david consulting group

How can an IT organization identify its own software best practices?

August 2012

Scope of this Report

We often read articles about software development best practices. Sometimes these articles entertain us with first-person stories about how a particular development practice or a software development tool helped to ósave the day!ôon an important product release or systems implementation. Other times, these best practices articles inform the reader about a specific tool, technique or process that warrants a ôbest practicesôlabel because it can improve an organizations productivity and quality.

There is no known industry standard or certification that is used to qualify something as a best practice; nor are there any rules or guidelines that help to classify something as a best practice. So how do we know if something is a best practice? What gives a practice or a process that special distinction of being the ôbestô?

How can we recognize a Best Practice?

Letôs look at three fairly standard quality related practices and processes that are typically referred to as a best practice. The first one that comes to mind is the practice of conducting a formal review. This involves the formal review of artifacts such as requirements documents or design specifications. The benefit of a formal review is to create a deliverable that is accurate and free of errors and omissions. I doubt there will be much debate among the readership that formal review are often referred to as a quality best practice.

A second example of best practice that is often mentioned in the best practices category for traditional waterfall approaches is requirements definition (an agile practitioner might take a different view!). From the waterfall perspective, requirements definition is a rigorous, definable and repeatable process that enables analysts to effectively extract requirements from a customer or end user. There are numerous methods for defining requirements, and so this best practice isnôt labeling a specific process, like a well defined formal design review process, but it is addressing the practice of rigorous and formal requirements definition.

The third best practice example is code inspections. This typically refers to the Fagan type of formal code inspection process that has a well defined process with defined roles and responsibilities. So here we have a best practice that is aligned with a specific process.

So how do we know that these are the ôbestôfour a particular context among the many development processes and practices? Well, for one, we have been told over and over again that this is the case and often times these success stories are substantiated with qualifying and quantifying evidence. In other words, they have made a positive and demonstrative difference in the life of a software development project.

Here is our first big clue as to why something may be called or labeled a best practice:

- It has been shown to be implementable in a specific context;
- It can be proven to be successful in that specific context;
- It can be quantified;
- The positive cause and effect can be broadly isolated and explained.

There is also a warning buried in this clue ï ñspecific contextò ï whatôs works for someone else in one part of their organization might not work for you in the area that you are looking at. It is important to dig below the surface a little to understand WHY a ôbest practiceö works.

Case in point ï have you ever worked in a software development shop that has initiated a process improvement strategy to include reviews and inspections (an agreed upon best practice) only to later realize that the program was not well defined and therefore not properly executed? Sooner or later the practice falls by the wayside for one reason or another? It is an all too common occurrence. So was it not a best practice? So is it now not a best practice? Possibly. But more likely; the best practice simply was not **executed** effectively and therefore it did not provide the ôbestö results for that particular organization.

The key points here are that a best practice is only useful in a particular context and is only as good as its execution.

The success of the execution is somewhat dependent upon measuring the effectiveness of the process and the end results. Measures donôt make the process work better but they will provide information along the way about:

- compliance to a process,
- the output of the process, and,
- the impact on the organization.

This ensures the effectiveness and long term use of the best practice which, in turn, maximizes the likelihood of a return on the investment made in implementing the particular best practices strategy.

Measuring Best Practices?

Some of the measures associated with the above named best practices include:

- Process compliance,
- Defect density,
- Effective removal rates

Process compliance is the basic practice of creating a formal mechanism to monitor and report compliance to a particular process. It does not provide insight as to the effectiveness or efficiency of the process but it does provide management with a view into the behaviors of the software development teams.

Defect density is often used to quantify and evaluate the number of defects attributed to a particular piece of software, systems application or software product. It is calculated by dividing the total number of defects found by the functionality delivered (measured in function points). The measure can be used to assess the overall quality of the software and also to predict the potential need for ongoing support.

The effective defect removal rate is used to measure the rate of defect removal throughout that lifecycle. The calculation involves calculating the number of defects removed at each phase of a lifecycle divided by the total number of defects discovered. This activity occurs at the various phases of a lifecycle. So for a waterfall lifecycle, you may have defect rates attributable to your requirements phase, your design phase, your coding phase, etc. This proves to be a very powerful quality measurement tool that provides insight as to the effectiveness of your quality practices. The resulting ódefect discovery curveòcan also be helpful in predicting, and then monitoring, the progress of future projects. The following chart is an example of measuring defect removal effectiveness.

