OSBR.CA

The Open Source Business Resource

MAY 2008

MAY 2008

PUBLISHER:
The Open Source Business Resource is a monthly publication of the Talent First Network. Archives are available at the website: http://www.osbr.ca

EDITOR:
Dru Lavigne
dru@osbr.ca

ISSN:
1913-6102

ADVERTISING:
Rowland Few
rowland@osbr.ca

GRAPHICS:
Ryan May

In January 2008, Gartner released their "Top Predictions for IT Organizations and Users, 2008 and Beyond" (http://www.gartner.com/DisplayDocument?id=591123). Their analysis around open source included the key finding that it "has become impractical for mainstream IT organizations to avoid or ignore the influence of open source across a wide variety of industry market segments. Doing so will put organizations at a serious disadvantage against competitors that are leveraging mature, stable and well-supported open-source technologies for significant return-oninvestment and total-cost-of-ownership opportunities." Does this mean that the enterprise is finally ready for open source?

As Bernard Golden points out in the first article, it is impossible to answer that question when it is framed that way--there are just too many open source projects, each possessing varying degrees of maturity and usability. Golden further posits a key point that enterprises themselves vary from early adopters to pragmatists. Fortunately, resources are available for gauging the applicability of a specific open source product to meet the needs of a particular organization.

Several methodologies exist for assessing open source (http://en.wikipedia.org/wiki/Open_source_software_assessment_methodologies) and this issue provides insights into two of these. Golden discusses Navica's Open Source Maturity Model (OSMM) and Raphaël Semeteys from Atos Origin describes the Qualification and Selection of Open Source software (QSOS) methodology. Both methodologies emerged from the respective company's interactions with enterprises and each provides a frame of reference for assessing open source.

Bruno von Rotz from Optaros describes another resource, the Enterprise Open Source Directory. Originally released as a static catalogue containing insights gained from working with enterprise customers, the directory has evolved into a dynamic and collaborative reference for finding mature, enterprise-ready open source products.

This issue also features three conference reports: two from the Technology Innovation Management (TIM) Lecture Series and one from the Partnership Conference Series. Topics addressed in these reports include wireless sensor networks, security and privacy in a connected world, and surviving as an entrepreneur. The Recent Reports section includes CAOS Research's report on open source in the enterprise database market, a report from the fourth conference on open source systems regarding the total growth of open source, and Coverity's 2008 open source report detailing trends observed from their scans of open source projects.

As always, we look forward to your feedback.

Dru Lavigne

Editor-in-Chief

dru@osbr.ca

Dru Lavigne is a technical writer and IT consultant who has been active with open source communities since the mid-1990s. She writes regularly for O'Reilly and DNSStuff.com and is author of the books BSD Hacks and The Best of FreeBSD Basics.

"You have a broad selection of open source projects to choose from...It's not easy to get the equations right--how strong is the community or how does it fit with us."

Bud Tribble, vice president of software technology at Apple

One of the questions always asked about open source is whether it's ready for the enterprise. But framing the question in that fashion blurs the issue. With over 100,000 open source products available for download at the click of a mouse, there is no blanket answer comprehensive enough to describe the entire universe of open source products.

The real question facing an enterprise is whether, based upon its unique requirements, a specific open source product will satisfy its needs. Far from being a vaguely existential question, this question is extremely pragmatic, completely localized, and, as we shall see, wholly capable of being answered.

This article, extracted from chapter four of "Succeeding with Open Source" (http://www.navicasoft.com/pages/bkoverview.htm) presents the Open Source Maturity Model (OSMM). The OSMM is designed to enable organizations to evaluate open source products and understand whether a product can fulfill the organization's requirements.

OSMM Methodology

While many discussions of open source focus on software and its functionality, the OSMM recognizes that mainstream IT organizations have many requirements beyond a given product's code base: support, training, documentation, integration, and services. The OSMM evaluates a product along all these dimensions, assigns a maturity score to each product element, and generates a numeric score assessing the overall maturity of the product.

A number without a context is less than useful, so the OSMM comes with recommended minimum maturity scores. These minimum maturity scores offer guidance as to what level of maturity should be present for a product to be considered for three different types of use: experimentation, pilot/departmental, and production. Naturally, these minimum maturity scores are only recommendations and may be adjusted according to the specific needs and capabilities of the organization.

Why do we need the OSMM? Haven't plenty of organizations implemented open source successfully without the OSMM? That's true, but overlooks the changing nature of the open source user base. Future open source users will require more complete, more mature products for their use. The OSMM is targeted toward the new breed of open source user.

Early Adopters and Pragmatists

In Crossing the Chasm (http://en.wikipedia.org/wiki/Crossing_the_Chasm), Geoffrey Moore identified two main types of technology users: early adopters and pragmatists. Early adopters are comfortable using unfinished products, whereas pragmatists prefer to wait for the mature product. Up to now, open source software (OSS) has been the province of early adopters; today, however, pragmatists are seriously considering open source solutions.

Traditionally, technology vendors begin by selling to early adopters, who are satisfied with the rudimentary products startups deliver. When vendors decide to begin selling to pragmatists, they experience a rude awakening. Pragmatists expect a vendor to deliver a complete product bundle that includes service providers, robust support, thorough documentation, and so on.

The product requirements of early adopters and pragmatists are radically different--different enough that Moore characterizes the distance between them as a chasm. Most vendors fail to successfully leap across this chasm. Open source seems like it would not face this problem; after all, the creators of the product are not focused on selling to any type of customer--early adopter or pragmatist--because the product is free. Customers make their own decision about whether to use a product, and never need to interact with a sales representative. This aspect of open source products overlooks one important fact: Even though no vendor is involved, it doesn't mean that pragmatists renounce their requirements. In the absence of a vendor, pragmatists often look elsewhere to procure a mature product.

In Moore's book, he noted that these distinct types of customers require very different products. Early adopters will accept immature products offering a competitive advantage. They are willing to forego access to sophisticated support, do not insist on high-quality training and documentation, and will even accept a lower quality product to achieve advantage. Consequently, early adopters are willing to work with small technology suppliers who are engineering-centric, short-staffed, and whose employees are "different", as long as the company provides advanced products. While early adopters are easier to work with, they represent only about 15% of any market; enough to get started on, but not enough for a vendor to prosper.

Pragmatists, by contrast, demand mature products. Mature products must be high quality and fully functional, but these factors are just the opening ante for pragmatists. To be accepted by them, products also must be accompanied by elements that make them easy to use and efficient to run.

Mature products come with a training program, a sophisticated support operation, well-written documentation, and marketing materials that make it easy to compare the product to its competitors to understand how it fits into a customer's existing computing infrastructure. Pragmatists requirements start with a particular piece of software, but they expect it to be bundled with a number of other product elements. Only when this entire bundle is available will pragmatists feel comfortable implementing a product. It's much more work to sell to pragmatists, but they represent a very lucrative 85% of the market share.

If you're an ambitious vendor, you'll have to deliver what this portion of the market demands. The world of open source, however, turns this process upside down. If you examine the over 100,000 open source products, it is clear that there are few, if any, open source providers that deliver a bundled product at the level of maturity pragmatic organizations require. The vast majority of open source products are freely available for download, with the expectation that the user organization will create the bundled product itself.

This highlights the open source mature product dilemma: a technology provider that, because of the economics of open source, cannot deliver a mature product, and a pragmatic technology consumer that requires a mature product to begin implementation.

The Open Source World

In the open source world, product elements are delivered by independent entities, with very little control exerted by the development organization. Organizations wishing to assess the maturity of a product must identify how each of the elements will be procured and the level of maturity of each element.

This means, for example, that if an organization wishes to assess the maturity of an open source network monitoring product, it must identify where training can be found for the product and how good the training is.

Because of the nature of open source development, organizations selecting software cannot expect what they get when selecting commercial software: a sales rep to track down answers to every question, a sales engineer who will perform a demo and perhaps even prototype an application, and a support organization to answer questions after the product is installed and running. Determining the maturity of the product is something the organization will need to take on. Open source offers organizations much more control of their destiny; it also imposes much more responsibility for their product choices.

One way to look at this is to depict the process differently: Rather than procurement from a single provider, it is more akin to creating a coalition of providers to deliver the finished mature product. In the world of open source, selecting software is less like going to a Wal-Mart and more like being a construction general contractor. General contractors draw together independent entities like carpenters, plumbers, electricians, tile setters, and a large number of other contributors. Each member of the project performs his or her task under the guidance of the general contractor, who is responsible for selection and assessment of the people and for the quality of the overall product.

The task for open source users is to identify the necessary product elements, assess their maturity, and determine whether the complete product meets the necessary maturity level for the intended use.

The challenge is to use a consistent assessment mechanism that ensures nothing is skipped and provides a formal set of assessment criteria.

The OSMM

The vast majority of the open source products available are probably not useful for an IT (information technology) organization. If even 1/10 of 1 percent of them are potential candidates for use, that represents a pool of more than 100 products that must be assessed for their maturity for a particular organization.

