Falling Blossoms

Questionnaire: Just how good are you – as a business – at developing software-intensive products and services?

Why a Questionnaire? Well, we hope that by inviting you to consider certain observable behaviours - which we call our “tell-tales” - you might come to a more objective recognition of the effectiveness of your software and product development efforts, and also begin to discern your business's place in the Rightshifting spectrum from “highly competent”, through e.g. “average”, to “totally dysfunctional”.

So, please reflect on each of the following questions in turn and indicate which one of the five provided answers most closely matches your understanding of the current situation in your own business. Don't worry if you're not able to answer some of the questions. You might like to go ask someone, or failing that, just leave the answer blank.

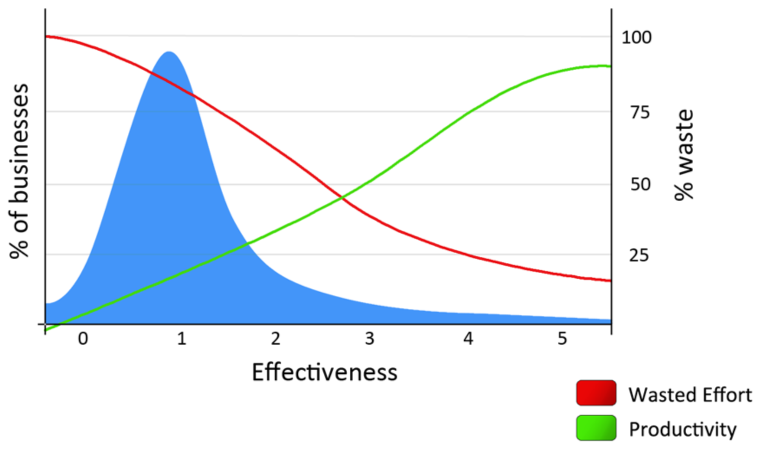
Note: Your reflections on the subject matter of each question are at least as important as the answers you come to choose.

Please try to be as scrupulously honest with yourself as you can bear.

q1: as you start this questionnaire, where do you think your business sits in the rightshifting spectrum of effectiveness?

Answers:

1. Way to the left – somewhere below x0.5 on horizontal axis of the chart, below.
2. Around the modal average – near to x1.
3. Above average – somewhere between x2 and x3.
4. Highly effective – to the right of x3.
5. Don't know.



q2: how long does it typically take in your development organisation for a bug to be spotted and fixed?

Answers:

1. We just don’t have bugs!
2. We catch most bugs the same day and fix them straight away.
3. We catch bugs at most a week or two after they’re created.
4. We find our bugs during integration testing at the end of a project and fix them then.
5. We don’t bother much looking for bugs – we let other people / our customers find them!

q3: how do your customers typically react when your development organisation delivers stuff to them?

Answers:

1. Wow! That’s just what we need!
2. Hey! That’s just what we asked for.
3. Ah. Um. That’s not quite what I was expecting.
4. Oh dear. I was hoping for something rather different.
5. Geez. What are we supposed to do with this turkey?

q4: what kind of overall response do your development organisation’s efforts typically receive?

Answers:

1. Everyone agrees that the project and its results have been amazing.
2. The customer is delighted but your own folks heave a sigh of relief that it’s over.
3. The users love it but their management doesn’t want to take it any further.
4. The customer’s management love it but nobody’s using it much.
5. You never want another project to turn out like this one did.

q5: which of these best reflects your internal or external customers’ response to your “sales closes”?

Answers:

1. Here’s the purchase order. Can you deliver the project yesterday?
2. When do you want the purchase order?
3. How much will it cost?
4. Can we have it delivered in stages?
5. That’s too expensive. Can we have it cheaper?

q6: how often does your development organisation meet its project timetables?

Answers:

1. All our projects hit their delivery dates right on the nose.
2. Sometimes we deliver late and sometimes early.
3. We deliver things on schedule some of the time, but equally often things are somewhat late.
4. We almost always deliver at least a couple of weeks behind schedule.
5. Things always ship months or years late, and sometimes not at all.

q7: how do things improve within your development organisation?

