Evolutionary Project Management:

Advanced Theory and Practical Experiences

By Tom Gilb

© Tom Gilb, Iver Holtersvei 2, N-1410 Kolbotn, Norway. Gilb@ACM.org

This paper is one of several supplementary documents for a one day DC SPIN Course June 17 1999

Introduction: 'EVO' is process control of projects.

Evolutionary project management is a 'new' way to get control over project results. It is new in the sense that most project cultures have little formal know-how or experience using this method. It is *not* new in the sense that there are at least three decades of successful high tech project management experience using this method. It has arguably the best track record of any project management method in the world. Since 1985, Hewlett-Packard has used it as a major competitive weapon [COTTON96] in at least 8 divisions. *"The evolutionary development methodology has become a significant asset for Hewlett-Packard software developers. Its most salient, consistent benefits have been the ability to get early, accurate, well-formed feedback from users and the ability to respond to that feedback"* [MAY96 p44]. Read it!

Listen also, to the largest (military & space) and longest (since 1970) user of the method:

" Management has learned to expect on-time, within budget deliveries.' 'LAMPS ... a 4 year ... 200 person-years" (project was delivered) "in 45 incremental deliveries. Every one of those deliveries was on time and under budget. NASA space program ... 7,000 person-years software development ... few late or overrun (budgets) .. in .... decade, and none at all in the past four years" (Harlan Mills, in IBM Systems Journal Number Four, 1980).

Is this how *your* projects are currently described? Is any reader interested in getting a reputation like this? These projects were some of the most difficult, state of the art quality, high tech, *fixed* budget and *fixed* deadline projects in existence. That is exactly what drove IBM Federal Systems Division (later Loral and Lockheed Martin) to develop these methods for their own use. Many others have used their own versions, but I know of no others with such long-term, large-scale industrial documented experience.

The basic principle of evolutionary delivery is simple: the 'Plan Do Study Act.' cycle. In other words, the process control cycle taught by Walter Shewhart of AT&T since the 1920's and W. Edwards Deming (his pupil) to the Japanese since the late 1940s. It is one of nature's great laws. Learn, adapt, survive.

The Evolutionary method is not necessary for predictable, low-risk, low-turbulence situations. But our entire political, technological, economic world is already very much more unpredictable, complex and large scale than ever before. So using evolutionary delivery to manage projects must be the norm, not an exception. Peter J . Morris in his book *"The Management of Projects"* [MORRIS94] concludes, after a deep analysis of project management since the wartime 1940s, that we need new project methods and those methods must have some kind of learning process built in. The US Department of Defense has issued a new MIL-STD-498 (5th Dec. 1994, already being replaced by 'civil' standards, IEEE doc 12207) which supports the use of evolutionary delivery, and projects which do not have 'final and correct user requirements' specified. "Crosstalk", a Dept. of Defense publication reports **"The waterfall method is not recommended for major software-intensive Air Force acquisition programs"** [SORENSEN95].

The basic idea is appealing. Deliver project results in early, high-profit, frequent, useful increments. Who could be against such a useful idea? The big problem is usually 'how do we do this in practice? How do we decompose the big project into a succession of monthly improvements in the hands of the customers or users?' Some people don't see any difficulty and just do it. However, many claim it is impossible; *they* are unable to envision how to do it. Are such

defeatists ‘stupid’? No, but they probably have had no *training* in the method. Certainly, there needs to be *motivation* to persist until an evolutionary route is identified and, strong technical know-how and a good insight into the customer environment are extremely useful.

1. Practical Example: Evolutionary Management (Evo)

The Naval Weapons System.

Typical Perceived Barrier to **Evo**: "It cannot be done until the new {thing, building, organization, system}.... is *ready* in some years time".

British Naval Weapons System case: Once, when holding a public course in London on the Evolutionary Method, a participant came to me in the first break and said he did not think he could use this early-incremental method. *Why?* "Because my system is to be mounted on a new ship, not destined to be launched for three years."

At that point, I did not know anything about his system, but I expressed confidence that there is *always* a solution, and ‘bet’ that we could find one during the lunch hour.

