The use of Bayesian Belief Networks for combining disparate sources of information in the safety assessment of software based systems

Bjørn Axel Gran
Software Engineering laboratory (SElab)
Computerised Operation Support Systems Division (COSS)
Institutt for Energiteknikk / OECD Halden Reactor Project
bjorn.axel.gran@hrp.no
69212395 (dir)
http://www.ife.no
About me

• M.Sc., Institute for Industrial Mathematics: "Usikkerhet i sikkerhetsstudier" (Uncertainty in safety studies) 1993.
• Ph.D.: "The use of Bayesian Belief Networks for combining disparate sources of information in the safety assessment of software based systems" (Thesis 2002:35)
• Employeed at IFE since Januar 1995, now as section head for the Software Engineering laboratory.
• Was work-package leader for “WP2: Risk Analysis” in the EU-project CORAS (2001-2003): IST-2000-25031
• Currently involved in
  • research within the OECD Halden Reactor Project
    • in particular Qualification of integrated Tool Environments
  • risk and safety analysis within Air Traffic Management (ATM)
    • in particular Park Air Systems (delivery to NATS, UK)
  • part time father leave ...
• Also coach for the German nationalteam in orienteering (since 1998)
Institute for Energy Technology

- Established 1948
- About 550 employees, about 270 in Halden
- Turnover: NOK 500 million
- Main activity areas
  - Nuclear technology
  - Petroleum research
  - Energy and environmental technology
- Research under contract
- Reorganised 1. January 2001
  - Safety and MTO
    (Man – Technology – Organisation)
Our main activity areas...

Nuclear research

Petroleum research

Energy and the environment

MTO - safety

Basic research in physics
The OECD Halden Reactor

- Nuclear safety research in cooperation with OECD-NEA
- 20 member countries
- More than 100 collaborating institutions and companies
- Three year programme periods
The Software Engineering laboratory (SElab)

- a part of Safety MTO at IFE in Halden
- focus on dependable development and analysis of sw-based systems.
- some of the activities target complex sw-intensive systems and aim at developing and integrating methods and techniques within areas such as requirements engineering, life cycle modelling and risk analysis and assessment to describe and manage especially safety- and security-critical complex systems. Others aim at developing and improving specific methods and techniques (particularly within deterministic and probabilistic risk assessment) to better analyze software components.
- has integrated an MTO perspective into almost all its current activities today.
- Currently, projects towards petroleum, transport (railway and aviation) and nuclear domains. In the past, projects within eHealth (telemedicine) and telecommunications.
- Extensive international collaboration: OECD-NEA and IAEA, or national organisations, particularly in Europe, USA and South Korea.
- Currently, 12 researchers (full-time or part-time).
- Main responsible/contacts: Principal Scientists Atoosa P-J Thunem (atoosa.p-j.thunem@hrp.no) and Bjørn Axel Gran.
Overview

• Background
  • why look for disparate sources of information
  • the first attempt with BBN (and an small example!)

• **Evaluate the Use of Bayesian Belief Nets in Safety Assessment of Software Based Systems**
  • Combined the BBN method with DO-178B
  • Follow up project (modelling and computing) together with VTT Automation, Finland

• New projects
  • Finding Upper Bounds for Software Failure Probabilities
  • 2 German research projects
Background:

HWR425: "Review Software Testing Methodologies"
HWR456: "Experimental Investigation of …"
HWR528: "Experimental Investigation of the PIE-technique"
  (ESREL’98, PSA’98)
HWR527: "The use of BBN …"
  Rule based /Probabilistic /Expert judgement
  BBN for safety assessment
  (FLINS’98, Int. J. General Systems Vol. 29(2) 2000)
  … to Model the Confidence in Fault Freeness
  (ESREL’99)
HWR627: "Estimating dependability … using BBNs"
  Combine DO-178B and BBNs, apply the method
  on a real, safety related programmable system.
  (SAFECOMP’00, NPIC&HMIT 2000)
Test case: a program from the PODS diversity project

tested (1986 Bishop et al.)

- after acceptance phase no implementation faults found
- specification ambiguities, (faults unique to each program)
- specification incorrectness: 2 common-mode faults
- back-to-back testing revealed 7 faults
  - observed reliability growth, but assumed to give pessimistic results.

All faults removed:
- observed a limiting failure rate at 1.5E-6 per cycle.
- 30 seeded faults
  - 25 found, remaining 5 proven to have no effect
  - indications of fault freeness.
EISTRAM: Experimental Investigation of the PIE-technique

“A prediction of the probability that a fault will cause a failure at a particular location under a specified input distribution”

For a test to reveal a fault at a location in a program, it must:

- **EXECUTE** the location
- The fault must **INFECT** (change) the data state at that location
- The infected data state must be **PROPAGATED** to the end of the program and cause a failure

**Mutants:**

\[
\begin{align*}
a_1 &:= x_1 + \max(y_2, z_1); \\
a_1 &:= x_1 \times \max(y_1, z_1); \\
a_1 &:= x_1 + \min(y_1, z_1); \\
\ldots 
\end{align*}
\]

**Process input**

\[
x_1, y_1, z_1, x_2, y_2, z_2
\]

**Process output**

\[
a := \max(a_1, a_2);
\]
Test case: (cont)

PIE-analysis (1998 Gran & Thunem: EISTRAM) execution-, infection- (mutants) and propagation analysis

- observed several predictions of lower bound (the minimum failure probability) equal to zero.

- Used several input distributions
  - still 142 out of 300 locations likely to contain unhidden faults.
  - Should decrease the confidence in fault freeness.
  - Could be indications of implemented fault tolerance.
  - Should increase the confidence in fault freeness.

