Metrics Matters

E-JOURNAL OF THE AUSTRALIAN SOFTWARE METRICS ASSOCIATION

November, 2004

© The e-Journal of the Australian Software Metrics Association, circulated to members Australia wide.

ASMA National
www.asma.org.au
e-mail:
info@asma.org.au

The Publisher
e-mail:
Metricsmatters@asma.org.au

Welcome

Sophia Poulos

This issue we look at the recent Australian Conference on Software Measurement (ACOSM 2004) held in Melbourne. Our partner this year was the Australian Computer Society (ACS). Congratulations to the ACOSM organisers Robyn Lawrie, Paula Holmberg and Robyn Smith and many others from both ASMA and ACS who contributed to the production of an interesting and insightful conference.

Tom McBride gives us a perspective on the Professional Certification of Software Engineers and what might be involved, and Brian Nelson explains the use of "Operational" size to better quantify the size of some types of software.

As always, feedback on either the contents or the format of the Metrics Matters e-Journal is always welcome. Please email comments to metricsmatters@asma.org.au.

Best Regards,

Sophia Poulos

Editor/Publisher

Contents

ASMA/SQA NSW Merger

Open Letter from ASMA/SQA NSW Chairman, Helen Aitken

The NSW chapter of ASMA has now formally merged with SQA (NSW).

Over the past few years we have been working very closely together and now will merge completely. Our philosophy is to have one meeting a month with an interesting speaker. Speakers are from a range of backgrounds and often provoke some significant discussion. Many of the speakers provide notes which can be found on our website www.ozemail.com.au/~sqain.

As well as speakers, we endeavour to provide a monthly newsletter with thought provoking ideas. Our aim is to help members learn something about the latest industry trends - I am very aware how easy it is to concentrate on work and forget about the bigger industry picture. Speakers this year have included Watts Humphrey, Sue Scully from SEA and Andrew Cowie who gave a very interesting talk on the impact of September 11 on his office which was situated a block or so away from the twin towers in New York. We have also had a number of local speakers, again providing thought provoking discussions.

We are very aware that many members want different things and are constantly asking our members for suggestions. We updated the SQA website last year and since then have attracted a lot of new members and inquiries – including one from a Nigerian bank with suspicious details.

We are now running two mailing lists, currently on Yahoo, one for full members who get the magazine and one for full members and any other interested parties who get the flier and the odd e-mail with regards to requests for surveys etc. I find the people on this second list frequently ask for more information and / or progress to be full members. Please let me know if you wish to be put on our "interested parties" e-mail list.

In Sydney, SPIN, ASMA and SQA have been running some joint meetings every year since SPIN was started in 1996. The philosophies of the three organisations have been different but the committees have always been in close contact and have shared meeting and speaker information. When we discussed the merger with members, one Sydney SQA member who had recently moved to Melbourne asked why the various Melbourne groups didn't share information and meetings too.

Helen Aitken
ASMA / SQA NSW chairman
helenaitken@ozemail.com.au

New Zealand Software Metrics Association (NZSMA) - off and running

The Inaugural Meeting of the New Zealand Software Metrics Association was held on 27th October 2004.

According to Alan Henskie from EDS, "nineteen people from four different companies were represented and there

[was] interest from at least sixteen others from another three companies".

At this stage the group will be organised fairly informally until member numbers increase.

NZSMA have created a group on Yahoo called NZSMA. To subscribe to this group, please send an email to NZSMA-subscribe@yahoogroups.com. To contact the list owner, send an email to NZSMA-owner@yahoogroups.com. The Group name is NZSMA, and the homepage is located at http://groups.yahoo.com/group/NZSMA

The next meeting of the association will be on 9th February 2005 in Wellington, New Zealand. If you would like to attend the next meeting of the NZSMA, please contact either Richard Johnston (richard.johnston@eds.com, 04-474-5649) or Alan Henskie (alan.henskie@eds.com, 04-474-5890) before the event.

⌘

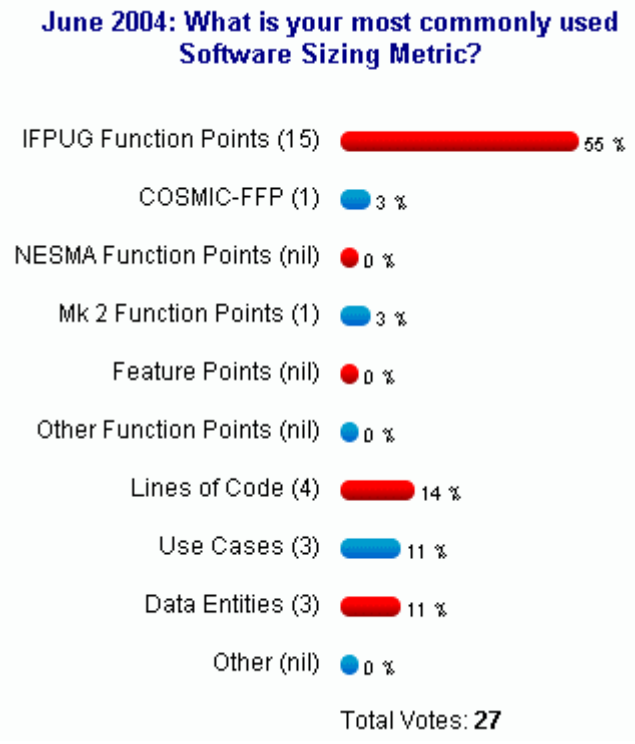
ASMA Monthly Poll Results

By David Cleary, ASMA National President

To vote in the current ASMA monthly poll, visit the ASMA website at www.asma.org.au.