He started lunchtime by explaining that his weapons research team made a radar-like device that had two antennas instead of the usual *one*, which had their signals analyzed by a computer before presenting their data. It was for ship-and-air traffic surrounding the ship it was on.

I made a stab at the ‘*results*’ he was delivering and, who his ‘**customer** was’ - two vital pieces of insight for making evolutionary delivery plans. "May I assume that the main result you provide is 'increased accuracy of perception', and that your 'customer' is Her Majesty's Navy?"…. "*Correct*," he replied.

"Does your 'box' *work*, more or less, now, in your labs?" I ventured. (Because if it did, that opened for *immediate* use of *some* kind.) "Yes," he replied. "Then what is to prevent you from putting it aboard one of her Majesty's current ships and ironing out any problems in practice, enhancing it, and possibly giving that ship increased capability in a real war?" I tried, innocently.

"Nothing!", he replied. And at that point I had won my bet, 20 minutes into the lunch.

"You know, Tom”, he said after five minutes of silent contemplation, “the thing that really amazes me, is that not *one* person at our research labs has *ever* dared think that thought!".

The thing to notice here was that the customer was *not* the ‘new ship’, and that the project was *not* to put the electronics box on the new ship. The project was to give increased perception to the *real* customer, Her Majesty's Navy.

Notice the “method” emerging from this example:
1. Identify the *real* customer, and *plan to deliver results* to them.
2. Identify the *real* improvement results and *focus on delivering* those results to the real customer.

in other words,
1. Do not get distracted by intermediaries (the new ship);
   think ‘Her Majesty’s Navy’ or even ‘The Western Alliance’.
2. Do not get distracted by the perceived project product (the new radar device for the new ship);
   think ‘increased accuracy of perception’.

2. Language core: Evolutionary Step Specification.

An Evo *step* can be defined in a variety of ways: for example, as a *set of previously defined functions* (FX) and/or *design ideas* (DIA, DIB).

STEP1[BASIS Current Product]: {**FX**[COUNTRY=USA], **DIA**, **DIB**}.

An evolutionary step is a package of ***functions*** *and* ***design ideas***, which when implemented for some real, or trial, system-user give *measurable testable* results, hopefully positive. The 'results' should be some degree of fulfillment of the current formally stated outstanding ***requirements***.

Here is one way of stating a complete Evo plan or parts of it.

EVO-PLAN: {**STEP1**[BEFORE **REST**], **REST**:{**SM, SV, SX** [AFTER **SM** ENDS:]**, SZ**}.

This means, there is a defined set of steps {STEP1, SM, SV, SX, SZ} which make up the Evo plan. STEP1 is defined in the plan as one to do 'before' 'REST'. The others have not had their sequence determined. But SX *must* be done 'after' SM 'ends'. See these three parameters in index or glossary.

The 'step trial customer' or 'delivery area' can be specified by a qualifier.

STEP22 [STATE={CA, NV, WA}] **REST**. "see STEP22 on table below"
STEP1.1 [COUNTRY={USA, CAN, MEX}] {**STEP1**, **SM, FX** [STATE**=**{**WA, GA, FL**}].

Evo Step 'attribute management' can be exercised using an Impact Estimation table.

Step->> Attribute	**STEP1** plan %	actual %	deviation %	**STEP2** to **STEP20** plan	plan cumul-ated to here	**STEP21** [CA,NV,WA] plan	plan cumul-ated to here	**STEP22** [all other steps] plan	plan cumul-ated to here
QUAL-1	5	3	-2	40	43	40	83	-20	63
QUAL-2	10	12	+2	50	62	30	92	60	152
QUAL-3	20	13	-7	20	33	20	53	30	83
COST-A	1	3	+2	25	28	10	38	20	58
COST-B	4	6	+2	38	44	0	44	5	49

In addition to the examples shown here, the *entire* Planguage is available to express Evo ideas.

3. Specific Rules : Evolutionary Planning (RULES.EVO)

Rules for specification of Evolutionary Project Plans: Tag: RULES.EVO.
Version 0.4. Owner: TsG. August 17th 1996. No QC exit.

