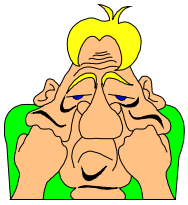
Thursday, March. 31, 2005
Vol. 21, No. 06

Soft•letter

BUSINESS INSIGHTS FOR SOFTWARE DEVELOPERS & PUBLISHERS

Muscle-Bound Messiah? A Look at the Current State of Agile Programing, Part II of II

We continue our look at the current state of agile development with Matt Stephens, senior architect, programmer, and project leader based in Central London. He's the coauthor of "Extreme Programming Refactored: The Case Against XP" (Apress, 2003) and "Agile Development with ICONIX Process" (Apress, 2005) and is also working on "Use Case Driven Object Modeling – Theory and Practice" (Addison-Wesley) to be published later this year. We strongly recommend you visit his site, www.softwarereality.com, where you can enjoy his lively musings on current development practices.



The news is grim for tech support salaries; salaries either stayed flat or lost ground for the most part. *See pages 4-5.*

Matt, agile adherents often make fun of "waterfall" approaches to product development (in other words, one "part" of a software product is completed before beginning the next; are there circumstances and companies which should use this development approach?

It's difficult to get away from the waterfall approach, because invariably you need to finish something before you move onto the next thing. The distinguishing factor is the granularity of each part; how much time is spent gathering requirements before coding. I could still see occasional uses for big monolithic waterfall projects, but it reminds me of the overuse of Enterprise JavaBeans (EJB) in the J2EE world. EJB is a powerful (if top-heavy) technology, because it naturally assumes that your application will deploy to a distributed cluster of servers and that you'll want to scale infinitely, therefore everything is set up in anticipation of this big, heavyweight project which ultimately never really happens.

So, to return to the question, I see the waterfall process as the EJB of the methodology world. Waterfall has been massively overused, and misapplied to thousands of projects which just didn't need its heavyweight approach, and which suffered as a result (just like with EJB). There are doubtless some large-scale or mission-critical projects where the waterfall process is appropriate and where the requirements are unlikely to change (e.g. payroll systems) so a more fluid, "seat-of-your-pants" process like XP wouldn't be appropriate. Having said that, certain agile practices can still be carefully applied to such mission-critical applications. *(continued on page three)*

Publisher & Managing Editor
Merrill R. Chapman
rickchapman@softletter.com
860/663-0552

Editor
Gail Wertheimer
gail@softletter.com
508/405-0375

Editor Emeritus
Jeffrey Tarter
jtarter@softletter.com
617/668-0028

Editorial office
Soft•letter
34 Sugar Hill Rd.
Killingworth, Conn.
06419

Subscription office
United Communications Group
11300 Rockville Pike
#1100
Rockville, Md. 20852
301/287-2718
866/313-0973
customer@softletter.com

www.softletter.com

Advisory Board

Non-Recurring Engineering, Part II of II

by Daniel Shefer

When costing out an NRE project, you need to be aggressive in establishing your margins. The idealized model I usually apply breaks down this way (adjust to your actual circumstances). Let us first assume an NRE project will take 10 hours to develop and each programmer is being paid $100 an hour, for an initial amount of $1000. Add a 40% overcharge to this amount; this brings us to $1400. Add an additional 25% "fudge" factor (to account for all unexepected delays and problems); we're now at $1750. Then add 13% on top of this figure to acount for your G&A costs (source: Softletter Benchmark 50: General and Administrative). This brings us to $1977.50. Multiply this amount by five for a total amount of $9887.50.

It is critical to understand the necessity for this "five" factor. Most companies spend approximately 20% of their revenue on R&D (source: Softletter Benchmark 50: Research and Development). Through experience, I've learned to correlate extensive NRE work with a "5X" negative impact on sales. This is because the resources allocated for NRE work are diverted from producing sale generating products. My formula accounts for the actual penalty paid for embarking on projects that generate short-term revenue but lose money over the long haul. Don't forget that support and maintenance costs should also be factored into the ultimate pricing equation, especially if the NRE work will only be available to a single or limited number of customers.

Many companies delude themselves into believing they can cut costs by hiring consultants to do NRE work. My experience belies this. You need to be aware that every problem or ambiguity an outside resource faces will result in them stopping work on the NRE project until they've obtained answers to their questions. The problem only becomes worse if your product's interface and API are not very well defined from an external party's technical viewpoint. In most cases, hiring a consultant to do NRE work is basically the same as hiring and training a new employee.

