SoftNet 2018 Panel on Software and Systems

Trends in Software Development and Verification

Markus Kelanti, University of Oulu, Finland
Luigi Lavazza, Università degli Studi dell'Insubria, Italy
Martin Zinner, Center of Information Services and High Performance Computing , University of Dresden, Germany
Roy Oberhauser, Aalen University, Germany

SoftNet 2018 Panel on Software and Systems

Trends in Software Development and Verification

Roy Oberhauser
Aalen University
Germany

Trends Affecting Development

- DevOps
- Post-Scrum
- Serverless
- Software-Defined Architecture
- Platform-as-a-Service
- Low-Code
- Augmented Software Development

Information visualisation in software development

Markus Kelanti

Empirical Software Engineering in Software, Systems and Services (M3S)

Faculty of Information Technology and Electrical Engineering

University of Oulu

Markus Kelanti

Bio

2015 Received M.Sc. in University of Oulu in the field of software engineering.

2017 Received Ph.D. after defending his thesis over stakeholder analysis in large-scale software-intensive systems.

Currently a postdoctoral researcher in the Faculty of Information Technology and Electrical Engineering at the University of Oulu.

Today's activities include project research in IoT analytics and web application development and teaching software engineering courses.

Research

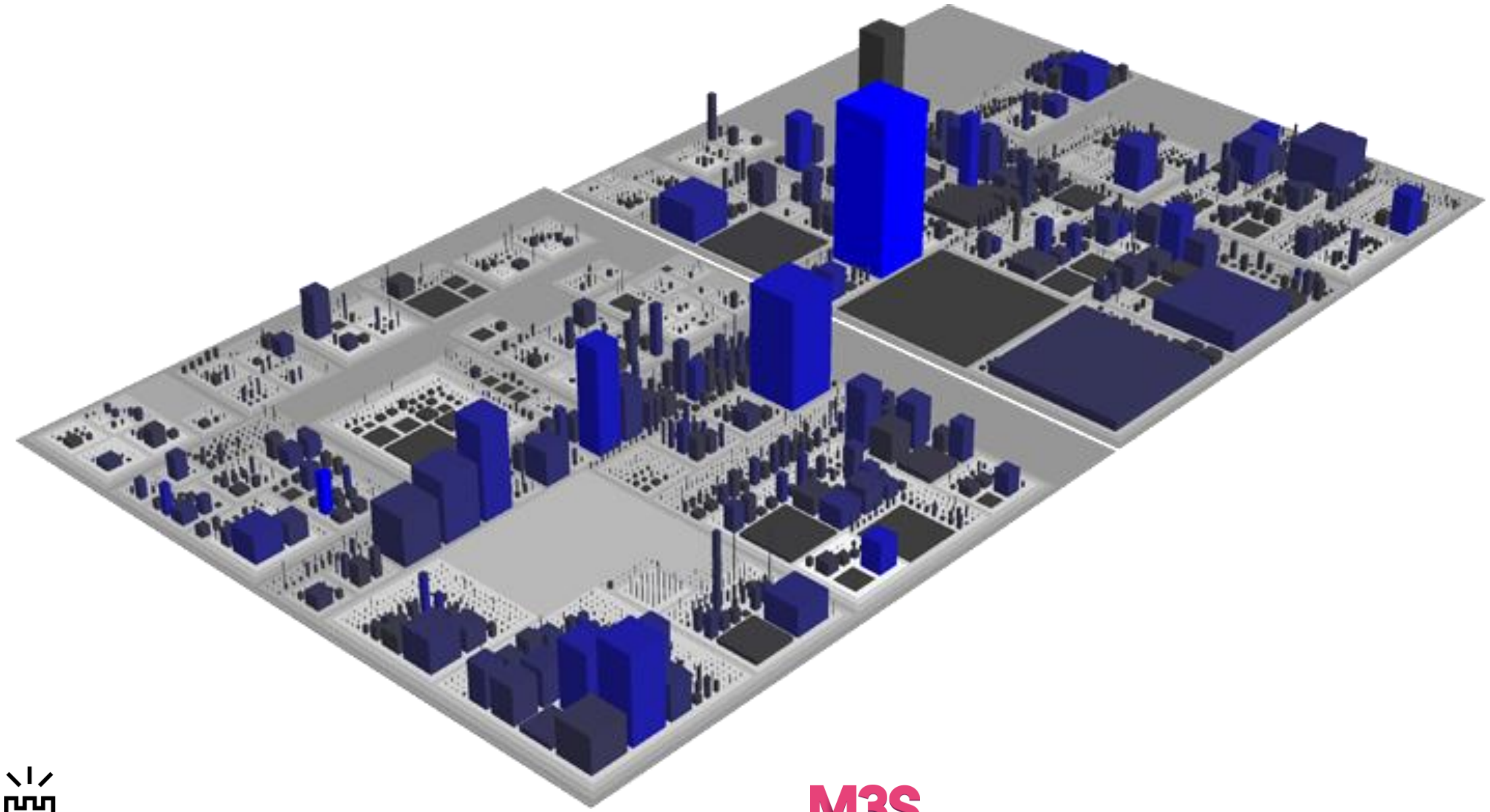
Past activities: Stakeholder analysis, very-large scale software-intensive systems development, RE tool integration, agile software development and software processes.

