



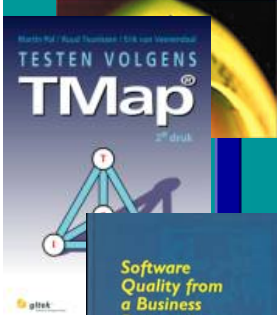
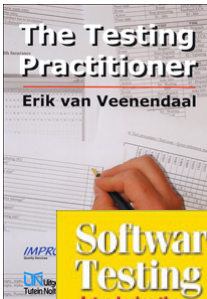
Building on Success



Beyond The Obvious

Erik van Veenendaal

(eve@improveqs.nl)



Erik van Veenendaal



- Founder and major shareholder Improve Quality Services
- In testing since 1989 working for many different clients and in many different roles
- Author of “TMap”, “ISTQB Foundations”, “The Testing Practitioner” and many other books and papers
- Vice-President International Software Testing Qualifications Board (ISTQB)
- Vice-Chair TMMi Foundation
- Keynote speaker EuroSTAR and STAReast
- Winner of the European Testing Excellence Award 2007



The Paradox

For many years only
25% of the IT projects
are successful

- We still release our systems (most of the times)
 - ‘A little bit too late’
 - ‘A few too many defects’
 - ‘Budgets are slightly overrun’
- How to be ‘good enough’ and ‘survive’
- This is NOT about how to be become better and the best!!



Core Practice # 1: Risk Analysis

ONE WAY 

Setting Clear Test Priorities

Testing is Risk Based - Hetzel

- Risk identification and analysis
- Determine a risk-based test approach
- Risk based reporting (stakeholders' language)

Setting Priorities: Risk Assessment

→ Risk identification

- Requirements based (e.g. PRISMA)
- Brainstorm workshops (e.g. FMEA)
- Failure history

→ Risk analysis

- Risk = impact x likelihood

» What is the impact for the business ?

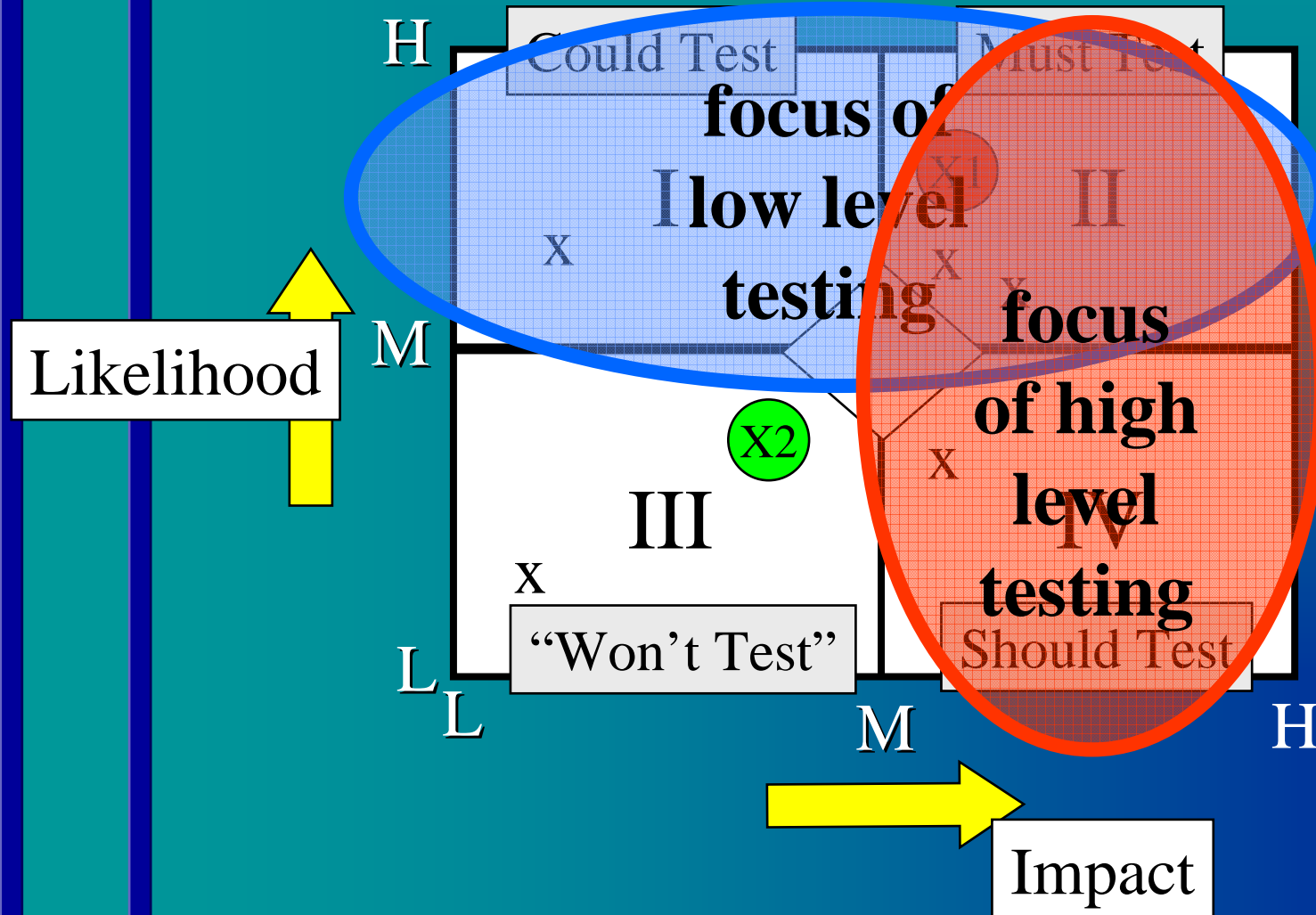
- damage, usage intensity, external visibility,

