

*Screwtape's Guide to*  
**How to Fake a Test Project**  
*(without getting caught)*



As told to  
James Bach  
james@satisfice.com

# The Challenge

---

So, you want to release bad software, but you have to make it look as if you *really tried* to test it well...

**Here's how I would fake it!**

# **You could just lie, of course,** *testing is hard to supervise*

---

- Your boss probably doesn't watch you closely.
- Say you tested it, but spend most of your time playing Spider Solitaire, instead.
- Report a few minor bugs to keep the heat off.  
(A tester did this to me in 1987)
- *But what if you were going to be audited?*

# My Basic Strategy

---

- **Behave Conventionally**  
(don't worry: conventional testing wisdom is empty)
- **Squander Energy** (so that you can't test)
- **Focus Narrowly** (don't make eye contact with bugs)
- **Deflect Scrutiny** (don't avoid it; co-opt it)
- **Minimize Humanity** (humans are too good at testing)
- **Blame Complexity, Ambiguity, and Volatility**  
(argue that no one can cope with these things)

# **Important Ingredient:**

## Find Some Bugs

---

- This is not difficult. Anyone can find a few bugs.
- Small bugs, mostly.
- A few big ones.
- Report them just badly enough so that they will be ignored, but not too badly.

# My Self-Presentation

---

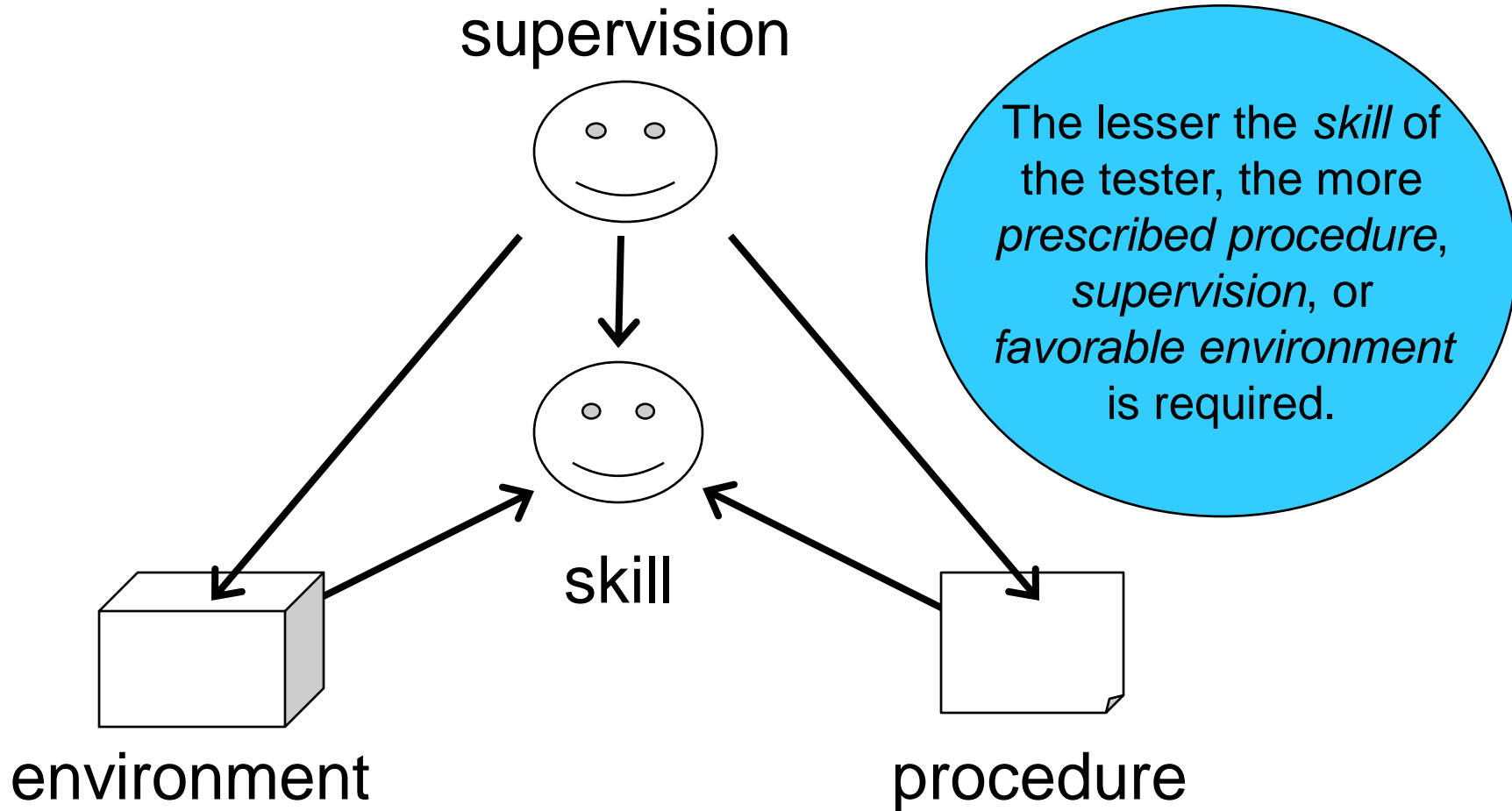
- I would call myself an “Engineer” and talk a lot about **“engineering discipline”**
- Or call myself “Quality Assurance” and talk a lot about **“best practices”** and **“process maturity”**
- Of course, I would also call myself an expert. It’s easy!
- Process maturity lets me to defend a slow and expensive process by featuring as a virtue its very slowness and expensiveness!
- **I would get ISTQB certification!**