Observed the same results for a program for performing a Power Range Monitoring, for which the sub-routine part was developed using formal methods.
Test Case Conclusions:

- Back-to-back testing and error-seeding indicated fault freeness.
- However, no guarantee for unhidden faults and depended upon the test distributions!
- The PIE-analysis: gave it good news (designed for fault tolerance) or bad news (likely to hide faults)?
  ➢ The BBN methodology can be a useful way to combine the different sources of information.
  ➢ The BBN method constitutes a systematic way to combine quantitative and qualitative measurements.
Rule based assessment

Characterised by:

- easy to follow for the developer.
- easy to check for the assessor.

However:

- gets easily rigid.
- the rules must be based on something,
  - viz. Probabilistic methods or expert judgement.
Software testing

Can try to measure software failures, but:
• if a fault is revealed, the program would be corrected.
• And, what if no faults are revealed?
• Various reliability growth models are suggested.

The techniques are widely spread, but also criticised:
• need automated oracles,
• human oracles can be wrong, and are slow
• specifications can be wrong,
• operational input distributions can be unknown,
• the number of all possible inputs may be infinite.
Probabilistic assessment

Frequentist’s:
- Suitable for hardware with large series of components with corresponding failure data. Difficult for software.
- Various reliability growth models are suggested, but not applicable for safety critical software.
- Statistical testing can provide measure of failure probability.

Subjectivist’s:
- “Failure probability does not really exist, it is just a measure of one’s (lack of) knowledge of the system.” Well suited to software failure probability.
- Bayes’ method:
  - Assume a prior failure probability based on best judgement.
  - Perform a measurement.
  - Adjust failure probability to a posterior estimate.
How: Bayesian Belief Networks

- The theory about BBNs is well established.
- There are smart tools: HUGIN tool, SERENE
- Applied with success in various areas.
- Activity to apply this method for safety assessment of programmable systems.

A method to:
- show the link between observable properties and the confidence one can have in a system.
- based on information of disparate nature,
- combines quantitative observations and human judgments.
- easy to interpret
Bayesian Belief Networks

A Bayesian Belief Network:

- a connected and directed graph.
- nodes and directed arcs between them.
- Each node represents uncertain variables.
- Some nodes are denoted as “observable”.
- Network edges model relations between adjacent nodes.
- The probability density expresses our confidence in the various variable outcomes.
- The construction of the BBN is normally made gradually.

Start from a target node and draw edges to influencing nodes towards observable nodes.
Safety assessment, based on multiple evidences

Info on:
- production method
- quality control
...

Info about program:
- code listing
- functionality
...

Info about:
- debugging reports
- acceptance test
...

- user reports
- number of installations
- total usage time
...

Information on:
- production

Information on:
- product

Information on:
- V&V

Information on:
- previous usage

Information on:
- V&V

reliability assessment

hazard/risk analysis

Safety assessment

safety defences
A small example

Elicitation of probability distributions, two types:

- continuous probability distributions.
- probabilities of discrete state variables.
- An illustration of the latter:
  - The two nodes from the previous slide
  - “Development Quality” and “Documentation”
  - and the edge between.
Elicitation of probability distributions continue

*Prior belief:*
The Development Quality is “medium” to “high”
- \( P(\text{Development Quality is “low” } ) = 0.1 \)
- \( P(\text{Development Quality is “medium” } ) = 0.5 \)
- \( P(\text{Development Quality is “high” } ) = 0.4 \)

*Conditional probability* table for the edge

<table>
<thead>
<tr>
<th>Documentation</th>
<th>Development Quality</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>low</td>
</tr>
<tr>
<td>bad</td>
<td>0.9</td>
</tr>
<tr>
<td>acceptable</td>
<td>0.1</td>
</tr>
<tr>
<td>excellent</td>
<td>0.0</td>
</tr>
</tbody>
</table>
“Documentation” is an observable.

Assume that the “Documentation” is considered excellent:

- \( P(\text{Documentation is “bad” } ) = 0.0 \)
- \( P(\text{Documentation is “acceptable” } ) = 0.0 \)
- \( P(\text{Documentation is “excellent” } ) = 1.0 \)

Using the Bayesian method this modifies the belief in the “Development Quality” to be:

- \( P(\text{Development Quality is “low” } ) = 0.0 \)
- \( P(\text{Development Quality is “medium” } ) = 0.26 \)
- \( P(\text{Development Quality is “high” } ) = 0.74 \)
Ref: HWR 527, (Dahll & Gran 1998)
# The nodes

