Designing To Schedule: Combining Critical Chain Planning and Incremental Development in Software Projects

SWT05

Eduardo Miranda
Ericsson Research Canada

PMI®
Global Congress
EUROPE 2004

Agenda

- Defining hard deadlines
- Critical issues in projects with hard deadlines
- The fundamentals
- Combining Critical Chain Planning with Incremental Development
- Summary

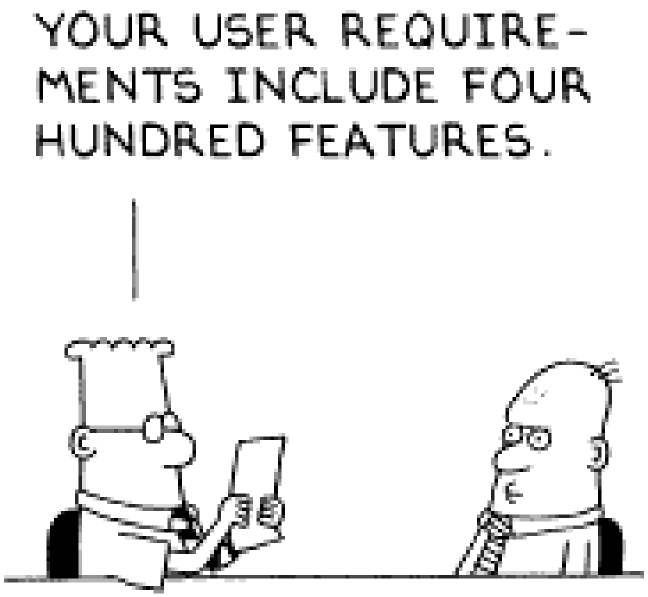
Defining hard deadlines

- Trade shows, campaign launches and regulatory dates are a reality.
- In the cases above, and in many others, the date of delivery is as important as the delivery itself.

Critical issues in projects with hard deadlines

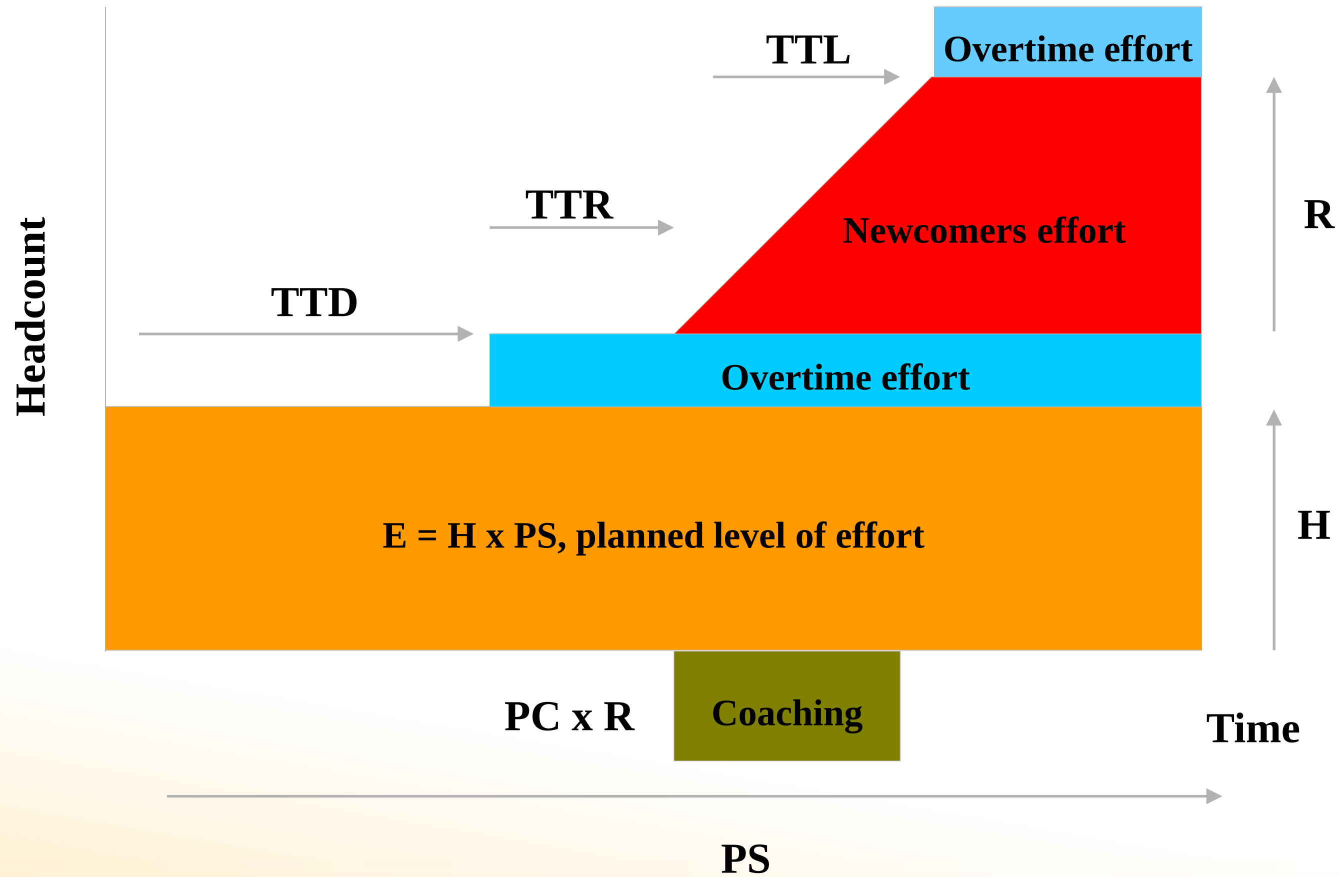
- Scope of the project
- Allowances for variations on the execution of the tasks that made up the project
- Assessing remaining work

Dealing with scope



- It is no the same…
  - To start with a twelve months project than to start with a six one, that is latter extended by an additional six.
  - To start with a small product than cutting in half a large product by the middle of the project to meet the deadlines.
  - To start a project with the right amount of people than to add resources anytime after.

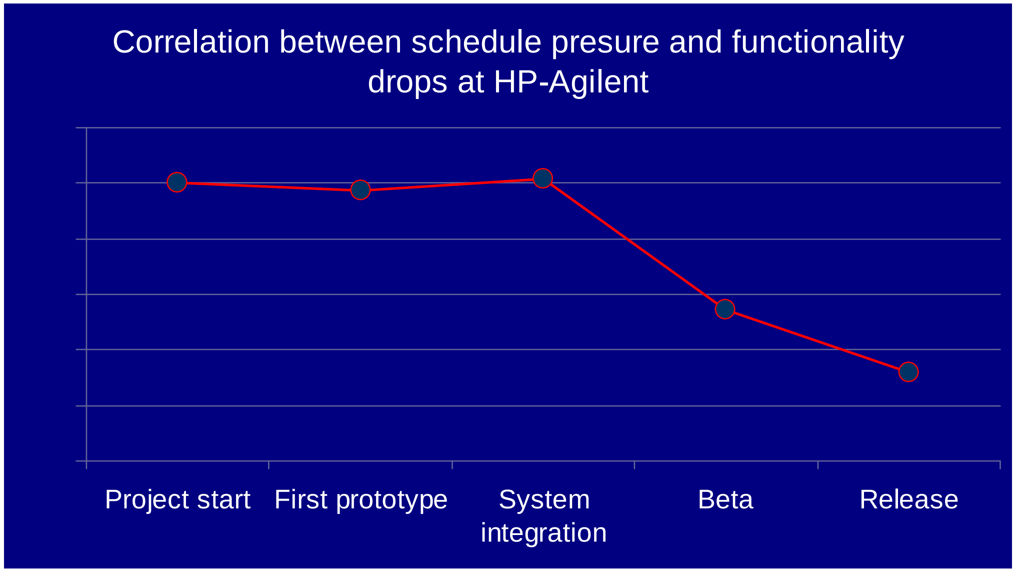
The consequences of starting with the wrong estimate



Definitions

- TTD (Time To Delay). The time it takes to the project team to realize/accept that the project is going to be late and they need to do something.
- TTR (Time To Recruit). The time it takes to get additional resources after the decision has been made.
- TTL (Time To Learn). The times it takes to a new comer to became fully functional in the project.
- H. Original project headcount
- R. Additional resources
- PS. Planned schedule
- PC. Percentage of time devoted to coaching for each new resource brought late into the project.

How do different companies deal with the scope issue?



Development Phase Feature development in 3 or 4 sequential subprojects that each results in a milestone release

Program managers coordinate evolution of specification. Developers design, code, and debug. Testers pair with developers for continuous testing.

- **Subproject I** First 1/3 of features (Most critical features and shared components)
- **Subproject II** Second 1/3 of features
- **Subproject III** Final 1/3 of features (Least critical features)

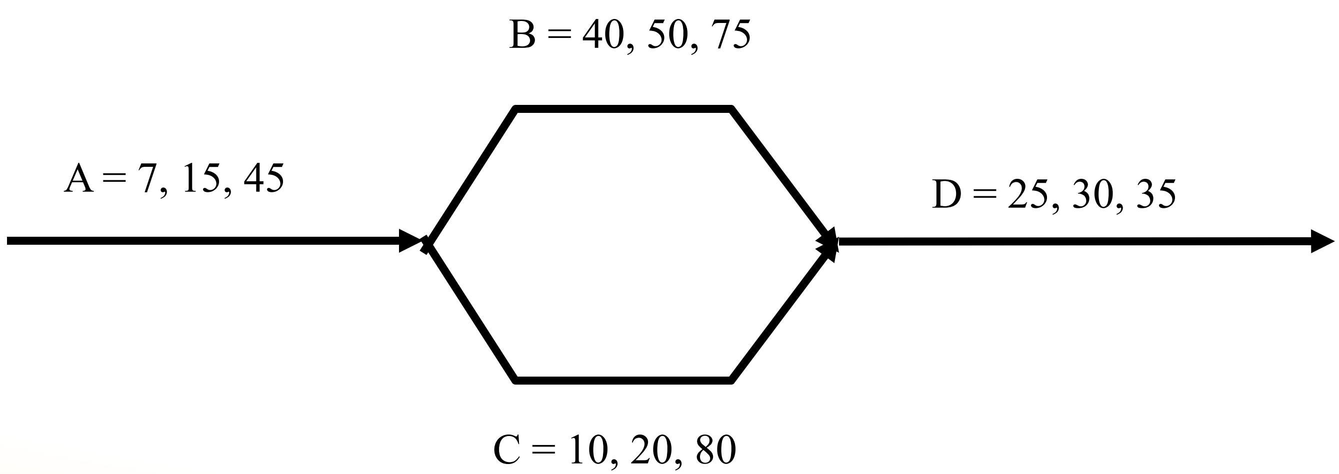
•After Rapid And Flexible Product Development: An Analysis Of Software Projects At Hewlett Packard And Agilent by Sharma Upadhyayula, MIT, 2001

•How Microsoft Builds Software, M. Cosumano and A. Selby, Communications of the ACM, 1997

Schedule allowances

- Allowances to compensate for:
  - Errors in estimations
  - Number of un-planned iterations
  - Activity familiarity
  - Team capacity
  - Unknown unknowns
- How big and where should they be located?

What is a realistic completion date for the project if we are not sure when each individual tasks will be finished?



How different organizations deal with uncertainty?