0:NEXT: The *next* **implementation cycle** *must* be 'planned' in detail. Later **steps** *can* be unsequenced, and represented by non-bold ('not yet defined') tags alone, by <fuzzy specifications>, or by **hierarchical tags** (**A.B**.C). *They will be 'detailed', as their turn to be 'next' approaches. 'Planned' means following these Specific Rules (RULES.EVO), RULES.GR, and Specific Rules which apply to the specification types being planned (like Function, Requirements, Design).*

1:IMPACT: The defined **steps** of the *next* **implementation cycle** *shall* be estimated for their **impact** on all **critical** quality and cost **requirements**. Estimate the step-impact for the *next* step in detail. Other steps *may* be more roughly estimated. *They will be estimated in detail, as their 'turn' approaches. We assume the use of an impact estimation table, but other formats are possible.*

2:UNCERTAINTY: Each impact estimate *should* include a ± **uncertainty** estimate (e.g. 30%**±10%**). The estimate's **sources**, **evidence**, and **credibility** rating (0.0 to 1.0) would be welcome. As a minimum, give the name of the estimator, and date.

3:PAST: The *actual* results of any *previous* **implementation cycle**, and the *cumulative* **impacts** on *all* **requirement** levels to-date *must* be included, in an Impact Estimation table format.

4:TAG STATUS: A tag's 'status' *shall* be *visually* indicated; *like 'steps defined in the plan'* in **BOLD**, and tags of *to-be-defined* steps in <FUZZY-BRACKETS>. *Bold **'defined tag'** use saves detail in planning specification, and the tag definitions make plan interpretation more precise.*

5:COST%: Any step with any estimated **cost attribute** incremental impact, at *that* step, *exceeding* 5% of the project's initial PLAN budget total, *shall* be divided into *smaller* step sizes *of that cost attribute*. An *average* of 2%-of-cost steps is desired (risk of economic loss is then at 2% maximum).

6:TIME%: Any step which would take *more than 5%* of the total project calendar *time to the main long-term deadline, must* be divided into smaller steps. An *average* of 2%-of-time steps is desired.

7: PRIORITY: steps on which impact estimates show the greatest 'multiple-requirement **benefit**-to-**cost** ratio' *shall* generally be done *earliest*, wherever logically possible, and when 'other considerations' *(such as a customer contract or request)* do not have higher priority. Specific priority factors, violating this general rule, *shall* be clearly documented.

Example: STEP44<-Contract Requirement 6.4. "reason for sequencing"

8:COMPLEX STEPS: Complicated steps, containing many functions and/or design ideas *should* be specified *separately* from the Project IE Table, and from any other form of project time plan.

*It is also expected that the **function** and '**design ideas'** are specified in detail, elsewhere. Step components must be specified so 'well', somewhere, that there is enough detail to understand their resulting impacts. This needs to be done without either complicating the evolutionary planning, or leading to oversimplified specification. Oversimplification loses attribute control.*

9:COMPLETENESS: all defined 'design idea and function' **implementation**, or change, *must* be represented *somewhere* on the plan. *This applies even it has to be done by using a 'high level tag' or catch-all phrase like "all other plans".*

4. Process Description/Standards: Evolutionary Management

Process Description The delivery step: Tag: PROCESS.SM

Entry Conditions ENTRY.SM [STEP n] "Step Management"

1.All **pre-requisite steps** considered before entry. (see BEFORE, ENDS, AFTER parameters)

Procedure PROCEDURE.SM [STEP n]

1. PLAN step detail. *Set step goals, select step design and function, decide [qualifiers].*
2. DO the step as planned. *Build, buy, install, train as needed to implement step.*
3. STUDY: measure result and compare with expected results. *Test, measure with METER, use IE.*
4. ACT re-plan and repeat this cycle, or exit. *Re-plan and repeat step, if necessary to reach goals.*

Exit Conditions EXIT.SM [STEP n]

1. Step completed, or dropped. *Exit if goals reached, give up if impractical*.

Process Description Evolutionary Project Management: Tag PROCESS.PM

Entry Conditions 'for evolutionary management' ENTRY.PM [PROJECT x]

1. At least one set of longer-term 'horizon' quantified project **objectives** have exited from QC.
2. At least one set of corresponding **design ideas** have exited from QC
3. The design ideas have been evaluated by **Impact Estimation** and have exited from QC
4. The **functional requirements** have been defined or identified.
5. The level of **uncertainty** acceptable to the project has been formally determined (±% deviation from a plan).