Answers:

1. We continuously review how well we're doing and act every few days to improve the way we go about doing things.
2. We regularly review how well we're doing and where we could be doing better.
3. We're much too busy to spend much time on improvement, but we'll always look at suggestions on better ways to tackle a particular task or issue.
4. Things never improve round here. It's just the same old grind day in day out.
5. Improve? You must be joking. Things just go from bad to worse.

q8: when there's a problem, change of direction, urgent issue, what typically happens?

Answers:

1. Everyone – our own folks, our customer's people, our suppliers - pitches in right away to understand the issue and get it resolved.
2. It's all hands to the pumps! Our folks drop everything and work overtime to get the issue resolved with as little disruption to the customer as possible.
3. We get together with the customer to understand the criticality of the issue and schedule a resolution or fix for the project team to implement as soon as they can.
4. We log the issue and get round to it as soon as we can.
5. Not much. Except some CYA, maybe.

q9: who takes responsibility for the way the work works?

Answers:

1. Everybody has taken some actions to address the way work works.
2. Team leaders and project managers periodically consult with folks to see how the work is working, identify areas for change, and ask people for suggestions on how to improve things.
3. Nobody has any assigned responsibility for the way the work works – sometimes somebody or other will get energised about a particular issue and try to resolve it.
4. Our immediate management has ownership of the processes. Any change initiatives come from the stroke of their pen.
5. We have a special process group that owns the way the work works – they design the processes, train people in how to use them and monitor subsequent compliance.

q10: what kind of attention does software / product development get from senior management?

Answers:

1. Our senior management:
   - ensures everyone in the organisation has a clear, quantified understanding of the objectives for both product development, organisational performance and organisational capability.
   - works with our immediate management to balance the flow of work across and through the whole organisation.
   - aligns the organisation's structure and systems to best serve those objectives.
   - regularly reviews results against those objectives.
   - holds folks accountable when results fall short.
2. We have lots of meetings with our senior management to set priorities for and review progress of each specific project.
3. Our senior management pays little attention to what's happing in product development - except when things go wrong, and then they parachute-in to tell us in excruciating detail how to do our jobs before moving on to something else before seeing any results of their intervention.
4. Our senior management have faith in our ability to manage ourselves and just leaves us alone to get on with the job at hand.
5. Our senior management is conspicuous by its absence – I guess they're spending their time fire-fighting in other areas of the business that are even more broken than product development.

q11: what does your development organisation typically set out to deliver – i.e. what typically comprises a product or service, as delivered?

Answers:

1. We generally plan to deliver a number of iterations of a "whole product", progressively adding functionality at the direction of the stakeholders. Typically our deliverables include the core software, ancillary hardware, documentation, people trained in supporting the product in operation, a help-desk, training materials, pre-delivery advice on the evolution of the product, and follow-up to ensure the product is delivering as much value as possible to all stakeholders.
2. We generally plan to deliver a complete turnkey service, including the core product as well as the infrastructure and staff needed to run the service, including servers, procedures, people and management.
3. We usually plan to deliver a working software package in the form of e.g. an installation CD, along with the necessary installation documentation, user manual, help files. etc., and a 90-day warranty.
4. We usually plan to deliver a working software package in the form of e.g. an installation CD.
5. We just set out to deliver code (or web pages, or whatever) – what the customer does with it ain't our concern.

q12: what risks do your projects manage on behalf of your stakeholders?

Answers:

1. Our projects identify, assign ownership of, and track, all business opportunities and associate risks - on behalf of all the stakeholders for every project. We specify these responsibilities in our contract terms and offer a money-back guarantee to reassure the customer that we're really paying attention.
2. We generally manage those project risks within our control in a structured way, but leave the customer and other stakeholders to manage as best they can those risks outside our immediate control.
3. We try our best to do a good technical job for the customer and other stakeholders – that should take care of any major risks, shouldn't it?
4. We pay some attention to our own risks, the customer and other stakeholders have to look out for themselves – Caveat emptor!
5. Risk? What's that? We don't worry about things that could go wrong – life's too short and attending to these kinds of thing would be very dull!

q13: what does your development organisation do to identify and reduce waste?

