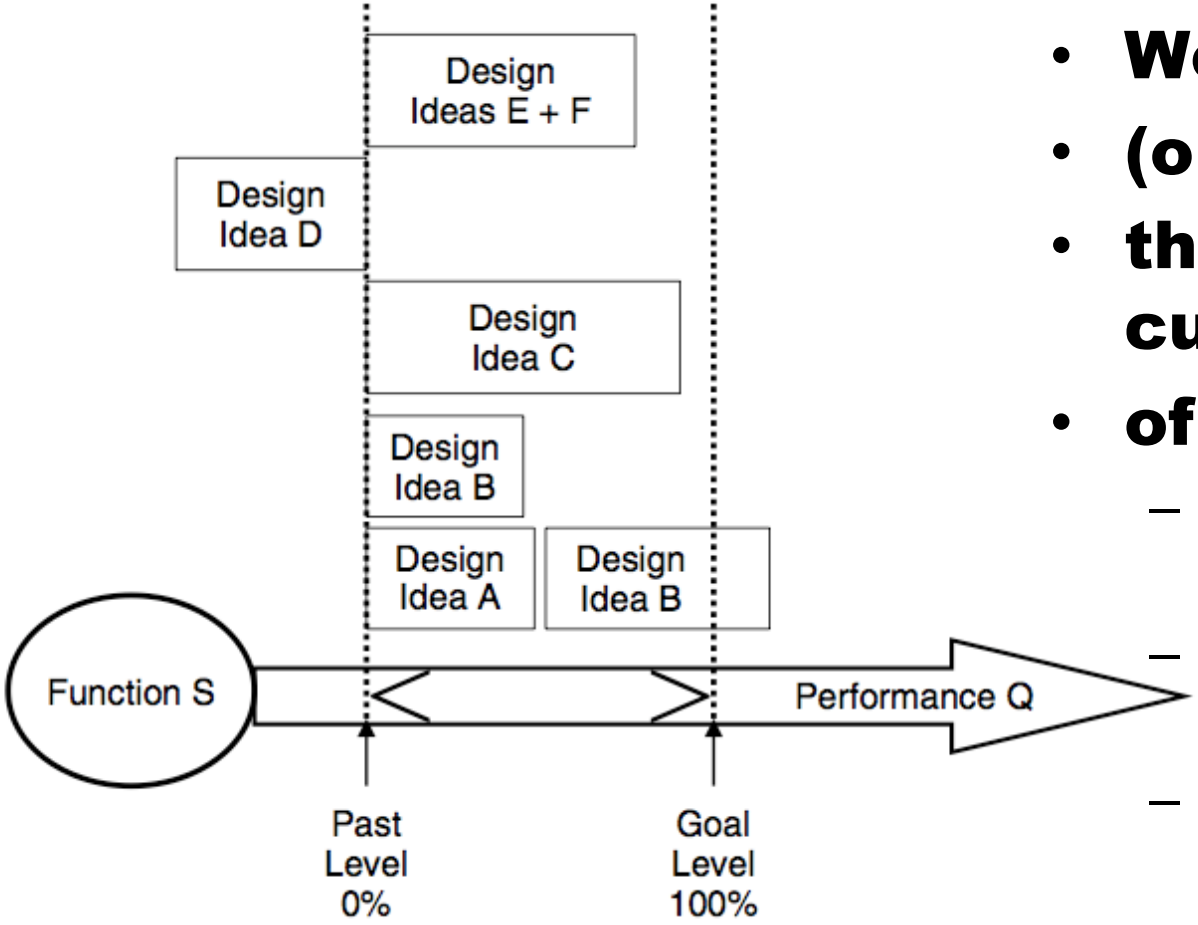
5. **Quality Engineering** Principle

- **A *set* of quality levels can be technically engineered,**
  - **to meet stakeholder ambitions,**
  - **within defined constraints, and priorities.**



Testing Note: testing cannot find and evaluate designs to deliver quality levels

How do we evaluate, for a single quality dimension, the impact of a design ?



