Time Boxing

Think of TimeBoxing as a train schedule. Every 15 minutes, a train leaves the station. It carries whoever arrives. Periodic TimeBoxing says that a release happens every TimeBox, and carries whatever features can be finished. The releases **always** happen, on time. The project then measures (in some form) the content of each release. TimeBoxing need not be periodic -- it simply says that the date of the release is held constant, and the content allowed to vary.

This is one of the basic principles of the DynamicSystemsDevelopmentMethod (DSDM).

It seems this is applicable to a larger degree to ongoing and continuous software development, such as that of a product which is undergoing ConstantChange, and which is being released in new versions, patches, and upgrades which represent the state of the art. Examples: AutoCad, Linux, Perl, Windows, ProductivitySuites such as Word, Excel, Access, PowerPoint, Outlook, etc.. This does not imply that it can not be applied to less continuous products, or products in the early phases of the SoftwareLifeCycle.

For better or worse, TimeBoxing is simply an alternative to FunctionBoxing. I know of no arguments that make it more or less suitable for the examples given above -- it is simply a different approach to managing deliverables.

Well, the failure modes seems to be different, which might be significant. Looks as if a TimeBox delivery will always occur, but might contain no increment in value, whereas a FunctionBox delivery might never happen. So, we might ask, which kind of failure is easier to manage (i.e., change into something less likely to fail): deliver in one second from now, or deliver as soon as you have proved that P=NP?

See:

- http://www.malotaux.nl/nrm/Evo/TimeBoxing.htm
- http://www.qsm.com/fmm_05.pdf
- IncrementalDelivery

CategoryTime

Last edit December 11, 2008, See github about remodeling.