

# MEASUREMENT OF SOFTWARE SIZE: CONTRIBUTIONS OF COSMIC TO ESTIMATION IMPROVEMENTS

Alain Abran
with C. Symons, C.Ebert, F.Vogelezang, H.Soubra

ICEAA International Training Week October 17-20, 2016, Bristol (UK)



### Presenter background: Alain Abran

#### 20 years





- **Development**
- Maintenance
  - **Process Improvement**

20 years









ISO: 19761, 9216, 25000, 15939, 14143, 19759





© Copyrights 2016: COSMIC and authors

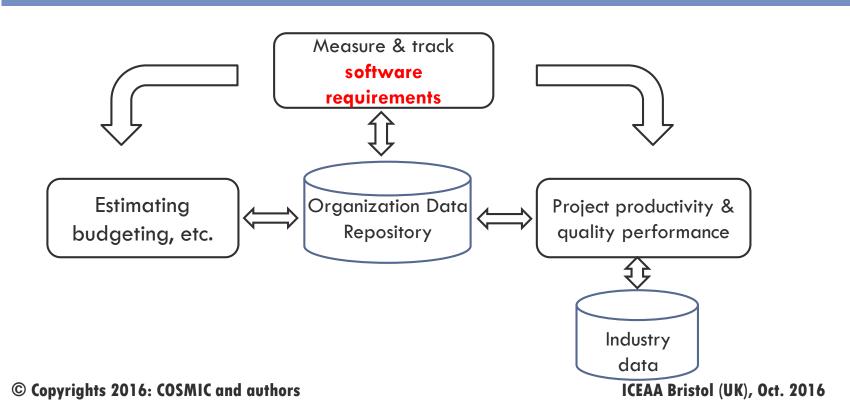
ICEAA Bristol (UK), Oct. 2016



# COSMIC Agenda

- Background to Functional Size Measurement (FSM) methods
- COSMIC Method Key features of ISO 19761
- **Measurement Guidelines**
- 'The proof of the pudding is in the eating': Good Estimation
- **Automation of COSMIC measurement**
- Conclusions

# Objective: we want to use performance data for estimating future projects



# **Software Sizing Options**

#### Sizing method options:

Lines of code:

- X Can't estimate until software designed
- Technology-dependent, no standards

Usecase Points, Object Points, ...

- Technology dependent, no standards,
- Mathematically invalid?



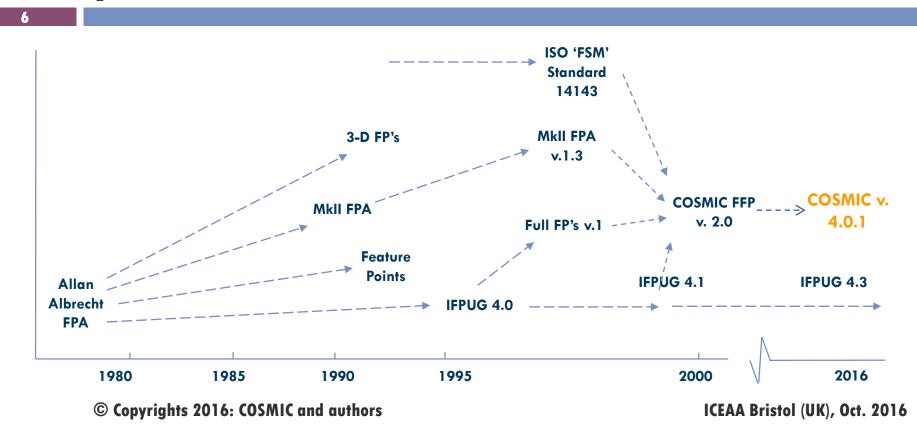
- Story Points (Planning X Poker):
- Entirely Subjective & Benchmarking impossible:
  - unaccountability

> Functional size

- International standard methods
- √ (Function Points): 
  √ Technology-independent



# Function Points have been around for a long time: + 35 years!





## COSMIC Agenda

- Background to Functional Size Measurement (FSM) methods
- - COSMIC FSM Method Key features of ISO 19761
  - **Measurement Guidelines**
  - 'The proof of the pudding is in the eating': Good Estimation
  - **Automation of COSMIC measurement**
  - **Conclusions**



## **Key Concepts of Measurements**

- Measurement of what is common
  - not what is different: measurement of a single concept independent of contexts
- Measurement standards for all people
- Standard Unit of measurement ('étalon') to ensure consistency-repeatability-reproducibility, etc.
- Measuring instruments:
  - Manual procedures & guidelines
  - Partially or Totally automated with measuring instruments



#### **COSMIC Method**

- Designed by an international group of software measurement experts
  - COSMIC: Common Software Measurement International Consortium
- To measure the <u>Functional User Requirements</u> of:
  - Business application
  - Real-time
  - Infrastructure software
  - Various other types of software
  - Hybrids of these
- Based on:
  - Metrology
  - Fundamental software engineering principles
- An ISO standard: ISO 19761
- Open, freely available (via <u>www.cosmic-sizing.org</u>)







# Software is an 'intellectual' product: can we measure it?

- Time: can we 'see' or 'touch' time?
  - ➤ Answer = .....
- Distance: is there a unique distance between 2 cities?
  - $\triangleright$  Answer = .....
- Do we need to 'see' something to measure it?
  - $\triangleright$  Answer = .....
- Can we measure something before it exists?
  - $\triangleright$  Answer = .....



# Software is an 'intellectual' product: can we measure it?

- Time: can we 'see' or 'touch' time?
  - > Answer = No (.. but measuring instruments have been built...)
- Distance: is there a unique distance between 2 cities?
  - Answer: It depends (by road, by car, by train, by highways, by plane)
- Do we need to 'see' something to measure it?
  - $\triangleright$  Answer = No (...ex. microscope)
- Can we measure something before it exists?
  - $\triangleright$  Answer = Yes (....from their representation in models & plans)

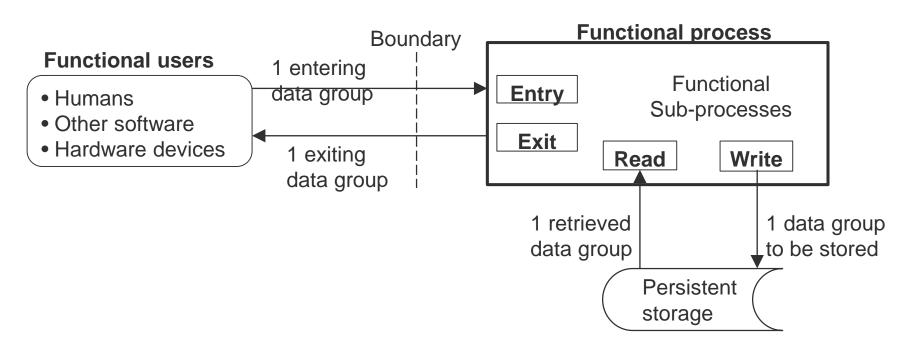


# COSMIC view of software

> What is common across all software, in different types of sofware, whether very small or extremely large?