Results for June 2004:

As expected, ASMA's first monthly poll highlighted the dominance of 'IFPUG Function Points' as the most popular software sizing metric with 15 votes.



Interestingly, and despite its well documented limitations, 'Lines of Code' still came in second place with 4 votes. A contributor to this may well be the continuing availability of various sophisticated 'Lines of Code' based estimating methods such as COCOMO and SLIM.

The other metrics to receive multiple votes were 'Use Cases' and 'Data Entities', both with 3 votes.

While much has certainly been written about leveraging the 'Use Cases' specifications for use in project estimation, it is usually deemed necessary to have some additional normalising mechanism to address the variation in size between individual 'Use Cases'. It would be very interesting to know, therefore, how the 'Use Cases' were being utilised as a sizing metric in these situations.

ACS 2004 Incorporating ACOSM 2004 Conference

By Sophia Poulos, IBM Global Services

Key Note Speakers:
Ben English
Hon. Dr Barry Jones
Gerry McGovern
Prof Michael Myers

The Australian Conference on Software Metrics (ACOSM) 2004 was jointly hosted by the Australian Computer Society (ACS) and ASMA.

With the theme "Sharing ideas and practical experience", the conference achieved its dual aims of bringing the software metrics community and the broader ICT community together in a shared forum.

David Cleary, ASMA National President - gives the opening address at the Banquet dinner.

Key Note Speakers

The keynote speakers were of a very high standard. Ben English from Microsoft looked at security issues in the IT Industry, Barry Jones covered the support (or lack) of the IT Industry from both government and academic agencies. Gerry McGovern talked about the turnaround of the Irish IT industry from backwater to a real European powerhouse. Gerry's key message was "embrace globalisation".

Another trend Gerry noted was that information is invaluable, is a process and can point to a measure of success. Information should promote action.

Also, that information is "like milk" and can quickly "go off" if not used. This is in contrast to the past where information was treated "like gold" and hoarded.

In this age, lots of data is coming to consumers, and there is a tendency for "information overload", so that the message to consumers is not being absorbed. The solution is to minimise input while maximimising output.

There are new roles being defined such as Knowledge worker and Information manager to cope with the growth in management of "knowledge".

Finally, Gerry summarised some key initiatives that should be undertaken to ensure success in this information-rich environment. Referred to as the 6 C's

Care	Understand what the customer cares about
Compelling	The message given must be compelling
Clear	The message given must be clear
Complete	Must bring the consumer to an action
Concise	Smallest number of words possible
Correct	trust is fundamental and dependent on accuracy

Another keynote speaker, Professor Michael Myers, spoke to the challenges facing the implementation of enterprise

wide applications and the challenges faced by implementing systems in a politically-sensitive environment. Refer to Mark Exall's comment on Prof. Michael Myers key note presentation in this e-Journal (see Last Words…).

