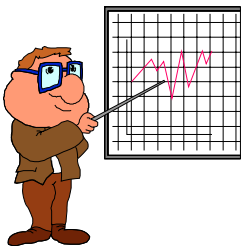
Tuesday, Mar.15th, 2005
Vol. 21, No. 05

Soft•letter
BUSINESS INSIGHTS FOR SOFTWARE DEVELOPERS & PUBLISHERS

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

In the late 90s a new movement swept the software development world, that of "agile" programming. Basic agile principles consist of focusing on building products in short iterations that deliver tangible capabilities, maintaining a close proximity to the prospective users and initiators of a software project (stakeholders), a de-emphasis on the development of written requirements, constant "refactoring" (rewriting) of your software as development progresses and unit testing. There are many different schools of agile development, including Extreme Programming (XP)—by far and away the most popular and well known—Crystal, Adaptive, SCRUM, and so forth.



VC investments in 2004 reverse a three-year downward trend
See pages 4-5.

Agile's initial poster child was the Chrysler Comprehensive Compensation project, or C3. C3 was begun as an ambitious attempt by XP advocate Kent Beck and other agile aficionados to replace the car company's payroll software with a new system developed under agile principles. C3 was (despite the vigorous denials of agile adherents) a failure; the software was not delivered on time and to specification and C3 was terminated by Chrysler (though in all fairness, portions of the new system were implemented and used for a brief period before the entire project was halted).

Despite this setback, agile programming techniques have proved very popular in the industry. However, after recently working with a medical systems software company struggling to survive after a product requirements disaster created under an XP regimen, we've become somewhat skeptical of the many claims made for agile techniques.

To help understand what a company can, and cannot, expect from agile development we turned to Matt Stephens, a 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. He is a popular speaker at software conferences and writes regularly for several software journals including Dr Dobbs, Software Development and Application Development Trends. We strongly recommend you visit his site, www.softwarereality.com, where you can enjoy his lively musings on current development practices.

Matt, if a software company implements Agile programming techniques as they're described in such classic books as Kent Beck's "Extreme Programming Explained" will a company be able to release their software products earlier to *(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 I of II

by Daniel Shefer

Non-recurring engineering (NRE) is a onetime development effort by a vendor for a customer. NRE requests are driven by a feature or capability that a product lacks and for which a customer is willing to pay. (It is **always** a mistake for a software company to agree to do NRE work without attaching a monetary value to the feature and the development effort.) Normally, a feature developed via a NRE process will be added to your product's overall capabilities.

While NREs are usually dreaded by most software companies, they often represent a marketing opportunity. If a company is willing to spend money on a NRE effort they are probably planning on being your customer for a long time.

NREs are generally requested by customers in the following circumstances:

- A customer feels the software doesn't "behave" properly. This can be because of a missing feature, an unsupported platform, unsupported device, a clumsy or hard-to-use interface, etc. A customer's willingness to pay for a NRE project is a vital indicator of how much they truly believe they need a capability.

- A customer elevates a feature's priority and persuades the vendor to move it into an earlier release. From the vendor's perspective, this is a resource scheduling issue.

Vendors need to carefully examine the implications of changing their schedule. Areas that can be impacted include:

- Revenue projections, as a features timeline can affect the willingness of customers to buy your product or purchase an upgrade.

- Customer expectations that a particular capability will be available; clients can become very upset if promised features are pushed off.

- Integration with other modules and products. Adding a feature or platform almost always affects other product modules and runs the risk of destabilizing the build.

NRE is **not** a replacement for the normal process of bug fixing and providing second level customer support. Established firms should avoid falling into the trap of resorting to NRE to finance the development of a product's core functionality. (Startups, however, often resort to the practice out of necessity.) Nor should NRE work be undertaken to develop capabilities that do not fit in with a product's strategic direction (for example, developing an inventory management module for a financial services system); reserve NRE development for a feature set's "middle ground."