Answers:

1. We analyse the constraints in our stakeholders' organisations and design multi-dimensional solutions to simply and directly address the elevation of these constraints, thus delivering maximum stakeholder value for minimum cost (waste).
2. We map our stakeholders' perception of value, and tailor all our work to simply meeting this perception of value.
3. We carefully look to see where we're spending our efforts and continually strive to reduce or eliminate low value-add activities.
4. Not much.
5. Waste? People here go out of their way to make extra needless work, over-design things, specify irrelevant detail, inject more bugs, have more and more pointless meetings, keep people waiting for information, and generally cause waste!

q14: what level of utilisation (efficiencies) does your senior management demand of development staff?

Answers:

1. Our senior management understands queuing theory, and encourages a certain level of "slack", eschewing utilisation targets ("efficiencies") in favour of global systems measures like overall throughput, total operating expenses and inventory.
2. We don't have any stated utilisation levels nor efficiency targets - we're trusted to do our best to get the work done as quickly and efficiently as possible.
3. We don't have any specific utilisation levels or efficiency targets, but our managers expect people to look busy at all times and woe betide any apparent slackers.
4. Our senior management demands a high level (70%-95%) utilisation of e.g. development folks.
5. Our senior management demands that we're busy all the time (100%+ utilisation of development folks).

q15: which measures (metrics) does your development organisation use to assess the health of its projects?

Answers:

1. We measure process improvement – in particular, reduction and elimination of waste – in the sure and certain knowledge that if these indicators are moving in the right direction then the businesses financial indicators will be moving in the right direction too.
2. We use Throughput Accounting measures, as well as technical project measures like velocity, cumulative flow and due date performance.
3. We use typical modern technical project measures like velocity, cumulative flow and due date performance.
4. We don't use any specific measures – we just kind of 'know' which projects are healthy.
5. We use Cost Accounting, and maybe Function Point counting and revenues to judge our projects' health. That is, we assign a cost to each activity or task in a project and sum these costs to see the cost of the project as a whole (and compare this sum to the billings to arrive at the project's profit/loss and thus it's health).

q16: what does the developers' physical environment look most like?

Answers:

1. Individual private offices, Aeron chairs, multi-LCD monitors, separate build and test servers, Gigabit Ethernet LAN, free books, tools upon request, lavish storage, personal lockers, private lounge areas.
2. Shared private offices (two or three developers per office), state-of-the-art workstations, large monitors, technical library, communal lounge area.
3. Cube farms, 15" CRTs, ageing PCs.
4. Noisy open plan offices, or developers use laptops rather than workstations or PCs.
5. Hot-desking.

q17: how do decisions typically get made within your development organisation?

Answers:

1. We choose to decide issues at the last possible responsible moment, on the merits of the argument in question (i.e. on a purely rational basis).
2. We try to be rational but the folks with strong personalities often seem to win arguments on the force of their rhetoric.
3. Decisions typically get made by cliques and cabals comprise of a limited circle of politically-advantaged people.
4. Decisions typically get made in haste at the very last moment, with little deliberation or discussion, by whoever happens to be in the office at the time.
5. Decisions mostly get made by opaque means, and typically appear to be management fiat or whimsy.

q18: to what extent do your development folks identify with the prevailing values and ethics of the wider business?

Answers:

1. Our people feel a great pride in the purpose and work of the business, and each actively subscribes to where we're going as a whole business. Folks bring passion to their work and are always fully engaged emotionally and practically in whatever they're doing.
2. Our people often feel pride in the purpose and work of the business. Folks enjoy their work and feel they have a stake in the business.
3. People enjoy working for the company but wouldn't necessarily recommend it as a good place to work to their friends.
4. People turn up each day and do the job that's expected of them with little ceremony or complaint.
5. People generally feel embarrassed to admit they work for the company, regularly find themselves cringing at our ineptitude and apologising to customers for poor service, and are mostly checked-out (disengaged) and looking to leave for pastures new.

q19: how does your development organisation go about setting pay scales, contract rates and related compensation levels?

Answers:

1. We have mechanisms that ensure that everyone gets paid what they as individuals – and to some extent everyone else – consider fair and just.
2. We pay above the market rate because we realise that better-than-average people are well worth better-than-average rates.
3. We base compensation levels on prevailing market rates for the job.
4. We pitch compensation levels a little below market rates, and try to snow people that this is such a great place to work they shouldn't expect great pay rates too.
5. We try to pay as little as possible, and then do all we can to keep a lid on the ensuing bad feeling this inevitably generates.

q20: how does your development organisation apply generic HR policies?