Without a formal methodology that implements a standardized analytical framework, organizations are limited in their ability to assess the maturity of a product. A framework also helps to identify the elements of a product that require improvement. Of course, lacking a way to formally assess products, organizations cannot compare open source products to determine which it should use. It is to address this challenge that Navica (http://www.navicasoft.com) developed the OSMM. The OSMM assesses a product's maturity in three phases:

1. Assess vital product elements for maturity and assign a maturity score.

2. Define a weighting for each element based on the organization's requirements.

3. Calculate the product's overall maturity score.

Phase 1: Assess Element Maturity

The first phase identifies key product elements and assesses the maturity level of each element. Key elements are those that are critical to implementing a product successfully:

- product software
- support
- documentation
- training
- product integrations
- professional services

Each element is assessed and assigned a score via a four-step process:

Step one: define requirements. The purpose of this step is to define the organization's requirements for a particular element. For example, if an organization wants to implement an open source web content cache, it must determine what functionality it requires in the software based on the organization's purpose: Is it attempting to reduce bandwidth load or response time, or does it have another purpose? As another example, if an organization is implementing an open source J2EE application server, its training requirements will be vastly different if it already has significant experience with a commercial application server than if it is beginning to use one for the first time. Defining the requirements for an element is a key step in assessing the usefulness of a product for a particular organization.

Step two: locate resources. Due to the loose coupling of product resources, locating resources for open source products is more complex than it is for comparable commercial products. There probably won't be an "approved partner" list for most products. Locating the resources for an element is more challenging, but there are a number of identification methods that can assist an organization in implementing OSS. As an example, product forums can be searched to locate a service provider that can supplement an organization's own personnel resources.

Step three: assess maturity. This is the key activity in determining the usefulness of a product element. Determining where the element lies on the maturity continuum--from nonexistent to production-ready--lets an organization determine how likely the product will satisfy its requirements.

Step four: assign maturity score. After the maturity assessment is complete, a maturity score between 0 and 10 is assigned to document how well the product element meets the organization's requirements. The score serves as a concrete output of step three: It documents the consensus of the organization. Assigning a score also compels the organization to crystallize its judgment.

Element scores are also helpful when comparing different products. It's easy to compare, say, the training maturity for two different open source content management systems in light of the organization's needs. This can become a decision tool for selecting one product or another based on the specific requirements of the organization.

Finally, the maturity score serves as an input into improving the element's maturity. If a product's overall maturity score is satisfactory, but one element's maturity score is low, the organization can choose to take steps to improve that element's maturity.

Phase 2: Assign Weighting Factors

The OSMM assigns a weighting to each element's maturity score, allowing each element to reflect its importance to the overall maturity of the product. For example, the heaviest weighting is typically assigned to the product software, whereas other elements have lower weighting factors to reflect the fact that they are less critical than the software itself in determining overall product maturity.

The default weightings for the elements are shown in Table 1.

Table 1: Default OSMM Element Weightings

Software	4
Support	2
Documentation	1
Training	1
Integration	1
Professional services	1
Total:	10

The weighted score of each element is summed to provide an overall maturity score for the product.

Organizations might choose to adjust the default weighting factors based on their specific needs. For example, if an IT organization is stretched very thin in terms of personnel, it might plan to have an open source product implemented by a professional services firm. In that case, it might increase the weighting factor for professional services to 2 or even 3 to reflect the relative importance of professional services.

This allows the OSMM the flexibility to apply to every organization's situation. A product's maturity score will reflect the organization's specific needs and resources. The only requirement for adjusting the maturity weighting is that the element scores must sum to 10, since the final step of the OSMM is to create an overall maturity score that is normalized to a 100 point scale.

Phase 3: Calculate Overall Maturity Score

After each element has been assessed and assigned a weighting factor, the overall product maturity score is calculated.

The element scores are summed to give an overall product maturity score on a scale of 1 to 100, where the highest possible maturity score is 100.

A blank template is downloadable from http://www.navicasoft.com/pages/osmm.htm. This site also provides blank worksheet templates that organizations can use as they work through assessing product elements.

Recommended OSMM Scores

Calculating a score and using it for a decision leaves out one of the most important factors in any decision: its purpose. A maturity score in an abstract consideration is meaningless; what is critical is the maturity score a product needs for a particular use.

The recommended minimum scores vary according to whether an organization considers itself an early adopter or a pragmatist. Pragmatic organizations are less willing to take risks with software products and therefore require higher maturity scores. In other words, there is an inverse relationship between risk tolerance and required maturity score. Depending on whether your organization is an early adopter or a pragmatist, you should adjust your minimum maturity scores to reflect your risk tolerance.

It must be emphasized, of course, that the recommended minimum scores are just that: recommendations. You might choose to use a product even though it fails to achieve the recommended minimum score for your purpose. In fact, you might decide that your organization would like to use different values for the minimum scores. The purpose of the recommendations is to provide a good starting point for determining how mature a product needs to be for a given purpose. If you feel a different value makes more sense for you, that's perfectly fine.

It's more important that you perform a maturity assessment and determine what your minimum acceptable score is than to rigidly adhere to recommendations that might not reflect your needs.

Conclusion

Many people have observed that OSS is a disruptive technology. It's radically different modes of software creation and distribution promise to shake up the IT industry and cause a massive shift of power from vendors to users. Less often observed is the fact that disruptive technologies also shake up assumptions and working practices entirely appropriate to the previous environment but unworkable in the new one.

The comfortable assumptions about the roles of vendor and user that underpinned the commercial software world must be superseded by a recognition that, in the open source world, the shift of power to users is accompanied by a shift of responsibility. In the future, users will be responsible for creating the mature product bundle required for pragmatic organizations to use a technology.

The OSMM was developed to assist in that bundle creation effort, offering organizations the ability to assess the maturity level of open source products. The OSMM can be a powerful part of your open source toolkit. The next time someone in your organization questions whether a particular open source product is "production-ready", consider using the OSMM to answer the question definitively.

The whitepaper upon which this article is based, as well as the OSMM templates mentioned in the article are available for download from the Navica website (http://www.navicasoft.com/pages/osmm.htm).

Bernard Golden is CEO of Navica, a Silicon Valley system integrator specializing in open source solutions. He previously served as a venture partner for an international venture fund and has been vice president and general manager in a number of private and public software companies, including Informix, Uniplex Software, and Deploy Solutions. He is a frequent speaker on information technology topics.

"A manager may be more interested in the overall quality rather than in a specific quality characteristic, and for this reason will need to assign weights, reflecting business requirements, to the individual characteristics."

ISO 9126
(http://en.wikipedia.org/wiki/ISO_9126)

For a company, the choice to opt for software as a component of its information system, whether this software is open source or commercial, rests on the analysis of needs and constraints and on the adequacy of the software to address these needs and constraints.

However, when one plans to study the adequacy of open source software (OSS), it is necessary to have a method of qualification and selection adapted to the characteristics of this type of software and to precisely examine the constraints and risks specific to OSS. Since the open source field has a very broad scope, it is also necessary to use a qualification method that differentiates between numerous candidates to meet technical, functional and strategic requirements.

This document describes the QSOS (Qualification and Selection of software Open Source) method, conceived by the technology services company Atos Origin SA (http://www.atosorigin.com/) to qualify, select and compare OSS in an objective, traceable and argued way. The method can be integrated within a more general process of technological watch which is not presented here. It describes a process to set up identity cards and evaluation sheets for OSS.

Why a Methodology?

When evaluating software, the following questions naturally arise:

- which software best meets the actual or planned technical requirements?
- which software best meets the actual or planned functional requirements?

In addition, every company should answer these questions before making any decision:

- what is the durability of the software and what are the risks of forks and how do we anticipate and manage them?
- what level of stability can be expected and how will we manage dysfunctions?
- what is the expected and available support level provided on the software?
- is it possible to influence further development of the software with the addition of new or specific functionalities?

To answer these questions and set up an efficient risk management process, it is imperative to have a method allowing:

- software qualification by integrating the open source characteristics
- software comparisons according to formalized needs requirements of weighted criteria, in order to make a final choice

Why a Free Methodology?

We believe that the method as well as the results it generates must be made available to all under the terms of a free license. A free license is capable of ensuring the promotion of the open source movement as it provides:

- the ability for all to re-use available works for qualification and evaluation
- the quality and objectivity of documents generated, perfected according to principles of transparency and peer reviews

For these reasons, we decided to make the QSOS method, and the documents generated during its application (functional grids, identity cards and evaluation sheets), available under the terms of the GNU Free Documentation License (http://www.gnu.org/copyleft/fdl.html).

General Process

The general process of QSOS is made up of four interdependent steps:

1. Definition: creation of frames of reference used in the following steps.

2. Evaluation: made on three axes of criteria: i) functional coverage; ii) risks for the user; and iii) risks for the service provider independent of any particular user or customer context.