Knowledge Areas	*Planning Processes*	*Use*
Integration	Project plan development	4.0
Scope	Scope planning Scope definition	4.1 3.6
Time	Activity definition Activity sequencing Activity duration estimating Schedule development	4.1 3.4 4.2 4.0
Cost	Resource planning Cost estimating Cost budgeting	3.7 3.0 3.2
Quality	Quality planning	2.9
Human Resources	Organizational planning Staff acquisition	3.8 3.6
Communications	Communication planning	2.3
Risk	**Risk management planning** **Risk Identification** **Qualitative risk analysis** **Quantitative risk analysis** **Risk response plan**	**2.2** **2.8** **2.0** **2.3** **2.3**
Procurement	Procurement planning Solicitation planning	

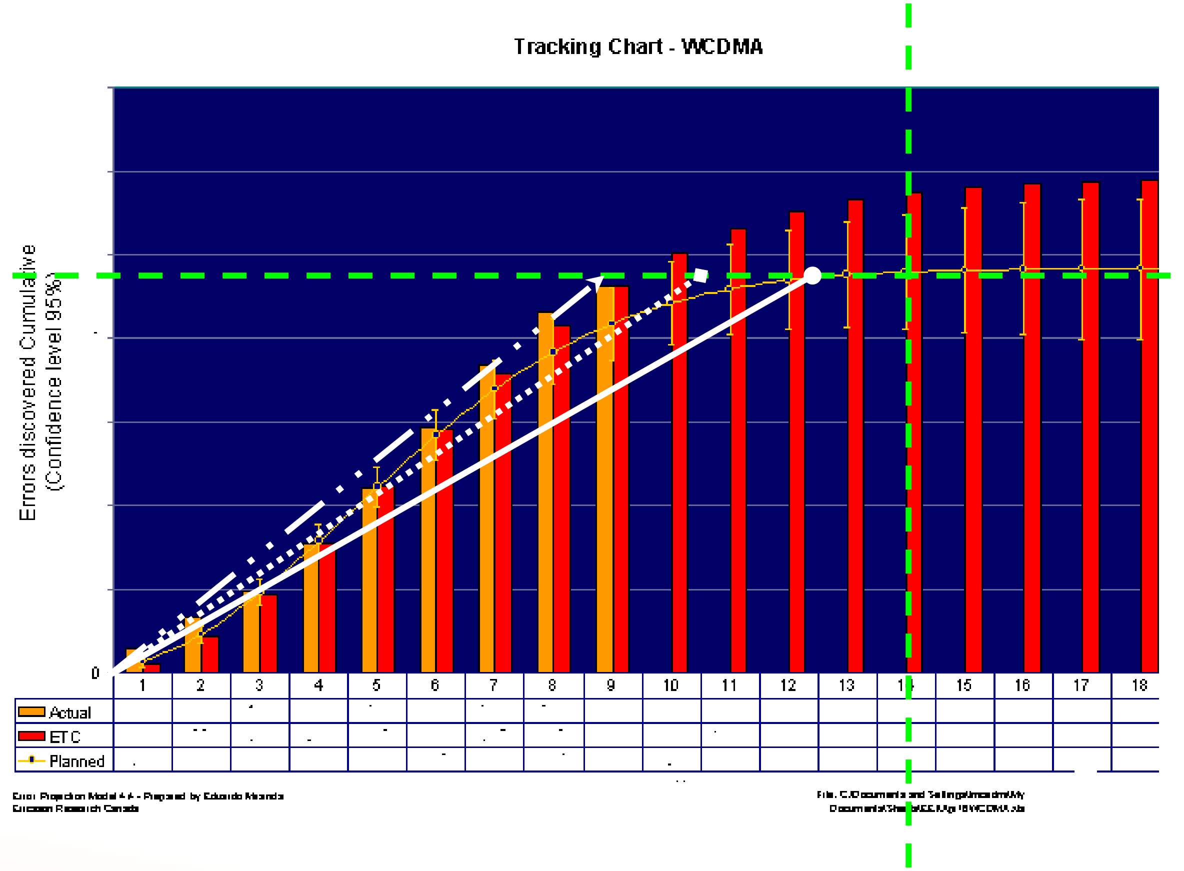
- Denial/wishful thinking
- Padding estimates
- Pert & Monte Carlo approaches
- Critical Chain Planning

Scale: - Always used / Most mature; 1- Hardly ever used / Least mature

PM Knowledge Areas	EC	IMM	IS	HTM	All 38 Companies
Scope	3.52	3.45	3.25	3.37	3.42
Time	3.55	3.41	3.03	3.50	3.37
Cost	3.74	3.22	3.20	3.97	3.48
Quality	2.91	3.22	2.88	3.26	3.06
Human Resources	3.18	3.20	2.93	3.18	3.12
Communications	3.53	3.53	3.21	3.48	3.44
Risk	2.93	2.87	2.75	2.76	2.85
Procurement	3.33	3.01	2.91	3.33	3.14
Overall PM Knowledge Areas Maturity	3.34	3.24	3.02	3.36	3.24

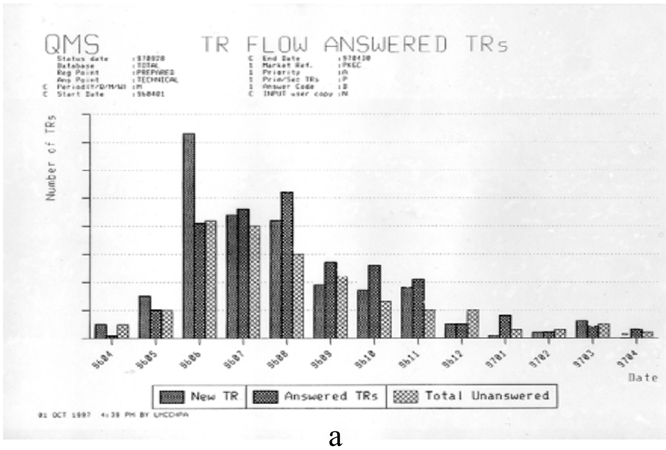
•The impact of the project manager on project management planning processes, S. Globerson and O. Zwikael, Project Management Journal, Sep. 2002
•Y. Kwak and C. Ibbs, PMI 28th Annual Seminars & Symposium, 1997

Assessing remaining work

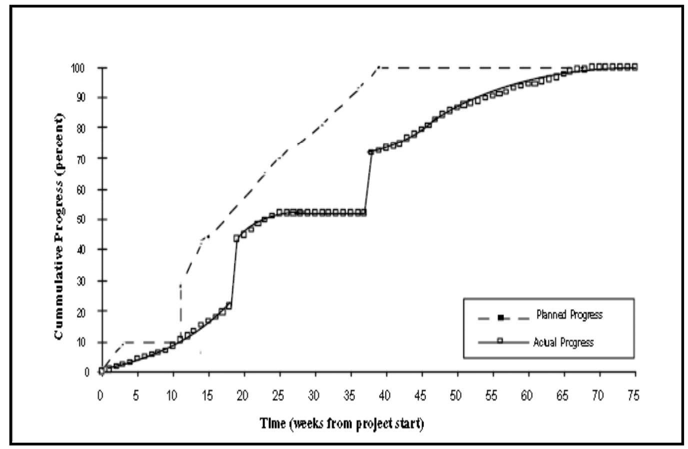


Target date

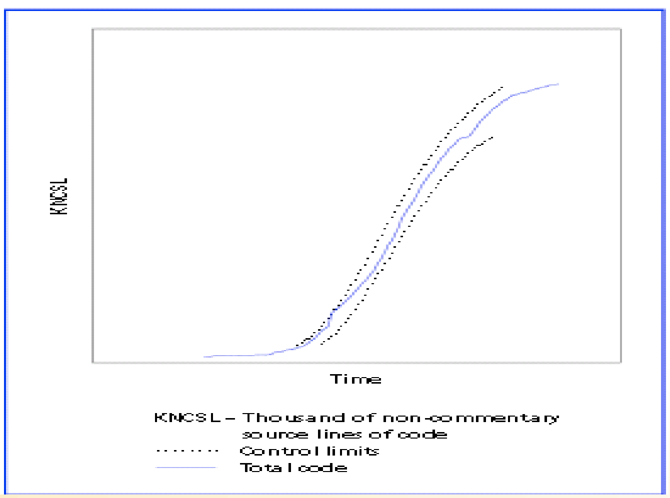
Work does not seem to progress at a constant rate



a



b



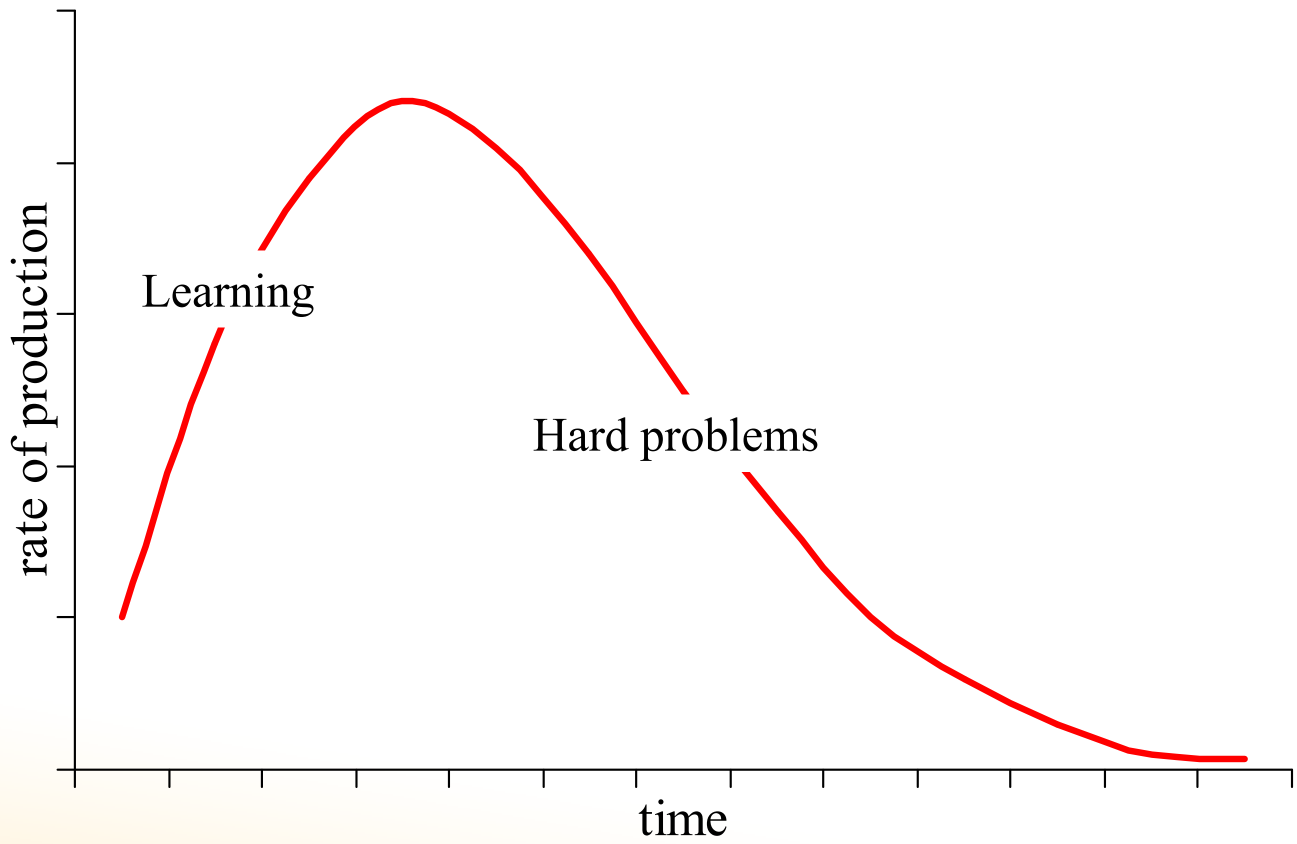
c

a) AXE Switch, error discovery pattern. Ericsson, 1997

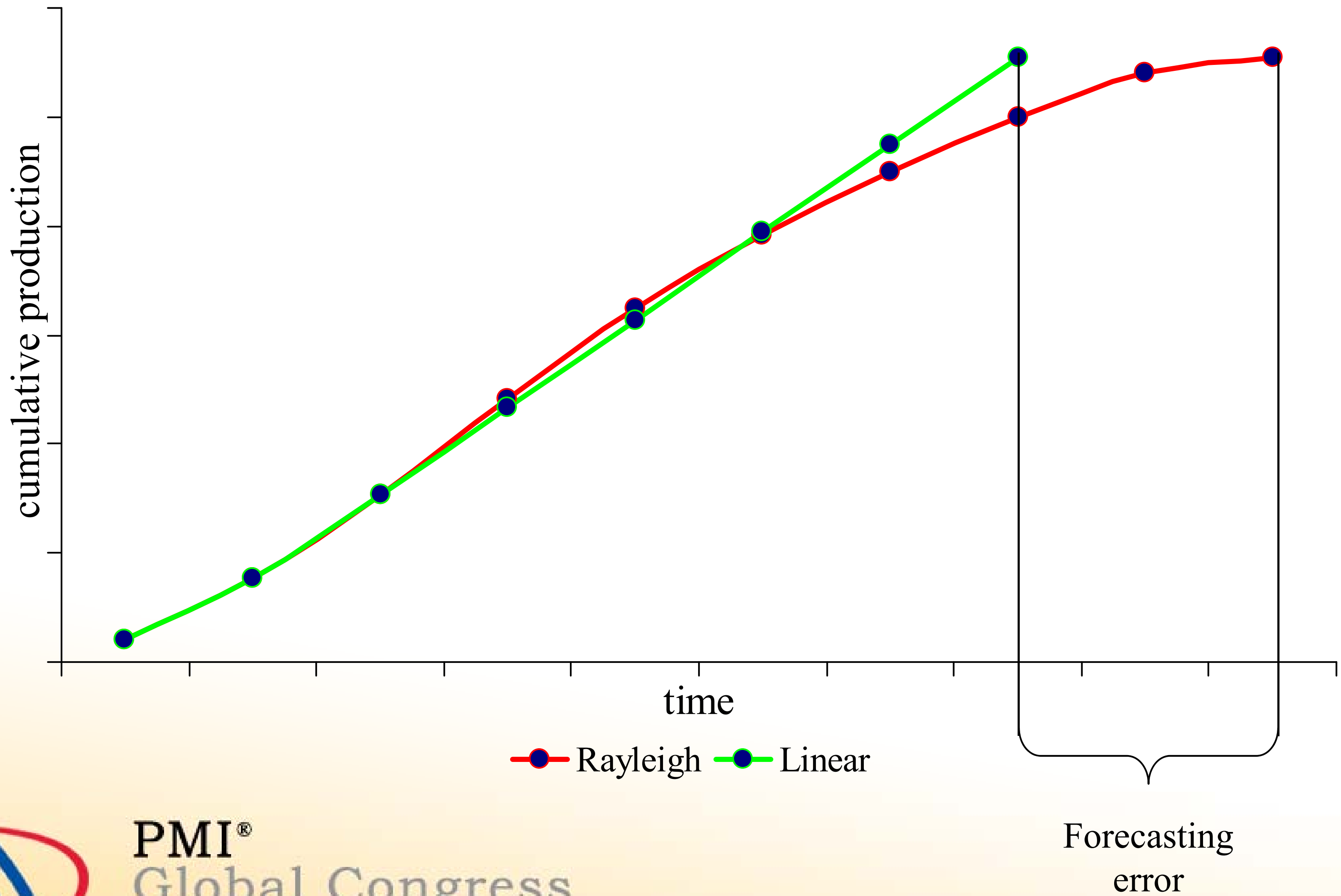
b) Python Project. Semiconductor development project. Reported by Ford and Sterman in Overcoming the 90% Syndrome: Iteration Management in Concurrent Development Projects.

