Project Management

Week 8

Productivity Tools

Lecture Overview

- Role of productivity tools in Rapid Development;
- Productivity-tool strategy;
- Productivity-tool acquisition;
- Productivity-tool use;
- Silver-bullet syndrome.

Productivity Tools

- Productivity tools make up the technology dimension of the people-process-product-technology foursome;
- Adopting a new tool can be one of the quickest ways to improve productivity - but it can also be one of the riskiest.
Most productive organizations have found ways to minimize the risks and maximize the productivity gains; this strategy depends on recognizing three critical realities:
- Productivity tools seldom produce the schedule savings their vendors promise;
- Learning a new tool or practice initially lowers productivity;
- Productivity tools that have been discredited may produce schedule savings anyway - just not as promised.

What is a productivity tool?
- A tool that has the potential to significantly change the way you work:
  - 4GLs (fourth Generation Languages);
  - Visual programming languages;
  - Code generators, code/class libraries;
  - Tools that offer dramatic reductions in workload and equally dramatic improvements in development schedules.

Computers are good at automatic repetitive tasks - however, there are many aspects of software development that are not repetitive and are not well suited to automation by hardware and software tools;
- Brooks (1987), published “No silver bullets, essence & accidents of software engineering” - infamous;
- Brooks argues that the difficult part of software development arises from the fact that a computer program is a set of interlocking detailed concepts - specifying & designing such concepts is more complex;
Role of Productivity Tools in RD

- Software programs are becoming more precise, more detailed & more complex, not less as the conceptual essence has to be thought through;
- The person who does this must understand each precise detail of the program, therefore, such an understanding is difficult, error-prone and time-consuming;
- Time required to achieve such an understanding is a major contributor to the time required to complete a software development project;

Role of Productivity Tools in RD

- To achieve dramatic reduction in development time, a new technology would have to simplify the essence of software development, making it easier to formulate the interlocking concepts of a computer program;
- The more a technology strikes at the essence of what makes software development difficult, the more effort it is likely to save;

Role of Productivity Tools in RD

- The move from low-level assembly languages to high-level languages such as C & Pascal freed you from having to think about what the underlying machine was doing when it ran the program;
- Move to visual programming languages such as Delphi & Visual Basic provides a similar kind of simplification - allowing you to forget about many of the tasks a windowing environment performs when it runs a graphically oriented program.
Role of Productivity Tools in RD

- Tools that do not strike at the conceptual essence, do not provide much assistance such as programming editors, source-code control tools, debuggers, execution profilers, automated testing tools etc. That only provide incremental productivity gains for the same reason;
- CASE tools and code libraries/code class libraries are in between high-level languages and coding tools in their ability to improve productivity as they strike at the root of the problem, but they do not strike very hard.

Role of Productivity Tools in RD

- Brooks (1987) asserted that no single technology or practice looked as though it would be capable of producing a 10-fold improvement in productivity over the next 10 years;
- We have to be satisfied with individual practices and technologies that generate productivity improvements of less than 25% per year.

Role of Productivity Tools in RD

Areas of Special Applicability

- Productivity tools tend to focus on software construction - how effective they are in supporting Rapid Development depends on how much of your project consists of software construction;
- How large is the project? How complicated is the project? How much of the lifecycle is occupied by software construction?
Role of Productivity Tools in RD

Areas of Special Applicability
- DBMS-oriented applications

Database applications are well supported by productivity tools on nearly all platforms. Tools exist to generate database schemas, generate queries, format reports, create data-entry screens using tools such as Visual FoxPro, Access, PowerBuilder, CA Visual Objects, FileMaker, Focus, CASE Tools - to take the repetitiveness out of the process.

Role of Productivity Tools in RD

Areas of Special Applicability
- Custom Applications

RD 4GL or visual programming languages work well for small applications in which the design is well understood.

In many custom-software projects, the customer is willing to accept a few limitations in look and feel in exchange for faster development time.