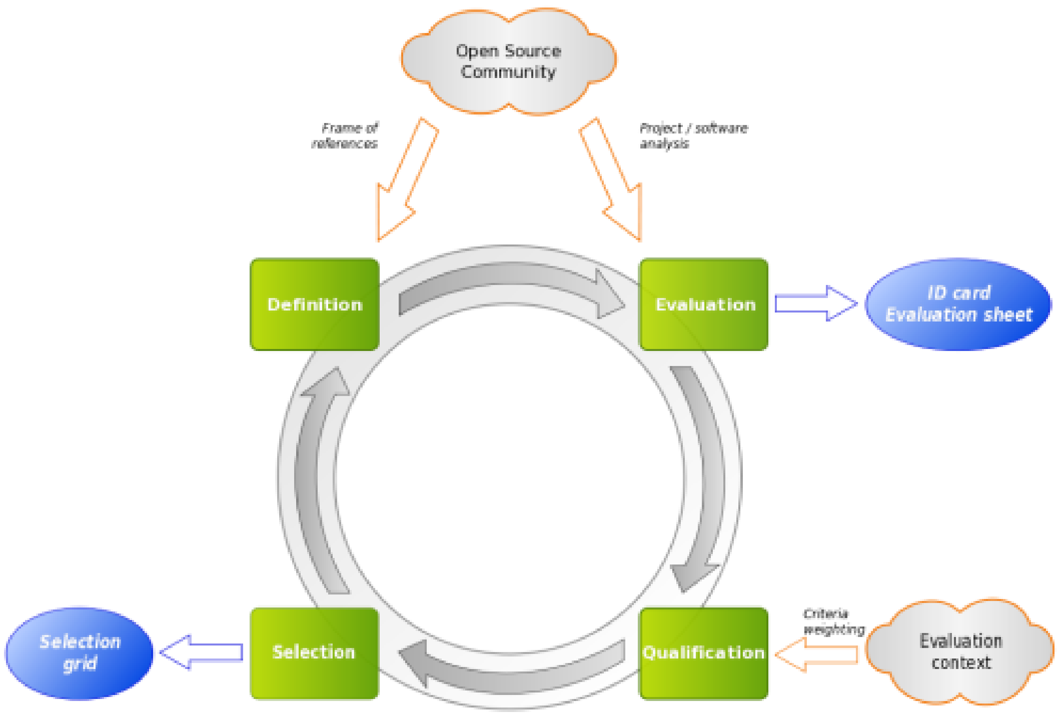
3. Qualification: weighting of the criteria split up on the three axes and modeling the context, user requirements, and/or strategy set by the service provider.

4. Selection: process the data provided in steps one and two through the filter set up in step three in order to proceed to queries, comparisons, and selections of products.

Figure 1 provides a visualization of the four step QSOS process. Each one of these steps is detailed further in this document.

The general process introduced here can be applied with different granularities. It enables the establishment of the desired level of detail for the process as well as advancement of the process by iterative loops to refine each of the four steps.

Figure 1: The Four Steps of QSOS



Tools developed by Atos Origin to apply the QSOS method in a coherent way are available to the community to coordinate creation, modification and use of QSOS evaluations (http://www.qsos.org/?page_id=5).

Step 1: Definition

The objective of this step is to define various elements of the typology to be reused by the three remaining steps of the general process. The frames of reference are:

Software families: hierarchical classification of software domains and description of functional grids associated with each domain. This frame of reference evolves the most because as software evolves, it offers new functionalities that need to be added to the frame of reference.

Types of licenses: this frame of reference lists and classifies the major licenses used for OSS. The criteria chosen to describe such a license are: i) ownership (can the derived code become proprietary or must it remain free?); ii) virality (is another module linked to the source code affected by the same license?); and iii) inheritance (does the derived code inherit from the license or is it possible to apply additional restrictions?). Note that a piece of software or code can be published under the terms of several licenses, including closed source licenses.

Types of communities: classification of community organizations existing around OSS and in charge of its life-cycle. The types of communities identified to date are: i) an insulated developer where the software is developed and managed by one person; ii) a group of developers where several people collaborate in an informal or not industrialized way; iii) an organization of developers where a group of developers manage the software life-cycle in a formalized way, generally based on role assignment and meritocracy; iv) a legal entity that manages the community, generally possesses copyrights, and manages sponsorship and linked subsidies; and v) a commercial entity employing the project's main developers who are remunerated by the sale of services or of commercial versions of the software.

The O3S tool is designed to be able to easily manage these frames of reference and to measure impacts generated by modifications on data already collected during other QSOS steps.

Step 2: Evaluation

The objective of this step is to carry out the evaluation of the software. It consists of collecting information from the open source community, in order to:

- build the identity (ID) card of the software
- build the evaluation sheet of the software, by scoring criteria split on three major axes: i) functional coverage; ii) risks from the user's perspective; and iii) risks from the service provider's perspective

Data constituting the identity card is raw and factual and is not directly scored. However, it is used as a basis for the scoring process described below. The main parts of an identity card are:

General information: this includes the: i) name of the software; ii) reference, date of creation, and date of release of the ID card; iii) author; iv) type of software; v) brief description of the software; vi) licenses to which the software is subjected; vii) project's webpage and demonstration site; viii) compatible operating systems; and ix) fork's origin, if the software is a fork.

Existing services: this component includes: i) documentation; ii) number of contractual support offers; iii) number of training offers; and iv) number of consultancy offers.

Functional and technical aspects: include the: i) technologies of implementation; ii) technical prerequisites; iii) detailed functionalities; and iv) roadmap.

Synthesis: includes the general trend and any comments.

Every software release is described in an evaluation sheet. This document includes more detailed information than the identity card as it focuses on identifying, describing and analyzing in detail each evolution brought by the new release.

Criteria are scored from 0 to 2. These scores will be used in step four to compare and select software according to the weightings, representing the user's requirements specified in step three. The following describe the criteria used for each axis of evaluation. Note that the same or similar criteria can appear on a different axis.

The functional grid is determined by the software's family and proceeds from the frame of reference of step one. Consult the QSOS website for details of functional grids by software families. For each element of the grid, the scoring rule is as follows:

Functionality	**Score**
Not Covered	0
Partially Covered	1
Completely Covered	2

In certain cases it is necessary to use several functional grids for the same software; for instance, when it belongs to more than one software family. In this case, the functional criteria are distributed on separated axes in order to be able to distinctly evaluate the functional coverage for each family.

The “risks from the user's perspective” axis of evaluation includes criteria to estimate risks incurred by the user when adopting OSS. Scoring of criteria is done independently of any particular user's context as the context is considered later in step three. Criteria are split into five categories:

• intrinsic durability

• industrialized solution

• integration

• technical adaptability

• strategy

Tables detailing each of these categories as well as their subcategories, by specifying the rule of notation to be used for each criterion, are available at
http://www.qsos.org/methode.php
#SECTION00083000000000000000.

The “risks from the service provider's perspective” axis of evaluation regroups criteria to estimate risks incurred by a contractor offering services around OSS such as expertise, integration, development, and support. It is notably on this basis that the level of commitment can be determined.

It is possible to iterate the QSOS process. At the evaluation step this brings the capacity to score criteria in three passes with different levels of granularity:

• first the five main categories

• then the subcategories of each category

• finally every remaining criterion

The general process is thus not hindered if not all of the scored criteria are available. Once all criteria have been scored, the score of the first two levels is calculated by the weighted average of scores of the directly inferior level.

The O3S tool allows the entry of raw data and the evaluation of software on the three major axes, as well as generation of the identity cards of evaluated software.

The granularity of evaluation is managed as follows: as long as all criteria composing a subcategory are not scored, its score is not calculated but entered by the user. As soon as all criteria are scored, its score is then automatically calculated.

Step 3: Qualification

The objective of this step is to define filters translating the needs and constraints related to the selection of OSS. This is achieved by qualifying the user's context which will be used later in step four.

A first level of filtering can be defined on data from the software's ID card. For instance, one could consider software only from a given family or software that's compatible with a given operating system. In general, although it is not mandatory, this filter does not include any weighting. It is mostly used to eliminate inadequate software in the specific context of the user.

Each functionality is attributed a requirement level selected among the following: i) required functionality; ii) optional functionality; and iii) not required functionality.

These requirement levels will be linked to weighting values at step four, according to the selected mode of selection.

The relevance of each criterion of the “user's risks” axis is positioned according to user's context as one of three criterion: i) irrelevant and therefore excluded from the filter; ii) relevant; and iii) critical. This relevance will be converted into a numerical weighting value at the following step, according to the chosen mode of selection.

The “filter on service provider's risks” is used by a service provider to evaluate software and services to be integrated into its offering and to determine the associated levels of commitment. The O3S tool allows the definition of these different filters.

Step 4: Selection

The objective of this step is to identify software fulfilling user's requirements or, more generally, to compare software from the same family. Two selection modes are possible:

Strict selection: based on direct elimination as soon as software does not fulfill the requirements formulated in step three. Reasons for immediate elimination include: i) incompatibility with the filter on the ID card; ii) not providing functionality required by the filter on the functional grid; and iii) scores on the "user's risks" axis do not meet the relevance defined by the user, as the score of a relevant criterion must be at least equal to 1 and the score of a critical criterion must be at least equal to 2. This method is very selective and may, depending on the user's requirement, return no eligible software. Selected software is attributed a total score, calculated by weighting.

Loose selection: this method is less strict as rather than eliminating non-eligible software, it classifies while measuring gaps with applied filters.