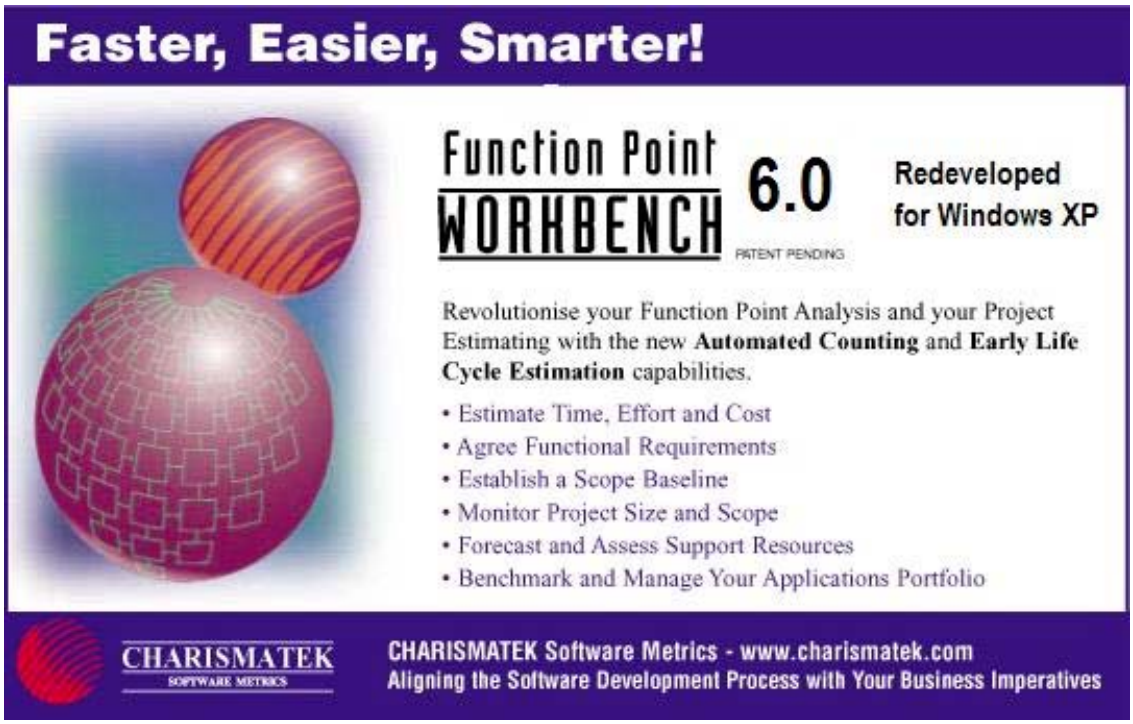
Global Offshoring Panel

One of the more controversial sessions of the conference related to a panel session discussing the question of global offshoring. There were a number of eminent people on the panel discussion from government, media, ACS and corporate areas.

The message was that global offshoring was here to stay and that the Australian software industry needs to accept this reality and manage this impact. Global outsourcing is driven by customer requests and is part of the "globalisation" of industry.

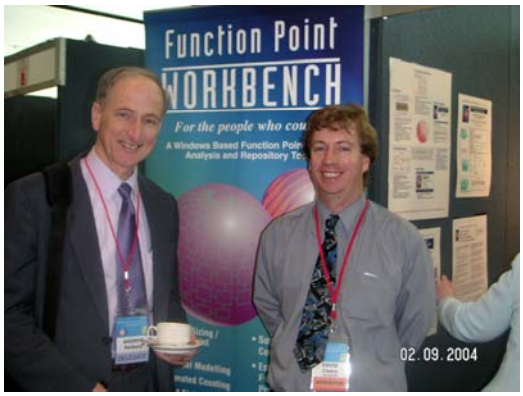
Off Shoring Panel Discussion, (from left), Graham Philipson, Gary Ebeyan, Karl Reed, John Gwyther, Eward Mandla, Ian Dennis.

It was suggested that the Australian government could be doing more to promote the IT industry overseas, so that Australia is seen to be more than a "tourist destination". And that further government support could be given to support the IT industry as has been the case in other small countries such as Japan, Singapore and some European countries.



Conference Photo-Journal

Here are a selection of pictures from the ACOSM 2004 and ACS 2004 conference. See you again at the next ACOSM 2005 conference! (Photos: Sophia Poulos)



Tea break Andrew Parbury and David Cleary (Charismatek)

ACOSM Conference Committee member, Paula Holmberg (IBM)

Conference Organiser Robyn Lawrie (Charismatek) with Judy Bamberger at the Melbourne Aquarium function

Photo opportunity at the Banquet dinner

Talking business, Ian Smith (IBM) and Ewa Wasylkowski (Total Metrics)

Certification for ICT Professionals

By Tom McBride, MACS, Chairman ACS National Standards Committee

The failings of software systems with spectacularly serious consequences has been tracked and documented for some time. The failings don't seem to be any different from failings in other professions, neither more nor less serious. But what is different about software is that anyone can practice without any qualifications or restrictions.

This doesn't seem to be a problem when the software is developed for personal use. Then it is like someone building a shed out the back or changing the oil in their car. Not something requiring great skill and not something of great consequence.

But when the software has the potential to cause physical or economic damage the situation is different. The profession of engineering accepts the responsibilities of engineering along with the rewards of engineering, the freedom to build neat things. Those responsibilities include the legal responsibility for things going wrong.

There are different forms of regulation. There is licensing and there is certification. Licensing is the most restrictive form of professional and occupational regulation. Under licensure laws, it is illegal for a person to practice without first meeting the relevant requirements. For example, you need a licence to perform electrical work or to perform plumbing. You need a license to perform any medical procedure, dental procedure but not to give financial advice. You don't need a license to write software nor does there seem to be any move to license programmers in Australia. In the USA each state can have its own licensing laws and in Texas you need a license to call yourself a software engineer, and presumably to practice software engineering, but not to call yourself a programmer.

Certification is an occupational designation issued by an organisation that provides confirmation of an individual's qualifications in a specified profession. It is voluntary. It provides an assurance that an individual possesses the specific knowledge or skill level pertaining to an occupation. It is neither a barrier nor gate to employment. It is simply an association's recognition of your knowledge and skills. A bit like a university degree or any other educational qualification.

Certification schemes always seem to conjure up visions of Big Brother and heavy handed regulation restricting the ability to do some really creative things. We fear that some imbecile will insist that some outmoded regulation be adhered to, that they will be a dead hand of regulation with no discernible benefit. Worse we fear that we will be caught in the contradictory position of having to do as we are told yet being held legally responsible for the consequences. Civil engineers, doctors, architects and other professions have lived with certification schemes for a long time, and they seem to cope. For how long will the IT industry refuse to accept responsibility for the disasters they allow.

