Software Improvement Group

Benchmark-based Software Product Quality

Tiago L. Alves

June 2011

T +31 20 314 0950
info@sig.eu
www.sig.eu

Background

About me

- Degree + MSc Informatics and System Engineering, University of Minho, Braga, PT
- 2006: Young Graduate Trainee at ESOC, Darmstadt, Germany
- Currently: PhD Researcher at University of Minho hosted at the Software Improvement Group, Amsterdam, Netherlands (finishing thesis)

Research interests

- Source code analysis techniques
- Software metrics and quality models to assess quality
- Industrial applications

Software Improvement Group

Who are we?

- Highly specialized research company for quality of software, founded in 2000 as a spin-off of the Centre for Mathematics and Information Technology
- Independent and therefore able to give objective advice
- Decorated with the Innovator Award 2007 and ICT Regie Award 2008

What do we do?

- Fact-based consultancy supported by our automated toolset for source code analysis
- Assessment across technologies by use of technology-independent methods

Our mission: We give you control over your software.

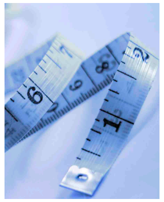
Services

Software Risk Assessment

- In-depth investigation of software quality and associated business risks
- Answers to specific research questions

Software Monitoring

- Continuous measurement, feedback, and development consultancy
- Guard quality from start to finish



Software Product Certification

- Five levels of technical quality (maintainability)
- Evaluation by SIG, certification by TÜV Informationstechnik

Who is using our services?

Financial and Insurance companies

- ABN·AMRO
- ING
- Rabobank
- achmea
- Friesland Bank
- LeasePlan
- Interpolis
- Bank Mendes Gans
- globalcollect international payment services
- Allianz
- PGGM
- InterBank KREDIET WAAR JE LANGER MEE GENIET
- Zorg en Zekerheid
- GENERALI verzekeringsgroep
- delta lloyd
- Eurobank EFG
- ZwitserLeven
- KAS BANK
- SNS Bank
- Volvo OtoFinans

Government

- Rijksoverheid
- kadaster
- Schweizerische Eidgenossenschaft Confédération suisse Confederazione Svizzera Confederaziun svizra
- Belastingdienst
- Raad voor Rechtsbijstand
- POLITIE
- cwi
- SVB
- RDW

Logistical

- DHL
- KLM
- TNT
- EUROMAX TERMINAL
- norfolkline DOVER - DUNKERQUE FERRIES
- ProRail
- Port of Rotterdam

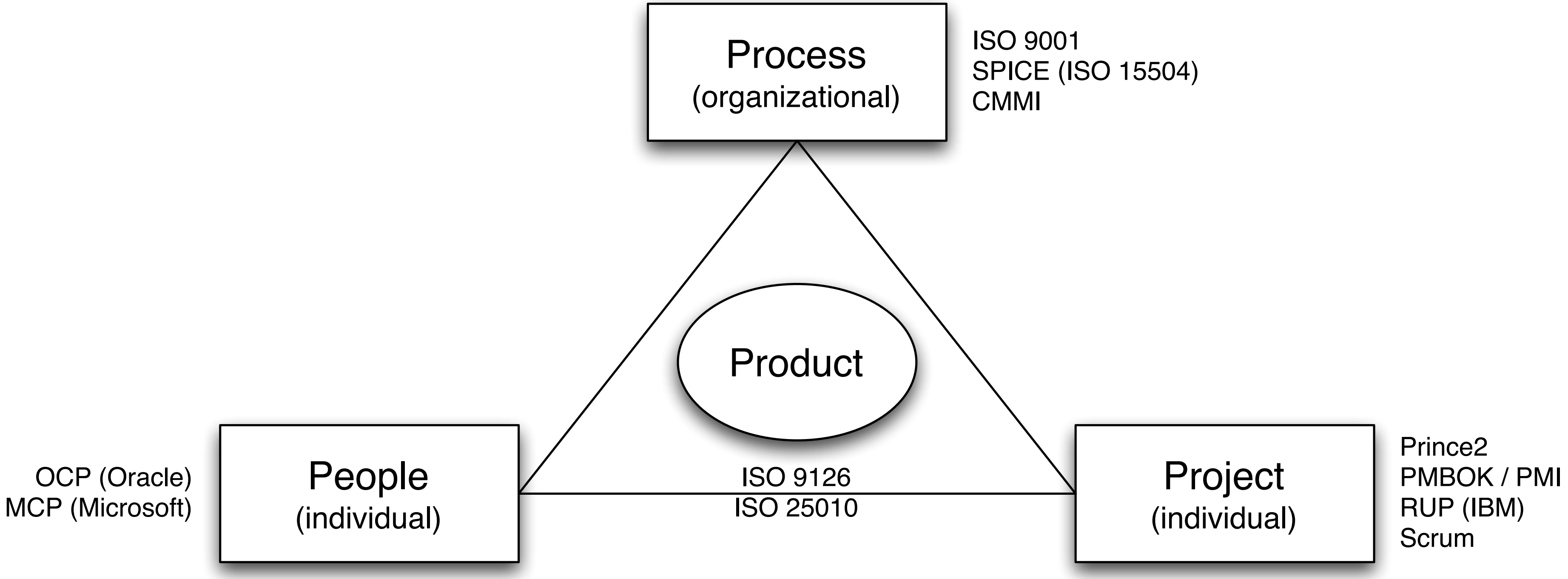
IT

- Getronics PinkRoccade
- CENTRIC
- Capgemini CONSULTING.TECHNOLOGY.OUTSOURCING
- Exact software
- IBM
- CHESS
- logica

Other

- ENECO energie
- PRICEWATERHOUSECOOPERS
- essent
- SWISSLEX
- KPMG
- NXP founded by Philips
- Gasunie
- TNO
- IKEA
- Alcatel·Lucent
- Electrabel

Bermuda triangle of software quality assurance



Process (organizational)

ISO 9001
SPICE (ISO 15504)
CMMI

Product

People (individual)

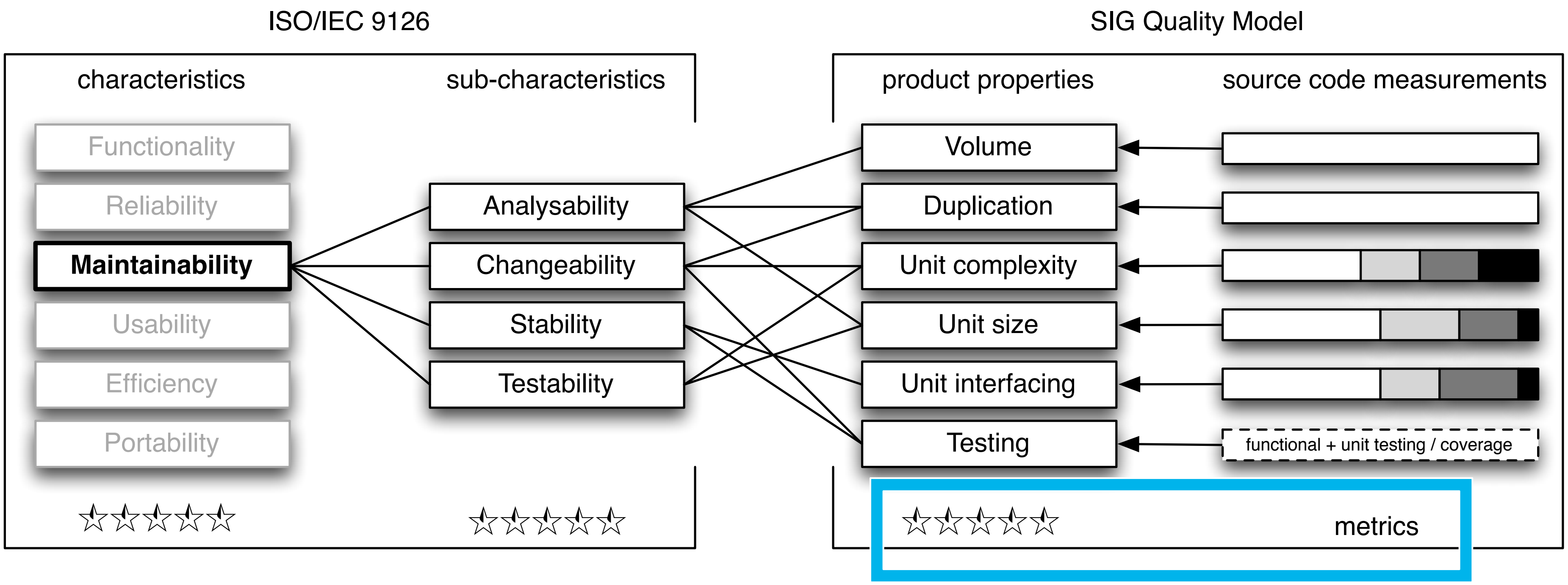
OCP (Oracle)
MCP (Microsoft)

