Fundamental Principles of Evolutionary Project Management

Tom Gilb
Tom@Gilb.com

Abstract: The Evolutionary Project Management method – abbreviated Evo – is arguably the best systems engineering project management method (Larman and Basili 2003). However, it is also probably the least known and the least discussed, so the aim of this paper is to shed some light on it. Evo is particularly good at dealing with large, complex, and innovative systems – it does so by breaking down the project into a series of numerous small incremental steps. Each Evo step is both an opportunity to deliver some useful results to the stakeholders, and an opportunity to learn more about the system.

INTRODUCTION

Let's discuss Evo by outlining its ten basic principles. They are as follows:

E1: Decompose by performance results and stakeholders;
E2: Do high-risk steps early, learn how 'unknowns' really perform;
E3: Focus on improving your most valuable performance objectives first;
E4: Base your early evolution on existing frameworks and stakeholders;
E5: Design to cost dynamically;
E6: Design to performance dynamically;
E7: Invest in an open-ended architecture early on;
E8: Motivate your team by rewarding results;
E9: Prioritize changes by value, not place in queue;
E10: Learn fast, change fast, adapt to reality fast.

PRINCIPLES

Principle E1: Decompose by performance results and stakeholders

Evolutionary project management demands that you deliver your requirements in small steps (for example 2% of budget). Many people initially assume that this means delivering certain components or functions. At the extreme, they falsely assume we are talking about delivering a car by starting with one wheel. This is nowhere near the reality of Evo.

The decomposition can follow many possible paths. The essential idea is this:

- Deliver real results to at least one stakeholder, at least one real person;
- Deliver some of the results that were planned in the requirements specification – this can be functionality or performance improvements;
- Make sure those results are a useful, high-priority, high-value increment to the stakeholder – an addition to what they have and something they need.

As an example, a military radar project in the UK told me they did not believe their project could be done evolutionarily. It only needed 20 minutes discussion, and two concepts to convince them differently. The first concept I suggested was that instead of taking purely a technical view - that the proposed system was primarily concerned with using 2 radar antennas, that the main point of the system was to increase accuracy of perception – a performance requirement (See Figure 1). The second concept was that the Royal Navy was one of the stakeholders (not just the new ship the radar system was scheduled for). With this a basis it became obvious that we could evolve the system (by incrementing ship and plane data, and processing logic) towards better perception ability. Second, we could do so using existing ships at sea (a stakeholder slice).

