Advanced Practices:
for Systems Engineering Project Management.
Based on
Value-and-Cost Metrics-Driven Architecture
and Value Delivery.

The Practice of The Planning Language 'Planguage' for Competitive Industrial Engineering

By Tom@Gilb.com. www.gilb.com
Author of ‘Competitive Engineering’ (2005)

MASTER 2016

Slides Version 1.2   2010

Summary

- **Quantify all values and qualities**
  - In requirements, project objectives
  - In architecture evaluation, presentation and reviews
- Focus on **Values/Costs** for **prioritization**
- Learn about complex systems **incrementally**
  - With respect to real value and real costs
- Planguage: is a rich language for systems engineering – and it models **quantified values and costs** – *unlike most other modelling options*
- Increase intelligibility of top level critical requirements by 10x, then 100x – measure by sampling, early, often

Advanced Practice = ?

- **Not commonly taught or used**
  - Quantification of ALL Quality dimensions, all ‘values’
  - Multidimensional evaluation of all designs and architectures
  - Incremental, small step (2%) measurement and learning about all qualities and costs
- But, is **actually practiced** in certain advanced communities to get superior *documented* results
  - Intel, Boeing, HP, Ericsson, Philips, IBM FSD, Raytheon, Confirmit (small example), Sony, Schlumberger

“Systems Engineering Project Management” =

- Proven delivery of primary system characteristics
- Within resource constraints
  - On time (early!), under budget

Toolkit

- **Quantification** of **all critical** stakeholder **variables**
  - Formal definition of Scale, Meter, Goal etc.
- **Estimation** of all critical impacts of all designs/**architectures**
- **Incremental** (Evolutionary) measurement of **value delivery** and costs
- **Prioritization** of **value/cost** (‘efficiency’) and ‘real’ delivery to stakeholders
- **Measurement** of Technical **Specifications**,
  - Exit Defect-level control

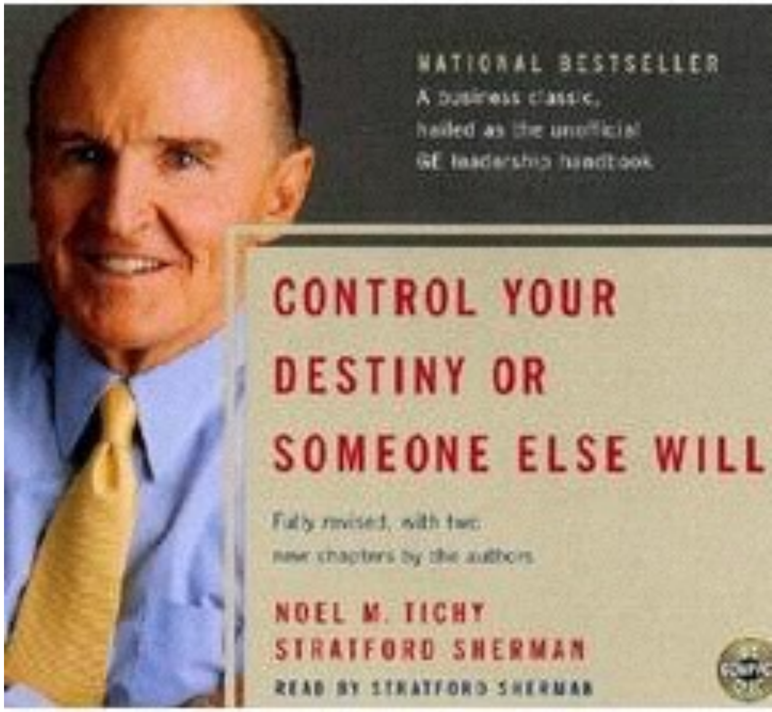
Toolkit: Quality Quantification

- Quantification of all critical stakeholder variables (including qualities and values)
  - Formal definition of Scale, Meter, Goal etc.

Few Clear Top Goals

- ***Instead of directing business according to detailed...strategic plan,***
  - ***[Jack] Welch [General Electric CEO]***
  - ***believed in setting only a few clear, overarching goals.***
  - ***Then, on an ad hoc basis,***
    - ***his people***
    - ***were free to seize any opportunities they saw to further those goals.***

Noel Tichy and Stratford Sherman, "Control Your Own Destiny or Someone Else Will"



Summary of Top '8' Project Objectives: Big Project

Real Example of ***Lack*** of Scales

*1. Central to The Corporations business strategy is to be the world's **premier** integrated_ <domain> service **provider**.*

*2. Will provide a much more efficient **user** experience*

*3. Dramatically scale back the **time** frequently needed after the last data is acquired to time align, depth correct, splice, merge, recompute and/or do whatever else is needed to **generate** the desired **products***

*4. Make the system much **easier** to **understand** and **use** than has been the case for previous system.*

*5. A primary goal is to provide a much more **productive** system **development** environment than was previously the case.*

*6. Will provide a richer set of functionality for **supporting** next-generation logging **tools** and applications.*

*7. **Robustness** is an essential system requirement (see rewrite in example below)*

*8. Major improvements in **data quality** over current practices*

This lack of clarity cost them $100,000,000 and 8 years lost project time

So, I offered to clarify objectives

- Over a beer

Rock Solid Robustness

<u>Rock Solid Robustness</u>:

Type: Complex Product Quality Requirement.

Includes: { Software Downtime, Restore Speed, Testability, Fault Prevention Capability, Fault Isolation Capability, Fault Analysis Capability, Hardware Debugging Capability}.