# All software does this!

H

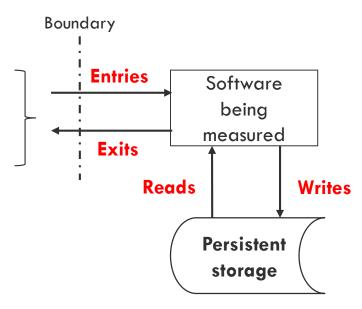


# 4 types of 'Data Movement'

#### **Functional Users**

- Hardware devices,
- Other software or
- Humans

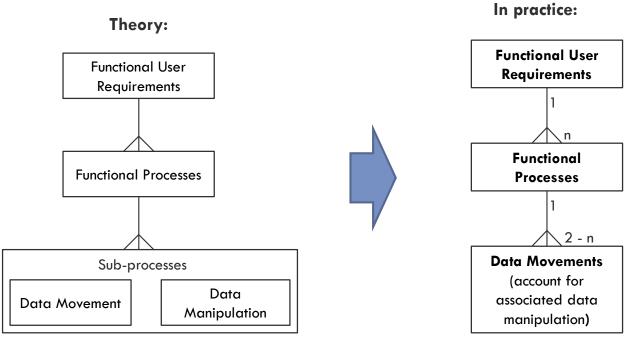
The 'Data Movement of 1 data group' is the unit of measurement: 1 CFP (COSMIC Function Point)





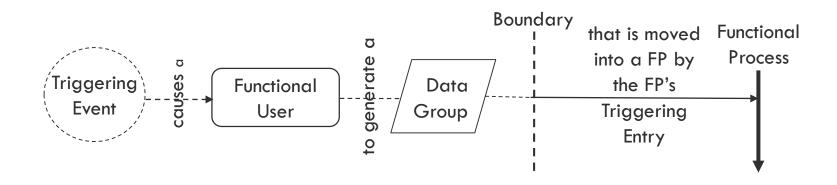
# All software Functional User Requirements can be broken down into functional processes

15



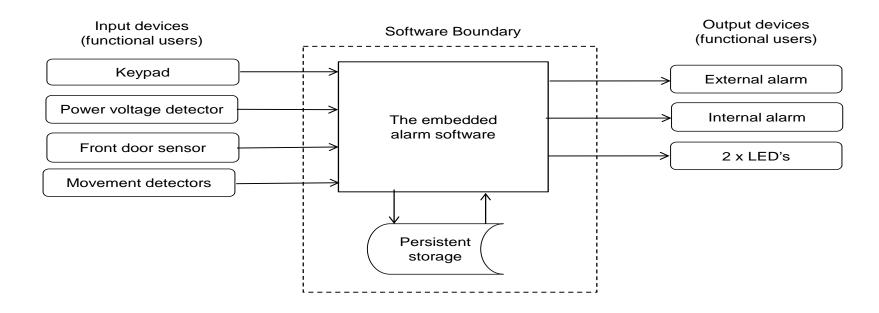
# Presented at the 2016 International Training Symposium: www.iceaaonline.com/bristol2016 A Functional Process responds to an 'Event' that a 'Functional User' detects or generates

16





### **Example: Intruder Alarm System**





# COSMIC Intruder Alarm System

Functional process: Possible intruder detected. Triggering event: Door opens whilst alarm system is activated.

Data Movement	Functional User	Data Group
Entry	Front-door sensor	'Door open' message (triggering Entry)
Read	- / Occupant	PIN (from persistent storage)
Exit	Green LED	Switch 'off' command
Exit	Red LED	Switch 'on' command
Exit	Internal siren	Start noise command
Entry	Keypad	PIN (If the wrong code is entered, the user may enter the PIN two more times but the process is always the same so it is only measured once.)
*	Green LED	Switch 'on' command (after successful entry of PIN)
*	Red LED	Switch 'off' command
Exit	Internal siren	Stop noise command (after successful entry of PIN)
Exit	External siren	Start noise command (after three unsuccessful PIN entries, or if the PIN is not entered in time)
Exit	External siren	Stop noise command (after 20 minutes, a legal requirement)

Size = 9 CFP



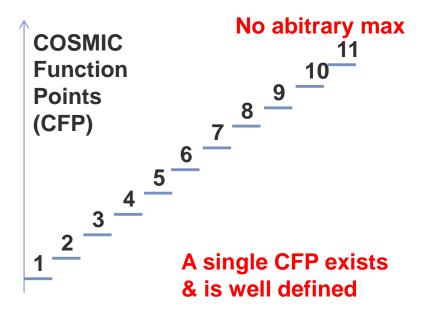
## 1<sup>st</sup> Generation of Function Points: Step Functions!

ЦΣ



3-step size range for the IFPUG External Input Transactions

### **2<sup>nd</sup> Generation with COSMIC**



# COSMIC sizes are measured on a true ratio scale

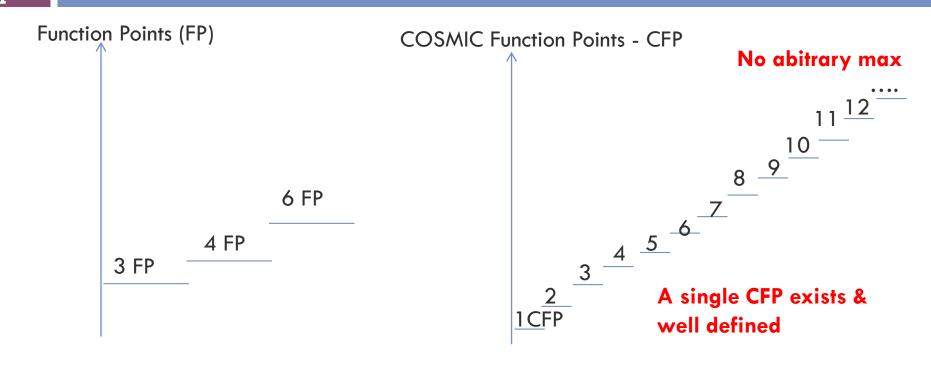
- There is no upper limit to the size of a functional process
- **Largest observed functional processes?** 
  - In avionics > 100 CFP
- The size of the smallest change to an existing functional process is 1 CFP

