Interpreting a successful testing process: risk and actual coverage

Mariëlle Stoelinga, Mark Timmer
University of Twente

7th Workshop on Quantitative Aspects of Programming Languages
March 29, 2009
Contents

1 Introduction

2 The WFS Model

3 Risk

4 Other Applications

5 Limitations and Possibilities

6 Conclusions and Future Work
## Why testing?

- Software becomes more and more complex
- Research showed that billions can be saved by testing better
- No need for the source code (black-box perspective)
Why testing?

- Software becomes more and more complex
- Research showed that billions can be saved by testing better
- No need for the source code (black-box perspective)

Model-based testing

- Precise and formal
- Automatic generation and evaluations of tests
- Repeatable and scientific basis for product testing
Why do we need risk and coverage?

- Testing is inherently incomplete
- Testing does increase our confidence in the system
- A notion of *quality* of a test suite is necessary
- Two fundamental concepts: risk and coverage
Why do we need risk and coverage?

- Testing is inherently incomplete
- Testing does increase our confidence in the system
- A notion of *quality* of a test suite is necessary
- Two fundamental concepts: risk and coverage
Introduction – Risk and coverage

Why do we need risk and coverage?

- Testing is inherently incomplete
- Testing does increase our confidence in the system
- A notion of *quality* of a test suite is necessary
- Two fundamental concepts: risk and coverage
Why do we need risk and coverage?

- Testing is inherently incomplete
- Testing does increase our confidence in the system
- A notion of *quality* of a test suite is necessary
- Two fundamental concepts: risk and coverage

Informal calculation
Why do we need risk and coverage?

- Testing is inherently incomplete
- Testing does increase our confidence in the system
- A notion of *quality* of a test suite is necessary
- Two fundamental concepts: risk and coverage

Informal calculation

Coverage: $\frac{6}{13} = 46\%$
Introduction – Risk and coverage

Why do we need risk and coverage?

- Testing is inherently incomplete
- Testing does increase our confidence in the system
- A notion of quality of a test suite is necessary
- Two fundamental concepts: risk and coverage

Informal calculation

Coverage: \( \frac{6}{13} = 46\% \)

Risk: \( 7 \cdot 0.1 \cdot \$10 = \$7 \)
Introduction – Existing approaches

Existing coverage measures

- Statement coverage
- State/transition coverage

Limitations:
- All faults are considered of equal severity
- Likely locations for fault occurrence are not taken into account
### Existing coverage measures

<table>
<thead>
<tr>
<th>Statement coverage</th>
<th>State/transition coverage</th>
</tr>
</thead>
</table>

Limitations:

- all faults are considered of equal severity
- likely locations for fault occurrence are not taken into account
- syntactic point of view
### Existing coverage measures

- Statement coverage
- State/transition coverage

**Limitations:**
- all faults are considered of equal severity
- likely locations for fault occurrence are not taken into account
- syntactic point of view

### Existing risk measures

- Bach
- Amland
## Existing coverage measures

- Statement coverage
- State/transition coverage

Limitations:
- all faults are considered of equal severity
- likely locations for fault occurrence are not taken into account
- syntactic point of view

## Existing risk measures

- Bach
- Amland

Limitations:
- Informal
- Based on heuristics
- Only identify testing order for components
Starting point: semantic coverage

Previous work by Brandán Briones, Brinksma and Stoelinga
- System considered as black box
- Semantic point of view
- Fault weights
Starting point: semantic coverage

Previous work by Brandán Briones, Brinksma and Stoelinga
- System considered as black box
- Semantic point of view
- Fault weights

Labelled transition systems

\[
\begin{align*}
&s_1 & & 10\text{ct}? & & 20\text{ct}? & & s_2 \\
&s_0 & & \text{tea!} & & \text{coffee!} \\
&\text{coffee!} & & \text{tea!} & & \text{coffee!}
\end{align*}
\]
Starting point: semantic coverage

Previous work by Brandán Briones, Brinksma and Stoelinga

- System considered as black box
- Semantic point of view
- Fault weights

Labelled transition systems

\[ s_0 \xrightarrow{\delta} 10\text{ct?} \quad 20\text{ct?} \quad \text{tea!} \xrightarrow{\delta} s_0 \quad \text{coffee!} \xrightarrow{\delta} s_1 \quad \text{coffee!} \xrightarrow{\delta} s_2 \]
Starting point: semantic coverage