c) 5ESS-2000 Switch, code production pattern, Lucent 1997

Activity characteristic curve



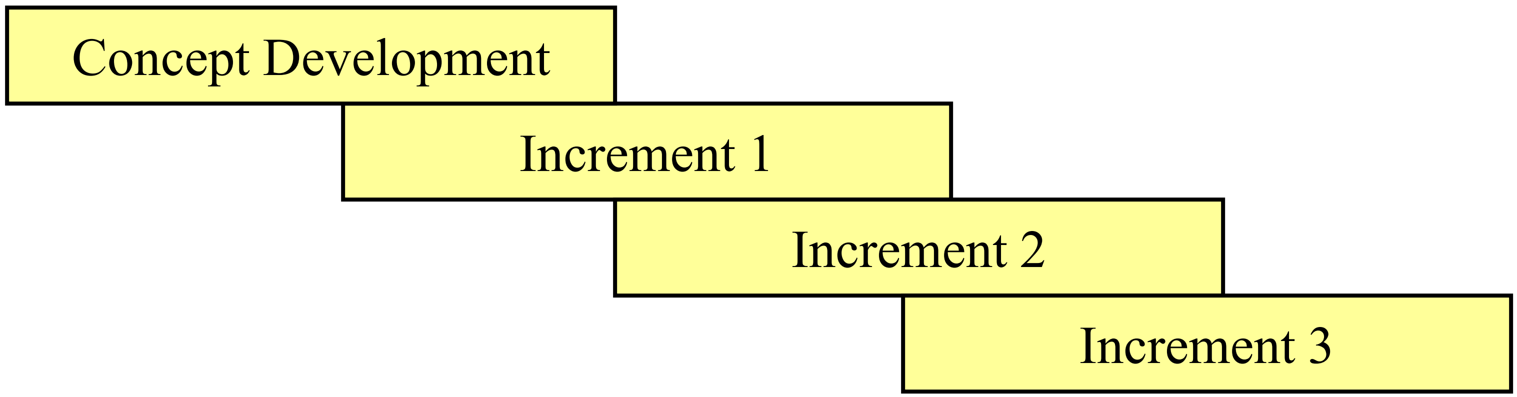
Forecasting task completion



Fundamentals

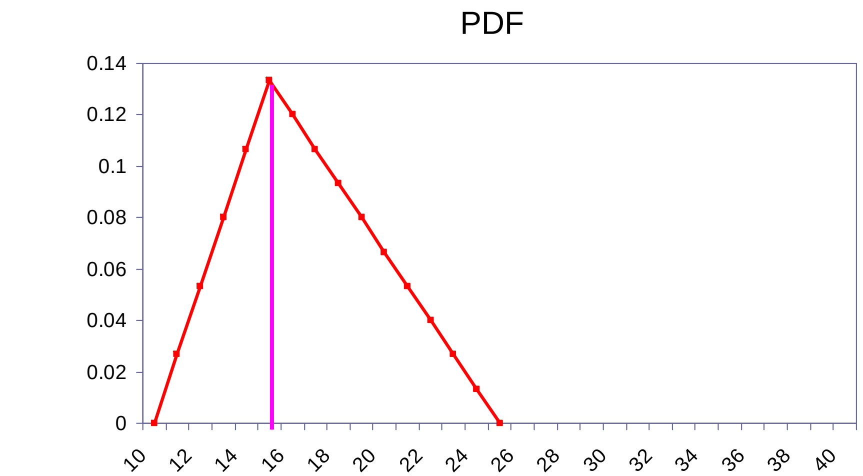
- The incremental approach
- Probabilities as a measure of the strength of a belief in an estimate
- Problems with traditional planning
- Critical Chain
  - Dealing with uncertainty
  - Resource conflicts & multitasking
  - Buffer management

The incremental approach



- Each increment includes a functionally complete set of requirements
- Each increment delivers a working system from the user perspective
- How big should an increment be?
  - Microsoft's criteria for defining increments is 1/3, 1/3, 1/3 of the total scope
  - Nortel CliP's criteria for defining increments is feature sets important to the customer

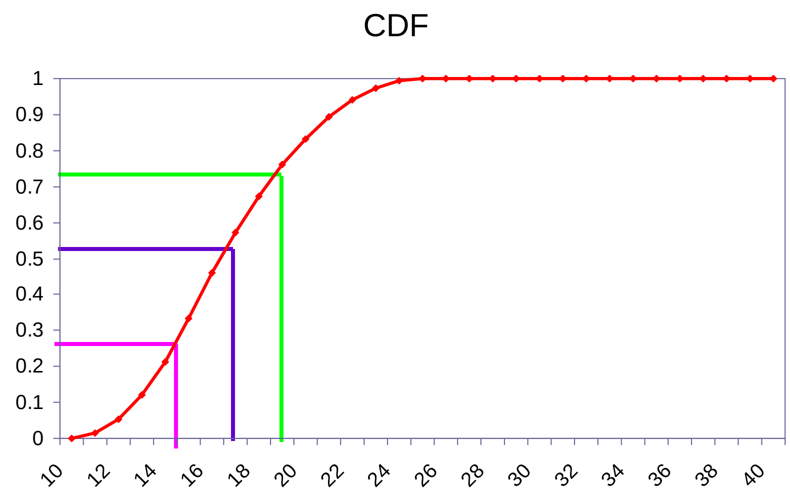
Probabilities as a measure of the strength of a belief in an estimate



Earliest Completion Date

Planned Completion Date

Latest Completion Date

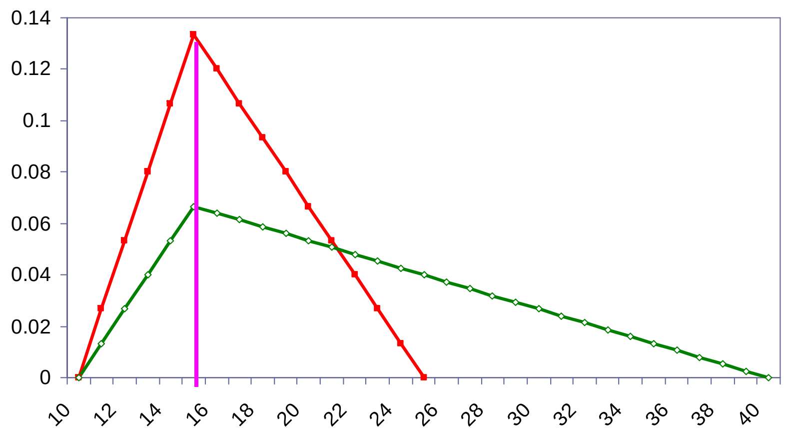


Planned Completion Date

Expected Completion Date

75% On-time Probability Date

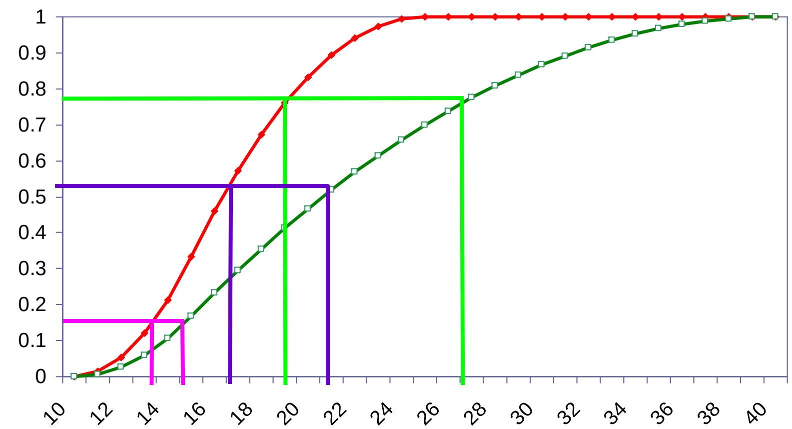
The standard deviation (σ) of the distribution measures the level of uncertainty