Procedure PROCEDURE.PM [PROJECT x]

1. PLAN: project **objectives** and **architecture**. *The 'head', the high level overview plan.*
2. DO: a series of **steps** until objectives reached. *The 'body', repeat until goals reached.*
3. STUDY: results of each step. *Test, measure, sample and compare to long range goals.*
4. ACT: change plans to succeed. *Change goals, qualifiers, designs, resources so as to reach highest priorities, to avoid failure and get success levels of priority goals.*

Exit Conditions EXIT.PM [PROJECT x]

1. If **resources** used up. *Stop project deliveries when no more money, time, people. Keep results!*
2. or **PLAN** levels reached. *Stop using resources when goals are reached, unless new goals agreed.*

5. Principles: Evolutionary Management

0. The Principle of 'Early to bed, early to rise, makes a man healthy, wealthy, and wise'.
The earlier you get real experience with your abstract ideas, the better.

1. The Principle of 'Truckloads of food need to be digested one bite at a time'.
The system user needs to digest new systems in small increments.

2. The Principle of 'Better to light a candle than curse the dark'.
A little practical experience beats a lot of committee meetings.

3. The Principle of 'Cause and Effect'.
Observe reaction to small changes, to understand causes of effects.

4. The Principle of 'Early bird gets the worm'.
Your customers will be happier with an early long-term stream of their priority improvements, than years of promises, culminating in late disaster.

5. A Principle of 'Tao Teh Ching'.
Deal with little troubles before they become big.

6. The Principle of 'Capablanca's next move'.
There is only one move that really counts, the next one.

7. The Principle of 'A bird in the hand being worth two in the bush'.
The 'next evolutionary delivery step' should give the best result you can get now.

8. The Principle of 'The devil you know'.
No matter how revolutionary your vision,

you are wise to start from where you *are*, what you *have*, and what your *customers* have

9. The Principle of 'Adaptive Architecture'.
Since you cannot be sure where or when you are going,
your first priority is to equip yourself to go almost anywhere, anytime.

The principles of Tao Teh Ching (500 BC)
That which remains quiet, is easy to handle.
That which is not yet developed is easy to manage.
That which is weak is easy to control.
That which is still small is easy to direct.
Deal with little troubles before they become big.
Attend to little problems before they get out of hand.
For the largest tree was once a sprout,
the tallest tower started with the first brick,
and the longest journey started with the first step.[1]

6. Advanced ideas: Evolutionary Management

How to decompose systems into small evolutionary steps.

1• *Believe* there is a way to do it, you just have not *found* it yet![2]
2• *Identify* obstacles, but don't use them as excuses: use your imagination to get *rid* of them!
3• Focus on *some usefulness* for the user or customer, however small.
4• Do not focus on the design ideas themselves, they are distracting, especially for small initial cycles. Sometimes you have to ignore them entirely in the short term!
5• Think; one customer, tomorrow, one interesting improvement.
6• Focus on the *results* (which you should have defined in your goals, moving toward PLAN levels).
7• Don't be afraid to use temporary-scaffolding designs. Their cost must be seen in the light of the value of making some progress, and getting practical experience.
8• Don't be worried that your design is inelegant; it is resultsthat count, not style.
9• Don't be afraid that the customer won't like it. *If* you are focusing on results *they want*, then by definition, *they* should like it. If you are not, then *do*!
10• Don't get so worried about "what might happen afterwards" that you can make no practical progress.
11• You cannot foresee everything. Don't even *think* about it!
12• If you focus on helping your customer in practice, *now*, where they *really* need it, you will be forgiven a lot of ‘sins’!
13• You can understand things much better, by getting *some* practical experience (and removing *some* of your fears).
14• Do *early* cycles, on willing local mature parts of your user community.
15• When some cycles, like a purchase-order cycle, take a long time, initiate them early, and do other useful cycles while you wait.
16• If something seems to need to wait for ‘the big new system’, ask if you cannot usefully do it with the ‘awful old system’, so as to pilot it realistically, and perhaps alleviate some 'pain' in the old system.
17• If something seems too costly to buy, for limited initial use, see if you can negotiate some kind of ‘pay as you really use’ contract. Most suppliers would like to do this to get your patronage, and to avoid competitors making the same deal.
18• If *you* can't think of some useful small cycles, then talk directly with the real ‘customer’ or end user. They probably have dozens of suggestions.
19• Talk with end users in *any* case, they have insights you need.