ISO 9126
ISO 25010

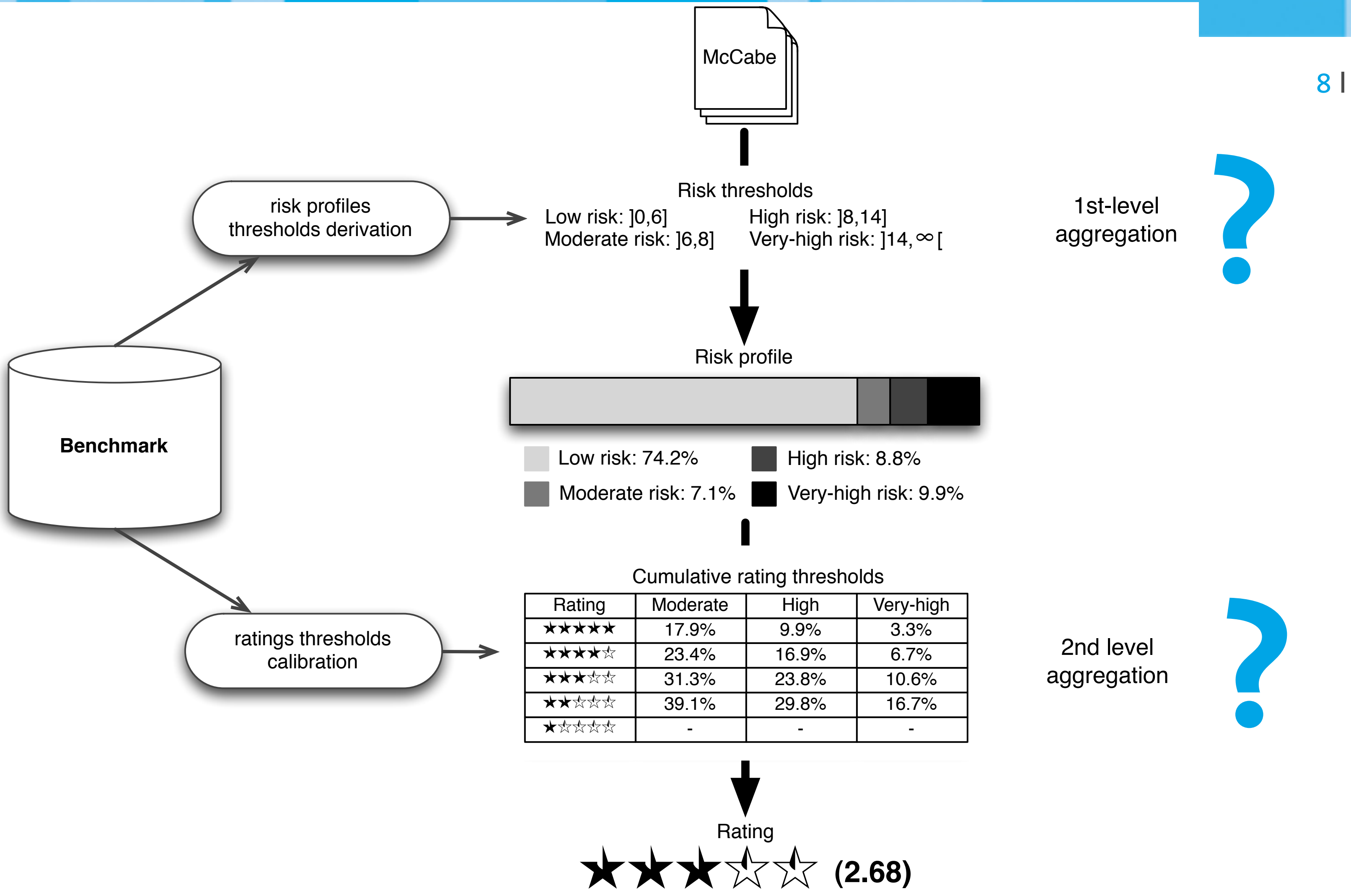
Project (individual)

Prince2
PMBOK / PMI
RUP (IBM)
Scrum

SIG Quality Model for Maintainability (operationalization of the ISO 9126)



Benchmarking metrics to ratings



McCabe
Benchmark
risk profiles thresholds derivation
ratings thresholds calibration
1st-level aggregation
2nd level aggregation

Risk thresholds

Low risk:]0,6]
Moderate risk:]6,8]
High risk:]8,14]
Very-high risk:]14,∞[

Risk profile

░ Low risk: 74.2%
▒ Moderate risk: 7.1%
▓ High risk: 8.8%
█ Very-high risk: 9.9%

Cumulative rating thresholds

Rating	Moderate	High	Very-high
★★★★★	17.9%	9.9%	3.3%
★★★★☆	23.4%	16.9%	6.7%
★★★☆☆	31.3%	23.8%	10.6%
★★☆☆☆	39.1%	29.8%	16.7%
★☆☆☆☆	-	-	-

