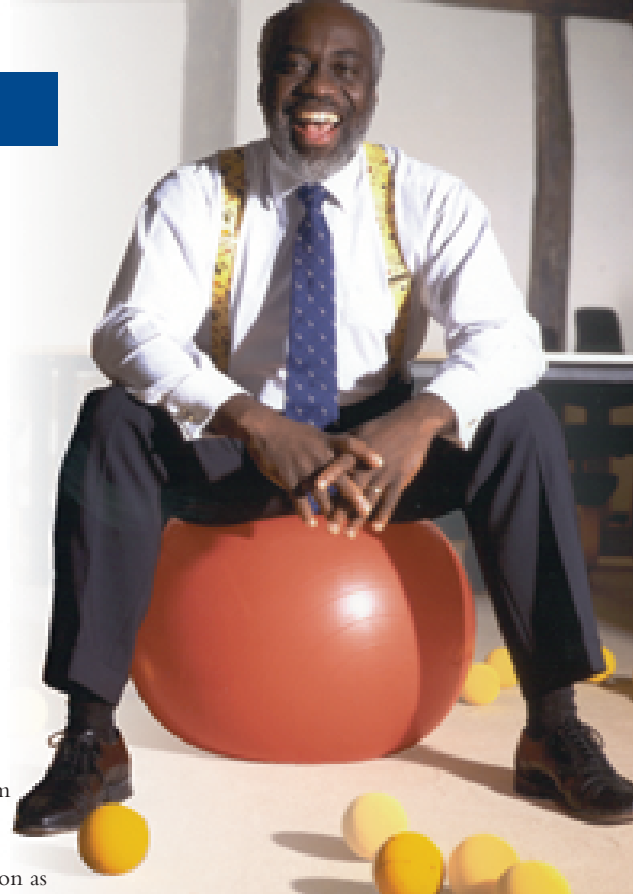# Helping stakeholders to understand requirements

Eddie Obeng continues his unique journey through Project Land.

I know you've done what I asked for, but what I said wasn't what I meant!

It's ridiculous. You're six months into the project, there's no way you can make such a major change. And anyway, there isn't any budget. This software project, like all the software projects you've ever worked on, is missing its milestones. You wanted a prototype-based approach with go/no-go points, but the client wanted a lower cost and wasn't prepared to pay for prototypes and reviews. It's the client's own fault. I mean, how were you supposed to know that they hadn't been thorough enough?

You didn't know that they hadn't read the requirements specification. It's not your fault, you tried your best to get them to write it and they wouldn't. What was it he said at the time? 'I'm far too deep in it to waste my time trying to double guess what you're going to do. I know what I want but it's up to you to deliver.'

So you wrote it under protest but you tried to do a thorough job. And you did – the document came to 300 pages … and anyway, beating up the subcontractor won't help, they'll just go bust and leave you with a bigger problem. And, to add insult to injury, someone has left an anonymous cartoon on your desk. The picture is of a medieval castle under construction. It seems normal enough until you realise that the partially constructed moat seems to run inside the castle walls instead of outside them. The caption reads: '...suddenly a heated row broke out between the King and the moat contractor!'

So why is it often so difficult to work out what is required by the customer stakeholders?

It's always been difficult to get the requirements specification right. I've noticed a certain behaviour within myself and you might behave the same way. When I'm not really sure of what I want, I am deliberately vague so that I can return to the problem and make up the solution as I go along. So, I guess, like anyone else driving changes that they don't fully understand, I am loath to write down what they ask for in case I'm (they are) held to it. And I'm definitely not going to sign off anything, especially not in my own blood!

But these days it's doubly difficult. The problem is that, today, projects where the goal is known and the method of delivery fully understood (what I call painting-by-numbers projects) are increasingly rare. Such projects have long since been turned into streamlined day-to-day processes, or been exported for execution to low-wage economies. The problem is simply that if it's so clear to you what to do and how to do it, chances are it's clear to other organisations and the margins and profits have been competed away to be razor thin. But this case isn't painting-by-numbers.

So what type of project is it?

The clue is in the statement: 'I know what I want but it's up to you to deliver'.

The project has a clear goal, but the method of delivery is missing. It's what I call a Quest. In practice, difficulty in delivery may mean that the goal has to be changed in scope or detail. Movies (method decided – goal missing) are an even bigger nightmare. Many software application deployments fall into this type and, as a rule, less than enough brain power is spent on fully establishing what is to be delivered. Foggy change (unclear goals and methods) is perhaps the hardest to specify up front.