When a customer pays for NRE, they often expect to obtain rights to the functionality they paid for. This is not always desirable from the ISV's standpoint. There are several approaches an ISV can take:

- Insist that the NRE work becomes an integral part of the product. This is the best solution for most vendors. Traditional commercial software companies almost always follow this model.

- Offer the customer a time-limited exclusivity to the added functionality. This allows the vendor to eventually leverage their work for other customers.

- Negotiate a revenue sharing scheme for money generated from the new functionality; in these cases, the NRE must be packaged and sold separately. A serious downside to this tactic is that it is hard to set up and track over time; the agreement should thus be time-limited. Few companies and customers are positioned to make this approach work; your chances are better if the NRE is done as part of an OEM agreement.

- Bundle the new NRE-sponsored capability with a discount on future purchases.

Daniel Shefer, director of product marketing, SanDisk, 140 Caspian Court, Sunnyvale, CA 94089; 408/228-7212. E-mail: dani@shefer.net.

There are many new variants of agile programming being introduced: SCRUM, for instance, and feature driven development; are any of these a significant improvement over the basic agile methodologies?

There's a real spirit of innovation and advancement in the agile world at the moment. Processes borrow openly from each other in order to create the "next big thing"; e.g. the planning game in XP was actually "borrowed" wholesale from SCRUM; and XP itself is predated by nearly a decade by DSDM and Boehm's spiral process. The trend now seems to be away from the extremes of XP and towards a more realistic, balanced approach.

"EJB has its uses for the very few really big projects that need all of EJB's enterprise features, but for most projects it's overkill."

—Matt Stephens
Software Reality

Feature driven development (FDD) is a step in the right direction, as it nudges the pendulum slightly away from the lightweight extreme and places a healthy emphasis on analysis and design modeling without overdoing it. So it's encouraging to see some more professionally presented agile methodologies which don't try to be cool or wicked (or is that "wiki"?), they just try to be a software development methodology.

For the latest evolutionary step, I would point you towards Agile ICONIX, which itself is based on a core, lightweight object modeling process which has been around for over a decade What we've tried to do is to define a core subset of agile techniques which work well together, which can be used as a starting-point in your own projects. (More about it here: http://www.softwarereality.com/design/agileiconix.jsp).

What, in your opinion, is the role of product marketing in the agile development process?

Product marketing is equally as important in the agile development process as in any other software development process. During development of any software product, there's invariably a tug-of-war between various stakeholders, who each want the project to go in their direction: product marketing, sales, the design team, the architects, the project manager, QA; and occasionally even the customer. This tug-of-war is as it should be, and if all goes well, a balance is achieved and the finished product provides a good compromise for all parties involved. If it goes wrong, of course, you end up with an unbalanced product which has a heavy bias toward one of the stakeholders involved.

"Open Source products are notoriously geared towards the technically minded: difficult to install, with out-of-date documentation, and often little cohesion or overall "vision" to bind one successive release to the next."