» What is the likelihood that there are defects ?

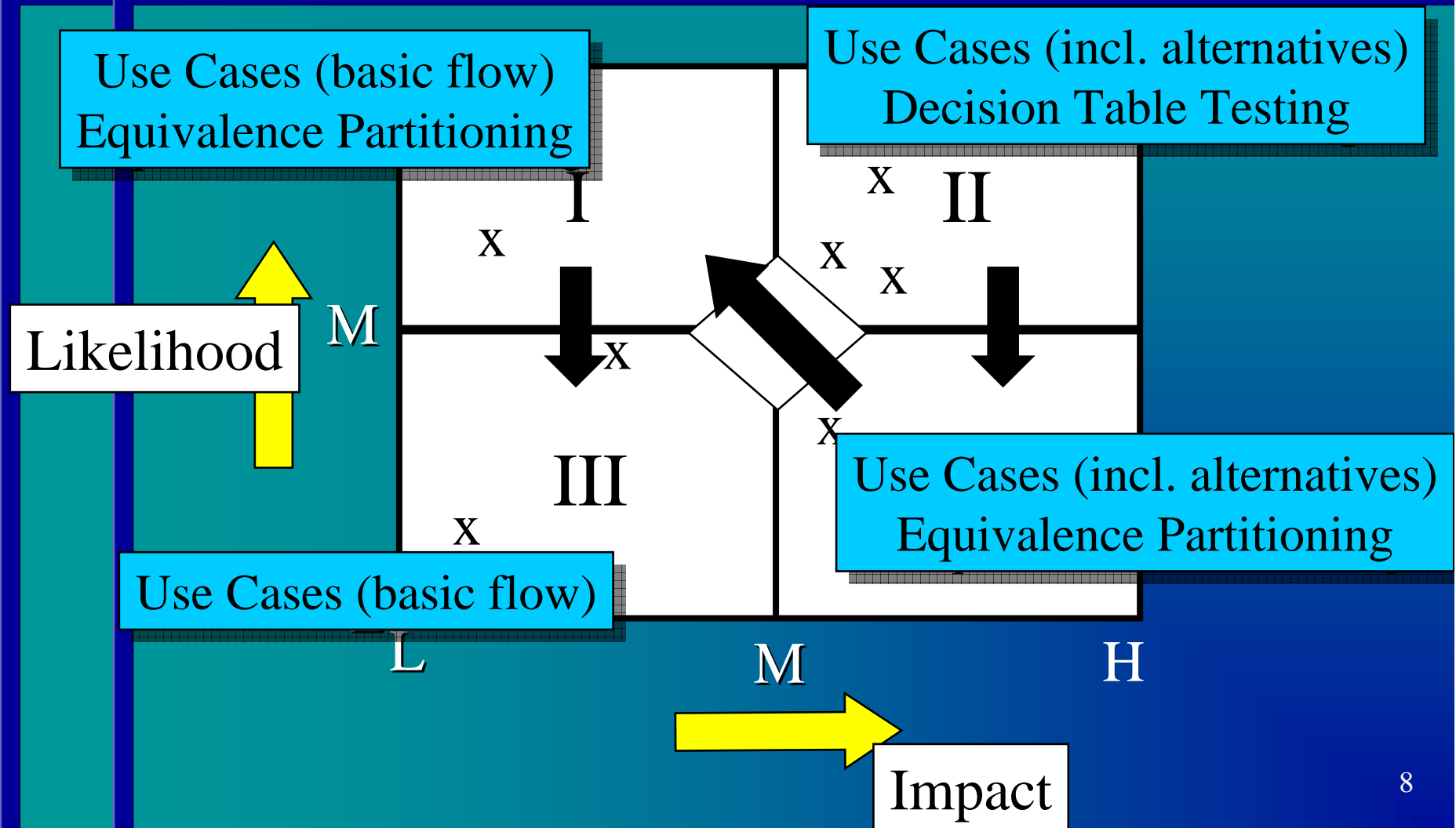
- new technology, complexity, interfacing,



