

# *Software Process Improvement*



## *Killer Strategies for SPI Leaders*

*David F. Rico*

# *Goals and Objectives*



- Make a big splash in a hurry
- Succeed in spite of
  - Overwhelmingly impossible odds
  - Highly chaotic organizations
  - Unrealistic resource constraints
  - Extremely debilitating politics
  - Limited team vision and talent
  - Seemingly impervious resistance
  - Fierce competition/individualism
- Create a long lasting legacy

# *Do's and Don'ts*



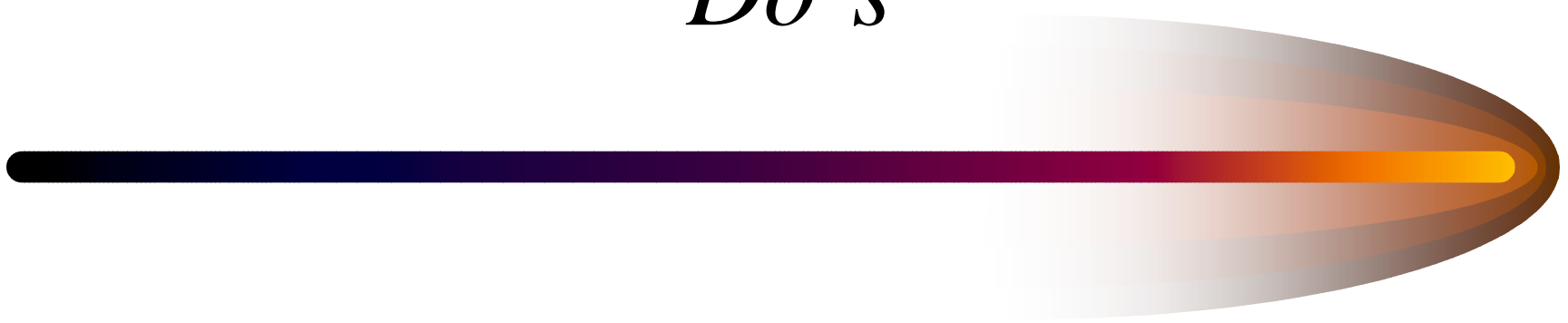
## *Do's*

- Create small, fast initiatives
- Design killer apps
- Build great websites
- Develop standard processes
- Deploy simplified methods
- Buy everything off-the-shelf
- Do much action, little talking

## *Don'ts*

- Play politics
- Form committees
- Create risky initiatives
- Use big bang methods
- Just meander along
- Reinvent the wheel
- Create a bureaucracy

*Do's*



# *Create Small, Fast Initiatives*



- Exploit power of time-boxed projects
- Create valuable products quickly
- Create portable enterprise standards
- Build highly visible monuments
- Make the most of limited resources
- Fly under radar of politics
- Make legacy before imminent departure

# *Examples*

- ★ Software policies and procedures
- ★ Websites with standard processes
- ★ Semi-automated workflow websites
- ★ Organizational metrics repositories
- ★ Intuitive, easy-to-use tools
- ★ Cost estimation/budgeting websites
- ★ Self-service verification/validation

# *Design Killer Apps*



- Exploit power of productivity tools
- Use inexpensive and useful tools
- Find intuitive, easy-to-use tools
- Find tools people want to use
- Embed transparent process in tools
- Embed transparent metrics in tools
- Use tools, not training and process

# *Examples*

- ★ Project planning wizards
- ★ Project management wizards
- ★ Document and deliverable wizards
- ★ Seamless reliability modeling
- ★ Seamless data and metrics capture
- ★ UML, IDEF1X, and code generators
- ★ Component and module reuse systems

# *Build Great Websites*



- Exploit power of web technologies
- Build great passive or active sites
- Use sites to propagate standards
- Use sites to collect metric data
- Use sites to engineer great process
- Explore navigable IDEF0-based sites
- Populate sites with great content

# *Examples*

- ★ Software cost estimation website
- ★ Software engineering website
- ★ IEEE 12207/15288 website
- ★ CMMI website
- ★ Software CMM website
- ★ Personal Software Process website
- ★ Team Software Process website

# *Develop Standard Processes*



- Exploit power of standard processes
- Create an enterprise-level process
- Create system and software process
- Don't forget auxiliary disciplines
- Use professional process principles
- Build consistent/verifiable process
- Use industry standards if possible

# *Examples*

- ★ IEEE 12207/15288 process
- ★ CMMI process
- ★ Software CMM process
- ★ Personal Software Process
- ★ Team Software Process
- ★ Rational Unified Process
- ★ Extreme Programming Process

# *Deploy Simplified Methods*



- Exploit power of simple methods
- Use intuitive, tool-driven methods
- Use point-n-click driven wizards
- Use database-driven expert systems
- Track project metrics transparently
- Use transparent statistical models
- Build tools to use without training