Role of Productivity Tools in RD

Areas of Special Applicability
- Throwaway Prototyping

RD languages are extremely well suited to development of throwaway user-interface prototypes;

To develop a throwaway prototype in C or C++ is ridiculous;

Using a visual programming language is most suitable to throwaway prototyping and developing rapid iterations of user interface design you plan to throwaway.
Role of Productivity Tools in RD

Areas of Special Applicability

- In general, the smaller & simpler the application, the more valuable productivity tools will be;
- On small projects - most of the project is consumed by code construction which is the part that is assisted most by productivity tools;
- On large projects - a smaller proportion of the project is devoted to coding, so the potential contribution of code focused RD will be smaller.

Role of Productivity Tools in RD

Productivity Tool Limitations

- “Three steps forward & two steps back - it is a fact of life that productivity tools giveth & taketh away”
- If you plan to use a productivity tool, you should add about 25% on top of the total time you expect to spend working with the tool for use in working around the tool’s limitations;
- There are always things that are harder to do because you are using a tool & sometimes there are defects in the tools.

Role of Productivity Tools in RD

Ultimate Role of Productivity Tools on RD Projects

- If you weigh-up the cost & benefits of using a PT, the argument for using PT’s on a RD project becomes less clear-cut;
- Best projects do not necessarily have state-of-the-art methodologies or extensive automating & tooling - they rely on basic principles such as strong team work, project communication & controls;
- Good organization & management are more critical success factors than technology in obtaining successful RD;
- PT’s are are only a secondary contributor to an organizations all over level of productivity.
Ultimate Role of Productivity Tools on RD Projects

- There are so many other influences on a software schedule that there is not any clear-cut data to support the contention that projects using 4GL’s are delivered any faster than those using 3GL’s;
- Significant variations in schedule are more likely to result from differences in planning, management, requirements specification, than in the technology used to construct the system.

In a 17 year study involving 100 experiments with live projects & 50 technologies, NASA’s SE Lab concluded that improvements are characterized by continual, sustained and methodical change;
- We should not wait or depend on technological break-throughs;
- Leading-edge productivity tools can play an important role in shortening development schedules, they are not sufficient alone.

Productivity Tool Strategy

- The use of PTs is best treated as a long-term strategic issue rather than a short-term tactical fix;
- Tool usage is not a short-term solution because it takes time & money to acquire & deploy tools effectively;
- If time & money are not spent, you stand to waste time & money using tools that you ultimately find ineffective;
Productivity Tool Strategy

Tool usage is also unlikely to provide any major competitive advantage as they are advertised heavily, therefore your competitors know about them too;

Being up-to-date in productivity tools will help you stay in the game, but not necessarily win;

If you deploy new tools in a haphazard way, the benefit you receive from them will wax & wane - there is not any point in jumping onto a cool band wagon three months ahead of competitors, if it is a crappy tool band wagon;

The strategy for acquiring & deploying new tools effectively should include the following elements:

- Early identification of promising new tools;
- Timely & accurate evaluation of new tools;
- Rapid deployment of new tools that are found to be ineffective;
- Non-deployment of new tools that are found to be ineffective;
- Continuing reliance on older, proven tools;

If you implement a program containing these elements within your organization, you will achieve a strategic competitive advantage;

The tools will be adopted more quickly and you will avoid the lost productivity associated with adopting the wrong tools;

Over time you will build on your successes by continuing to use, older, proven tools.
## Productivity Tool Acquisition

- Organizations that have random or causal methods of acquiring software tools waste about 50% of all the money they spend on tools;
- Organizations that use formal acquisition strategies can drop their wastage to about 10% and avoid associated schedule problems (Jones, 1994);

### Common Problems With Acquisition of Tools:

- The software market is prone to gimmicky & exaggerated claims;
- Bad-tool acquisition precludes acquisition of more beneficial tools;
- 30% of acquired tools do not meet enough user needs to be effective;
- 10% are never used after acquisition;
- 25% are used less than they could be because of lack of training;
- 15% are seriously incompatible with existing tools & trigger some form of modification to fit the new tool into the intended environment;