<table>
<thead>
<tr>
<th>Node variables</th>
<th>States of variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quality-control-documents</td>
<td>“none”</td>
</tr>
<tr>
<td></td>
<td>“partly”</td>
</tr>
<tr>
<td></td>
<td>“completed”</td>
</tr>
<tr>
<td>QA-standards</td>
<td>“none”</td>
</tr>
<tr>
<td></td>
<td>“generic”</td>
</tr>
<tr>
<td></td>
<td>“detailed”</td>
</tr>
<tr>
<td>Failures-in-other-products</td>
<td>“&gt;50%”</td>
</tr>
<tr>
<td></td>
<td>“10% - 50%”</td>
</tr>
<tr>
<td></td>
<td>“&lt;10%”</td>
</tr>
<tr>
<td>Number-of-products</td>
<td>“&lt;10”</td>
</tr>
<tr>
<td></td>
<td>“10-100”</td>
</tr>
<tr>
<td></td>
<td>“&gt;100”</td>
</tr>
<tr>
<td>Usage-time</td>
<td>“100 h”</td>
</tr>
<tr>
<td></td>
<td>“100-10000 h”</td>
</tr>
<tr>
<td></td>
<td>“&gt;10000 h”</td>
</tr>
<tr>
<td>Documentation</td>
<td>“bad”</td>
</tr>
<tr>
<td></td>
<td>“acceptable”</td>
</tr>
<tr>
<td></td>
<td>“excellent”</td>
</tr>
<tr>
<td>Quality-measures</td>
<td>“&lt;0.1”</td>
</tr>
<tr>
<td></td>
<td>“0.1-0.8”</td>
</tr>
<tr>
<td></td>
<td>“&gt;0.8”</td>
</tr>
<tr>
<td>Quality-control</td>
<td>“strict”</td>
</tr>
<tr>
<td></td>
<td>“lousy”</td>
</tr>
<tr>
<td>QA-policy</td>
<td>“bad”</td>
</tr>
<tr>
<td></td>
<td>“acceptable”</td>
</tr>
<tr>
<td></td>
<td>“excellent”</td>
</tr>
<tr>
<td>Producer’s-pedigree</td>
<td>“low”</td>
</tr>
<tr>
<td></td>
<td>“medium”</td>
</tr>
<tr>
<td></td>
<td>“high”</td>
</tr>
<tr>
<td>System-quality</td>
<td>“low”</td>
</tr>
<tr>
<td></td>
<td>“medium”</td>
</tr>
<tr>
<td></td>
<td>“high”</td>
</tr>
<tr>
<td>Development-quality</td>
<td>“low”</td>
</tr>
<tr>
<td></td>
<td>“medium”</td>
</tr>
<tr>
<td></td>
<td>“high”</td>
</tr>
<tr>
<td>User-experience</td>
<td>“low”</td>
</tr>
<tr>
<td></td>
<td>“medium”</td>
</tr>
<tr>
<td></td>
<td>“high”</td>
</tr>
<tr>
<td>Producer-quality</td>
<td>“low”</td>
</tr>
<tr>
<td></td>
<td>“medium”</td>
</tr>
<tr>
<td></td>
<td>“high”</td>
</tr>
</tbody>
</table>
BBN for System Quality

- **Producer Quality**
  - low 0.25
  - medium 0.50
  - high 0.25

- **User experience**
  - low 0.30
  - medium 0.40
  - high 0.30
A child node with two parents

### Failures in Other Products given the Producer’s Pedigree and the User Experience

<table>
<thead>
<tr>
<th>User Experience</th>
<th>low</th>
<th>medium</th>
<th>high</th>
<th>low</th>
<th>medium</th>
<th>high</th>
<th>low</th>
<th>medium</th>
<th>high</th>
</tr>
</thead>
<tbody>
<tr>
<td>Producer’s Pedigree</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>low</td>
<td>0.30</td>
<td>0.20</td>
<td>0.00</td>
<td>0.90</td>
<td>0.20</td>
<td>0.10</td>
<td>0.90</td>
<td>0.20</td>
<td>0.00</td>
</tr>
<tr>
<td>medium</td>
<td>0.40</td>
<td>0.20</td>
<td>0.10</td>
<td>0.10</td>
<td>0.60</td>
<td>0.20</td>
<td>0.10</td>
<td>0.60</td>
<td>0.10</td>
</tr>
<tr>
<td>high</td>
<td>0.30</td>
<td>0.60</td>
<td>0.90</td>
<td>0.00</td>
<td>0.20</td>
<td>0.70</td>
<td>0.00</td>
<td>0.20</td>
<td>0.90</td>
</tr>
</tbody>
</table>
## Prior values

<table>
<thead>
<tr>
<th>Node variables</th>
<th>States</th>
<th>Prior pdfs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quality-control-documents</td>
<td>(none, partly, completed)</td>
<td>(0.28, 0.37, 0.37)</td>
</tr>
<tr>
<td>QA-standards</td>
<td>(none, generic, detailed)</td>
<td>(0.31, 0.35, 0.35)</td>
</tr>
<tr>
<td>Failures-in-other-products</td>
<td>(&gt;50%, 10% - 50%, &lt;10%)</td>
<td>(0.30, 0.29, 0.41)</td>
</tr>
<tr>
<td>Number-of-products</td>
<td>(&lt;10, 10-100, &gt;100)</td>
<td>(0.31, 0.38, 0.31)</td>
</tr>
<tr>
<td>Usage-time</td>
<td>(100 h, 100-10000 h, &gt;10000 h)</td>
<td>(0.31, 0.38, 0.31)</td>
</tr>
<tr>
<td>Documentation</td>
<td>(bad, acceptable, excellent)</td>
<td>(0.41, 0.31, 0.28)</td>
</tr>
<tr>
<td>Quality-measures</td>
<td>(&lt;0.1, 0.1-0.8, &gt;0.8)</td>
<td>(0.40, 0.28, 0.31)</td>
</tr>
<tr>
<td>Quality-control</td>
<td>(strict, lousy)</td>
<td>(0.45, 0.55)</td>
</tr>
<tr>
<td>QA-policy</td>
<td>(bad, acceptable, excellent)</td>
<td>(0.35, 0.37, 0.28)</td>
</tr>
<tr>
<td>Producer-s-pedigree</td>
<td>(low, medium, high)</td>
<td>(0.35, 0.37, 0.28)</td>
</tr>
<tr>
<td>System-quality</td>
<td>(low, medium, high)</td>
<td>(0.43, 0.26, 0.31)</td>
</tr>
<tr>
<td>Development-quality</td>
<td>(low, medium, high)</td>
<td>(0.55, 0.37, 0.28)</td>
</tr>
<tr>
<td>User-experience</td>
<td>(low, medium, high)</td>
<td>(0.30, 0.40, 0.30)</td>
</tr>
<tr>
<td>Producer-quality</td>
<td>(low, medium, high)</td>
<td>(0.25, 0.50, 0.25)</td>
</tr>
</tbody>
</table>

