# Mass hysteria and the delusion of crowds

**Michael Kenny**
itopia
Technoparkstr 1
8005 Zurich
Switzerland
kenny@acm.org

**ABSTRACT**

It is easy to forget why lightweight Methods have emerged to tackle today's software projects. This paper retraces the journey that has given us XP and other new methods and offers some suggestions for the further adaptive development of these methods in the broader context of management strategies for the Internet era.

**Keywords**

**Adaptation, Attractors, Chaos, Commodification, Design, Emergence, Evolution, Landscape theory, Observation, Pattern switching, Planning. Systems theory.**

## 1 GIVE ME PLANNING OR GIVE ME …

Fowler [1] says planned design has being around since the 70s. In those times software could be predicted because changes demanded by bricks and mortar industry occurred at very slow rates (e.g. Highsmith "balances" [2]). Software was rolled out after many years in development and, if it survived the launch at all, was expected to live for many years (this justified huge development cost), to be replaced by new systems, which were only marginally different from the original. In this culture of simple repetition there was an understandable need for more complexity to keep minds active. Observed repeatability fostered the assembly of abstract models, transcendence, and methodology followed at some later stage by ever more elaborate designs (with the future built in) and a deep desire for components.

Methods retrofitted to any culture have a tendency to reinforce that culture until what is observed no longer matches what is predicted. Sometimes the apparatus of observation falsifies what is seen. Methods are a statement of belief in the predictability of things.

## 2 THE EMPEROR'S CLOTHES

The nice thing about prediction and linearity is that it pleasantly supports hierarchies in organizations. It fosters division in skills between those who design and those who build.

For a long time nothing was said. For many there were years of doubt. In the face of a huge body of evidence perhaps it was the observations that were at fault, the intuition that was open to question. These methods were based on so much experience it discounted your own.

With the dawn of the Internet era a number of things happened. Observations were more easily compared and traded worldwide. Traditional channels based on local proprietary hardware and software networks were suddenly open to global inspection and competition. And the infrastructure and even the very topography of the net itself: anti-genealogical, non-linear, and rhizomorphic[10] as it is, indicated new models for distributed networked collaborative creativity.

With this transparency it suddenly became apparent that what was not understood was that the software method sales-folks paraded on the conference podiums of the world were stuck in a groove doing something rather simple and dressing it up. If you had been around in the 70s you would have known. The emperor was as naked as naked could be.

## 3 SOME PLANNING HISTORY

Planning of course was with us before the 70s. With Copernicus and the predictions of the trajectories of planets etc. modern science brought accurate prediction into fashion. The industrial age viewed the heavens as a piece of clockwork, and relied in turn on this very predictability to build the unprecedented machine hierarchies of Empire.

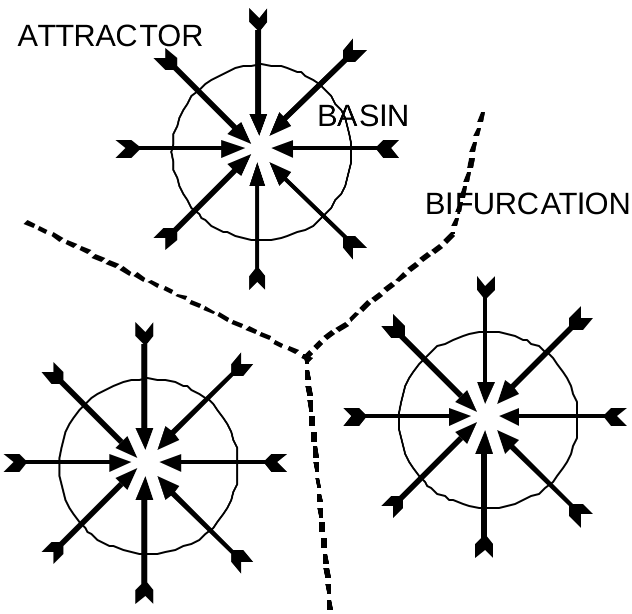
Today, in the age of one-to-one marketing just as we become less and less predictable ourselves, we seemingly demand and expect increasing predictability in our every-day lives. If something is unknown the next billion-dollar research program will reveal the math of tomorrow's weather or electron trajectories. The unspoken hope is that God is in the numbers.