# *Examples*

- ★ PSP/TSP workflow
- ★ CMM/CMMI workflow
- ★ IEEE 12207/15288 workflow
- ★ Project management workflow
- ★ Quality/reliability estimating tools
- ★ RUP/UML workflow
- ★ Relational database design workflow

# *Buy Everything Off-the-Shelf*




- Exploit power of off-the-shelf tools
- Look for low-cost, high-value COTS
- Use de facto standard office suites
- Collect suite of static analyzers
- Use simple programming environments
- Use simple graphical drawing tools
- Identify as much freeware as possible

# *Examples*

- ★ Microsoft Office (not Framemaker)
- ★ Microsoft Visio (ubiquitous in use)
- ★ Relational database static analyzers
- ★ Free object oriented static analyzers
- ★ Free UML CASE Tools
- ★ Free PSP workflow tools
- ★ Free configuration management tools

# *Do Much Action, Little Talking*



- Exploit power of show-and-tell
- Build products, not committees
- Speak with accomplishments, not lips
- Build foundation standards for future
- Create usable and visible products
- Create products with high appeal
- Make the most with a limited staff

# *Examples*

- ★ Organizational software standards
- ★ Organizational policies/procedures
- ★ Organizational software websites
- ★ Organizational software databases
- ★ Push technologies for products
- ★ Newsletters for communication
- ★ Automated software distribution

*Don'ts*



# *Play Politics*



- Don't substitute politics for action
- Politics don't leave a lasting legacy
- Actions more important than words
- Engineers are good at ignoring words
- Everyone can see accomplishments
- Accomplishments silence the critics
- Politics more expensive than action

# *Examples*

- ★ Agreeing to all methods, using none
- ★ Management by walking around
- ★ Creating posters, slogans and sayings
- ★ Sending out occasional policies
- ★ Having endless meetings/interviews
- ★ Delegating initiatives to others
- ★ Creating decrees rather than tools

# *Form Committees*



- Don't form committees, form projects
- Committees are forums for politics
- Committees have managers not doers
- Projects have doers not managers
- Projects result in products
- Engineers do work, not managers
- Committees lack technical expertise

# *Examples*

- ★ Forming a hierarchy of committees
- ★ Forming a hierarchy of SEPGs
- ★ Forming splinter groups
- ★ Forming committees to investigate
- ★ Creating meetings, not products
- ★ Having lots of meetings, no action
- ★ Spinning wheels for many years

# *Create Risky Initiatives*



- Don't create large/complex projects
- Create small, low risk projects
- Don't create custom software
- Okay to create small custom websites
- Large projects are subject to failure
- Large projects subject to bad plans
- Large projects require large staffs

# *Examples*

- ★ Creating resource intensive projects
- ★ Large projects that hurt credibility
- ★ Large projects that have poor quality
- ★ Large projects that ignore COTS
- ★ Building products versus buying
- ★ Not completing a single project
- ★ Not building a single product

# *Use Big Bang Methods*



- Don't try to proselytize the planet
- Proselytizing is a waste of time
- Choose a few low profile projects
- Don't train the building at outset
- Fewer successes better than failure
- Form new organizational structures
- Hard to adapt to old culture

# *Examples*

- ★ Making 100s of projects use process
- ★ Training thousands of people
- ★ Making everyone accept one approach
- ★ Using resources before completion
- ★ Underestimating power of resistance
- ★ Not optimizing use of resources
- ★ Doing more than what is required

# *Just Meander Along*



- Don't meander along without a plan
- Wandering from week-to-week is bad
- Living without a schedule is futile
- No accomplishments without products
- Meandering causes much criticism
- Meandering leaves one vulnerable
- Meandering is a ticket to replacement

# *Examples*

- ★ Form committees instead of projects
- ★ Form committees without deliverables
- ★ Form committees without schedules
- ★ Form committees to talk things over
- ★ Form committees to debate politics
- ★ Form committees to show off
- ★ Form committees to share blame

# *Reinvent the Wheel*



- Don't overestimate your abilities
- Very few project management experts
- Very few quality management experts
- Very few life cycle experts
- Most people can't manage projects
- Most people can't estimate quality
- Most people don't understand metrics

# *Examples*

- ★ Building tools versus buying them
- ★ Creating methods versus adopting them
- ★ Ignoring creators of methodologies
- ★ Ignoring project management experts
- ★ Ignoring quality management experts
- ★ Ignoring industry standards
- ★ Creating custom methodologies

# *Create a Bureaucracy*



- Don't build a complex methodology
- Difficult methodologies won't be used
- Most people don't understand methods
- Most people won't use methods
- Hard to teach a difficult method
- Simple methods used without training
- Manual methods difficult to use

# *Examples*

- ★ Software Capability Maturity Model
- ★ Capability Maturity Model Integration
- ★ MIL-STD-498
- ★ ISO/IEEE 12207
- ★ Rational Unified Process
- ★ Extreme Programming
- ★ ISO 9001:2000