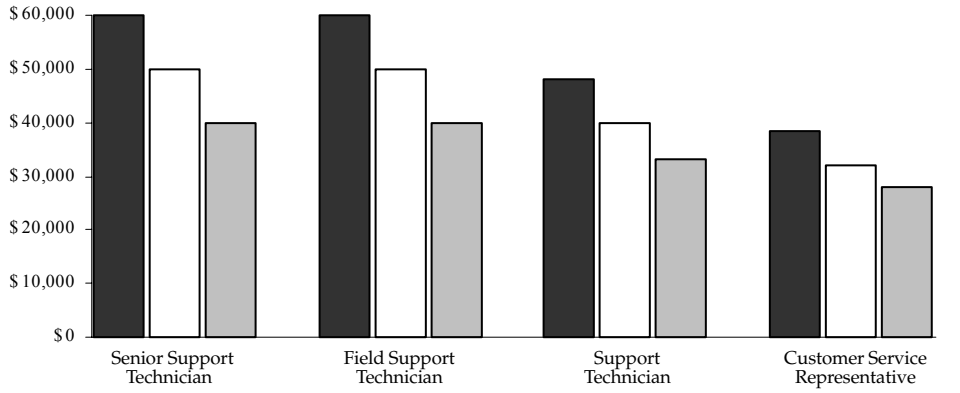
—Matt Stephens
Software Reality

For example, if product marketing became too powerful, the product would be all bubble and no substance. If the sales team overdid it, the product would be great for demos (e.g. very easy for a beginner to produce quick results, but there'd be nothing there for intermediate or more advanced users; in other words, not particularly useful for any "real-world" purposes). If the programmers are allowed to run riot, the whole thing will be too inwardly focused on the software design, and in fact may never even get finished because the team is too busy refactoring endlessly (the failed C3 project springs to mind) and making it all really "cool". If QA gets too much power, the team may spend more time filling out forms than writing software. *(continued on page six)*

Pay for Performance

Skill levels are an important variable in support pay for most larger organizations. The "most skilled" employees **(black bars)** typically earn 20% more than their "average" counterparts **(white bars)**, while the "least skilled"**(gray bars)** earn about 15%-20% less.

Source: ASP 2005 Tech Support Salary Survey



Benchmarks: Tech Support Salary Trends

Especially in enterprise markets, software companies have come to rely more than ever on revenue from maintenance and fee-based support services. Yet the latest salary data from the Association of Support Professionals (ASP) shows that the growth in service revenues hasn't translated into higher pay for the people who deliver those services.

In fact, the ASP's 2005 salary survey shows a surprisingly widespread salary *freeze* across almost every major job title that the ASP tracks. Support executives, department managers, analysts, senior-level and entry-level technicians—all ended up with no increase in median pay for 2004. Field support technicians, a relatively small group, actually suffered a hefty 14% drop in pay for the year. The only group to see any gains was the least-skilled category, customer service reps, whose median pay rose by 6.7%.

According to the ASP's survey report, "This salary freeze seems to have little to do with economic conditions or tight services margins. IT budgets have largely recovered from the 2002-2003 recession, and services continue to be more profitable and faster-growing than almost any other source of software company revenues."

The ASP points out that the current loss of momentum in support pay is bound to make it difficult for the software industry to recruit and retain talented people in support-related jobs. Eventually, the talent drain is likely to impact the ability of software companies to grow their services businesses and use services as a competitive edge. In addition, support jobs are often an entry point for staff jobs in R&D and sales support, so depressed salaries may have a long-term effect on company-wide recruiting as well.

2005 Technical Support Salary Survey, by The Association of Support Professionals, 122 Barnard Ave., Watertown, Mass. 02472; 617/924-3944. Web: www. asponline.com. Price: $60.

Tech Support Salaries	Count	High	Low	Median
Senior Support Executive	135	**$130,000**	**$85,000**	**$100,000**
Department Manager	162	**$85,000**	**$54,000**	**$70,000**
Analyst/Project Manager	106	**$73,000**	**$50,000**	**$60,000**
Senior Support Technician	154	**$62,500**	**$41,450**	**$50,000**
Least skilled	89	$55,000	$35,000	$40,000
Most skilled	100	$74,000	$45,000	$60,000
Field Support Technician	68	**$62,000**	**$40,000**	**$50,000**
Least skilled	38	$54,500	$35,000	$40,000
Most skilled	42	$75,513	$50,000	$60,000
Support Technician	150	**$50,000**	**$35,000**	**$40,000**
Least skilled	95	$40,000	$30,000	$33,000
Most skilled	95	$60,000	$39,000	$48,000
Customer Service Rep	82	**$40,000**	**$27,750**	**$32,000**
Least skilled	54	$35,000	$25,000	$28,000
Most skilled	56	$52,500	$31,500	$38,500

Source: 2005 ASP Tech Support Salary Survey. Note: "Count" is the number of responses in each salary category or sub-category. "High" salary is the median for the top 50% of all salaries in each category; "Low" salary is the median for the bottom 50%.

The ASP's tenth annual Tech Support Salary Survey reflects compensation data collected from 196 tech support organizations with a total of more than 16,000 support employees and managers. The full survey includes breakouts by company size, support organization size, product price, employee skill level, and geography. As part of its methodology, the survey has used the following job descriptions consistently since 1996:

- **Senior Support Executive (vice president or director level):** "Coordinates activities and budgets of multiple support groups or sites. Meets regularly with senior corporate management and key customers."
- **Department manager:** "Manages day-to-day activity of a single support center staff."
- **Analyst/Project Manager:** "Manages major business activity; usually has no direct reports."
- **Senior Support Technician:** "Answers escalated calls; may function as a group or team leader."
- **Field Support Technician:** Provides on-site service, primarily for enterprise products."
- **Support Technician:** "Provides first-level solutions, primarily over the phone."
- **Customer Service Rep:** "Answers routine service questions; routes calls to technicians."