The values in the prior pdfs column are calculated downwards from the base values.
one bad finding

<table>
<thead>
<tr>
<th>Node variables</th>
<th>States</th>
<th>Prior pdfs</th>
<th>Posterior pdfs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quality-control-doc.</td>
<td>( none, partly, completed)</td>
<td>(0.28, 0.37, 0.37)</td>
<td>(1.00, - -)</td>
</tr>
<tr>
<td>QA-standards</td>
<td>( none, generic, detailed)</td>
<td>(0.31, 0.35, 0.35)</td>
<td></td>
</tr>
<tr>
<td>Failures-in-other-prod</td>
<td>(&gt;50%, 10% - 50%, &lt;10%)</td>
<td>(0.30, 0.29, 0.41)</td>
<td>(0.28, 0.28, 0.44)</td>
</tr>
<tr>
<td>Number-of-products</td>
<td>(&lt;10, 10-100, &gt;100)</td>
<td>(0.31, 0.38, 0.31)</td>
<td>(0.31, 0.38, 0.31)</td>
</tr>
<tr>
<td>Usage-time</td>
<td>( 100 h, 100-10000 h, &gt;10000 h)</td>
<td>(0.31, 0.38, 0.31)</td>
<td>(0.31, 0.38, 0.31)</td>
</tr>
<tr>
<td>Documentation</td>
<td>( bad, acceptable, excellent)</td>
<td>(0.41, 0.31, 0.28)</td>
<td>(0.50, 0.29, 0.21)</td>
</tr>
<tr>
<td>Quality-measures</td>
<td>( &lt;0.1, 0.1-0.8, &gt;0.8)</td>
<td>(0.40, 0.28, 0.31)</td>
<td>(0.49, 0.26, 0.26)</td>
</tr>
<tr>
<td>Quality-control</td>
<td>( strict, lousy)</td>
<td>(0.45, 0.55)</td>
<td>(-, 1.00)</td>
</tr>
<tr>
<td>QA-policy</td>
<td>( bad, acceptable, excellent)</td>
<td>(0.35, 0.37, 0.28)</td>
<td>(0.57, 0.33, 0.10)</td>
</tr>
<tr>
<td>Producer-s-pedigree</td>
<td>( low, medium , high)</td>
<td>(0.35, 0.37, 0.28)</td>
<td>(0.46, 0.35, 0.20)</td>
</tr>
<tr>
<td>System-quality</td>
<td>( low, medium, high)</td>
<td>(0.43, 0.26, 0.31)</td>
<td>(0.55, 0.23, 0.22)</td>
</tr>
<tr>
<td>Development-quality</td>
<td>( low, medium, high)</td>
<td>(0.35, 0.37, 0.28)</td>
<td>(0.46, 0.25, 0.20)</td>
</tr>
<tr>
<td>User-experience</td>
<td>( low, medium, high)</td>
<td>(0.30, 0.40, 0.30)</td>
<td>(0.30, 0.40, 0.30)</td>
</tr>
<tr>
<td>Producer-quality</td>
<td>( low, medium, high)</td>
<td>(0.25, 0.50, 0.25)</td>
<td>(0.39, 0.47, 0.14)</td>
</tr>
</tbody>
</table>

P(system quality = low): 0.43 -> 0.55
P(system quality = high): 0.31 -> 0.22
### One good finding

<table>
<thead>
<tr>
<th>Node variables</th>
<th>States</th>
<th>Prior pdfs</th>
<th>Posterior pdfs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quality-control-doc.</td>
<td>(none, partly, completed)</td>
<td>(0.28, 0.37, 0.37)</td>
<td>(&lt;, &lt; 1.00)</td>
</tr>
<tr>
<td>QA-standards</td>
<td>(none, generic, detailed)</td>
<td>(0.31, 0.35, 0.35)</td>
<td>(&lt;, &lt; 1.00)</td>
</tr>
<tr>
<td>Failures-in-other-prod.</td>
<td>(&gt;50%, 10% - 50%, &lt;10%)</td>
<td>(0.30, 0.29, 0.41)</td>
<td>(0.32, 0.30, 0.38)</td>
</tr>
<tr>
<td>Number-of-products</td>
<td>(&lt;10, 10-100, &gt;100)</td>
<td>(0.31, 0.38, 0.31)</td>
<td>(0.31, 0.38, 0.31)</td>
</tr>
<tr>
<td>Usage-time</td>
<td>(100 h, 100-10000 h, &gt;10000 h)</td>
<td>(0.31, 0.38, 0.31)</td>
<td>(0.31, 0.38, 0.31)</td>
</tr>
<tr>
<td>Documentation</td>
<td>(bad, acceptable, excellent)</td>
<td>(0.41, 0.31, 0.28)</td>
<td>(0.31, 0.33, 0.36)</td>
</tr>
<tr>
<td>Quality-measures</td>
<td>(&lt;0.1, 0.1-0.8, &gt;0.8)</td>
<td>(0.40, 0.28, 0.31)</td>
<td>(0.30, 0.31, 0.39)</td>
</tr>
<tr>
<td>Quality-control</td>
<td>(strict, lousy)</td>
<td>(0.45, 0.55)</td>
<td>(1.00, 0.00)</td>
</tr>
<tr>
<td>QA-policy</td>
<td>(bad, acceptable, excellent)</td>
<td>(0.35, 0.37, 0.28)</td>
<td>(0.08, 0.42, 0.51)</td>
</tr>
<tr>
<td>Producer-s-pedigree</td>
<td>(low, medium, high)</td>
<td>(0.35, 0.37, 0.28)</td>
<td>(0.21, 0.40, 0.39)</td>
</tr>
<tr>
<td>System-quality</td>
<td>(low, medium, high)</td>
<td>(0.43, 0.26, 0.31)</td>
<td>(0.29, 0.30, 0.41)</td>
</tr>
<tr>
<td>Development-quality</td>
<td>(low, medium, high)</td>
<td>(0.35, 0.37, 0.28)</td>
<td>(0.21, 0.40, 0.39)</td>
</tr>
<tr>
<td>User-experience</td>
<td>(low, medium, high)</td>
<td>(0.30, 0.40, 0.30)</td>
<td>(0.30, 0.40, 0.30)</td>
</tr>
<tr>
<td>Producer-quality</td>
<td>(low, medium, high)</td>
<td>(0.25, 0.50, 0.25)</td>
<td>(0.08, 0.54, 0.38)</td>
</tr>
</tbody>
</table>