Rating

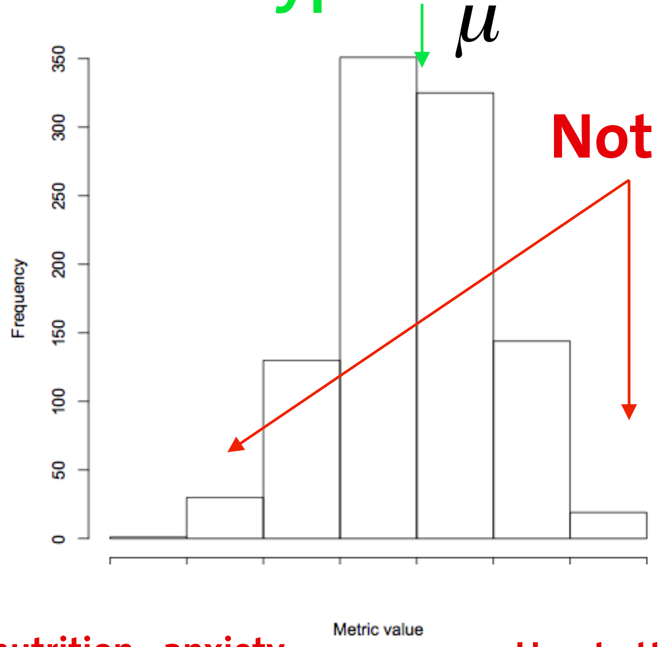
★★★☆☆ **(2.68)**

Part I
Derivation of risk thresholds

How to derive thresholds?
Life sciences vs. software sciences

Cholesterol levels

Typical values μ

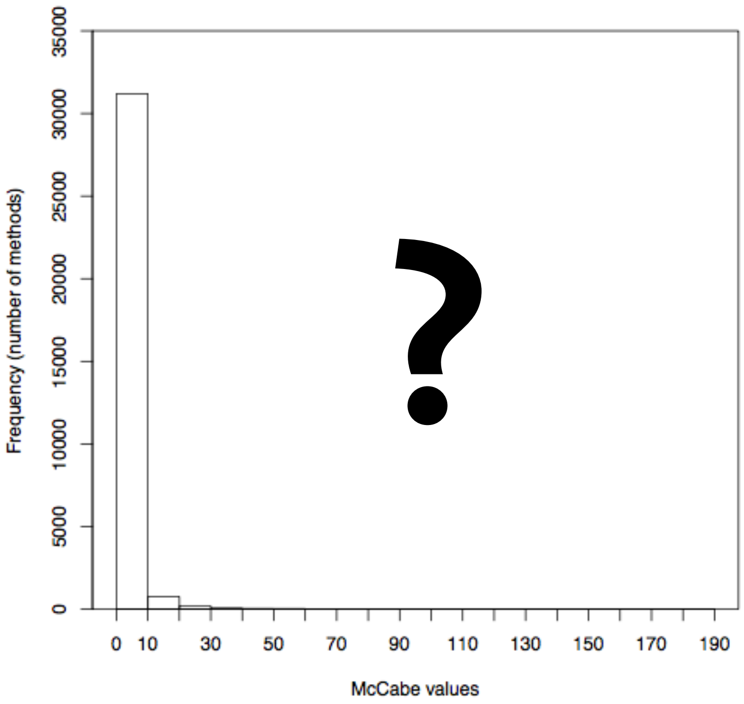


Not so good $\mu \pm \delta$

Malnutrition - anxiety, depression, suicide

Heart attack

Complexity

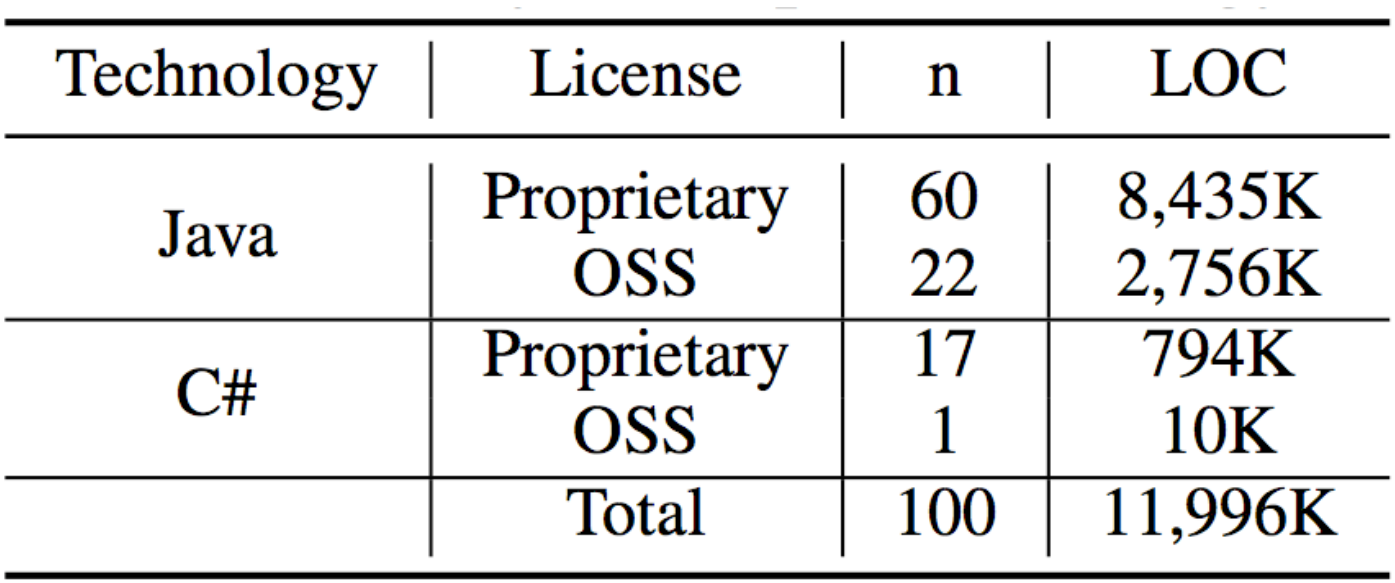


Derivation of risk thresholds: requirements

Requirements

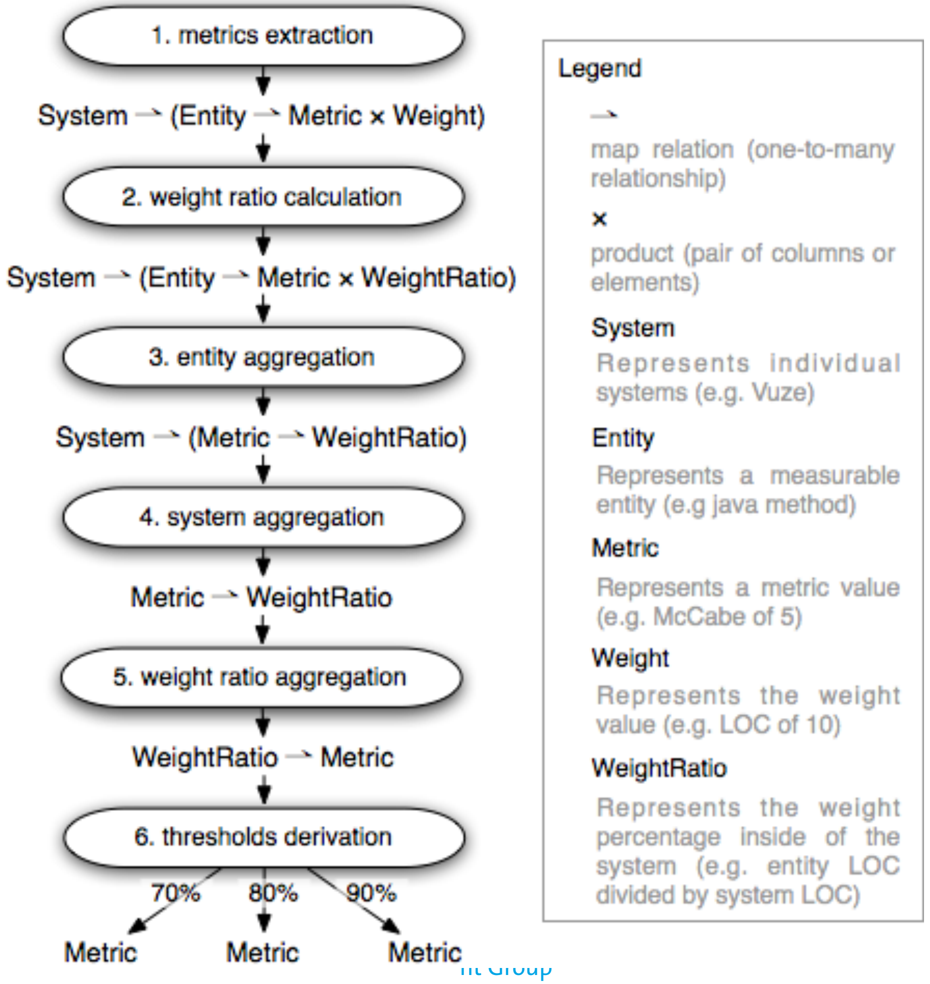
1. Respect the statistical properties of the metric (scale and distribution)
2. Based on data analysis from a representative set of systems (benchmark)
3. Repeatable, transparent, and of straightforward execution.
4. Enable traceability of results

Experimental benchmark



Technology	License	n	LOC
Java	Proprietary	60	8,435K
	OSS	22	2,756K
C#	Proprietary	17	794K
	OSS	1	10K
	Total	100	11,996K

Derivation of risk thresholds: methodology



1. metrics extraction

System ⇀ (Entity ⇀ Metric × Weight)

2. weight ratio calculation

System ⇀ (Entity ⇀ Metric × WeightRatio)

3. entity aggregation

System ⇀ (Metric ⇀ WeightRatio)

4. system aggregation

Metric ⇀ WeightRatio

5. weight ratio aggregation

WeightRatio ⇀ Metric

6. thresholds derivation

70% Metric

80% Metric

90% Metric

Legend

⇀ map relation (one-to-many relationship)

× product (pair of columns or elements)

System Represents individual systems (e.g. Vuze)

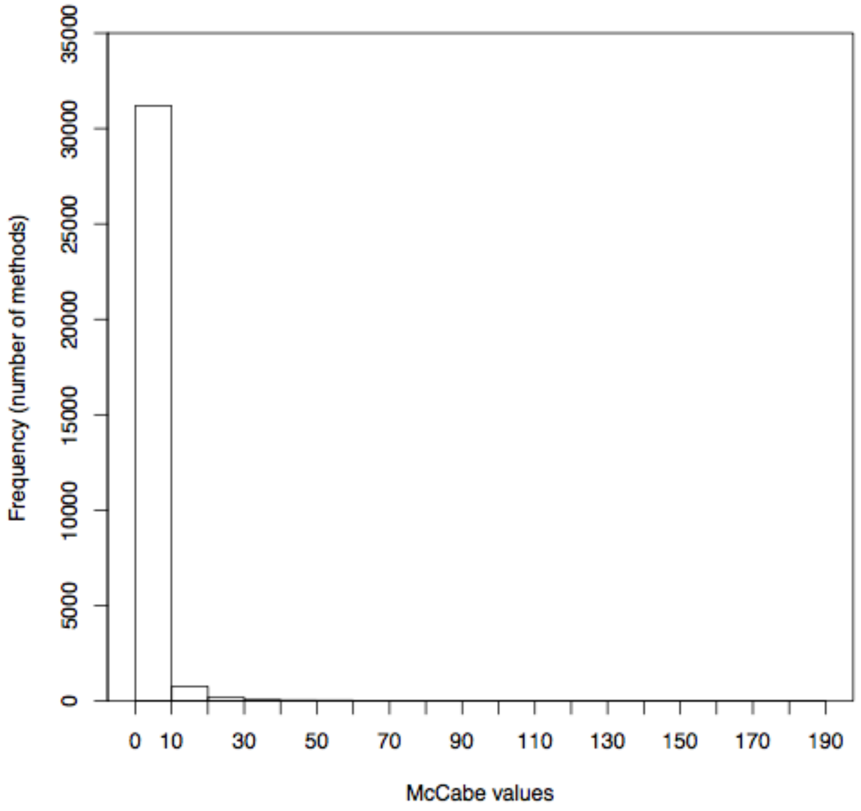
Entity Represents a measurable entity (e.g java method)

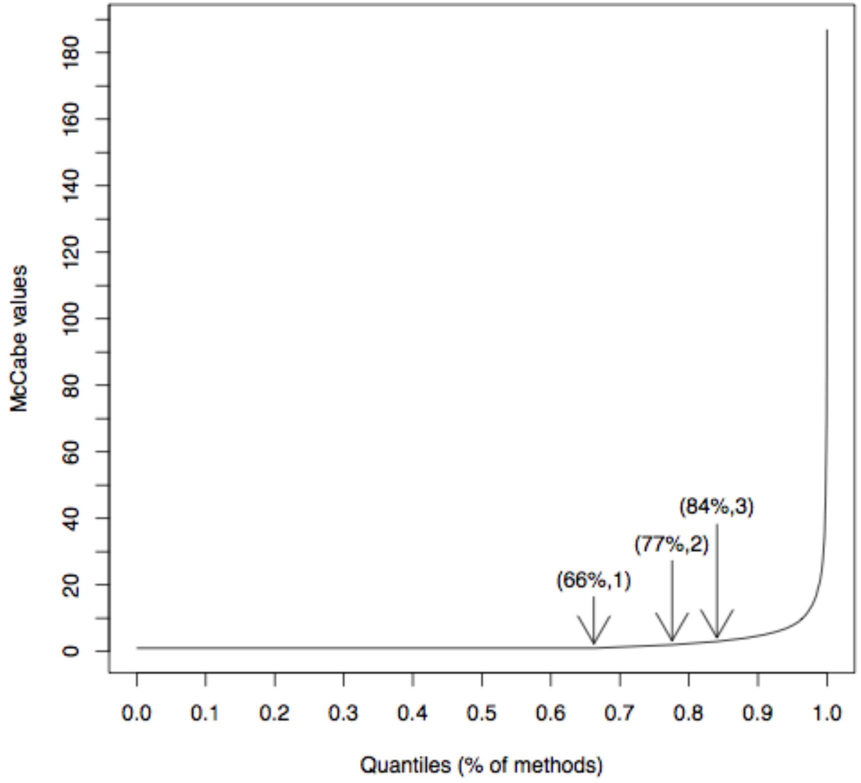
Metric Represents a metric value (e.g. McCabe of 5)

Weight Represents the weight value (e.g. LOC of 10)

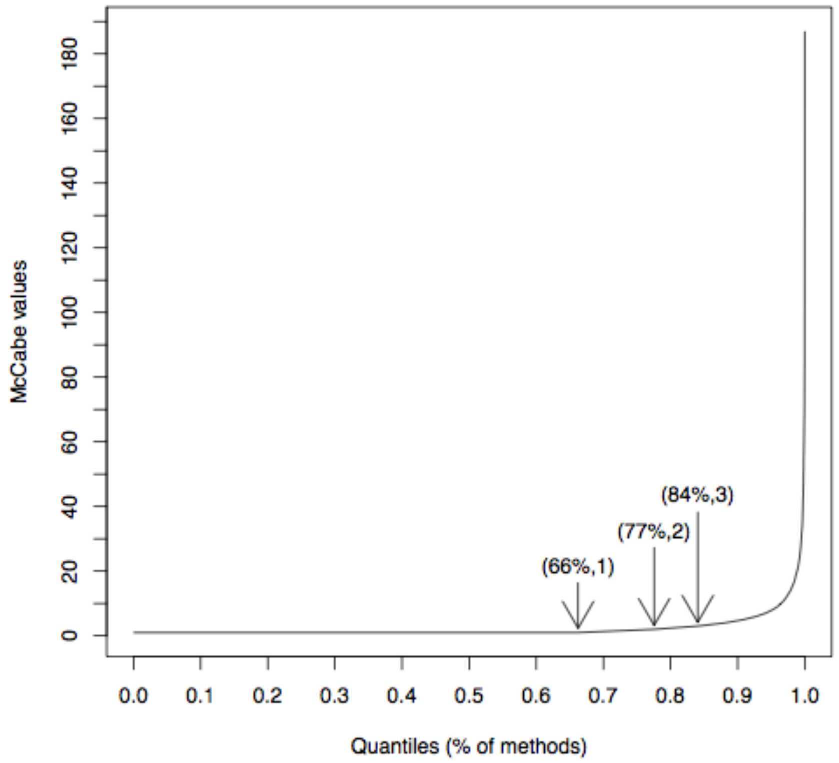
WeightRatio Represents the weight percentage inside of the system (e.g. entity LOC divided by system LOC)