Earliest Completion Date

Planned Completion Date

Latest Completion Dates



Planned Completion Date

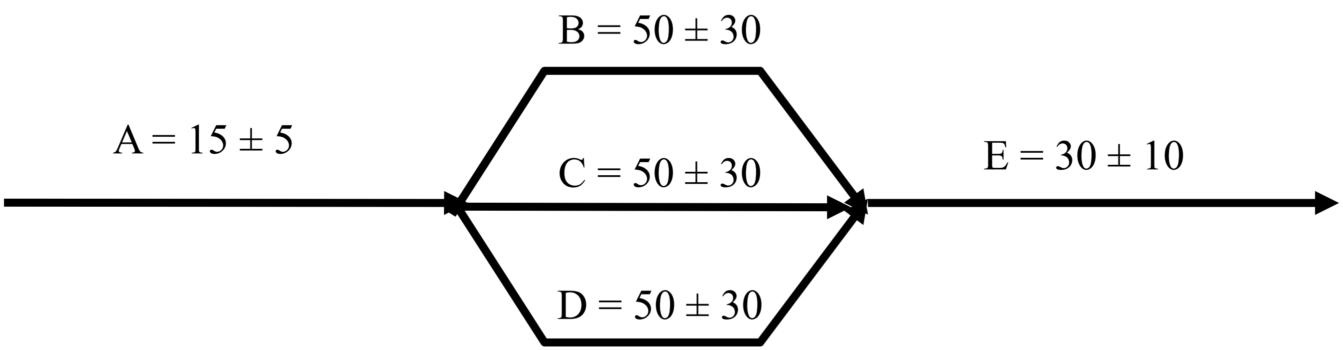
Expected Completion Dates

75% On-time Probability Dates

Problems with traditional critical path calculations and planning

- Merging paths
- The independence assumption
- Task level contingencies

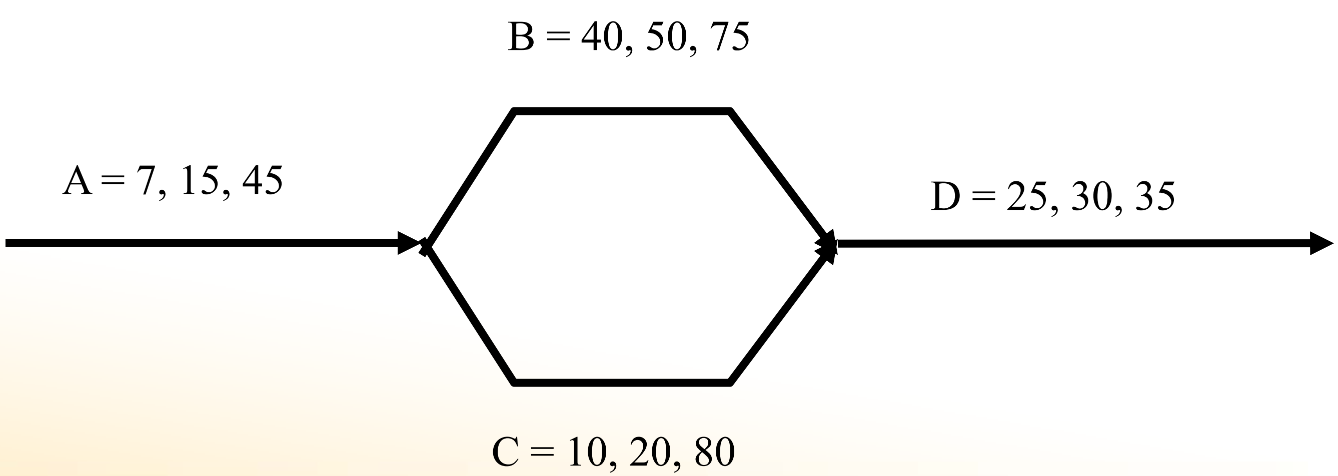
Merging Paths



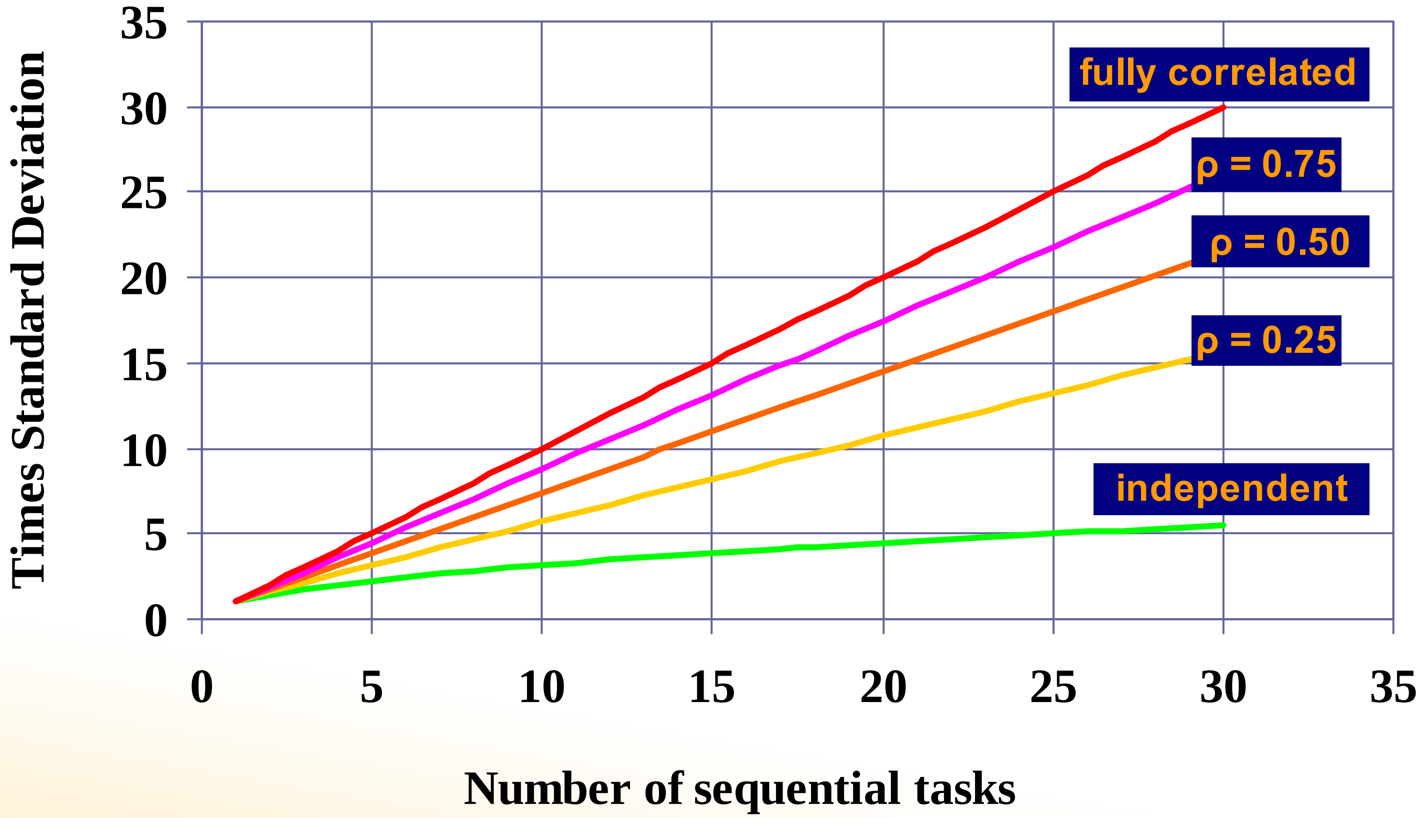
Path merging acts like a filter that eliminates positive fluctuations, and passes on the longest delay.

The independence assumption

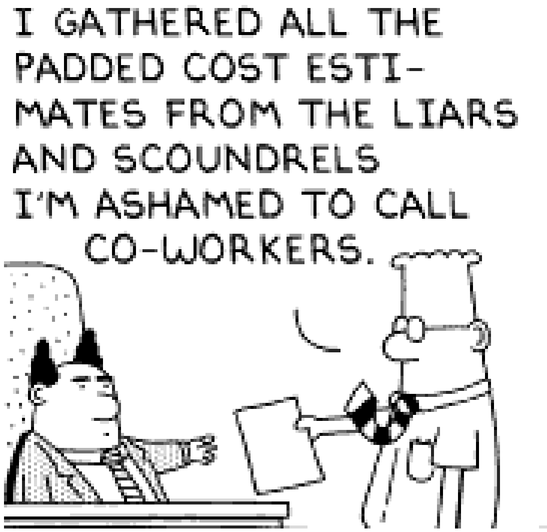
- It is generally assumed, that the duration of the tasks in a project are independent, so if one takes a little bit longer others might take a little bit less and in the end everything will be compensated. This assumption is correct, unless there is an underlying cause linking those tasks.
- If the tasks are correlated, all durations tend to shift in the same direction