The agile development process helps to achieve this balance because it gets all the stakeholders actively involved, so product marketing (along with all the other stakeholders) gets to give early and continual feedback on the evolving product—although I would say that XP makes this intrinsic aspect of product development more difficult, because XP requires all the project stakeholders to "speak with a single voice" [to the programmers]. This is difficult if the stakeholders are all mixing in the same room as the programmers. Traditionally in non-XP projects, "speaking with a single voice" is achieved by producing a requirements spec, thrashing out the details until a balanced specification is achieved which is agreeable to all involved. With XP this fundamental issue is glossed over.

Do you regard Open Source development as an "agile" methodology?

You know, there's this curious myth surrounding Open Source that they're essentially these huge, distributed-team projects with thousands of work-from-home programmers contributing disparately to this big, organic code base. While this might be the case for a very few high-profile Open Source projects, the vast majority are small efforts each run by a tiny group of "core" programmers. These programmers are often friends or colleagues, and they more than likely meet regularly in person. But even in the big OS projects, it tends to be a small group of core programmers who contribute the majority of the code and the bug fixes.

That's why Linus Torvalds is widely regarded as a control freak, and rightly so; treading on people's toes is probably in his job description. It's the only way to keep such a huge project focused. Notice how the areas outside his core sphere of control have splintered into a number of different, largely incompatible and competing "product-ettes". Where there's a lack of control there's diversification, plain and simple. That's a good thing when it happens between different products, but when it happens inside a single product, it's infighting and it isn't healthy. Software agility, for all its fluidity, is by its nature high-discipline. It has to be, otherwise everything just splinters and falls apart. So for an agile Open Source project to work, it has to be tightly controlled.

OS projects typically are very fluid. It's commonplace to see an OS product rearchitected, and big refactorings take place (e.g. the JGraph product's APIs seem to change fundamentally with each minor point release). However, because of the nature of OS, the shift of balance tends to be very much in favor of the programmers so marketing rarely even arises as an issue, which is scary. OS products are notoriously geared towards the technically minded: difficult to install, with out-of-date documentation, and often little cohesion or overall "vision" to bind one successive release to the next. The balance is wrong for what is an increasingly consumer-targeted product.

Luckily things are changing in the OS world in general, and commercial interests have already begun to provide the much-needed balance, the crucial missing stakeholders—i.e. marketing, sales, and QA. We're starting to see documentation improve (e.g. Spring Framework and Eclipse both have very good up-to-date documentation); and OS products (at least the ones that have the backing of a large influential company) are gradually becoming more focused towards the appropriate marketing demographic.

So (to answer the question) we're seeing small closely-knit teams (contrary to popular perception), high levels of discipline and maintenance, multiple iterations, requirements evolving over time, and an unclear single specific customer. That seems pretty agile to me!

Matt Stephens, head of tools development, Corizon, 119-121 Middlesex Street, London E1; +44 (0) 207/539-6800. E-mail: matt@softwarereality.com; www.softwarereality.com.

Passive Vs Active Shareholders

By Miro Parizek

Shortly after founding one of my first IT companies, my partner and I received an offer of $1 million for our share of the company and $50,000 for the outside investors' share. The nascent company was not profitable, had no IP, and was facing significant challenges. My partner and I might have accepted the $1 million for our share; however, we—the "active shareholders"—only owned 48% of the company and the buyer was not willing to increase their offer. Unfortunately, receiving less than 5% of the total deal was not acceptable to the "passive shareholders," who held over 50% of the shares.

Another example of this dilemma involved a corporate owner with a 51% share of a subsidiary where the business unit's management and its employees held 49%. A prospective buyer had proposed to purchase 100% of the shares for up to $7 million in total consideration, of which 40% of the consideration was in the form of an earn-out ($2.8 million) and the rest in cash ($4.2 million). Wisely enough, expectations were set in advance and agreed upon: the corporate ("passive") seller would receive cash only and the unit's management and employees would receive any earn-outs.

