

The Quality Mark

The Newsletter for the Kitchener-Waterloo Software Quality Association

Spring 2001

Volume 1, Issue 3

<http://www.kwsqa.org/>

Testing... Testing... 1-2-3...

By Rosalind Mountford

When I was asked to write an article on testing, I thought: it's a big enough subject, so what's the problem? Well for me, that *is* the problem. Where do you start when talking about testing? And how do you say things that others haven't already said before, and probably with more eloquence?

Well, I guess we can always start at the beginning - What is Testing?

The Oxford English Dictionary* describes test as: "that by which the existence, quality, or genuineness of anything is, or may be, determined." I could continue with numerous other definitions of testing, but why? We all know what testing is, don't we?

What makes testing challenging?
There are both technical and business reasons:

- Complexity of the application
- Lack of documentation
- Unavailability of useable data
- Risk Assessment Knowledge
- Communication
- Changing specs
- Time
- Perception of overhead
- Failure to see the benefits

When done properly, a test is a formal procedure where:

- Tests are documented
- Inputs are prepared
- Outputs are predicted
- Commands are executed
- Results are observed

money. How does it do this? By detecting defects - not just in the code, but also in the design and the analysis, thereby increasing confidence in both the design and analysis, and reducing risk.

Testing then, is the process that leads to customer satisfaction, or if done correctly, maybe even customer delight!

Far too often testing is viewed as the process that takes place at the *end* of development. The application is handed to the tester who then executes and verifies the test cases, and so on. Why then, if this is the view, do we testers claim that ours is an "extremely creative and intellectually challenging task" (to quote from Myers)? Well maybe, it's because if done properly, there is more to testing than pounding through test cases.

It is important to have a thorough understanding of the product's specifications, what it's supposed to do, and what it's not supposed to do. We must verify, that the requirements have been correctly translated into system specs, that the system specs have been correctly translated into program specs, and that the code produced complies with the program specs.

So why is testing important? Testing is important because it adds value to the software product by improving quality and saving

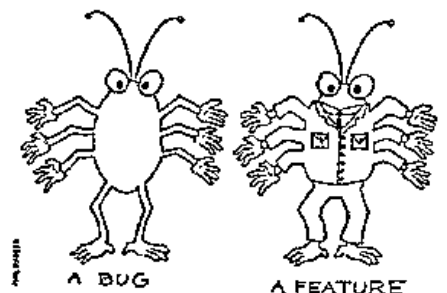
*Please see the Software QA and Testing FAQ for further definitions at: <http://www.softwareqatest.com/qatfaq1.html>

TOM



WHAT'S INSIDE...

Quality Article.....	2
IQ Test for Professionals.....	3
Tips and Tricks.....	4
Quizzes & Challenges.....	6
KWSQA Corner.....	7
Events Calendar.....	8



"Quotable Quotes"

On System Testing vs. Unit Testing:

"Testing only to end-user perceived requirements is like inspecting a building based on the work done by the interior decorator at the expense of the foundations, girders, and plumbing."

- Boris Beizer

People forget how fast you did a job - but they always remember how well you did it.

-- Howard Newton

A common mistake that people make when trying to design something completely foolproof is to underestimate the ingenuity of complete fools.

-- Douglas Adams

If I cannot do great things, I can do small things in a great way.

--Katherine Brown

You can pay us a little now, or pay us a lot more later.

-- Sign in auto dealership suggesting a tune-up

Quality Article



AN INTRODUCTION TO METRICS IN SQA

By Mechelle Gittens

There are many metrics and models that are used to ensure quality in a software development project. In outlining that quality, several definitions have been utilized by those in industry and academia. The U.S. Department of Defense in the March 1992 draft of MIL-STD-2168A has defined quality as ⁱ:

The conformance to explicitly stated functional and performance requirements, explicitly documented development standards, and implicit characteristics, such as maintainability and modularity

This definition is an expansion of a definition presented by Crosby in 1979, which stated that: *Quality is a conformance to requirements*. These definitions (and others) succeed in covering software products and processes, and recognize: that quality should be defined and measured against some standard; that there are differing aspects of quality; and that the performance standards should be applicable to all software products.