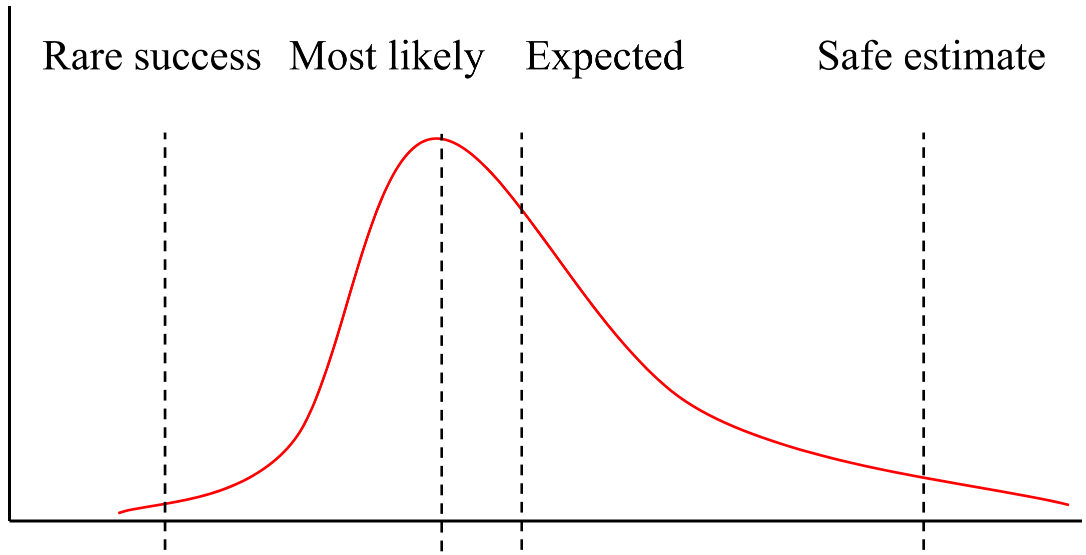


Effect of a common cause in the amount of risk



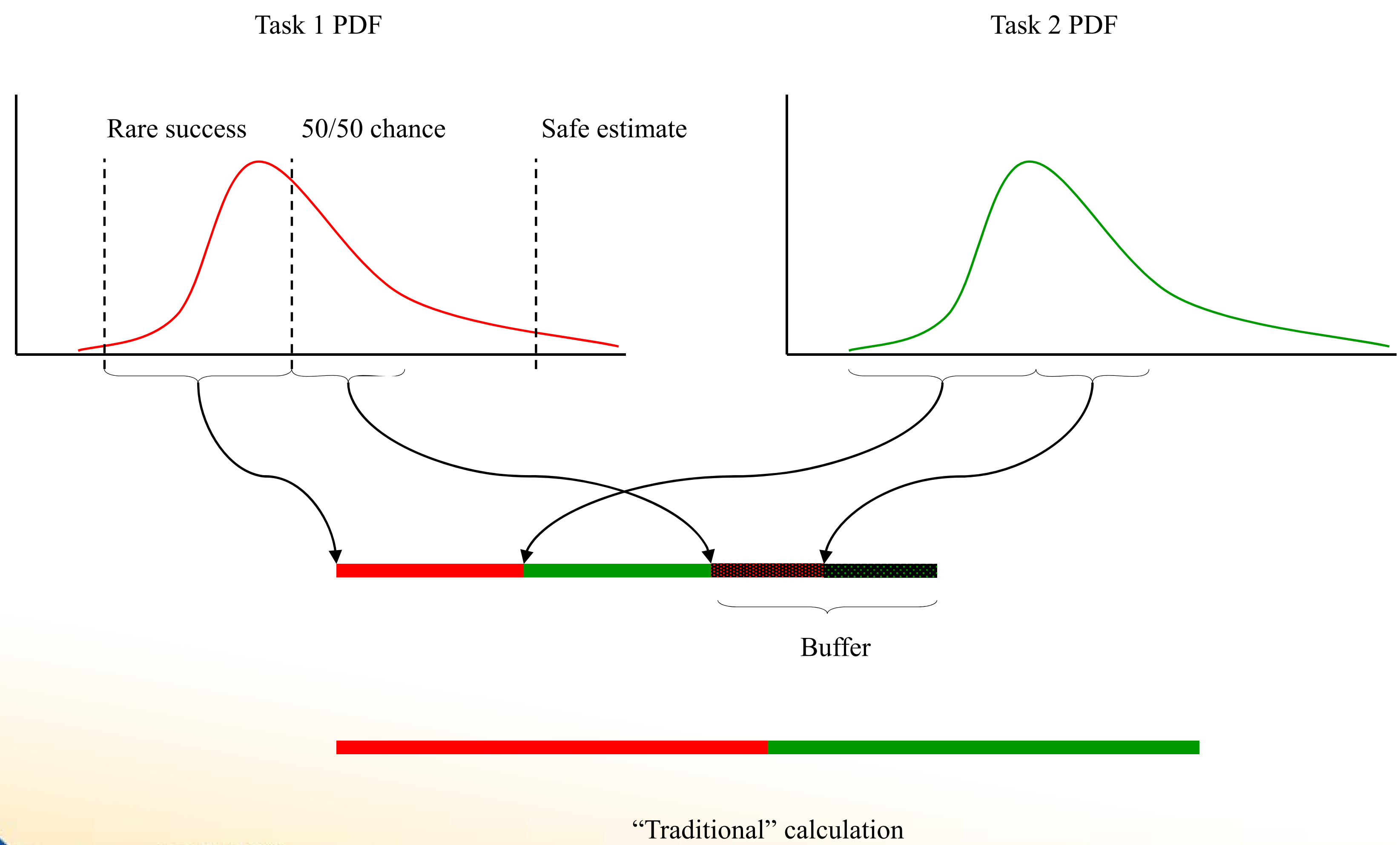
Task level contingencies



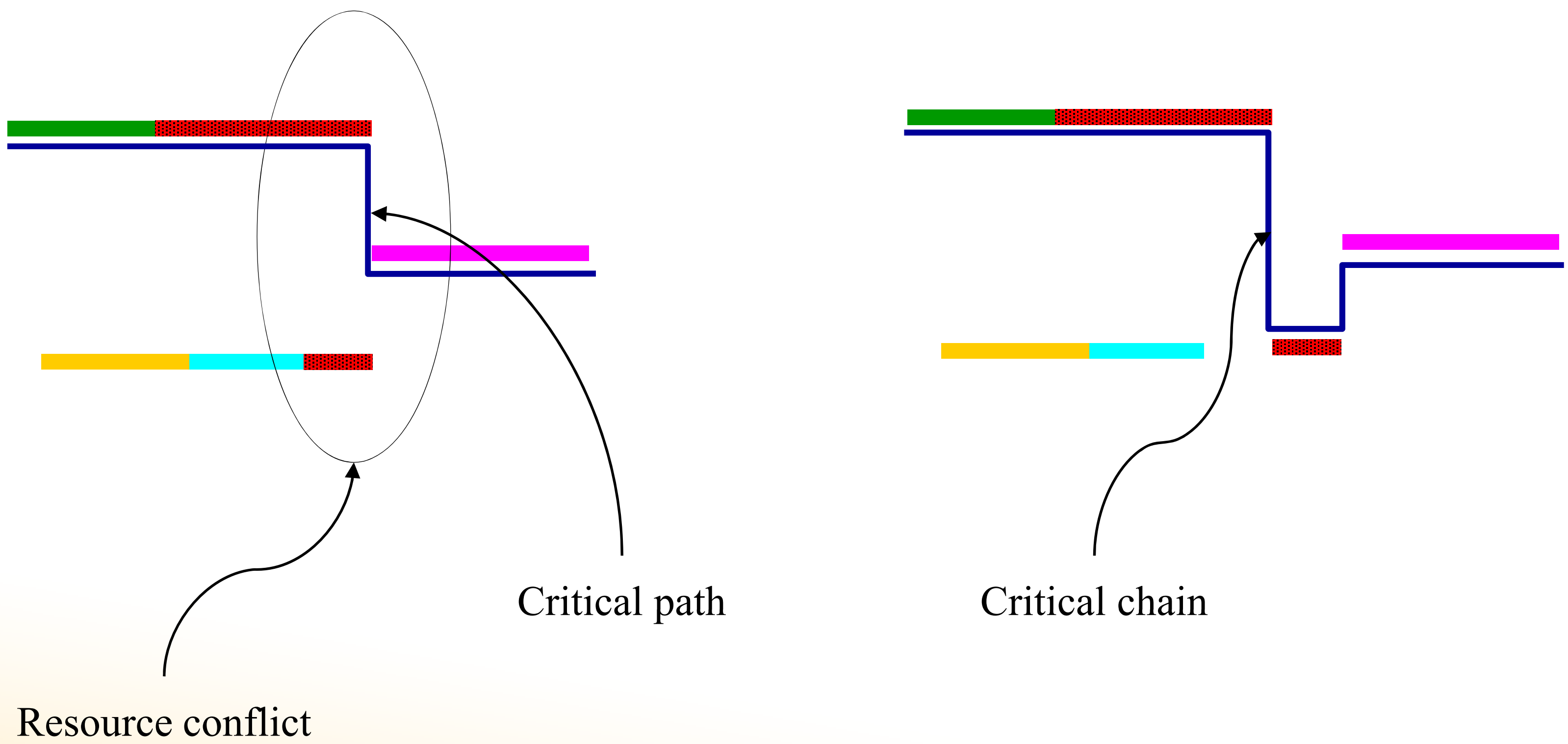


Which one of the four above values do you use for scheduling?

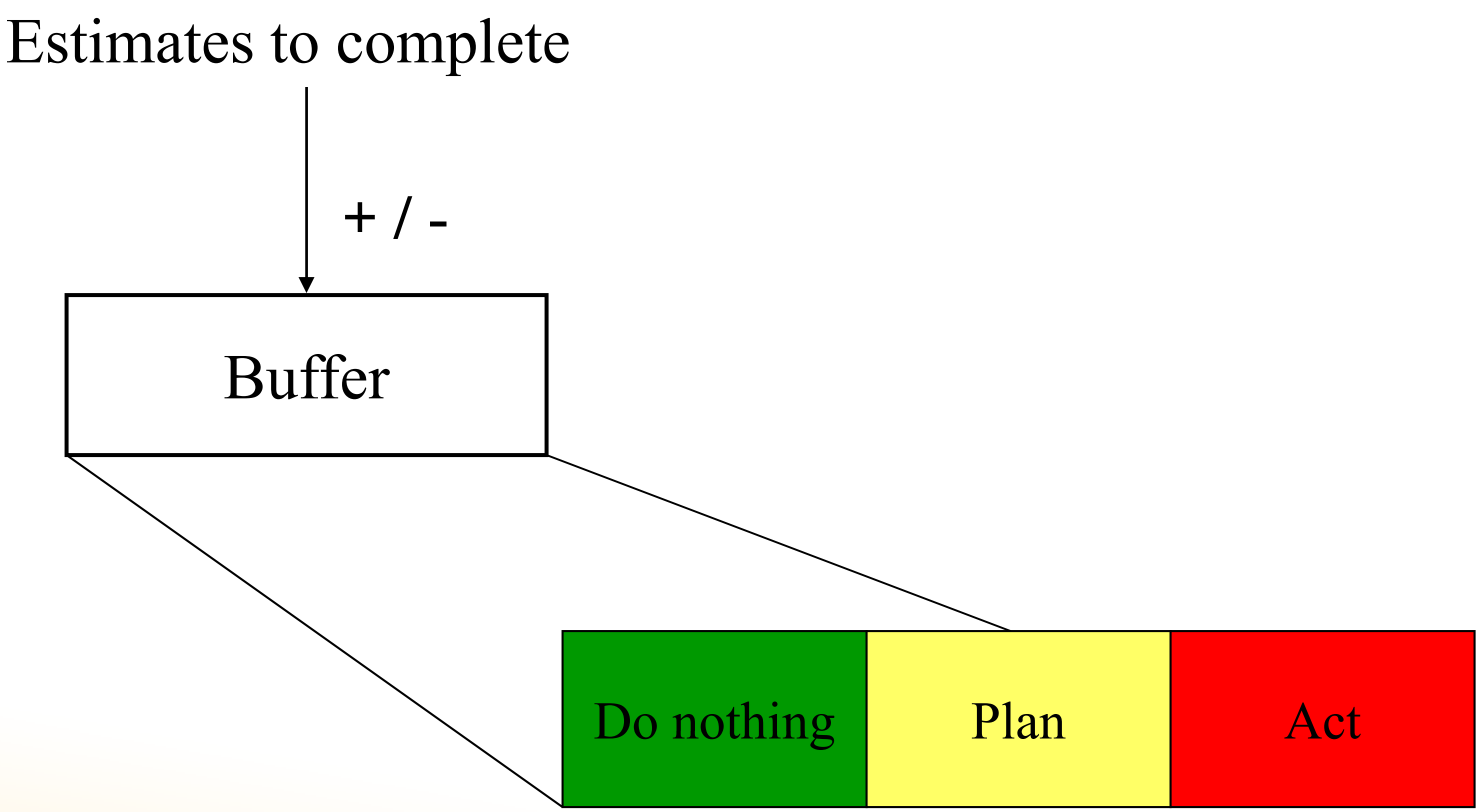
Dealing with uncertainties in CC



Resource conflicts in CC



Buffer management in CC



Pros and cons of Critical Chain Planning

+ Critical chain, The longest sequence of dependent tasks or resource usage
+ Make safety explicit
+ Aggregate all safety into project and feeder buffers
+ Control the project by monitoring the buffers
+ Plan resource readiness alerts along the critical chain

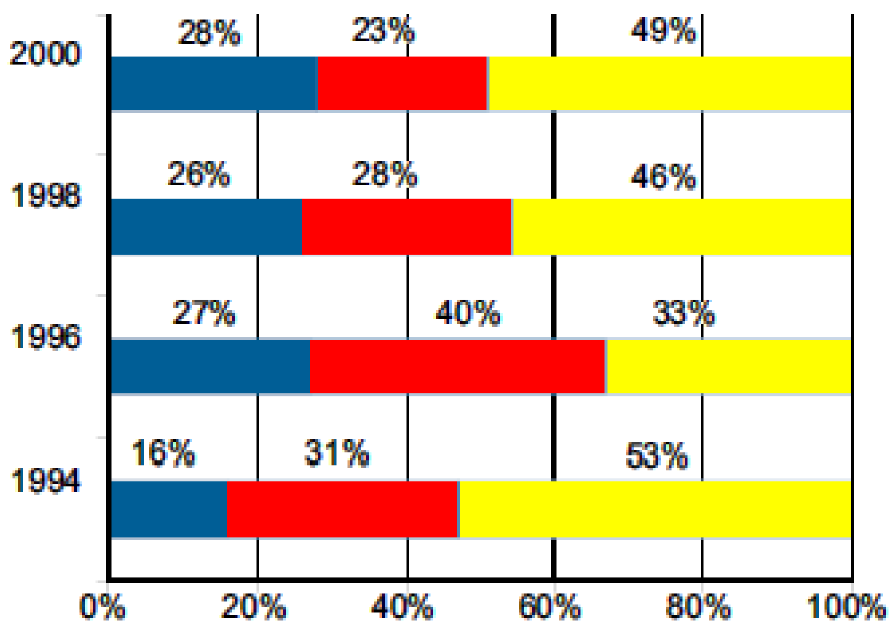
– Goldratt's method is based on the idea that everybody introduces a lot of safety on their estimates

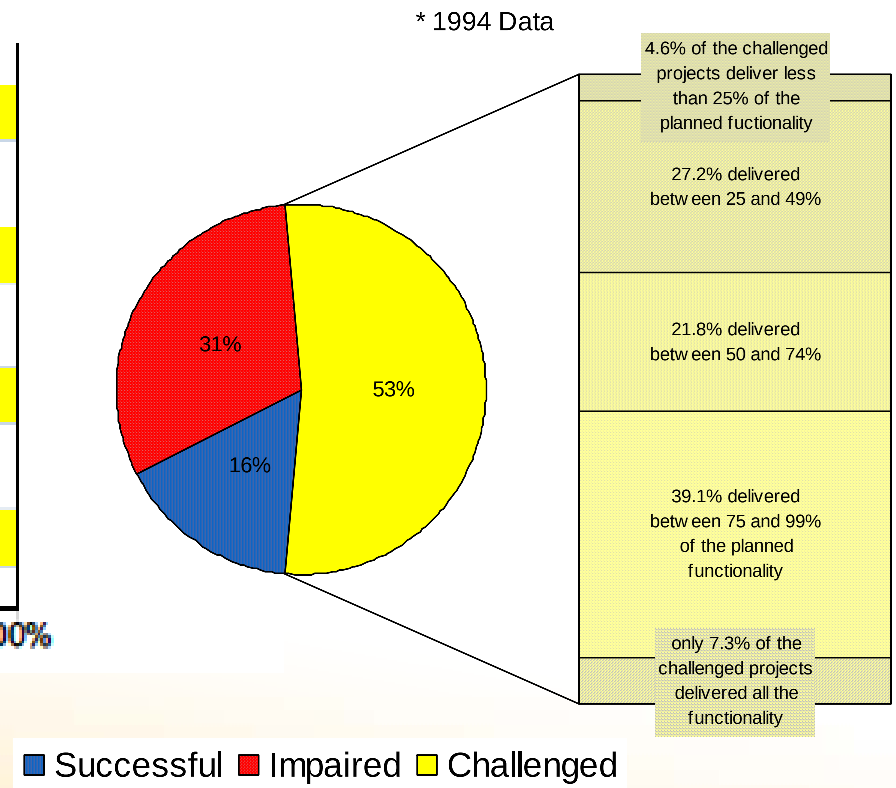
– Does not account for correlated tasks

Still there is a problem

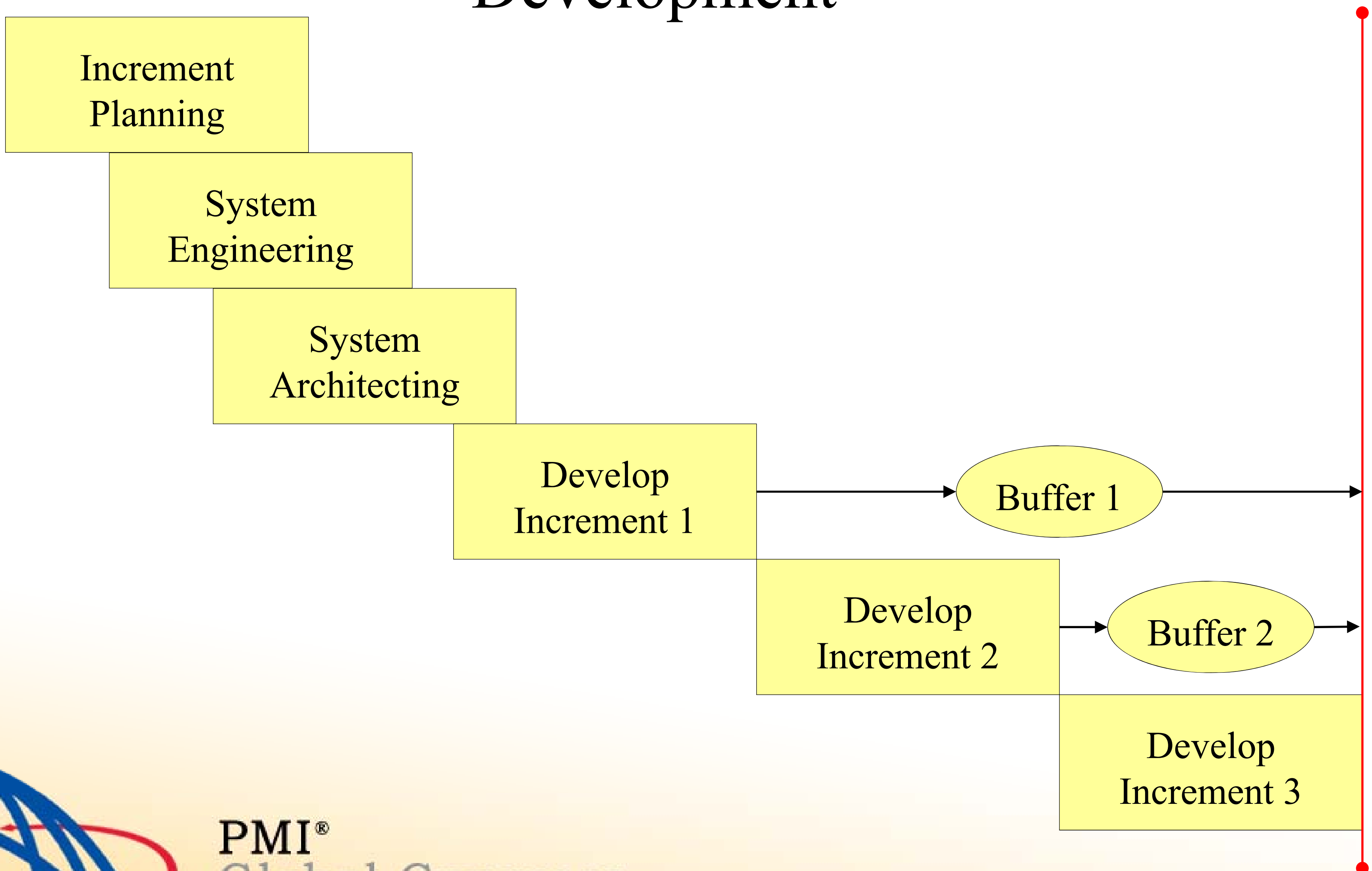
- **The Standish Group research shows that in the year 2000, 49% of the projects were over-budget, over the time estimate, and offer fewer features and functions than originally specified.**