It is also important to remember that NRE is not a replacement for professional services. Customization and integration are the domain of professional services, not development. Your product architecture needs to support this differentiation and you need the resources and skills in place to ensure you can execute.

Daniel Shefer, director of product marketing, SanDisk, 115 Hobart Ave, Apt. B, San Mateo, CA 94402; 408/228-7212. E-mail: dani@shefer.net.

market with customer-desired features and less bugs?

There's no simple answer to this, as all agile processes aren't the same, and you'll get better results with some than with others. Also, not all software projects are the same—the number of different types of software projects is roughly equal to the number of different software projects. Getting a product to market on time with less bugs depends as much on the people involved as it does on the methods they use. It depends on which agile techniques the company adopts, whether they follow them to the letter and tailor them at the right times, or if they adopt the wrong practices for their project, or let the right practices slip, and so on.

For example, if they choose to co-locate their entire team of programmers and analysts into one room, the cost of communication will go down and they can relax certain other disciplines such as documenting their requirements and design in detail. Why write something down when you can always ask Bob the business analyst sitting three desks away from you? But if the project is too big for everyone to fit in one room, or the business analysts just can't be near the programmers all the time, then following the same "agile techniques" (not creating permanent documentation in detail) suddenly becomes a high risk, because if the information is not at hand, people use their "best-available-guess" so that they can get on; or the work grinds to a halt until they can get the answers they need.

So, simply adopting a set of agile practices isn't a surefire guarantee of success. You have to pick and choose or tailor the practices to fit your project. I do believe, however, that there's a minimum, core set of agile practices which is suited to perhaps 90% of projects (with a little tailoring, of course).

You also have to look at the economics involved. All things remaining optimistic, XP gets the first release out quicker, but by the third or fourth release the unwieldy design and need for Constant Refactoring After Programming (sic) means an up-front design approach would have paid off better.

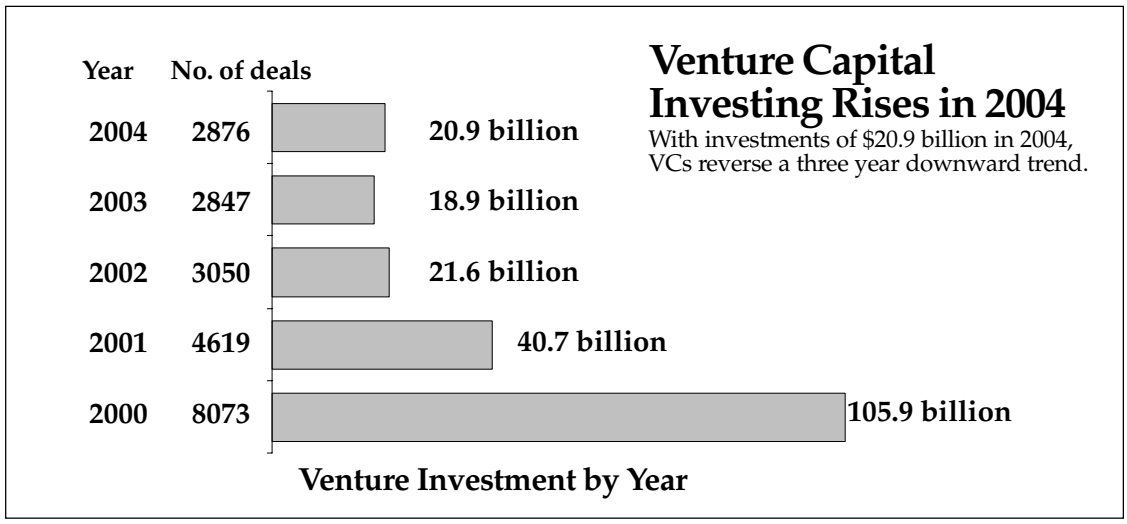
Do you feel some agile processes work better than others?