"For how long will the IT industry refuse to accept responsibility for the disasters they allow?"

The Professional Standards Act 1994 (NSW) contained some slightly different requirements for individuals in the area of risk management and required a complaints and discipline system of the governing body. This didn't seem to matter until recently. The ICT sector was happily going about its business and didn't seem to need any form of professional standards. Two things changed. The first was that professional indemnity insurance premiums rose alarmingly and the second was that more people were employed as contractors. When you are an employee you are protected to some extent from personal responsibility for products or services sold by the company. Being unsure of just how protected or unprotected you are as an employee I'll leave that there. If, on the other hand, you are a consultant then you are responsible for your own actions and if your advice or actions cause damage then you are liable. Not a significant problem until insurance premiums rose alarmingly.

The Professional Standards Council's Cover of Excellence offers a cap on professional indemnity in exchange for meeting their requirements for professional associations. The scheme applies to the association, not the individuals. In general this means that the ACS determines what knowledge a certified professional must have, administer a code of ethics and a complaints scheme. Part of the knowledge requirements include maintenance of that knowledge.

"The ACS is developing its Certified ICT Professional (CICTP) scheme."

The ACS is developing its Certified ICT Professional (CICTP) scheme. Part of that development is a draft policy currently out for member comment, but there is no schedule for completion yet. In the past there has been the Certified Member of the Australian Computer Society (CMACS) and the Practicing Computer Professional (PCP). While these are aimed at professional recognition they don't meet the requirements of the Professional Standards Council.

On the international scene, the IEEE has a scheme to certify software developers (CSDP) and the UK has several different schemes aimed at different levels. Europe has a European Certification of IT Professionals.

ISO has started a study group to look into an international certification of software engineers. The scope is restricted to software engineers because the particular ISO committee deals with Software and Systems Engineering so isn't really able to develop any standards outside that area. Nevertheless, there is a study group looking at the problem to see if an international standard can serve a useful purpose in the area. This initiative has just started and the normal ISO schedule would see a standard emerge in about five years.

A Chartered Professional Engineer is something else again. In UK and its former colonies various professions were given a charter that gave them exclusive rights to practice in exchange for maintaining the standards of the members. In other words, if they maintained an acceptable level of professional competence and conduct among their members, they enjoyed a protected monopoly. The association is obliged to regulate its membership and to maintain currency. The Australian Institute of Engineers maintains a four level competency scheme that is in addition to any obligations of

membership. You can be a member of IEAust without ever being a CPEng.

Internationally, several countries have negotiated an agreement to recognise each others' qualifications for both engineers, through the Washington Accord, and Engineering Technologists through the Sydney Accord.

So why would you want to become a Chartered Engineer? Because some jobs demand such a demonstration of competence. And why wouldn't they. If someone is going to be the engineer in charge of something that can kill me, I'd want to know that they were competent to ensure that it didn't kill me. I'd like to know that some unforseen combination of circumstances won't lose my life savings, that the medical machinery won't subject me to a lethal dose, and that some transmission error won't consign me to some bureaucratic hell as I travel about this planet.

"So far the industry has relied on reputation….[but] the job of assessing competence [is becoming] harder."

If you want to be an engineer and call yourself a software engineer that gets into a whole new area. There have been some quiet rumblings for some time over using the term "engineer". Professional engineering associations all over the world want to preserve the professional title of "engineer" for those who practice "engineering". In particular they don't want the term to mean one thing for a civil engineer and something entirely different for a software engineer. So the Australian Institute of Engineers (IEAust) and the Australian Computer Society have formed a Joint Board to decide what knowledge and skills someone would need to have in order to be a software engineer. This particular scheme is aimed at professional competence.

Where this all comes together is when you consider how anyone can know or find out if the person they want to engage for a specific range of duties is competent to carry out those duties professionally. If the particular task is of small consequence or there is someone available who can supervise what they do and detect any shortcomings in that person's skills then the task of assessing their competence doesn't matter too much.

But as the consequences mount up and the ability to supervise them, or indeed to understand what it is they do, declines then the job of assessing their competence becomes harder.

So far the industry has relied on reputation. That is, references are checked to confirm that the person did what they said they did and conducted themselves professionally. Standards of knowledge and skills vary with the particular organisation having one organisation say that someone was competent in one place doesn't mean they will be competent in another.

Professional certification puts a uniform measure in place and relieves potential employers of much of the burden of assessing someone's professional skills.

Some industries are becoming very tired of software project failures. They are actively seeking some way to reduce their chance of failure. They have tried ISO 9001, tried someone's magic methodology, invested in all the technology and still projects seem to fail. Reports from IEAust are that industry wants something more than a certificate of knowledge and wants something that will certify competence.