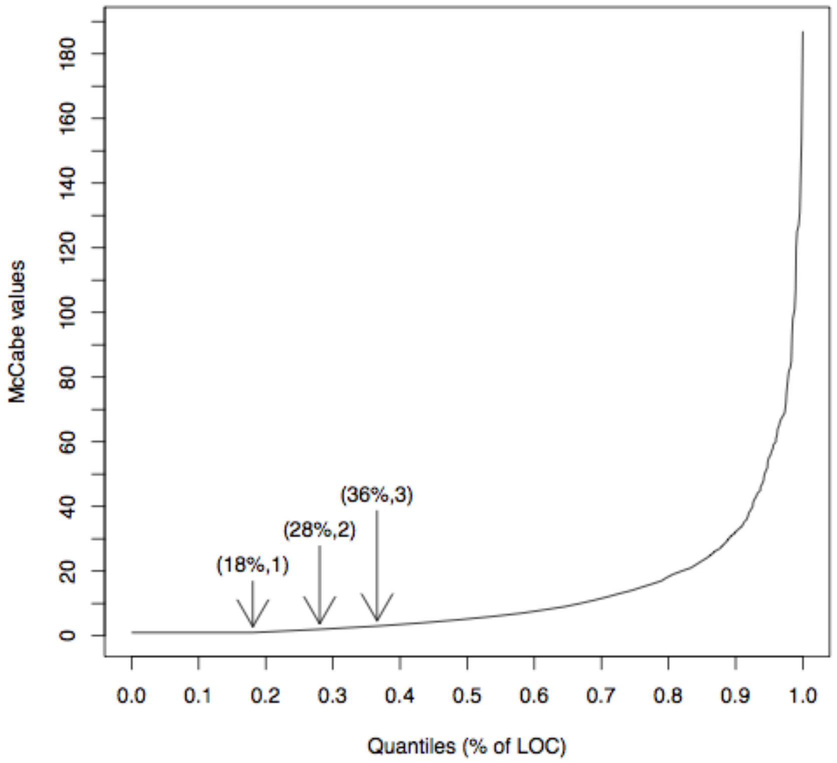
Derivation of risk thresholds: background
Histogram vs. Quantile plots