Open, freely available (via www.cosmic-sizing.org )



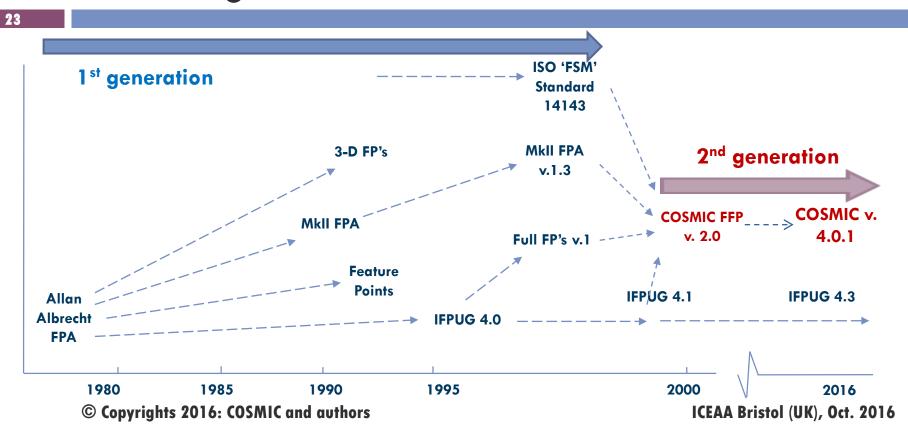
## 1<sup>st</sup> and 2<sup>nd</sup> Generations of FSM

22





# 1st & 2nd generation of Function Points Methods



## **COSMIC** - at any level of software requirements

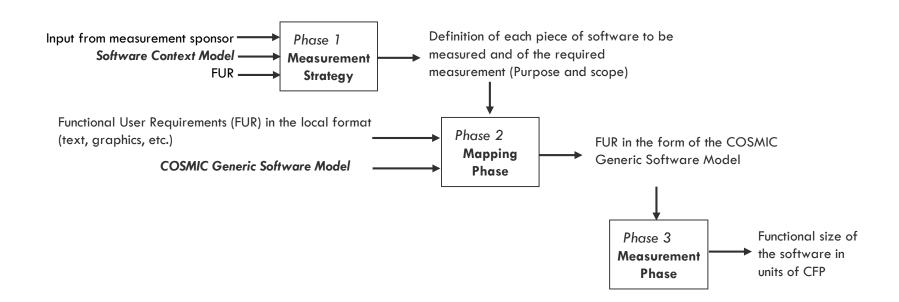
**Application Layer** App 2 App 'n' App 1 Middleware Layer (Utilities, etc) **Database Management** DBMS<sub>1</sub> DBMS 2 System Layer Operating System Layer Print Keyboard Screen Disk Driver Driver Driver Driver **VDU** Hard Disk Central Keyboard Printer Hardware Drive Screen Processor

© Copyrights 2016: COSMIC and authors

ICEAA Bristol (UK), Oct. 2016

#### 0 0 141

### There is a well-defined Measurement Process



### Agenda

- Background to Functional Size Measurement (FSM) methods
- COSMIC FSM Method ISO 19761
- Measurement Guidelines
- 'The proof of the pudding is in the eating': Good Estimation
- Automation of COSMIC measurement
- Conclusions



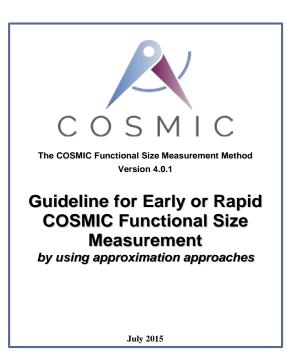
### **Recent Guidelines for Practitioners**

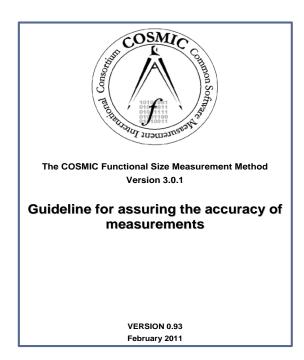
27

A Guideline describing a range of Approximate Sizing methods

Size/Cost estimates are usually needed before the FUR have been defined in detail

A Guideline on 'Assuring the accuracy of COSMIC measurements'







### **Guidelines by Application Domains**

28

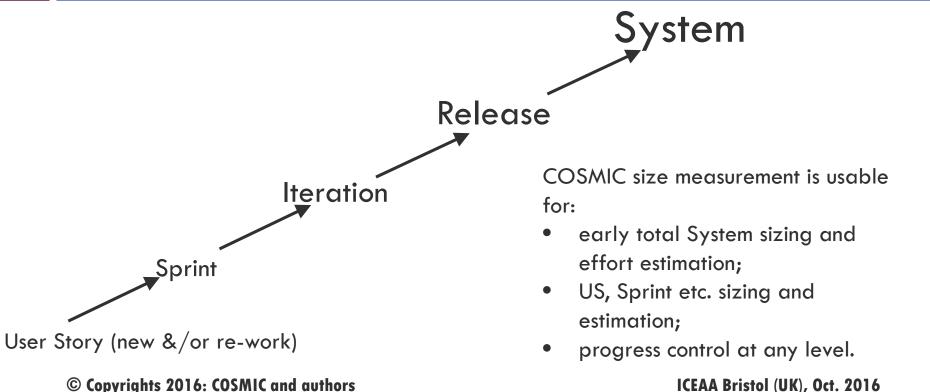
- Business applications
- Real-time software
- Data Warehouse software
- SOA software
- Mobile appsand for Agile Developments





# Aggregation rules for components, sprints, etc. up to whole software systems

29



#### What to do about NFR?

30

Again, there was no good standard definition of a NFR

A joint COSMIC/IFPUG effort developed good definitions and a Glossary of NFR and Project Requts.