Answers:

1. We all realise that technical folks have significantly different needs and parameters for job satisfaction than folks in other areas of the business, and have worked extensively with our Human Resources people to help them understand these differences, and together amended the general HR mechanisms and policies to best suit our special needs in respect of recruiting, retaining and motivating our technical people.
2. We have managed to convince our Human Resources department that technical folks have special needs and have obtained some special dispensations and opt-outs from the generic HR policies.
3. We try to work around the obstacles that generic Human Resources policies put in our path in respect of recruiting, retaining and motivating our technical people. The HR folks in turn are smart people who can see the benefits and generally turn a blind eye to our contraventions despite senior management edicts on the need for conformity.
4. We try to work around the obstacles that generic Human Resources policies put in our path in respect of recruiting, retaining and motivating our technical people, but the HR folks don't like it and try to rein us in whenever they can.
5. We've given up fighting the system and simply conform to the prevailing generic Human Resources policies, despite the significant deleterious effects they have on recruiting, retaining and motivating our technical people.

q21: if some hiccup happens during the course of a piece of work, what typically happens?

Answers:

1. The people involved stop what they're doing and examine the situation to discover the root cause of the problem. They then set-to to fix the root cause, recording and publishing the incident for future reference, before picking-up from where they left off.
2. People take a look at the problem as soon as they finish what they're doing, discover the root cause of the problem and set-to to fix it.
3. People put aside what they're doing, and progress various other things that fall to hand, until someone fixes the problem and they can get back to what they were doing in the first place.
4. People put aside what they're doing, and sit around twiddling their thumbs waiting for someone (like a supervisor or manager) to fix the problem before they can get going again.
5. People try to disguise the problem and pretend it's not happening - in case they get blamed for it or roped into doing something about it.

q22: how would you characterise the team dynamic of typical projects within your development organisation?

Answers:

1. People in our teams trust one another implicitly, challenge each other's assumptions, call each other on poor performance, collaborate to establish and then achieve common goals, debate issues robustly, confront difficult issues, commit themselves to outcomes, and hold one another directly accountable for results.
2. People in our teams trust one another implicitly, challenge each other's assumptions, call each other on poor performance, collaborate to establish and then achieve common goals, debate issues robustly, confront difficult issues, but sometimes pay little or no attention to actual the outcomes (results) of their work.
3. People in our teams generally trust one another and have established a climate of healthy debate, but often fail to force clarity and closure, preferring to avoid sticking their heads above the parapet.
4. People basically trust one another to work for the common good, but avoid exposing unpleasant truths for fear of making waves.
5. People have no common bond of trust, lack basic humanity, and spend most of their time defending their turf, shifting blame and covering their arses.

q23: when faced with a choice on e.g. a technical issue, how do people generally decide between the various options?

Answers:

1. People always choose to "do the simplest thing that could possibly work".
2. People consider the various options and use their combined skill and experience to choose the most reasonable option.
3. People seek out an expert on the question at hand and let them decide.
4. People generally choose the most difficult, involved or complex option because it'll be fun, or to advertise how brainy they are.
5. Every decision requires an options harvest, impact analysis, business justification, and sign off from three different managers.

q24: in your development organisation, how do teams – for example project teams - typically get formed, re-formed and disbanded?

Answers:

1. People form and re-form themselves as needs dictate.
2. People decide how they need to be structured to get a job done and then get management to sanction and enact the necessary organisational changes.
3. Management look for volunteers and assign them to teams as required.
4. Management assign and reassign people to and from teams as they see fit.
5. Teams are put together, rearranged and broken apart by some outside unit (for example, Human Resources) using the pool of people who are not totally busy at present.

q25: to what extent do people in your development organisation get together to: expand their capacity to create results? nurture new perspectives and ways of thinking? learn to see more of the whole?

Answers:

1. People are constantly engaged with each other, talking and exploring these issues and focussed on acting to expand our capacity to create the common future we all want.
2. Our organisation has numerous on-going initiatives to continually try to engage people in dialogue, reflection and action on these issues.
3. Management arranges regular workshops and suchlike for people to come together and discuss these issues.
4. Occasionally someone in authority will get a bee in their bonnet about this kind of thing and institute a programme of action – which we all mostly ignore in the entirely reasonable expectation that it will all blow over in due course.
5. Say what? Eh? Doh!?

q26: how does your business share the intellectual property, skills, and experience it's continually acquiring and creating, and guard against it walking out the door?