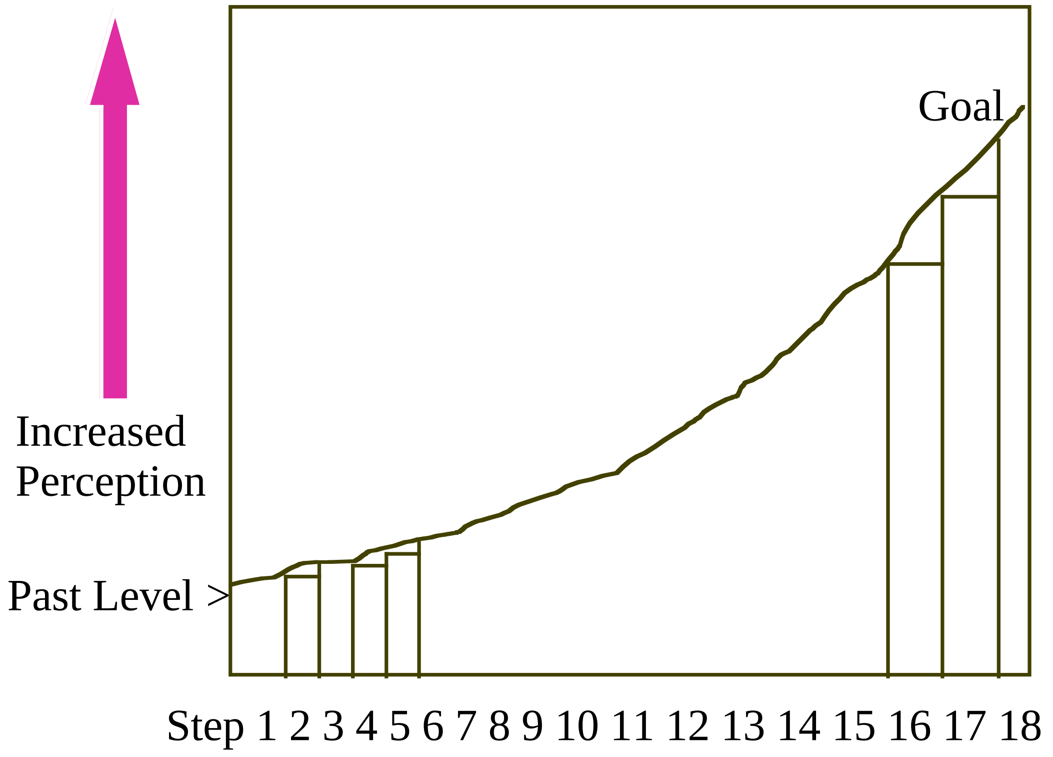


Figure 1. This diagram shows conceptually the increase in perception (of ships and planes) of a UK radar system

Principle E2: Do high-risk steps early, learn how 'unknowns' really perform

The Evo project manager, like the chess player, always has a lot of alternative step packages to consider, at each new step (See Table 1). There are many different policies they can use to select the 'best' step. In general, value for money is a good general paradigm. But one type of value, at early project stages, is to get facts on how things really work, so as to reduce the risk of bad designs when scaling up.

Evo is a good opportunity to manage risks, not by 'risk models', but by practical experience. Evo gives you a series of prototyping opportunities. You can get a realistic trial of new or risky technologies, in a realistic stakeholder environment, and integrated with some of the other technologies.

Design Ideas -> *Requirements*	**Candidate Step A: {Design X, Function Y}**	**Candidate Step B: {Design Z, Design F}**
Performance		
Reliability **99% <-> 99.9%**	**50% ± 50%**	**100% ± 20%**
Time Saving **11 seconds <-> 1 second**	**80% ± 40%**	**30% ± 50%**
Usability **30 minutes <-> 30 seconds**	**-10% ± 20%**	**20% ± 15%**
Cost		
Capital Cost **1 million US dollars**	**20% ± 1%**	**5% ± 2%**
Engineering Hours **10,000 Hours**	**2% ± 1%**	**10% ± 2.5%**
Worst Case Performance to Cost Ratio	**(0+40-30)/(21+3)** **= 0.42**	**(80-20+5)/(7+12.5)** **= 3.33**
Best Case Performance to Cost Ratio	**(100+120+10)/(19+1)** **= 11.5**	**(120+80+35)/(3+7.5)** **= 22.38**

Table 1: Using an Impact Estimation table to estimate uncertainty of alternative steps. By getting specific feedback from reality of a delivered step, the plus/minus uncertainties can be reduced substantially. The % figures is on a scale of 100% means we estimate that the goal

Principle E3: Focus on improving your most valuable performance objectives first

Dealing with risk is one useful Evo option, but delivering something to stakeholders (others than those interested in the risk aspects) of direct value is a crowd pleaser. In principle you find your next step by asking questions that include:

- Who is the most interesting stakeholder to deliver some value to now?
- What value (via a performance characteristic like reliability or security) does that stakeholder most want an improvement in now?

Imagine delivering 80% of the total value of the system with 20% of the budget, time and effort? That's what we aim for: 'skimming the cream off the top'.

If you do run out of time, or budget, at least you will have great credibility as a team that delivers useful results; and that is likely to put you first in line when resources are handed out next time around. It beats the Waterfall syndrome where the budget is used up, the deadline is passed, and you have not delivered anything useful.

Principle E4: Base your early evolution on existing frameworks and stakeholders

Evolutionary project management normally means starting from where you are and moving step by step in the direction of your dream horizon. It is irrelevant that your dream is radically different architecturally. If you want quick results, with real stakeholders, you have to exploit existing architecture, with existing customers.

Time and time again I have seen projects totally reject this idea, or never even consider it. Time and time again projects get into deep trouble – nearing failure. It seems people have a strong aversion to the awful old system or product they know. They have huge enthusiasm for the new untried system (the one where nobody could know any grief or disappointments yet).