The COSMIC Guideline advises how to deal with NFR





Glossary of terms for
Non-Functional Requirements
and Project Requirements
used in
software project performance
measurement, benchmarking and
estimating

VERSION 1.0 September 2015



The COSMIC Functional Size Measurement Method Version 4.0.1

#### Guideline on Non-Functional & Project Requirements

How to consider non-functional and project requirements in software project performance measurement, benchmarking and estimating

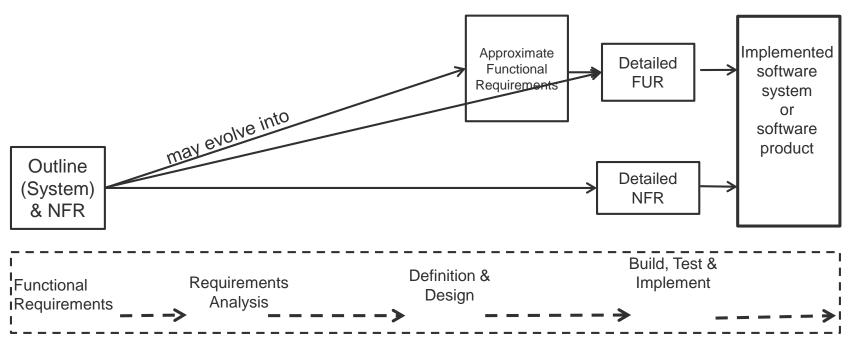
Version 1. November 2015

© Copyrights 2016: COSMIC and authors

ICEAA Bristol (UK), Oct. 2016

# Abran & Al Sarayreh showed that requirements that appear as NFR may evolve into FUR, that the COSMIC method can measure

31



© Copyrights 2016: COSMIC and authors

ICEAA Bristol (UK), Oct. 2016



### **Examples of NFR leading to FUR with COSMIC**

#### A Standards-Based Model of System Maintainability Requirements

Khalid T. Al-Sarayreh\*

Software Engineering Department, Prince Hussein Bin Abdullah II for Information Technology, Hashemite University (HU), Zarga, Jordan.

Software Engineering Department, University of Quebec (ETS), 1100 Notre-Dame Street West, Montréal, Québec H3C 1K3 Canada.

Juan J. Cuadrado-Gallego

Departamento de Ciencias de la Computación, Universidad de Alcalá, 28805 Alcalá de Henares Madrid Spain

#### 1. SUMMARY

The non functional requirements are often captured only generically at a fairly high level, and they do not include the levels of detail necessary at this stage for the system engineers to allocate them as specific functionalities to be handled either by the software or the hardware, or a specific combination of the two. The European ECSS series of standards for the aerospace industry includes maintainability requirements as one of sixteen types of non functional requirement (NFR) for embedded and real-time software. A number of maintainability-related concepts are dispersed throughout the ECSS, ISO 9126, and IEEE standards to describe. at varying levels of detail, the various types of candidate maintainability requirements at the system, software, and hardware levels. This paper organizes these dispersed maintainability concepts into a standards-based reference model of system maintainability requirements. The availability of this reference model can facilitate the early identification of the system maintainability-NFR and their detailed allocation as specific maintainability functions to be handled by the specified allocation to hardware or software, or a specific combination of the two. In the absence of such a reference model, these NFR requirements are typically handled in practice much later on in the software development life cycle, when, at system testing time, users and developers find out that a number of maintainability requirements have been overlooked and additional effort has to be expended to implement them. The approach adopted in this research for the structure of this reference NFR model is based on the generic model of software functional requirements proposed in the COSMIC - ISO 19761 model, thereby allowing the functional size of such maintainability requirements allocated to software to be measured. Copyright © 2011 John Wiley & Sons, Ltd.

KEYWORDS: Software Engineering, Non Functional Requirements - NFR, Maintainability Requirements, ECSS International Standards, Maintainability Measurement, Functional size, COSMIC - ISO 19761.

 Correspondence to: Khalid T. Al-Saravreh. Software Engineering Department. Hashemite University (HU) 1100 Zarqa, Jordan. / E-mail: khalidt@hu.edu.jo



Computer Standards & Innerthon 35 (2012) 390-390

#### Contents Esta available at Schlares ScienceDirect Computer Standards & Interfaces

ournal homeoage: www.elxevier.com/locate/cx

A standards-based reference framework for system portability requirements Alain Abran \*\*, Khalid T. Al-Sarayreh b, Juan I, Cuadrado-Gallego \*c

\* Software Engineering and Information Technology Department Easte die inchmitigle asperieurs (TES), University of Queller, Consula \* Software Engineering Copyrament, Vinderson University (VELS) orden \* Copyrater Komen Department, University of Model Special

ARTICLE INFO

Received in resteed three 4 October 2012 Accepted 14 November 2012 Available ordine & Laurence 2012

In the option ray invented plane, the non-functional requirements (NSC) are often optioned only generally at a fieldy high-level, and begin in order to include the level of detail necessary for the optioner regimeers to all starting the contraction of the companies of the contraction of the contra one of sixteen types of non-functional requirements (NEI) for embadded and mol-time software. A number of portability-dated concepts are dispensed throughout the ELSS, EEE-ASS, ESO SEES, ESO APRS, and ESO SEES-1 standards to describe, activative levels of default the variantspees of candidate portability requirements. at the system, software, and hardware levels. This paper organizes these dispersed portability excepts and terms into a standards-hand reference framework of system portability requirements. The availability of this framework can fastitus in heavily dentification and questi eaties of the system portability NE Mand their detailed adoustin as specific portability of sactions to the hand delty the question allocation as and an downer or additioner, or a specific combination of the two. The approach selected in this rewards for the structure of this reference framework is haved on the gas at it model of software proposal in the CDIMIC-BO 19701 model, thereby allowing the functional size of the portability requirements allocated to software to be measured.

© 2013 Hawsier R.V. All rights reserved.

In the system require ments phase, the focus is often on detailing and documenting the system functional requirements (FR) and their alloca-tion to the software and hardware parts of the system being designed. Non-functional requirements (NFR) play a critical role in system development, including their use as selection criteria for choosing a mong alternative designs and ultimate implementations. NFR may also have a onsiderable impact on project effort, and should be taken into account for estimation numoses and when comparing project productivities.

Typically, these NFR are described at the system level, not at the software level, and as yet there is no consensus on how to describe and measure them. In practice, they may be viewed, defined, interpreted, and evaluated differently by different people, particularly when they are stated briefly and vaguely [1-3], it is a challenge therefore, to take NFR into account in software estimation and software benchmarking, and they are definitely less well understood than other cost factors [2,4,5]. Without measurement, it is not an easy matter to take them as quantitative inputs to an estimation process or to productivity benchmarking.

In practice, requirements are initially addressed at the system level [6–8], either as high level system functional requirements (system-R) or as high level system NFR (system-NFR). Normally, such high-level

Corresponding author.
 E-mail address: sinte brandwood za (A. Abran).