20• Don't be afraid to use the old system and the old ‘culture’ as a launching platform for the radical new system. There is a lot of merit in this, and many people overlook it.

THE FUNDAMENTAL *EVOLUTIONARY PLANNING* POLICY

PP1:Budget: No project cycle shall exceed 2% of total budget before delivering measurable results to a real environment.
PP2:Deadline: No project cycle will exceed 2% of total project time (one week for a year's projects) before it demonstrates practical measurable improvement, of the kind targeted.
PP3:Priority: Project cycles which deliver the most planned results to customers, for the resources they claim, shall be delivered first, to the customer.

7. Case/Advanced Example: Evolutionary Management

The German Telecommunications Company.

Perceived Barrier to use of the evolutionary method: "It is too late, we have already invested so much the old way, that we just have to see it through".

At a large German telecommunications business in December 1984, about 985 software engineers had been working for three years on a major new world-market product. December 1984 was the deadline for delivery of the product, but their 40,000 node PERT chart, the Financial Director told me, estimated that they had 2 or 3 years more software effort left. The hardware was ready, but the software was late. Corporate top management had given them one more year, to December 1985. Deliver, or forget the whole market, which by that time would be taken over by competitors.

I suggested re-planning the project in smaller and critical increments first. They told me that this was unthinkable. The software was already written, they claimed, only testing remained. They also had a rather long list of other reasons why incremental result delivery would not work with them.

Using common sense, we worked out a basic evolutionary plan. The small-model software first (there were 35 signed contracts for it, none for the medium and large systems). Then fundamental telephone services before advanced fancy stuff (‘holographic transmission’ was my private name for the fancy stuff).

After what seemed like seven management layers (there were probably only four) of "you must present this to my boss", we ended up in the office of Herr R., The project Director. He thought it was all good common sense, and stared coldly at his (cowardly, cautious?) subordinates as he asked: "Can you do it this way?", (assenting nods) "Then do it!".

They did too. By November 1985, on a return visit, they told me that the small systems had been operating for over six weeks with several real customers, with no problems whatsoever. Note, three months *before* the impossible deadline!

As in many other cases, I had to spell out the basic steps myself. I had to make them obvious clear simple steps. I could not merely suggest the method. And then, in spite of the obviousness, I had to get to the right person to make a decision.

The product is still a major successful product on the market 10 years later. Herr R. correctly concluded *then* that unless the organization changed its mode of thinking, the same type of project problem would continue to recur. So he took steps to improve the organization.