Answers:

1. Our organisational systems and policies actively encourage and reward folks to share knowledge and learning with each other. Our development processes have in-built checkpoints where new knowledge and learning is regularly captured, acted-upon and shared across the whole organisation.
2. We make sure that each iteration of each project captures lessons learned, process improvement suggestions, technical innovations, ideas, etc. and we require that the project teams log this information in our organisation's IP (Intellectual Property) asset repository from where it's available to everyone.
3. We insist each project has a retrospective or review at the end of the project to document lessons learned, etc..
4. We don't do much to capture or share organisational knowledge – we leave it up to individuals to develop their skills and knowledge and share when necessary.
5. Our organisational systems and policies actively encourage folks to keep knowledge and learning to themselves, and when they threaten to leave we desperately offer them huge raises or promotion just to stay a little longer.

q27: what's your development organisation's prevailing attitude to investing in tools (e.g. software tools)?

Answers:

1. We encourage people to understand the work they're doing and the best way to organise it, before considering how tools could fit into the equation.
2. We'll buy tools whenever and wherever we can make a bona fide business case for them.
3. We automatically reject the idea of expensive tooling – most jobs can be done well enough without having to spend time and money on special tools, can't they?
4. Tools are great, aren't they? Any time we can reduce headcount and staff costs by buying a tool, it's a no-brainer.
5. Whenever we think of changing or enhancing our processes, or getting into a new line of business, we select the most sophisticated tools we can find – often as a substitute for having to invest the considerable time and effort in understanding the problem we're actually facing.

q28: when things go well, how does your development organisation mark success?

Answers:

1. We throw a big party funded by the company, with all our people, the press, and other stakeholders, invited to attend. At the party we present a shared award to the folks involved. We also have regular events to celebrate the company and any outstanding successes.
2. We throw a small party for the project, with selected other folks - like immediate sponsors - invited to attend.
3. We send round an email and/or post a notice on the company notice-board congratulating the folks involved.
4. We completely ignore our successes. It's not very professional to get emotional about business, is it?
5. We actively play down our successes - because we don't want people getting too big for their boots, do we?

q29: to what extent does your development organisation co-locate people working on a project?

Answers:

1. We bring all the necessary folks, from many different disciplines, together in a "Big Room" for the entire duration of their contribution to the project.
2. We co-locate all the project's developers (engineering staff) together in one space for the entire duration of the project, with sufficient room for customers and other stakeholders to sit in as often as required.
3. We co-locate all the project's developers together in one space for the entire duration of the project, with an outreach program for them to go visit customers and other stakeholders when necessary.
4. We try to get the necessary people together in the same building or campus but it's rare for them to sit near one another.
5. We rarely have all the skills we need to build a product - so we have to make do by using non-specialists in certain specialist roles. We never get as far as thinking about co-locating these folks.

q30: what marks the "operational rhythm" of your development organisation?

Answers:

1. The almost-daily shipment of new product features to customers.
2. The regular release of new products and product upgrades into the market.
3. We don't really notice any particular rhythm to our activities as an organisation.
4. The quarterly or annual budgeting activity, or the Annual Report to Shareholders.
5. Rhythm? Our organisation is more like the jarring fits-and-starts discord of a Stockhausen symphony (or the manic fibrillation of a victim of Cardiac Arrhythmia).

q31: how do your people typically go about designing a new software-intensive product or service?

Answers:

1. Developers design and implement a product evolutionarily - feature by individual feature – gathering requirements "just-in-time", as they go.
2. Evolution of design – derived from clear evolving requirements.
3. Architects and/or designers take a completed set of explicit requirements and BDUF (Big Design Up Front) from explicit requirements.
4. BDUF from implicit or no requirements.
5. We don't do design (per se).

q32: how does a new product idea make it into development in your business?