The weighting value for both selection methods is based on the level of requirement defined on each functionality of the functional grid as follows:

Level of Requirement	**Weight**
Required Functionality	+3
Optional Functionality	+1
Not Required Functionality	0

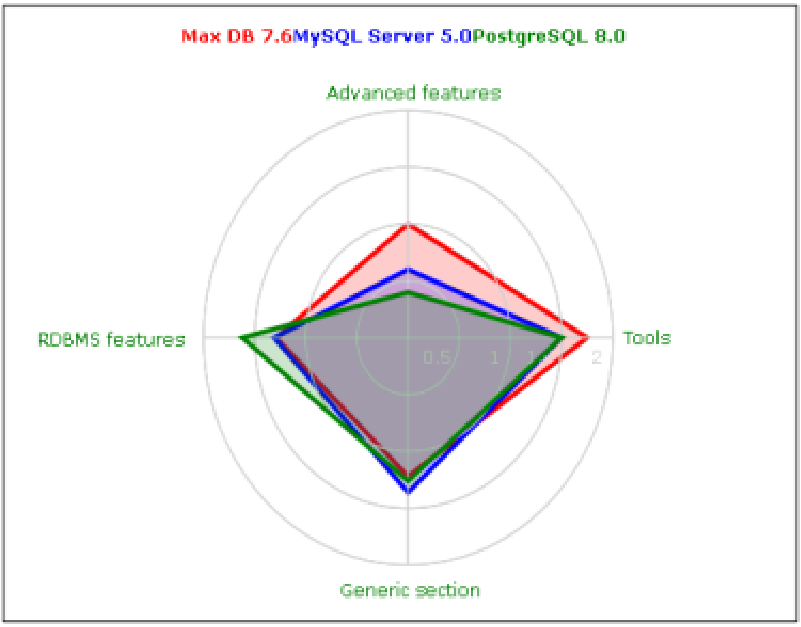
The weighting value on the "user's risk" axis is based on the relevance of each criterion as follows:

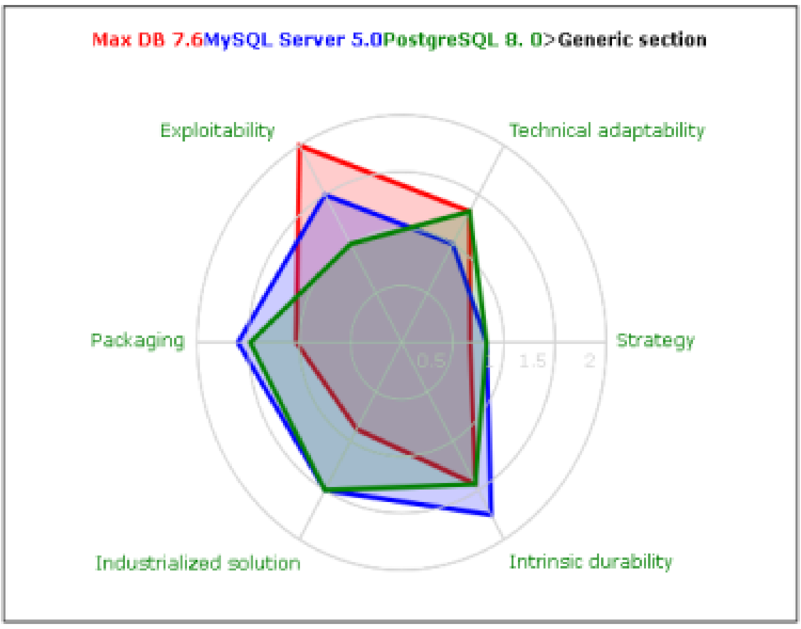
Relevance	**Weight**
Irrelevant Criterion	0
Relevant Criterion	+1 or -1
Critical Criterion	+3 or -3

The weight's value sign represents a positive or negative impact relating to the user's requirements.

The software of a same family with a common functional grid can also be compared by using weighted scores determined earlier. Figure 2 is provided as an example showing that weightings on the various axes are not representative of all kinds of relational database management systems (RDMBS, http://en.wikipedia.org/wiki/RDBMS) utilizations.

Figure 2: Comparison of RDMBS on QSOS Axes





Besides implementing the strict and loose selection modes, the O3S tool also enables the consultation of data related to a specific software (ID card and evaluation criteria) and the comparison (integrally, by filtering or differentially) of software in the same family.

Conclusion

The vast amount of available OSS software requires a methodology to allow for the evaluation of potential candidates to meet business requirements. The QSOS methodology allows for an iterative needs analysis for gauging the technical, functional, and strategic capabilities of OSS products. The QSOS website centralizes documents and information on the methodology and the creation, modification, and certification of functional grids, ID cards, and evaluation sheets.

Raphaël Semeteys is in charge of consulting activities for Atos Origin's French Open Source Skill Center. He produces and manages feasibility studies and technological watch reports on open source and free software. He created the QSOS method and is leader of the associated free project of community technological watch which documents, equips and organizes the collaborative evaluation work.

"The difference between the successful open source implementation, in which the value of open source is realized for a company, and the unsuccessful one, in which the struggle to use open source is not worth the effort, amounts to knowing your problem, knowing the software, and knowing yourself."

Open Source for the Enterprise (http://www.oreilly.com/catalog/9780596101190/)

Of all the choices available when selecting open source software (OSS), which ones are likely to meet business and technology requirements? What tools, if any, exist to help companies assess the enterprise readiness of a proposed open source solution? This article introduces the Enterprise Open Source (EOS) Directory, a resource which was designed to help corporations accustomed to evaluating commercial closed source software find enterprise-ready open source solutions.

Evaluating Open Source Software

OSS continues to gain momentum worldwide due to its low entry barrier, high quality and customizability. More information technology (IT) decision makers are favouring OSS over traditional packaged software as it becomes more aligned with organizational needs. OSS is now part of the IT mainstream, supporting many of the world's largest companies and government institutions. The role of OSS continues to expand, from deep within the infrastructure to the key applications that drive a business.

Despite its increased adoption, the perception of OSS as being "enterprise-grade" continues to be called into question. One reason is that the evaluation and selection of OSS is significantly different from the traditional approaches that have been used in the enterprise for decades.

Proprietary software vendors have sales and marketing teams to inform their customer base and provide detailed responses to RFIs (Requests for Information), RFPs (Requests for Proposal) or RFQs (Requests for Quotation). To simplify the search, there are usually a few obvious market leaders or customers can choose from a short list identified by independent analysts. Unlike proprietary software, there are hundreds of thousands of open source projects, and the software provided by these projects is often designed to address a specialized need.

Projects are usually run by individuals or small, unknown companies without the capacity or local presence to engage in traditional one-on-one sales and marketing relationships. This is one reason why costs are so low. But it also puts an added burden on the customer, who is responsible for exploring a sea of choices to identify a likely candidate.

Current Environment Supports Developers

There are currently more than 200,000 open source projects, making it difficult to identify those which are appropriate for enterprise use from the multitude of others that range from untested concepts to varying degrees of usability and maturity. It is time-consuming for most corporate IT departments to navigate through the available options as open source addresses a wide range of needs and originates from may different sources.

For a fast-paced enterprise, finding the right software in such a large and diverse marketplace is problematic. While OSS can be readily downloaded and analyzed, few companies have the resources or interest to conduct in-depth evaluations or to scan a large technology landscape.

Online communities and repositories often employ ratings that are based on popularity such as the number of downloads or amount of activity. This criterion may be of interest to developers, but it does not indicate that a product is enterprise-ready.

Support for Corporate IT

In January 2007, Optaros released its Open Source Catalog (http://tinyurl.com/56e32q) containing reviews of 262 projects. Following overwhelming global interest, the Optaros EOS Directory (http://www.eosdirectory.com) was created.

The EOS Directory provides a constantly updated list of the most relevant enterprise-ready open source offerings. The free online directory includes platforms, components, frameworks and solutions which have been evaluated and pre-qualified by Optaros as a neutral, expert third party. Corporate IT staff can receive advice, learn what others are doing, and interact with the open source community. The EOS Directory includes the community-building and knowledge-sharing features of Web 2.0 to meet the needs of both business enterprises and developers.

The EOS Directory does not contain all open source projects. Rather, it lists only those projects that Optaros believes are worth serious consideration for enterprise deployment – in short, quality over quantity. Optaros ratings are based on the company's worldwide consulting and integration experience, substantial research and evaluations, as well as interaction with open source communities and companies. Products are rated using criteria relevant to corporate IT departments such as functional richness, maturity, and competitive trends. The directory also includes independent ratings provided by software users to provide additional perspectives.

The directory reduces the time and cost of researching open source options, and provides an incentive for developers to deliver superior products. Subsequently, it provides an essential reference for enterprise customers, as well as valuable visibility for developers. The EOS Directory covers the major software domains: i) infrastructure such as operating systems and systems management tools; ii) application development such as programming languages, database technologies, and integration technologies; iii) infrastructure solutions such as enterprise content management and business process management; and iv) business applications such as customer relationship management and office applications.

More than 300 technologies, solutions and platforms are listed. For each project, the reader can find a short description, the license model, support models, four ratings representing enterprise readiness, functionality, maturity and community, as well as a trend indicator and a link to the project page. Using this information, it is easy to come up with a short list of technologies to be investigated further.

EOS Directory Criteria

In the EOS Directory, enterprise readiness is determined based on the rating provided by four indicators: i) the functionality; ii) the community; iii) the maturity; and iv) the perceived trend. The individual ratings are seen as an indication and not as absolute decision criteria. These four indicators can be summarized as follows:

1. Functionality: in most situations, a product's functionality is driven by what commercial products have to offer. The rating of the software can range from covering what is required by a typical mid-size or large enterprise, to having large gaps in functionality but providing a good basis for further development.

