



# Software sizing – the weakest link in estimating?

Charles Symons

Joint Project Leader

The Common Software Measurement International Consortium  
Galorath/SEER User Conference, Manchester, March 2009

© Charles Symons, 2009



# My thesis

- Software industry performance is very poor in certain respects
- Measurement of software size is the weakest link in performance measurement and estimating
- Poor software sizing and estimating practices contribute to poor industry performance
- Improving industry performance will depend on adopting better software sizing methods (e.g. COSMIC) and better integration of estimating with project management processes (e.g. Southern Scope)



# Let's start by looking at some losses in 2008 from the credit crunch

- Merrill Lynch \$41.2 billion
- Citigroup \$18.7 bn
- UBS \$18 bn
- Lehman Brothers \$47 bn\* (bankrupt)
- Bear Sterns \$22 bn\* (bankrupt)
- Royal Bank of Scotland \$36 bn (preliminary)
- Etc

\* Market capitalization loss



## Now compare these with the estimated annual waste in the software industry

- US software industry: \$100 Billion\*
- European software industry €100 Billion\*\*

‘Waste in the software industry’ = write-offs due to project failures, and cost and time over-runs

\* Standish CHAOS Report, 2006, [www.standishgroup.com](http://www.standishgroup.com)

\*\* McManus, J. and Wood-Harper, T., “A Study in Project Failure”, [www.bcs.org](http://www.bcs.org), June 2008



# How does this waste arise?

- Poor project management?
- &/or shifting requirements?
- &/or ?
- &/or poor **'estimating practices'**?

i.e.

- Estimating the size of the software to be built
- The method of converting size to effort and time
- The way in which estimating is integrated with the project management process



# The fundamentals of 'top-down' estimating methods

$$\text{Productivity} = \text{Software size} / \text{Project effort}$$

$$\text{Estimated new project effort} = \frac{\text{Estimated software size}}{\text{Assumed project productivity}}$$

$$\text{Estimated new project effort} = \left\{ \frac{\text{Estimated software size}}{\text{Assumed project productivity}} \right\} \times \left\{ \text{Adjustments for project-specific factors} \right\}$$

Conclusion: Reliable software sizing, especially Functional Size Measurement (FSM), is critical for productivity measurement and for estimating methods



## In total, what do we know about software industry average performance?

Reliable FSM  
and estimating  
essential?

- Delivery to budget and time is dreadful
- Productivity has improved only very slowly over the last 10 – 20 years
- Speed of delivery? (rarely reported)
- Delivered software quality (in terms of defects) is often outstanding