In considering these varying definitions that all software professionals seek to achieve in their development process, David Garvin has observed varying perceptions of quality in a number of disciplines. He concludes that "quality is a complex and multifaceted concept," and outlines the following five perspectives ⁱⁱ:

- The **transcendental view** sees quality as intangible - as some ideal that all projects strive towards but never achieve. It is unquantifiable.
- The **user view** observes a product's fitness for use, based on its ability to satisfy the user's needs.
- The **manufacturing view** sees quality as the product's conformance to specifications. This view examines quality before and after product release; and focuses on meeting specs and conformance to process standards to build the product "right the first time."
- The **value-based view** sees the level of quality as being dependent on the investment that the customer has in it. This outlook requires the consideration of the trade-offs between cost and quality.
- The **product view** examines the software product through its distinctive characteristics. It assumes that measuring and controlling internal product indicators will improve the product's behaviour.

The majority of those in software development and research tend to agree that the *user and manufacturing views* are of definite importance ⁱⁱ. Both of these prospects examine the product's external characteristics. The ideas of *fitness for use* and conformance to specifications perpetuated in these views imply the absence of failure in the product, or at least require that any problems be at a tolerable level.

Failures, and therefore depletion in quality, will often occur as a result of the nature of the software project. In a *project* ⁱⁱⁱ:

- There are **limited resources** (i.e. time, money and personnel).
- There is a **precise goal**.
- There are **partially ordered sequenced activities**.

Continued on next page

Metrics continued...

These constraints lend uncertainty to the development process. For example: in all projects, resources are finite (as outlined above); and in examining these limited resources, questions arise as to optimal allocation - *How should they be allotted in the entire development process in order to deliver the best possible product?* And in addition to this - *Exactly how much of each resource is needed to result in a specified level of performance?* The project manager must take all of the above factors into consideration, and pose similar questions, in order to complete a project on time, on budget, on performance, with a satisfied customer.

Therefore if projects are to succeed, there must be good project management; and, if it must be managed, one must have an idea of its scope, therefore it *must* be measured. This measurement is performed using some product quantification, some count of product characteristics, that is, the software quality metric.

These metrics can be subdivided into: Project, Process, and Product metrics. Project metrics are those which outline project characteristics, this involves the quantity and level of resources (for example: the number of programmers and peak staffing in specific life cycle phases). Process metrics are those which can be used to improve software development or maintenance methods. The Product metric is one that defines product characteristics, represented in such measures as size and complexity.

In attacking the problem of managing finite resources, *estimates* are formulated based on metrics (specifically project and product metrics) in order to *project* the resources needed. These projections, if accurate, will reduce failures due to insight into what to expect. Among the best estimation methods for software are *Empirical Methods*. Empirical methods analyze metric data sets from previous development experience in their entirety, in order to develop a system for prediction of many properties including defect behaviour. Some of the better empirical methods for software estimation are ^{iv}:

1. Halstead Metrics
2. McCabe's Cyclomatic Complexity
3. Function Points
4. Rayleigh Curves (of particular interest to those developing a defect-tracking program)
5. The COCOMO Model

Other useful estimation methods which may be utilized to verify results from empirical estimation are *Prototype Methods*, i.e. where a partial representative implementation is built, and an idea of the scale of resources required is obtained from this *prototype*; and *Statistical Methods*, i.e. where the system is broken into its composing functions, and estimates are made based on knowledge of previously developed similar functions.

The best method in estimating, and attempting to reach the goal of quality, would be to utilize many differing methods, to enable comparison and therefore recognize the most suitable metrics and models for your organization and development situation. The work that I have reviewed seems to strongly suggest, and in some cases explicitly states ^v, that it is wise to utilize, in any one software estimating situation, more than one of the methods that one's experience has proven, to implement a necessary checking system. Therefore the use of metrics, and the models that incorporate them into the process of software prediction, should be part of the software development project to achieve software quality assurance.

We will discuss useful software metrics for your software projects in more detail in future issues of *The Quality Mark*.

ⁱ Bache, Bazzana. *Software Metrics for Product Assessment*. 1994. McGraw-Hill.

ⁱⁱ Kitchenham, Pfleeger. "Software Quality: The Elusive Target." *IEEE Software*. 1996. IEEE Press.

ⁱⁱⁱ J. Michael Bennett. *Computer Science 333Y Lecture Notes*. 1995. University of Western Ontario.

^{iv} Stephen H. Kan. *Metrics and Models in Software Quality Engineering*. 1995. Addison Wesley Press.

^v Abdalla, Abdel-Ghaly, Chan, Littlewood. "Evaluation of Competing Software Reliability Predictions." *IEEE Transactions on Software Engineering*, Vol. 12 No. 9. 1986.