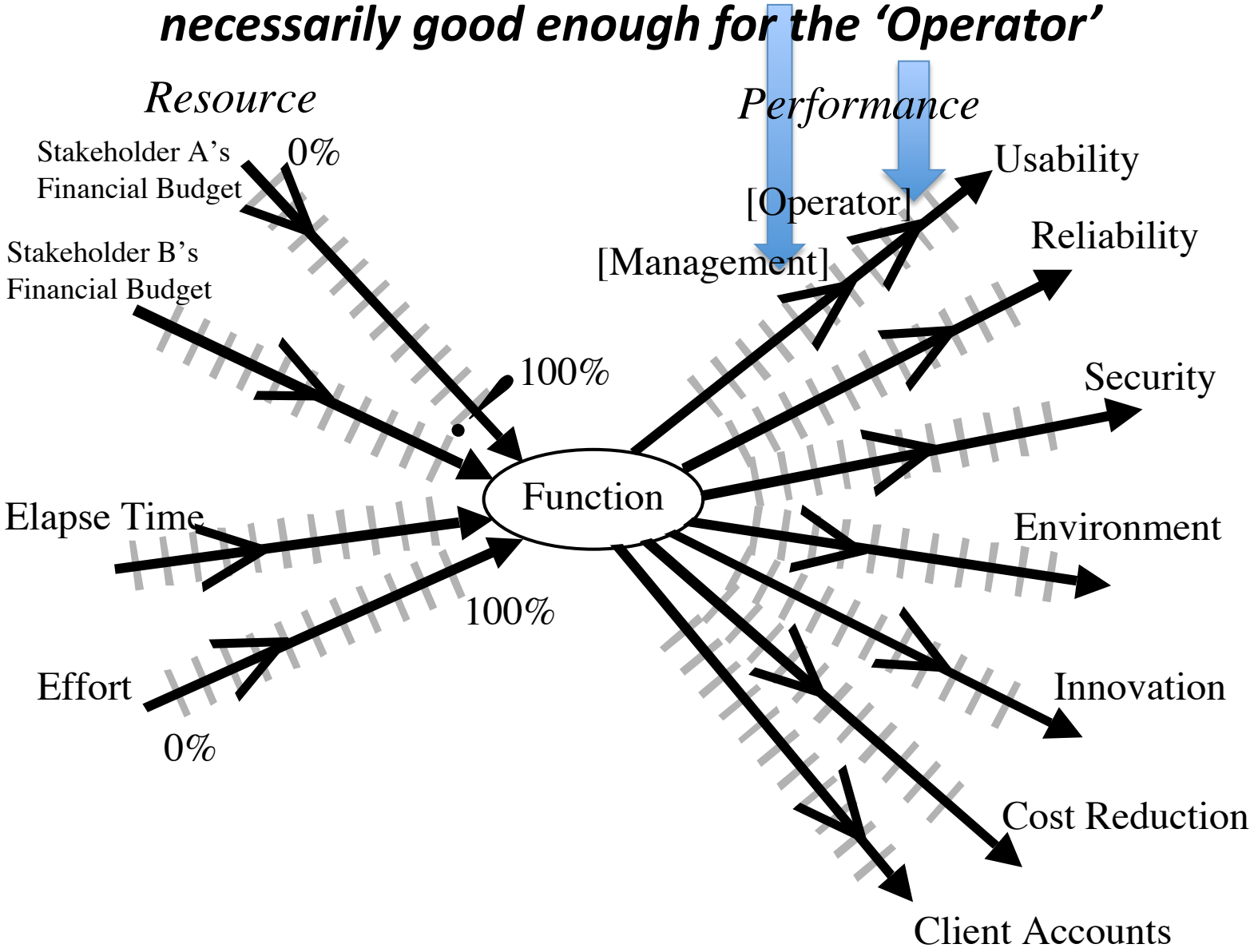
- We must estimate
- (or measure)
- the numeric cumulative impact
- of the design
  - on a defined Scale (units),
  - using a defined Meter (test process),
  - with respect to requirement levels.

6. **Quality Perception** Principle

- **Quality is in the eyes of the beholder:**
  - **objective system quality levels may be simultaneously valued as**
    - ***great*** **for some stakeholders,**
    - **and *terrible* for others.**

Testing Note: testing cannot decide on right quality levels for stakeholders

Usability level good enough for 'Management' is not necessarily good enough for the 'Operator'



7. **Design Impact on Quality** Principle

- **any system design component,**
  - **whatever its *intent*,**
  - **will likely have *unpredictable* main effects,**
    - **and side effects,**
    - **on *many* other quality levels,**
    - **many constraints, and**
    - **many resources.**

Design Ideas ->	*Technology Investment*	*Business Practices*	*People*	*Empowerment*	*Principles of IMA Management*	*Business Process Re-engineering*	*Sum Requirements*
Customer Service ? <->0 Violation of agreement	50%	10%	5%	5%	5%	60%	185%
Availability 90% <-> 99.5% Up time	50%	5%	5–10%	0%	0%	200%	265%
Usability 200 <-> 60 Requests by Users	50%	5–10%	5–10%	50%	0%	10%	130%
Responsiveness 70% <-> ECP's on time	50%	10%	90%	25%	5%	50%	180%
Productivity	45%	60%	10%	35%	100%	53%	303%
3:1 Return on Investment	50%	5%	75%	45%	15%	61%	251%
Morale 72 <-> 60 per month on Sick Leave							
Data Integrity 88% <-> 97% Data Error %	42%	10%	25%	5%	70%	25%	177%
Technology Adaptability 75% Adapt Technology	5%	30%	5%	60%	0%	60%	160%
Requirement Adaptability ? <-> 2.6% Adapt to Change	80%	20%	60%	75%	20%	5%	260%
Resource Adaptability 2.1M <-> ? Resource Change	10%	80%	5%	50%	50%	75%	270%
Cost Reduction FADS <-> 30% Total Funding	50%	40%	10%	40%	50%	50%	240%
Sum of Performance	*482%*	*280%*	*305%*	*390%*	*315%*	*649%*	
Money % of total budget	15%	4%	3%	4%	6%	4%	36%
Time % total work months/year	15%	15%	20%	10%	20%	18%	98%
Sum of Costs	*30*	*19*	*23*	*14*	*26*	*22*	
Performance to Cost Ratio	*16:1*	*14:7*	*13:3*	*27:9*	*12:1*	*29:5*	

Testing Note: testing can give us measurable feedback on current incremental levels of quality

DoDef. Persinscom Impact Estimation Table:

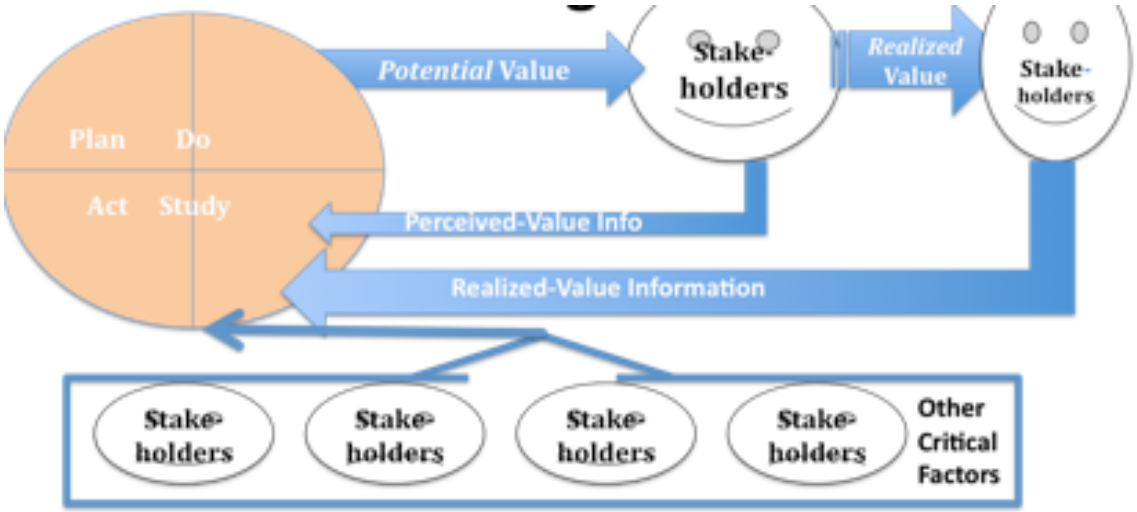
Designs

Design Ideas ->	*Technology Investment*	*Business Practices*	*People*	*Empowerment*	*Principles of IMA Management*	*Business Process Re-engineering*	*Sum Requirements*
Requirements	50%	10%	5%	5%	5%	60%	185%
Availability 90% <-> 99.5% Up time	50%	5%	5–10%	0%	0%	200%	265%
Usability 200 <-> 60 Requests by Users	50%	5–10%	5–10%	50%	0%	10%	130%
Responsiveness 70% <-> ECP's on time	50%	10%	90%	25%	5%	50%	180%
Productivity	45%				100%	53%	303%
3:1 Return on Investment	50%				15%	61%	251%
Morale 72 <-> 60 per month on Sick Leave							
Data Integrity 88% <-> 97% Data Error %	42%	10%	25%	5%	70%	25%	177%
Technology Adaptability 75% Adapt Technology	5%	30%	5%	60%	0%	60%	160%
Requirement Adaptability ? <-> 2.6% Adapt to Change	80%	20%	60%	75%	20%	5%	260%
Resource Adaptability 2.1M <-> ? Resource Change	10%	80%	5%	50%	50%	75%	270%
Cost Reduction FADS <-> 30% Total Funding	50%	40%	10%	40%	50%	50%	240%
Sum of Performance	*482%*	*280%*	*305%*	*390%*	*315%*	*649%*	
Money % of total budget	15%	4%	3%	4%	6%	4%	36%
Time % total work months/year	15%	15%	20%	10%	20%	18%	98%
Sum of Costs	*30*	*19*	*23*	*14*	*26*	*22*	
Performance to Cost Ratio	*16:1*	*14:7*	*13:3*	*27:9*	*12:1*	*29:5*	

R→ D Impacts

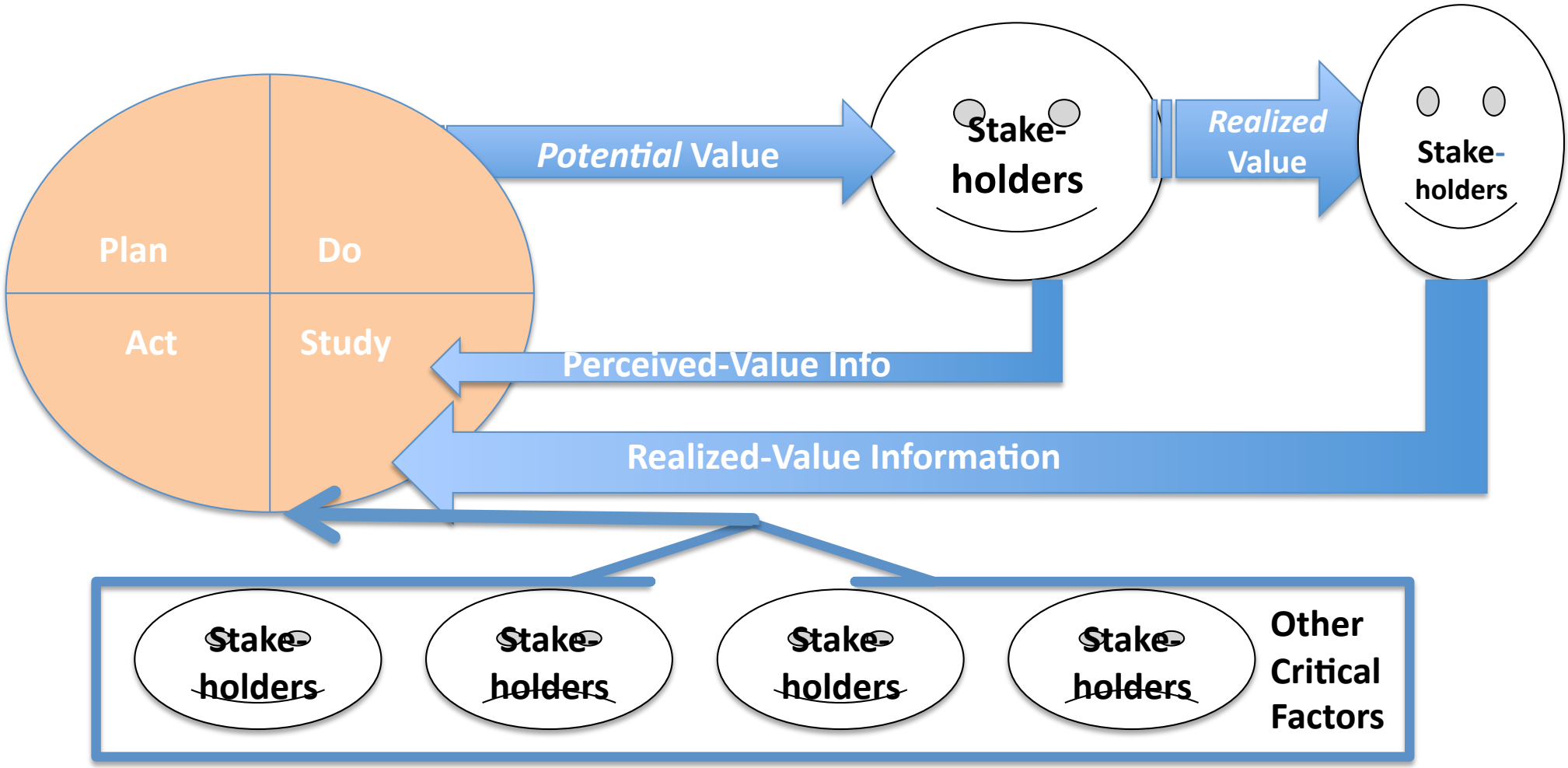
8. Real Design Impacts Principle: (tricky) (so you need frequent feedback!)