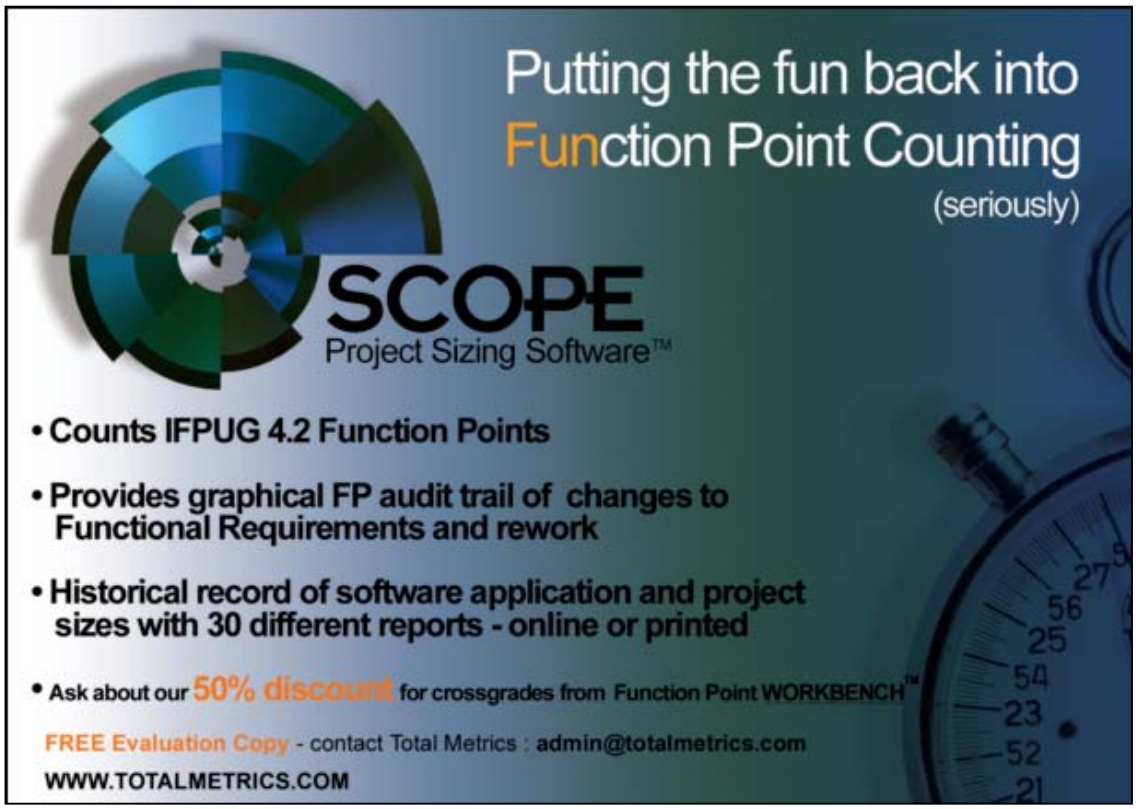
The various qualifications do serve to attest to varying levels of knowledge, skills and competence. Industry needs to be assured of professional knowledge, skills and competence so it shouldn't be long before professional engagement depends on professional certification.

References

The following sources were used in this article.

European Certification of IT Professional http://www.bcs.org/BCS/Products/Qualifications/EUCIP/
British Computer Society Chartered Professional Status http://www.bcs.org/BCS/Join/Grades/Chartered.htm
IEEE Certified Software Development Professional http://www.computer.org/certification/
Certification Road Map: The Journey and the Destination http://www.computer.org/certification/cert_for_you.htm
Professional Certification: Fact or Fiction http://www.computer.org/certification/FactorFiction.htm
Australian Computer Society: Certification http://www.acs.org.au/certification/index.cfm
Australian Institute of Engineers: Professional Development http://www.ieaust.org.au/membership/professional.html
The Washington Accord http://www.washingtonaccord.org/
Skills Foundation for the Information Age http://www.sfia.org.uk

⌘



Using Operational Size in the Real World

By Brian Nelson, Centrelink

Abstract

Operational size measurement can inform the selection of application architecture and justify high effort per (IFPUG/NESMA) function point for systems with complex application architecture.

This article builds on Eberhard Rudolph's concept of operational size measurement [1], by providing a practical method for measuring operational size in the context of programming effort within the application architecture.

To demonstrate the technique the real application architectures Microsoft Access, Java Servlets and Web Services are compared.

Operational Size

Operational size represents the actual size of installed software based not only on user requirements but also the IT architecture upon which the user requirements are delivered.

Due to differences in application architecture (among other things), the effort required to develop the same user functionality varies between environments. This article focuses on application architecture in order to measure user functionality in the specific context of the programming work done by application developers.

By measuring the *operational size* of software, based on the conceptual view of the underlying application architecture and the user requirements, the *software overhead* and *software efficiency* for different application architectures can be calculated and used to inform the selection of application architecture or to justify high effort per (IFPUG/NESMA) function point for systems with complex application architecture.

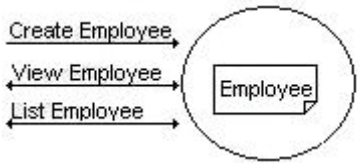
To illustrate the measurement and use of *operational size* the *software overhead* and *software efficiency* for the application architectures of Microsoft Access, Java Servlets and Web Services are compared. The functional size is measured in IFPUG function points. The operational size is measured based on the conceptual view of the application architecture instead of the application boundary. To avoid confusion, operational size measurement is expressed as Operation Points (OP).