And yet it now appears that the math delivered at every turn throws up more questions than it answers.

Of all the bad breaks trust the software development industry to adopt a scientific model just as it was going out of fashion.

## 4 STRANGE ATTRACTORS

The issue of predictability is nicely illustrated by the theory of attractors and basins [3]. In simple dynamical systems all states tend to converge to a single equilibrium state. In thermodynamics this is the state with maximum entropy. Such a state is known as an attractor.




A system with 3 Attractors. Along the bifurcation line systems can go one way or the other causing a pattern switch. With 1 Attractor prediction of where the system will go is trivial. With an infinite number of Attractors it is impossible.

As energy or flow is increased to the system the number of attractors increase. The example often used is that of a water tap opened to various degrees. When water runs slowly through the tap a predictable dripping or thin uniform stream of water is produced. Small fluctuations in pressure lead from one pattern to the other and back again. As the tap is opened further pattern switches occur due to the increasing amount of attractors until finally with an infinite number of attractors the system switches erratically and becomes truly chaotic and unpredictable.

## 5 MARKET SURVIVAL

Prior to the Internet era the survival of bricks and mortar corporations depended in many ways on the maintenance of restrictive practices. Markets rarely changed, physical sales channels secured oligarchic positions. It was easy to make out attractors. You could get away with building monuments to yesterday's functionality since your audience had few alternatives.

By contrast markets these days are moving so fast that survival handbooks become obsolete before the ink is dry. The accelerated commodification of goods and services in addition to the simple and cheap replication of the business models of others lead to a constant redefinition and fragmentation of markets in the endless search for the next high-margin differentiated products and services.

It is as if the markets and our lives have been suddenly injected with new energy. The number of attractors we as individuals are exposed to is an example of these new accelerated lifestyles (see e.g. [4]).

The position of software in all of this acceleration and abundance of attractors and constant pattern switches is central. It is no longer the unwanted stepchild of the old economy but has suddenly been transformed into the star of the show. It is the key adder of value. It makes the show happen. It is the show. We make the new attractors, which drive the search for the next ones, which we again will make. No longer only developers we are the expert users that drive innovation.

At the same time we are subject ourselves to the push and pull of competing attractors, as energy is pumped into the system it becomes hard to know which if any pattern to rely on. The choices at every turn are increasing. Which features should be introduced? Which features left out to avoid featuritis? Which technology should be followed? How can you differentiate yourself from the competition with standard software, with components? Which way will the market move? Which software development process to use? Which practices to emphasize therein? ….

It has been said that, "trade is largely driven by experienced guesswork and simple rules, when not driven by confusion, error, and nonsense, supplemented by (as one financial journalist puts it) 'testosterone and cocaine.' " [5].

In short it is impossible to make rational decisions. And prediction only works for simple systems. We could all sure use some help.

## 6 BIG M PREDICTION AND OPTIMISATION

Old fangled "Big Method planned design" seeks to box complexity with its relentless use of decomposition "to bring order to chaos"[6]. It tries to know what to build from the beginning and pass it on down the production line of phases to the big launch. It seems often to be discipline for the sake of it. Innovation and creativity, the lifeblood of corporations in competitive markets, are shackled by "bureaucratic overload" [7], by "being in the wrong phase for that". According to role, permitted thought-space is arbitrarily delineated. Innovative ideas are accepted only if they come from people with the correct roles, e.g. designers or even board members unable to use the Internet. Implementers have no personal stake in the ideas. This model of stark roles betrays a lack of trust and will only survive in markets based on restrictive practices and captive audiences. Markets reflect their suppliers.

It is as if the deterministic world of computers has been mapped to the market and to employees [see e.g. 9]. Indeed according to the Chicago school the market is deterministic. As long as this is assumed, as long as you can make your markets linear, a command-control structure for software development seems valid.

Even so since the Fordian model of treating men as machines was already discredited in the 1930s the glaring question is why was it embraced by the software industry? The answer is because it was simple to do so. The metaphor is easy to grasp and the strategy seemed to work.