- you cannot be *sure* of the totality of effects, of a design for quality,
  - on a system,
    - except by measuring them in *practice*;
    - and even then, you cannot be sure the measure is *general*,
    - or will *persist*.



Test Note: Quality Tests must be frequent, to discover problems early

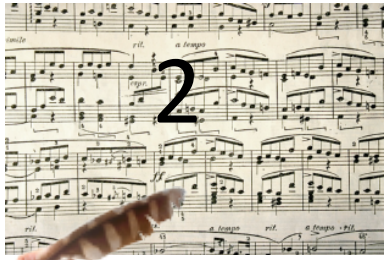
Real Relevant Measurement Frequently for Multiple Critical Factors,
Is the only reliable way to measure and control many simultaneous effects of incremental design implementation



- 2 Kinds of Feedback from Stakeholders, when value increment is *really* exploited in practice after delivery.
- Combined with other information from the relevant environment. Like budget, deadline, technology, politics, laws, marketing changes.

The Simplest and Best Agile Project Method; 'Evo'

- **Process Description**
  - 1. Gather from all the key stakeholders the top few (5 to 20) most critical goals that the project needs to deliver.
    - Give each goal a reference name (a tag).
  - 2. For each goal, define a scale of measure and a 'final' goal level.
    - For example: *Reliable: Scale: Mean Time Before Failure, Goal: 1 month.*



3. Define approximately 4 budgets for your most limited resources
   - (for example, time, people, money, and equipment).
4. Write up these plans for the goals and budgets
   - (*Try to ensure this is kept to only one page*).

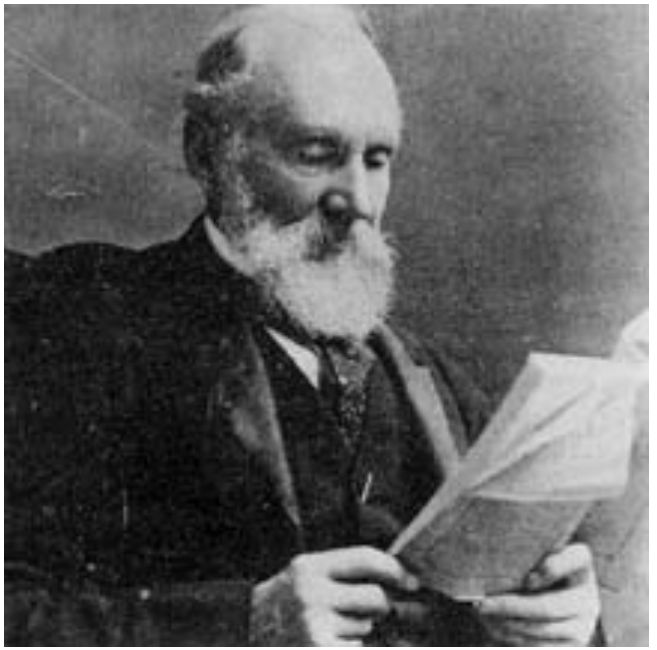
  - 5. Negotiate with the key stakeholders to formally agree the goals and budgets.
  - 6. Plan to deliver some benefit
    - (that is, progress towards the goals)
    - in *weekly* (or shorter) increments (Evo steps).

  - 7. Implement the project in Evo steps.
    - Report to project sponsors after each Evo step (weekly, or shorter) with your best available estimates or measures, for each performance goal and each resource budget.
    - *On a single page*, summarize the *progress to date* towards achieving the goals and the costs incurred.

  - 8. When all Goals are reached: 'Claim success and move on'
    - a. Free remaining resources for more profitable ventures

9. **Design Independence** Principle

- **Quality levels can be**
  - ***measured*,**
  - **and *specified*,**
  - **independently of the means (or designs) needed to achieve them**



What *does* the Lord say about specification and measurement?