8. Diagrams/icons: Evolutionary Management

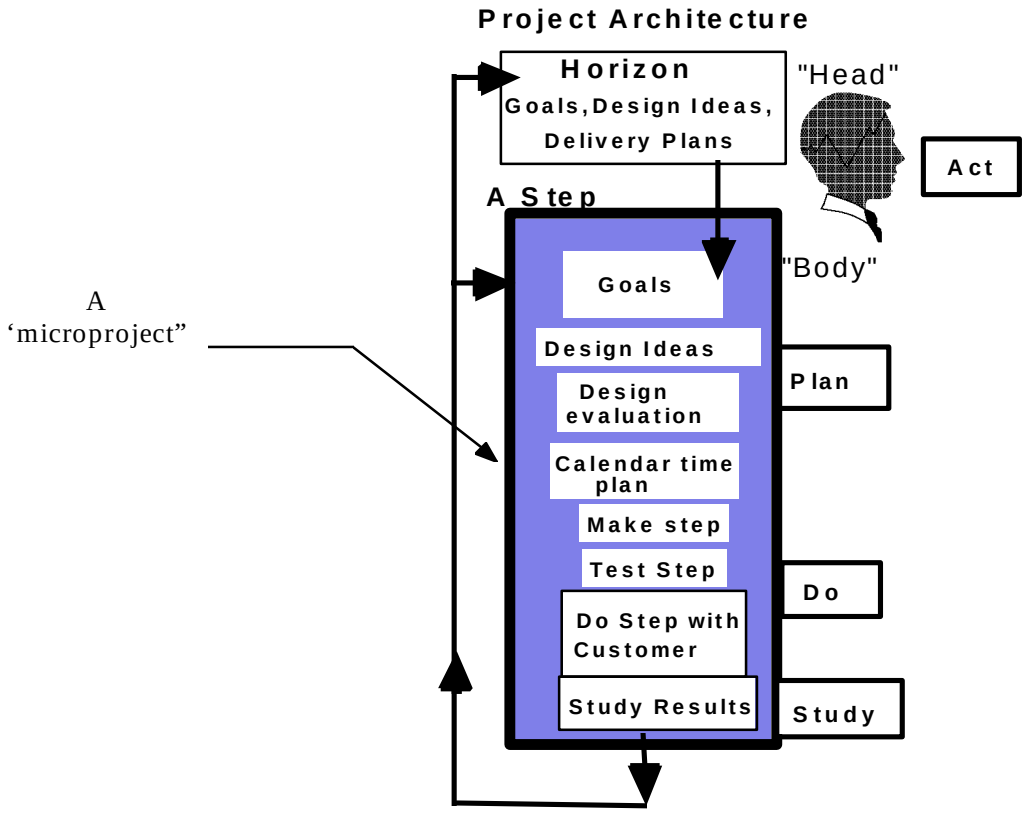


Figure 1: The Evolutionary Results Delivery process cycle is a process control cycle (Plan, Do, Study, Act) for project management.

9. Summary: Evolutionary Management

Evolutionary project management is probably the fastest *sure* way to get radically improved results. It does *not* mean 'slow and small change'. It means getting high priority improvements very early, by simply not worrying about the whole world at once.

It means *not* wasting years and millions with failed dream projects, which give nothing except weakened economy and reputation. It means proving that you know how to deliver really useful results, and through credibility, attracting resources to continue to do so. It is systematic engineering work [KOEN84] towards a grand vision of quantified improvements, using a grand architecture as a guide and an tool. It is like travelling to a remote destination one path at a time.

Evo management presumes that any large scale project can be systematically divided into a series of smaller mini-projects, each of which delivers some useful measurable result to the end user (or trial users at least). 'Small steps' is merely one possible strategy for reducing the risk and cost of failure. When large steps in terms of resources are necessary, then alternative strategies for controlling risk are to use conservative known technology, contract out the risk to others, or use insurance.

Evo management is, above all, the application of the Shewhart process control cycle, 'Plan, Do, Study, Act'. It is learning from doing and acting on that learning. It is adapting to the complex and changing realities of a project.

Incremental development is not the same as evolutionary management. Evo is also incremental. But incremental development is defined (US DoD MIL-STD-498[3], IEEE Std. 1498) as "determines … requirements then" … "performs the rest of the development in a sequence of builds". **Evolutionary**: "develops a system in builds, but differs in acknowledging that the user need is not

fully understood and all requirements cannot be defined up front. In this strategy, user needs and system requirements are partially defined up front, then are refined in each succeeding build."

Evo management primarily guides by well defined, quantified, but not necessarily static, multiple values of quality and cost. Deviation from the path to these central values is corrected with minimum loss of resource. Our ears are open for corrections to these central values, either because the world has changed, or because we better understand how to formulate our 'values'.[KEENEY92]

> *"The rationalistic approach is … characterized by the pretension to universality of its solutions, its intolerance of tradition and authority, quantification, simplification, and lack of flexibility. Its very efficiency prevents flexibility by eliminating what does not contribute to achieving the current objective so that alternative means are not available if the objective is changed"*
>
> *British Defense analyst Gregory Palmer <- [MINTZBERG94:120*

It is my hope that the evolutionary method will be characterized by humility and constant openness to alternative means, to what is proven to work, to moving towards the *long* term objectives, in short term increments. It should be characterized by sensitivity to change of objectives, and consequent need for alternatives to satisfy that change.