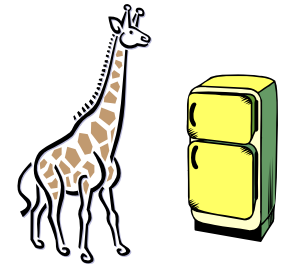


IQ Test for Professionals

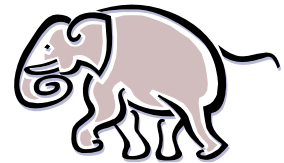
The following short quiz consists of 4 questions and tells whether you are qualified to be "professional". Answers are on the next page - No Cheating!

The questions are not that difficult.

1. How do you put a giraffe into a refrigerator?



2. How do you put an elephant into a refrigerator?



3. The Lion King is hosting an animal conference. All the animals attend except one. Which animal does not attend?



4. There is a river you must cross. But it is inhabited by crocodiles. How do you manage it?



IQ Test Answers

1. The correct answer is: Open the refrigerator, put in the giraffe and close the door.

This question tests whether you tend to do simple things in an overly complicated way.

2. Wrong Answer: Open the refrigerator, put in the elephant and close the refrigerator.

Correct Answer: Open the refrigerator, take out the giraffe, put in the elephant and close the door.

This tests your ability to think through the repercussions of your actions.

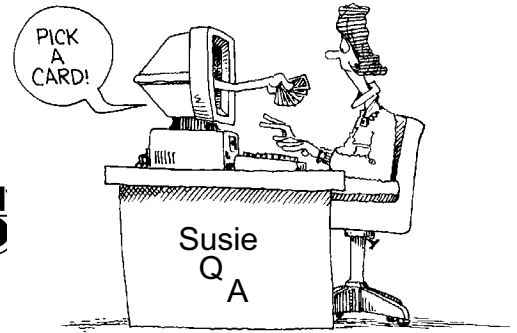
3. Correct Answer: The Elephant. The Elephant is in the refrigerator.

This tests your memory. OK, even if you did not answer the first three questions correctly, you still have one more chance to show your abilities.

4. Correct Answer: You swim across. All the Crocodiles are attending the Animal Meeting. This tests whether you learn quickly from your mistakes.

According to Andersen Consulting Worldwide, around 90% of the professionals they tested got all questions wrong.

TIPS & TRICKS



Thanks go out to all of our readers who submitted their tips for this issue! We decided to award the best submission for each issue with a prize. Five independent judges ranked each tip out of 5 and the highest score was awarded the prize. This issue's prize was awarded to Chris Mintz who won a Chapters gift certificate. Congratulations Chris! ☺

Please send your questions and/or useful tips to Susie QA via email to news@kwsqa.org.

Q: Are there any tools to help me when testing software Installs?

A: Submitted by Tom Cloutier

InCtrl is an installation-tracking tool I discovered while reading PC Magazine a few years ago. It has come in handy for troubleshooting & regression testing installations. The newest version has a wonderful HTML output option and the ability to track more files during an install.

The most common way a test department might use the tool is to capture an image of the system before installing a program or group of programs, and tell it when the installation is finished. It then compares the before and after situation for you and outputs a file to a format and location you choose. Another use I found was to find out what registry entries have changed while a particular task is performed in a program.

The best part is that it is **free** (*key* word for a small testing group) and the source code is included. I have included the link to the article below but it should be enough to search for InCtrl on the ZDNet.com website and download the latest version.

URL: <http://www.zdnet.com/downloads/stories/info/0,,001CV2,.html>

Q: How can I tell if my system is clean when testing software Installs?

A: Submitted by Chris Mintz

Avoid System File and Registered Component Pile-up on QA systems.

If your product installs registered DLLs, OCXs, or overwrites/updates any Microsoft DLLs (e.g.: msvcrt.dll) you may not be testing your product with the proper file configuration on your QA systems. If there are registered file components that you used to install but have recently removed dependency, they may still be on your test systems affecting the functionality of your product. QA systems in general tend to collect many "garbage" files like unfinished OCXs or DLLs due to the massive amounts of installing and uninstalling that gets performed on these systems on a regular basis.

Here's a tip: From a fresh O/S install use a tool called Norton Ghost. This will make an exact mirror of your drive that you can burn to a CD (alternatively, PowerQuest's Drive Image is another product for doing this). Once a week plan to have system "restores" performed where the systems are rolled back to a fresh and clean state using your burned drive images. An even faster solution is to make a boot diskette that will give you network support and host your drive "images" over the network!