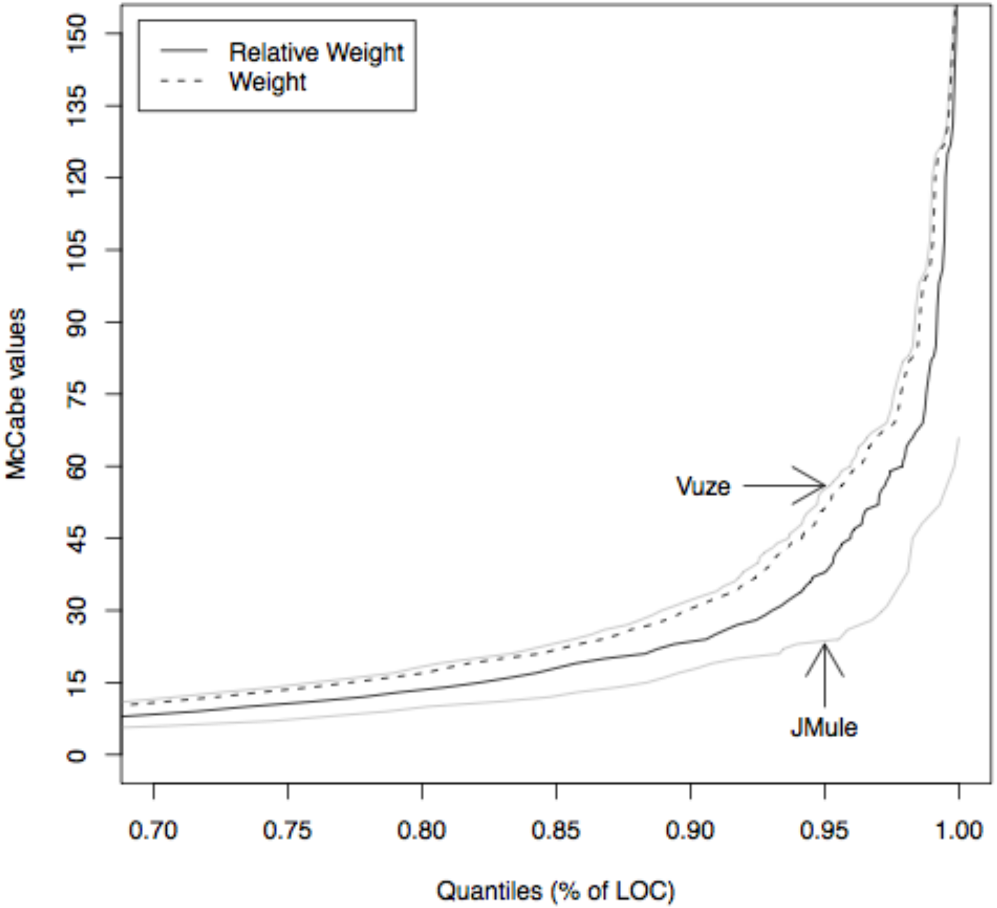


Derivation of risk thresholds: Weight by size

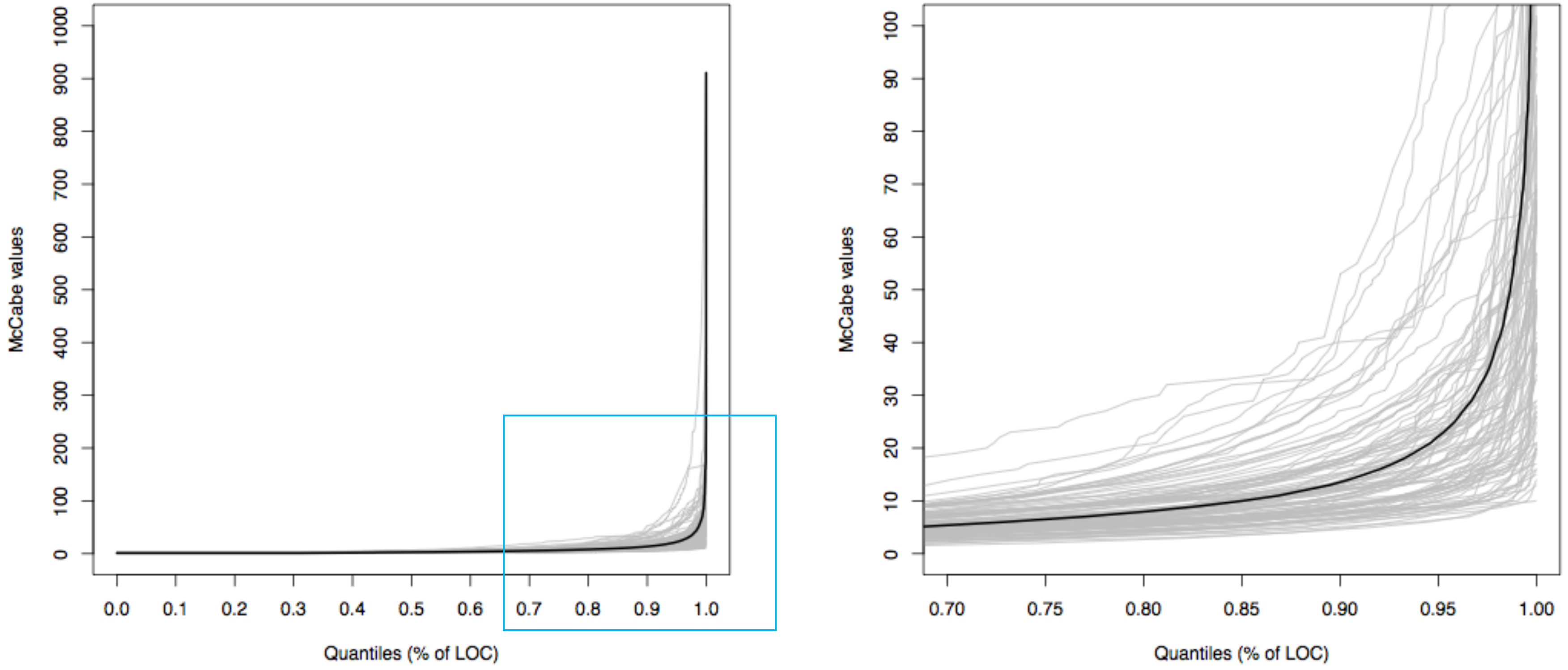




Derivation of risk thresholds
Summarizing a metric distribution

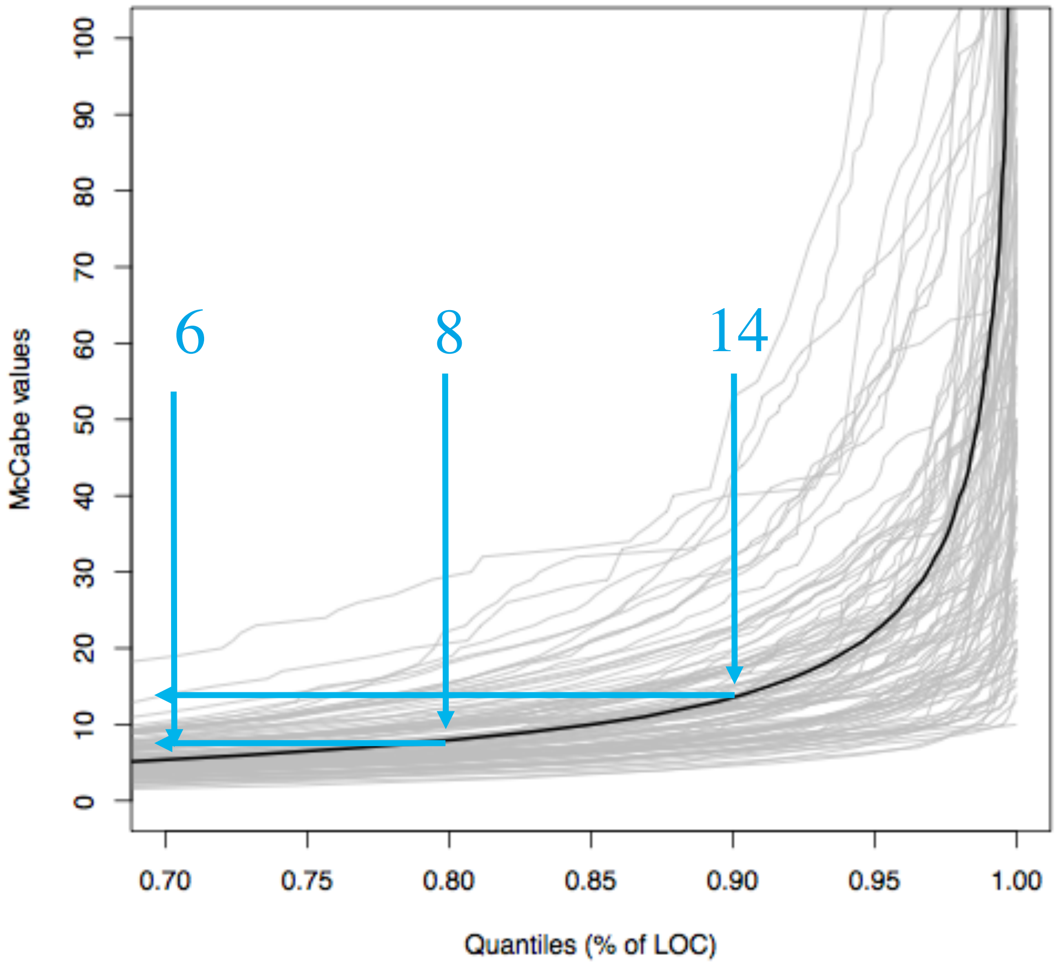


Derivation of risk thresholds
Relative weighting

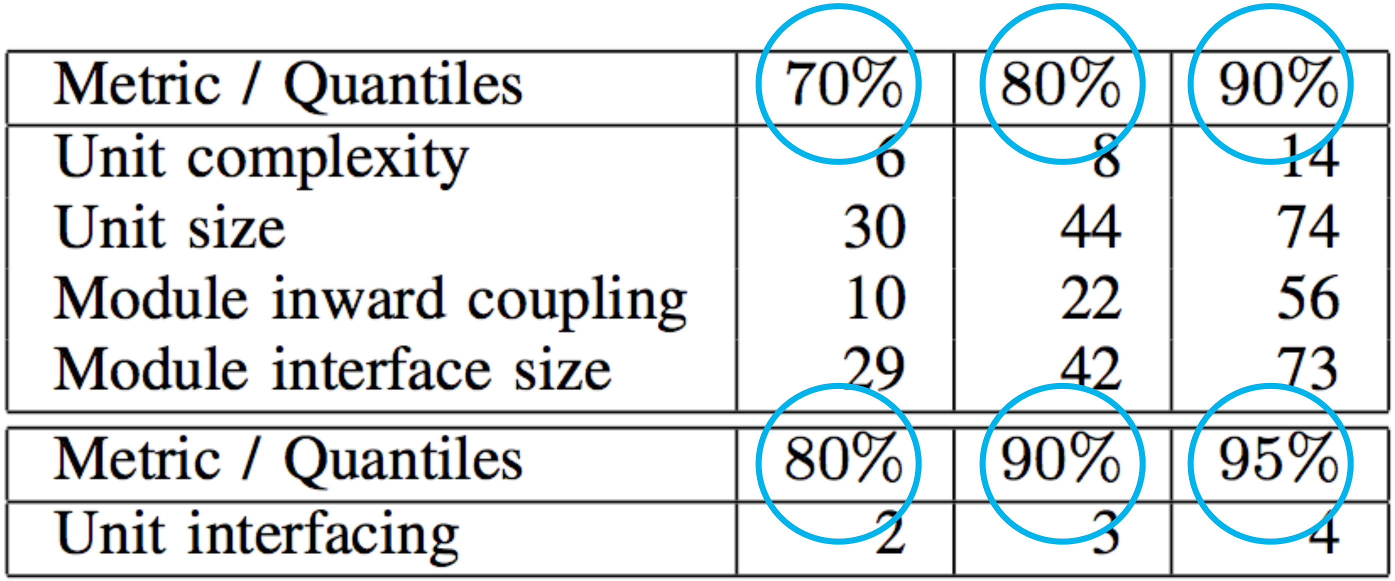


Derivation of risk thresholds: values

Software Improvement Group



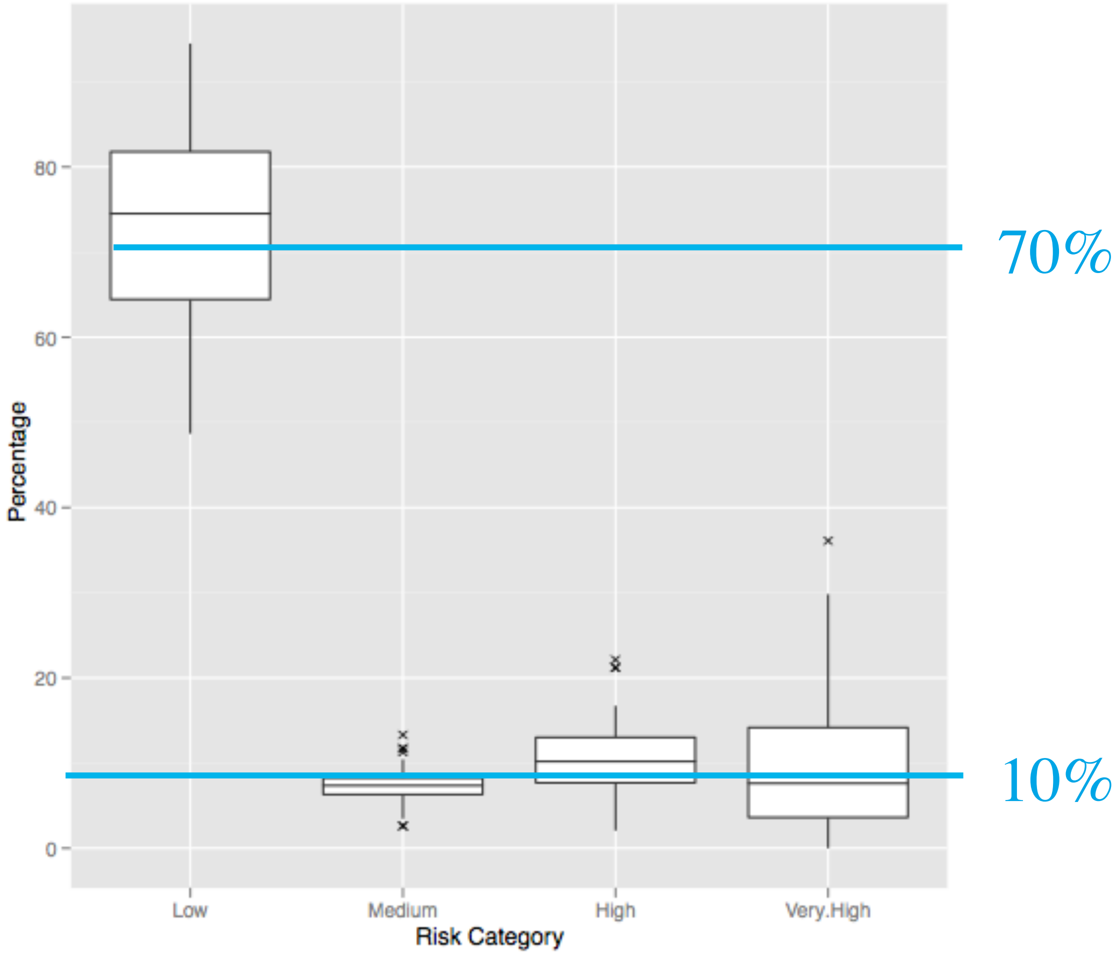
Risk Thresholds for the SIG Quality Model



Metric / Quantiles	70%	80%	90%
Unit complexity	6	8	14
Unit size	30	44	74
Module inward coupling	10	22	56
Module interface size	29	42	73

Metric / Quantiles	80%	90%	95%
Unit interfacing	2	3	4

Analysis of risk thresholds



Novel methodology to derive metric thresholds

- Solid methodology based on benchmark (depart from expert opinion)
- Agrees with expert opinion (thresholds are sensible)

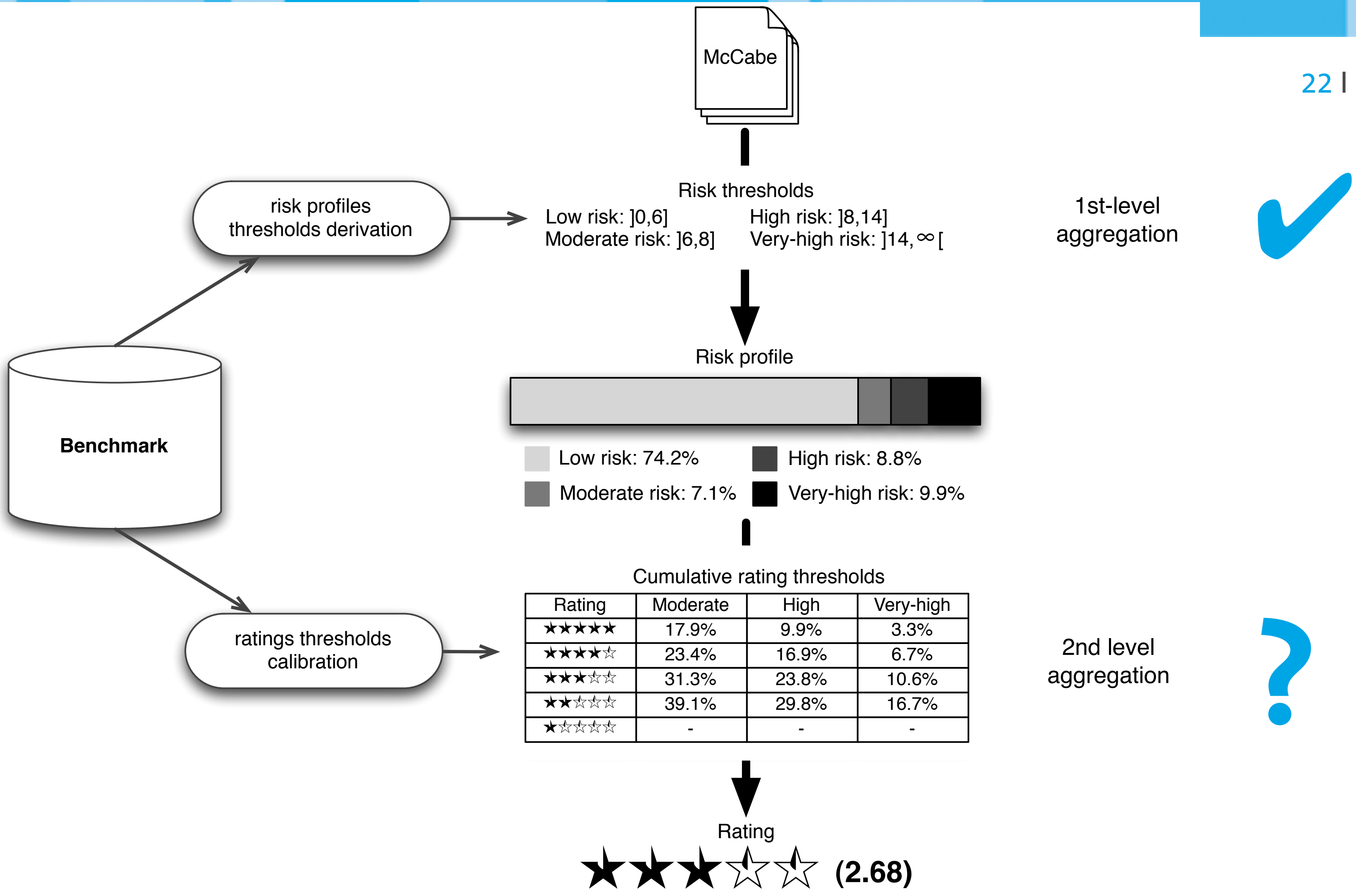
Plans for the future

- Validate with external characteristics

“The” Lessons

- We can attribute meaning to thresholds
- Benchmarks are of extreme importance

Benchmarking metrics to ratings



McCabe
Benchmark
risk profiles thresholds derivation
ratings thresholds calibration
1st-level aggregation
✓
2nd level aggregation
?