Case Study

To demonstrate *operational size*, *software overhead* and *software efficiency*, a user requirement for maintaining employee details is presented below. The functional representation is shown, followed by the operational representation in three different application architectures with the resultant *software overhead* and *software efficiency* calculations.

The User Requirement

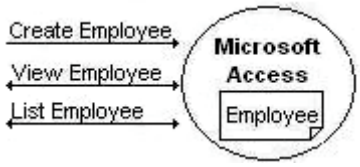
Create Employee	3 IFPUG FP (EI <16 DET, 1 FTR)
View Employee	3 IFPUG FP (EQ <20 DET, 1 FTR)
List Employees	3 IFPUG FP (EQ <20 DET, 1 FTR)
Employee File	7 IFPUG FP (ILF <20 DET, 1 RET)
Net Software Size	16 IFPUG Unadjusted Function Points (functional size)



Application Architecture 1

The User Requirement Delivered in Microsoft Access

Create Employee	3 OP (EI <16 DET, 1 FTR)
View Employee	3 OP (EQ <20 DET, 1 FTR)
List Employees	3 OP (EQ <20 DET, 1 FTR)
Employee File	7 OP (ILF <20 DET, 1 RET)
Gross Software Size	16 OP (operational size)



Software Overhead = Operational Size – Functional Size = 16 – 16 = 0

Software Efficiency = Functional Size/ Operational Size = 16 / 16 = 1 = 100 %

In this standalone application, all functionality is written/developed in one development environment with only one architectural boundary, there is no overhead and the software is 100% efficient.

Application Architecture 2

The User Requirement Delivered in a Java Servlet and any SQL Database

Information Tier (Data Base Management System):

Employee File	7 OP (ILF <20 DET, 1 RET)

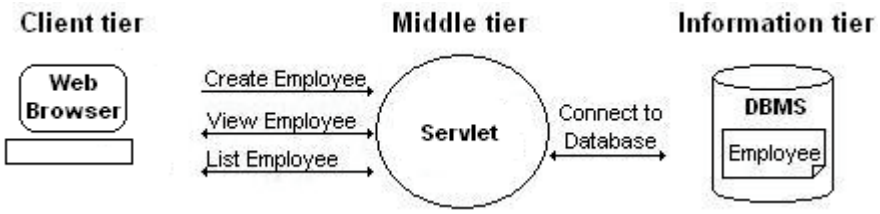
Middle Tier (Servlet):

Connect to Database	3 OP (EQ 2 DETs (query and result), 1 FTR (the database))
Create Employee	3 OP (EI <16 DET, 1 FTR (Employee File))
View Employee	3 OP (EQ <20 DET, 1 FTR (Employee File))
List Employees	3 OP (EQ <20 DET, 1 FTR (Employee File))

Client Tier (Web Browser):

0 OP

Gross Software Size	19 OP (operational size)



Software Overhead = Operational Size – Functional Size = 19 – 16 = 3

Software Efficiency = Functional Size / Operational Size = 16 / 19 = 0.84 = 84 %

In this web application the functionality is written in two development environments. The servlet is written in Java and the database is written in SQL.

All the transactional functions for the project are written in the servlet. The middle tier connects to the information tier to send SQL queries and receive back results. The middle tier also allows connection from the client tier. The middle tier (servlet) receives data from the web browser and sends back HTML (written at run time by the servlet).

No transactional functions are counted for the DBMS (information tier) or the web browser (client tier) because none are built for this project (an employee system must be built, not a web browser and not a database management system).

Application Architecture 3

The User Requirement Delivered via Web Services (B2B)

Information Tier (Data Base Management System):

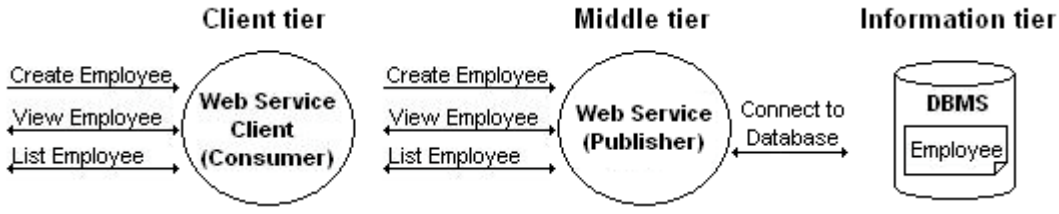
Employee File	7 OP (ILF <20 DET, 1 RET)

Middle Tier (Web Service Publisher):

Connect to Database	3 OP (EQ 2 DETs (query and result), 1 FTR (the database))
Create Employee	3 OP (EI <16 DET, 1 FTR (Employee File))
View Employee	3 OP (EQ <20 DET, 1 FTR (Employee File))
List Employees	3 OP (EQ <20 DET, 1 FTR (Employee File))

Client Tier (Web Service Consumer):