Continued on next page

Before we used drive imaging, reinstalling an O/S could take up to 45 minutes. Using a drive imaging product it takes no more than 5 minutes to roll a system back to a clean state! You can also use this method to quickly swap O/S flavours. You could be running Win98 and 5 minutes later you can easily be running Win2K without the hassle of dual boot.

Q: Is there a way to enable Filename completion at a Command Prompt in NT?

A: Submitted by Paul Carvalho

I am used to using the [Tab] key to complete filenames at a Unix prompt but didn't know how to do it in Win NT until recently. Here's what you do to **Enable** this feature:

- Open the NT registry (run Regedit)
- Search for "completion" (no quotes)
- You should find a match at:
`\HKEY_CURRENT_USER\Software\Microsoft\Command Processor`
- Double-click on "CompletionChar"
- Change the Value data from 0 to 9 (Hex)
- Click [OK] and exit the Registry

You should now be able to use the [Tab] key at the Command Prompt to complete filenames.

To disable this feature - if you ever really want to do that - just reset the value back to 0.

Q: Are there any quick Windows keyboard shortcuts that I should know about?

A: Submitted by Jack van Ham

Here are some handy shortcuts (Windows keyboard required, [Win] = Windows key):

- [Win] + [E] = Brings up the Windows Explorer
- [Win] + [F] = Brings up the Find Dialog Box
- [Win] + [M] = Minimizes all windows
- [Win] + [R] = Brings up the Run Dialog Box

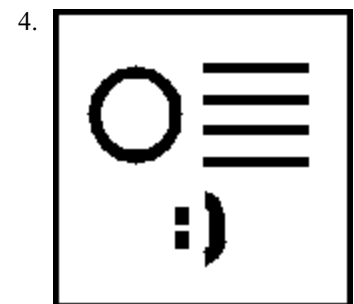
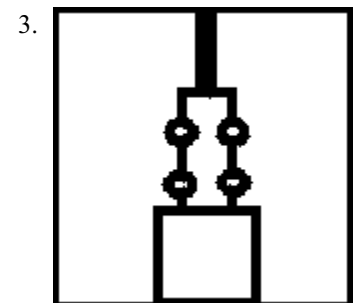
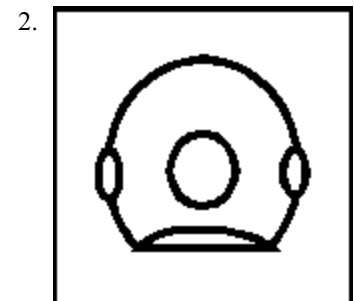
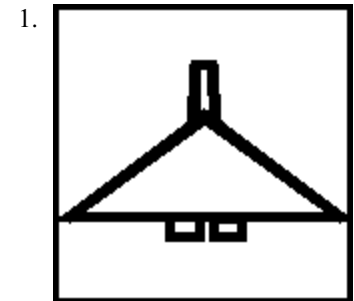


RESOURCES ON THE W³

Site	Strengths
StickyMinds.com: http://www.stickyminds.com/	From the creators of STQE (pronounced "sticky") Magazine, comes a web site for software managers, testers, and QA people to gather and share information.
Software QA / Test Resource Center: http://www.softwareqatest.com/	Useful FAQ's, lists of test tools, job boards, recommended book lists and more!
Human Factors International: http://www.humanfactors.com/home/	Advancing the science of "user friendliness" in applications, HFI offers training and development services, provides a free e-newsletter and other cool free stuff through their web site.
The Easter Egg Archive: http://www.eeggs.com/	A site devoted to helping you discover the amusing gems that creators hide in their creations. Includes: computers, movies, music, TV, books and art.

Doodles

Take your best shot at the following doodles. Suggested answers may be found elsewhere in this issue. Good Luck!



THE \$ALARY THEOREM

Engineers and scientists can never earn as much as business executives and sales people.

This theorem can now be supported by a mathematical equation based on the following two postulates: (which we all accept as true)

Postulate 1: Knowledge is Power.

Postulate 2: Time is Money.

