Application Development – Who calls the shots?

(Original creator: michaelrabone)

Once upon a time (otherwise known as when I first worked in analysis) it was possible for the guardians of the purse to call for a set of numbers indicating the cost of procuring a new application. Much activity would take place, usually lasting months or even years, whilst information on operational, technical and, most importantly, economic feasibility was gathered. Thereafter estimates were produced and, in due time, formal plans. With luck, during this process, the users of the proposed application were consulted and a list of all that would be included in the development was drawn up and formally signed off. At this point the users knew what they thought the application would include; those involved with development knew what they considered was expected of them (even if they were unsure how to produce it) and, most significantly, the guardians of the purse knew exactly how much money had to be set aside for the task. Invariably, even in the best regulated environments, the story then took a turn for the worst: Beset by problems of technical difficulty because so much was untried; hampered by arguments about changes in the business that were not being incorporated into the new application and hamstrung by the need to stay within budgets set before a realistic estimate could be arrived at.

The experiences of those involved in these epics are stuff of legend. Everyone knew how awful things could become with disillusioned project teams and users being given out-of-date and irrelevant application . Everyone was unhappy except, perhaps, the money managers. They had their estimates, they could argue for economies, they could set aside the correct amount of money, as they saw it, and they could berate the project teams whenever any limit was breached. Who were these guardians of the purse? They ranged from true financial staffs right down to everyday line managers. All these individuals had a vested interest in ensuring that they could forecast, plan and account for the activities associated with the project. What about the others involved? Yes, the users want to know how long the application would take to develop and the project teams desire the satisfaction of finishing a phase or even a project by the due date. But such matters were unlikely to be the major concerns of these groups. Users really want systems that will be useful; that will assist them in their day to day work; will make their life easier or more profitable or more satisfying. Project teams want to produce beautifully crafted technical solutions, utilising the most up to date technology and moving their career resume on to new heights.

When a group has a single or unified aim it is, obviously, likely to be more easily achieved than if objectives differ or are, at their worst, in direct competition or confrontation. To provide an environment where the aspirations of all groups are met really is likely to be a fairy tale. What can be done to allow applications to be developed in an atmosphere which is more harmonious than haunting? The users must consider themselves the owners of the application. If the development is seen as an “IT system” it is likely seen as being the property of the “IT Department”. Projects are, by their very nature, transitory. Time, effort and money have to be expended; business managers have to be absolutely clear about the importance of including their resources in the development as well as those of the IT Department or Project Team. It is recognised that there is little “fat” in any job. Most workers already have a full day without the introduction of a project team who will take up time, space and equipment for even a limited period. However, if the users don’t own the application they are unlikely to get what they require and the only useful training for developers will be a course in “Fortune Telling”. Modern development techniques need not and indeed, in my opinion, should not be an unstructured exercise. But the use of discovery and prototyping methodologies mean it is very difficult to produce even good estimates of time and cost upfront. Using methodologies like Agile, third-party fixed price contracts will either have to contain enough contingency to make cost-benefit unrealistic or present such a risk to the developers as to be not worth the effort. Some people say that that trust must be engendered. Trust between buyer and seller? In a multi-million dollar system development? Does this make discovery and prototyping suitable only for in-house projects? If it does, then the sad fact is that the tools, techniques and technology will be wasted and users will be the biggest losers.