Create Employee	3 OP (EI <16 DET, 0 FTR)
View Employee	3 OP (EQ <20 DET, 0 FTR)
List Employees	3 OP (EQ <20 DET, 0 FTR)
Gross Software Size	28 OP (operational size)



Software Overhead = Operational Size – Functional Size = 28 – 16 = 12

Software Efficiency = Functional Size / Operational Size = 16 / 28 = 0.57 = 57 %

In this web application the functionality is written in three development environments. The database is written in SQL, the Publisher is written in C++ .NET and the consumer is written in Java. There are connections between all three tiers. The connection between the information tier and the middle tier is the same as above. The connection between the middle tier and the client tier is client server. The middle tier publishes the three transaction functions (web services). The client tier consumes the web services. To consume a web service is not an elementary process, it is part of the transaction function provided to the user.

Analysis

Application Architecture	Software Overhead	Software Efficiency
Microsoft Access	0 (constant)	100 % (constant)
Servlets	3 (constant)	84 % (approaching 100 % for functional size increases)
Web Services	12 (increasing with project size)	57 % (fairly constant)

From the results above it is apparent that application architectures **can** be characterised according to the relationship between functional size and operational size.

For the standalone application, *software overhead* and software efficiency is constant regardless of functional size.

For the servlet, *software overhead* is constant regardless of functional size and *software efficiency* approaches 100 % as functional size increases.

For the web service, *software overhead* increases as functional size increases and *software efficiency* remains fairly constant as functional size increases (influenced mainly by the number of Firs in transaction functions).

Because we can predict the correlation between functional size and operational size we do not need to measure operational size for all user functions. A subset of operational sizes from each application component, compared to the corresponding subset of user functions, is sufficient to make judgements about the *software efficiency* and *software overhead* of given application architectures.

Quite clearly, the more places procedural code has to be written to accomplish a user task the worse the *software overhead* and *software efficiency*.

Conclusions

By measuring the *operational size* of the software developed and calculating *software overhead* and *software efficiency,* it is evident that different application architectures require different amounts of code to be written and therefore require different amounts of development effort during code writing.

Many organisations have turned to web services and other application architectures to enable online access to their mainframe systems. By measuring the efficiency of our application architectures we can justify the additional development and maintenance costs due to complex application architecture.

Selecting or designing application architecture is a very important decision for an organisation. By measuring just a small amount of an organisation's business functionality, from a functional point of view and an operational point of view, we can quickly compare the efficiency of proposed application architectures to make a more informed choice.

It is the author's view that project estimation should be performed based on functional size measurement and historical effort figures for the same development environment and similar business requirements. A development environment is more than a programming language and a hardware platform. Among other things, a development environment includes the application architecture.

By calculating *software overhead* and *software efficiency* it has been demonstrated that a simple web application is more expensive than a standalone application and a complex web application is more expensive again, but this is not the whole story, operational size is only one attribute software development.

In practice standalone applications are small, less complex and development is less formal. The whole project has fewer overheads, not just the code.

When interfaces between different organisations are required web services are the contemporary solution. Such interfaces are measured effectively using IFPUG and NESMA counting practices. While IT architecture (including application architecture) is a cost driver, when organisations have to interface the real cost blowouts come from the meshing of different organisational cultures.

Established software estimation models are based on the *operational size*. Organisations conducting project estimation with software estimation models based on *operational size* could consider *software efficiency* when converting from function points to lines of code but the ideal solution would be to develop software estimating models based on functional size and *software efficiency*.

Reference

[1] Rudolph, E.: 2004, "Operational Software Size", ACOSM 2004 Melbourne 2-4/9/2004

ASMA Products

ASMA has generated an extensive library of information on software measurement, establishing measurement programs, estimating and benchmarking. Much of this information is available for purchase in the form of books, tools, conference proceedings and reports.

Please note:

- ASMA and Australian Computer Society (ACS) members are eligible to purchase at the member price.
- All prices are in Australian dollars.
- All Australian buyers must purchase at the GST inclusive price.

Product	Member Price* (GST Inclusive)	Non-Member Price* (GST Inclusive)
The Software Metrics Compendium	A$198	A$253
Practical Project Estimation Toolkit	A$165	A$220
The Benchmark - Release 9	A$165	A$220
ISBSG Estimating, Benchmarking & Research Suite Release 9		
Single CD Pack	A$1,320	A$1,820
5 CD Pack (site license)	A$2,500	A$3,000
10 CD Pack (site license)	A$3,300	A$3,800
Combination Pack 1	A$1,420	A$1,920
ISBSG Estimating, Benchmarking & Research Suite Release 9		
The Benchmark - Release 9		
Combination Pack 2	A$1,450	A$1,950
ISBSG Estimating, Benchmarking & Research Suite Release 9		
The Software Metrics Compendium		

* prices subject to change without notice.