History shows that the best-intended methods might not work as well as intended, and can be counterproductive [MINTZBERG94, MORRIS94]. This is outside this author's control. The reader is hereby charged with the responsibility for sane and practical use of the ideas here!

References

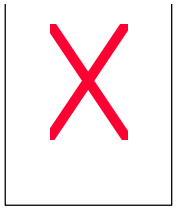
COTTON96: Cotton, Todd, "Evolutionary Fusion: A Customer-Oriented Incremental Life Cycle for Fusion".

Hewlett-Packard Journal, August 1996, Vol. 47, No. 4, pages 25-38. This is adapted from the book Object-oriented Development at Work: Fusion in the Real World, by Ruth Malan et al (Eds.), Prentice Hall PTR, 1996.
An excellent detailed view of the Evolutionary project management process as taught Corporate-Wide at HP. The author was on a project team at HP in 1989 which Gilb taught early versions of the Planguage method. See another member of that Project in MAY96.
Available as pdf Adobe Acrobat file at http://hpcc920.external.hp.com/hpj/aug96/augart3.htm

HP Journal subscription free to qualified individuals: Write Distribution Manager, HP Journal, M/S 20BH, 3000 Hanover Street, Palo Alto, CA, USA-94304, or Email: hp_journal@hp_paloalto-gen13.om.hp.com. HP Journal is available on World Wide Web at "http://www.hp.com/hpj/journal.html". Author Email:Todd_Cotton@HP.com.

CUSUMANO95: Michael A. Cusumano and Richard W. Selby.: "Microsoft Secrets : How the World's Most Powerful Software Company Creates Technology, Shapes Markets, and Manages People",

The Free Press (div. of Simon and Schuster), 1995,
ISBN 0-02-874048-3, 512 pp.
This book makes it clear that the evolutionary method is a dominant method at several levels of technology and marketing at Microsoft. Most dramatic is the daily 5 PM software build cycle, but it extends to much larger cycles too. The depth of factual insights in this book is such that no amount of quotes can fully bring out the detail of Microsoft's organizational practices. Truly a handbook for the manager who wants practical ideas for their organization.



Author Email: Selby@amadeus.com

DoD Mil-Std-498 Dec 94

Replaced by IEEE doc 12207 (This used to be DoD 498)

GILB97Evo: Gilb, Tom, "Evolutionary Project Management", Unpublished book manuscript, 1st draft 1997, available on web site www.result-planning.com

GILB97RDM: Crosstalk (DoD US) June 1997 "Requirements-Driven Management: A Planning Language".

Electronic downloadable copies at STSC site.
http://www.stsc.hill.af.mil see Crosstalk
http://www.stsc.hill.af.mil/Crosstalk/1997/jun/jun97ind.html
page 18-42

JARKVIK94: Järkvik, Jack & Lars Kylberg et al: 'Om att Lyckas' ('On Succeeding'): 80 pages.

Published December 1994 by Ericsson, Kista, Sweden for internal use. By Mr. Stellan Nennerfelt. It describes a project of delivering a base station product to Japan on a tight delivery schedule in 9 months in about 9 evolutionary stages. This is very different project management technology from conventional Ericsson waterfall model methods. The authors have explained the deeper philosophy behind the method. An english language edition ('On Succeeding') was published (EN/LZT 123 1987) in 1994.
Publisher at Ericsson Stellan Nennerfelt Vice President, Operational development,
Torshamnsgatan 23, Kista S-164 80, Telephone 757 05 32, fax 757 14 42, Mobile 010-214-37-36
Memoid: eri era eranen

KOEN84: Koen, Billy V., 'Towards a definition of the Engineering Method', Engineering Education, December 1984.

MAY96: Elaine L. May and Barbara A. Zimmer, "The Evolutionary Development Model for Software Hewlett-Packard Journal, August 1996, Vol. 47, No. 4, pages 39-45.

