Why project fail

Why Projects Fail Computer projects fail when they do not meet the following criteria for success: It is delivered on time. It is on or under budget. The system works as required. Only a few projects achieve all three. Many more are delivered which fail on one or more of these criteria, and a substantial number are cancelled having failed badly. So what are the key factors for success? Organisations and individuals have studied a number of projects that have both succeeded and failed and some common factors emerge.

A key finding is that there is no one overriding factor that causes projectfailure. A number of factors are involved in any particular project failure, some of which interact with each other. Here are six of the most important reasons for failure. 1 . Lack of User Involvement Lack of user involvement has proved fatal for many projects. Without user involvement nobody in the business feels committed to a system, and can even be hostile to it. If a project is to be a success senior management and users need to be involved from the start, and continuously throughout the development.

This requires ime and effort, and when the people in a business are already stretched, finding time for a new project is not high on their priorities. Therefore senior management need to continuously support the project to make it clear to staff it is a priority. 2. Long or Unrealistic Time Scales Long timescales for a project have led to systems being delivered for products and services no longer in use by an organisation. The key recommendation is that project timescales should be short, which means that larger systems should be split into separate projects.

There are always problems with this approach, but the benefits of oing so are considerable. Many managers are well aware of the need for fast delivery, leading to the other problem of unrealistic timescales. These are set without considering the volume of work that needs to be done to ensure delivery. As a result these systems are either delivered late or only have a fraction of the facilities that were asked for. The recommendation here is to review all project plans to see if they are realistic, and to challenge the participants to express any reservations they may have with it. . Poor or No Requirements Many projects have high level, vague, and generally unhelpful requirements. This has ed to cases where the developers, having no input from the users, build what they believe is needed, without having any real knowledge of the business. Inevitably when the system is delivered business users say it does not do what they need it to. This is closely linked to lack of user involvement, but goes beyond it. Users must know what it is they want, and be able to specify it precisely.

As non-lT specialists this means normally they need skills training. 4. Scope Creep Scope is the overall view of what a system will deliver. Scope creep is the insidious growth in the scale of a system during the life of a project. As an example for a customer bills, then these bills will be provided on the Internet, and so on and so forth. All the functionality will have to be delivered at one time, therefore affecting time scales, and all will have to have detailed requirements. This is a management issue closely related to change control.

Management must be realistic about what is it they want and when, and stick to it. 5. No Change Control System Despite everything businesses

change, and change is happening at a faster rate then ever before. So it is not realistic to expect no change in requirements while a system s being built. However uncontrolled changes play havoc with a system under development and have caused many project failures. This emphasises the advantages of shorter timescales and a phased approach to building systems, so that change has less chance to affect development.

Nonetheless change must be managed like any other factor of business. The business must evaluate the effects of any changed requirements on the timescale, cost and risk of project. Change Management and its sister discipline of Confguration Management are skills that can be taught. 6. Poor Testing The developers will do a great deal of testing during development, but eventually the users must run acceptance tests to see if the system meets the business requirements.

However acceptance testing often fails to catch many faults before a system goes live because: Poor requirements which cannot be tested Poorly, or non planned tests meaning that the system is not methodically checked Inadequately trained users who do not know what the purpose of testing is Inadequate time to perform tests as the project is late Users, in order to build their confidence with a system, and to utilise their experience f the business, should do the acceptance testing.

To do so they need good testable requirements, well designed and planned tests, be adequately trained, and have sufficient time to achieve the testing objectives. Conclusion These six factors are not the only ones that affect the success or failure of a project, but in many studies and reports they appear

near, or at the top of the list. They are all interlinked, but as can be seen they are not technical issues, but management and training ones. This supports the idea that IT projects should be treated as business projects.