0630-548(\$ - see foot matter © 2013 Elsevier R.V. All rights reserved

requirements must then be detailed and allocated to specifics related functions, which may be implemented in hardware or software as software functional user requirements (software-RIR), for instance — see System-FR describe the functions required in a system, while

astem NFR describe how those functions must behave in the astem [9,10]. In the software requirements engineering step, system-NFR may then be detailed and specified as software-FLR, to allow a software developer to develop, test, and configure the final deliverables to system users Functional requirements are the functions that the system finduding the software) is to offer, while NFR detail the manner in which those functions are performed. FRare described using subject or predicate constructions (i.e. noun/verb), such as: "The system can run on two or more kinds of devices or with two or more kinds of operating asterns," NFR are described using adverts or modifying clauses, such as: "The system can runnetwoor more kinds of devices, or with two or more kinds of op-

erating systems, that are easily or unsweinedytransported."

Within the European standards for the aerospace industry (ECSS) [11-15], ISO 9126 [16], IEEE 830 [17], ISO 24765 [18], and ISO 2382-[19], a number of concepts are provided to describe various types of candidate portability require ments at the system, software, and hard-ware kivels. However, these standards vary in their views, terminology, and portability coverage.

Currently, there exists no reference framework for the identification and spedification of system nortability-NFR from the various views documented in international standards or in the literature. Recent Advances on Electroscience and Compute

#### Model of Early Specifications of Performance Requirements at Functional Levels

Khalid T. Al-Sarayreh

Abstract— This paper presents an integrated standards-based measurement for a single type of NFR, which is the performance requirement. The development stages of the standards-based framework have passed by two main steps: the first step is constituted in identifying and analyzing the system performance requirements and their allocated software performance requirements that are dispersed into the IEEE and ECSS nternational standards, the second step is modeling the identified intermitteenin innanarat, the second step is moderning the incumentary system/software performance requirements using the Soft-goal Interdependency Graph; and clarifying the interdependency

Keywords- Performance Requirements, International Standards, Soft-goal Interdependency Graphs.

#### I. INTRODUCTION

I the system requirements at early development phases offered. constitute the most significant factor to build a successful system that satisfies the stakeholder expectations and needs. In software engineering, the requirements are classified under two types: the functional requirements (FR) which are defined as the functionality that is required to be provided by the system (for instance: "The system shall be able to transfer data via internet") and the non-functional requirements (NFR) are defined as the restrictions that should be amilied on the required functions (for instance: "The system shall be able to transfer data via internet with low response time").

In the academic field, several researchers have referred in their reports to the difficulties and challenges that the developers are faced to handle with NFR, for instance: taking NFR as a quantitative input to be measured and involved in the project budget estimation alongside with the FR [1-2]. Several approaches and methods are proposed from different Section 4 presents The Foundations of the proposed model of researcher's perspectives to facilitate dealing with these performance Requirements. A conclusion is presented in challenges; nevertheless, there is currently a lack of generic models for early addressing and measuring these requirements

This work was supported in part by the U.S. Department of Commerce ader Grant BS123456 (sponsor and financial support acknowledgment goes here). Paper titles should be written in uppercase and lowercase letters, not all uppercase. Avoid writing long formulas with subscripts in the title, short

Engineering, 13115 Zanya, Jordan. (e-mail: khalidnithu edu in)

at the system level and their related functional requirement at the software/hardware level [3-4].

In parallel with the academic field, international standards organizations (such as the ECSS and the IEEE) are interested in describing and categorizing the NFR types. Since the European Cooperation for Space Standardization (ECSS) and the Institute of Electrical and Electronics Engineer (IEEE) categorized the performance requirements as a single type of NFR and discussed them by various terminologies and views. This paper will account a new model for early specifications

of performance remirements at functional levels based on the finding of international standards in parallel with academic previous work of some of the respected models regarding non functional performance requirements as an self-sufficient model to identify the size of the software performance separately of the languages types, whereas keep away from the The proper identification, specification and measurement of limitations viewed in the performance measures presently

> The paper scope is to classify independently the all functionality allocated to software performance as a part of set pieces of the system application in the requirements phase for any software applications, whether the application has been built or it has already to be delivered.

> In addition, the main contribution of this paper is the proposed model of software performance requirements. The proposed nonspecific model is considered as type of a orientation model in the common sense of an etalon standard that is being used for the measurement of software

> This paper is organized as follows. Section 2 presents the related works. Section 3 presents Performance REQUIREMENTS as defined in International Standards. Section 4 presents The Foundations of the proposed model of

#### II. RELATED WORK

Many early efforts have been concerned with defining specifying and modeling NFR. For instance: [5] this paper proposed a performance requirements model; it joins together specimen, roots wiring any firmans with authority in its fact, their Privated<sup>2</sup> is the first all mass of authors are privated in the authority proposed 3 performance requirements model; a joint substitute to are not required. For a spec between authors instals.

The contraction of the contraction following performance conceptions, software performance

ISBN: 978-1-61804-290-3

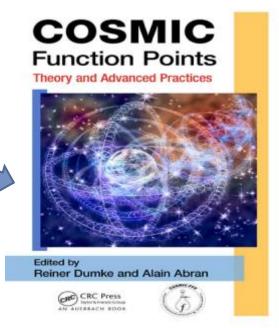
- Background to Functional Size Measurement (FSM) methods and their uses
- COSMIC FSM Method ISO 19761
- Measurement Guidelines
- 'The proof of the pudding is in the eating': Good Estimation
- Automation of COSMIC measurement
- Conclusions

### **COSMIC** data from Industry

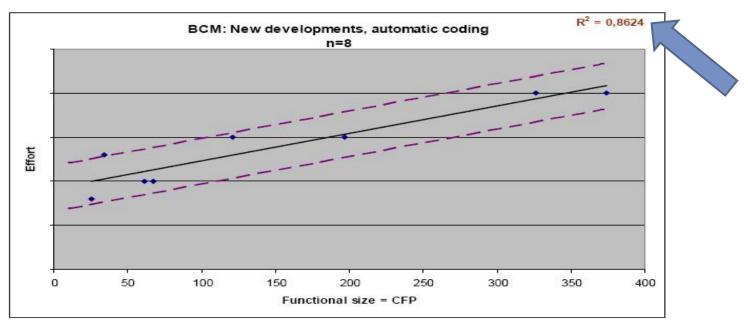
Practical experimentations with the COSMIC method in Automotive embedded software field