Software Downtime:

Software Downtime:
Type: Software Quality Requirement.
Ambition: *to have minimal downtime due to software failures <- HFA 6.1*

***Issue**: does this not imply that there is a system wide downtime requirement?*

Scale: <mean time between forced restarts for defined [Activity], for a defined [Intensity].>

Fail [Any Release or Evo Step, Activity = Recompute, Intensity = Peak Level] **14 days** <- HFA 6.1.1

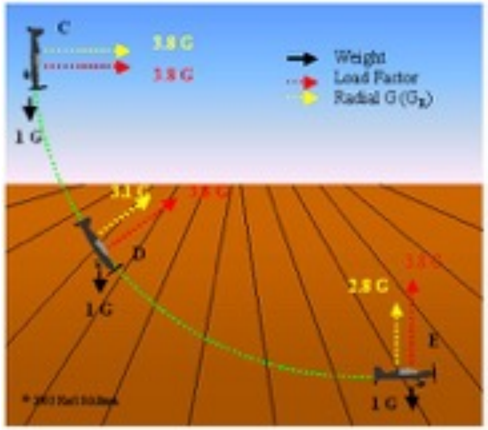
Goal [By 2008?, Activity = Data Acquisition, Intensity = Lowest level] **: 300 days** ??

Stretch: **600 days**

Restore Speed:

<u>Restore Speed:</u>
Type: Software Quality Requirement.

Ambition: Should an error occur (or the user otherwise desire to do so), Horizon shall be able to restore the system to a previously saved state in less than 10 minutes. <-6.1.2 HFA.



Scale: Duration from Initiation of Restore to Complete and verified state of a defined [Previous: Default = Immediately Previous]] saved state.

<u>Initiation</u>: defined as {Operator Initiation, System Initiation, ?}. Default = Any.

Goal [Initial and all subsequent released and Evo steps] **1 minute?**

Fail [Initial and all subsequent released and Evo steps] 10 minutes. <- 6.1.2 HFA

Catastrophe: 100 minutes.

Testability:

Testability:
Type: Software Quality Requirement.
Version: 20 Oct 2006-10-20
Status: Demo draft,
Stakeholder: {Operator, Tester}.
Ambition: Rapid-duration automatic testing of <critical complex tests>, with extreme operator setup and initiation.

Scale: the duration of a defined [Volume] of testing, or a defined [Type], by a defined [Skill Level] of system operator, under defined [Operating Conditions].

GOAL [All Customer Use, Volume = 1,000,000 data items, Type = WireXXXX Vs DXX, Skill = First Time Novice, Operating Conditions = Field, {Sea Or Desert}**.** **<10 mins.**

Design Hypothesis: *Tool Simulators, Reverse Cracking Tool, Generation of simulated telemetry frames entirely in software, Application specific sophistication, for drilling – recorded mode simulation by playing back the dump file, Application test harness console <-6.2.1 HFA*

‘Define Value’ to ‘Deliver Value’

- This is a practical example of what you have to do to ***‘Deliver Value’ don’t fight entropy of old systems’***
  - Which yesterday’s speaker (Yannick Cras Technical Director SAP) recommended
- And which Florian Guillermet (Chief Programme Officer, SESAR, EEC) clearly showed us by
  - **quantifying the top 4 requirements for SESAR**

Estimation of all critical impacts of all designs/architectures: IMPACT ESTIMATION TABLES

- This is not essentially different from the
  - Multidimensional Design Analysis tables that Prof. Olivier de Weck (MIT) showed us at 11:25 today
- But there are some interesting differences ☺️

- http://esd.mit.edu/Faculty_Pages/deweck/deweck.htm

DoD Persinscom **Impact Estimation** Table:

Designs

Requirements

R→ D Impacts

Design Ideas ->	*Technology Investment*	*Business Practices*	*People*	*Empowerment*	*Principles of IMA Management*	*Business Process Re-engineering*	*Sum 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 3:1 Return on Investment	45%				100%	53%	303%
Morale 72 <-> 60 per month on Sick Leave	50%				15%	61%	251%
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*	