Yes

Helpful

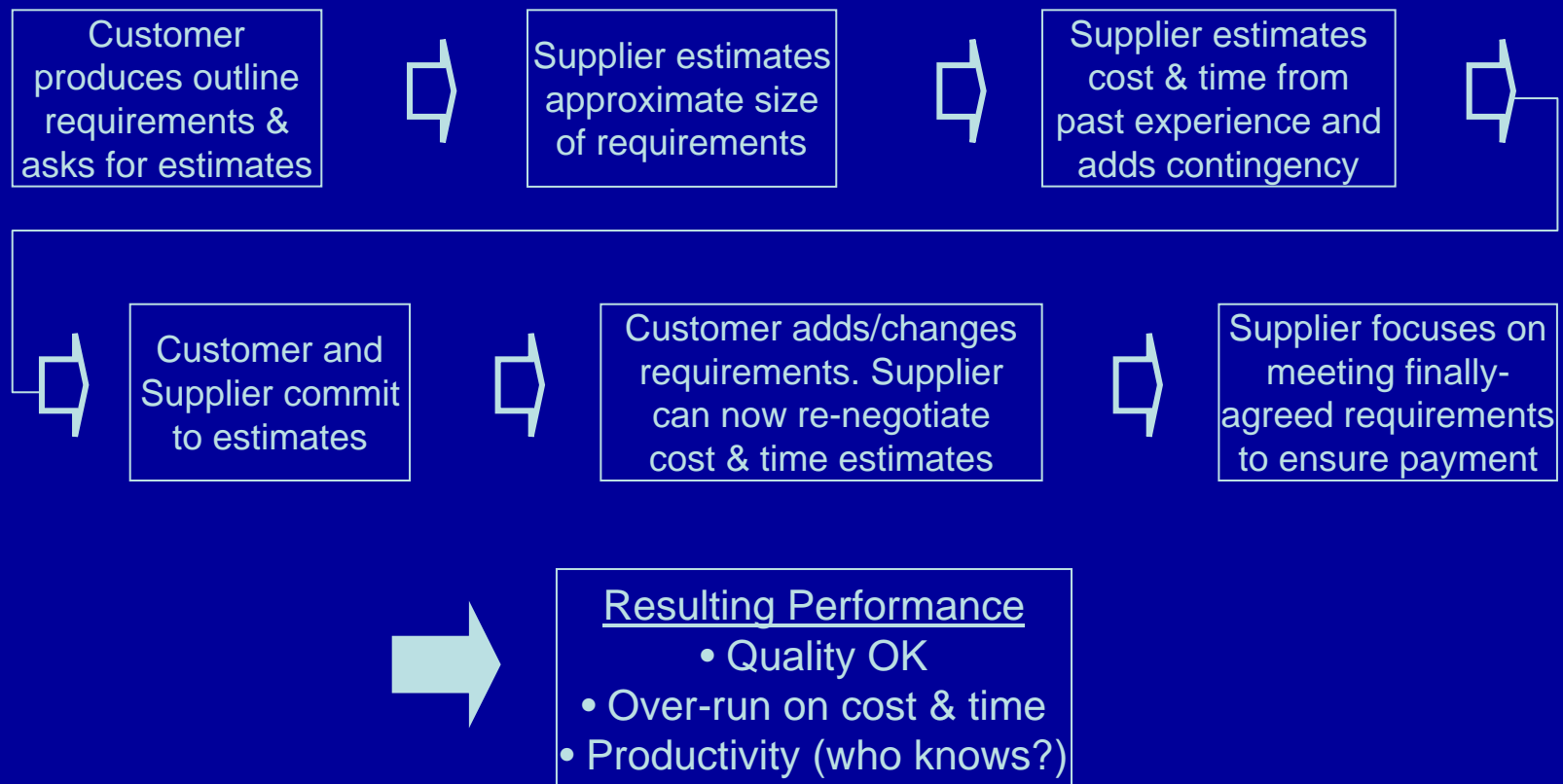
Helpful

No



# These various threads suggest a causal link between poor sizing and estimating and poor software project performance

A typical project process:





## Summary of this section: re-cap my thesis

- Software industry performance is very poor in certain respects