The Product Risk Matrix



Example High Level Testing



Differentiated Test Approach !!

ONE WAY 

- Reviews & inspection
- Test design start-up meetings
- Reviews of test design
- Level of detail of test cases
- Exit criteria
- Level of independence
- Most experienced person
- Priority setting
- Re-testing
- Regression testing



*without this risk management
doesn't make much sense !!*

Risk Based Reporting (1)

	TS1	TS2	TS3	TS4	TS5	TS6	TS7	TS8	
Risk item 1	X	X	X						Red
Risk item 2	X			X	X		X		Red
Risk item 3	X		X						Yellow
Risk item 4					X	X	X		Yellow
Risk item 5								X	Yellow

Can we release the product?

Management view

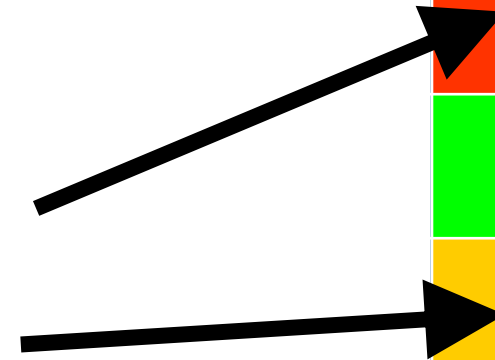
Risk item 1

Risk item 2

Risk item 3

Risk item 4

Risk item 5

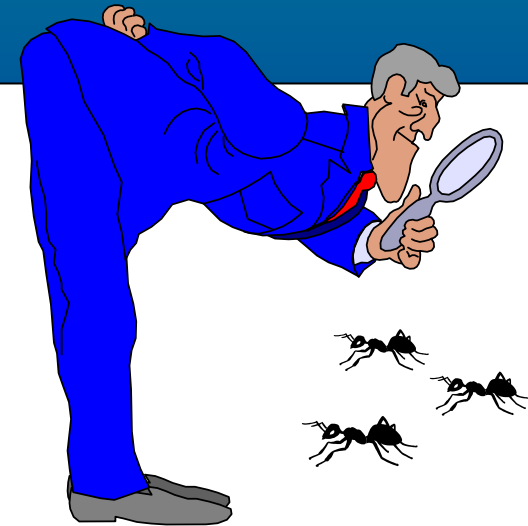


Core Practice # 2: Reviews

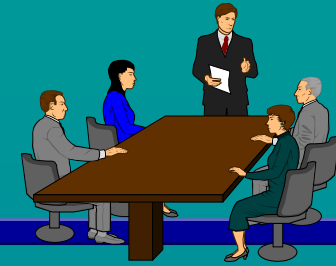
Testing is preventing defects - Hetzel

Upstream Focused Formal Reviews

- Walkthrough / Inspection
- Priority to the profitable
- Apply review practices that make the difference



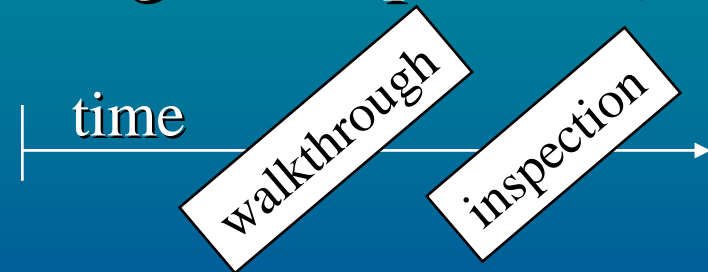
Why? and What?