Impact Estimation Concepts

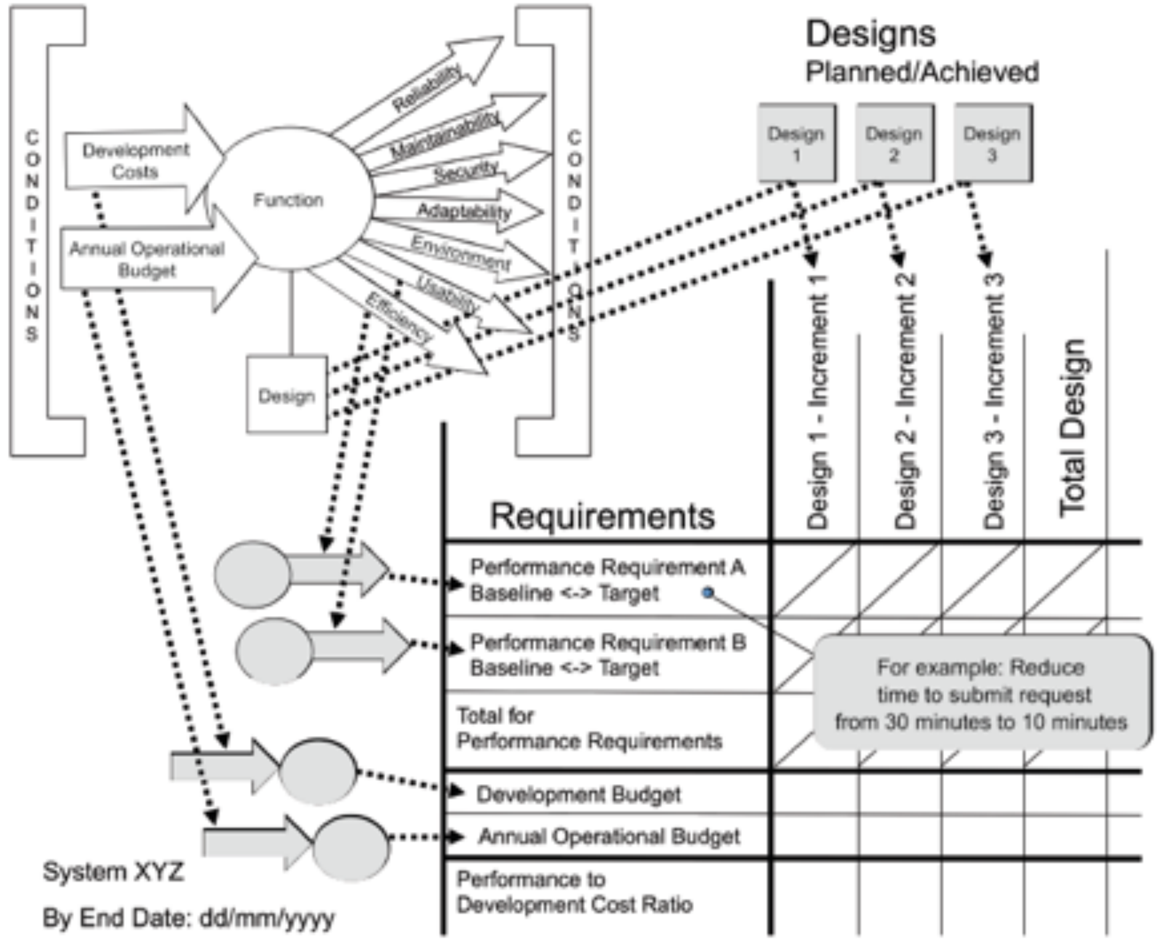


Fig. 3 Mapping of the system concepts to an IE table

SOURCE

Using Metrics within System Requirements to Express Quality and Derive Stakeholder Value

Lindsey Brodie • Mark Woodman

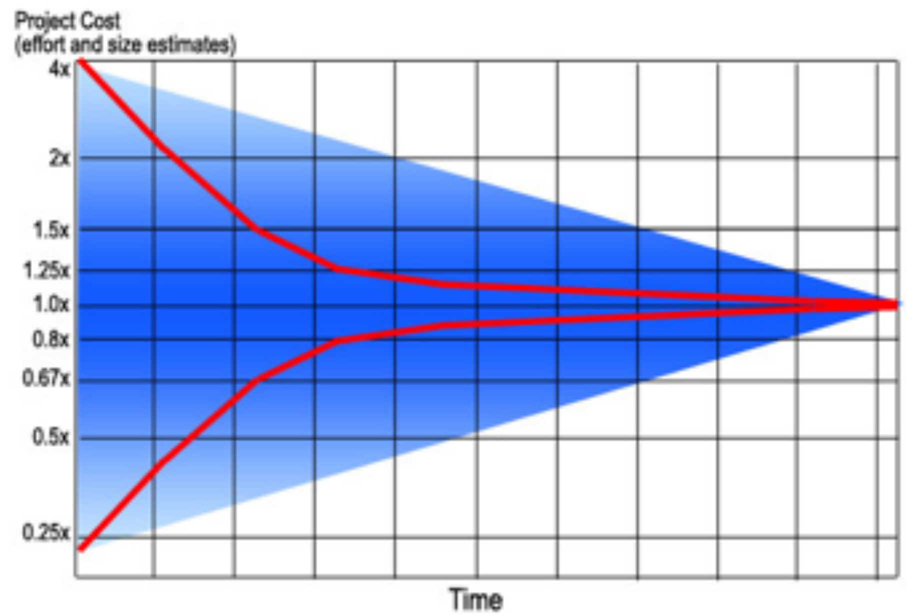
Lindsey is completing her PhD on this class of evaluation. lindseybrodie@btopenworld.com

Regarding
Systems Engineering Automation
(Bran Selic's Point this morning)

- Professor lech Krzanik, Oulu University, Finland
  - Lech.Krzanik@oulu.fi
- In about 1979 (used in his PhD)
  - Made the Aspect Engine, on Apple II in Forth
    - He says he can run the tool today
  - Stage 1
    - Automation of fuzzy requirements in to quantifi requirements
  - Stage 2
    - Quantified Requirements input and finding best fit design from a design option database

NEXT → **Incremental (Evolutionary) measurement of value delivery and costs**

- Maybe we **have** to learn about design effects from **real** systems, **NOT** just **models**?
- It is more ***realistic*** (better than 'models')
  - *The best model is the 'real thing'*
- And more informative (about side effects)



9th week Snapshot for 1 of 4 teams

Cumulative value delivered towards goals, in 9 increments

	A	B	C	D	E	F	G	BX	BY	BZ	CA
1											
2		Current Status	Improvements		Goals			Step9			
3								Recoding			
4								Estimated impact		Actual impact	
5		Units	Units	%	Past	Tolerable	Goal	Units	%	Units	%
6					Usability.Replacability (feature count)						
7		1,00	1,0	50,0	2	1	0				
8					Usability.Speed.NewFeaturesImpact (%)						
9		5,00	5,0	100,0	0	15	5				
10		10,00	10,0	200,0	0	15	5				
11		0,00	0,0	0,0	0	30	10				
12					Usability.Intuitiveness (%)						
13		0,00	0,0	0,0	0	60	80				
14					Usability.Productivity (minutes)						
15		20,00	45,0	112,5	65	35	25	20,00	50,00	38,00	95,00
20					Development resources						
21			101,0	91,8	0		110	4,00	3,64	4,00	3,64

confirmit everywhere

- Courtesy: 'Confirmit' Oslo Norway
- Case: http://www.gilb.com/tiki-download_file.php?fileId=32

Evo's impact on Confirmit product qualities

Product quality	Past	End state
Usability.Productivity: Time for the system to generate a defined complex survey	7200 s	15 s
Usability.Productivity: Time to set up a typical specified Market Research report	65 min	20 min
Usability.Productivity: Time to grant a set of end-users access to a report set and distribute report login info	80 min	5 min
Usability.Intuitiveness: The time it takes a medium experienced programmer to create a complete and correct data transfer definition with Confirmit web services without any user documentation or other aid	15 min	5 min
Performance.Runtime.Concurrency: Maximum number of simultaneously respondents executing a survey with a click rate of 20 seconds and a response time <500 ms given a defined [Survey complexity] and a defined [Server configuration, Typical]	250 users	6000 users