Risk thresholds

Low risk:]0,6]
Moderate risk:]6,8]
High risk:]8,14]
Very-high risk:]14,∞[

Risk profile

░ Low risk: 74.2%
▒ Moderate risk: 7.1%
▓ High risk: 8.8%
■ Very-high risk: 9.9%

Cumulative rating thresholds

Rating	Moderate	High	Very-high
★★★★★	17.9%	9.9%	3.3%
★★★★☆	23.4%	16.9%	6.7%
★★★☆☆	31.3%	23.8%	10.6%
★★☆☆☆	39.1%	29.8%	16.7%
★☆☆☆☆	-	-	-

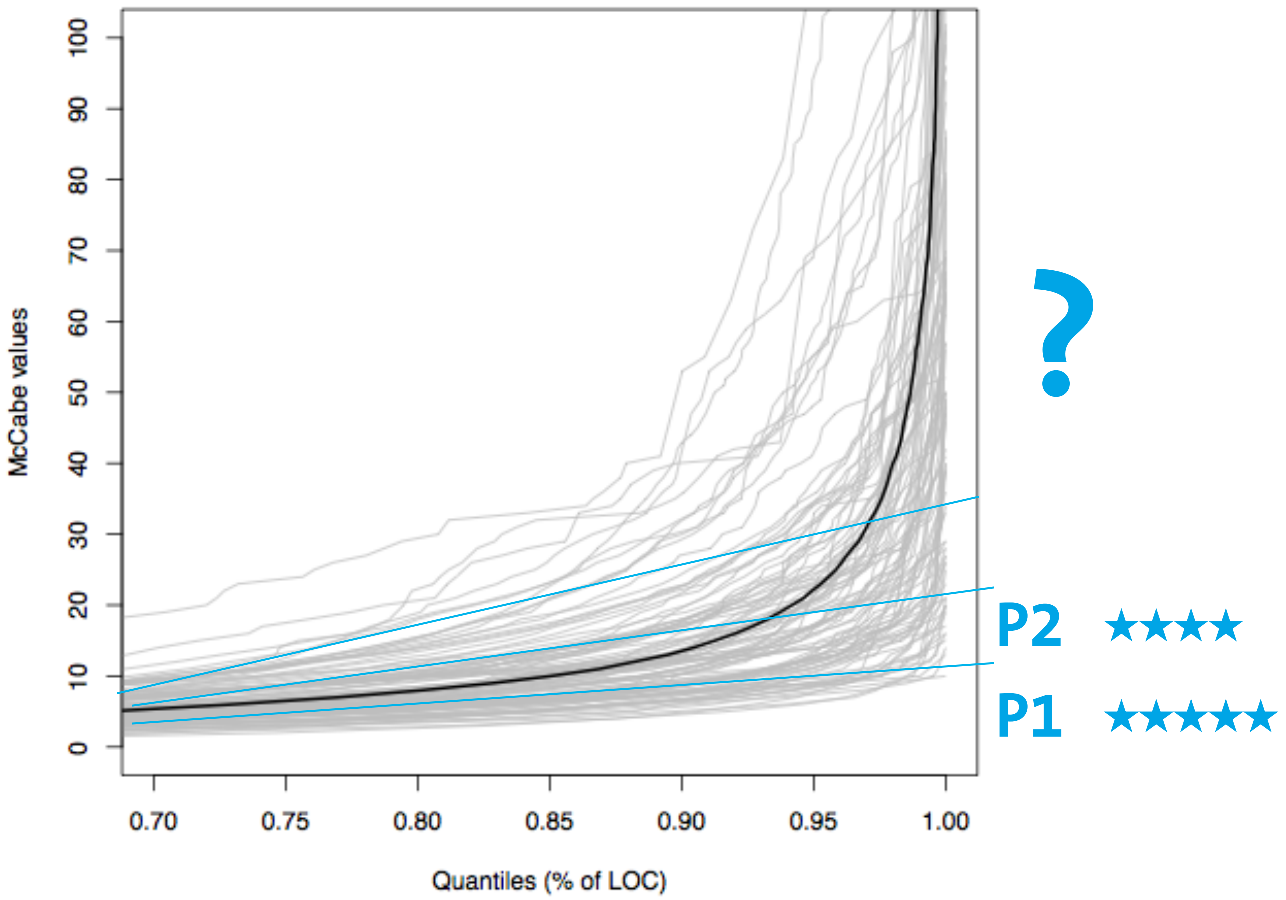
Rating

★★★☆☆ **(2.68)**

Part II

Calibration of rating thresholds

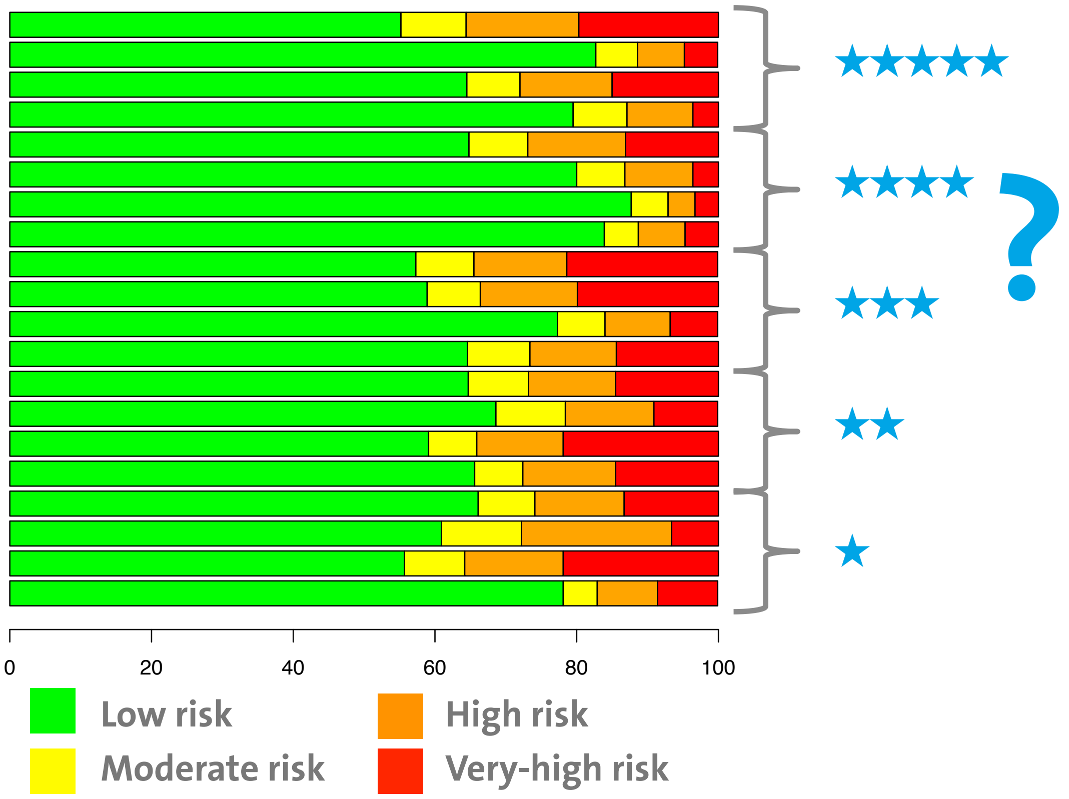
Benchmark partitioning
The problem



Benchmark partitioning
The problem #2

Software Improvement Group

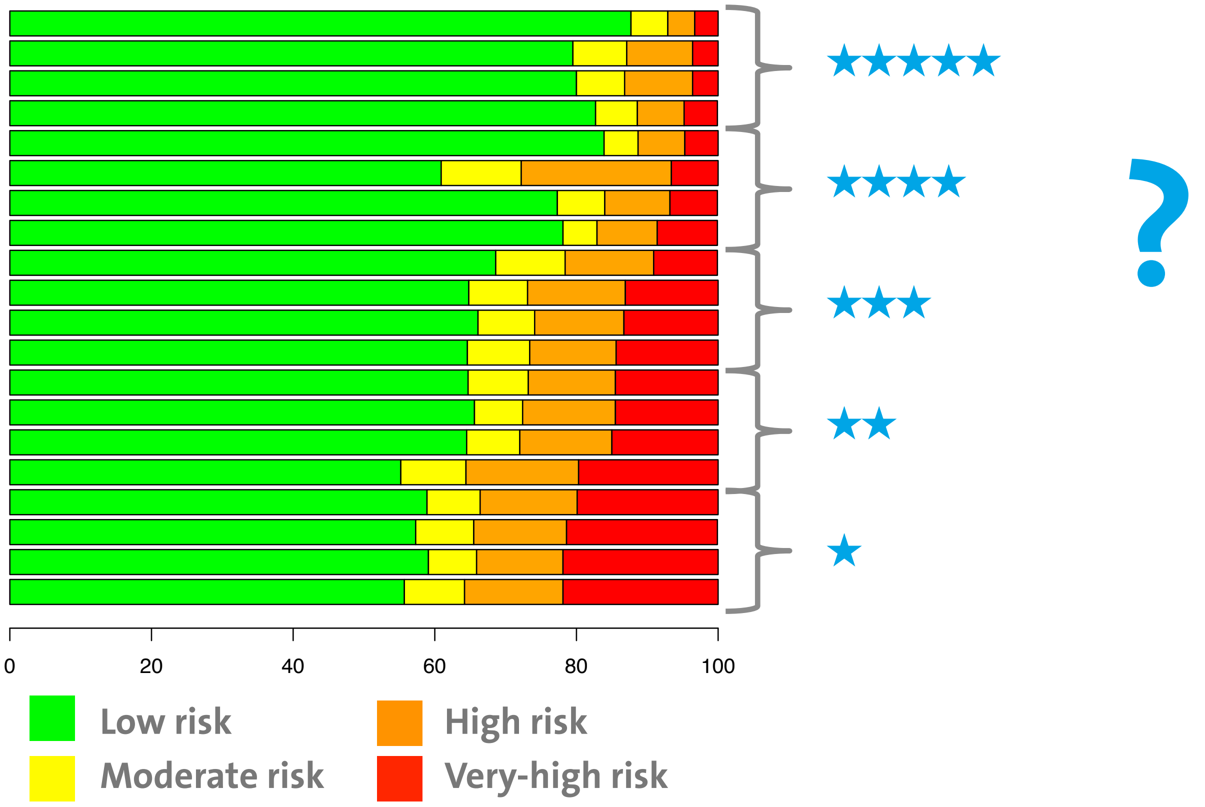
Unit complexity risk profiles for 20 random systems