As every engineer and scientist knows:

$$\text{Power} = \text{Work}/\text{Time}$$

Since:

$$\text{Knowledge} = \text{Power}$$

and

$$\text{Time} = \text{Money}$$

then:

$$\text{Knowledge} = \text{Work}/\text{Money}$$

Solving for Money, we get:

$$\text{Money} = \text{Work}/\text{Knowledge}$$

Thus, as Knowledge approaches zero, Money approaches infinity, regardless of the amount of work done.

Conclusion: The less you know, the more you make.

QUIZZES & CHALLENGES

☆ = Very Easy, ☆☆ = Easy, ☆☆☆ = Average, ☆☆☆☆ = Difficult

Solutions to these problems will appear in the next issue. Enjoy!



1. Time Travel ☆

Angela left on a trip the day after the day before yesterday and she will be back on the eve of the day after tomorrow. How many days is she away?

2. To Get to the Other Side ☆☆☆

If two of the following statements are false, what chance is there that the egg came first? Round to the nearest whole percent. Note: If any part of a statement is false, then the entire statement must be false.

- A. The chicken came first.
- B. The egg came first.
- C. A is false, and B is true.

3. A Quick Reading Test ☆☆☆

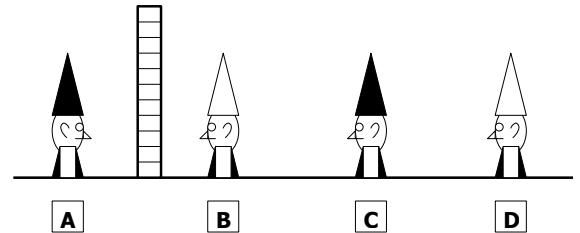
Read the sentence below and count the F's in that sentence.

Count them ONLY ONCE. Do not go back and count them again.

FINISHED FILES ARE THE RESULT OF YEARS OF SCIENTIFIC STUDY COMBINED WITH THE EXPERIENCE OF YEARS.

4. A Hat Trick ☆☆☆☆

Shown are 4 men buried up to their necks in the ground. They cannot move so can only look forward. Between A and B is a brick wall which cannot be seen through. They know that between them are 4 hats, 2 x black and 2 x white, but they do not know which colour they are wearing. In order to avoid being shot one of them must call out to the executioner the colour of his hat. If he gets it wrong, everyone will be shot. They are not allowed to talk to each other and have 10 minutes to figure it out.



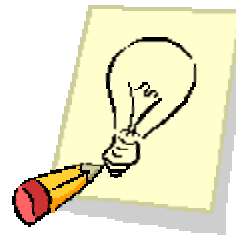
After 1 minute:

- Q1. Which one of them calls out?
- Q2. Why is he 100% certain of the colour of his hat?

This is not a trick question. There are no outside influences or other ways of communicating. They cannot move and are buried in a straight line. So A & B can only see their respective sides of the brick wall, C can see B, and D can see B & C.



Solutions to Last Issue's Challenges



1. What's Next?

"E" (the letters in the series are the initial letters of: one, two, three, four ... and so on).

2. Star Gazing

There are 49 triangles.

3. Looks Okay

There is not a single "e" in the paragraph.

4. Murder in a Small Software Company

The youngest member was not the victim, from [4]; was not the accessory, from [5]; and was not the guilty party, from [7]. So, from [5], there are only three possibilities (A = accessory, V = victim, G = guilty, and W = witness):

	<u>I</u>	<u>II</u>	<u>III</u>
oldest member	A	A	G
next-to-oldest member	V	G	A
next-to-youngest member	G	V	V
youngest member	W	W	W

From [6], the Sr. Developer was the oldest member; so from [1], the next-to-oldest member was the Sr. Marketing person. From [3] and the above possibilities, the youngest member was the Junior Developer; so the next-to-youngest member was the Tester. Then from oldest member to youngest member the three possibilities are:

	<u>I</u>	<u>II</u>	<u>III</u>
Senior Developer (M)	A	A	G
Senior Marketing (F)	V	G	A
Junior Tester (M)	G	V	V
Junior Developer (F)	W	W	W

From [4], I is impossible. From [2], III is impossible. So II becomes the only remaining possibility. Therefore, *the Senior Marketing person intentionally crashed the system.*