Prioritization of value/cost (‘efficiency’) and ‘real’ delivery to stakeholders

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

Business Objectives Quantified

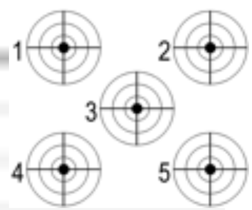
Business objective	Measure	Goal (200X)	Stretch goal ('0X)	Volume	Value	Profit	Cash
Time to market	Normal project time from GT to GT5	<9 mo.	<6 mo.	X		X	X
Mid-range	Min BoM for The Corp phone	<$9[illegible]	<$30	X		X	X
Platformisation Technology	# of Technology 66 Lic. shipping > 3M/yr	4[illegible]	[illegible]	[illegible]		[illegible]	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						[illegible]	[illegible]
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	>1y[illegible]	[illegible]rs	[illegible]	[illegible]	[illegible]	[illegible]
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

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

Technical Strategies

Objectives



	Viking Deliverables											
Business Objective	hardware adaptation	Telephony	Reference designs	Face	Modularity	Defend vs Technology 66	Tools	User Experie	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]	15%	0%	30%	15%	5%	10%	5%	5%	0%	0%
Platformisation Technology	25%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0%	5%	0%	10%	0%	5%
Interface	5%	15%	15%	0%	5%	0%	5%	0%	0%	10%	0%	10%
Operator preference	0%	[illegible]	[illegible]	15%	[illegible]	20%	5%	10%	10%	20%	5%	10%
Get Torden	25%	[illegible]	[illegible]	[illegible]	[illegible]	20%	0%	10%	-20%	10%	10%	5%
Commoditisation	20%	10%	20%	10%	-20%	25%	15%	0%	0%	5%	10%	5%
Duplication	15%	[illegible]	[illegible]	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%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	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%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	5%
Cost (£M)	£ 2.85	£ 0.49	£ 3.21	£ 2.54	£ 1.92	£ 2.31	£ 0.81	£ 1.21	£ 2.08	£ 0.79	£ 0.62	£ 0.60
ROI Index (100=average)	106	358	[illegible]	33	78	[illegible]	148	107	10	152	202	174

"Benefits"

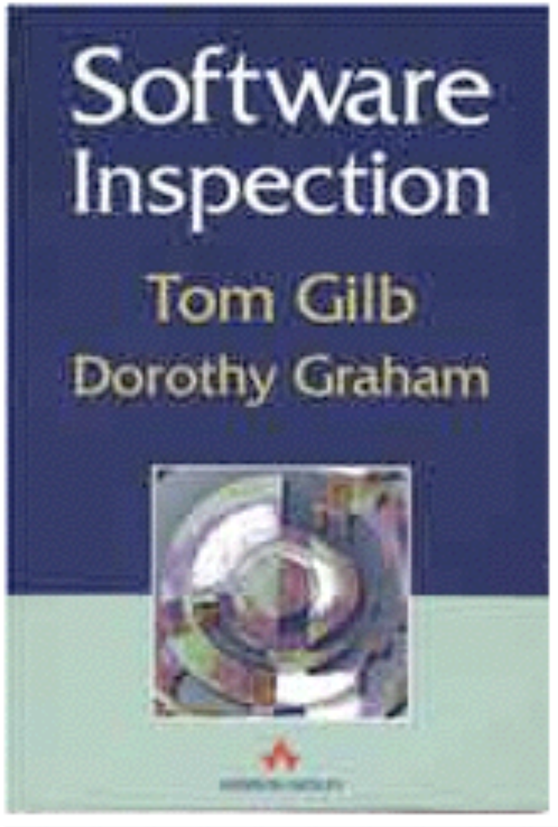
Strategy Impacts on Objectives

Cost

Benefit/Cost ratio

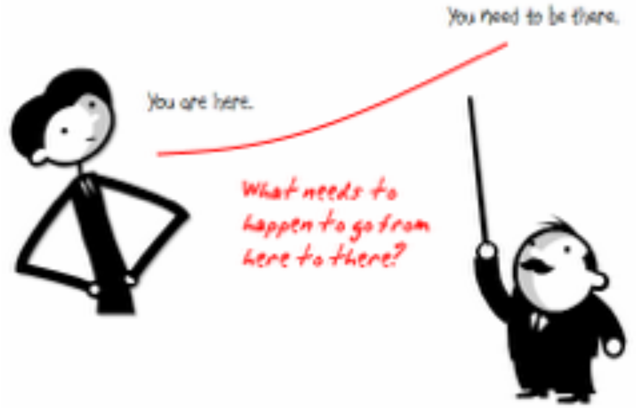
358 !

