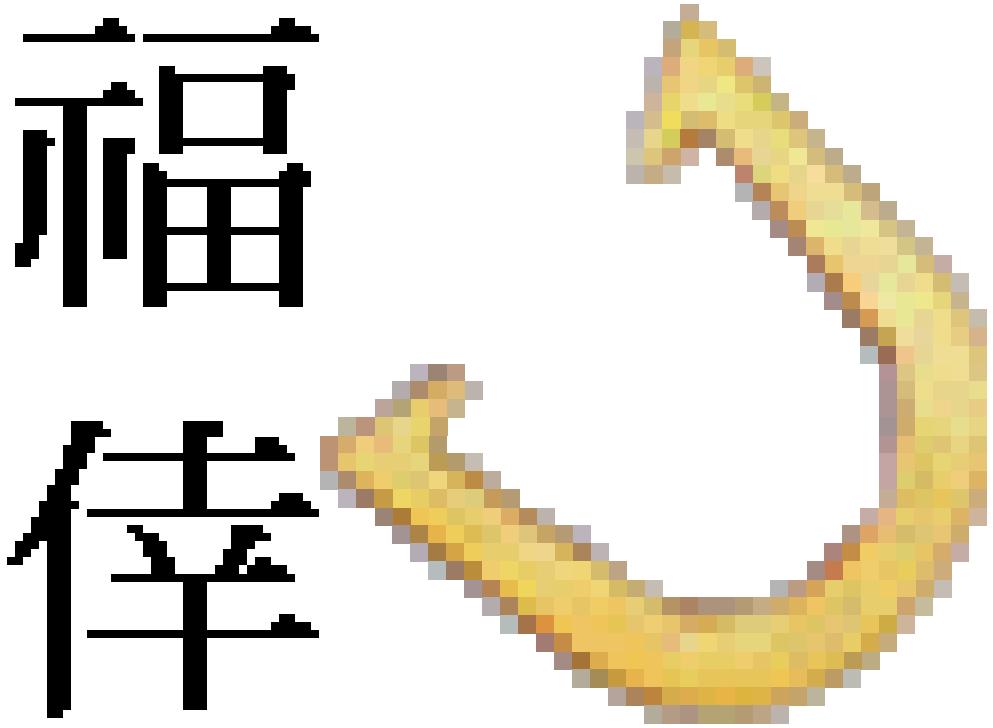
Create Your Own Luck

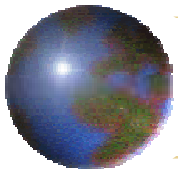
Get Organized for Test Success

福



Rex Black
RBCS, Inc.
31520 Beck Road
Bulverde, TX 78163 USA

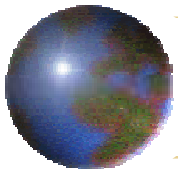
Phone: +1 (830) 438-4830
www.rexblackconsulting.com
rex_black@rexblackconsulting.com



"Do You Feel Lucky? Well, Do Ya?"

- Conference attendance: A great way to get new ideas for your test efforts
- Contextual factors that can make you lucky at adopting these ideas
  - Similar technologies (e.g., Java, mainframe)
  - Similar testing activities (e.g., load, performance)
  - Similar application domain (e.g., banking, Web)

What context-independent organizational factors can create good luck for your test team?



Four Lucky Organizational Factors

1. Clearly defined roles within—and interfaces between—test team and project
2. Early test team involvement in project
3. Sharing of test cases, data, and tools across test participants and phases (levels)
4. Project culture that promotes understanding and valuing test team's contributions

? How do these factors promote test success?

? How can we institute these auspicious circumstances on our projects?

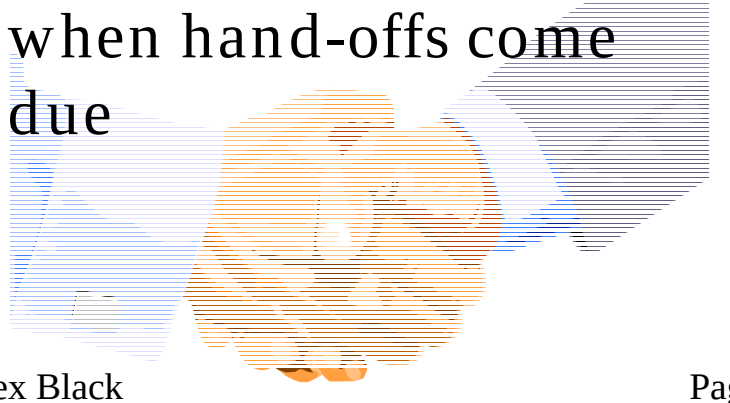
Clearly Defined Roles and Interfaces

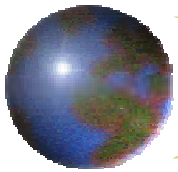
Lucky factor because…

- Lots of dependencies between test and project
  - SUT from developers
  - Test environment from sys. admin. / NOC
  - Bug reports to developers
  - Test status reports to project management
- Clear definitions prevent chaos, missed hand-offs, blame games

To create luck…

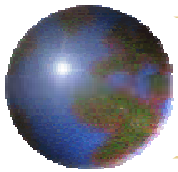
- Work with project team to build support for roles and interfaces
- Define roles and interfaces in test plan
- Reinforce roles and interfaces (tactfully) when hand-offs come due