Project Resolution History (1994–2000)





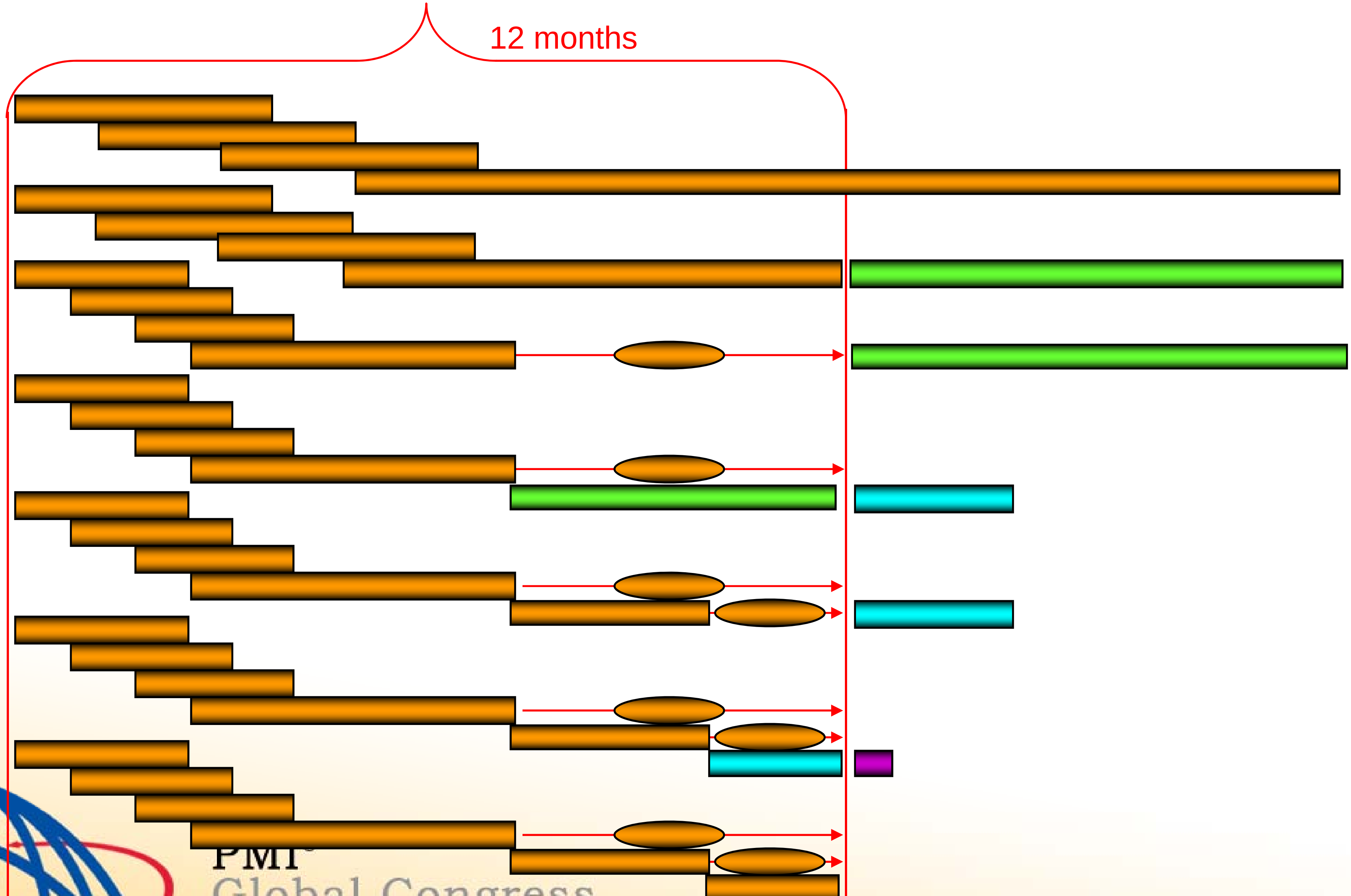
Combining Critical Chain and Incremental Development



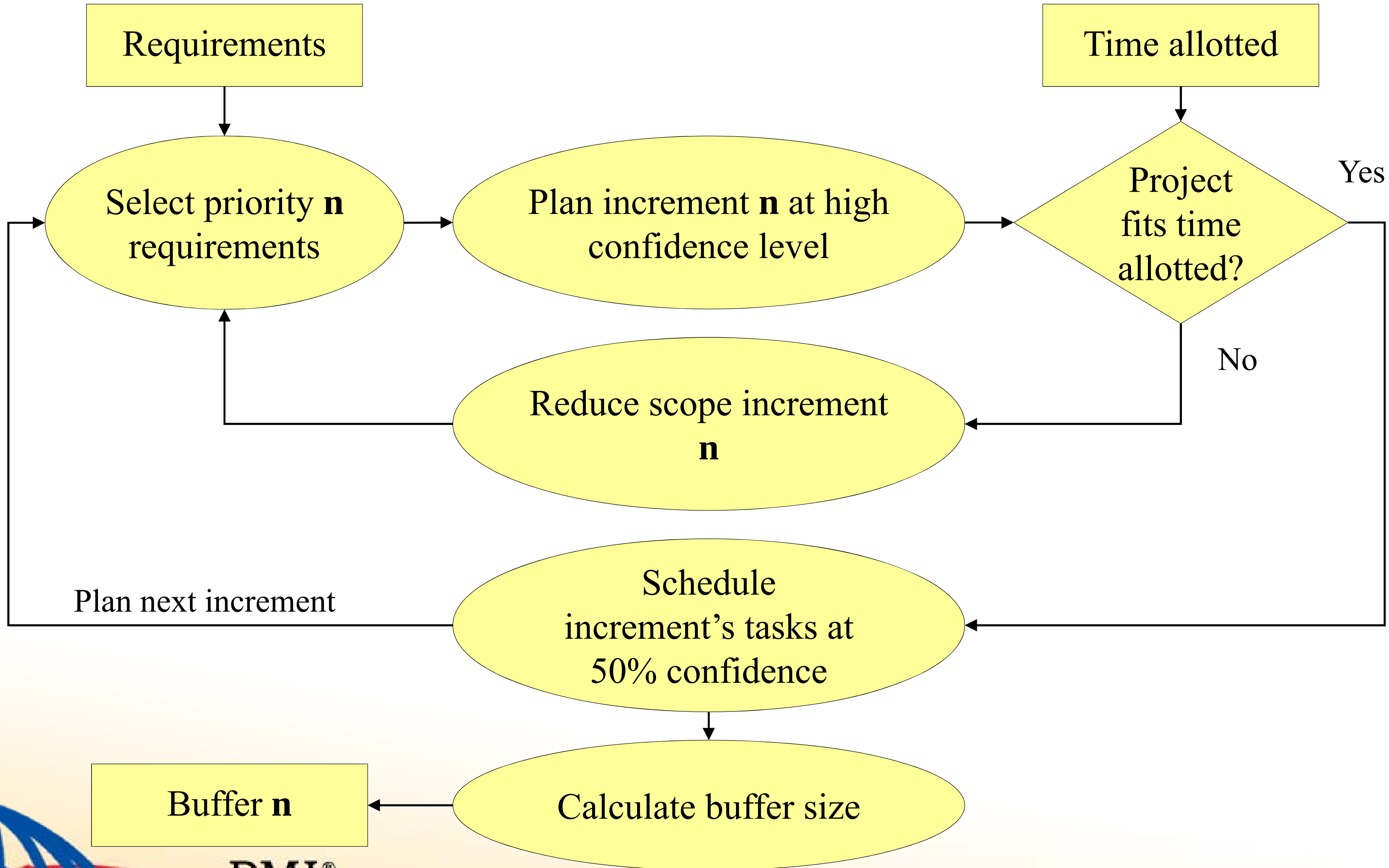
Increment Characteristics

- Each increment includes a functionally complete set of requirements
- Each increment delivers a working system from the user perspective
- The content of each increment is defined together by the Product Manager, the Project Manager & the System Architect
- All the project team works in one increment at a time
- Work on a second increment is not started until the previous increment is finished
- The completion of each increment is tied to a reward objective

Planning example



Planning process



PMI®
Global Congress
EUROPE 2004

Planning example using MinimumTime

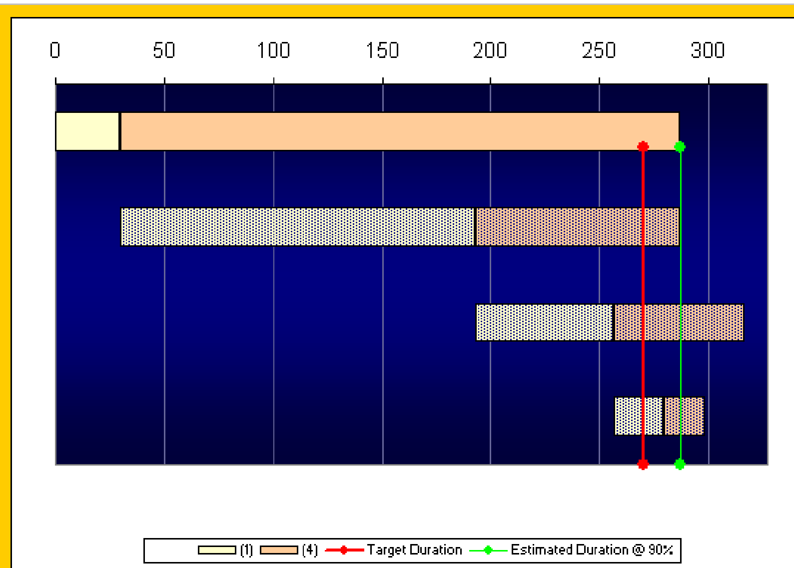
Include in estimated duration?	Task	Minimum Duration	Most Likely Duration	Maximum Duration	Lag (+/-)	Influenced by	Degree of Correlation
☑	Planning	20	30	40			
☑	Increment 1	100	150	240		Planning	0.5
☐	Increment 2	30	40	120			
☐	Increment 3	10	20	40			
☐							
☐							
☐							
☐							
☐							
☐							

Target Duration: 270

Desired Safety Level: 90%

Calculate

Clear



Details

Task	Expected (1)	Std.Dev. (2)	Lag (3)	Start Pred(1)+(3)	Buffer (4)
TRUE	30.00	4.08	0.00	0.00	256.95
TRUE	163.33	28.96	0.00	30.00	93.62
FALSE	63.33	20.14	0.00	193.33	60.42
FALSE	23.33	6.24	0.00	256.67	18.71
	Expected	Std. Dev.	Buffer	Duration (1)+(3)	
Project	193.33	31.21	93.62	286.95	