- 60% of the defects have already been made before coding/implementation has started
 - Upstream defects are the most important
 - for costs, but also for project success
 - requirements, architecture documents
 - Focus on the high risk items
 - you usually can't afford anything else
 - a 'risk based' review plan helps enormously
- Limited number, but thorough reviews

Don't fool yourself

- Different review types for different objectives (e.g. Walkthrough / Inspection)



- Trained moderators
 - ‘moderator du jour’ does not work
- Apply roles and kick-off
- Entry check
- Not too many pages (± 20)
- Reasonable checking rate (± 10)

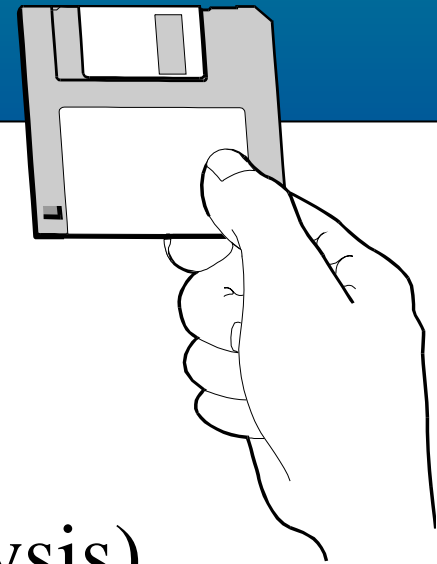


Core Practice # 3: Unit Testing

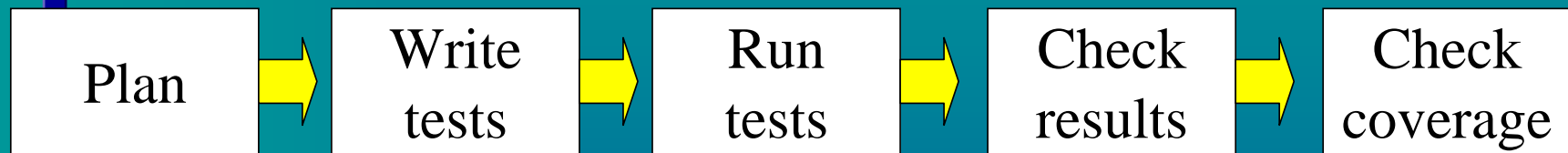
Testing requires independence - Hetzel

Apply Real Unit Testing

- Code Coverages
- Buddy Testing (XP)
- Different from debugging
- Pair Inspection (or Static Analysis)



Unit Test Process (BS7925-2)



Unit test plan

- technique
- environment
- **exit criteria**
- **level of independence**

Will often refer to:
Project Unit Test Plan

Unit test spec.

- test input
- expected result
- test objective

May act as:
Automated
Test Script

Unit test log

- test cases
- test results
- **test coverage**
- tester
- environment
- etc

Practical Implications

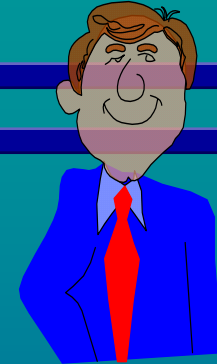
Remember: most high level testing will only achieve 30 - 40% code coverage

- Buddy (independent) testing
- Buy and install a coverage tool
- Use formal exit criteria
 - Risk based / Statement and Decision Coverage (be realistic!!)
- Free format test designs (code)
- Developers are testing and it is fun!!
- Higher quality delivered to the test team

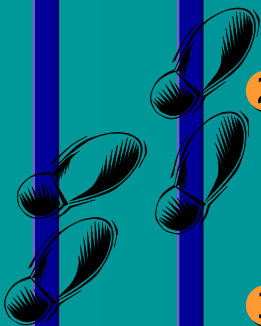


To be reviewed with
business & development

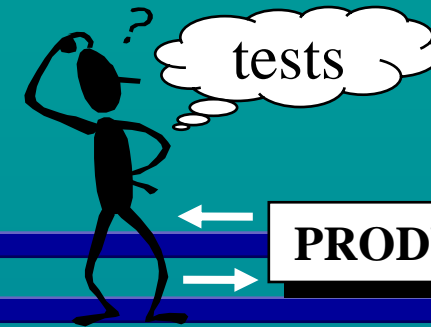
Test Use Cases



- Specification of user scenario's
 - Testing the most important business risks
 - Useful for functionality, usability, validation, integration, regression testing
- ① Identification of users (actors) of the system
 - internal roles of both internal and external customers
- ② Identification use cases for each actor
 - studying requirements, user-manual, interviewing users
- ③ Identification of paths for each use case
 - each use case has a basic, deviations and failure paths