### Acquisition Plan

- An organization that waits until it needs a tool to begin researching has waited too long - tool evaluation & dissemination should be an ongoing activity via:
- A **tools group** who can conduct/be responsible for the following:
  - **Intelligence gathering** - information on cutting edge tools;
  - **Evaluation** - evaluate new tools on the market;
  - **Co-ordination** - co-ordinate tool experiments, learning;
  - **Dissemination** - of tool information to all;
Productivity Tool Acquisition

Acquisition Plan

- **Risks of Setting up a Tools Group**
  - Over-control - group only allows use of tools they have approved;
  - Group becomes a standards organization - instead of providing a service to other organization members;
  - Tools group needs to be staffed by people whose recommendations will be heard.

Productivity Tool Acquisition

Selection Criteria

- **Estimated gain** - you expect to realize from the use of the tool;

- **Vendor stability** - Can you stake your future on the vendor who provide the tools, can they support your future outlook?;

- **Quality** - if vendor’s tool is not qualitative and is buggy, so too will the program you try to develop;

- **Maturity** - tool maturity is a good indication of quality & vendor commitment, as a rule of thumb, some organizations will not purchase a tool until it has reached Version 3.0;

Productivity Tool Acquisition

Selection Criteria

- **Training time** - how much productivity time will you loose over the learning curve, depending on how many can use the tool, how many have to learn;

- **Applicability** - is the tool applicable or does it have to be force-fitted?;

- **Compatibility** - Does the new tool work well with the tools you are already using?;

- **Growth envelope** - will the tool support the directions in which you might want your product to go?
Productivity Tool Acquisition

Selection Criteria
- Customizing selection criteria - in defining a set of criteria to use in tool selection, be sure that you buy tools according to your own criteria rather than someone else's;

Commitment
- Once you have made the tool selection, commit to it, do not keep wondering if another tool would have been better.

Productivity Tool Use

When To Deploy
- On a software project, there is a trade-off between the learning curve you climb in becoming familiar with the new tool and the productivity you gain, once you become familiar;
- When new tools are deployed you incur overheads of training, experimentation, learning the tools weaknesses & arguing among team members about using the tool to it's full advantage;
- If you expect the project to take a substantial length of time, you can expect to recoup your learning curve investment;

Productivity Tool Use

When To Deploy
- If you expect the project to take a short length of time - you would be better off not adopting that tool from the point of view of that particular project;
- For a RD project on a long-term basis you would need to keep introducing new and more effective tools to improve productivity;
- On a short-term RD project, it is not suitable to introduce a new tool - the time you are in a hurry is not the time to introduce a new tool as a shortcut;
- Choose a less time sensitive project and soak up the learning curve.
Productivity Tool Use

Importance of Training

- The more powerful a tool is, the more difficult it can be to use effectively - without adequate training, tools that could provide significant productivity benefits often sit on shelves unused.

Productivity Tool Use

How Much Schedule reduction to Expect

- No practice is implemented in a vacuum, therefore, the environment in which the practice is used plays a large role in determining whether it succeeds;
- Vendor productivity claims are usually claims for maximum potential of a practice in an ideal environment;
- To estimate how much schedule reduction to expect from a productivity tool, think in terms of the entire lifecycle you plan to use, determine the effort you expect to save per phase.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Nominal effort</th>
<th>Decrease in effort</th>
<th>Final effort</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture (high level design)</td>
<td>8</td>
<td>80%</td>
<td>6</td>
<td>Same amount of architectures required</td>
</tr>
<tr>
<td>Detailed design</td>
<td>8</td>
<td>75%</td>
<td>2</td>
<td>75% reduction due to switch from 3GL to 4GL</td>
</tr>
<tr>
<td>Code debugging</td>
<td>8</td>
<td>75%</td>
<td>2</td>
<td>75% reduction due to switch from 3GL to 4GL</td>
</tr>
<tr>
<td>Unit test</td>
<td>6</td>
<td>95%</td>
<td>6</td>
<td>Same unit tests required for same functionality</td>
</tr>
<tr>
<td>Integration</td>
<td>6</td>
<td>75%</td>
<td>1</td>
<td>75% reduction due to switch from 3GL to 4GL</td>
</tr>
<tr>
<td>System test</td>
<td>6</td>
<td>95%</td>
<td>6</td>
<td>Same system tests required for same functionality</td>
</tr>
</tbody>
</table>
Productivity Tool Use