2. Community: for the long term success of an open source project, it is important that there is an active and well-supported community behind the project. In commercial open source products, this community is often the software development unit of the company behind the product. The input and contributions of external people is less important and influential. This can be seen as a risk, especially when a company is small or has limited funds available.

3. Maturity: to put a software product in production, it needs to be able to run in a stable and error-free manner. Maturity measures the quality and robustness of a software product. The rating of the software ranges from being a strong, high quality solution that is stable and meets advanced performance expectations, to a poor solution, only usable for test and demonstration purposes.

4. Trend: open source projects and products develop quickly and dynamically. It is important to understand whether a product is becoming more feature-rich and robust, whether there is no improvement, or whether the quality and richness is decreasing compared to the competition. The trend category indicates the expected future progress of the software and whether or not the solution is progressing along most of the criteria and growing in importance overall.

Based on the above ratings, the “enterprise readiness” is determined. The Optaros rating for enterprise readiness indicates how capable an open source product is to cope with the needs and requirements of midsize and large enterprises and organizations. Optaros rates products using one to four stars and a product without a star would mean it cannot be recommended for enterprises and therefore is not part of the directory.

Of course, there might be many additional open source projects out there that would deserve one or more stars but still have not been added to the directory. Optaros balances the directory by including primarily the products that serve a broad range of situations and find significant adoption. To continuously extend the directory, users can propose new projects to be added to the repository.

Better Choices, More Informed Buyers

While many open source products and projects do not measure up to the EOS Directory standards, they can still be used in certain situations. Not all enterprise-ready products and platforms in the open source ecosystem are listed in the directory but it remains a subjective selection aimed at helping enterprise decision makers identify OSS that best meets their requirements.

The directory has become one of the key references and information sources in the evolving OSS landscape as products change and improve based on their received ratings. The most promising trend is the substantial increase in higher ratings. Companies considering open source alternatives to existing software or planning a new project are likely to find the online directory a valuable resource.

The statistics of the site show a strong interest of enterprises in open source infrastructure and business applications. The two most popular sub-domains on the EOS Directory are "Enterprise Content Management" and "CRM, ERP and eCommerce", capturing 27% of all the research requests. Much interest is also seen in projects in subdomains such as collaboration, web servers and systems management.

Planned Enhancements

The EOS Directory is developed following the “perpetual beta” principle. Naturally, the EOS Directory has been implemented based on OSS, following the Optaros Assembly Method (OptAM, http://www.optaros.com/assembly-methodology). Key components include PHP, symfony, MySQL, Wordpress and phpBB. The functionality and the user interface are continuously improved and extended.

The next major release will include tagging for people to categorize open source projects using individual terms. OpenID (http://en.wikipedia.org/wiki/Openid) will allow people to easily sign up and web services will allow other sites to access and display the content in the EOS Directory. Lastly, to perpetuate the instructional nature of the directory, the open source educational content will be continuously extended and updated.

Bruno von Rotz is the country manager for Switzerland at Optaros. He has more than 20 years of IT consulting and system integration experience. Prior to Optaros, he was the Consulting Practice Lead for Enterprise Integration Solutions in EMEA for Novell and Cambridge Technology Partners. Prior to Novell, he worked for McKinsey & Company, where he focused on IT strategy and architecture. Bruno graduated from the Federal Institute of Technology (ETH) in Zurich with specialization in Information Systems.

Wireless Sensor Networks: What and Why?

"Revolutionary networking concepts and an unprecedented mix of technical challenges have made Wireless Sensor Networks (WSN) one of the major research trends of the 21st century. However...despite years of research and development and technical maturity, WSN products and solutions are yet neither fully adopted nor widely deployed."

Laurent Chalard et al
(http://tinyurl.com/5g4rx4)

On April 30, 2008, Thomas Kunz, Director of the Technology Innovation Management (TIM) program at Carleton University delivered a presentation entitled "Wireless Sensor Networks: What and Why?". The slides from the presentation are available from (http://www.talentfirstnetwork.org/wiki/images/7/73/Wireless_sensor_networks_April_30.pdf).

The TIM Lecture Series provides a forum to promote the transfer of knowledge from university research to technology company executives and entrepreneurs as well as research and development (R&D) personnel. This conference report presents the key messages and insights from the three sections discussed during Professor Kunz's presentation.

Introduction to Wireless Sensor Networks

The first section served as an introduction to wireless sensor networks (WSN, http://en.wikipedia.org/wiki/Wireless_sensor_networks). On the hardware side, sensors range in price and functionality from cheap and unreliable to expensive and mission critical. There are still many engineering challenges yet to be resolved. These challenges include: i) providing sustainable power and overcoming distance limitations in sensors; ii) creating global standards for radiation and privacy; iii) providing sufficient address space for nodes; iv) determining the environmental implications of discarded sensors; and v) studying long term health effects. Designing efficient networks is also challenging as the design must provide redundancy for nodes that fail and sensors require strategic placement in order to provide effective data collection.

Applications for WSN are many and varied and the application possibilities seem limitless. Particularly attractive application areas are smart homes and real-time traffic information. WSN is an emerging technology, providing many business opportunities to engineers who can solve the technical challenges and entrepreneurs who can capitalize on the new markets.

WSN technology raises many interesting dilemmas. Sensors now have the ability to collaborate as peers instead of merely uploading their data to a central server; an example of this is ZebraNet (http://www.princeton.edu/~mrm/zebranet.html). In WSN where humans are being monitored, such as in medical scenarios and smart homes, privacy of the data collected is critical. Also, will the fact that people know that they are being monitored result in a change in their behaviour?

Distributed sensors that are not under one entity's control are an important trend which raises the question of "who owns the data?" Actuators, mechanisms which introduce motion or which clamp an object so as to prevent motion, have alarming privacy implications.

WSN Applications and Challenges

The second section of the lecture concentrated on the challenges in designing WSN applications. Key insights from this section include:

- when designing hardware, processing is cheap but transmission of the data is expensive
- programming sensor networks is different than programming for the Internet
- database queries for WSN data also differ in their logic, though SQL does provide a well-known abstraction to developers of applications
- sensor data eventually ends up in a data warehouse for scientists to use
- application designers should be aware that currently the monetary value is in the hardware, not the software
- software design is based on current hardware constraints and we may be overcompensating for these constraints; conversely, conservative software design allows hardware to be made smaller

It was noted that some of the technologies developed for WSN will find their way into the Internet. An example provided by the audience was cloud computing as propagated by Google, which seems very similar to the data aggregation techniques discussed in the WSN community.

WSN Networking and Local Research

The final section concentrated on WSN networking. While most networks have been standardized for many years, WSN still offers many design opportunities. Network protocol stack designs are difficult but provide research opportunities. The traditional layered protocol architecture has proven successful in the Internet, but has overhead and redundancy challenges, making it less appealing for constrained wireless/embedded devices. Moreover, much more research is needed for transducers and actuators. Timing (synchronization) and localization are also important engineering challenges.

In addition to network design challenges, realistic simulators are needed for testing sensors, applications, and network protocols. A technology road map for inflection points would be also be useful. The core hardware platforms are becoming cheap, but academic research which pushes the range of applications and develops new sensors is expensive. Distributed wireless networks are still not yet built from off the shelf components for creating one's own research testbed. New hardware such as radios, sensors, and micro controllers are still being developed in research labs.

Two domains have emerged: i) geographically static sensors and ii) moving sensors. The second domain is more relevant for commercial opportunities. The Ottawa area has a range of activities in the WSN domain: i) companies that build sensor platforms; ii) research into sensors and actuators, particularly in the biomedical domain; and iii) work in academia and government research labs on network protocols and building testbeds. What may be missing is research on operating systems for embedded systems and middleware.

After the lecture, the audience held a brainstorming session on possible business opportunities around WSN technologies. A WSN opportunities page has been added to the Talent First Network wiki (http://www.talentfirstnetwork.org/wiki/index.php?title=WSN_opportunities).

Readers are encouraged to add new opportunities or contribute to an existing opportunity. Contacts are provided should you wish to discuss value propositions with the champion of an opportunity.

Thomas Kunz received a double honours degree in Computer Science and Business Administration and the Dr. Ing. degree in Computer Science from the Technical University of Darmstadt. His research focuses on various problems in mobile computing and distributed systems and mobile ad-hoc networks. He has published well over 60 technical papers in journals and conferences and is a member of ACM and the IEEE Computer Society.

Recommended Resources

An FDL'ed Textbook on Sensor Networks
http://www.informatik.uni-mannheim.de/~haensel/sn_book/

Mobile Ad hoc and Sensor Network Systems
http://www.crc.ca/en/html/manetsensor/home/home

Privacy and Security in a Connected World

"Privacy is an economic problem and often is associated with an economic tradeoff."

Fei Lee
(http://tinyurl.com/6md3nl)

On May 7, 2008, Douglas G. King, Assistant Professor of Systems and Computer Engineering at Carleton University, delivered a presentation entitled "Privacy and Security in a Connected World". The slides from the presentation are available here (http://www.talentfirstnetwork.org/wiki/images/f/f8/Security_and_privacy_May_7.pdf).

The TIM Lecture Series provides a forum to promote the transfer of knowledge from university research to technology company executives and entrepreneurs as well as research and development (R&D) personnel. This conference report presents the key messages and insights from the two sections discussed during Professor King's presentation.

