Elephant Bungee Jumping

A Software Project Manager's Survival Guide

The downs & (hopefully) ups of acquiring software

For the manager smart enough to know what they don't... (by authors smart enough to never admit to making the same mistake twice)

Authors

Dr. Robert Sandkaager
Mr. Emmett Rigsbee
Mr. August Nitzel
Mr. Robert (SAM) Johnson
Elephant Bungee Jumping

Why This is Important

- Many new and inexperienced Project Managers in the NRO
- Our history is hardware & large scale system integration *NOT SOFTWARE!*
- Most of our acquisition effort is now software related
- We’ve learned from Both Success and Failure
- We were told to!
- Not all Project Managers have the same qualifications/training/certification

- Not all Project Managers live and breathe the latest industry trends

- All Project Managers are accountable for the cost, schedule, and performance of their effort.
How Best to Emphasize

- How many guides does one subject need?
- How many guides look really good on a shelf?
- Project Managers are already struggling hard to stay current
- Technical demands require S/W process and domain expertise
- Humor is only funny if it is relevant AND NOT derogatory
- If you’ve never acquired S/W it is entertaining and informative
- If you have -- it’s better!
Part One

1. The Dating Game
   Understanding your users and managing their expectations

2. Easier Than Teaching Pigs to Fly
   Selecting and getting the most from your development team
Universal Truth #1

Involve users early, regularly, and often

... but in a managed way.
Universal Truth #2

Watch what they do and not what they say
(this also applies to management, both theirs and yours).
3 No Whips Allowed

Establishing and controlling requirements

4. Betting Your Career on Blind Faith

Identifying and controlling risk

5. A Whip, a Chair... And a Bag of Money

Managing cost and schedule

6. Help... I've Fallen and I Can't Get Up

Using configuration management to control your program's baseline

7. Metrics Mania

Making decisions based on performance measures
Universal Truth #3

If it's not written down, you don't get it. Verbal promises often last past the first "speed bump."
4  Betting Your Career On Blind Faith

Identifying & Controlling Risk

Universal Truth #4

The risks you manage determine the quality you deliver.
Universal Truth #5

If you don't know where you're going, any plan will get you there.
Using Configuration Management to Control Your Program's Baseline

Universal Truth #6
Software insolvency, when not managed, will lead to chaos when the 'rubber hits the road'.
Making Decisions Based on Performance Measures

Universal Truth #7

Base your decisions on performance measures.
If it’s not measured, you probably don’t care.
Speed Bumps & Potholes

Part Three

8. A Poke in the Eye with a Sharp Stick
Making modifications to commercial software . . . NOT!

9. Avoiding Diseases that Are Fun to Catch
Incorporating commercial products into your development

10. Cutting Edge and Bloody Mistakes
Avoiding unproven technologies in development

11. Move Over, You’re Obsolete
Hardware and operating system upgrades

12. Reheated Spaghetti Is Still Bad Code
Incorporating re-use in your development

13. No Free Lunch
Using automatic code generators to cut development time

14. You’ll Have Nobody to Blame but Yourself
The pitfalls of Government - furnished equipment

15. Self-Eating Watermelons
Using multiple development contractors
Making Modifications to Commercial Software . . . NOT!

Universal Truth #8
Never allow software developers to modify commercial operating systems or other COTS products.
Avoiding Diseases that Are Fun to Catch

Incorporating Commercial Products into Your Development

Universal Truth #9

COTS products are not necessarily the best solution. They bring as many risks as benefits. Understand both and make sure your developer does too before adopting.
Avoiding Unproven Technologies in Development

Universal Truth #10
Cutting-edge products lead to bleeding-edge projects.
Hardware & Operating System Upgrades

Universal Truth #11

Hardware and software system baselines (as well as operating systems) will become obsolete and require upgrading during the development cycle. GET OVER IT!
Reheated Spaghetti Is Still Bad Code

Incorporating Re-use in Your Development

Universal Truth # 12
Re-use costs extra in the initial development in terms of dollars and schedule as well as to the user in terms of system flexibility.
Using Automatic Code Generators to Cut Development Time

Universal Truth #13

Avoid using automatic code generators where timeline performance is critical.
Universal Truth #14

Government-furnished equipment shifts the risk from the developer to the Government. Leave the risk with the developer.
Self-Eating Watermelons

Using Multiple Development Contractors

Universal Truth #15
Avoid ping pong developments.
Part Four

Finishing Touches

16. Nothing Short of Complete Victory
Transitioning everything to operations

17. When the Air Tastes like Water, You Know You’re Drowning
Where to go for help
Transitioning Everything to Operations

Universal Truth #16
Deliver to the O&M organization all tools, drivers, and test data used during development.
When the Air Tastes Like Water, You Know You’re Drowning

When to Go for Help

Universal Truth #17
Know when to hold ‘em, when to fold ‘em, and when to RUN away.