\[
P(\text{system quality} = \text{low}): 0.43 \rightarrow 0.29 \\
P(\text{system quality} = \text{high}): 0.31 \rightarrow 0.41
\]
**Observations on one node**

<table>
<thead>
<tr>
<th>Finding on QA-standards</th>
<th>Posterior pdfs for the “System-quality” (low, medium, high)</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>(0.55, 0.23, 0.22)</td>
</tr>
<tr>
<td>generic</td>
<td>(0.43, 0.28, 0.29)</td>
</tr>
<tr>
<td>detailed</td>
<td>(0.33, 0.27, 0.40)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Finding on Documentation</th>
<th>Posterior pdfs for the “System-quality” (low, medium, high)</th>
</tr>
</thead>
<tbody>
<tr>
<td>bad</td>
<td>(0.77, 0.14, 0.09)</td>
</tr>
<tr>
<td>acceptable</td>
<td>(0.29, 0.44, 0.27)</td>
</tr>
<tr>
<td>excellent</td>
<td>(0.09, 0.25, 0.66)</td>
</tr>
</tbody>
</table>
## Observations on all nodes

<table>
<thead>
<tr>
<th>Node variables</th>
<th>States</th>
<th>Prior pdfs</th>
<th>Posterior pdfs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quality-control-doc.</td>
<td>( none, partly, completed)</td>
<td>(0.28, 0.37, 0.37)</td>
<td>(γ, γ, 1.00)</td>
</tr>
<tr>
<td>QA-standards</td>
<td>( none, generic, detailed)</td>
<td>(0.31, 0.35, 0.35)</td>
<td>(γ, γ, 1.00)</td>
</tr>
<tr>
<td>Failures-in-other-prod.</td>
<td>( &gt;50%, 10% - 50%, &lt;10%)</td>
<td>(0.30, 0.29, 0.41)</td>
<td>(γ, γ, 1.00)</td>
</tr>
<tr>
<td>Number-of-products</td>
<td>( &lt;10, 10-100, &gt;100)</td>
<td>(0.31, 0.38, 0.31)</td>
<td>(1.00, γ, γ)</td>
</tr>
<tr>
<td>Usage-time</td>
<td>( 100 h, 100-10000 h, &gt;10000 h)</td>
<td>(0.31, 0.38, 0.31)</td>
<td>(γ, γ, 1.00)</td>
</tr>
<tr>
<td>Documentation</td>
<td>( bad, acceptable, excellent)</td>
<td>(0.41, 0.31, 0.28)</td>
<td>(0.03, 0.11, 0.84)</td>
</tr>
<tr>
<td>Quality-measures</td>
<td>( &lt;0.1, 0.1-0.8, &gt;0.8)</td>
<td>(0.40, 0.28, 0.31)</td>
<td>(γ, γ, 1.00)</td>
</tr>
<tr>
<td>Quality-control</td>
<td>( strict, lousy)</td>
<td>(0.45, 0.55)</td>
<td>(1.00, γ)</td>
</tr>
<tr>
<td>QA-policy</td>
<td>( bad, acceptable, excellent)</td>
<td>(0.35, 0.37, 0.28)</td>
<td>(0.01, 0.11, 0.88)</td>
</tr>
<tr>
<td>Producer-s-pedigree</td>
<td>( low, medium, high)</td>
<td>(0.35, 0.37, 0.28)</td>
<td>(0.16, 0.22, 0.62)</td>
</tr>
<tr>
<td>System-quality</td>
<td>( low, medium, high)</td>
<td>(0.43, 0.26, 0.31)</td>
<td>(0.02, 0.04, 0.94)</td>
</tr>
<tr>
<td>Development-quality</td>
<td>( low, medium, high)</td>
<td>(0.35, 0.37, 0.28)</td>
<td>(0.01, 0.11, 0.86)</td>
</tr>
<tr>
<td>User-experience</td>
<td>( low, medium, high)</td>
<td>(0.30, 0.40, 0.30)</td>
<td>(0.04, 0.04, 0.92)</td>
</tr>
<tr>
<td>Producer-quality</td>
<td>( low, medium, high)</td>
<td>(0.25, 0.50, 0.25)</td>
<td>(0.00, 0.22, 0.78)</td>
</tr>
</tbody>
</table>

P(system quality = low): 0.43 -> 0.02  
P(system quality = high): 0.31 -> 0.94
The objective:

Evaluate the Use of Bayesian Belief Nets in Safety Assessment of Software Based Systems

- Risk assessment of a system based on disparate evidences.
- Transfer quality and safety guidelines for a system life cycle model into a format which makes it possible to use BBNs.
- Apply the approach on a real example
Overview of work:

Stage I:
Milestones:
• Combined the BBN method with DO-178B (standard for safety critical software in civil aviation).
• Applied the method on a real, safety related programmable system. (The M-ADS Airborne Equipment)
• Evaluated the results and experiences
• Lessons learned and issues for further investigation

Stage II:
• Follow up project (modelling and computing) together with VTT Automation, Finland
• Evaluation of specific results from Stage I
Overview Stages

Stage I
- DESIGN FEATURES
- DEVELOPMENT PROCESS

Stage II
- TESTING
- OPERATIONAL EXPERIENCE
The M-ADS Airborne Equipment

- Designed by KDA
- a system for automated transmission of graphical position information from helicopters to land based control stations.
- Aid in a rescue operation
- Developed according to DO-178B

- if the helicopter has made an emergency landing on the sea a correct localization is necessary for a successful rescue operation
- the system is therefore safety critical.
DO-178B:

Standard for safety critical software in civil aviation

- Discusses aspects of airworthiness certification with respect to the production of software.
- Does not provide guidelines concerning the structure of the applicant’s organization, relations to suppliers and personnel qualification criteria.
- Defines a set of five software levels
- The main recommendations are given in a set of 10 tables
- Each table relates to a certain stage in the development and validation process, and contains a set of objectives.
The Construction process

Identify
the main aspects that may influence the dependability of a system.
Distinguish between aspects that are:
• related to the interaction between the system and its environment, and
• related to the system itself:
  Quality of the producer
  Quality of the production process
  Quality of the product
  Quality of the analysis
The higher level network shows how nodes representing **four quality aspects** are combined with **other nodes** in the net and lead to **nodes** representing the **reliability** of the system.
The lower level

- One BBN for each of the quality aspects
- Associate the objectives of DO-178B with the quality aspects.
- Each top node linked to nodes representing the lifecycle processes, e.g.: the Software Planning Process (A1)
- These linked to nodes representing the objectives of the tables, e.g.: ”Completeness of the software development plan”
Questions on the end nodes:

Questions for the different end nodes,

e.g.: “are all future phases in the lifecycle included in the plans?”

formulated in a way that makes it possible to transfer the answer into a quantitative measurement.
Expert Judgement

A rather complex system of BBNs,
- a large number of conditional probability tables to be assessed:
  - The "P(A|B)"s assessed by expert group
  - lead by a BBN specialist
  - Ranked the different "children" w.r.t importance
  - Applied a scheme of conditional probability tables

Expert judgement about DO-178B
- brainstorming activity among the project participants.
- this opens for some subjectivity.
- some of the project members were considered as experts within their field.
Observations and Scenarios

The collection of observations

- Questions answered by developers, owners, assessors (KDA)
- Number between 0 and 1, representing degree of certainty
- Developer judgement about M-ADS

4 initial scenarios:

- no observations, KDA, best case, worst case

Additional scenarios:

- “Partial”: the effect of observations during only one stage in the development and validation process, such as A1, A2... A10
- “Incremental”: the effect of first observing during the stage A1, then A2, and so on
- Sensitivity analysis for the node $P(\text{failure State})$. 
Several results from M-ADS were surprising

- Discovery of a wrongly entered observation.
  - After correcting this error, results as originally expected.
  - demonstrated that one negative observation could have a significant effect on a partial result.
- Observations are added subsequently during the life-cycle stages:
  - What would be the result if more observations were negative?
  - Is it possible to find a set of “negative observations” that belong to all or more phases?
19 observations related to 2 or 3 processes

- These 19 observations were divided into 5 groups
- Negative observations entered to each group.

<table>
<thead>
<tr>
<th>Observation Group</th>
<th>Related to processes:</th>
<th>Related to quality aspect:</th>
<th>Observed effect related to quality aspect</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A4, A5, A7, other</td>
<td>Product</td>
<td>minor</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Analysis</td>
<td>minor</td>
</tr>
<tr>
<td>2</td>
<td>A1, A2, A6</td>
<td>Process</td>
<td>minor</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Analysis</td>
<td>no</td>
</tr>
<tr>
<td>3</td>
<td>A5, A6</td>
<td>Product</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Process</td>
<td>minor</td>
</tr>
<tr>
<td>4</td>
<td>A9, A10</td>
<td>Product</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Process</td>
<td>minor</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Producer</td>
<td>significant</td>
</tr>
<tr>
<td>5</td>
<td>A3, A5</td>
<td>Product</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Process</td>
<td>minor</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Analysis</td>
<td>no</td>
</tr>
</tbody>
</table>
Comparing partial lifecycle process results

A4: “verification of outputs of software design processes”
A5: “verification of outputs of software coding and integration process”

<table>
<thead>
<tr>
<th>Quality aspect</th>
<th>Partial results (A4 and A5)</th>
<th>Observed difference in</th>
<th>Effect of difference on the partial results (A4 and A5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Producer</td>
<td>almost equal</td>
<td>differences in the cpts and topology</td>
<td>differences in the cpts and topology</td>
</tr>
<tr>
<td>Analyses</td>
<td>almost equal</td>
<td>different number of questions</td>
<td>different observations</td>
</tr>
<tr>
<td>Process</td>
<td>different</td>
<td>differences in the cpts and topology</td>
<td>different observations</td>
</tr>
<tr>
<td>Product</td>
<td>very different</td>
<td>different number of questions</td>
<td>different observations</td>
</tr>
</tbody>
</table>
Conclusions from the M-ADS-project