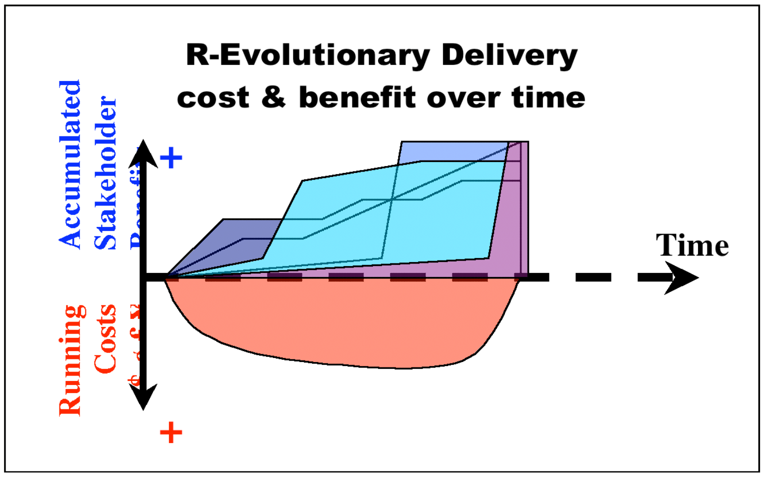


Figure 2. Stakeholder value delivery as early as possible is structurally possible with Evo project management. Diagram courtesy Kai Gilb (Gilb, K. 2005)

I have found it amazing how fast you can move towards the dream, for real, by starting from today's reality – warts and all. There probably will be a step where you can swap out the underlying architecture with something more 'modern'. But, the important thing is that you will be producing something useful for your stakeholders. See Figure 2.

A US Army system (Persincom) was in major crisis. General (Stormin' Norman) Schwartzkopf did not get the response he needed from it during the First Gulf War. I met with the system team, who were ready to spend years and millions of dollars making a better system. By taking the evolutionary approach, focusing on the performance they needed, and asking what we could do 'next week' to deliver it, the team came up with a shocking observation. They could deliver the thing most needed – the much faster response to the General's requests for information, if they simply programmed the system to respect rank better.

They told me I would never believe that they had not thought of this before, but all they had to do was note that the General was asking the questions, and move him to the head of the queue. 50% of what was needed could be done in practice, they assured me, with about 1% of the effort they had expected to use in building a new system from the ground up.

We would never have arrived at that conclusion without our commitment, for a week, to explore the evolutionary alternative. See Table 2 for an overview of the project's requirements and the design ideas we considered.

Principle E5: Design to cost dynamically

Design to cost is a well-known engineering paradigm. You don't ask an engineer to estimate the cost, you tell the engineer what cost you want and ask them to design the product to fit the constraint. With Evolutionary project planning, you have the opportunity to 'design to step cost'

Design Ideas -> ***Requirements***	Technology Investment	Business Practices	People	Empow-erment	Principles of IMA Management	Business Process Re-engineering	***Sum for Require-ment***
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 3:1 Return on Investment	45%	60%	10%	35%	100%	53%	303%
Morale 72 <-> 60 per month on Sick Leave	50%	5%	75%	45%	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**	

Table 2: An Impact Estimation table done as part of the Army project where we found powerful evolutionary steps for delivering the value that the stakeholders needed. The analytical process of looking for value impacts on performance requirements leads to insights about the best evolutionary steps

on each and every step. You take 2% of your budget and see if you can deliver some value for that. Many people are in denial about this task. They can spend 300% of your budget and deliver you failure. For my money if they cannot figure out how to deliver value for 2% of my budget, then they are almost certainly incompetent and should not get any more budget to waste. It is a great test of competence. Ericsson used such Evo methods in developing mobile telephone base stations for the Japanese market, and reduced their time to market by about half, compared to their previous Waterfall methods. The regular exercise of design to fit a step, and then getting immediate (end of Evo step) feedback, makes us quickly confront reality.

"If the organization cannot even manage its first simple task in the time agreed, it certainly should question the ability to manage more difficult tasks."
Jack Järkvik in 'On Succeeding" in an internal Ericsson publication

Principle E6: Design to performance dynamically

Dynamic design to performance levels is similar to design to cost. You throw designs into a step and measure progress towards the goal levels (See Table 3). As a basic strategy we would target the performance dimensions of greatest value, to the most important stakeholders first. We would select first the design ideas that are estimated to deliver the greatest impact on the valued performance dimensions. When we reach the goal levels of performance, we can stop expending resource. We know we are done, and the resource can safely be redeployed to meeting unfulfilled goals. When all performance goals are met, measurably, the development phase is by definition done. Compare this to a Waterfall project management situation, where resource would be much more likely to be expended unnecessarily.