Previous work by Brandán Briones, Brinksma and Stoelinga

- System considered as black box
- Semantic point of view
- Fault weights

Labelled transition systems

\[ \delta \]

\[ s_1 \xrightarrow{10\text{ct}?} s_0 \xrightarrow{20\text{ct}?} s_2 \]

\[ \text{coffee!} \quad \text{tea!} \quad \text{coffee!} \]

10ct? coffee! 20ct? tea! $\delta$
Starting point: semantic coverage

Previous work by Brandán Briones, Brinksma and Stoelinga

- System considered as black box
- Semantic point of view
- Fault weights

Labelled transition systems

Test cases

Interpreting a successful testing process: risk and actual coverage
Starting point: semantic coverage

Previous work by Brandán Briones, Brinksma and Stoelinga

- System considered as black box
- Semantic point of view
- Fault weights

Labelled transition systems

Test cases
Starting point: semantic coverage

Previous work by Brandán Briones, Brinksma and Stoelinga
- System considered as black box
- Semantic point of view
- Fault weights

Labelled transition systems

Test cases

Interpreting a successful testing process: risk and actual coverage
Starting point: semantic coverage

Previous work by Brandán Briones, Brinksma and Stoelinga

- System considered as black box
- Semantic point of view
- Fault weights

Labelled transition systems

\[
\begin{align*}
S_1 & \xrightarrow{\delta} S_0 & S_0 \xrightarrow{\delta} S_2 \\
10\text{ct}? \quad 20\text{ct}? & \quad \text{coffee!} & \text{tea!} \quad \text{coffee!}
\end{align*}
\]

Test cases

\[
\begin{align*}
10\text{ct}? & \quad \delta \quad \text{coffee!} \quad \text{tea!} \\
\text{fail} & \quad \text{fail}
\end{align*}
\]
A WFS consists of

- An LTS (expected system behaviour)
- An error function (probability of faults)
- A weight function (severity of faults)
A WFS\(^-\) consists of

- An LTS (expected system behaviour)
- An error function (probability of faults)
- A weight function (severity of faults)
A WFS consists of
- An LTS (expected system behaviour)
- An error function (probability of faults)
- A weight function (severity of faults)

\[
p_{\text{err}}(10\text{ct? coffee!}) = 0.02
\]
\[
p_{\text{err}}(20\text{ct? tea!}) = 0.03
\]
A \textbf{WFS} consists of

- An LTS (expected system behaviour)
- An error function (probability of faults)
- A weight function (severity of faults)

\[ p_{\text{err}}(10\text{ct? coffee!}) = 0.02 \]
\[ p_{\text{err}}(20\text{ct? tea!}) = 0.03 \]

\[ w(\epsilon) = 10 \]
\[ w(10\text{ct?}) = 15 \]
\[ w(10\text{ct? coffee!}) = 9.5 \]
A WFS consists of
- An LTS (expected system behaviour)
- An error function (probability of faults)
- A weight function (severity of faults)

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$

$\delta$
Fault weight: 10 + 15 = 25
(We are only interested in whether a fault can occur, not in which one)

Interpreting a successful testing process: risk and actual coverage

The WFS Model – Fault Weight

The WFS Model

Nov. 27, 2008 8 / 18
Interpreting a successful testing process: risk and actual coverage

The WFS Model – Fault Weight

Fault weight: 10 + 15 = 25
(We are only interested in whether a fault can occur, not in which one)

\[ w(\epsilon) = 10 \]
\[ w(10\text{ct?}) = 15 \]
\[ w(10\text{ct? coffee!}) = 9.5 \]
The WFS Model – Fault Weight

\[
\begin{align*}
\delta \\
\delta
\end{align*}
\]

Interpreting a successful testing process: risk and actual coverage
Fault weight: \( 10 + 15 = 25 \)
Fault weight: $10 + 15 = 25$

(We are only interested in whether a fault can occur, not in which one)
Given a test suite $T$ and a passing execution $E$, we define

$$\text{risk}(T, E) = \mathbb{E}[w(\text{Impl}) | \text{observe } E]$$

i.e., the fault weight still expected to be present after observing $E$. 
Definition

Given a test suite $T$ and a passing execution $E$, we define

$$\text{risk}(T, E) = \mathbb{E}[w(\text{Impl}) \mid \text{observe } E]$$

i.e., the fault weight still expected to be present after observing $E$.