- Measurement of software size is the weakest link in performance measurement and estimating
- Poor software sizing and estimating practices contribute to poor industry performance

Poor estimating may not be the biggest cause of poor software industry performance, but evidence suggests it is a major contributor.



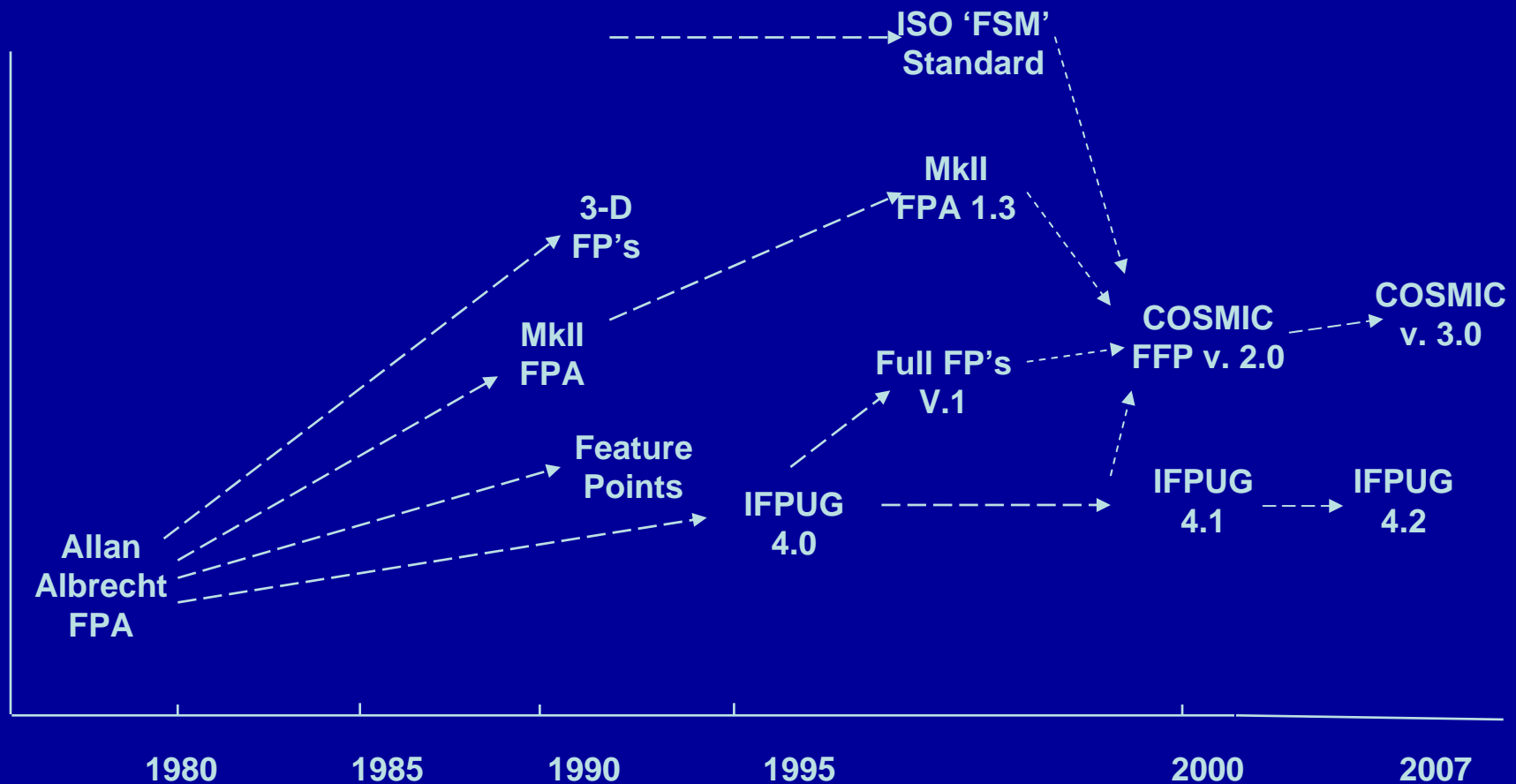
# Re-cap: the elements of 'estimating practices'