	Step 1 Plan A: {Design X, Function Y}	Step 1 Actual	Step 1 Deviation ('+' is good and '-' is bad)	Total Step 1	Step 2 Plan B: {Design Z, Design F}	Step 2 Actual	Step 2 Deviation	Total: Step 1 + Step 2	Step 3 Next Step Plan
Performance									
Reliability 99% <-> 99.9%	50% ± 50%	40%	-10%	40%	30% ± 20%	20%	-10%	60%	0%
Time Saving 11 seconds <-> 1 second	80% ± 40%	40%	-40%	40%	30% ± 50%	30%	0%	70%	30%
Usability 30 minutes <-> 30 seconds	10% ±	12%	+2%	12%	20% ± 15%	5%	-15%	17%	83%
Cost									
Capital Cost 1 million US dollars	20% ± 1%	10%	+10%	10%	5% ± 2%	10%	-5%	20%	5%
Engineering Hours 10,000 Hours	2% ± 1%	4%	-2%	4%	10% ± 2.5%	3%	+7%	7%	5%
Calendar Time (Weeks)	1	2	-1	2	1	0.5	0.5	2.5	1

Table 3: An example of using an Impact Estimation table to estimate, measure, and incrementally track progress towards performance goals. The first number cited in the Usability goal cell (30 minutes) is the benchmark, and corresponds to 0% progress towards the goal. The second number cited (30 seconds) is the goal level, and corresponds to 100% estimated or actual progress

towards the target

Principle E7: Invest in an open-ended architecture early on

Open-ended architecture is the entire set of system design that arguably contributes to making the system or product easier to change. It includes, as well as structural and technical devices, things like organizational, motivational, and contractual devices.

Evolutionary development project management, and evolutionary life cycle management (Rajlich and Bennett 2000) both work more efficiently to the degree that the new evolutionary steps are easier to carry out as a result of the openness of the architecture.

Adaptability:
Ambition: The ability of our system to easily tolerate unexpected changes. The set of all other ease-of-change qualities below {Extendibility, Portability, Serviceability}.

Extendibility:
Scale: The engineering effort needed to add [defined Capacity] to the product.
Goal [Capacity = memory by factor 10]: Less than 10% of cost of memory itself.

Portability:
Scale: The engineering effort needed to move [defined System Elements] to [defined Target Environments] using [defined tools or Skilled People or processes].
Goal [System Elements = software logic and data, Target Environments = East Asian Markets, Skilled People = Average Programmers]: 1 hour per 100 lines of code.

Serviceability:
Scale: The ease of giving [defined Service Types] in [defined Service Locations] by [defined levels of Service People].
Goal [Service Types = Shop Counter, Service Locations = Major Chains,
Service People = Certified Trained Specialists]: 90% of Service Cases within 30 minutes "in shop wait".

Figure 3. An example of using Planguage to define the degree of open-endedness needed in the systems architecture

The degree of open-endedness needed can be specified quantitatively, as in the example in Figure 3, and the system architects can target those requirements by finding architecture that meets the goal levels. We can get some measure of whether we have actually got the required levels of adaptability by measurement of the costs of delivering an Evo step.

I would argue that this systematic approach to open-endedness, as opposed to the 'throw nice sounding structures into the design' approach, is a sounder systems engineering approach to ensuring systems architecture necessary for evolutionary project management.

Principle E8: Motivate your team by rewarding results

Does your project get motivated to fail? Do they get paid more, or less, if the project is late? I have a theory that there are at least two major reasons that so many projects fail, entirely or partly. These reasons are as follows:

• We don’t do them evolutionarily;
• We don’t motivate people to deliver real measurable results to stakeholders.

Management is not managing well when they take such high risks with the entire project, and they reward people for effort, not results.

A Simple Evolutionary Project Management Method

Tag: Quantified Simple Evo Project.

Version: July 8, 2003 (3). Owner: Tom@Gilb.com. Status: Draft.

Project Process Description

1. Gather from all the key stakeholders the top few (5 to 20) most critical performance (including qualities and savings) 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: Reliability: Scale: Mean Time Between 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.

- Based on numeric feedback, and stakeholder feedback; change whatever needs to be changed to reach goals.

8 When all goals are reached, “Claim success and move on” (Gerstner 2002). Free the remaining resources for more profitable ventures.