This is an excellent complimentary article to COTTON96. Elaine attended a Gilb course at HP in 1989.
It gives facts and figures for 10 Evo projects in 8 HP divisions.
Available as pdf Adobe Acrobat file at http://hpcc920.external.hp.com/hpj/aug96/augart4.htm
HP Journal subscription free to qualified individuals: Write Distribution Manager, HP Journal, M/S 20BH, 300 Hanover Street, Palo Alto, CA, USA-94304, or Email: hp_journal@hp_paloalto-gen13.om.hp.com. HP Journal is available on World Wide Web at "http://www.hp.com/hpj/journal.html".

MINTZBERG94: Henry Mintzberg. "The Rise and Fall of Strategic Planning: Reconceiving Roles for Planning, Plans, Planners", The Free Press, (A Division of Macmillian, Inc. New York, ISBN 0-02-921605-2, 458 pages, hardcover, $32.95.

MORRIS94: Morris, Peter W G, THE MANAGEMENT OF PROJECTS, Thomas Telford, London, 1994, 358 pages, ISBN 0 7277 1693 X, £55, Publisher 1 Heron Quay, London E14 4JD. Tel. 0171-987-6999, Fax 538-4101.

This extensive survey of planning methods and case studies of projects concludes that we do not have and have never had really good methods of getting control over projects! In a chapter on the 'new model' the concept of using more feedback and learning in the project management model emerges as a hypothesis. But, interestingly enough, the concrete success of using this method (at IBM FSD, Mills IBM Systems Journal, No. Four 1980) is not

known or cited. Morris also makes it clear that there seems to be a strong relationship between good clear requirements and project success.
Evolutionary notes on pages 66, 314,230, 289, 291, 218-221, 231, 300-301
Generally the author concludes that smaller steps of implementation are necessary to reduce and control project risk.

SORENSEN95: Reed Sorensen (Software Technology Support Center) "A Comparison of Software Development Methodologies", Crosstalk January 1995, pp12->18

Acquiring Crosstalk: custserv@hillwpos.hill.af.mil, 801-777-8045 (Customer Service).
World Wide Web site: "http://www.stsc.hill.af.mil/"
A quotation page 13 in this article referencing: "Guidelines for Successful Acquisition and Management of Software Intensive Systems: Weapons Systems, Command and Control Systems, Management Information Systems", September 1994.

This article compares Waterfall, incremental, spiral models, prototyping, Cleanroom and object-oriented techniques. It relates them to MIL-STD-498. DoD
Replaced by IEEE doc 12207.

SPUCK93: William J Spuck, The Rapid Development Method (RDM)

December 1993
Jet Propulsion Laboratory JPL
21 page paper., JPL D-9679 (internal document)
See Rapid Change Process Study (for NWS) Sept 1995
Their cycle is 9 months of 4 year projects.
William.H.Spuck-III@jpl.nasa.gov (William H Spuck)

Grand Master of chess, the Cuban, José Raul Capablanca Y Granperra (1888-1942, World Champion 1921-1927) was reported to have commented on people's interest in depth of search by this remark.

[1]From Lao Tzu in Bahn, 1980 (also quoted in Gilb, Principles of Software Enginering Management page 96)

[2]I have never seen an exception in 33 years of doing this with many varied cultures. Oh Ye of little faith!

[3]The definitions I was talking about appear in MIL-STD-498 of Dec 5 1994, appendix G page 47 and are from DODI 8120.2

1. "Grand design" (=big bang, waterfall)
2. Incremental "determines .. requirements then" "performs the rest of the development in a sequence of builds"
also called "Pre-planned Product Improvement"
3. Evolutionary: "develops a system in builds, but differs in acknowledging that the user need is not fully understood and all requirements cannot be defined up front. In this strategy, user needs and system requirements are partially defined up front, then are refined in each succeeding build.
See also figure 11 page 52 "One possible way of applying mil std 498 to the Evolutionary program strategy.
{ PRINT \p PAGE "I recall other passages which made it clear that this standard allowed you to commission work without having all the requirements in place in advance, precisely to allow contracting for evolutionary delivery.