- a systematic way to combine quantitative and qualitative evidences.
- a framework to show the confidence one has in e.g. "high quality".
- could be used at all stages throughout the whole software lifecycle.
- establishment of the BBNs and prior distributions was rather time consuming. However, most was related to DO-178B.
- it may be feasible to transfer this knowledge to other safety related software engineering standards.
- was difficult to perform the expert judgment.
- collecting the different observable properties, and performing the calculations, were fairly easy and not so time consuming.
- a great advantage with knowledge within BBN/probabilistic theory.
Stage II: The merged network

Quality of Producer → Quality of Process → Problem Complexity

Quality of Analysis → Solution Complexity → Quality of Product

Problem Complexity → Theta_priori

Theta → P

P → 0 failures in N tests

N
Results Stage II

HUGIN and WinBUGS give slightly different results when no soft evidence is used. With soft evidence, both models converge to the results obtained from testing alone.

Sc.1, median HUGIN
Sc.2, median, HUGIN
Sc.1, 97.5% HUGIN
Sc.2, 97.5% HUGIN
Sc.1, median WBUGS
Sc.2, median, WBUGS
Sc.1, 97.5% WBUGS
Sc.2, 97.5% WBUGS

P
N
VTT-case: A SW-based motor protection relay

- SPAM 150C motor protection relay (ABB Substation Automation)
- Focus on the whole life-cycle
- Combined quantitative and qualitative reliability related evidences in a BBN
- Qualitative: for building the prior estimation
- Quantitative: operational experiences during its life-cycle
- A 6 step expert judgement process.
- Results for a conservative and a neutral approach
A SW-based motor protection relay
The conservative approach

![Graph showing software version vs failure frequency]

- Software version
- Failure frequency
- [a]
- Software faults
- All inconveniences
Ongoing/new BBN projects

• “Finding Upper Bounds for Software Failure Probabilities – Experiments and Results”
  • Ph.D-study - Monica Kristiansen
    • Østfold University College / Institute for Energy Technology

• Applying BBN
  • in the assessment of embedded quality
    • German research project: VeNuS
  • in the development of a framework for qualifying Integrated Tool Environments
    • German research project: QUITE
  • for assessing human reliability
Finding Upper Bounds for Software Failure Probabilities – Experiments and Results

- The test cases investigated were based on the following different types of prior probability distributions:

- **Case 1**: In the first case, a uniform prior probability distribution for $\theta$ is assumed both under the null hypothesis and the alternative hypothesis. In addition, three different prior beliefs in the null hypothesis are assumed.

- **Case 2**: In the second case, an expert is allowed to set a certain upper bound on the failure probability under the alternative hypothesis. A uniform prior probability distribution for $\theta$ is assumed both under the null hypothesis and the alternative hypothesis. In addition, two different prior beliefs in the null hypothesis are assumed.

- **Case 3**: In the third case, the prior probability distribution for $\theta$ is based on a continuous beta distribution over the entire interval $\langle 0, 1 \rangle$. 
Case 1

Prior probability distribution:

\[
\pi(\theta) = \begin{cases} 
\frac{1}{\theta_0}, & \theta \leq \theta_0 \\
\frac{1}{(1 - \theta_0)}, & \theta > \theta_0 
\end{cases}
\]

<table>
<thead>
<tr>
<th>(\theta_0)</th>
<th>(C_0)</th>
<th>No. of tests (P(H_0) = 0.01)</th>
<th>No. of tests (P(H_0) = 0.1)</th>
<th>No. of tests (P(H_0) = 0.6)</th>
<th>No. of tests using classical statistical testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>0.99</td>
<td>457</td>
<td>228</td>
<td>50</td>
<td>459</td>
</tr>
<tr>
<td>0.001</td>
<td>0.99</td>
<td>2378</td>
<td>636</td>
<td>63</td>
<td>4603</td>
</tr>
<tr>
<td>0.0001</td>
<td>0.99</td>
<td>6831</td>
<td>853</td>
<td>65</td>
<td>46050</td>
</tr>
<tr>
<td>0.00001</td>
<td>0.99</td>
<td>9349</td>
<td>886</td>
<td>65</td>
<td>460515</td>
</tr>
<tr>
<td>0.000001</td>
<td>0.99</td>
<td>9752</td>
<td>890</td>
<td>65</td>
<td>4605168</td>
</tr>
</tbody>
</table>
Case 1 – results

• When a uniform prior probability distribution is assumed both under \( H_0 \) and \( H_1 \), the number of tests is greatly reduced compared to the number of tests required by classical statistical testing.

• Choosing two separate uniform probability distributions under \( H_0 \) and \( H_1 \) does not at all represent a conservative approach.

• In addition, it can be seen that the higher the prior belief in \( H_0 \) is the fewer tests are needed.
Case 2

In Case 2, an expert is allowed to set a certain upper bound on the failure probability under $H_1$, i.e. to state a value $\theta_1$ for which the probability of having a failure probability higher than this is zero.