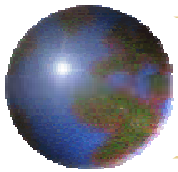
Case Study: Poorly Defined Roles

- One test manager found that no one on project was responsible for make installable releases from the course code
- She decided to accept role, since she needed builds for system testing
- She was able to figure out how to build test and customer releases, but unable to handle planned testing due to added workload
- Test effort was seen as a failure by peers and management, and executives



Case Study: Well-Defined Roles

- Another test manager was careful to define test release and test environment management processes for both client- and server-side software as well as hardware components
- Builds arrived on-time, as promised
- Problems with test release installation were seen as mutual problems
- No unanticipated downtime occurred due to unapproved lab reconfiguration



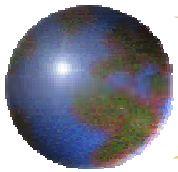
Early Involvement

Lucky factor because…

- Bug fix costs rise throughout project
- Some test development tasks take long time
- Good relationships built easier under low stress
- Early involvement allows earlier bug fixes, more thorough testing, stable testing context, honest, friendly dialogs

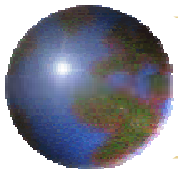
To create luck…

- Promote awareness in project re: advantages of early involvement
- Start on day 1 of project
- Have test team review requirements and design specifications
- Analyze quality risks, develop and sell an estimate, write test plan, create test context



Case Study: Late Involvement

- One test manager spent test team's time dealing with previous maintenance releases
- Test manager spent his (limited) time haranguing project about their bad processes
- Testers didn't get involved with new release until modules written, integration in progress
- Opportunities to create appropriate test context (including automation) were limited
- Test team was unable to fully contribute, seen as extraneous distraction by rest of project



Case Study: Up-front Involvement

- Another test manager allocated two test engineers at the beginning of development
- Testers reviewed early requirements / design specifications, finding ~100 errors / omissions
- Test team had complete testing context (quality risk analysis, plan, team, tools, cases, data, environment) ready for integration test
- Test team seen as a major player in project
- Test team brought credible assessments of quality to project status meetings

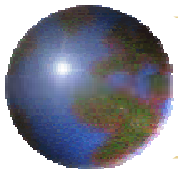
Sharing of Test Tools, Cases, and Data

Lucky factor because…

- Test tools, cases, data can take lots of effort
- Test artifacts used for unit/component test can be leveraged for integration/system test
- Sharing promotes re-use, reduces redundancy, builds cross-functional teamwork

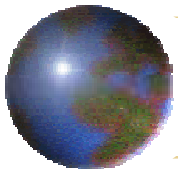
To create luck…

- Have test engineers work with developers on unit/component test
- Design test tools, cases, and data for re-use
- Build automated harnesses using COTS, freeware, or custom test frameworks



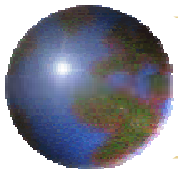
Case Study: Unshared Test Artifacts

- One development team built a (functional) unit test harness with no test participation
- This test framework was custom-built using special-skill tools for a specific environment
- Another development team built a load generator after rejecting test engineer advice
- This load generator was worthless for performance testing, being overly intrusive
- The test team had to recreate these tools, so over a person-year of effort was duplicated



Case Study: Regression and Smoke Test

- One development engineer worked with a test engineer to adapt an automated test harness to smoke test nightly builds
- This harness submitted 200+ queries against SUT (multi-OS / multi-DB reporting tool), compared them against baselines, e-mailed report to development and test teams
- Regressions detected during system test were greatly reduced, test cycles shortened, monthly maintenance releases made possible



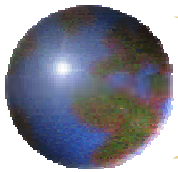
Test-Friendly Project Culture

Lucky factor because…

- Alignment of test effort with critical quality risks requires teamwork
- Test resources (time / $) often insufficient
- Expectations of testing benefits often unclear
- Promotes "important" testing, adequacy of test effort, use of test results for project tracking

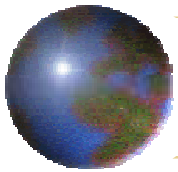
To create luck…

- Explain testing benefits to managers, executives
- Distinguish assessing versus assuring quality
- Involve appropriate stakeholders in test design, estimation, planning, development
- Apply quality risk management (e.g., FMEA, 9126) techniques



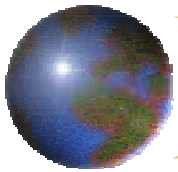
Case Study: Unrealistic Expectations

- One client wanted a test engineer to come in, , plan the test process, create a test context, build an automated test harnesses for Web and legacy apps, and train development team—at half her usual rate—in six weeks!
- One executive referred to his test manager as "Quality Assurance" manager and expected testing to make quality problems go away
- When project teams misunderstand testing, we can't help but fail…



Case Study: Pervasive Testing

- One test manager clarified expectations and project context as first step
- He used quality risk management with project team to determine project scope
- Test team worked with developers to develop test tools, data, and cases
- Test team helped marketing, customer support, and development define “correct”
- Test dashboard was key project indicator
- Test exit criteria became the ship criteria



Lucky You!

- Lucky testers work on project where they…
  - … Have clearly define roles and hand-offs
  - … Get involved early in the project
  - … Work cooperatively with developers for re-usable test tools, cases, data, and other artifacts
  - … Contribute valued and clearly understood quality risk management information services and products to a test-friendly project team

You can be lucky, too….because, really, there's no luck involved at all…just careful planning, calm and reasoned persuasion, and lots of attention to organizational details.