Benchmark partitioning
Order by Very-high risk category

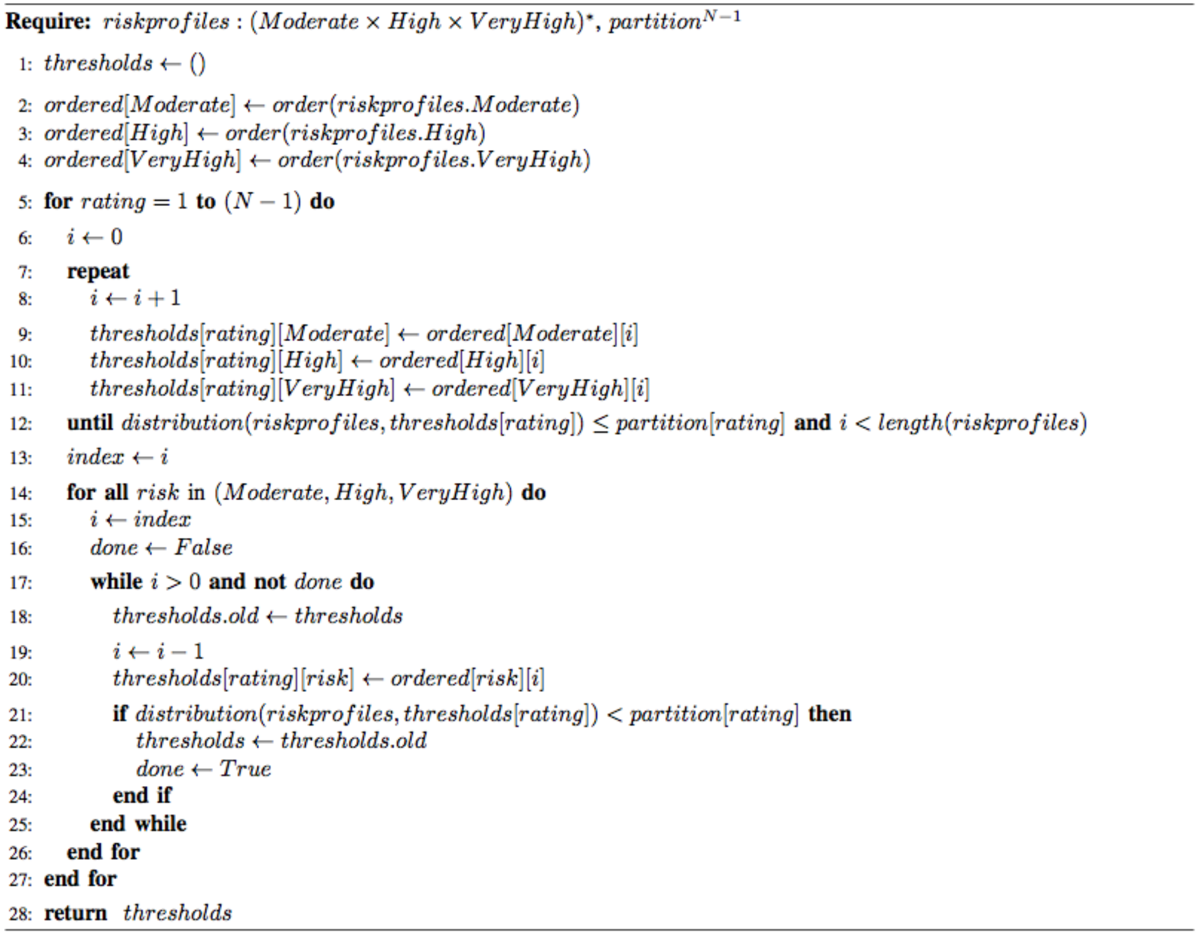
Software Improvement Group

Unit complexity risk profiles for 20 random systems



Other risk categories do not follow same order

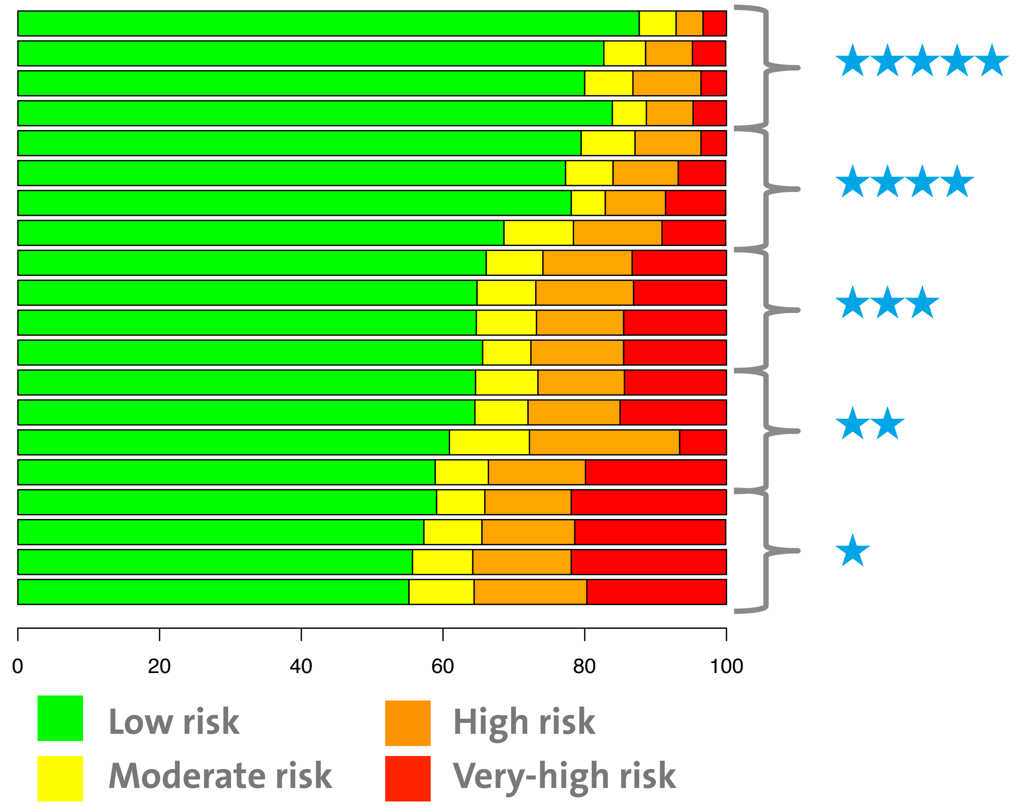
Ratings calibration algorithm



```
Require: $riskprofiles : (Moderate \times High \times VeryHigh)^*$, $partition^{N-1}$
$thresholds \leftarrow ()$
$ordered[Moderate] \leftarrow order(riskprofiles.Moderate)$
$ordered[High] \leftarrow order(riskprofiles.High)$
$ordered[VeryHigh] \leftarrow order(riskprofiles.VeryHigh)$
for $rating = 1$ to $(N - 1)$ do
  $i \leftarrow 0$
  repeat
    $i \leftarrow i + 1$
    $thresholds[rating][Moderate] \leftarrow ordered[Moderate][i]$
    $thresholds[rating][High] \leftarrow ordered[High][i]$
    $thresholds[rating][VeryHigh] \leftarrow ordered[VeryHigh][i]$
  until $distribution(riskprofiles, thresholds[rating]) \leq partition[rating]$ and $i < length(riskprofiles)$
  $index \leftarrow i$
  for all $risk$ in $(Moderate, High, VeryHigh)$ do
    $i \leftarrow index$
    $done \leftarrow False$
    while $i > 0$ and not $done$ do
      $thresholds.old \leftarrow thresholds$
      $i \leftarrow i - 1$
      $thresholds[rating][risk] \leftarrow ordered[risk][i]$
      if $distribution(riskprofiles, thresholds[rating]) < partition[rating]$ then
        $thresholds \leftarrow thresholds.old$
        $done \leftarrow True$
      end if
    end while
  end for
end for
return $thresholds$
```

Benchmark partitioning
Calibration algorithm result

Unit complexity risk profiles for 20 random systems



Calibrated rating thresholds
Unit complexity (McCabe)

Cumulative rating thresholds calibrated from 100 system benchmark

Rating	Low	Moderate	High	Very-High
	]0,6]	]6,8]	]8,14]	> 14
★★★★★	-	17.9%	9.9%	3.3%
★★★★	-	23.4%	16.9%	6.7%
★★★	-	31.3%	23.8%	10.6%
★★	-	39.1%	29.8%	16.7%
★	-	-	-	-

Calibration of rating thresholds
Concluding remarks

Novel methodology to aggregate metric to ratings

- Support of N-point scale (used 5-point star rating)
- Solid methodology based on benchmark
- Enables traceability using thresholds and risk profiles