Test Note: testers of critical qualities must expect numeric specifications to test against.

THE PRINCIPLE OF 'QUALITY QUANTIFICATION'

All qualities can be expressed quantitatively,
***'qualitative'* does *not* mean unmeasurable.**

"In physical science the first essential step in the direction of learning any subject is to find principles of numerical reckoning and practicable methods for measuring some quality connected with it.

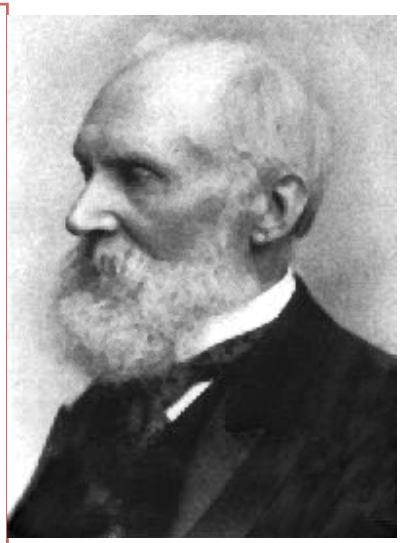
I often say that when you can measure what you are speaking about, and express it in numbers, you know something about it;

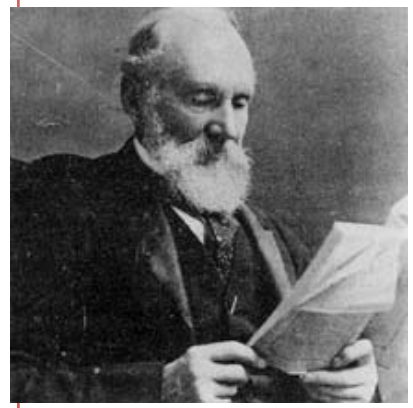
but when you cannot measure it, when you cannot express it in numbers, your knowledge is of a meagre and unsatisfactory kind;

it may be the beginning of knowledge, but you have scarcely in your thoughts advanced to the state of Science, whatever the matter may be."

Lord Kelvin, 1893

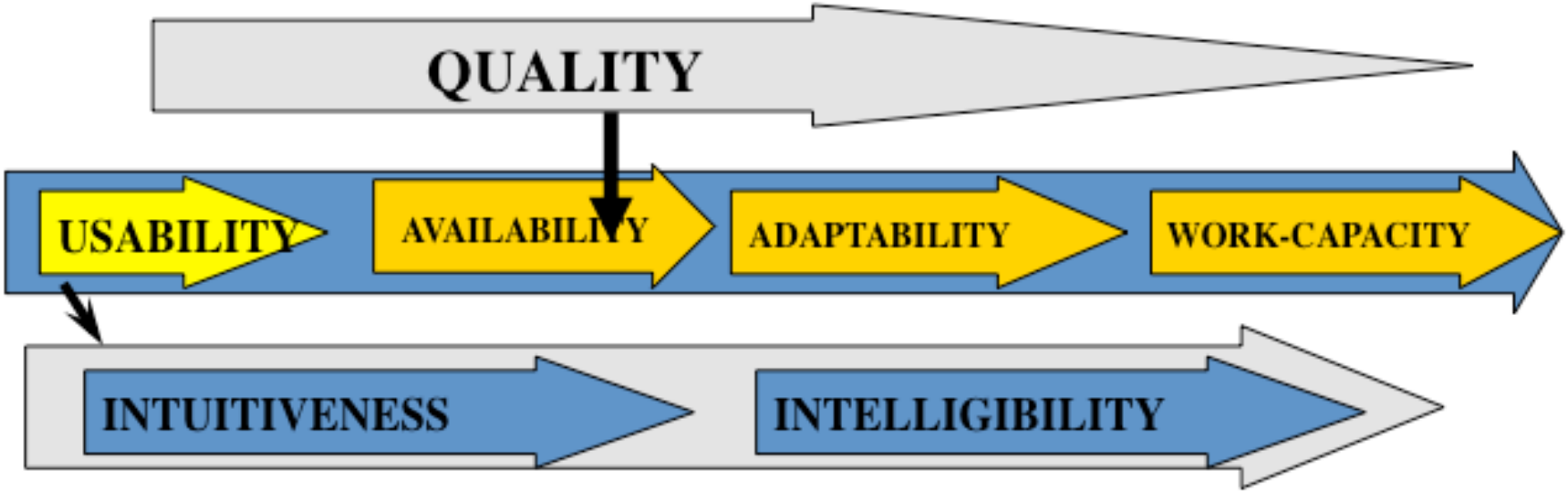
From http://zapatopi.net/kelvin/quotes.html