I seriously believe that if a team follows XP to the letter, then they've just increased the risk of their project failing, not decreased it. XP doesn't place sufficient emphasis on up-front requirements analysis and design. Instead they have catchy sayings like "Big Design Up Front" (BDUF), "Do the Simplest Thing That Can Possibly Work" (DTSTTCPW) and "You Ain't Gonna Need It" (YAGNI) which basically ridicule any attempts to spend a little extra time working on the design prior to coding, or to produce a comprehensive design which looks further ahead than today's programming activities.

It's good that XP has helped to swing the *(continued on page six)*

"The hype around XP in particular has gone too far. Someone in marketing at XP Towers found the Hyperdrive button, and since then we're led to believe that XP can be applied to all sorts of projects (large-scale, mission-critical, life-critical etc.) that it really shouldn't be used for; and that anyone who doesn't agree with the XP way must be either deluded or a social degenerate. So, I'd like to see some toning down of the hype and a more honest and realistic self-appraisal of the applicability of XP."

—Matt Stephens
Software Reality



Benchmarks: Q4 Venture Capital Investments

First-time financings increased over 2003 with a total of 796 companies receiving $4.4 billion, or 28% of all financings in 2004, compared to 699 deals receiving $3.6 billion in the prior year.

In the fourth quarter of 2004, venture capitalists concluded the most active year for VC capital commitments since 2001, reversing a three-year downward trend by investing $20.9 billion into 2,876 deals in 2004. This compares to a six-year low of $18.9 billion into 2,847 deals in 2003, according to the MoneyTree Survey by PricewaterhouseCoopers, Thomson Venture Economics, and the National Venture Capital Association. The increase in 2004 was largely attributable to late stage investments jumping nearly 50% to $7.2 billion in 2004 compared to $4.9 billion in 2003.

A total of 647 later stage deals were completed in 2004 compared to 530 in 2003. Later stage funding in 2004 obtained 34% of all venture capital. This represents the largest dollar amount for later stage funding in three years and the largest percentage in the last 10 years. Average funding per company rose to $11.1 million compared to $9.2 million in 2003. The average post-money valuation rose to $70.2 million for the 12 months ending Q3 2004 compared to $65.4 million for the prior year. (Note that valuation data lags investment data by one quarter.)

Expansion stage investing fell only slightly. In 2004, 1,217 expansion stage deals totaled $9.5 billion, compared to 1,354 deals for $10.3 million in 2003. Average funding per company remained essentially flat $7.8 million. But, the average post-money valuation increased significantly to $57.2 million (for the 12 months ending Q3 2004) versus $39.4 million in the prior year.

Early stage investing increased in 2004; a total of 841 deals accounted for $3.9 billion, or 29% of all deals and 19% of all dollars, compared to 770 deals for $3.4 billion in 2003. In terms of both dollars and deals, early stage investing represented a slightly greater percentage of all venture capital activity in 2004. Average funding per company was $4.6 million, virtually unchanged from 2003. The average post-money valuation of early stage companies increased slightly to $13.9 million for the 12 months ending Q3 2004.

Software companies accounted for the most first-time fundings with 217 companies getting $1.1 billion. In 2004, the software industry hit a three-year high with $5.1 billion invested and remained in the top slot as the largest single industry category. Software represented 24% of all venture capital dollars, in line with historical norms. In terms of number of deals, software was far and away the leader with 862 fundings, or 30% of all venture capital deals for the year.