Answers:

1. We have an internal market that allows everyone to get involved in establishing a "price" for each idea and those ideas that top the "price" list get implemented ahead of the rest.
2. Big product ideas get broken down into their minimal constituent parts, and the highest-value add "mini-ideas" get implemented and sold to customers, with the other related ideas being implemented as and when these "beachhead" ideas succeed.
3. We have a series of Gates that an idea has to pass through to move from a raw idea to a finished (saleable) product. One of the early gates is the business case – each new idea must have a valid and compelling business case illustrating the proposed product (or service) costs, risks, markets, revenues and ROI.
4. New ideas wallow around in a fuzzy front-end until someone or something happens to cause them to move into development.
5. We have no established means to move ideas into development (and thence into production and sale) in any organised or repeatable way.

q33: how much of your development organisation's effort goes into coding (programmers writing source code, e.g. Java, C#, Python, or VB)?

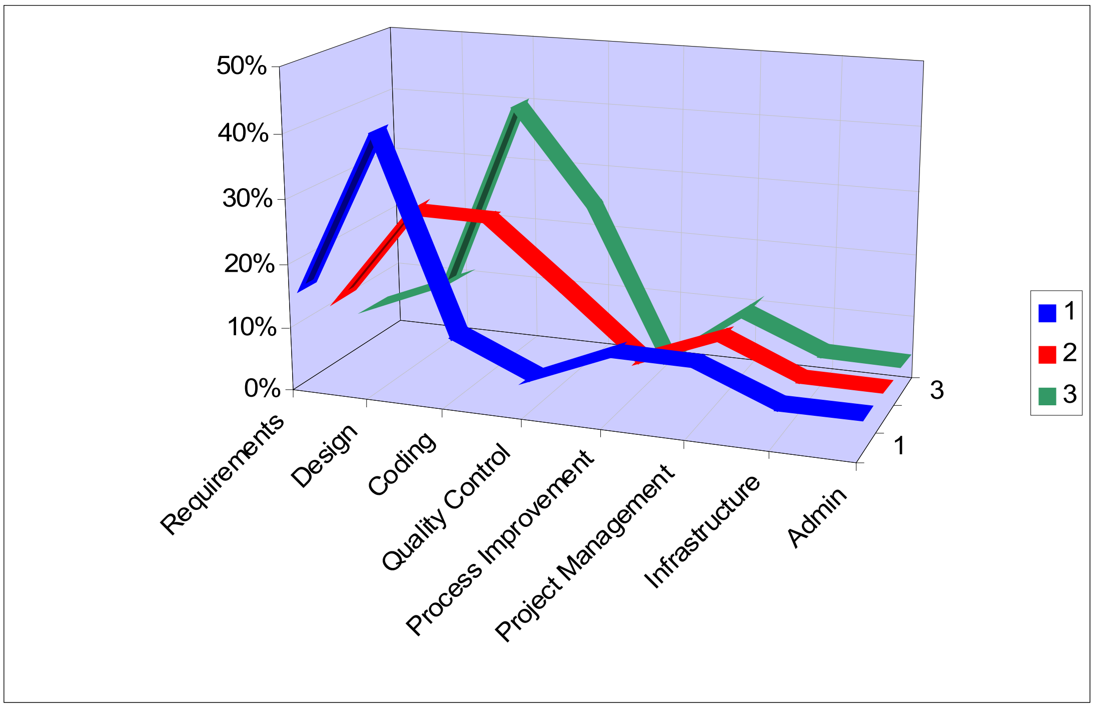
Answers:

1. Close to zero percent – most of our development consists of understanding stakeholders' needs and re-wiring existing components with simple configuration tools in order to meet those needs.
2. Around 10%-15%; we spend a lot of time on understanding our stakeholders and their requirements, managing risks and ensuring we deliver business value and process improvement – in all of which coding plays a minor part.
3. Some 25%-30%.
4. Anywhere between 50% and 80%. Most of the remainder goes into testing.
5. Close to 100% - what else matters?

q34: how does effort divide across a typical project within your development organisation?

Answers:

1. Our effort breakdown looks something like the blue line (#1) on the chart, below.
2. Our effort breakdown looks something like the red line (#2) on the chart, below.
3. Our effort breakdown looks something like the green line (#3) on the chart, below.
4. We choose to spend most of our time just coding. What else matters?
5. We generally have no idea where our effort goes.



q35: how many different tasks does a person in your development organisation typically work on at the same time?