10. **Complex Qualities** Principle

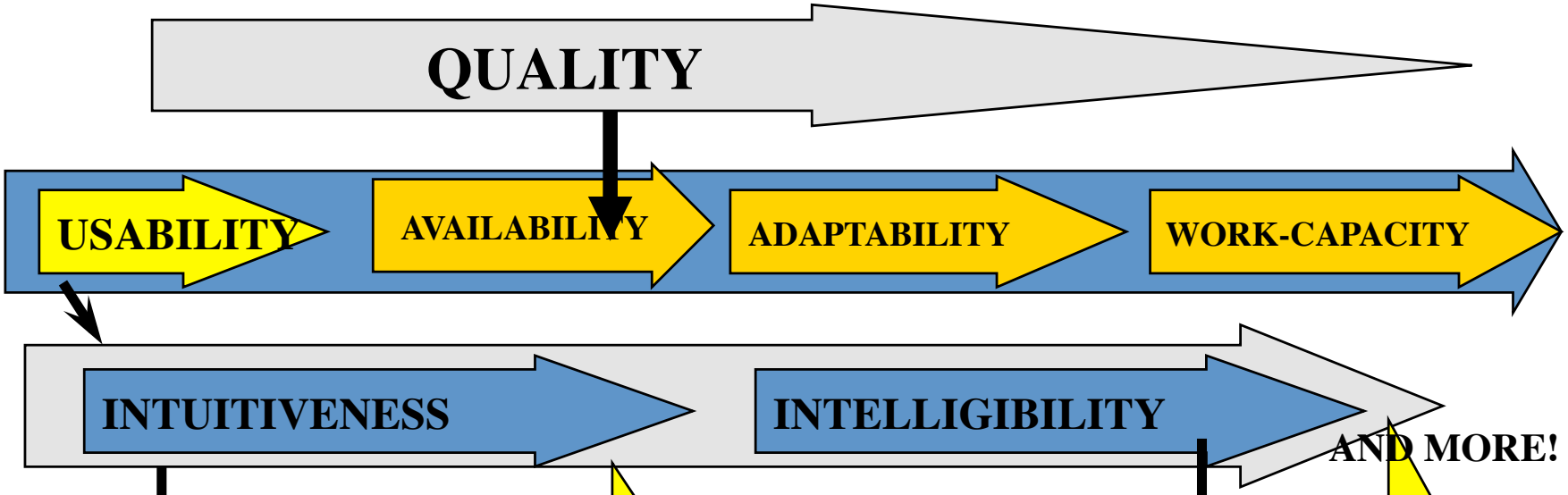
- many qualities are best defined as a *subjective*, but **useful**, **set of** elementary quality dimensions;
- this depends on the degree of control you want over the *separate* quality dimensions.
- CE Chapter 5, download, 'Scales of Measure' http://www.gilb.com/community/tiki-download_file.php?fileId=26 will give rich illustration to this point. See for example Maintainability, Adaptability and Usability.



Testers Note: every one of the defined sub-dimensions should be testable economically in practice frequently

Quantifying 'Usability' (Erieye C&C System)

SIMPLIFIED PLANNING LANGUAGE: 'PLANGUAGE'