Behind the scenes

$$\overline{ProjectDuration} = \sum_{i \in included} \overline{TaskDuration_i} - Lag_i$$

$$ProjectVariance = \sum_{i \in included} TaskVariance_i + 2 \sum_{i \in included} \sum_{j \in included} \sqrt{TaskVariance_i} \times \sqrt{TaskVariance_j} \times \rho_{ij}$$

$$ProjectContingency = k \times \sqrt{ProjectVariance}$$

$$SafeProjectDuration = ProjectDuration + ProjectContingency$$

$$k = \sqrt{\frac{1}{1 - SafetyLevel} - 1}$$ (Single tail Chebyshev inequality)

or

$$k = \sqrt{\frac{1}{2.25 \times (1 - SafetyLevel)}}$$ (Camp and Meidell inequality)

$$Buffer_{i \forall i \in included} = SafeProjectDuration - \sum_{i \in included} \left(\overline{TaskDuration_i} - Lag_i \right)$$

$$Buffer_{i \forall \notin included} = \text{Max}\left(SafeProjectDuration - \sum_{i} \left(\overline{TaskDuration_i} - Lag_i \right), k \times \sqrt{TaskVariance_i} \right)$$

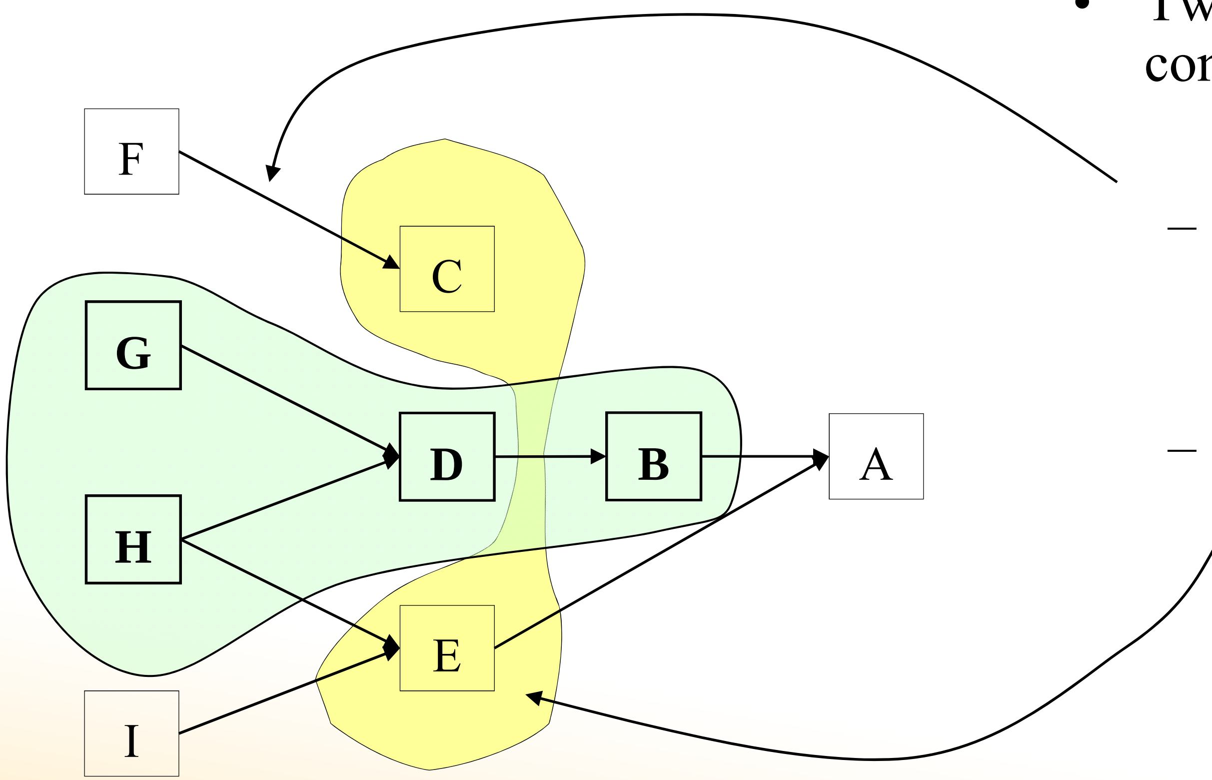
Calculating contingency

Desired safety level	*K*			
	Normal Distribution[1]	Camp & Meidell Inequality[2] Unimodal, symmetric distribution	Single tail Chebyshev Inequality[3]	Chebyshev Inequality[4]
75%	0.68	1.33	1.73	2.0
80%	0.84	1.49	2.0	2.23
85%	1.03	1.72	2.38	2.58
90%	1.28	2.10	3.0	3.16

$$ProjectContingency = k \times \sqrt{ProjectVariance}$$

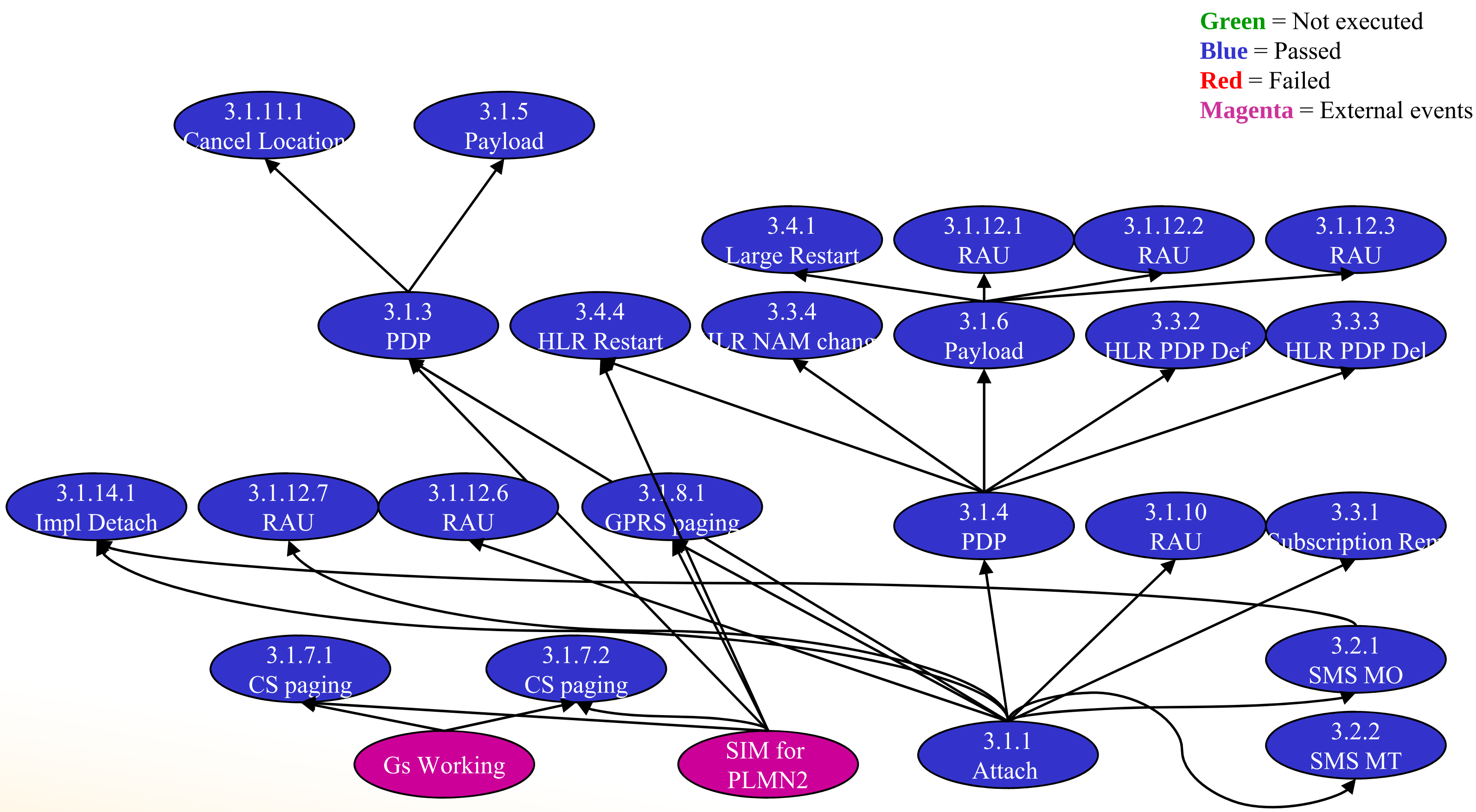
1. Common assumption in the PM literature
2. Practical Software Measurement: Measuring for Process Management and Improvement W. Florac R. Park & A. Carleton, SEI, 1997
3. The Economic Analysis of Industrial Projects, L. Bussey, Prentice-Hall series in Industrial and System Engineering, 1978
4. Probability and Statistics in Aerospace Engineering M.Rheinfurth and L. Howell, NASA, 1998

Incremental Development - Features dependency & completeness (Anatomy)

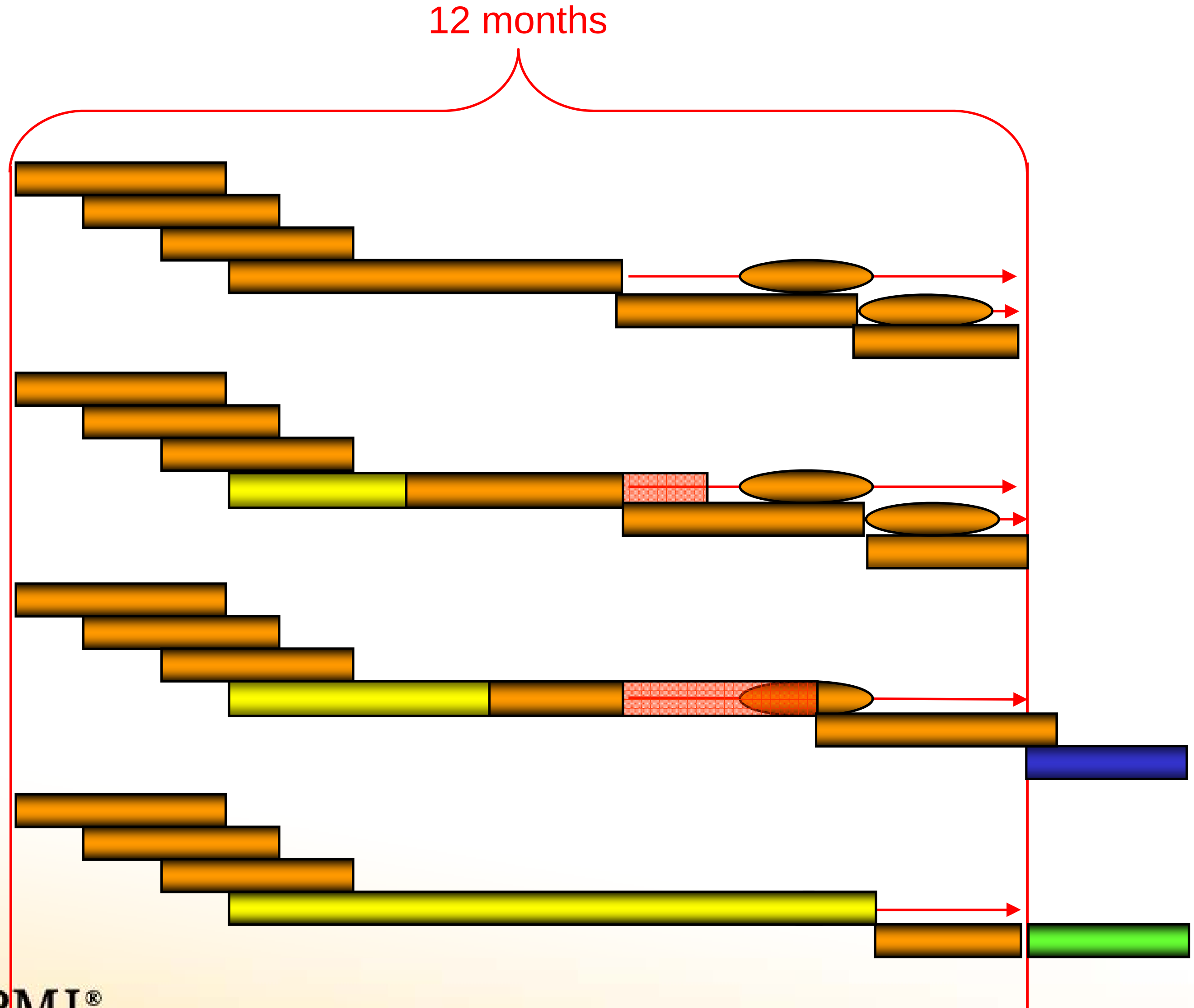


- Two aspects to be considered
  - Technical dependencies
  - Functional completeness from the user point of view