By: Sophie Stern



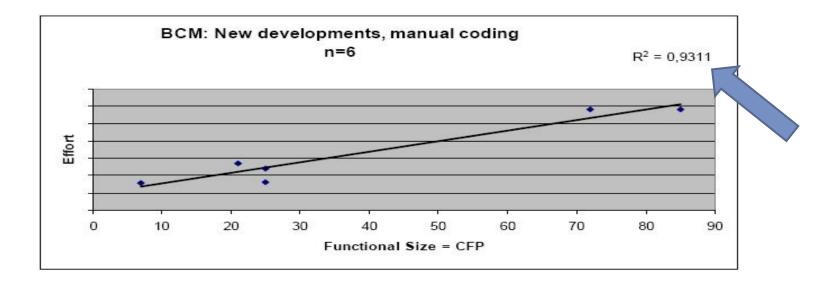


#### **Renault - 2012**



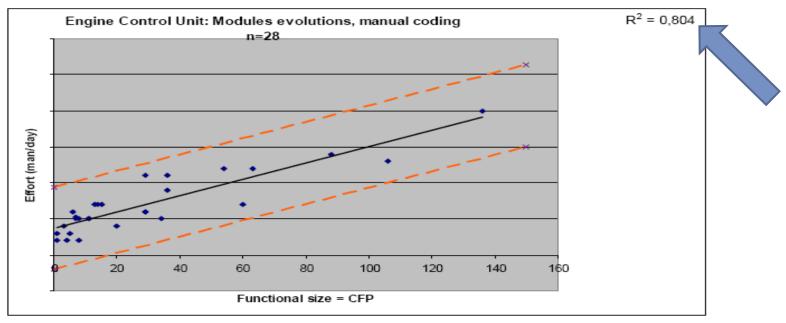
© Copyrights Renault 2012

### **Renault – 2012**



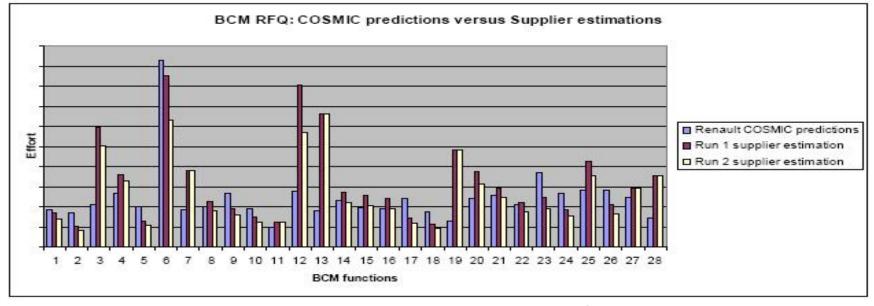
© Copyrights Renault 2012





© Copyrights Renault 2012

#### Renault: Estimation & Negociations

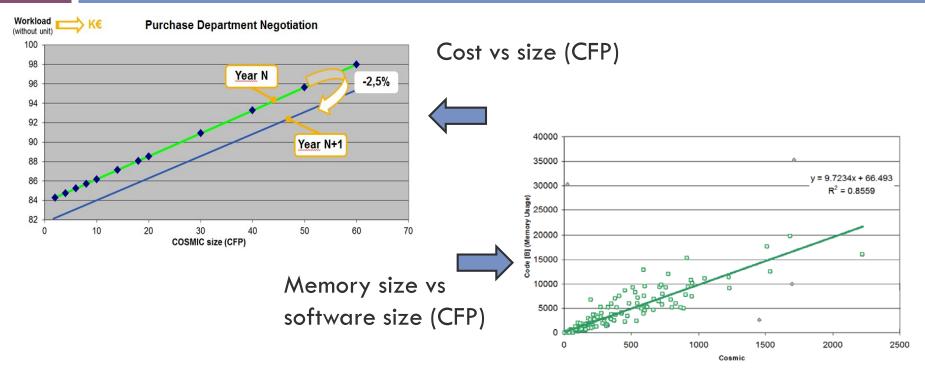


© Copyrights Renault 2012



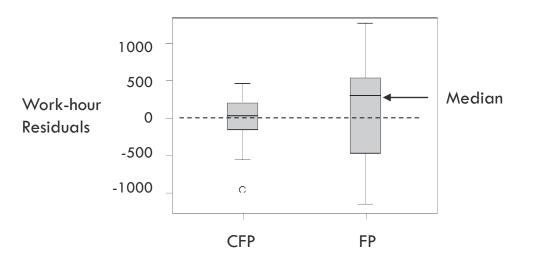
## Renault - Remarkable cost estimation accuracy from its ECU software specifications





# Case: Web effort estimation is more accurate with COSMIC than using classic FP

40



#### 25 industrial Web applications

#### Conclusions:

'The results of the ... study revealed that COSMIC outperformed Function Points as indicator of development effort by providing significantly better estimations'

Ref.: 'Web Effort Estimation: Function Point Analysis vs. COSMIC

Sergio Di Martinoa, Filomena Ferruccib,\*, Carmine Gravinob, Federica Sarroc

Information and Software Technology 72 (2016) 90–109



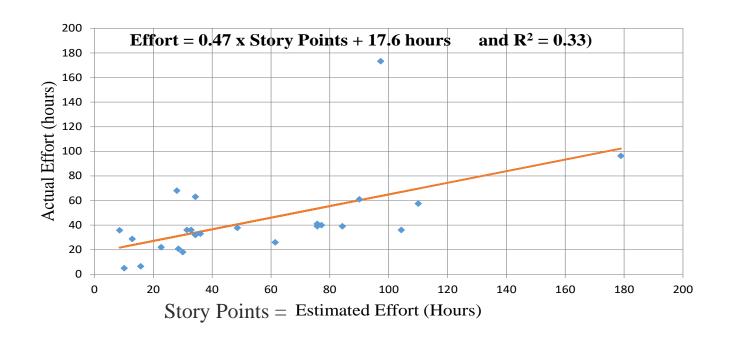
### Case: A Canadian supplier of security and surveillance software systems

- Uses Scrum method
- Teams estimate tasks within each iteration in Story Points, and convert directly to effort in work-hours
- Study involved measurements on 24 tasks in nine iterations
  - Each task estimated in Story Points Effort;
  - Task actual effort recorded
  - Each task also measured in CFP

Ref. 'Effort Estimation with Story Points and COSMIC Function Points - An Industry Case Study', C. Commeyne, A. Abran, R. Djouab. Obtainable from www.cosmic-sizing.org 'Software Measurement News'. Vol 21, No. 1, 2016

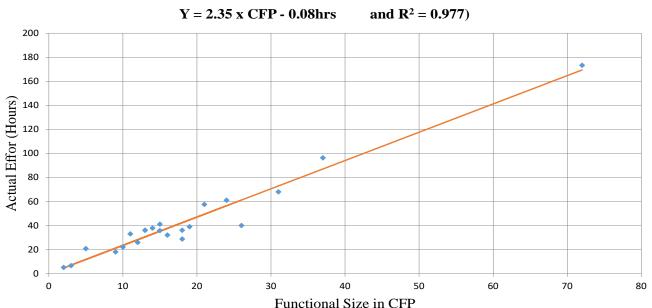


### Effort vs Story Points (24 tasks) = a poor predictor of effort





#### Effort vs COSMIC size is good for estimating



As a result of COSMIC measurement: two tasks were identified with very low effort/CFP.