Toolkit: Measurement of Technical Specifications, Exit Defect-level control



Real case of large project 'objectives' 2009, London

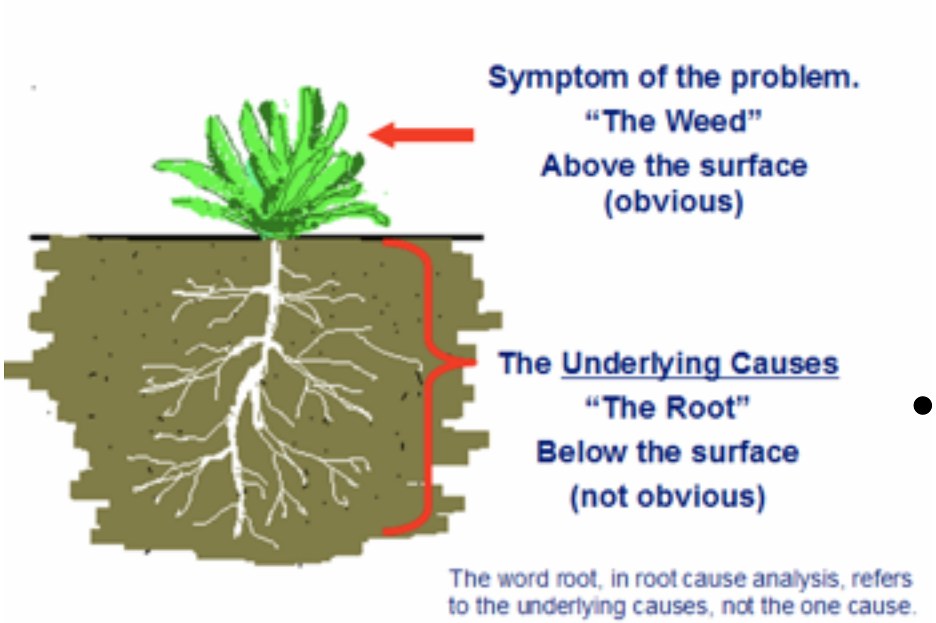
How many 'Major Defects' are there here?



- Rationalize into a smaller number of core processing platforms. This cuts technology spend on duplicate platforms, and creates the opportunity for operational saves. Expected 60%-80% reduction in processing cost to Fixed Income Business levies.
- International Securities on one platform, Fixed Income and Equities (Institutional and PB).
- Global Processing consistency with single Operations In-Tray and associated workflow.
- Consistent financial processing on one Accounting engine, feeding a single sub-ledger across products.
- First step towards evolution of "Big Ideas" for Securities.
- *Improved development environment*, leading to increased capacity to enhance functionality in future.
- Removes duplicative spend on two back office platforms in support of mandatory message changes, etc.

‘Rules’ (spec standards) are needed

- To define *specification* defects



- **Main Objectives Defects**

(***root causes***) lead to *potential defects* in *the next stages*
  - Architecture
  - Design
  - Testing
  - Construction
- Any of which can result in FAULTS in the final system
- Faults can result in breakdown of the real product.

QC Rules for Top Level Objectives

- **CLEAR: Every word and phrase should be clear enough to allow objective test of a delivery.** (we need to know exactly what is required and expected)
- **UNAMBIGUOUS: Every word and phrase should be unambiguous to all potential intended readers.** (no different than intended interpretations should be possible)
- **QUANTIFIED QUALITY**: all qualities (good things we want to improve) shall be expressed quantitatively.

- After we started the exercise I regretted not adding the usual rule:
- 4. **NO DESIGN: objectives shall not be expressed in terms of a design or architecture**
  – (a 'means' to reach the 'real' objective), when it is possible and is our real intent, to express the improvements in terms of quality, performance, and cost that are expected, instead.

Potential consequence of major defects in architecture specs

COUNT MAJOR ‘DEFECTS’ (RULES VIOLATIONS)
Rules Reminder:
1. Clear, 2. Unambiguous, 3. Quantified Qualities, 4. No Design/Architecture

- **“Rationalize into a smaller number of core processing platforms. This cuts technology spend on duplicate platforms, and creates the opportunity for operational saves. Expected 60%-80% reduction in processing cost to Fixed Income Business lines.**
- **International Securities on one platform, Fixed Income and Equities (Institutional and PB).**
- **Global Processing consistency with single Operations In-Tray and associated workflow.**
- **Consistent financial processing on one Accounting engine, feeding a single sub-ledger across products.**
- **First step towards evolution of “Big Ideas” for Securities.**
- ***Improved development environment*, leading to increased capacity to enhance functionality in future.**
- **Removes duplicative spend on two back office platforms in support of mandatory message changes, etc.”**

LINK WORDS: OBJECTIVE:ARCHITECTURE
RULE 4. No Design/Architecture
Link words expose 'designs' in 'requirements'

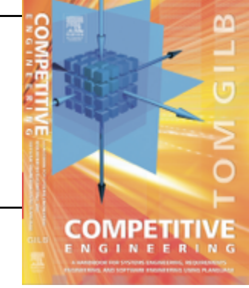
- Rationalize into a smaller number of core processing platforms. *This cuts technology spend* on duplicate platforms, and *creates the opportunity for* operational saves. Expected 60%-80% reduction in processing cost to Fixed Income Business lines.
- International Securities on one platform, Fixed Income and Equities (Institutional and PB).
- Global Processing consistency with single Operations In-Tray and associated workflow.
- Consistent financial processing on one Accounting engine, feeding a single sub-ledger across products.
- First step towards evolution of "Big Ideas" for Securities.
- *Improved development environment*, leading to increased capacity to enhance functionality in future.
- Removes duplicative spend on two back office platforms in support of mandatory message changes, etc.

Agile Spec QC Results

- **Reported** major defects =
- 1^{st} QC: 15, 17, 21
- **Second QC =18**, 15, 15, 13

- Estimated appx. Total defects found by a small team (2-4 people) = 36±6
  - 2x highest found.
- Estimated appx. Total Majors in the 110 words = **100**±10
  - (3x group total (36).
  - 30% effectiveness for the team)
- Estimated approximate total defects
  - in normalized page (300 words) = **280**±20
    - (Majors in 110 words x 3)

How can we improve such bad specification? (‘Planguage’)



Development Capacity:

Version: 3 Sept 2009 16:26

Type: Main <Complex/Elementary> Objective for a project.

Ambition Level: radically increase the capacity for developers to do defined tasks. <- Tsg

Scale: the Calendar Time for defined [Developers] to Successfully carry out defined [Tasks].

Owner: Tim Fxxx

C<u>alendar Time</u>: defined as: full working days within the start to delivery time frame.

Past [2009, {Bxx, Lxx, Gxx}, If QA Approved Processes used, Developer = Architect, Task = Draft Architecture] **15 days** ±4 ?? <- Rob

Goal[2011, { Bxx, Lxx, Gxx }, If QA Approved Processes used, Developer = Architect, Task = Draft Architecture] **1.5 days** ± 0.4 ?? <- Rob

Justification: Really good architects are very scarce so we need to optimize their use.