The Top 50: Software Venture Capital Investments—Q4 2004

	Company	Business Focus	Lead Investor	Investment
1	IronPort Systems	Email infrastructure products and services for the Global 2000	New Enterprise Associates	$45,000,000
2	Linux NetworX	Linux cluster computer applications	Oak Investment Partners	$40,000,000
3	Altair Engineering	Engineering software and computing technologies	General Atlantic Partners	$30,000,000
4	PlanView	IT governance solutions	Apax Partners, Inc.	$25,000,000
5	Visage Mobile	Wireless networking software	Worldview Technology Partners	$22,800,000
6	Riverbed Technology	IT infrastructure and wide area network performance software	Goldman, Sachs & Co.	$20,000,000
7	CareMedic	Healthcare revenue management software	Oak Investment Partners	$19,850,000
8	TradeBeam Holdings	Supply-chain/logistics management software	The Carlyle Group	$18,250,000
9	iRise	Java-based Internet software	Morgan Stanley Venture Partners	$15,800,000
10	FinaPlex	Internet-based tools for the private banking sector	Menlo Ventures; Mobius Venture	$15,655,000
11	Tuvox	Speech recognition software	Norwest Venture Partners	$15,000,000
12	Veveo.tv	Video and multimedia software	Matrix Partners; North Bridge Venture	$14,000,100
13	AlterPoint	Network Management software	AV Labs; Austin Ventures	$12,000,000
14	Groxis	Visual information software technology	Draper Fisher Jurvetson ePlanet Vent	$12,000,000
15	Skybox Security	Corporate network security solutions	Benchmark Capital; Carmel Ventures	$10,100,000
16	Amperion	Networking hardware and software	Aspen Ventures; Global Internet Vent.	$10,000,000
17	Archivas	Archive management and storage software	North Bridge; Polaris Venture Part.	$10,000,000
18	Lumigent Technologies	Enterprise data auditing solutions	Greylock; North Bridge Venture Part.	$10,000,000
19	Ounce Labs	Security solutions software	BlueStream Ventures	$10,000,000
20	Six Apart	Internet web-blogging software and services	August Capital Management	$10,000,000
21	Xaware	XML-based integration software	vSpring Capital	$10,000,000
22	KXEN	Analytical software services	XAnge Capital; Motorola	$9,800,000
23	Bocada	Back-up reporting analysis and billing software	Partech International	$9,500,000
24	T3Ci	Software for the emerging RFID market	Red Rock Ventures; SAP Ventures	$9,400,000
25	Apptran Software	Linux based messaging software	Matrix Partners; Worldview Tech Part.	$9,000,000
26	Fortify Software	Security software	Kleiner Perkins Caufield & Byers	$9,000,000
27	Personeta	Standards-based telecom application servers	Duchossois Technology Partners	$9,000,000
28	IQNavigator	Workforce management software	Baker Capital Corp	$8,700,000
29	Kontiki	Web-base networking software	MK Capital	$8,650,000
30	Applied Identity	Internal security networking software	Bay Partners; Sigma Partners	$8,000,000
31	Pragmatech Software	Web-based sales effectiveness software solutions	Commonwealth Capital Ventures	$8,000,000
32	Sigaba Corporation	Internet security software	Liberty Venture Partners; Symantec	$8,000,000
33	Click Tactics	Outsourced business solutions	TH Lee Putnam Ventures	$7,750,000
34	Bowstreet Software	Software tools for Web Services Automation	IDG Ventures	$7,500,000
35	Guidewire Software	Claims processing software	Bay Partners; U.S. Venture Partners	$7,500,000
36	Meiosys	Information technology systems software	Credit Lyonnais Private Equity	$7,500,000
37	InMage Systems	Information management solutions for enterprise data centers	Hummer Winblad Venture Partners	$7,300,000
38	Cyota	Secure financial internet transaction software	Bessemer Venture Partners	$7,250,000
39	Solsoft Inc	Software solution for network access policy implementation	The Carlyle Group; Credit Lyonnais	$7,000,000
40	Emagia Corporation	Collaborative e-business software solutions	Sigma Partners; Timeline Ventures	$7,000,000
41	IntelliReach	Security/compliance mgt. solutions for corporate messaging	M/C Venture Partners	$7,000,000
42	NSI Software	Data replication technologies and services	ABS Capital Partners	$7,000,000
43	Centrify Corporation	Extends function of various Microsoft software services.	Accel Partners; Mayfield Fund	$6,800,000
44	Optinuity	Software automation/management of complex IT procedures	New Enterprise Associates	$6,500,000
45	NTAG Interactive	Software applications for networking and event management	Pilot House Ventures Group	$6,500,000
46	Frictionless Commerce	Enterprise-wide sourcing software	Polaris Venture Partners	$6,200,000
47	Active Decisions	Electronic CRM decision-support applications	Liberty Partners	$6,000,100
48	Roving Planet	Proximity-based wireless technology for the travel industry	StarVest Partners	$6,000,000
49	Silicon Navigator	Electronic design automation software	ITU Ventures; Gefinor Ventures	$6,000,000
50	SugarCRM	Open source customer relationship management applications	Draper Fisher Jurvetson	$5,750,000