Observe:

$$\text{risk}(\langle \rangle, \langle \rangle) =$$
Given a test suite $T$ and a passing execution $E$, we define

$$\text{risk}(T, E) = \mathbb{E}[w(\text{Impl}) \mid \text{observe } E]$$

i.e., the fault weight still expected to be present after observing $E$.

Observe:

$$\text{risk}(\langle \rangle, \langle \rangle) = \sum_{\sigma} w(\sigma) \cdot p_{\text{err}}(\sigma)$$
Nondeterministic output behaviour yields difficulties.

How to calculate risk (expected fault presence)?
Risk

Nondeterministic output behaviour yields difficulties.

How to calculate risk (expected fault presence)?

Interpreting a successful testing process: risk and actual coverage
Risk

Nondeterministic output behaviour yields difficulties.

How to calculate risk (expected fault presence)?

Interpreting a successful testing process: risk and actual coverage
Nondeterministic output behaviour yields difficulties.
Nondeterministic output behaviour yields difficulties.

How to calculate risk (expected fault presence)?
Nondeterministic output behaviour yields difficulties.

How to calculate risk (expected fault presence)?

$$\text{risk}(\langle \rangle, \langle \rangle) = \sum_{\sigma} w(\sigma) \cdot p_{\text{err}}(\sigma)$$
Nondeterministic output behaviour yields difficulties.

How to calculate risk (expected fault presence)?

\[
\text{risk}(T, E) = \sum_{\sigma \neq 10\text{ct}?} w(\sigma) \cdot p_{\text{err}}(\sigma) + f(10\text{ct}?)
\]
Nondeterministic output behaviour yields difficulties.

How to calculate risk (expected fault presence)?

\[
\text{risk}(T, E) = \sum_{\sigma \neq 10\text{ct?}} w(\sigma) \cdot p_{\text{err}}(\sigma) + f(10\text{ct?})
\]
Nondeterministic output behaviour yields difficulties.

How to calculate risk (expected fault presence)?

\[ \text{risk}(T, E) = \sum_{\sigma \neq 10\text{ct}?} w(\sigma) \cdot p_{\text{err}}(\sigma) + w(10\text{ct}?) \cdot \mathbb{P}[\text{error after 10ct?} \mid E] \]
A weighted fault specification (WFS) consists of:

- An LTS (expected system behaviour)
- An error function (probability of faults)
- A weight function (severity of faults)
- A failure function (probability of failure in case of fault)
Weighted fault specifications (revisited)

A WFS consists of:

- An LTS (expected system behaviour)
- An error function (probability of faults)
- A weight function (severity of faults)
- A failure function (probability of failure in case of fault)

\[
\delta(s_0, 10ct?) = s_1
\]

\[
\delta(s_0, 20ct?) = s_2
\]

Interpreting a successful testing process: risk and actual coverage
A WFS consists of

- An LTS (expected system behaviour)
- An error function (probability of faults)
- A weight function (severity of faults)
- A failure function (probability of failure in case of fault)

\[
p_{\text{fail}}(\epsilon) = 1.0
\]

\[
p_{\text{fail}}(10\text{ct?}) = 0.5
\]
\[ p_{\text{fail}}(\epsilon) = 1.0 \]
\[ p_{\text{fail}}(10\text{ct?}) = 0.5 \]

\[ P[\text{error after 10ct?} \mid \text{observation of } E] \]
\[
\Pr[A \mid B] = \frac{\Pr[B \mid A] \cdot \Pr[A]}{\Pr[B]}
\]

\(\Pr[\text{error after 10ct?} \mid \text{observation of } E] = \Pr[\text{error after 10ct?} \mid \text{correct after 10ct? once}] \]

\[
\Pr[\text{correct after 10ct? once} \mid \text{error after 10ct?}] \cdot \Pr[\text{error after 10ct?}] = \frac{\Pr[\text{correct after 10ct? once} \mid \text{error after 10ct?}]}{\Pr[\text{correct after 10ct? once}]}
\]

\(p_{\text{fail}}(\epsilon) = 1.0\)