The challenge was how to value the earn-out versus the cash. Is a $2.8 million earn-out worth the same amount in cash? No, but it is clearly worth some amount of cash. That relative value depends on the discount factor associated with the earn-out. The shareholders had felt they could negotiate these relative values amongst themselves. The divesting corporation was willing to accept a discount on the comparable value of the earn-out but not to the level the "active sellers" desired. Even though all the sellers agreed the offer was acceptable as a whole, the deal fell apart because the selling groups could not agree on the relative values of cash versus earn-out.

This deal might have been salvaged if the transaction had been kept simple and involved only cash. Another option was to agree to split all forms of consideration on a pro-rata basis.

Miro Parizek, senior vice president, Corum Group, 10500 NE Eighth St., Bellevue, Wash. 98004; 425/455-8281. E-mail: mparizek@corumgroup.com.

Company/Description	Acquired by	Price/Terms	Revenues	Multiple
Ascential Software (ASCL) *• Data integration software*	IBM (IBM)	$619,300,000 *Terms: All cash*	$271,900,000	**2.28**
Retek Software *• ERP software for the retail industry*	Oracle (ORCL)	$579,004,000 *Terms: All cash*	$174,240,000	**3.32**
Ulead Systems *• Video, imaging and DVD authoring software*	InterVideo (IVII)	$48,680,000 *Terms: All cash*[1]	$44,000,000	**1.11**
Pinnacle Systems (PCLE) *• Image and DVD editing*	Avid Technology(AVID)	$371,920,000 *Terms: Cash/stock*	$331,230,000	**1.12**

1. Acquires majority interest with cash.

CORUM
MERGERS & ACQUISITIONS

CardFile

Software Cost Estimation Resources

- **Costar** (www.softstarsystems.com): Cost estimation software supports COCOMO II, allows live links to Excel.
- **Costxpert Group** (www.costxpert.com): Cost estimation software supports COCOMO compliancy, 32 life cycles and standards, and numerous languages.
- **CrossTalk: The Journal of Defense Software Engineering** (www.stsc.hill.af.mil/crosstalk/about.html): Site provides very useful articles on software development and requirements issues.
- **Estimating Software Costs by Capers Jones** (McGraw-Hill): Very comprehensive tome on the topic of estimating software development costs. Also covers support and documentation costs. Does not provide extensive coverage of object-oriented development.
- **SEER** (www.galorath.com/tools_soft.shtms): Develops software tools to assist in estimating cost of software development.

ZD NET REPORTER CHRIS JABLONSKI on software brokers: "Brokers fill the chasm between software publishers and software customers, and they'll be taking care of everything vendors and CIOs don't have the time and resources to deal with, including, but not limited to, securing the solution, testing and certification, handling SLAs, actively managing licenses, identifying and recommending industry innovations, and creating intelligence through integration. They'll use standards like SOA and ITIL, an integration platform, possess a large IP inventory, and build stable alliances." (Quote on ZD Net, 03/23/05)

SÉRGIO AMADEU, president of Brazil's National Institute of Information Technology, discussing Brazil's PC Conectado, a program that subsidizes purchases of computers by millions of low-income Brazilians: "We're not going to spend taxpayers' money on a program so that Microsoft can further consolidate its monopoly. It's the government's responsibility to ensure that there is competition, and that means giving alternative software platforms a chance to prosper." (Quoted in the New York Times, 03/29/20005)

FRED MEYER, CEO OF START-UP CAST IRON SYSTEMS, on software purchasing habits: "Customers today are starting to buy software the way they bought hardware 10 years ago. There's no longer magic in software—it's just another tool to get the job done." (Quoted on CNET News.com, 03/28/2005)

SOFTLETTER EDITOR EMERITUS JEFF TARTER on discovering the actual origins of the "Ashton" in Ashton-Tate: "I'm totally devastated to learn that Ashton the parrot is an urban myth....This is like learning there's no tooth fairy." (Quoted on www.adamgreen.org, podcast "Naming Software and Companies in the Eighties.")

Soft•letter is published 24 times per year; entire contents copyright © 2005 by Soft•letter.

ISSN: 0882-3499.

Subscription rates: $395 worldwide. Subscription office: United Communications Group, 11300 Rockville Pike, #1100, Rockville, Md. 20852-3030; tel 301/287-2718 866/313-0973 customer@softletter.com

www.softletter.com