Project Policy

1. The project manager, and the project, will be judged exclusively on the relationship of progress towards achieving the goals versus the amounts of the budgets used. The project team will do anything legal and ethical to deliver the goal levels within the budgets.

2. The team will be paid and rewarded for ‘benefits delivered’ in relation to cost.

3. The team will find their own work process, and their own design.

4. As experience dictates, the team will be free to suggest to the project sponsors (stakeholders) adjustments to ‘more realistic levels’ of the goals and budgets.

Figure 4. An agile evolutionary process that includes motivation to focus on results, not hours worked (Gilb 2003)

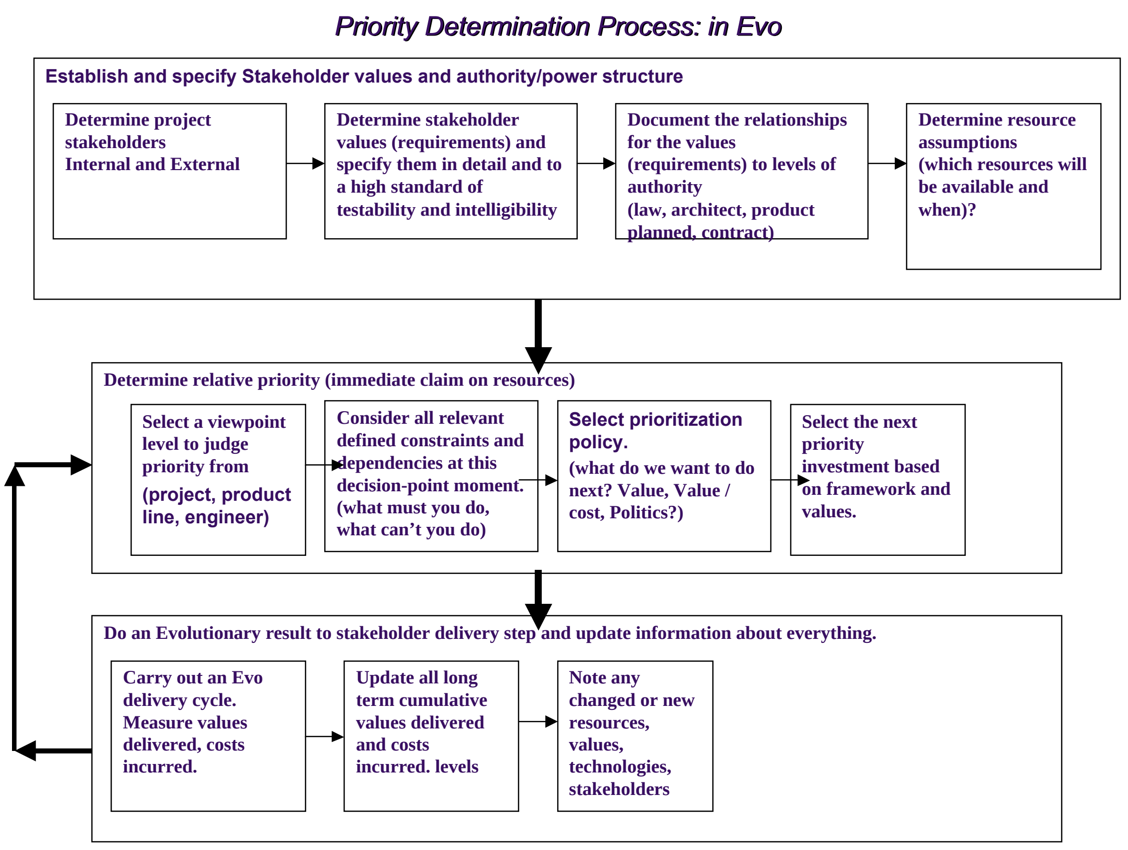
Principle E9: Prioritize changes by value not place in queue

It does not matter when an idea for change comes up. The decision what to do in the next Evo step should be based on the value (to cost ratio) of the idea. One major built-in advantage of Evo is exactly that we can make the best possible move immediately - even if we just saw the opportunity recently (say, during the last week). We are not tied down to committee-approved decisions from last year. The job of the committee is to pre-approve that the project makes the decisions that give maximum competitiveness at all times.