These were found to involve significant software re-use, so were considered separately

© Copyrights 2016: COSMIC and authors

ICEAA Bristol (UK), Oct. 2016

### A User view of 'COSMIC for Agile'

"We have found that adopting this approach provides us with excellent predictability and comparability across projects, teams, time and technologies."

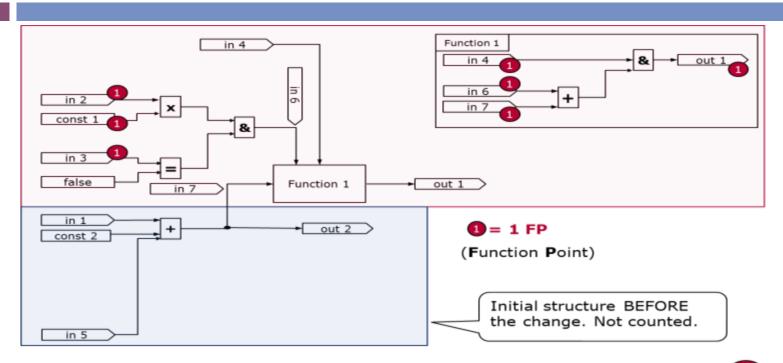
The reality of achieving predictable project performance has driven me to investigate many methods of prediction. COSMIC is the method that lets me sleep at night."

Denis Krizanovic, Aon Australia, August 2014



## Case Study — Vector (Germany): Estimating Maintenance Effort

45



Size<sub>FP</sub> (Change) = size(added data) + size(modified data) + size(deleted data) =

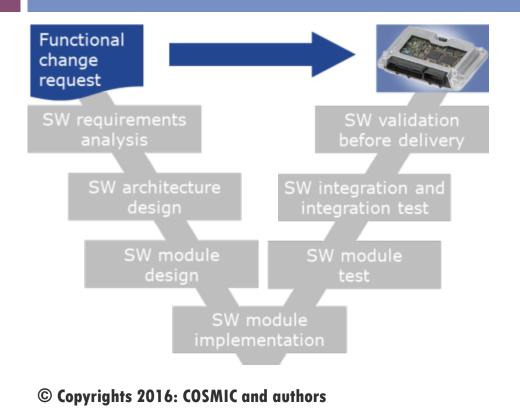
© Copyrights 2016: COSMIC and authors

ICEAA Bristol (UK), Oct. 2016



## Case Study — Vector (Germany): Maintenance Constraints

46



Requirements and design specification:
Consistent level of documentation

Modeling:
Same method,
notation, semantics
and visibility

Change documentation: All changes are covered and clearly marked

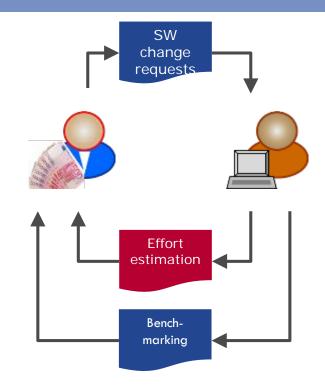
ICEAA Bristol (UK), Oct. 2016



## Case Study — Vector (Germany): COSMIC Benefits

47

- Agreed model for measuring functional size
- Solid baseline for benchmarking
- Transparent effort estimations on the basis of functional changes
- Ad-hoc & fuzzy evaluations and negotiations for single SW changes are reduced
- Significantly increased efficiency & trust for better collaboration between supplier & customer



### Case Study — Vector (Germany):

#### Results

- Vector achieved with many clients a preciseness of 10-20% within one year of building the estimation program.
- Consider business impacts
  - Clearly distinguish goals, estimates & plans
  - Challenge results & improve your efficiency each year
  - Don't stay with same parameters for over 1 year.
- Establish repeatability
  - Immature processes invalidate your overall estimation & ruin trust
  - Establish a robust process to report & store data
  - Insufficient data quality & environmental constraints need experienced counting to avoid errors & weakening the method.



#### Agenda

4

- Background to Functional Size Measurement (FSM) methods and their uses
- COSMIC FSM Method ISO 19761
- Measurement Guidelines
- 'The proof of the pudding is in the eating': Good Estimation
- Automation of COSMIC measurement
- Conclusions



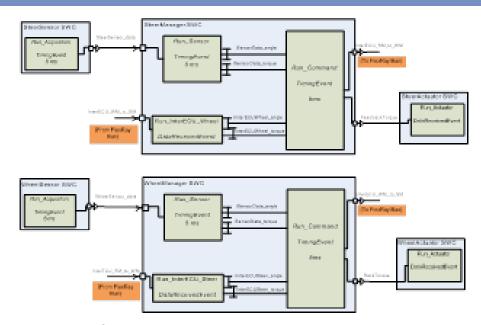
### COSMIC specifications for Automation with Matlab-Simulink

50

MENALEZ COSMIC Rules	Rev.: v 1.0 Date: 10 May 2012
COSMIC Rules for Embedded Softv	
Requirements Expressed using Simu	ilink®
ABSTRACT This document presents the rules for measuring with COSMIC (ISO 1976 software specifications documented with Matable-Simulania).	1) the functional size of
postage specifications documented that reduced united time.	
© REMAULT 2012 Page 1/5	CORRECTION
RENAULT PROPERTY	

#### 51

# Real-time embedded software specifications in graphical format — An example



#### Runnables inside the software components -Steer-by-Wire system

Ref. H. Soubra, and K. Chaaban, "Functional Size Measurement of Electronic Control Units Software Designed Following the AUTOSAR Standard: A Measurement Guideline Based on the COSMIC ISO 19761 Standard," IWSM-MENSURA Conference, Assisi (Italy), IEEE CS Press, 2012.