Definitions

This section of the lecture introduced the domains of privacy and security in the context of a global market, with an emphasis on the privacy and freedom of information legislation applicable to organizations based in Canada. It also promoted discussion around the questions "In a connected world, is privacy still an issue?" and "Is it a problem if organizations share an individual's personal information or transaction history without the knowledge of the individual?".

Several key messages emerged from the definitions introduced during the first half of the lecture.

Regarding the difference between privacy and security, it was noted that:

- personal privacy is often traded off for national and personal security
- the balance between privacy and security is mediated by user profiles
- it is difficult to find an optimal balance between privacy and security since as the number of profiles increases, privacy is enhanced, while security is often enhanced by reducing the number of profiles
- corporations find it increasingly difficult to maintain their legal obligations regarding privacy and security

An important point is that security is multi-faceted in that it is much more than information technology (IT). IT security relies heavily on physical security and personnel security mechanisms, creating layers of safeguards. An emerging trend is being seen in security design. In the candy analogy for security architecture, there is a movement away from hard shell with soft center to the more clustered crunchy center approach.

Global vs. Canadian Context

The second half of the lecture discussed how Canadian business is affected by the Patriot (http://en.wikipedia.org/wiki/Patriot_act) and Sarbanes-Oxley (http://en.wikipedia.org/wiki/Sarbanes-oxley) Acts. The complexity and cost of adhering to these acts is often unworkable by small companies. Moreover, many small companies are still not compliant with Personal Information Protection and Electronic Documents Act (PIPEDA, http://en.wikipedia.org/wiki/PIPEDA). On the flip side, a long list of unsolved privacy and security problems provides many commercialization opportunities.

He then described the reasons why replacing existing Ontario health cards with smart cards failed as an example of how privacy trumps technology when people refuse to adopt. It was also noted that increased surveillance does not provide increased security.

Many questions were raised in the ensuing discussion. When asked if the tipping point from privacy to security was due to increased connectivity or an increased perception of threat in a post-911 world, Professor King responded that fear drives the process, but connectivity enables the technology and increased connectivity increases the fear of global threats. Other questions included:

Q. Is this the beginning of the end where we are subject to multinational global surveillance?

A. Global agreements won't happen, so there is no threat of a hard shell approach. However, clusters are quite likely to occur within geopolitical boundaries, or across domains with common interests such as OPEC (http://en.wikipedia.org/wiki/OPEC).

Q. If we're at the tipping point, what is/was the right balance?

A. There is no aggregate balance. In theory, there is a natural oscillation among contributing factors. Indeed, there are other contributing factors to both security and privacy, so it is not the case of a closed system or zero-sum game. There is a natural linkage through feedback between security and privacy that will result in oscillations due to feedback. It is possible for both privacy and security to be increased through this natural feedback. Increasing connectivity in both IT and global perspectives is one of the strong pressures toward reducing our personal privacy and increasing our collective security.

Q. Is it a question of balance or is it possible to increase both privacy and security?

A. Most mechanisms increase privacy for both good and bad purposes, but there are examples of side effects such as the common good provided by gun amnesties, needle exchanges, and anonymous Internet access.

Q. What about OpenID (http://en.wikipedia.org/wiki/Openid)?

A. This is one of many initiatives over the years which works well for existing communities but which doesn't build trust with who you will communicate. A trust relationship is required. OpenID and similar initiatives like PGP (http://en.wikipedia.org/wiki/Pretty_Good_Privacy) provide only the raw mechanisms for authentication and authorization, but rely on an external process to form a hierarchy or web of trust and guarantee trustworthiness.

Q. At least NSA's tactics are supported by the Patriot Act. What about the Communications Security Establishment (CSE, http://www.cse-cst.gc.ca/index-e.html)? Does anyone really know what the CSE is doing?

A. The CSE is working within the Canadian legal context, and is careful to make sure it abides by the rules of evidence within Canada. It is important in legal proceedings to make sure that the trail of evidence begins with information obtained through legal means.

Douglas King received his B.Sc., M.Sc. and Ph.D degrees in Theoretical Physics from the University of Guelph. In January 1989, he joined the Simulation and Modelling Research Group in the Department of Computer Science, University of Ottawa, as Research Associate and Part-time Professor. He has founded three high-technology companies, with a proven record of applying research to practical problems for both product development and strategic consulting. Dr. King's current research interests include: IT security management; Public Key Infrastructure applications; project management best practices; collaborative work environments; high-volume web site engineering; repositories and their access protocols; and copyright management.

Recommended Resources

Access to Information and Privacy
http://www.justice.gc.ca/eng/pi/atip-aiprp/index.html

Privacy Commissioner of Canada
http://www.privcom.gc.ca/

Canadian Internet Policy and Public Interest Clinic
http://www.cippic.ca

Blood on the Tracks: 6 Years of Technical Entrepreneurship in Ottawa

"Opportunity is missed by most people because it is dressed in overalls and looks like work."
Thomas Edison

In 2002, twenty nine engineers and computer scientists completed a Lead-to-Win (LTW) program in Technical Entrepreneurship. The LTW program was a pilot program designed for former Nortel employees to gain the skills needed to become entrepreneurs. Of the participants, fifteen started technology businesses, ten tried to attract venture capital funding, eleven tried to grow their companies with no venture capital funding, and seven established five technology businesses headquartered in Ottawa. These businesses attracted over $91 million from venture capital firms during one of the worst economic times to hit this region and created over 280 jobs globally.

On May 15th at the Partnership Conference Series, John Callahan and Tony Bailetti, directors of the LTW program, and three of the LTW graduates spoke about the lessons learned during and since the program. In addition to this conference report, the full text of the key messages and the slides from each speaker's presentation are available from http://tinyurl.com/3qe3vb.

While each speaker provided a slightly different perspective, several commonalities emerged. For example, successful entrepreneurs:

- enjoy what they do
- know their strengths, weaknesses and desires as well as the terrain in which they operate
- have a supportive family
- align key stakeholders, customers, management, employees, investors, board members, complementors, and intermediaries around a common purpose
- frequently interact with customers and potential customers
- network with other entrepreneurs, start-up CEOs, and risk capital providers
- remain optimistic, thick skinned, and perseverant
- do not distinguish between weekend and weekdays
- know where to invest
- are realistic about how much money they need and are aware they can't get it all at once
- recruit world class team members with diverse business experience
- select a business model rather than assume one
- are patient for growth and impatient for profits

Advice for first time entrepreneurs included:

- decide what success means to you
- regardless of your definition, success is validated by a paycheck
- value differences as there is great value in managing a diverse team
- find an office on the ground floor as at some point you will want to jump out a window
- find your defining question and use it to provide laser like focus and guidance when making decisions

• aim for base hits, not home runs

• find your differentiation that adds value to your customers

• first dominate a domain and then outsource it

A venture capitalist and an angel investor offered insights into funder-entrepreneur interactions. Angel and VC (venture capital) investors decide whether or not to invest in a startup by assessing the following: i) enthusiasm; ii) trustworthiness; iii) sales potential of product; iv) expertise of entrepreneur; v) likability of entrepreneur upon meeting; vi) growth potential of market; vii) quality of product; viii) perceived investor financial rewards; ix) niche market; and x) track record of entrepreneur.

When raising VC funds, a Canadian startup faces: i) scarcity of capital in Canada; ii) overcoming the "not in the US" barrier; and iii) lack of connections into VCs in the US. When seeking VC funding, go into it with your eyes open. You will have to give up control and take on additional challenges that may add little to the business. Do not mix and match angel and VC investment as it is extremely difficult to manage their conflicting priorities.

The regional director for NRC-IRAP (http://irap-pari.nrc-cnrc.gc.ca/) described the benefits of this program to both startups and existing businesses. This program provides technical specialists who understand the development and commercialization process; they are not just transfer payment specialists. An IRAP investment provides many benefits as it: i) reduces risk; ii) reduces burn rate; iii) validates the venture due to a rigorous due diligence process; iv) demonstrates resourcefulness of management teams; and v) requires firms to keep good records.

Professor Bailetti concluded the talks with a presentation outlining plans to make Ottawa a hub for ecosystem keystones. The ecosystem approach to wealth creation and appropriation requires the technical entrepreneur to:

• use a community's shared vision and then contribute to it

• launch market offers using the ecosystem's foundation platform

• compete for leadership positions in market space, niche and governance

• draw on a global pool of talent

• develop the capability to collaborate

• open development and commercialization processes to customers

In conclusion, the nine day LTW program provided first time technical entrepreneurs with: i) rules of thumb and practical experiences; ii) access to an ecosystem comprised of technical entrepreneurs and local legal, marketing, financial, and risk capital services; and iii) effective professors with abundant practical experience. The LTW program was one of the most economically impactful events to have occurred in Ottawa in a long time.

Recommended Resources

Bootstrapper's Bible
http://www.changethis.com/8.BootstrappersBible

SR&ED Program
http://www.cra-arc.gc.ca/taxcredit/sred/menu-e.html

Q. Most commercial software companies employ product managers to handle the planning and marketing of software products, whereas few open source projects have a product manager. Does lack of product management impact the users of open source?

A. Plenty of open source software (OSS) that can save businesses millions of dollars is available right now for download from sites like SourceForge (http://sourceforge.net). More importantly, OSS offers feature sets and mixes that often aren't available in commercial products because the market is too small, commercial companies don't understand it, or the problems aren't profitable enough to solve.