Prior probability distribution:

$$\pi(\theta) = \begin{cases} P(H_0)/\theta_0, & \theta \leq \theta_0 \\ P(H_1)/(\theta_1 - \theta_0), & \theta_0 < \theta \leq \theta_1 \\ 0, & \theta > \theta_1 \end{cases}$$
Case 2 – results (2)

- The number of required tests increases significantly when the upper bound for the failure probability ($\theta_1$) decreases.
- Some realistic test cases, where the upper bound was set to 0.1 and 0.05, indicate an increase in the number of required tests with a factor of respectively 10 and 20.
- In addition, as for Case 1, it can be seen that the number of required tests is reduced radically when one’s prior belief in the null hypothesis increases.
In Case 3, a continuous probability distribution for $\theta$ is assumed over the entire interval $\langle 0, 1 \rangle$, e.g. a beta distribution.

$$
\pi(\theta) = \begin{cases} 
\frac{1}{P(H_0)} \frac{\tau(\alpha + \beta)}{\tau(\alpha)\tau(\beta)} \theta^{\alpha-1} (1 - \theta)^{\beta-1}, & \theta \leq \theta_0 \\
\frac{1}{P(H_1)} \frac{\tau(\alpha + \beta)}{\tau(\alpha)\tau(\beta)} \theta^{\alpha-1} (1 - \theta)^{\beta-1}, & \theta > \theta_0 
\end{cases}
$$
Case 3 – results (1)

\[ C_0 = 0.99 \]
\[ \theta_0 = 10^{-4} \]

<table>
<thead>
<tr>
<th>P(H_0)</th>
<th>( \alpha )</th>
<th>( \beta )</th>
<th>No. of tests</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>1</td>
<td>( \rightarrow 0 )</td>
<td>46050</td>
</tr>
<tr>
<td>10^{-4}</td>
<td>1</td>
<td>1</td>
<td>46049</td>
</tr>
<tr>
<td>0.10</td>
<td>1</td>
<td>1054</td>
<td>44996</td>
</tr>
<tr>
<td>0.20</td>
<td>1</td>
<td>2231</td>
<td>43819</td>
</tr>
<tr>
<td>0.30</td>
<td>1</td>
<td>3567</td>
<td>42483</td>
</tr>
<tr>
<td>0.40</td>
<td>1</td>
<td>5108</td>
<td>40942</td>
</tr>
<tr>
<td>0.50</td>
<td>1</td>
<td>6931</td>
<td>39119</td>
</tr>
<tr>
<td>0.60</td>
<td>1</td>
<td>9163</td>
<td>36887</td>
</tr>
<tr>
<td>0.70</td>
<td>1</td>
<td>12039</td>
<td>34011</td>
</tr>
<tr>
<td>0.80</td>
<td>1</td>
<td>16094</td>
<td>29956</td>
</tr>
<tr>
<td>0.90</td>
<td>1</td>
<td>23025</td>
<td>23025</td>
</tr>
<tr>
<td>0.95</td>
<td>1</td>
<td>29956</td>
<td>16094</td>
</tr>
<tr>
<td>0.98</td>
<td>1</td>
<td>39119</td>
<td>6931</td>
</tr>
<tr>
<td>0.99</td>
<td>1</td>
<td>46050</td>
<td>0</td>
</tr>
</tbody>
</table>

(b)
Case 3 – results (2)

- The number of required tests decreases towards 0 when one’s prior belief in the null hypothesis increases.
- The number of required tests increases significantly when the $\alpha$-value increases, which can be interpreted as an increase in the number of failures detected during testing.
- In addition, it can be seen that the total number of tests required by using the Bayesian hypothesis testing approach both can result in fewer as well as even more tests compared to classical statistical testing.
Conclusions – Upper bounds

- The results from the experiments showed that the number of required tests is highly dependent on the choice of prior distribution in the Bayesian hypothesis testing approach.
  - the prior confidence in the software component
  - the shape of the prior distribution
- The total number of tests required by using this approach both can result in fewer as well as even more tests compared to classical statistical testing.
- To choose a prior probability distribution for a software component’s failure probability that correctly reflects one’s prior belief is therefore of great importance!
VeNuS (Vorgehen zum effizienten Nachweis der Benutzbarkeit und Sicherheit rechnergestützter Leittechniksysteme)

- Linking quality characteristics to the requirements of 3 standards relevant for the nuclear field (IEC61508, IEC62138 and NUREG 800)
  - distributing target values
  - comparing with achieved values based upon observations

- Experiences on applying BBN
  - Applying BBN’s, means modelling according to the probabilistic rules, e.g. $P(A|B)$ and $P(B|A)$ are not independent ...
  - When calculating with small numbers, the BBN collapses
  - e.g., if $P(Bx|Ax)=1$, $P(By|Ay)=0$
  - observations on C have no effect on A
QUITE (Qualification of Integrated Tool Environments for the Development of Computer-Based Systems in NPP)

Apply BBN for
- the combination of qualification results into an integrated qualification measurement
Bayesian Belief Nets in combining evidence

- A flexible and coherent way of integrating qualitative and quantitative pieces of evidence from different sources.
- Includes the connection between these quantities and software reliability.
- Freedom to utilise the evidences of qualitative nature.
- The combination and reasoning of different kind of evidence can be carried out using fundamentally the same methodological approach.
- The use of BBNs provides a systematic approach.
Bayesian Belief Nets as decision support

- BBNs formulate a general and consistent way of reasoning one’s beliefs about the reliability of a system.
- BBNs can be used to provide a decision support
  - after the system has been implemented.
  - Even better if applied early in the development of a system.
- Implications of usefulness related to the improvement of reliability in a system’s design and development process.
- BBNs establish a formal ground for the communication in the licensing process of software-based systems.
- BBNs provides a good starting point for the decision making in the licensing process of sw-based systems.
The construction and validation of a BBN

- Establishment of BBNs can be rather time consuming.
- The experiences from modelling DO-178B as BBNs, can be transferred to the modelling of other sw standards
- Difficult to assign values to the conditional probability distributions.
- The most difficult questions are if a node should be added or an arc is missing.
- There are problems of performing a verification of a BBN.
Articles