Risks: we use effort that should be directed to really high volume or even more critical areas (like Main Objective).

‘Planguage’ defined -→

A ‘Common language’

for Complex Systems Engineering

- ‘Planguage’ (pronounced PLAN-GUAGE)
- Is intended to be a ‘common language’ for large scale complex systems engineering
- Which our speakers Leon Kappelman (u N. Tx) and Marko Erman (Thales) referred to the need for yesterday (27 Oct, CSD&M, Paris))
- Planguage is a simple and forgiving modelling language that Bran Selic asked for
- Planguages dominant distinguishing modelling idea is the explicit inclusion of **<u>quantified qualities</u>**
  - **<u>Including ‘soft’ qualities (usability, security, adaptabiity)</u>**
  - _in both requirements, design, and project management
- Planguage models SYSTEMS, not ‘logic’

Requirement Defect Rates Improvement in 6 months in Financial Business, London, Gilb Client Using Spec Quality Control /Extreme Inspection + Planguage Requirements

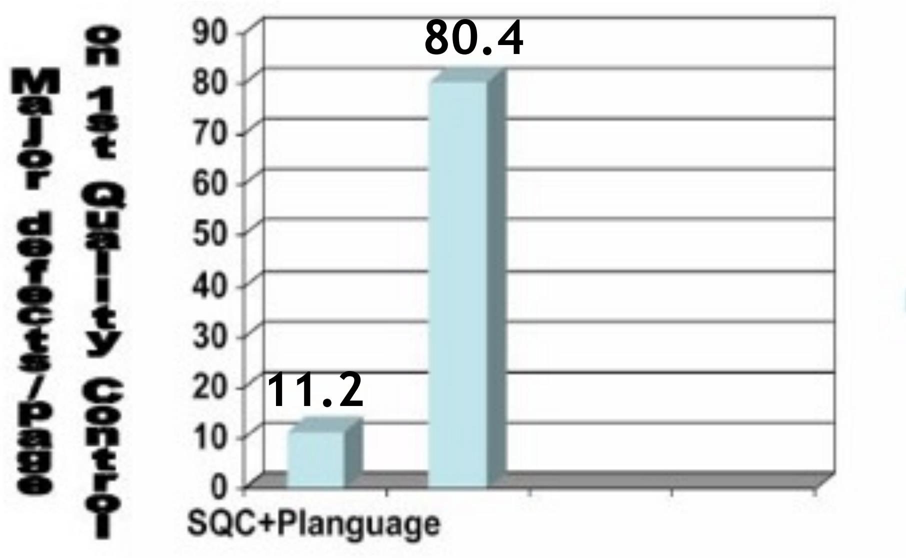
Across 18 DV (DeVelopment) Projects using the new requirements method, the average major defect rate on first inspection is 11.2 per logical page.

4 of the 18 DV projects were re-inspected after failing to meet the Exit Criteria of 10 major defects per page.

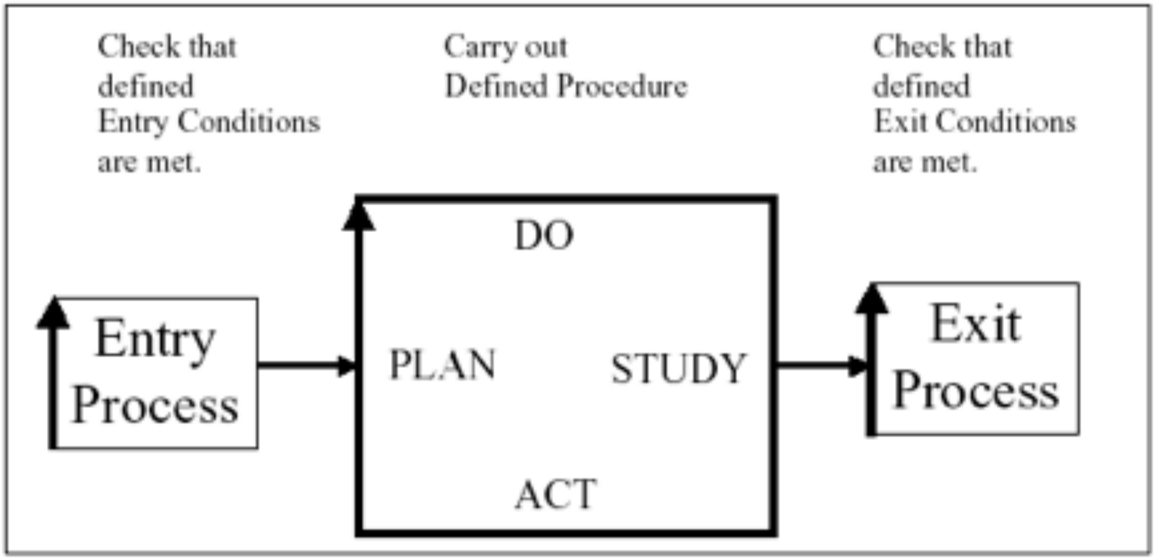
A sample of 6 DV projects with requirements in the 'old' format were tested against the rules set of:

> **1. The requirement is uniquely identifiable**
> **2. All stakeholders are identified.**
> **3. The content of the requirement is 'clear and unambiguous'**
> **4. A practical test can be applied to validate it's delivery.**

The average major defect rate in this sample was 80.4 per logical page.