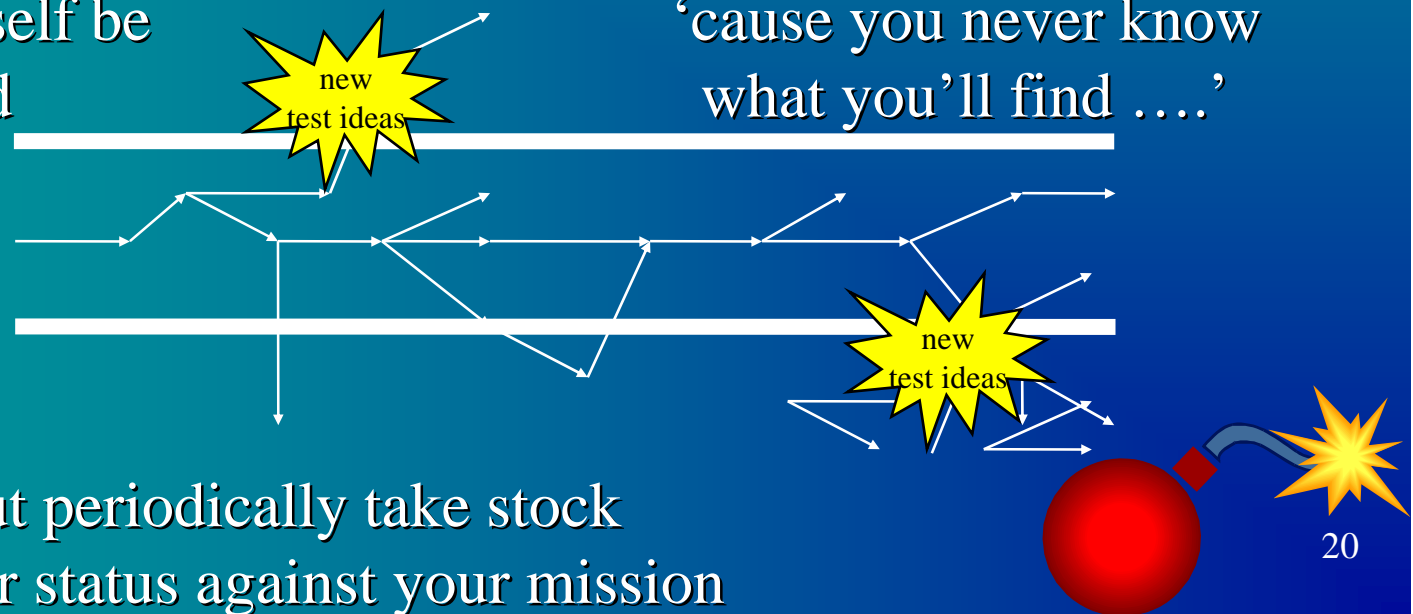
Exploratory Testing



- Simultaneous exploration, test design and execution
- Finding the most important defects in the time available