**DANGER:** someone may realize that maturity is a moral concept that begs the question of what practices and skills are actually needed on the project.

**SOLUTION:** I accuse those people of being philosophers instead of practical like me. That way, I maintain hegemony over the “regular guy” mytheme within the axiological dialectic.

# Minimize Humanity

*It increases cost without raising suspicion*



# Thick Official Documents!

---

- Thickness discourages scrutiny.
- Templates give appearance of analysis.
- IEEE 829 is a faker's best friend!
- Contrast your handsome docs to the crude ones you receive.
- Make a big show of keeping them up to date.
- The time you spend on these documents will prevent you from testing.
- Consider computer generated docs! Cool!!

✓ Behave Conventionally	✓ Deflect Scrutiny
✓ Squander Energy	✓ Minimize Humanity
✓ Focus Narrowly	✓ Blame Complexity, Ambiguity, and Volatility



# Thick Official Documents!

- I will be sure to include in every document:

- *Title page*
- *Approvals page*
- *Version history*
- *Table of contents*
- *Introduction to the project*
- *Purpose of the document*
- *Document reference list*
- *Acronyms and definitions*
- *Chatty tutorial text to discourage review*
- *LOTS OF FORMATTING*

**Little useful content,  
but plenty of excuses  
for including it!**

✓ Behave Conventionally	✓ Deflect Scrutiny
✓ Squander Energy	Minimize Humanity
Focus Narrowly	✓ Blame Complexity, Ambiguity, and Volatility

# Detailed Scripted Test Procedures with Specific Expected Results Executed After Each Build by Unskilled Testers!

- This is the gold standard of testing fraud.
- Real expected results are impossible to document fully, so it's hard for people to accuse you of doing too little.
- Most managers think any intellectual process can and should be written down, so you are going with the flow.
- Make them simple function tests so that they are unlikely to find problems even the first time through.
- It helps to relocate the test team thousands of miles from the programmers.

**DANGER: Testers may accidentally find bugs because they don't follow the scripts precisely.**

**SOLUTION: Accuse them of lacking discipline and maturity.**

✓ Behave Conventionally	✓ Deflect Scrutiny
✓ Squander Energy	✓ Minimize Humanity
✓ Focus Narrowly	✓ Blame Complexity, Ambiguity, and Volatility

# Test Case and Pass Rate Metrics!

- Test cases are just containers, easily manipulated.
- Make your tests easy to pass, and all similar.
- It should not be difficult to produce thousands of them, just by using copy and paste.
- You need more than 1000 tests. Make the pass rate climb slowly.
- If necessary, restrict the oracles so that more pass.
- Golden Rule: *Make the graphs fit expectations.*

**DANGER:** They may realize this depends on easily manipulated assumptions.

**SOLUTION:** Remind them that EVERY commercial test management tool offers on test case metrics and ask “How could they all be wrong?”

✓ Behave Conventionally	✓ Deflect Scrutiny
✓ Squander Energy	✓ Minimize Humanity
✓ Focus Narrowly	Blame Complexity, Ambiguity, and Volatility

# Expensive GUI Test Automation!

---

1. Purchase an expensive GUI test execution tool.  
(see Rational, Mercury, Compuware, etc.)
2. Define a lot of paper test procedures.
3. Hire an automation team to automate each one.
4. Build a comprehensive test library and framework.
5. Keep fixing it.
6. BONUS: *Test Management Software*

✓ Behave Conventionally	✓ Deflect Scrutiny
✓ Squander Energy	✓ Minimize Humanity
✓ Focus Narrowly	✓ Blame Complexity, Ambiguity, and Volatility