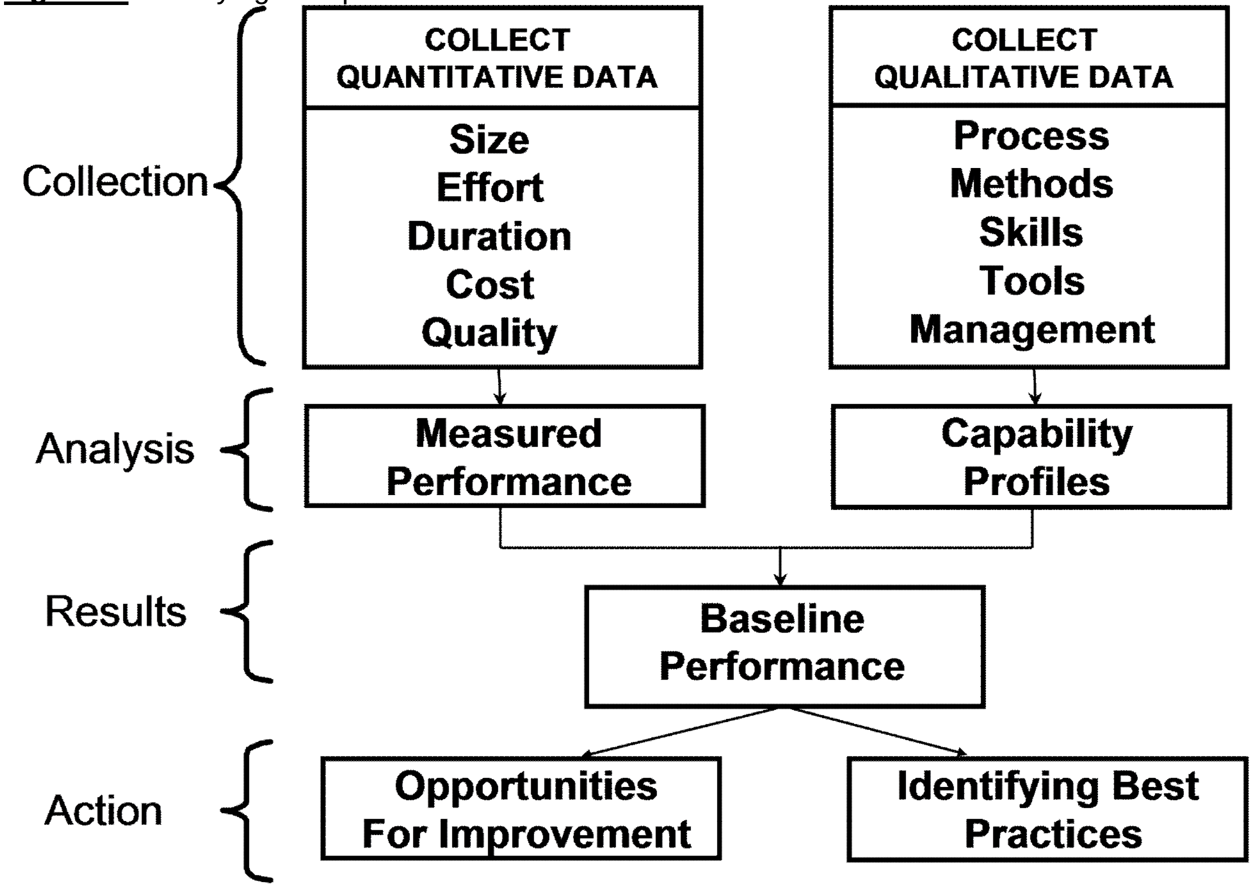
Rate = Ave. Defects/KSLOC	*Peer Reviews*			*Testing*				
Range	**Reqs**	**Design**	**Code**	**Unit Test**	**Int Test**	**Sys Test**	**Field**	**Total**
Insertion Rate	2.5	7.2	22.7	0.9	0.3	0.1	0.0	
Detection Rate	1.0	5.0	17.0	4.0	2.5	2.0	2.0	33.6
Leakage Rate	1.5	3.6	9.3	6.1	3.9	2.0	0.0	
Removal Effectiveness	40%	58%	65%	40%	39%	50%	100%	
Average* | *Best in Class*	*40% | 50%*	*70% | 85%*	*65% | 85%*	*35% | 55%*	*35% | 45%*	*40% | 55%*		
		Peer Review Effectiveness->	71%		***Total Effectiveness->***	94%		
		Best in Class->	85+%		***Best in Class->***	99+%		

* from Capers Jones, *Software Assessments, Benchmarks and Best Practices*, Addison Wesley, 2000

How can you identify your own Best Practices?