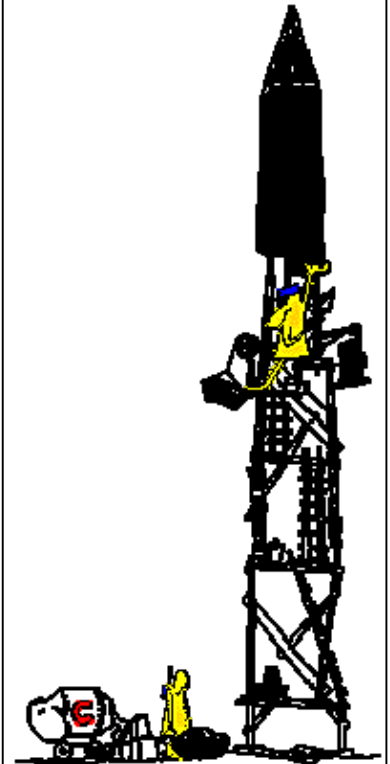
KWSQA Corner – Contact Information for 2000-2001

President Mechelle Gittens pres@kwsqa.org	Secretary Paul Carvalho info@kwsqa.org
Vice-President Marcela Prichici vp@kwsqa.org	Webmaster Jack van Ham webmaster@kwsqa.org
Treasurer Sarah Jones sejones@rim.net	Mailing List Moderator Paul Carvalho kwsqa-owner@yahoogroups.com

Subscribe to the KWSQA Mailing list for FREE! Check out:
<http://groups.yahoo.com/group/kwsqa/>



How do you build your software?



Drawn by Angus Macdonald

Droodle Answers

These are the suggested answers, but you can submit your own!

1. Top view of elephant hanging-sliding.
2. Top view of a Mexican holding on to his hat on a windy day.
3. King Kong hanging on to the Empire State Building.
4. Spare parts for a stick man.

About...

The Quality Mark

The Quality Mark is the official newsletter of the Kitchener-Waterloo Software Quality Association (KWSQA). It is published quarterly (in September, December, March, and June), and is distributed free to all local KWSQA members.

Challenge us to make a difference! *The Quality Mark* is about you and your interests. We are committed to making this newsletter a practical resource for you by responding to your suggestions and criticism. We invite you to give us feedback about what you find useful and what you don't. Send us your real-world ideas for future issues.

News@kwsqa.org

Editor: Paul Carvalho

Contributors:
Sherri Howell,
Terri Zuccherato,
Michael Schmidt




If you are interested in helping out with the next issue of *TQM* please drop us an email at the above address. No experience is required. We're looking for levity, imagination, and a desire to help create a useful, professional resource for the local high-tech community. Please let us know if you wish to contribute in any way.

Mark Your Calendar

For updates, visit: <http://www.kwsqa.org/>

The KWSQA Meetings occur on the last Wednesday of each month (except in December, July and August), and are usually held in the Communitech building on King Street in Waterloo, unless otherwise indicated via the Yahoo!Groups mailing list.



MARCH	6 - STC General Meeting	21 - Communitech Luncheon
	17 - <i>St. Patrick's Day</i>	27 - ASQ KW: Business side of APQP - TASSQ Dinner Meeting
	20 - <i>First day of Spring</i>	28 - KWSQA Meeting
APRIL	1 - Daylight Savings Time begins: * Advance your clocks ahead 1 hour * 	17 - Communitech Luncheon
	3 - STC General Meeting	18 - ASQ KW Dinner Meeting: Risk Assessment
	4 – 6 - QAI Seminar in Toronto 'Client / Server Testing'	22 - <i>Earth Day</i>
	7 - Elmira Maple Syrup Festival	24 - TASSQ Dinner Meeting
	15 - <i>Easter</i>	25 - KWSQA Meeting
MAY	1 - STC General Meeting	22 - ASQ KW Dinner Meeting
	13 - <i>Mother's Day</i>	22 – 26 - QAI Seminar in Ottawa 'Boot Camp for Software Testers'
	18 - Deadline for <i>TQM</i> Submissions	29 - TASSQ Dinner Meeting
	21 - <i>Victoria Day</i>	30 - KWSQA Meeting

For more information on the above events, check out their corresponding web sites:

- Local (KW) community events: <http://www.kw-visitor.on.ca/>
- Elmira Maple Syrup Festival: <http://www.elmiramaplesyrup.com/>
- Communitech seminars: <http://www.communitech.org/>
- Society for Technical Communication (STC): <http://www.stc.waterloo.on.ca/>
- ASQ Kitchener Section: <http://www.asqkitchener.com/>
- TASSQ events (in Toronto): <http://www.tassq.org/>
- QAI events: <http://www.qaicana.com/>

Know of any other groups whose events we should add to our calendar? Let us know!