pendulum away from the "big monolithic project" (aka waterfall, big-bang integration, and big over-designed frameworks) but the pendulum has swung too far.

Agile programming has been presented as the answer to the following problems: poor communications, incomplete and/or unclear requirements, scope issues, insufficient or inadequate testing, and integration. What aspects of these problems are best addressed by current agile methodologies? Which are aspects are least addressed?

"Agile methods (and XP in particular) shook up an industry. The main problem now is that things have gone too far in the other direction. Teams are being encouraged to "just start programming" and to design the product as they go along. This inevitably stores up problems, because the software must be redesigned to fit whatever new feature has suddenly been discovered.

The trouble is, the team won't see this as a problem because they're deeply immersed in the project, and they see the required changes simply as refactoring, a necessary part of the process."

—Matt Stephens
Software Reality

That's a big question. XP attempts to address all these issues, but the result is a high-risk methodology which propagates unclear requirements by leaving key questions unanswered until late in the project, and demanding that all the project stakeholders "speak with a single voice" whilst simultaneously downplaying the role of written requirements specifications.

What XP does do a good job of is increasing the focus on software testing, by making the programmers write unit tests as they go along (this doesn't rule out additional QA testing, of course). As we discuss in our ICONIX book, it's possible to extract this aspect of XP and apply it to more rigorous agile methodologies.

Use cases, popularized by Alistair Cockburn, have been presented as the answer to the problem of incomplete or poor requirements. What is your opinion?

I'm a big fan of use cases for defining behavioral requirements, but the definition of a use case has become somewhat hazy and ambiguous. Read a dozen books on use cases and each one will tell you to write your use cases in a different way. They range from "lightweight" approaches (such as the ICONIX process) to the big, heavyweight template containing preconditions and post-conditions, etc., which usually results in 10 or 20-page use cases including a random mixture of behavioral and functional requirements.

Use cases work best when each one is no more than two paragraphs (including the basic and alternative courses). Also, they should be written in the active voice and as user action/system response "couplets" ("The user does this; the system responds by doing that"). You need to tie your use cases to your objects. Another way to put this is that you write the use cases in the context of your domain model, and then drive the design directly from the use cases.

So, if you write your use cases using these criteria, and make sure you capture the functional requirements separately then yes, use cases are a good answer to the problem of how to capture all of the requirements, and also how to "disambiguate" the requirements (i.e., cut out all the misunderstandings) as early in the project lifecycle as possible.

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.

How to Detect a Non-Negotiating Buyer

By Mark Reed, Corum Group

When you're in discussions about an acquisition of your company, be alert for signs the deal is not "real." Some "buyers" may engage in talks just to learn more about your business, technology or customer base; others may simply not execute an M&A process very well. In either case, the cost to your company can be significant in terms of the drain on management resources, loss of business momentum, and disclosure of confidential information.

You can protect yourself against competitive snooping and tire kicking as you enter into discussions by:

- Discovering early in the process the buyer's motivations for the deal, especially if you are dealing with a competitor. Be sure you have an adequate non-disclosure agreement in place to protect confidential information. Ask the buyer about their internal process for acquisitions and make sure senior management is informed and committed to the process..