This established strategy to secure value added margins is based on optimisation. Existing products are made faster, features are added, business-processes are re-engineered etc. This approach is always backward looking since it is based on optimising what one already has. It is served and indeed driven by linear hierarchies of competent technologists. Technology dictates what is made available not the consumer. The trajectory of incremental improvement can be plotted according to Moore's law.

## 7 GOING NON LINEAR

The emergence of the Internet has highlighted the fact that the model of deterministic markets was at most a convenient delusion (the delusion of this paper's title), and further has increased pressure in the system and made markets more unpredictable.

Hamel[11] notes "the economy is experiencing a competition between 2 regimes. The hierarchy that serves to protect status and jobs is losing out to the market as the regulation principle", and further, "companies pretend linear strategies can sustain them in a non-linear world".

Optimisation as a corporate strategy no longer secures the highest margins. Because it is the simplest and most copyable path to follow. Also as markets become more transparent and consumer driven features people never use and product complexity threaten sales. Technology alone is no longer the key.

Recognising that old strategies (and their Big Methods) only really work for a handful of special cases, and based on observations of dot.com start-ups etc. a number of things have happened. In the case of software development lightweight methods have emerged to tackle the reality of the increased fragmentation, transparency, changeability and unpredictability of the market. And management strategies have been reformulated for the same reasons.

Moving away from the industrial metaphor, we are now asked to embrace a biological one and treat corporations as living organisms. Here structure is not imposed from above, that would require predictability to be in place. Instead, according to the new theory of complexity, a self-organising structure continually emerges to tackle problems against the backdrop of non-linear change.

It is this emergent self-organising structure in the transition zone between linear and non-linear systems which one seeks to foster with the latest software methods and management strategies.

Now we don't have to pretend the world is linear to get things done. Nor need we throw up or hands and give up in the face of chaos.

## 8 SMALL M AND TRANSITION ZONE SURFING

The transition zone is where as Highsmith[12] observes, "there is enough control to keep from spinning off into chaos, yet enough spontaneity for creativity and enough innovation to enable it to adapt to changing environments".

In XP self-organisation is focused around a framework of practices (supported by values including *communication* and *feedback*) which taken together allow one to keep the XP car rolling in the transition zone and leads to very changeable software and emergent design. Command-control hierarchies are converted to distributed networks of collaboration.

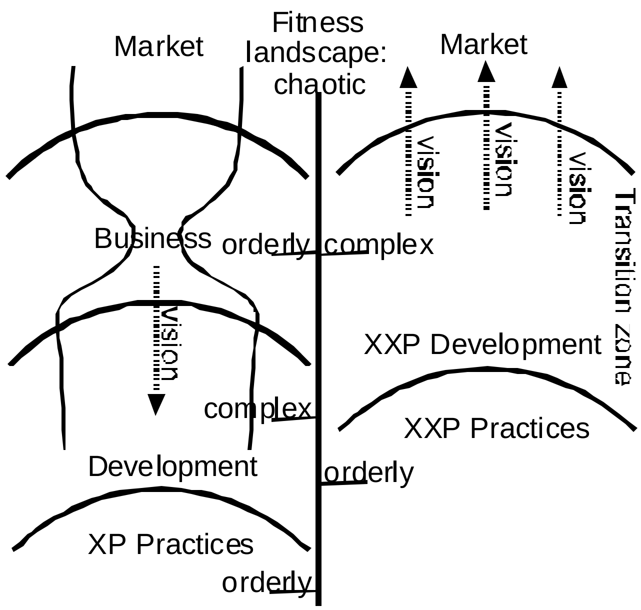
XP is about communicating, networking, distributing knowledge through the team. No attempt at prediction is made. Roles are distributed amongst the team too. People are dignified by being placed back in control of what they do and through such practices as the Planning Game they can bring their influence and innovative ideas to bear. Software writing for corporations has been re-humanised. The optimising instinct has been tamed.

## 9 XXP AND THE END OF MANAGEMENT

Applying XP to teams of technologists can produce technological emergence. The "*how*" is emergent. The question is, is this enough? If the rest of the organisation (or customer) is still command-control driven, if "*what*" is to be done is decided from above, as a software technologist your market remains linear. And practicing XP might be considered a luxury.

It is interesting to note that the XP practices diagram includes just one explicit role – that of the on-line customer. Unlike all other roles, this role is not distributed into the team.