How Much Schedule reduction to Expect
- The switch from a 3GL to a 4GL language, is one of the most powerful productivity improvements you can make, but even when you can implement a program entirely in a 4GL, you only cut the schedule by about 25%.

Silver-Bullet Syndrome
- The biggest risk associated with software tool use is undoubtedly the SBS - the belief that a single tool or technology will by itself dramatically reduce development time;
- Switching to a new programming language, trying out a CASE tool, moving to O.O programming, have become classic exercises in wishful thinking;
- The desire to believe in the SBS on RD projects, is especially strong.

Silver-Bullet Syndrome
- The belief that a single tool can dramatically reduce development time leads us to try new tools, one at a time, rather than to take a more systematic approach;
- We adopt beneficial tools in serial rather than in parallel;
- We plan for short-term improvements & neglect long-term planning altogether;
- SBS is a causative factor for cancelled projects & cost overruns as well as excessive development times.
Silver-Bullet Syndrome

Identifying Silver Bullets

- You can dismiss any claim of an ability to improve productivity by more than 25% per year out of hand as a silver bullet claim;
- As time goes by, each tool is represented as a silver bullet, until people in the software industry become more familiar with them;
- Current silver bullets that do not live up to their claims include: 4GL’s, CASE Tools, RAD, O.O Programming.

Silver-Bullet Syndrome

Identifying Silver Bullets

- 4GL’s - can be powerful aids to software productivity, but they provide incremental gains, not revolutionary;
- CASE Tools - the ability of CASE tools to contribute to SD has been overrated, they can be beneficial in some environments, but in other organizations they amount to high-end tools for creating design diagrams;

Silver-Bullet Syndrome

Identifying Silver Bullets

- RAD (Rapid Application Development) - is an Information Systems set of practices that are somewhat adaptable to individual circumstances as a result of it’s specific areas of applicability, however, it is not much use outside of it’s database-centered IS systems orientation.
Silver-Bullet Syndrome

Identifying Silver Bullets

- **Object-Oriented Programming**: O.O technologies have not panned out the way people hoped they would. One survey found that O.O projects had dropped from a 92% success rate in 1991 to a 66% success rate in two years;
- It has delivered major benefits in the area of re-usability, but promised benefits in the areas of ease of use have been disproved;
- O.O integrates many of the best ideas of the last 35 years of SD, but learning to use it well is difficult;
- It should be viewed as an expert technology, valuable in SD toolbox.

Silver-Bullet Syndrome

Biting the Bullet

- Software people buy what they think are magic beans or tools whenever they believe claims that their best judgement tells them it is impossible;
- Vendors continue to claim that they have the ultimate tool that can solve all problems;
- Developers & managers continue to spend money on these tools - the $64 million question is - how many more times will this happen, time has not taught a lesson;

Silver-Bullet Syndrome

Biting the Bullet

- There is no magic, there is no point waiting for it, the more we wait, the more we deprive ourselves of valuable, incrementally better solutions to our problems;
- There are not any easy solutions, but there are easier solutions, which can provide modest improvements individually and dramatic improvements collectively.
Lecture Overview

- Role of productivity tools in Rapid Development;
- Productivity-tool strategy;
- Productivity-tool acquisition;
- Productivity-tool use;
- Silver-bullet syndrome
- Next Week: Classic Mistakes & Development Fundamentals.