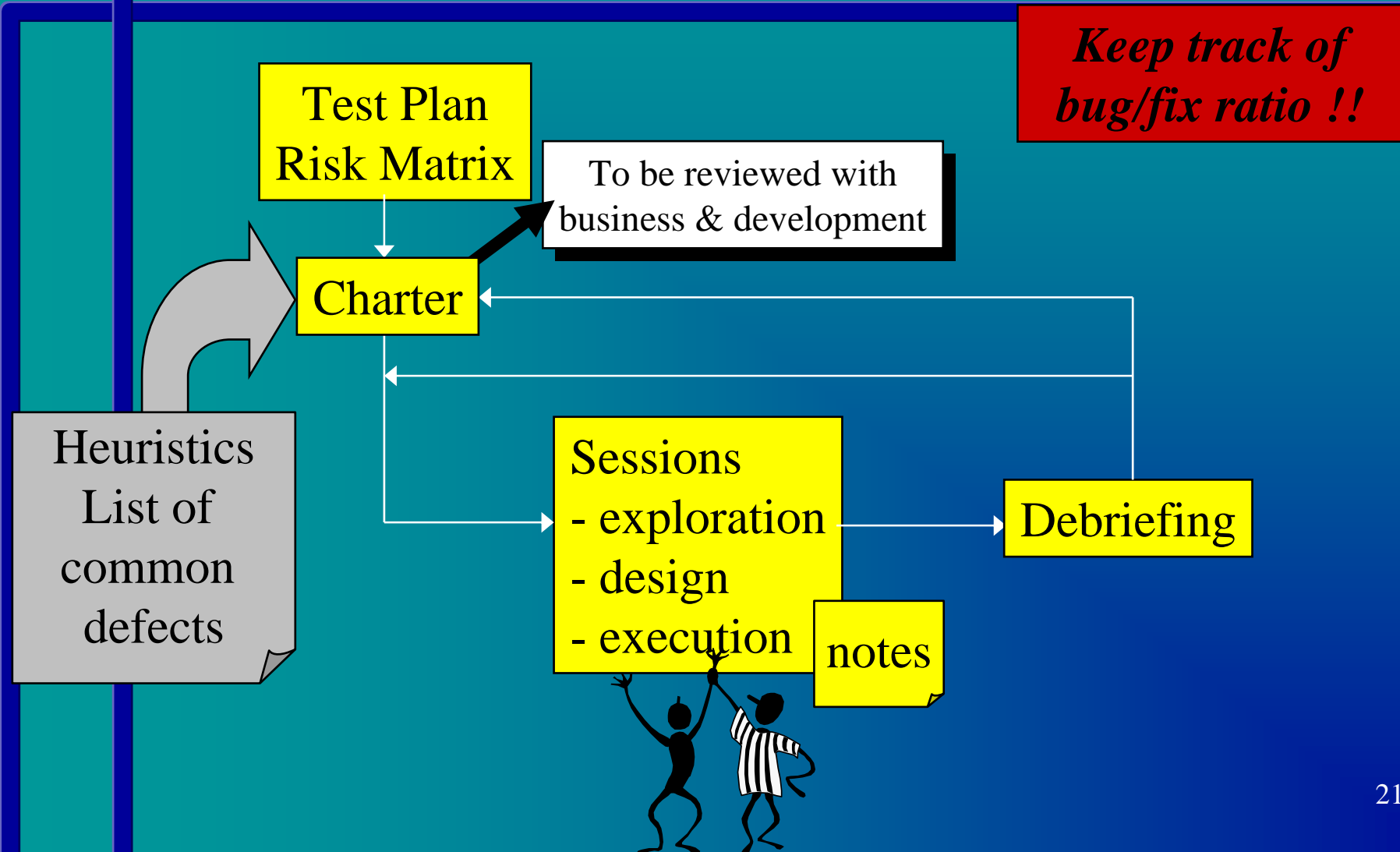
Let yourself be distracted

‘cause you never know what you’ll find’



But periodically take stock of your status against your mission

Exploratory Testing Process



Are you really doing ET or just EG?

- Are you using test charters?
- Are stakeholders involved in producing test charters?
- Is there a standard list of ideas / common defects?
- Are experienced testers involved that know all about test design?
- Do you work in pairs?
- Is the maximum length of a test session one day?
- Are test logs available from the test sessions?
- Are new risk, ideas and issues documented?
- Are test experiences discussed on a daily basis during debriefing sessions?



Core Practice # 5: People



Testing is an extremely complex and intellectually challenging task - Myers

Build Experienced & Skilled Testers

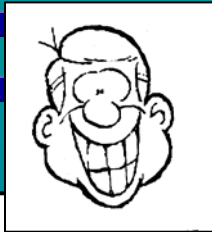
- Training (-on-the-job)
- Exchanging Experiences
- Test Team Dynamics
- Not only for Testers !!



Testing Skills

via (exchanging) Practical Experiences, Coaching and (formal) Training

Best practices and presentations



Test knowlegde

- test principles
- techniques
- tools, etc.



IT knowlegde

- software
- Requirements (IREB)
- CASE tooling

Domain knowlegde

- business process
- user characteristics

Soft skills

- communication
- critical mindset
- presentation & reporting



Test Team Dynamics

Effective teams don't just happen!

- Correct experience and skills
- Correct attitude
- Common purpose
- Team management
- The right team mixture, e.g. Belbin profiles

Balanced teams are more effective

Nobody is perfect but a team can be



Summary



- How to become ‘good enough’
 - *Risk-Based Test Approach & Reporting*
 - *Upstream Focused Formal Reviews*
 - *Apply Real Unit Testing*
 - *Use Informal Techniques Formally*
 - *Build Experienced & Skilled Testers*
- The next time it will be about how to become **better** and the **best**





Questions??