NotesOn: Risk Management - Risk Based Project Management

Introduction (V1.0):

Risk Based Project Management, as a subject, has come up more than once and did again recently. It’s doing so reminded me of a short email chain with a bright young project manager named Linda. The discussion is succinct, to the point, and of value to the project management community so I am publishing it as written.

Introduction (V1.0):

Background:

Subject: Risk Based Project Management Question

Email #1:

Hi,

[A friend of ours] referred me to you with my question. Here at my new job, I have encountered a methodology I’m not familiar with - risk-based project management. The definition is that the highest risk tasks are front-loaded into the project to ensure they are completed. Sounds great on paper but I can see flaws. Have you ever encountered this methodology? I don't see anything exactly like it when I do a web search.
Thanks,
Linda

Email #2:

Linda,
Yes, I’ve heard of this. It is most typical with High Risk app development environments/projects; though it can be used on any project where even one item has a lot of question marks around it.

The idea is to identify the MAJOR risks up front (and where projects are thinking of using cutting/bleeding edge technology there can be many) and then address them as early as possible via Proofs Of Concept and Prototypes.

Let’s say that you’re trying out a new version of MS’s SQL Server 2008 database clustering, with encryption, AND your app dev environment is the bleeding edge .NET 3.0. You want to (1) set up the clustered environment and test it, thoroughly -- there are a number of ways of doing this and things you want to do, like make sure your test environment is a mirror to what Prod will look like (a good excuse to build a parallel QA environment) -- and (2) to resolve the new development environment you (a) train the team on the new dev environment, bringing in a mentor who is thoroughly experienced in 3.0, or beg/borrow one from Microsoft, and (b) build basic components first that prove the environment will work -- leave all the bells, whistles and bling for later.

Hopefully this gives you the idea; put more succinctly: you want to POC / Prototype ALL of your high-risk project aspects first, and put go/no-go decision gates around them so that if “it ain’t workin’ right” you can check down that solution and shift to a Plan B. However, this approach does NOT mean that the PM can ignore the rest of PMBOK. Right? Right. :o)

Email #3:

This makes perfect sense given the product I’m working on (master data management). What threw me was the sentence that indicated that ALL major risks would be addressed up front - when I know that the cut over from test to Prod is a huge risk (as in how can you test that early?).

Thanks again,
Linda

Email #4:

On major projects, where there is a huge risk in upgrading, you really have to make sure you’ve bullet-proofed the entire upgrade process, in addition to all of the normal QA steps (load/stress testing, etc.).

One way I’ve done this in the past is to (a) have one or more QA environments just for practicing the upgrade, including file conversions, etc. and (b) using that environment, checklist each and every step of the upgrade/conversion process and then practice it over and over until the team hits it 100% and there are no items missing from the checklist. [For] one application upgrade we had 7 separate environments -- 3 for dev and 4 for QA; with one [of the QA’s] dedicated to UAT and User Training.

This situation you describe is also a perfect scenario for beta testing too, i.e. inviting selected users to “have at it” in their totally separate environment.

As you’ve noted, the risk isn’t always in the software or hardware, sometimes the processes around them can be the high risk items.
Summary:

While the above is true, the full truth is that every project should be approached from a Risk Management perspective. “Know before you code” is a very handy philosophy that will save you endless hours of teeth grinding, days of re-do’s, weeks of missed vacations and holidays, months of lost sleep, a good deal of money and quite possibly your job. An example of how this project style is approach and applied is also discussed in the above referenced “NotesOn: IT Fundamentals – Simple Defined More”.

Hope this helps,

DP Harshman

PDF Link