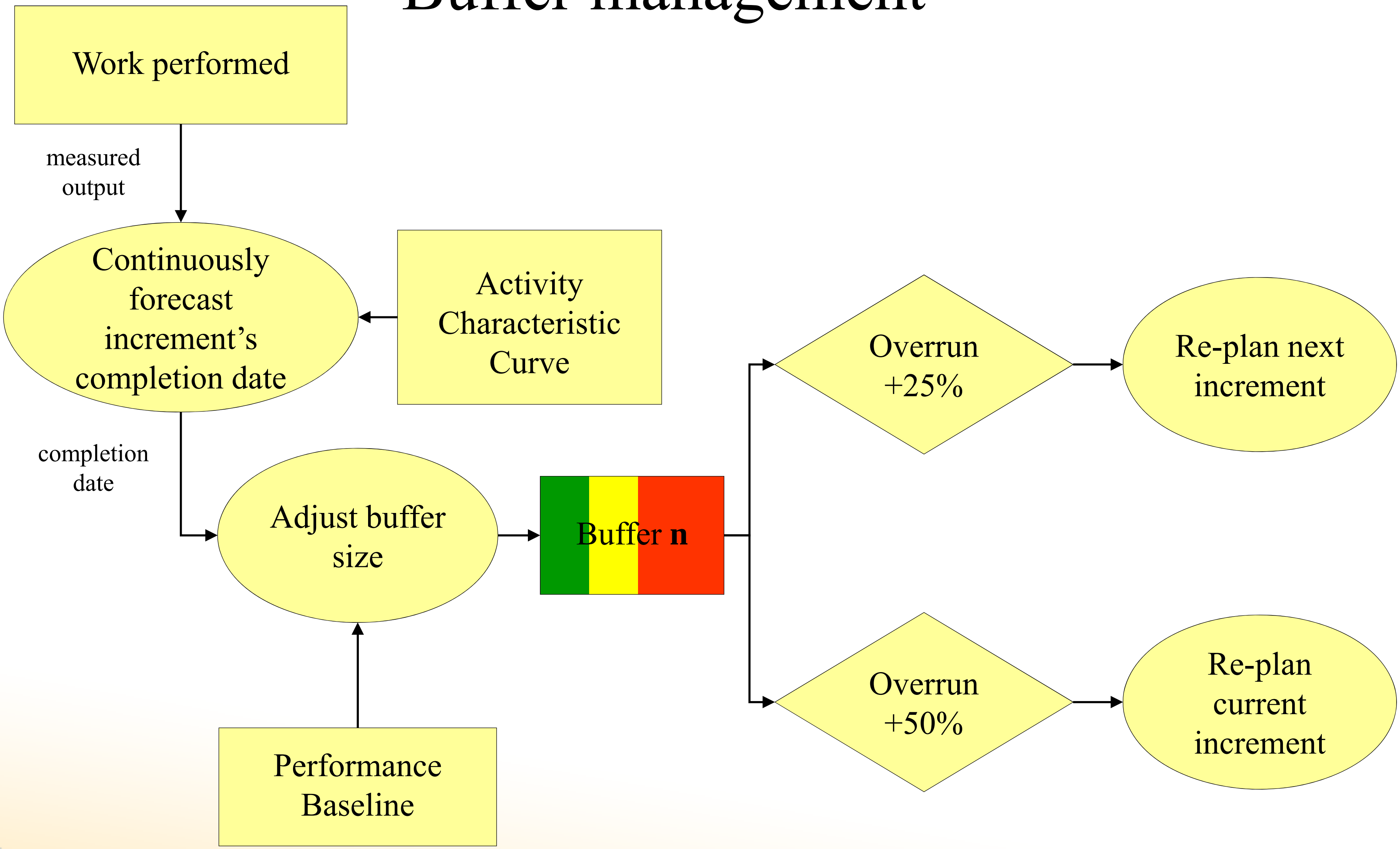
Anatomy example



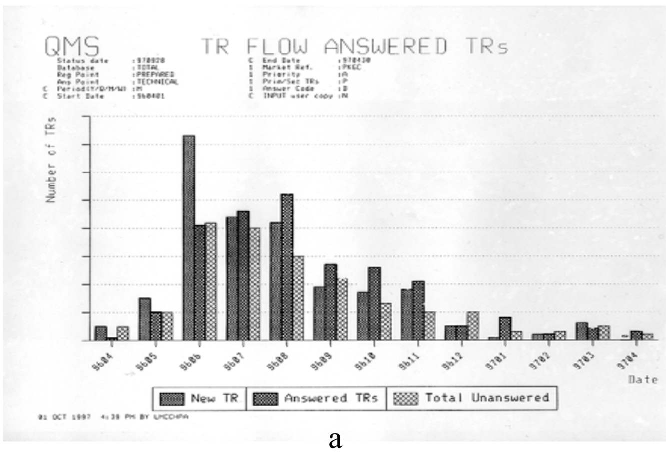
Managing the buffers example



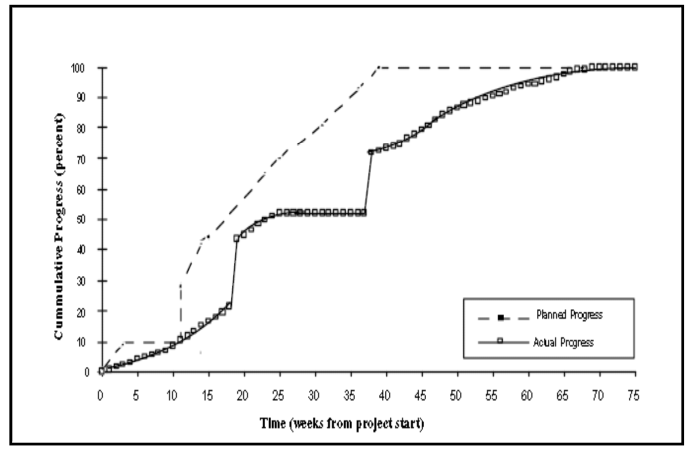
Buffer management



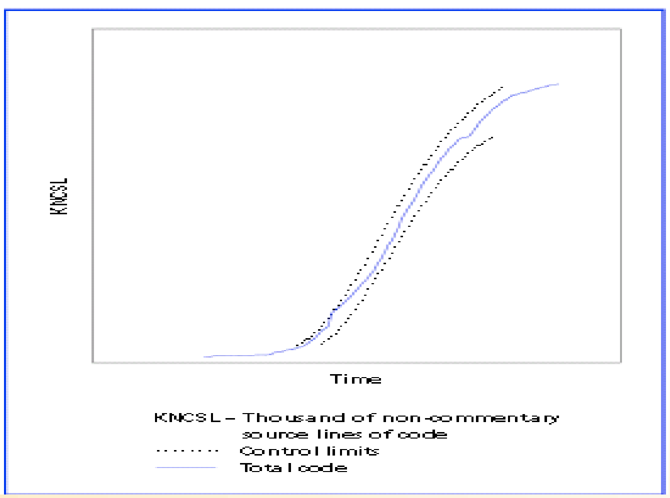
Work does not seem to progress at a constant rate



a



b



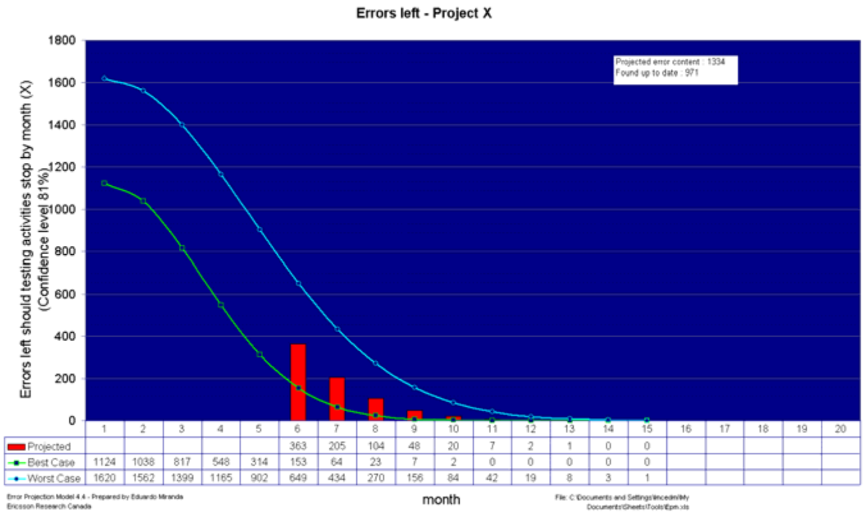
c

a) AXE Switch, error discovery pattern. Ericsson, 1997

b) Python Project. Semiconductor development project. Reported by Ford and Sterman in Overcoming the 90% Syndrome: Iteration Management in Concurrent Development Projects.

c) 5ESS-2000 Switch, code production pattern, Lucent 1997

Tool support



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Projected						363	205	104	48	20	7	2	1	0	0					
Best Case	1124	1038	817	548	314	153	64	23	7	2	0	0	0	0	0					
Worst Case	1620	1562	1399	1165	902	649	434	270	156	84	42	19	8	3	1					

Adaptive Forecast vs. Plan

Summary View

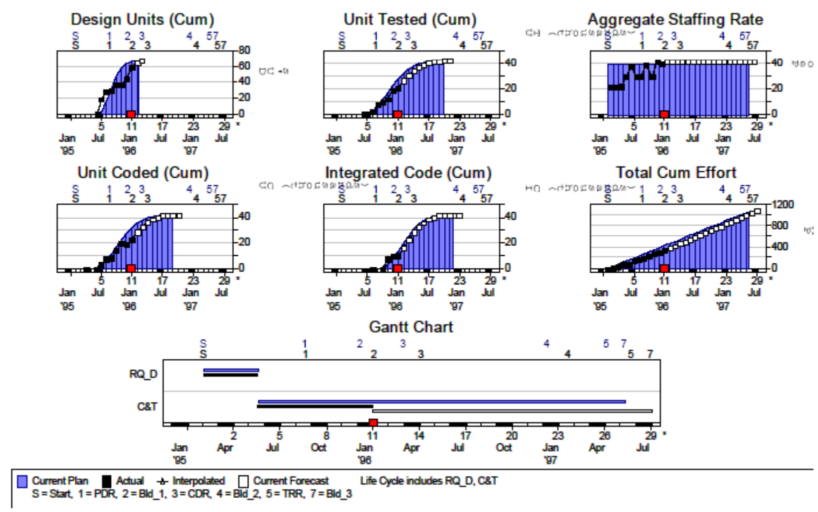


Figure #6. Summary of plan vsactuals for the key metrics.

1. Error Projection Model, Ericsson
2. Slim Control, QSM

Rewards and Incentives

- Employee rewards associated with increment completion, suppress overtime and provide larger bonuses after successful deployment
- Contracts could include price incentives to be paid on increment delivery
- Amount of reward and incentive should be calculated using the probabilities of a successful delivery, i.e. :
  - increment 2 probability of success = 40%, bonus = 5,000$, expected value of the reward = 2,000$
  - increment 3 probability of success = 10%, bonus = 10,000$, expected value of the reward = 1,000$
  - To act as a motivator bonuses should be re-structured, for example increment 2 = 3,000$ and increment 3 = 18,000$. In both cases the total expected pay-out is the same 3,000$ but the motivation power very different.

Summary

- Delivery reliability
- Simplified product, project and resource planning
- Higher productivity
- The above are accomplished by:
  - Limiting scope at the outset of the project, thus preventing people from working in things that may never get implemented anyway
  - Creating buffers that protect the delivery date of the most important features from the uncertainty of project work
  - Focusing the work of people in a single set of objectives at a time
  - Having small, integrated product teams

References

- Miranda E., Planning Time Bounded Projects, IEEE Computer, March 2002,Volume 35, Number 3
- Goldratt E., Critical Chain, The North River Press, 1997
- Newbold R., Project Management in the Fast Lane, St. Lucie Press, 1998
- McConnelS. l, Rapid Development, Taming Wild Software Schedules, Microsoft Press, 1996
- Pisano N., Technical Performance Measurement, Earned Value and Risk Management: An Integrated Diagnostic Tool for Program Management
- Grey S., Practical Risk Assessment for Project Management, John Wiley & Sons, 1995
- Pillai K. and Nair S., A Model for Software Development Effort and Cost Estimation, IEEE Transactions on Software Engineering, Vol. 23, No.8, 1997
- Putnam G., Measures for Excellence – Reliable Software On Time, Within Budget, Prentice-Hall, 1992
- Martino J., Technological Forecasting for Decision Making, McGraw-Hill, 1993
- Miranda E., The Use of Reliability Growth Models in Project Management, 9th International Symposium in Software Reliability, IEEE, 1998
- Gaffney J., On Predicting Software Related Performance of Large-Scale Systems, CMG XV, San Francisco 1984
- Miranda E., Running the Successful Hi-Tech Project Office, Artech House, 2003

Contact Information
Eduardo Miranda
Ericsson Research Canada
eduardo.miranda@ericsson.com

The End