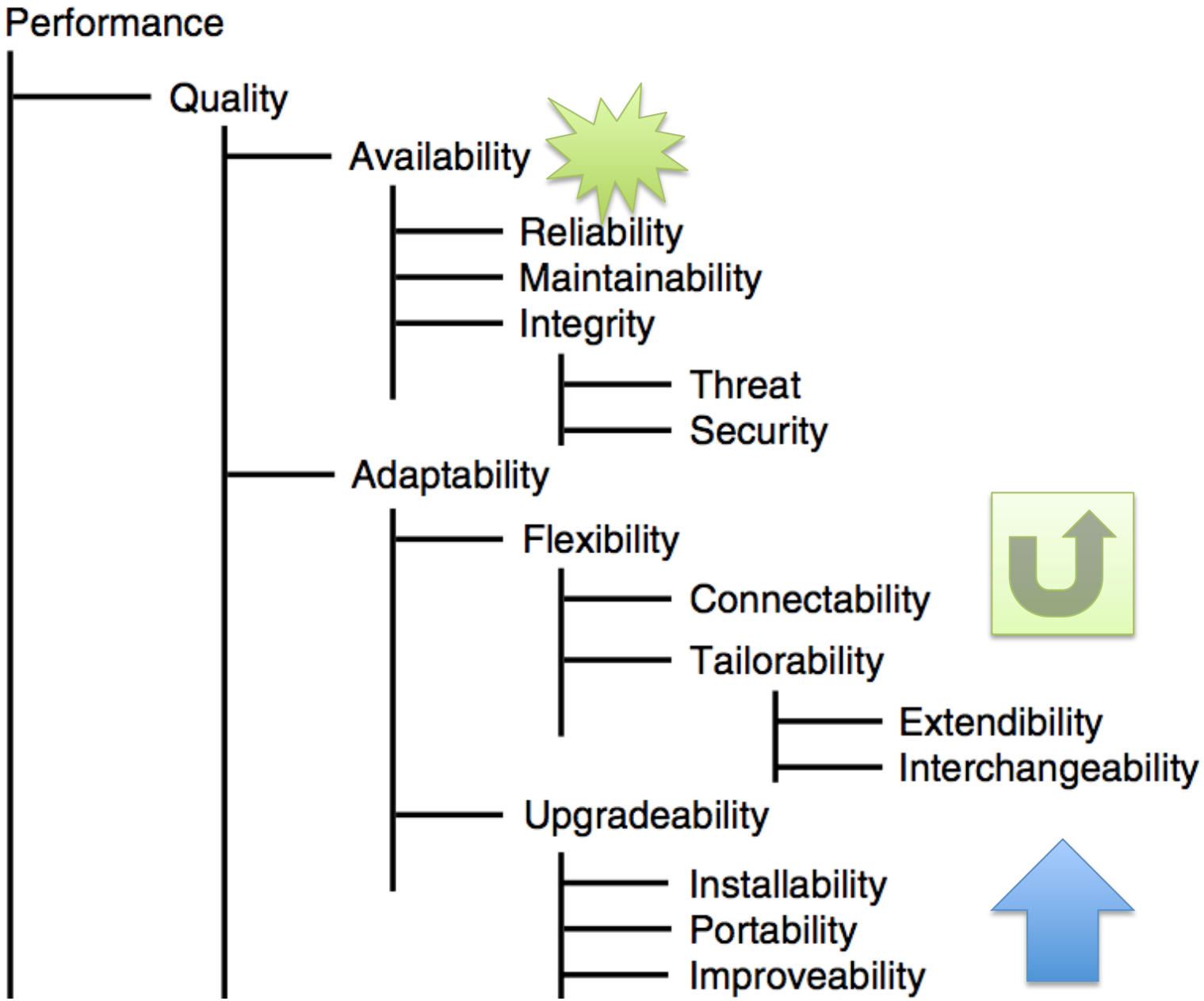


Intuitiveness
Ambition: Great intuitive capability
SCALE: Probability that intuitive guess right.
METER: <100 observations.>
PAST **[GRAPES]** 80% <-LN
RECORD **[MAC]** 9%?<-TG
Fail **[TRAINED, RARE]** 50-90%
Goal **[TASKS]** 99% <-LN

Intelligibility
Ambition: Super ease of immediate understanding
SCALE:% OK interpretations.
METER: 10 ops., 100 infos, 15 mins.
P:PAST[20 ops., 300 info, 30 min.]99%
RECORD **[P]** 99.0%
Fail **[DELIVERY[1]]**99.0%<-MAB
[ACCEPTANCE] 99.5%
Goal **[M1]** 99.9% <-LN

TRAINED: DEFINED:C&Ctl. operator, approved course, 200 hours duration.
RARE: DEFINED: types of tasks performed less than once a week per op.
TASKS: DEFINED: onboard operator distinct tasks carried out.
ACCEPTANCE: DEFINED: formal acceptance testing via customer contract.
DELIVERY: DEFINED: Evolutionary delivery cycle, integrated and useful.

A Generic Set of Performance measures – including several related to 'change'



References

- 2008 Test Experience Paper on Real QA http://www.gilb.com/tiki-download_file.php?fileId=288

The main idea with this paper is to wake up software engineers, and maybe some systems engineers, about quality. The software engineers (sorry, 'softcrafters') seem to think there is only one type of quality (lack of bugs), and only one place where bugs are found (in programs). My main point here is that the quality question is much broader in scope. The only way to get total necessary quality in software, is to treat the problem like a mature systems engineer. That means to recognize all critically interesting types of quality for your system. It means to take an architecture and engineering approach to delivering necessary quality. It means to stop being so computer program-centric, and to realize that even in the software world, there a lot more design domains than programs. And the software world is intimately entwined with the people and hardware world, and cannot simply try to solve their quality problems in splendid isolation. I offer some principles to bring out these points.