- Detecting early failures to define terms and execute against an agreed timeline. Be alert for endless meetings, the involvment of new parties who need to be brought up to speed, and information requests that lead to yet more information requests.

- Refusing to provide highly sensitive information such as source code reviews or contacts with key customers unless a firm offer is on the table.

- Being alert to a buyer's refusal to meet milestones or deliver documents on a timely basis. These can include long delays in making a firm offer or in delivering drafts of definitive agreements after an offer is accepted. Protect against this by setting up increasingly specific milestones with the buyer so that you and they have common expectations about how the process should unfold.

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

Company/Description	Acquired by	Price/Terms	Revenues	Multiple
CGI Group's credit union division *• Provides core data processing for credit unions*	Open Solutions	$24,000,000 *Terms: All cash*	$16,000,000	**1.50**
Blue Martini *• Provider of sales optimization systems*	Golden Gate Capital	$21,100,000 *Terms: All cash*	$28,300,000	**0.74**
Financial Models *• Investment management technolgy*	SS&C Technologies	$132,000,000 *Terms: All cash*	$61,100,000	**2.16**
Symfonia *• Accounting solutions for SMB marketing in Poland*	Sage Group	$19,700,000 *Terms: All cash*	$8,400,000	**2.34**

CORUM
MERGERS & ACQUISITIONS

CardFile

Pod Casting Software

- **Doppler Radio** (www.dopplerradio.net): Podcast aggregator that subscribes to RSS feeds. Doppler retrieves the files enclosed in the RSS feed and automatically adds them to a media player.
- **Gold Wave** (www.goldwave.com): Inexpensive but powerful digital audio editor. (We strongly recommend investing in such a program if you are serious about creating your own podcasts; even very high-quality recordings will often require "touch up" before being posted on a website.)
- **Ipodder** (http://ipodder.sourceforge.net/index.php): Open Source software for Windows/Mac enables allows you to create your own customized list of podcasts from different websites.
- **MT-Enclosures** (http://brandon.fuller.name/archives/hacks/mtenclosures/): Plug-in for Movable Type blogging software adds the ability to manage audio files.
- **Xpodder** (http://xpodder.sourceforge.net/): Python-based aggregator that can be used with podcasts.

GARTNER GROUP ANALYST NEIL MCDONALD on Microsoft's new security software: "Microsoft's overriding goal should be to eliminate the need for anti-virus and anti-spyware products, not simply to enter the market with look-alike products at lower prices." (Quoted in Information Week, 02/22/2005)

DONALD CAMERON, co-chairman of the Toronto Intellectual Property Group, on Canadian business and patents: "Patenting your product is a form of business insurance. If companies don't patent a new and inventive way of doing something more profitably, then you might get into trouble with it." (Quoted in The Globe and Mail, 02/24/2005)

REPORTER BETH STACKPOLE on product innovation: "For every 3,000 raw ideas, only one successful product makes its way to market. Others say nearly two-thirds of new products fail within two years. And a recent survey by the Product Development Management Association (PDMA) reports that 67% of industrial firms are not satisfied with the success of new product launches." (Quoted in Managing Automation, 12/07/2004)

ANALYST ROB ENDERLE on Open Source software reliability: "Many people believe that open-source software such as Firefox is naturally more reliable than proprietary software. They assume that when a large number of people are able to look at the code, they're also able to ensure the quality of the product. These people have clearly not read *The Tipping Point* or done any real software quality assurance work." (Quoted on Linuxpipeline, 03/07/2005)

MICROSOFT DISTINGUISHED ENGINEER Mark Lucovsky on Microsoft: "I am not sure I believe anymore, that Microsoft 'knows how to ship software'." (Quoted on http://mark-lucovsky.blogspot.com/2005/02/shipping-software.html)

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