Principle E10: Learn fast, change fast, adapt to reality fast

This principle is a good summary of the Evo method. The chase is to the swift.

```
But if the arrow is straight
 And the point is slick,
    It can pierce through dust no matter how thick.
```
Bob Dylan (1941 -) http://bobdylan.com/songs/restless.html, Restless Farewell



Priority Determination Process: in Evo
Establish and specify Stakeholder values and authority/power structure
Determine project stakeholders Internal and External
Determine stakeholder values (requirements) and specify them in detail and to a high standard of testability and intelligibility
Document the relationships for the values (requirements) to levels of authority (law, architect, product planned, contract)
Determine resource assumptions (which resources will be available and when)?
Determine relative priority (immediate claim on resources)
Select a viewpoint level to judge priority from (project, product line, engineer)
Consider all relevant defined constraints and dependencies at this decision-point moment. (what must you do, what can't you do)
Select prioritization policy. (what do we want to do next? Value, Value / cost, Politics?)
Select the next priority investment based on framework and values.
Do an Evolutionary result to stakeholder delivery step and update information about everything.
Carry out an Evo delivery cycle. Measure values delivered, costs incurred.
Update all long term cumulative values delivered and costs incurred. levels
Note any changed or new resources, values, technologies, stakeholders

Figure 5. Evo permits dynamic prioritization at each step. We can plan to deliver the step content that creates the most competitive value for effort expended, even if the insights leading to that next step were generated as feedback data from the previous step

SUMMARY

Evolutionary project management is based on some fundamental concepts:

- Do some useful changes for your stakeholders, early and frequently;
- Learn from live systems experience, what really works and what does not;
- Use numeric specification of your key performance objectives;
- Estimate numeric impacts of design ideas on your requirements;
- Measure the effect of each Evo step;
- Focus on delivering highest available value at every step;
- Don't plan too far in advance – you have long-term objectives, but you need to react to short-term realities in order to meet them.

REFERENCES

Gerstner, Louis V., *Who Says Elephants Can't Dance?* Harper Collins, 2002.

Gilb, Kai, *Evo*. 2005. Draft manuscript at http://www.gilb.com

Gilb, Tom, "Software Project Management: Adding Stakeholder Metrics to Agile Projects." *Novática*, Issue 164, July-August 2003. (Special Edition on Software Engineering - State of an Art, Guest edited by Luis Fernández-Sanz. Novática is the journal of the Spanish CEPIS society ATI (Asociación de Técnicos de Informática.) See http://www.upgrade-cepis.org/issues/2003/4/upgrade-vIV-4.html In Spanish: http://www.ati.es/novatica/2003/164/nv164sum.html

Gilb, Tom, "Adding Stakeholder Metrics to Agile Projects." *Cutter IT Journal: The Journal of Information Technology Management*, July 2004, Vol. 17, No.7, pp31-35. See http://www.cutter.com.

Gilb, Tom, *Competitive Engineering: A Handbook for Systems Engineering, Requirements Engineering, and Software Engineering using Planguage*. Elsevier Butterworth-Heinemann, Due June 2005. ISBN 0750665076.

Goth, Greg, "New Air Traffic Control Software Takes an Incremental Approach." IEEE Software, July/August 2000, pp 108-111.

Larman, Craig and Basili, Victor R., "Iterative and Incremental Development: A Brief History." *IEEE Computer Society*, June 2003, pp 2-11.

Larman, Craig, *Agile and Iterative Development: A Manager's Guide*. Addison Wesley, 2003. See Chapter 10 on Evo.

Rajlich, Václav T., and Bennett, Keith H., "A Staged Model for the Software Life Cycle." *IEEE Computer*, July 2000, pp. 66-71.

BIOGRAPHY

Tom Gilb is the author of 'Competitive Engineering: A Handbook for Systems & Software Engineering Management using Planguage' (due June 2005), 'Principles of Software Engineering Management' (1988) and 'Software Inspection' (1993). His book 'Software Metrics' (1976) coined the term and, was used as the basis for the Software Engineering Institute Capability Maturity Model Level Four (SEI CMM Level 4). His most recent interests are development of true software engineering and systems engineering methods.

Tom Gilb was born in Pasadena CA in 1940. He moved to England in 1956, and then two years later he joined IBM in Norway. Since 1963, he has been an independent consultant and author. He is a member of INCOSE.

Author Contact: Tom@Gilb.com

Edited by Lindsey Brodie, lindseybrodie@btopenworld.com