You should have NUMERIC exit and entry quality levels from both test processes and related development processes



- **Entry and Exit Condition example:**
  - **Maximum estimated 1.0 Major defects per logical page remaining.**
  - **NASA example max 0.1 major/page**
- **This was the MOST important lesson IBM learned about software processes (source Ron Radice, co-inventor Inspections, Inventor of CMM)**
- **No ‘Garbage In’ to Next engineering process!**

The downstream alternative cost of quality at a UK Defence Electronics Factory. 9 to 1 more

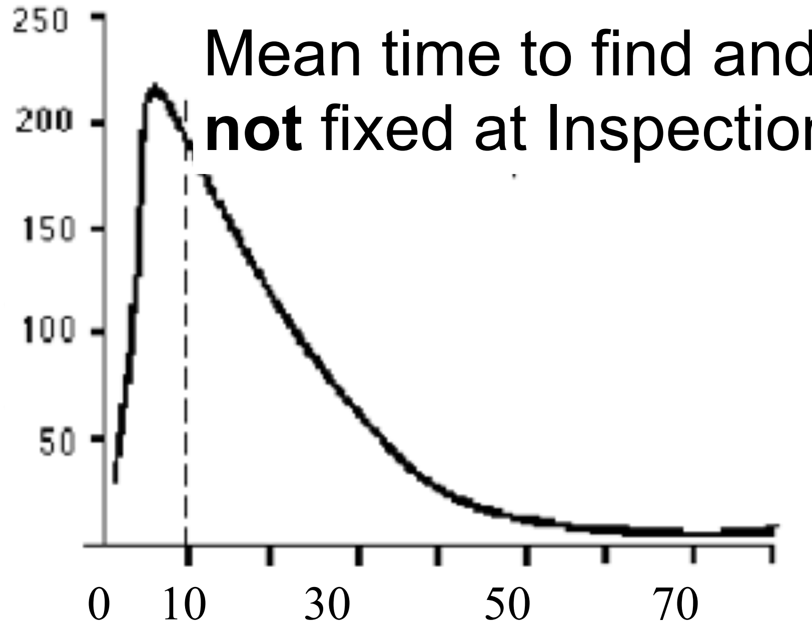
(all types of documents for electronics).

Number of defects of the 1,000 sampled Majors ------>

That we

manually estimated

downstream costs to fix

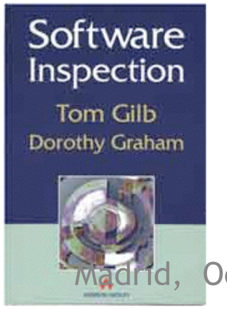


Mean time to find and correct a Major if **not** fixed at Inspection was **9.3** Hours.

It cost about **1** hour to find and fix a Major **at** time of Inspection

Estimated hours to find and correct later in test, or in field

Trevor Reeve

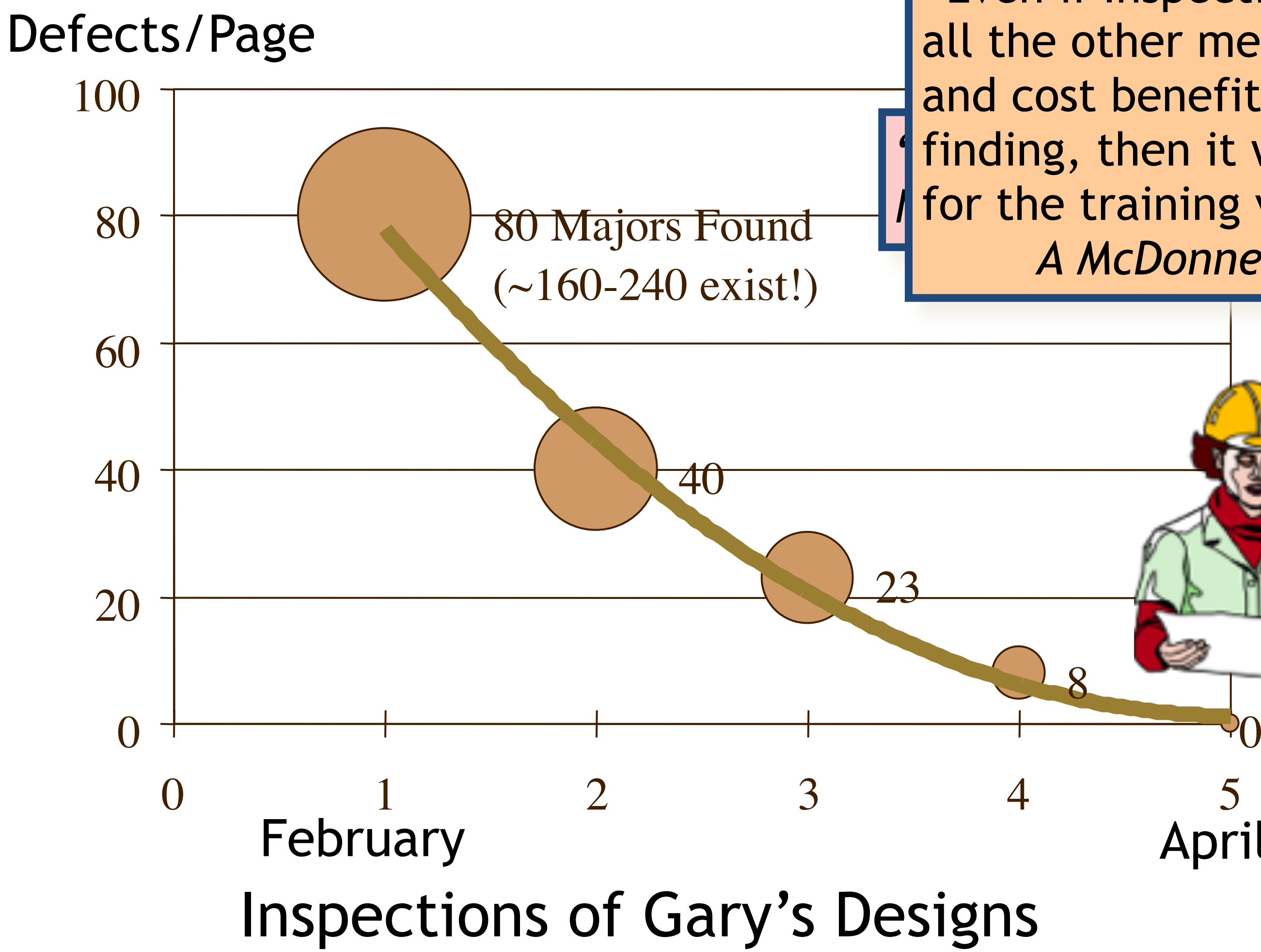


Source: Trevor Reeve, Case Study Chapter in "Software Inspection", Gilb client.