Answers:

1. Only one – ever. We all understand the deleterious effects of having people multi-task.
2. Mostly just one. But folks like project managers, architects, marketers, analysts, and so on typically have more than one thing on the go at the same time - to keep these expensive specialists as busy as possible.
3. Most people find themselves working on two or three different projects at any given time.
4. If anyone looks like they're not fully utilised, their manager will always find something extra for them to do to ensure they're kept 100% busy.
5. Folks have so many things on the go at one time that they never know what they're supposed to be working on from one day to the next.

q36: what kind of computing resource does a typical development project team have available to them?

Answers:

1. One (or more) workstations per developer, each with several 21"+ monitors, plus several dedicated servers; a server to hold the project Wiki and website, code repository, etc.; one or more servers to host dedicated test environments; a build and smoke cluster; and at least one master database server.
2. One workstation, each with a 21"+ monitor, per developer, plus at least one dedicated project server.
3. One workstation per developer, plus a share of a project server.
4. One workstation per developer, no servers.
5. Less than one workstation per developer.

q37: how do you represent the size of your products or projects (for estimating, monitoring trends, etc.)

Answers:

1. We use customer-oriented, output-related sizing measures, such as value, billing, revenue, etc.
2. We use established sizing measures such as Functional Size Measures, COSMIC, COCOMO, etc.
3. We use input-related measures such as man-days, development cost and/or task- or work-breakdown (GANTT).
4. We typically use KLOC (Lines of code)
5. We don't use any means to represent e.g. product size.

q38: how does your development organisation typically select new employees (at all levels)?

Answers:

1. Teams review submitted applications (not CVs!) and invite likely candidates for an informal chat over coffee or a pint. If a majority of the folks agree, selected candidates are then invited to meet the CEO, and on his/her go-ahead, the candidate is invited work with the team (on a paid, contract basis) for a week or two so everyone can get to know each other. At then end of this time, if a majority of the team – and the candidate – approve, a formal job offer is made.
2. As 1. above, but we require the explicit approval of a manager in addition to a majority of the team.
3. Team leaders or project managers review submitted CVs and invite likely candidates for an informal chat over coffee or a pint. Selected candidates are then invited to meet the team over a day's worth of meetings. On the team's unanimous go-ahead, the team leader or project manager issues a formal job offer, perhaps with the involvement of HR.
4. Line managers review submitted CVs and invite likely candidates for an first interview. Approved candidates are then invited to meet their prospective colleagues and attend a second interview. The line manager, possibly assisted by HR, subsequently issues a formal job offer.
5. The HR department appoint one or more agencies to filter CVs. CVs that make the cut are then filtered by the HR department itself, before being passed on to the relevant line manager. The line manager decides who to interview, interviews candidates – either themselves or through a panel – and subsequently asks HR to issue a formal job offer.

q39: how does your development organisation typically select and deploy development technologies and tools (such as programming languages, notations and methods)?

Answers:

1. We continually look to see where we can reduce waste (such as defects and rework). Whenever we spot an opportunity to take action to reduce waste, some folks in the development organisation get together to evaluate - sometimes on live projects but more often in evaluation spikes – alternate candidate technologies and tools. If something proves likely to reduce waste, then the folks involved organise its addition to the deployment 'mix' (the technologies and tools available to the development folks.)
2. As 1. above, but we require the explicit approval of a manager in addition to the approval of the evaluation team.
3. Team leaders or project managers mandate appropriate technologies.
4. We have uses the same technologies and tools year-in, year-out and change only comes about haphazardly, typically coinciding with a changeover in the relevant senior management.
5. We've used COBOL (or FORTRAN or PASCAL or BASIC or APL or PERL or ASSEMBLER or some such) for as long as we can remember – we have lots of folks who know this stuff and see no reason to change.

q40: now you've completed this questionnaire, where do you think your business sits in the rightshifting spectrum of effectiveness?

Answers:

1. Way to the left – somewhere below x0.5.
2. Around the modal average – near to x1.
3. Above average – somewhere between x2 and x3.
4. Highly effective – to the right of x3.
5. Don't know.

For a complimentary, personalised 5-page report about the effectiveness of your software and product development business, entirely free of charge and with no obligation, please send your full list of answers to: bob.marshall@fallingblossoms.com.