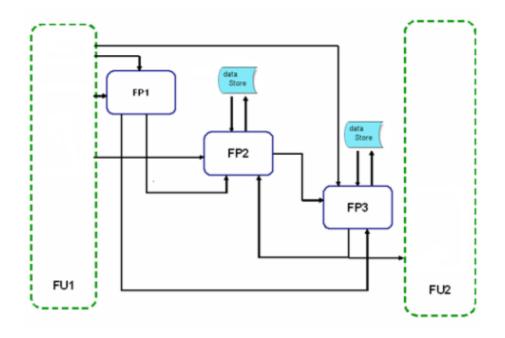


# When Requirements are described with Graphs: Map to COSMIC

#### 57

#### TABLE I.

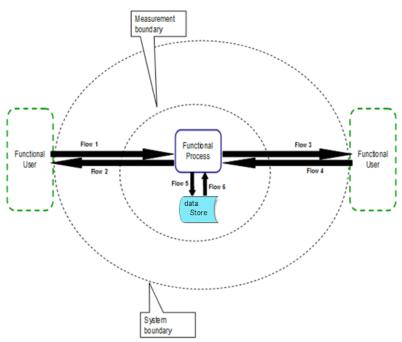
COSMIC concepts	COSMIC abbreviation	Proposed graphical representation	Proposed graphical description	
Functional user	FU		Green dashed box	
Functional process	FP		Blue box	
Data group movement	E/X/W/R		Black arrow	
Persistent storage		data Store	ISO 5807 stored data symbol in light blue	

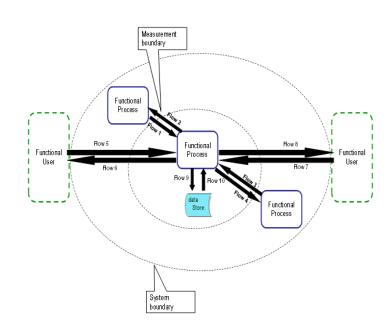




# COSMIC representation of functional process (Single & Many) — For testing scenarios

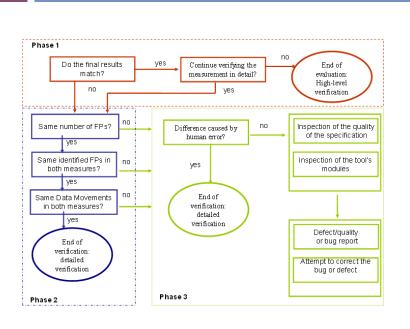
<u> 53</u>

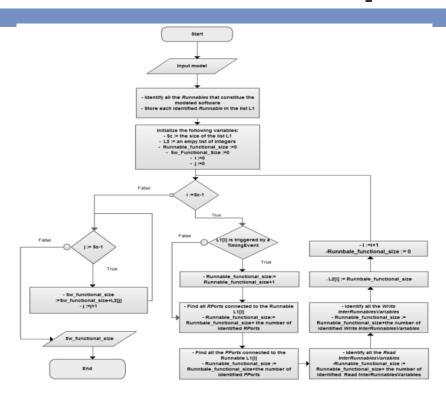






#### 3-Phase Verification Protocal of Automation Accuracy





#### **AUTOMATION ACCURACY REACHED WITH COSMIC**

Steer-by-Wire Runnable	Functional size obtained by the manual FSM procedure (CFP)	Functional size obtained by the automated FSM procedure (CFP)
Steer_Run_Acquisition	3	3
Steer_Run_Sensor	4	4
Steer_Run_Command	7	7
Steer_InterECU_Wheel	3	3
Steer_Run_Actuator	2	2
Wheel_Run_Acquistion	3	3
Wheel_Run_Sensor	4	4
Wheel_Run_Command	7	7
Wheel_InterECU_Steer	3	3
Wheel _Run_Actuator	2	2
Total	38	38

Total Number of Models	Total Size obtained manually (CFP)	Total Size obtained using the prototype tool (CFP)	Difference (%)	Accuracy
76 fault-free models	1,729	1,739	Less than 1%	>99%
All 77 models	1,758	1,791	1.8%	>98%



Ref.: Hassan Soubra, Alain Abran, A. R. Cherif,

'Verifying the Accuracy of Automation Tools for the Measurement of Software with COSMIC – ISO 19761 including an AUTOSAR-based Example and a Case Study,' Joint 24<sup>rd</sup> International Workshop on Software Measurement & 9<sup>th</sup> MENSURA Conference, Rotterdam (The Netherlands), Oct. 6-8, 2014, IEEE CS Press, pp. 23-31.

- Background to Functional Size Measurement (FSM) methods and their uses
- COSMIC FSM Method ISO 19761
- Measurement Guidelines
- 'The proof of the pudding is in the eating': Good Estimation
- Automation of COSMIC measurement
- Conclusions



#### The COSMIC method is very widely used

57

- COSMIC Measurement Manual standard (11 languages)
- Size of user base is unknown
  - Of known users, 50% are software houses
  - Adopted by two Governments (Mexico, Poland)
  - > 30,000 downloads of research & conference papers
- + 600 certification exam holders (notably Brazil, China. India, Italy, Poland, Turkey)
- Two active forums (on Linkedin CUG, www.cosmic-sizing.org)

### Summary of benefits

- Free, open
- Fundamental SE Principles: \_\_\_\_\_uture-proof, stable
- Very wide applicability
- Proven value for performance measurement & estimating
- ISO standard & GAO 1, NIST 2 endorsed
- Can be automated with very high accuracy & traceability
- 'Cost Estimating and Assessment Guide' http://www.gao.gov/new.items/d093sp.pdf , March 2009
- 'A Rational Foundation for Software Metrology', National Institute for Standards & Technology, NIST IR 8101, January 2016
  - © Copyrights 2016: COSMIC and authors



### Thank you for your attention

(www.cosmic-sizing.org)

Alain Abran <u>alain.abran@etsmtl.ca</u>
Charles Symons <u>cr.symons@btinternet.com</u>
Christof Ebert <u>christof.ebert@vector.com</u>
Frank Vogelezang <u>frank.Vogelezang@ordina.nl</u>
Hassan Soubra

### Acknowledgements

The authors wish to acknowledge the efforts of members of the COSMIC Measurement Practices Committee and many others who, over the last 18 years, have contributed to the development and implementation of the COSMIC method