The great promise of open source is that you can have equal or more functionality than commercial software for free, and you have access to the source code if you have the desire, time, and skills to hack it into something new. This model was perfect when developers were writing tools for each other. Most OSS projects aren't under the stewardship of a commercial entity, although some of the most successful ones are, such as RedHat, Firefox, and OpenOffice. Most are built by and for a handful of developers "scratching an itch," who are not working with a product manager. Unfortunately, OSS has become a victim of its own success, and today, open source developers are facing a problem that threatens to turn legions of users against the software they rely on.

Most OSS projects are a meritocracy (http://en.wikipedia.org/wiki/Meritocracy), meaning that the developers care about developing for themselves and their own problems. If non-contributing users' problems happen to be solved, great. If not, "you have access to the code, feel free to build that feature yourself!"

Users understand that free software comes with limitations: there is typically only ad hoc support, updates are only as frequent as the developers care to make them, and bugs may go un-addressed forever. Users realized they were getting something for nothing, and were willing to put up with the lack of polish found in most OSS. However, OSS has become so pervasive, the boundaries in users' minds between OSS and commercial software have blurred.

Many OSS products are nearly indistinguishable to an end user from commercial software. This has changed the expectations of users to think that they are the persona that the developer is writing code for. But are they? Some applications, such as Firefox, have made the leap and are clearly developing for an end user. For an example of an OSS project that hasn't, look no further than Pidgin.

Pidgin is an open source instant messaging (IM) client. Recently, they changed the action of the field where the user types their message from a manually re-sizable window to a fixed size window that auto-re-sizes based on the amount of text typed. This sounds like a minor change, but it triggered a massive user revolt! Why?

First, the Pidgin developers violated the Principle of Least Astonishment (http://en.wikipedia.org/wiki/Principle_of_least_astonishment). Never take away functionality from the user when they upgrade. Second, the Pidgin developers let "Perfect become the enemy of Good." If you take the time to read through the entire discussion (http://developer.pidgin.im/ticket/4986), you see statements from the developers such as (paraphrased): "We want to find one solution that fits the needs of all users; we don't understand/don't agree with the use case that calls for a resizable input window."

Third, the developers became more entrenched as the discussion progressed, rationalizing the feedback as a “vocal minority,” and recommending that users use other applications for their needs. Finally, the thread devolves into the developers reminding everyone that they do this work on their own time, for their own enjoyment, and by the way, they are closing the bug report and tagging it as “will not fix.”

The most interesting reply on the bug report is from Dan Livingston:

"I teach “Collaboration in an Open Source World” at a local college. I have been searching for, and in this ticket have found, a perfect example where communication between open source developers and users fails at multiple, fundamental levels.

Obviously, the motivations of open source developers are varied; some do it for technical enjoyment, others enjoy knowing they are contributing intellectual capital to a better world. The problem is when the motivations of open source developers conflict with the expectations of users.

Consider every wildly successful open source project: the users are enthralled with their ability to perform new activities in ways previously unimagined. Rabid dedication grows, and an evangelical fan base results. Pretty soon, it’s obvious why users would not want to go with non-open source software alternatives.

What happens when those same new-found powers are taken away? What happens when the developers impose their personal dogmas upon the project? Even for as small an issue as chat window resizing, a minority (or majority) of users will emphatically express dissent."

“...The initial lure of open source software is that quality software should resoundingly meet the needs of users. As demonstrated up until Pidgin 2.4, the fan base has emphatically been extolling the virtues of Pidgin. But when developers take a feature away, presumably to implement a “better version”, and that better version in fact is a step backwards from the functionality previously available, they had better have a damn good reason. Such a reason is lacking here...(many of the statements seen in this ticket), which if executed within a corporate arena, would get developers fired. Developers, make note: you are doing a disservice to the community you claim to represent, and are doing so with false illusions that you are “right” because you have convictions in your justifications."

Later, Professor Livingston dresses down the development team with some well-placed satire, by proposing a fictitious letter ghost written for the Pidgin development team (http://developer.pidgin.im/ticket/4986#comment:287).

Obviously, there is a huge gap between the expectations of the users and the developers. Who normally bridges that gap? Product management (http://en.wikipedia.org/wiki/Product_management). A product manager would raise a flag on the change in functionality and help the engineering team prioritize feature sets based on the needs of the target users. Unfortunately, most OSS projects don’t have product managers, written personas, or target users; they have developers working for themselves.

I propose that product management should take a more active role in OSS by teaming with developers to identify the target audience and prioritize the users' needs. If developers can donate their time, there is no reason product managers can't do the same thing.

Another solvable issue is one of expectations. Open source projects should adopt a clear tag, license, or other marking indicating if the goal of that project is to "scratch an itch" or service a base of non-contributing users. Indicating the goal of a project up front at download-time (for the user), and at code writing time (for the developer) would set expectations appropriately.

Even on a "serviced" project, users need clear guidelines about what is and isn't acceptable feedback. The discussion above started out as civil, but quickly turned into a shouting match, with each side digging in. At that point, no one could compromise without losing face. An OSS product manager could have diffused the issue early, by involving users in the design (pre-code), and being a neutral party to explain the decision making process and tradeoffs.

The "Product Management Problem" is not unique or limited to OSS. You can easily find examples of commercial products that had poor product management. Pidgin offers a teaching opportunity in the OSS world, showing that no model is perfect.

It is common in the OSS world to hear statements such as "we don't need product management, that's for commercial companies; OSS already has a method to determine needs across groups of users: forking!" Forking is when a developer decides to branch off of a project to create something different, usually within the same vein and built on top of the work that has occurred to-date. A fork of Pidgin called Funpidgin exists to give users back the features that the Pidgin team "took away." On the one hand, forking is an inefficient way to solve this problem. On the other hand, unlike in commercial companies, open source projects are not resource-bound and can afford to be inefficient.

An OSS project can have one of two primary goals: either the developers are creating for themselves, or they are creating for others. To create for yourself means that you recognize user input but don't feel any obligation to take it. If the user happens to enjoy it, great. If not, fork and make something you like.

If you're creating for others, you should be interested in the wants and needs of your target user base. This might mean forgoing a "cool" or technically challenging feature like auto-resizing text boxes. Now that OSS looks, feels, and acts like many commercial packages, users assume that the application was developed with their needs in mind. If the application does not meet their needs, they feel justified in offering feedback. This is where the disconnect comes from: users who assume that an application was developed for them and programmers who believe they are building primarily for themselves. If the programmers are interested in working for a larger user base, a strong product manager could help them fill that gap.

For more information on this topic, see the author's blog which includes his initial entries and ensuing discussion (http://www.productbeautiful.com).

Paul Young completed his undergrad work at The University of Texas at Austin, and received a B.S. in Radio-Television-Film. He worked in various product management and marketing roles for Cisco's security and WAN managed services before becoming director of product management at a startup in Austin.

The goal of the Talent First Network Proof of Principle (TFN-POP) is to establish an ecosystem anchored around the commercialization of open source technology developed at academic institutions in Ontario.

The priority areas are the commercialization of open source in:

- Mapping and geospatial applications
- Simulation, modeling, games, and animation
- Conferencing
- Publishing and archiving
- Open educational resources
- Social innovation
- Business intelligence
- Ecosystem management
- Requirements management

Expected Results

The TFN-POP is expected to:

- Establish a healthy ecosystem anchored around the commercialization of open source assets
- Maximize the benefits of the investment in the Talent First Network by the Ministry of Research and Innovation
- Accelerate the growth of businesses in Ontario that use open source assets to compete

Eligibility to Receive Funds

Individuals eligible to receive funds are faculty, staff, and students of universities and colleges in Ontario.

Budget and Size of Grants

A total of $300,000 is available. Applicants' requests should not exceed $30,000.

The TFN-POP may provide up to 50 percent of total project costs.

Criteria

Proposals will be judged against the following five criteria:

- Strength and novelty of open source technology proposed
- Extent of market advantage due to open source
- Project deliverables, likelihood that the proposed activities will lead to deliverable completion on time, and effectiveness of the plan to manage the project
- Track record and potential of applicants
- Extent of support from private sector

Application

The electronic version of the application received by email at the following address: TFNCompetition@sce.carleton.ca will be accepted as the official application. The email must contain three documents: a letter of support, project's vitals, and a project proposal.

Letter of support: (maximum 2 pages) a letter, signed by the person responsible for the Technology Transfer Office or Applied Research Office of the academic institution that proposes to host the project and the faculty member or student who will lead the project, must be included. This letter should describe the nature of the support for the project from the academic institutions, companies and other external organizations.

Project's vitals: (maximum 1 page) The project's vitals must include:

- Person responsible for applied research or technology transfer at the college submitting the proposal: name, mailing address, telephone number, and email address

- Project leader: name, mailing address, telephone number, and email address

- Team members: names, mailing addresses, telephone numbers, and email addresses

- Budget: Total budget, with TFN's contribution and that of other organizations

- TFN investment: TFN contribution broken down by payments to students, payments to faculty, and payments to project awareness activities

Project proposal: (maximum 5 pages) Project proposal must include the following:

- Benefits: (maximum 1/2 page) Description of the benefits of the proposed project, and an overview of the context within which the project is positioned

- Advantage: (1/2 page) Market advantage provided by open source assets used in the project

- Information on applicants: (maximum 1.5 pages) Background information to help assess the track record and potential of the people who are key to the project and the college

- Project plan: (maximum 2.5 pages) Description of the deliverables (what will be delivered and when); key project activities; nature of the involvement from companies, and other external organizations; and plan to manage the project

Evaluation & Deadline

Proposals will undergo review by the Expert Panel established by the TFN-POP. The Chair of the Panel may contact the applicants if required. A final decision will be communicated to the applicants within 30 days after the email with the official application is received.