⌘

ASMA Contacts

ASMA NATIONAL - for membership enquiries : President: David Cleary Administration Support: Robyn Smith	Phone: 03 9723 5145 Fax: 03 9723 8545 Mobile: 0411 649909 Email: info@asma.org.au
Postal Address	ASMA, PO Box 127, Warrandyte VIC 3113

ASMA National 2005 Executive Committee*

POSITION	NAME	EMAIL
President	David Cleary	david.cleary@charismatek.com
Vice President	Pam Morris	Pam.Morris@totalmetrics.com
Administrative Support	Robyn Smith	info@asma.org.au
Metrics Forum Coordinator	John McRae	John.mcrae@mcraeit.com.au
Marketing	Anne Goucher	agoucher@ozemail.com.au
Committee Members	Terry Wright George Ansell Sophia Poulos Kim Olsen Brent Cahill Alan Henskie Eberhard Rudolph Paul Holmberg Ann Goucher Erin Cracknell Paul Radford John McRae Gary Chitty	

About Metrics Matters

Publisher and Copy Editor: Sophia Poulos
Editing: Mark Exall

Advertisements
If you wish to advertise your products or services in Metrics Matters and reach metrics-aware individuals and companies, please contact the Publisher.

Disclaimer
The opinions expressed here are those of the individual authors and are not necessarily those of the Publisher, Editor or the Australian Software Metrics Association.

Permission is granted to reproduce any item from Metrics Matters on the condition that Metrics Matters and ASMA are fully acknowledged as the source.

Contributions...

Metrics Matters is seeking contributions from IT Professionals. If you have any form of contribution, please email them to "The Publisher" on metricsmatters@asma.org.au. The document must be in MS Word 97 or Rich Text Format (.rtf). Pictures or advertisements may be in pdf format.

Last Words...

ACS/ACOSM 2004 Conference Keynote: The Challenges of Implementing Enterprise-Wide Information Systems – A Software Metrics Context

By Mark Exall, Centrelink

Ultimately, software metrics is about improving present and future performance using historical evidence. The keynote by Professor Myers reminds us of a broader set of criteria in which we need to assess the performance of information systems. Many metrics we use only represent elements of the whole picture. While Professor Myers presented a compelling case for us to reassess how we judge success, and challenges conventional wisdom on how to achieve successful outcome, he did not offer solutions for improving performance. Maybe it is because it is up to us to do, or it could be there are no universal solutions to political and social problems. While the keynote was thought provoking, the author was still left pondering where software metrics fits into the bigger picture.

When success can turn into failure

Professor Michael D Myers keynote speech represented the human side of implementing information systems in an organisation, a perspective easily ignored by decision makers, with human factors accounting for many project failures. While the implementation of an enterprise-wide information system may be a technical success, coming in on time and under budget, it can often be a business failure that can threaten an organisation's competitiveness.

The evidence from the four case studies was clear. All four projects were considered highly successful, and yet they turned into major business failures. One was so disastrous that the company that implemented an enterprise-wide system was subject to a takeover and the system was replaced. In the case studies, the implemented information system was either tolerated, with the organisation seemingly operating despite the system, or it was abandoned. Yet, all of these systems were initially considered a success. They also adhered to the conventional wisdom for developing information systems, such as adopting world's best practice, getting top management support, and buy-in from users, in order to achieve a successful outcome.

So what went wrong?

One failing of conventional wisdom was to consider the environment within which the system was to be implemented. According to Dr Myers, each information system fell short for social or political reasons! Therefore, technical success is only a part of the overall scorecard upon which performance should be judged. However, it is the technical indicators that are often emphasised and used.

It follows that if human factors could be eliminated from the equation, there would be a far greater number of successes. There is a strong argument that the use of technology must have a business basis, otherwise its introduction is an unnecessary waste of resources. However we operate in an environment where it is people who implement new information systems, and if they are not prepared, or are resistant to change for any reason, then the outcome will fall short of meeting the original business intention. There needs to be buy in at a social and political level, past, present and future. Without it, we are

compromising the success of the implemented information system.

What is the solution?

While Professor Myers was excellent at illustrating the reasons for success and failure, and for exposing the conventional wisdom as incomplete, he did not explore alternative views on implementing enterprise-wide information systems. As often is the case, it is easy, and necessary, to point out the problem. However, it is far more challenging to develop a set of collective wisdoms that offer a viable solution.

The challenge ahead of us

Where does software metrics play a role? Are we a part of conventional wisdom or can we shed some new light the problem? What type of measures would be effective at monitoring and improving overall performance of information systems, including measures for political and social acceptance? Maybe we should develop new measures, or highlighting existing measures that gauge the business success of information systems, taking these factors into consideration, to act as an early warning sign to improve the end outcome.

Is it up to us to broaden our view, if software metrics is to play a major role at a higher level? After all, part of our objective is to improve the performance of information systems. Therefore, as participants in the IT industry, we have a responsibility to improve general awareness of potential barriers to the success of any information systems project.

The final word…

Thankyou Professor Myers for broadening our view of the world. The content of your keynote left a lasting impression and it throws down the challenge to us in the IT industry to improve the way we approach the development and implementation of information systems.

ASMA Web site

www.asma.org.au

Australian Software Metrics Association

Home Page

Membership

Events

Products

Activities

Service Providers

Contacts