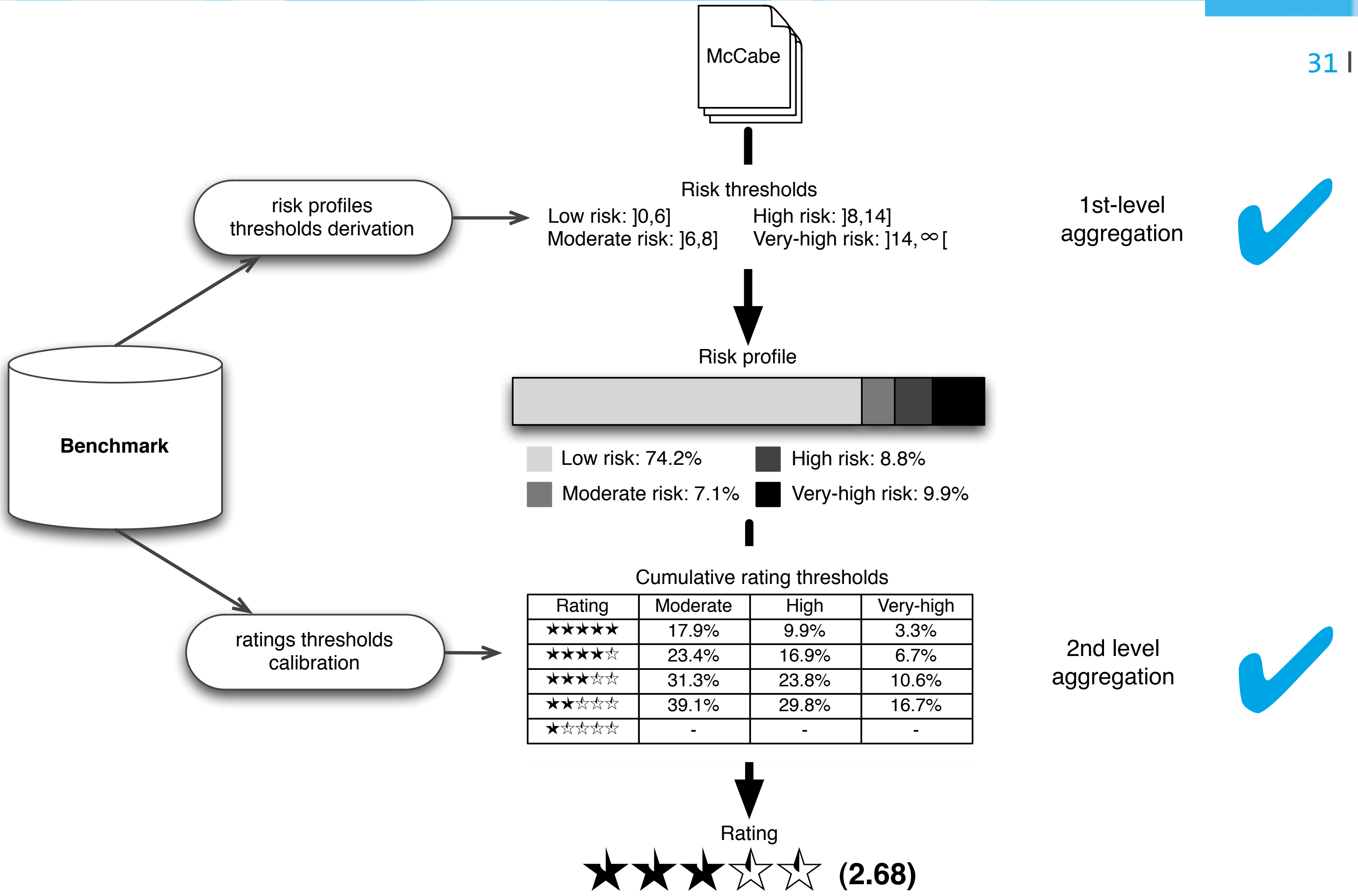
Plans for the future

- Validate with external characteristics
- Step for building quality models

“The” Lessons

- We can attribute meaning to ratings
- Ratings can be used to rank and evaluate software systems

Benchmarking metrics to ratings



Cumulative rating thresholds

Rating	Moderate	High	Very-high
★★★★★	17.9%	9.9%	3.3%
★★★★☆	23.4%	16.9%	6.7%
★★★☆☆	31.3%	23.8%	10.6%
★★☆☆☆	39.1%	29.8%	16.7%
★☆☆☆☆	-	-	-

Part III
Using ratings for quality evaluation

Simulators for Space Domain

EuroSim

- Commercial simulator
- Consortium: Dutch Space, NLR, TASK24
- Supports hardware-in-the-loop and man-in-the-loop
- Hard real-time
- Has non-space applications (e.g. aircraft simulation)

ESA/ESOC SimSat

- ESA owned (free)
- Mainly used for spacecraft telecommunication simulation
- Real-time

Technical Analysis on EuroSim
Research Motivation

Comparison of the technical quality between systems of the same domain

- Dutch Space EuroSim mk4.1.5
- ESA SimSat v4.0.1 issue 2

Research question

- How does the technical quality of EuroSim compare to SimSat?
  (how to use ratings to evaluate and compare quality)

Scope of the analysis

Analyzed programming languages

- C/C++
- Java

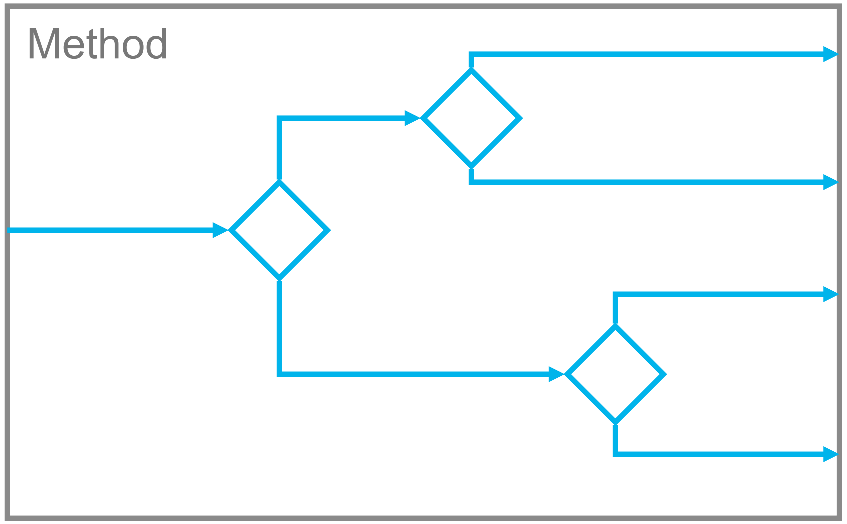
Considerations

- Production and test code analyzed separately
- Excluded documentation code and examples
- Excluded generated code (Icon files generated by XMP)
- Excluded open-source libraries
- Excluded drivers supplied by hardware suppliers
  - device drivers developed by EuroSim Consortium were included in the analysis
- Excluded code not in use: PerfIGS, LibCadese

Unit Complexity definition

- McCabe, *IEEE Transactions on Software Engineering*, 1976
- Cyclomatic Complexity = Number of decision points per unit (method/function)

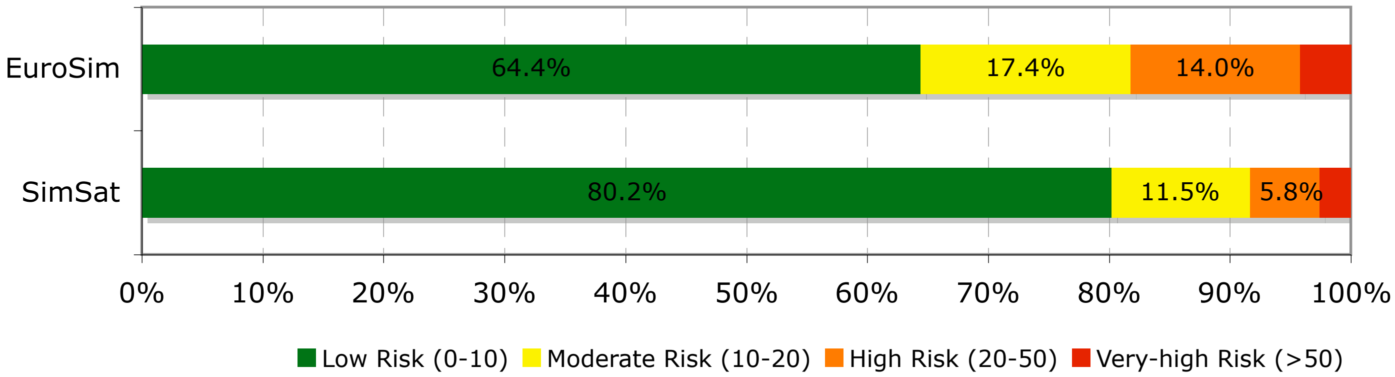
- Widely accepted measurement for code complexity
- Should be as small as possible



McCabe: 4

Unit Complexity comparison

Software Improvement Group



Discussion

- Both EuroSim and SimSat rank two stars
- Very-high complexity was found: 4.2% for EuroSim and 2.6% SimSat
- For EuroSim, very-high complexity is localized in 16 methods.
- Taking the last three risk categories, EuroSim has almost twice the risk of SimSat
- For EuroSim, comparing versions reveals slow decrease of the quality

ISO 9126 Maintainability
Comparison between EuroSim and SimSat

Software Improvement Group

		Volume	Duplication	Unit size	Complexity	Unit interfacing	Test quality	
Dutch Space EuroSim	Score	★★★★	★★★	★★	★★	★★	★★	Score
	Analysability	X	X	X				★★★
	Changeability		X		X		X	★★
	Stability					X	X	★★
	Testability			X	X			★★

		Volume	Duplication	Unit size	Complexity	Unit interfacing	Test quality	
ESA SimSat	Score	★★★	★★	★★	★★	★★★	★★	Score
	Analysability	X	X	X				★★★
	Changeability		X		X		X	★★
	Stability					X	X	★★★
	Testability			X	X			★★

Dutch Space Eurosim Maintainability: ★★★

ESA SimSat Maintainability: ★★

Conclusion

Contributions

- Generic methodology to use metrics for quality evaluation
- Demonstration of the methodology using space-domain software

Benchmark-based approach benefits

- Meaningful (relation with industrial systems)
- Operational (thresholds can be obtained automatically)

Future work

- Use the methodology to validate metrics external characteristics
- Finalize PhD thesis!!!

Shameless Commercial Alert

Software Improvement Group

- Internships (currently hosting 3 PhD + 4 MSc students)
- Jobs
- Research cooperation

- Very interesting clients
- You get to see lots and lots of source code from all over the world
- Software Engineering or Consultancy skills (or both)

- See www.sig.eu

More info? Feel free to contact...

Tiago L. Alves

E: t.alves@sig.eu
W: www.sig.eu
T: +31 20 3140950