\(p_{\text{fail}}(10\text{ct?}) = 0.5\)
\[
\mathbb{P}[\text{error after 10ct?} \mid \text{observation of } E] \\
= \mathbb{P}[\text{error after 10ct?} \mid \text{correct after 10ct? once}] \\
\overset{\text{Bayes}}{=} \frac{\mathbb{P}[\text{correct after 10ct? once} \mid \text{error after 10ct?}] \cdot \mathbb{P}[\text{error after 10ct?}]}{\mathbb{P}[\text{correct after 10ct? once}]} \\
= (1 - p_{\text{fail}}(10\text{ct?}))^{1} \cdot p_{\text{err}}(10\text{ct?})
\]

\[
p_{\text{fail}}(\epsilon) = 1.0 \\
p_{\text{fail}}(10\text{ct?}) = 0.5
\]
\[ P[A] = P[A \mid B] \cdot P[B] + P[A \mid \neg B] \cdot P[\neg B] \]

\[ p_{\text{fail}}(\epsilon) = 1.0 \]
\[ p_{\text{fail}}(10\text{ct?}) = 0.5 \]

\[ P[\text{error after 10ct?} \mid \text{observation of } E] \]
\[ = P[\text{error after 10ct?} \mid \text{correct after 10ct? once}] \]
\[ \text{Bayes} \Rightarrow P[\text{correct after 10ct? once} \mid \text{error after 10ct?}] \cdot P[\text{error after 10ct?}] \]
\[ = \frac{(1 - p_{\text{fail}}(10\text{ct?}))^1 \cdot p_{\text{err}}(10\text{ct?})}{P[\text{correct after 10ct? once}]} \]
\[ P[A] = P[A \mid B] \cdot P[B] + P[A \mid \neg B] \cdot P[\neg B] \]

\[ p_{\text{fail}}(\epsilon) = 1.0 \]
\[ p_{\text{fail}}(10\text{ct}?) = 0.5 \]

\[
\begin{align*}
P[\text{error after 10ct? \mid observation of } E] &= P[\text{error after 10ct? \mid correct after 10ct? once}] \\
&\overset{\text{Bayes}}{=} \frac{P[\text{correct after 10ct? once} \mid \text{error after 10ct?}] \cdot P[\text{error after 10ct?}]}{P[\text{correct after 10ct? once}]} \\
&= \frac{(1 - p_{\text{fail}}(10\text{ct}?)^1 \cdot p_{\text{err}}(10\text{ct}?)}{(1 - p_{\text{fail}}(10\text{ct}?)^1 \cdot p_{\text{err}}(10\text{ct}?)}
\end{align*}
\]
\[ P[A] = P[A \mid B] \cdot P[B] + P[A \mid \neg B] \cdot P[\neg B] \]

\[
\begin{align*}
\mathbb{P}[\text{error after 10ct?} \mid \text{observation of } E] &= \mathbb{P}[\text{error after 10ct?} \mid \text{correct after 10ct? once}] \\
&= \frac{\mathbb{P}[\text{correct after 10ct? once} \mid \text{error after 10ct?}] \cdot \mathbb{P}[\text{error after 10ct?}]}{\mathbb{P}[\text{correct after 10ct? once}]} \\
&= \frac{(1 - p_{\text{fail}}(10\text{ct?}))^1 \cdot p_{\text{err}}(10\text{ct?})}{(1 - p_{\text{fail}}(10\text{ct?}))^1 \cdot p_{\text{err}}(10\text{ct?}) + (1 - p_{\text{err}}(10\text{ct?}))}
\end{align*}
\]

\[ p_{\text{fail}}(\epsilon) = 1.0 \]
\[ p_{\text{fail}}(10\text{ct?}) = 0.5 \]
Interpreting a successful testing process: risk and actual coverage

\[
\text{risk}(T, E) = \sum_{\sigma \neq 10ct?} w(\sigma) \cdot p_{\text{err}}(\sigma) + w(10ct?) \cdot \mathbb{P}[\text{error after 10ct?} \mid E]
\]
risk( \(T, E\) )

\[= \sum_{\sigma \neq 10ct?} w(\sigma) \cdot p_{err}(\sigma) + w(10ct?) \cdot \mathbb{P}[\text{error after 10ct?} \mid E] \]