Philips MEL became "Thorn EMI", then Racal. Crawley UK. 1999 Raytheon

Positive Motivation

Personal Improvement



"We find an hour of doing Inspection is worth ten hours of company classroom training."

A McDonnell-Douglas line manager

"Even if Inspection did not have all the other measurable quality and cost benefits which we are finding, then it would still pay off for the training value alone."

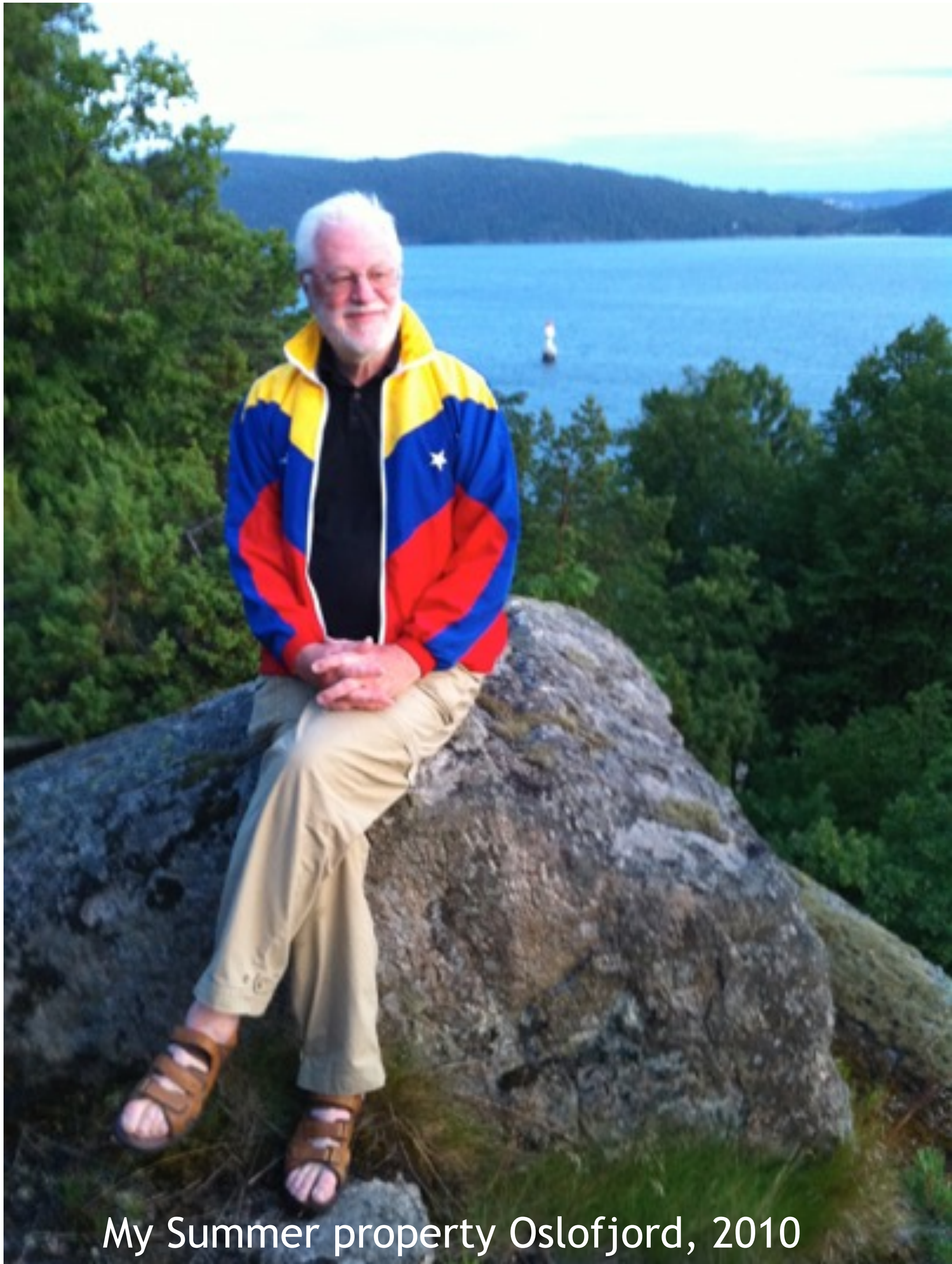
A McDonnellDouglas Director

Ten Principles for Complex Systems Engineering

- **Focus on clear top-level results:**
  - let architecture and process be your servants
- **Estimate design impact numerically**,
  - for top critical values, for all architecture
- **Prioritize Value delivery first:**
  - Prove it is delivered early, frequently
- **Contract for stakeholder-valued results**;
  - not for 'effort' or 'technology'
- **Estimate incrementally**:
  - learn the truth - don't fight bad estimates
- **Design to cost dynamically**:
  - brilliant design beats bad budgets
- **Collect rich relevant facts**
  - on requirements and designs
- **Reward results**, not effort
- **Measure Engineering Specification Quality**:
  - Exit numeric quality only
- **Reduce Specification Defects** by 100x

That's It

- Detail: Gilb.com downloads/papers
- Very happy to discuss with you here, and
  - +47 92066705 Mobile
- by email Tom@Gilb.com
  - Request a free digital copy of "Competitive Engineering" (by email or personally)
- These slides will be downloadable at gilb.com/downloads/slides



My Summer property Oslofjord, 2010