- Estimating the size of the software to be built
- The method of converting size to effort and time
- The way in which estimating is integrated with the project management process

Let's examine each element in turn

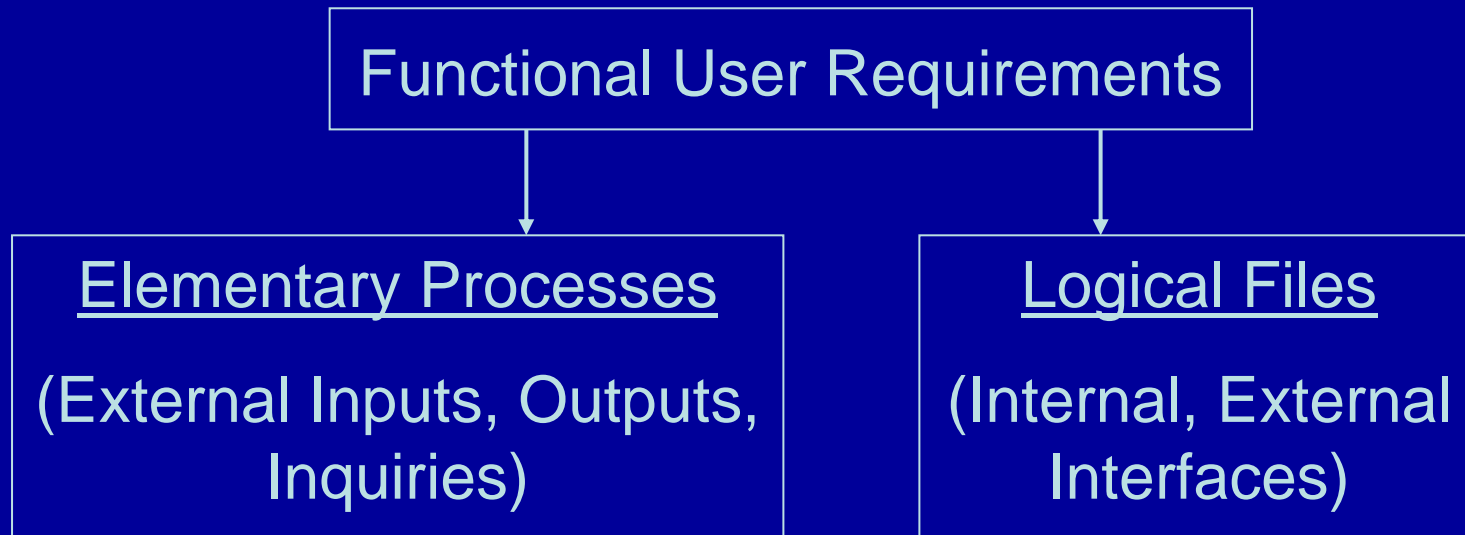


# Functional Size Measurement (FSM) methods have evolved over 30 years





# Structure of the Albrecht / IFPUG FSM method



'Weights'      3 – 7 FP's

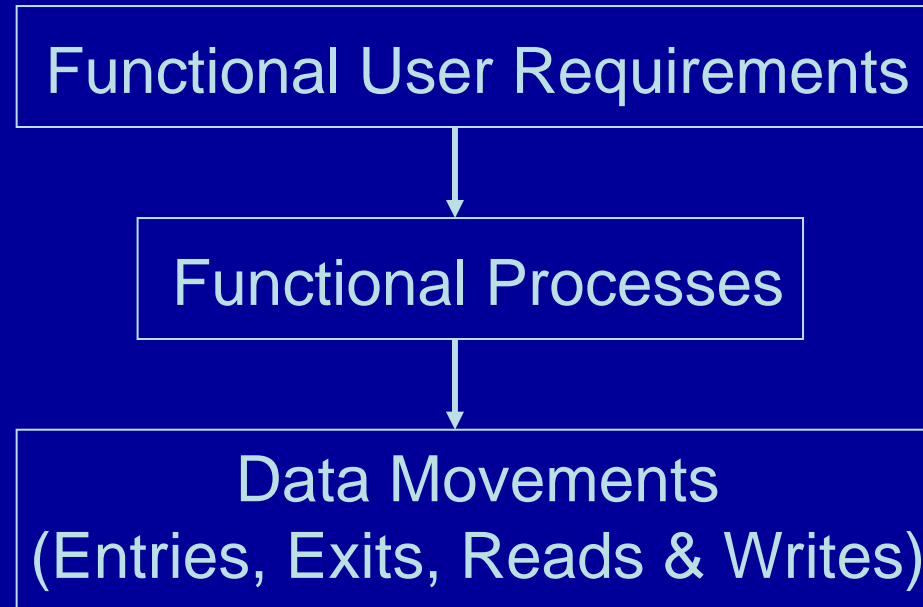
7 – 15 FP's

- derived from early IBM estimating methods

(Ignored: a 'Value Adjustment Factor' intended to account for technical and quality requirements)



# Structure of the COSMIC FSM method



'Weights'      Entry = Exit = Read = Write = 1 CFP

Size of a Functional Process: Min = 2 CFP; Max = No limit



# IFPUG vs COSMIC FSM methods: The key differences

	<b>IFPUG</b>	<b>COSMIC</b>
Design domain of applicability	Business Apps	Business, Real-time, Infrastructure software
Basis of sizing model	Ad hoc	SE Principles
Calibration of sizing model	IBM estimating methods (1970's)	FSM Principles (ISO 14143/1:2007)
Component size limit?	Arbitrary cut-off	No limit



# Estimating methods: how could we better convert from COSMIC FP's to: **'COSMIC locally-calibrated standard hours'?**

Traditional top-down estimating method:

$$\text{Estimated new project effort} = \left\{ \frac{\text{Estimated software size}}{\text{Assumed project productivity}} \right\} \times \left\{ \text{Adjustments for project-specific factors} \right\}$$

The future?

$$\text{Estimated new project effort} = \left\{ \text{'COSMIC Standard Hours' for estimated software size} \right\} \times \left\{ \text{Adjustments for project-specific factors} \right\}$$



# Standard hours per data movement type will be obtained by statistical correlations for coherent sets of ISBSG\* data

Example dataset

Project	#E	#X	#R	#W	Effort (WH)
1	232	287	197	110	2503
2	22	19	6	8	740
3	35	99	125	14	699
4	332	414	59	31	2678
5	107	14	26	132	385
etc					

## Current research goal:

To demonstrate that standard hours per Entry, Exit, etc can be obtained for specific environments that can be used for more accurate estimating