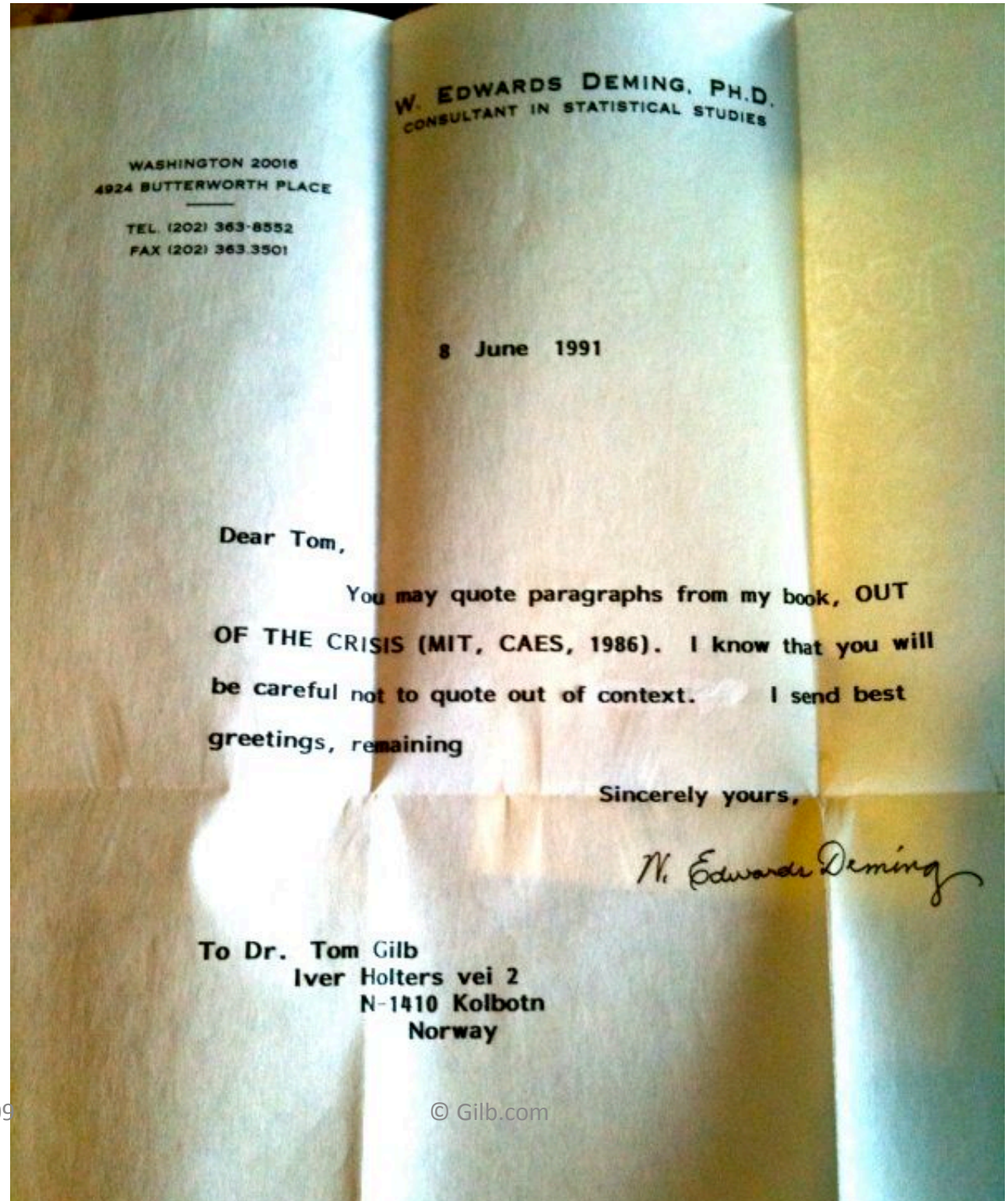
W. Edwards Deming

(as I knew him)

1984, London

Out Of Crisis , p. 417

- Any rule, if it is to be practicable, must be simple in administration.



W. EDWARDS DEMING, PH.D.
CONSULTANT IN STATISTICAL STUDIES

WASHINGTON 20016
4924 BUTTERWORTH PLACE

TEL. (202) 363-8552
FAX (202) 363.3501

8 June 1991

Dear Tom,

You may quote paragraphs from my book, OUT OF THE CRISIS (MIT, CAES, 1986). I know that you will be careful not to quote out of context. I send best greetings, remaining

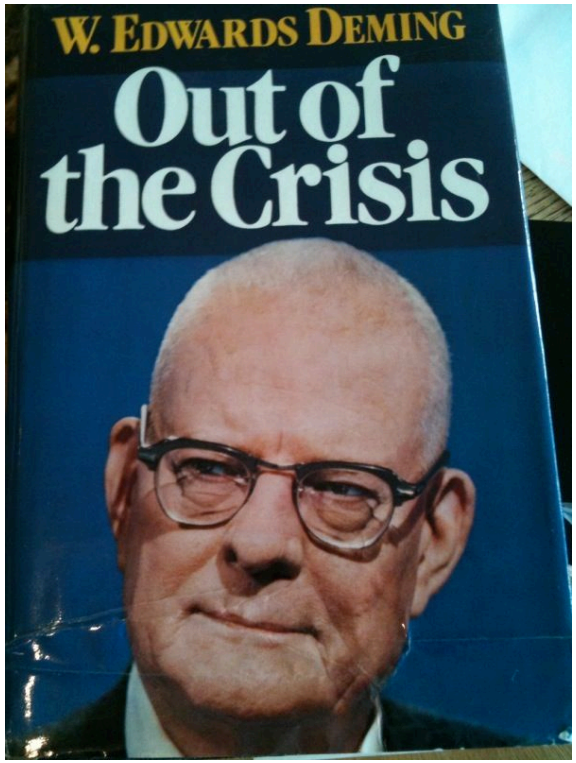
Sincerely yours,

W. Edwards Deming

To Dr. Tom Gilb
Iver Holters vei 2
N-1410 Kolbotn
Norway

Deming's Wisdom

Pages 28-29

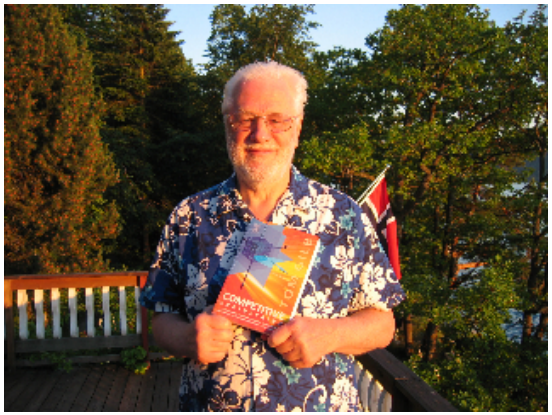


Inspect Early, Don't Test

- <Testing> is …. unreliable, costly, ineffective.
- It I important to carry out <QC> at the right point for minimum cost.
- You cannot <QC> quality into a product <-Harold F. Dodge

Details


COMPETITIVE
ENGINEERING
TOM GILB
COMPETITIVE
ENGINEERING
A HANDBOOK FOR SYSTEMS ENGINEERING, REQUIREMENTS ENGINEERING, AND SOFTWARE ENGINEERING USING PLANGUAGE
GILB



Tom Gilb

- Tom Gilb (born 1940, California) has lived in UK since 1956, and Norway since 1958.
- He is the author of 9 published books, including “Competitive Engineering: A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage”, 2005.
- He has taught and consulted world-wide for decades, including having direct corporate methods-change influence at major corporations such as Intel, HP, IBM, Nokia.
- He has had documented his founding influence in Agile Culture, especially with the key common idea of iterative development.
- He coined the term 'Software Metrics' with his 1976 book of that title.
- He is co-author with Dorothy Graham of the static testing method book 'Software Inspection' (1993).
- He is known for his stimulating and advanced presentations, and for consistently avoiding the oversimplified pop culture that regularly entices immature programmers to waste time and fail on their projects.
- More detail at www.Gilb.com

Last Slide!

- Questions: now, briefly
- After lecture
- By Email: tomsgilb at gmail.com

- Copy of these slides will be on SIGIST conference site
  - http://www.bcs.org/server.php?show=nav.9268
- And in Downloads: www.gilb.com
  - Library