The less the requirements are understood the more important it is to 'chunk' up the project and review frequently

The theory of best practice in this area is that the project customer/owner stakeholders should be responsible for fully detailing their needs. This rarely happens, either because the type of project makes it difficult, or they don't know how to do it, or they have the problem I mentioned earlier and are unwilling to do it.

It is important not to rush this fence. In fact, it's worth making the fence a little higher by linking the needs and requirements to the business benefits directly.
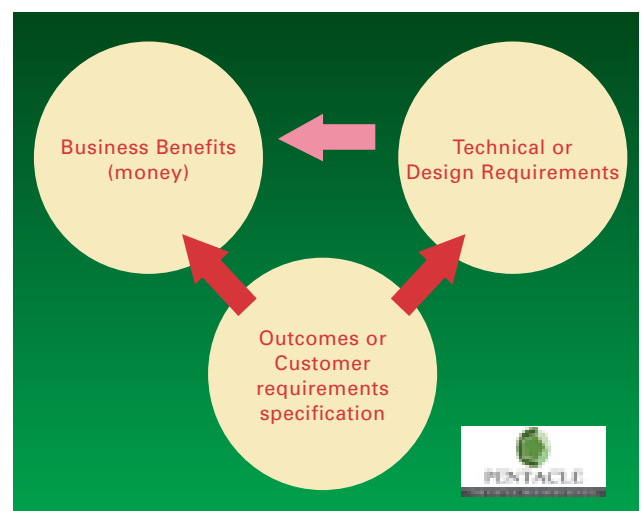
This way, later in the project, if it needs to be reviewed or re-scoped, it is very clear what the business impact of the decision is. But, far more importantly, linking to business benefits makes the project deliverables more realistic, tangible and less boring for the project customer/owner.

If possible, incorporate the business benefits, customer specification and the technical/design requirements in the same document, using a framework:

What are we trying to achieve? **Business benefits** 1. Revenue increase 2. Operating cost reduction 3. Resource re-deployment 4. Compliance **Prepared by project owner** Financial language	To be achieved through: **Outcomes or customer requirements specification** New processes/activities/ interfaces etc. **Prepared by project customer/ owner** Operational language	To be delivered by: **Technical or design requirements** Detail of structure/ architecture Detail of performance criteria etc. **Prepared by project manager/ technical expert** Technical language

In this way every requirement is linked to a benefit. But it still relies on the project owner and project customer; and what happens if they won't help to produce it? 'You didn't know that they hadn't read the requirements specification.'

As they say on airlines at take-off: 'In the unlikely event that you should have to write the specifications and business cases…' 'So you wrote it under protest but you tried to do a thorough job. And you did – the document came to 300 pages.'

First, be sure YOU fully understand WHY the work is to be done:

● Write it in small 'chapters' and keep passing these to your stakeholders for comments – stakeholders are notorious for not reading any document that contains more than a single word
● Keep it simple
● Use diagrams and images if possible
● Make sure the language in the first two columns is non-technical

What about the moat contractor?
● **Preview each stage before starting development of the deliverables:**
A good preview should brainstorm what is expected to be delivered, anticipate risks and cover how the deliverables will be incorporated into the processes and activities and check the delivery of benefits
● **Use 'implications questions' a lot:**
An 'implications question' is a question which asks: 'If we do this, what happens next?' 'How will this affect you?', etc. Check what happens if it works/doesn't work/works too fast/works too slowly
● **Make it visible using prototypes and mock-ups:**
How will it look/sound? I remember once working on a project where we hired a warehouse to mock-up the entire new workspace before starting on delivery
● **Put it into real life using walk-through/role plays:**
How will it feel? How is it used?

There are also some project-type hints you should be aware of. On Movies – the project stakeholders won't be aware of what benefits can be gained, so spend time making sure that they are clear. On Quests – the project stakeholders need to be clear on the level of benefit that is acceptable for the effort involved. In the Fog you may need to do several rounds of business benefit, customer requirements and technical design spec. to converge on an acceptable solution!

PS. Thanks for all the suggested titles for my new innovation book. I've had some great ones, eg, 'From Dreams to Dollars' and some dreadful ones, eg, 'Kissing Rabbits', but they are all welcome. Like all good projects, the final title will be decided just before it goes to print – so I'll keep you informed.