There is no deadline. Applications will be evaluated on a first-come basis until the $300,000 available is committed.

Contacts

Luc Lalande: Luc_Lalande@carleton.ca

Rowland Few: rfew@sce.carleton.ca

About the Talent First Network

The Talent First Network (TFN) is an Ontario-wide, industry driven initiative launched in July 2006 with the support of the Ministry of Research and Innovation and Carleton University. The objective is to transfer to Ontario companies and Open source communities: (i) Open source technology, (ii) knowledge about competing in Open source environments and (iii) talented university and college students with the skills in the commercialization of Open source assets.

Turning the Tables: The Impact of Open Source on the Enterprise Database Market

Copyright: 451 CAOS Research Service

From the Abstract:

This report examines the adoption of open source database software to date and explores what barriers the open source vendors have to overcome to mount a meaningful long-term challenge to the big three. The report also assesses the response of the incumbent vendors to the open source challenge, and includes a survey assessing the attitudes toward open source and proprietary databases among executives responsible for the procurement of database management systems.

http://www.the451group.com/caos/caos_detail.php?icid=539

The Total Growth of Open Source

Copyright: Amit Deshpande, Dirk Riehle

From the Abstract:

Software development is undergoing a major change away from a fully closed software process towards a process that incorporates open source software in products and services. Just how significant is that change? To answer this question we need to look at the overall growth of open source as well as its growth rate. In this paper, we quantitatively analyze the growth of more than 5000 active and popular open source software projects. We show that the total amount of source code as well as the total number of open source projects is growing at an exponential rate. Previous research showed linear and quadratic growth in lines of source code of individual open source projects. Our work shows that open source is expanding into new domains and applications at an exponential rate.

http://www.riehle.org/2008/03/14/the-total-growth-of-open-source/

Scan Open Source Report 2008

Copyright: Coverity

From the Executive Summary:

Since 2006, the Scan site has analyzed over 55 million lines of code on a recurring basis from more than 250 popular open source projects such as Firefox, Linux, and PHP. This represents 14,238 individual project analysis runs for a total of nearly 10 billion lines of code analyzed. The collection of such a large, consolidated set of data regarding the security and quality of source code provides a unique opportunity to examine coding trends from a diverse collection of code bases. The Scan Report on Open Source Software 2008 was created to provide an objective presentation of code analysis data from the Scan site

http://www.coverity.com/library/pdf/Coverity-Scan_Open_Source_Report_2008.pdf

May 26

CodeFactory Open House

Ottawa, ON

TheCodeFactory provides co-working and casual work space as well as private office space for start-ups, complete with Internet access and business services as required. The co-working space provides a lounge area where co-workers can relax, chat with other like minded people, or work in a very casual setting. Meeting rooms can be booked for private meetings and a business services area provides print, scan, fax or photocopy services.

http://thecodefactory.ca/

May 30

GOSLING Anniversary

Ottawa, ON

GOSLING (Getting Open Source Logic INto Governments) started as a couple of informal Friday get-togethers after work at the pub, to bounce around some ideas ahead of the first F/LOSS event hosted by the Government of Canada. On Friday May 30, we're organizing the biggest gaggle ever to celebrate the 6th Anniversary. Please RSVP via the following website link so the Parliament Pub staff can plan the food.

http://goslingcommunity.org/anniversary

June 2-3

OSBOOTCAMP 6: Geospatial Software

Ottawa, ON

This will be a two day event focusing on open source geospatial software. Come and hear industry experts present talks on web mapping, GIS analysis, OSGEO projects and more.

http://www.osbootcamp.com/index.php?page=osbc6

June 2-5

Geotec

Ottawa, ON

The GeoTec Event provides a unique gathering place for geospatial technology professionals from all disciplines to interact and learn from each other's experience and knowledge. The program is designed to help you discover cutting-edge geospatial technology solutions.

http://www.geoplace.com/ME2/dirsect.asp?sid=F1E958ECB4E84C1C97324D4851580DDB&nm=GeoTec+Event

---

June 10-12

Infosecurity Canada

Toronto, ON

Infosecurity Canada is the most up-to-date resource defining where the IT security industry is going. No matter what industry you're in—from finance to government to education to healthcare, you'll find state-of-the-art technologies and new solutions for all your information infrastructure needs.

http://www.infosecuritycanada.com

June 17-18

Government Web 2.0 and Social Media

Ottawa, ON

Attend Canada's first-ever conference on Government Web 2.0 and Social Media and get the answers you need from seasoned experts, both inside and outside government. Learn the latest technology and communication strategies and how they can positively impact your department. Topics include proven techniques for developing a strategic plan to incorporate new technology and learning how to shift towards an open source platform for social media integration.

http://www.infonex.ca/829/overview.shtml

June 25

Symposia On Eclipse Open Source Software

Ottawa, ON

Eclipse and OMG are jointly organising symposia to promote and build on the partnership between Eclipse's open source software and OMG's open standards during the OMG Technical Meeting in Ottawa. The symposia is a unique opportunity to participate in shaping the joint future of the Eclipse Open Source community and the OMG Open Standards community. Please join us for a day of stimulating technical planning and discussion.

http://www.omg.org/news/meetings/eclipse-omg-2008/index.htm

June 25-27

Open Scholarship

Toronto, ON

The objective of the conference is to bring together researchers, lecturers, librarians, developers, business executive, entrepreneurs, managers, users and all those interested in issues regarding electronic publishing in widely differing contexts. This year's presentations include the topic of Open Access.

http://www.elpub.net/

The goal of the Open Source Business Resource is to provide quality and insightful content regarding the issues relevant to the development and commercialization of open source assets. We believe the best way to achieve this goal is through the contributions and feedback from experts within the business and open source communities.

OSBR readers are looking for practical ideas they can apply within their own organizations. They also appreciate a thorough exploration of the issues and emerging trends surrounding the business of open source. If you are considering contributing an article, start by asking yourself:

1. Does my research or experience provide any new insights or perspectives?

2. Do I often find myself having to explain this topic when I meet people as they are unaware of its relevance?

3. Do I believe that I could have saved myself time, money, and frustration if someone had explained to me the issues surrounding this topic?

4. Am I constantly correcting misconceptions regarding this topic?

5. Am I considered to be an expert in this field? For example, do I present my research or experience at conferences?

If your answer is "yes" to any of these questions, your topic is probably of interest to OSBR readers.

When writing your article, keep the following points in mind:

1. Thoroughly examine the topic; don't leave the reader wishing for more.

2. Know your central theme and stick to it.

3. Demonstrate your depth of understanding for the topic, and that you have considered its benefits, possible outcomes, and applicability.

4. Write in third-person formal style.

These guidelines should assist in the process of translating your expertise into a focused article which adds to the knowledgable resources available through the OSBR.

Upcoming Editorial Themes

June 2008	Security
July 2008	Accessibility
August 2008	Education
September 2008	Social Innovation

Formatting Guidelines:

All contributions are to be submitted in .txt or .rtf format and match the following length guidelines. Formatting should be limited to bolded and italicized text. Formatting is optional and may be edited to match the rest of the publication. Include your email address and daytime phone number should the editor need to contact you regarding your submission. Indicate if your submission has been previously published elsewhere.

Articles: Do not submit articles shorter than 1500 words or longer than 3000 words. If this is your first article, include a 50-75 word biography introducing yourself. Articles should begin with a thought-provoking quotation that matches the spirit of the article. Research the source of your quotation in order to provide proper attribution.

Interviews: Interviews tend to be between 1-2 pages long or 500-1000 words. Include a 50-75 word biography for both the interviewer and each of the interviewee(s).

Newsbytes: Newsbytes should be short and pithy--providing enough information to gain the reader's interest as well as a reference to additional information such as a press release or website. 100-300 words is usually sufficient.

Events: Events should include the date, location, a short description, and the URL for further information. Due to the monthly publication schedule, events should be sent at least 6-8 weeks in advance.

Questions and Feedback: These can range anywhere between a one sentence question up to a 500 word letter to the editor style of feedback. Include a sentence or two introducing yourself.

Copyright:

You retain copyright to your work and grant the Talent First Network permission to publish your submission under a Creative Commons license. The Talent First Network owns the copyright to the collection of works comprising each edition of the OSBR. All content on the OSBR and Talent First Network websites is under the Creative Commons attribution (http://creativecommons.org/licenses/by/3.0/) license which allows for commercial and non-commercial redistribution as well as modifications of the work as long as the copyright holder is attributed.

The Talent First Network program is funded in part by the Government of Ontario.

The Technology Innovation Management (TIM) program is a master's program for experienced engineers. It is offered by Carleton University's Department of Systems and Computer Engineering. The TIM program offers both a thesis based degree (M.A.Sc.) and a project based degree (M.Eng.). The M.Eng is offered real-time worldwide. To apply, please go to: http://www.carleton.ca/tim/sub/apply.html.