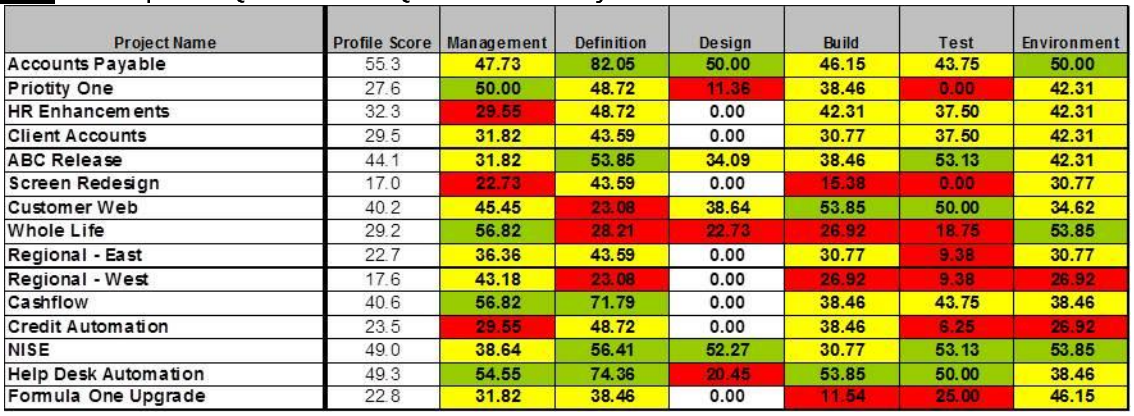
An effective way for an organization to identify its own best software practices is to conduct an internal performance measurement assessment. This involves collecting quantitative data relating to productivity and quality indicators for a selection of projects and at the same time collecting qualitative data about the development practices used on those same projects ï as noted in the Figure 1 (Collection).

Figure 1: Identifying best practice



An analysis of the collected data will help an organization to identify what development practices are the most and least productive. Each project will have a resulting measured performance level along with a profile of practices and how they contributed to high or low yields of performance. Table 1 is an example of the result of just such an analysis.

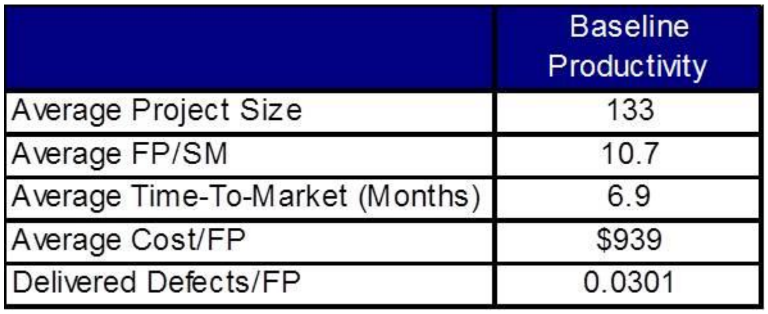
Table 1: Example of Quantitative/Qualitative Analysis of Processes:



Project Name	Profile Score	Management	Definition	Design	Build	Test	Environment
Accounts Payable	55.3	47.73	82.05	50.00	46.15	43.75	50.00
Priotity One	27.6	50.00	48.72	11.36	38.46	0.00	42.31
HR Enhancements	32.3	29.55	48.72	0.00	42.31	37.50	42.31
Client Accounts	29.5	31.82	43.59	0.00	30.77	37.50	42.31
ABC Release	44.1	31.82	53.85	34.09	38.46	53.13	42.31
Screen Redesign	17.0	22.73	43.59	0.00	15.38	0.00	30.77
Customer Web	40.2	45.45	23.08	38.64	53.85	50.00	34.62
Whole Life	29.2	56.82	28.21	22.73	26.92	18.75	53.85
Regional - East	22.7	36.36	43.59	0.00	30.77	9.38	30.77
Regional - West	17.6	43.18	23.08	0.00	26.92	9.38	26.92
Cashflow	40.6	56.82	71.79	0.00	38.46	43.75	38.46
Credit Automation	23.5	29.55	48.72	0.00	38.46	6.25	26.92
NISE	49.0	38.64	56.41	52.27	30.77	53.13	53.85
Help Desk Automation	49.3	54.55	74.36	20.45	53.85	50.00	38.46
Formula One Upgrade	22.8	31.82	38.46	0.00	11.54	25.00	46.15

Table 1 displays an evaluation of the current state for selected projects. The (vertical) ĩcategoriesòrepresent various lifecycle and project management phases. For each (horizontal) project, there is a numeric value and a color to indicate strong (green) or weak (red) areas of performance relative to benchmarks. Table 2 provides a composite view of the resulting measureable performance levels. Those process areas that score favorably would be noted as a measureable best practice for the organization. It is also worth noting that not all projects necessarily show the same areas as best practices. Indeed, it is common for one team to be very good at one area and not so good at other areas. Table 1 identifies opportunities for cross-training and helps with change management.

Table 2: Composite View of Measurable Performance Levels



	Baseline Productivity
Average Project Size	133
Average FP/SM	10.7
Average Time-To-Market (Months)	6.9
Average Cost/FP	$939
Delivered Defects/FP	0.0301

The next step would be to create an improvement plan for those practice areas which are not performing at a best practices level.

We have touched on just a few of the quality processes that we have come to think of as best practices. The key point to remember is that they will only be a best practice in your shop if they are well defined, properly executed and, most importantly, measured for success.

Conclusion

It is important for an organization to adopt and deploy software development best practices. Best practices can be introduced to the organization based on proven techniques such as formal reviews and inspections or they may be identified as already existing internally as a result of a performance measurement activity. To be sure best practices are being properly executed and results being realized, it is necessary to apply a consistent use of metrics. The software industry is mature enough to have learned and evolved software development practices that are known to have a positive impact on software development in the appropriate context.