# Further development of the COSMIC method?

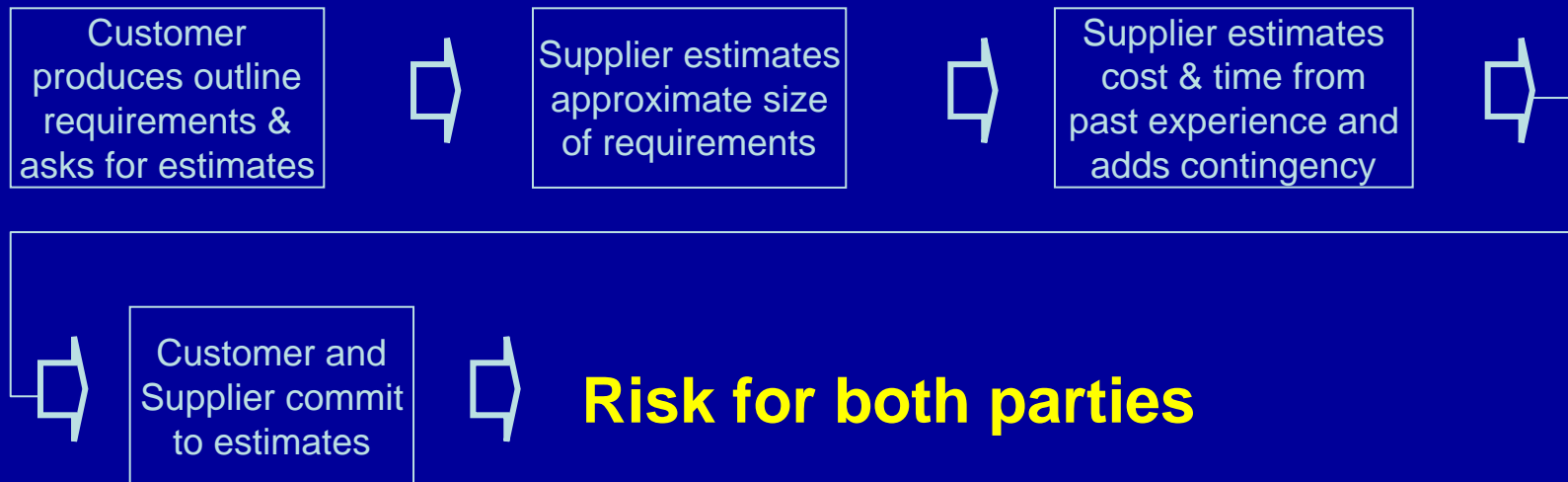
- Standard COSMIC functional sizing
  - Count the E's, X's, R's and W's to obtain the standard functional size
  - Ideal for cross-technology performance comparisons
- Rules for locally-calibrating estimating methods to accept 'COSMIC standard hours'
  - Count the E's, X's, R's and W's
  - Weight them by standard hours per E, per X, etc., calibrated for the local environment

**This is work in progress. Watch this space!**



# Lastly, we need better processes by which estimating is integrated with project management

The typical project process:



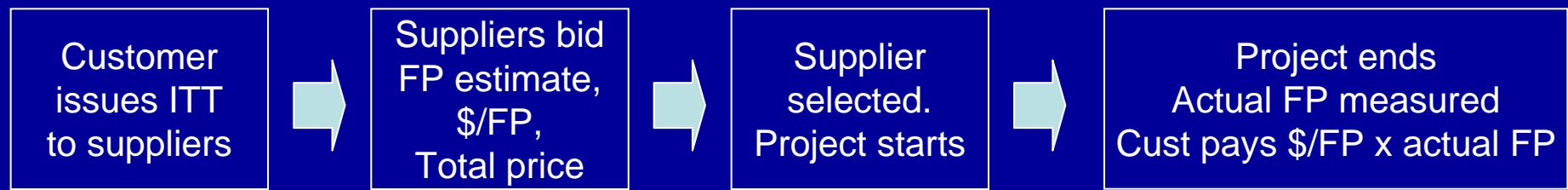
The 'Southern Scope'\* process makes risk management much easier

\* [www.mmv.vic.gov.au](http://www.mmv.vic.gov.au)



# The 'Southern Scope' process uses FSM for successful project control

## Process overview:



## Results: \*

Project management approach	\$/FP	Av. cost over-run
Traditional	1500	84%
Southern Scope	500	<10%

**Scope management role (metrics expert) is critical – and highly visible!**

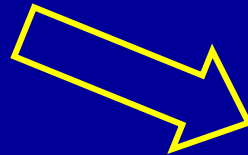
\* 'Software development projects in Government; performance, practices and predictions', ISBSG, 2005



# In summary, the ingredients for improvement exist

Improving estimating needs better:

- FSM methods (COSMIC)
- integration of estimating with project management (Southern SCOPE)



**The benefits of improved software industry performance are huge for software producers and customers**



**Thank you for your  
attention**

**Charles Symons**

**[cr.symons@btinternet.com](mailto:cr.symons@btinternet.com)**

**[www.cosmicon.com](http://www.cosmicon.com)**