Yet it is exactly this kind of distribution that according to Hamel is required in the Internet era. In moving from technology and optimisation strategies to ones based on creativity, innovation and surprise Hamel suggests that "Innovation is too important to be left to the visionaries", and further, "it should be everybody's responsibility, systematically fomented throughout the workforce". Hamel's theories are based on observations of Silicon Valley. Viewing the whole valley as an organism one sees companies vying for capital from VCs. Instead of resource allocation from above, there is resource attraction from below. Here visions too come from below; they are in fact emergent.




By distributing diverse roles into the XP team (which now no longer just programs) you take it closer to the market, to the transition zone where it makes sense. Vision is freed from the business bottleneck.

With a market regime and resource attraction employees move from being change embracers to becoming change agents. Companies can get to the future before the competition. Instead of reacting to fitness-landscape changes they set out to redefine the landscape itself.

With self-organisation in place, what is the need for management?

## 10 WHOSE VALUE SYSTEM IS IT ANYWAY?

The way in which processes are followed depends on value systems; of the process, of the culture in which it is practised and of the individuals concerned. Those who seek to enforce XP in very structured ways tend themselves to lead very structured lives. The danger here is that XP becomes Big M and emergence is threatened.

Those more ambivalent to societal programming are the innovative change agents companies need. But to prevent them spinning of into chaos they need their more structured team colleagues. We must endeavour to embrace team diversity.

## 11 CONCLUSION

The preface of "Extreme programming explained" dangerously asks you to make your own XP [8].

Looking at the diagram above you have to decide which conditions apply to you. And then where it is you want to go, which market you seek to define. Are you making the *n*th Web Search Engine just as the market for search engines collapses, or the next new thing? You might for example decide to extend or reduce the XP framework of practices to support distribution of more roles into the team. You might not.

The reason for this (and this is the only real generalisation one can make about software development) is that the conditions under which software is being developed are never the same twice. A single right way of developing software was a fashionable delusion widely believed in the 1990s.

Instead of proceeding towards a single attractor we are now aware of countless attractors. Picture constantly reinventing (in true Professor Pat Pending style) and guiding your XP surfboard through a terrain of increasing/decreasing attractors, and riding on the pattern switches. "Man!"

## ACKNOWLEDGEMENTS

Thanks to Kent Beck, Mel Kenny, Roger Leuenberger, Peter Sommerlad, Frank Westphal and the anonymous reviewers for their helpful comments.

## REFERENCES

1. Fowler, M.. "Is Design Dead". On-line at. http://www.martinfowler.com/articles/designDead.html
2. Jim Highsmith, "Extreme Programming". On-line at http://www.cutter.com/ead/ead0002.pdf.
3. Attractors, Bifurcation etc. feature in the work of Henri Pioncare and Robert May, featured online at many sites, e.g. http://www-chaos.umd.edu/research.html and http://members.nbci.com/Templarser/beffect.html .
4. George Ritzer, "The McDonaldization of Society", Pine Forge Press, 2000. ISBN: 0 761 98628 6
5. Cosma Shalizi, "Why Markets Aren't Rational but Are Efficient". Santa-Fe Institute Bulletin, On-line at http://www.santafe.edu/sfi/publications/Bulletins/bulletinSpring00/index.html
6. Grady Booch, "Bringing order to Chaos", Object-oriented Analysis and Design. 1994. ISBN. 0 80 53534 02 pp 16-21
7. A. Cockburn, "What is not suited" in "Surviving object oriented projects", Addison Wesley 1998. ISBN 0 201 49834 0
8. Kent Beck, "Extreme programming explained" - Addison Wesley, 2000. ISBN 0 201 61641 6
9. L.J. Osterweil „Software processes are software too". In ICSE 9 Conference Proceedings. IEEE Press 1987.
10. Deleuze, Guattari „A thousand plateaus" Capitalism and schizophrenia, University of Minnesota Press 1987 ISBN 0-8166-1406-4
11. Gary Hamel, "Leading the Revolution", Harvard Business School Pr, 2000. ISBN 1578511895
12. Highsmith "Adaptive Software Development", Dorset House, 2000. ISBN 0932633404