\[= \sum_{\sigma \neq 10ct?} w(\sigma) \cdot p_{err}(\sigma) + w(10ct?) \cdot \frac{(1 - p_{\text{fail}}(10ct?))^{1} \cdot p_{\text{err}}(10ct?)}{(1 - p_{\text{fail}}(10ct?))^{1} \cdot p_{\text{err}}(10ct?) + (1 - p_{\text{err}}(10ct?))} \]
risk(\(T, E\))
\[
= \sum_{\sigma \neq 10ct?} w(\sigma) \cdot p_{\text{err}}(\sigma) + w(10ct?) \cdot \mathbb{P}[\text{error after } 10ct? | E]
\]
\[
= \sum_{\sigma \neq 10ct?} w(\sigma) \cdot p_{\text{err}}(\sigma) + w(10ct?) \cdot \frac{(1 - p_{\text{fail}}(10ct?))^n \cdot p_{\text{err}}(10ct?)}{(1 - p_{\text{fail}}(10ct?))^n \cdot p_{\text{err}}(10ct?) + (1 - p_{\text{err}}(10ct?))}
\]
Calculation of risk

\[
\text{risk}(T, E) = \text{risk}(\langle \rangle, \langle \rangle) - \sum_{\sigma \in E} w(\sigma) \cdot \left( p_{\text{err}}(\sigma) - \frac{(1 - p_{\text{fail}}(\sigma))^{\text{obs}(\sigma, E)} \cdot p_{\text{err}}(\sigma)}{(1 - p_{\text{fail}}(\sigma))^{\text{obs}(\sigma, E)} \cdot p_{\text{err}}(\sigma) + 1 - p_{\text{err}}(\sigma)} \right)
\]

with \(\text{obs}(\sigma, E)\) the number of observations in \(E\) after \(\sigma\).
Calculation of risk

\[
\text{risk}(T, E) = \text{risk}(\langle \rangle, \langle \rangle) - \sum_{\sigma \in E} w(\sigma) \cdot \left( p_{\text{err}}(\sigma) - \frac{(1 - p_{\text{fail}}(\sigma))^{\text{obs}(\sigma, E)} \cdot p_{\text{err}}(\sigma)}{(1 - p_{\text{fail}}(\sigma))^{\text{obs}(\sigma, E)} \cdot p_{\text{err}}(\sigma) + 1 - p_{\text{err}}(\sigma)} \right)
\]

with \(\text{obs}(\sigma, E)\) the number of observations in \(E\) after \(\sigma\).

Although \(\text{risk}(\langle \rangle, \langle \rangle) = \sum_{\sigma} w(\sigma) \cdot p_{\text{err}}(\sigma)\) seems infinite, it can be calculated smartly:

- \(w\) defined by truncation: the sum is already finite
- \(w\) defined by discounting: system of linear equations
Optimisations

- Find the optimal test suite of a given size
- Apply history-dependent backwards induction (Markov Decision Theory)
Other Applications

Optimisations
- Find the optimal test suite of a given size
- Apply history-dependent backwards induction (Markov Decision Theory)

Actual Coverage
- Only consider the traces that were actually tested
- Use error probability reduction as coverage measure
- Methods very similar to risk
Probabilities might be hard to find, but

- We show what can be calculated, and the required ingredients
- We facilitate sensitivity analysis
- To compute numbers, we have to start with numbers...
### Probabilities might be hard to find, but
- We show what can be calculated, and the required ingredients
- We facilitate sensitivity analysis
- To compute numbers, we have to start with numbers... 

### It looks like we need many probabilities and weights, but
- The framework can be applied at higher levels of abstraction
- Compute risk based on error / failure probabilities of modules
## Main results

- Formal notion of risk
- Both evaluation of risk *and* computation of optimal test suite
- Easily adaptable to be used as a coverage measure

---

**Conclusions and Future Work**

Nov. 27, 2008 17 / 18
### Main results

- Formal notion of risk
- Both evaluation of risk and computation of optimal test suite
- Easily adaptable to be used as a coverage measure

### Directions for Future Work

- Validation of the framework: tool support, case studies
- Dependencies between errors
- On-the-fly test derivation

For more details, see the technical report (http://fmt.cs.utwente.nl/~timmer)
Conclusions and Future Work

Main results

- Formal notion of risk
- Both evaluation of risk and computation of optimal test suite
- Easily adaptable to be used as a coverage measure

Directions for Future Work

- Validation of the framework: tool support, case studies
- Dependencies between errors
- On-the-fly test derivation

For more details, see the technical report (http://fmt.cs.utwente.nl/~timmer)