Current activities: IoT systems analytics and development, software metrics and analytics, software service development and information visualisation in software production.

Information visualisation

- Generally a presentation of information (often in incomprehensible format) in a visual format to aid human understanding
- A hot topic in software development:
  - Allows a quick understanding of a situation (feature completion)
  - Provides summaries from complex data sets (metrics)
  - Directs decision-making (feature use forecasts)
  - Etc…
- Potential in visualisation has sparked new research initiatives and start-ups to cater the need for new and novel visualisation approaches

Code as city with a chosen metric [1]

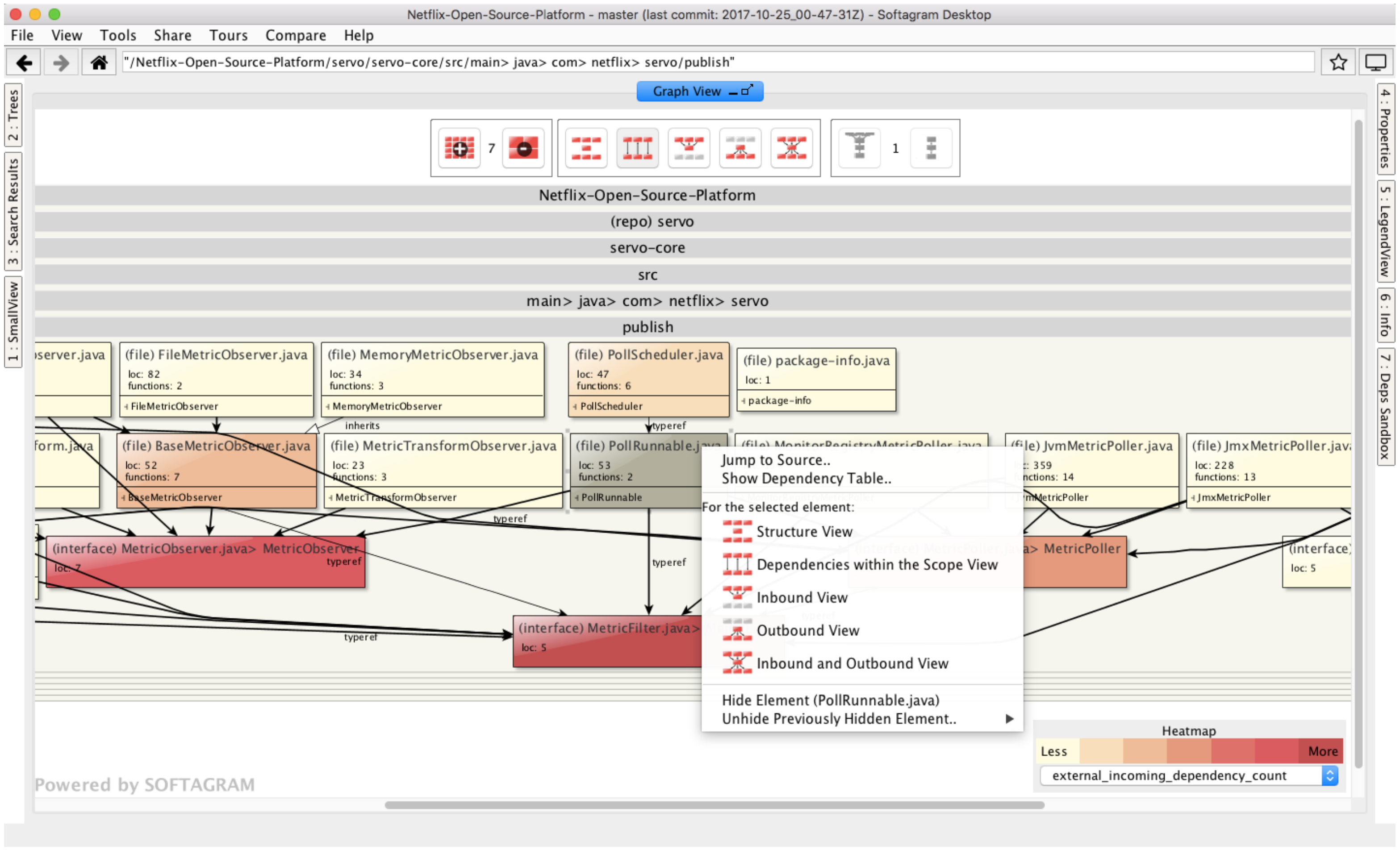


UNIVERSITY OF OULU

M3S

[1] https://wettel.github.io/codecity.html

Code structure visualisation [2]

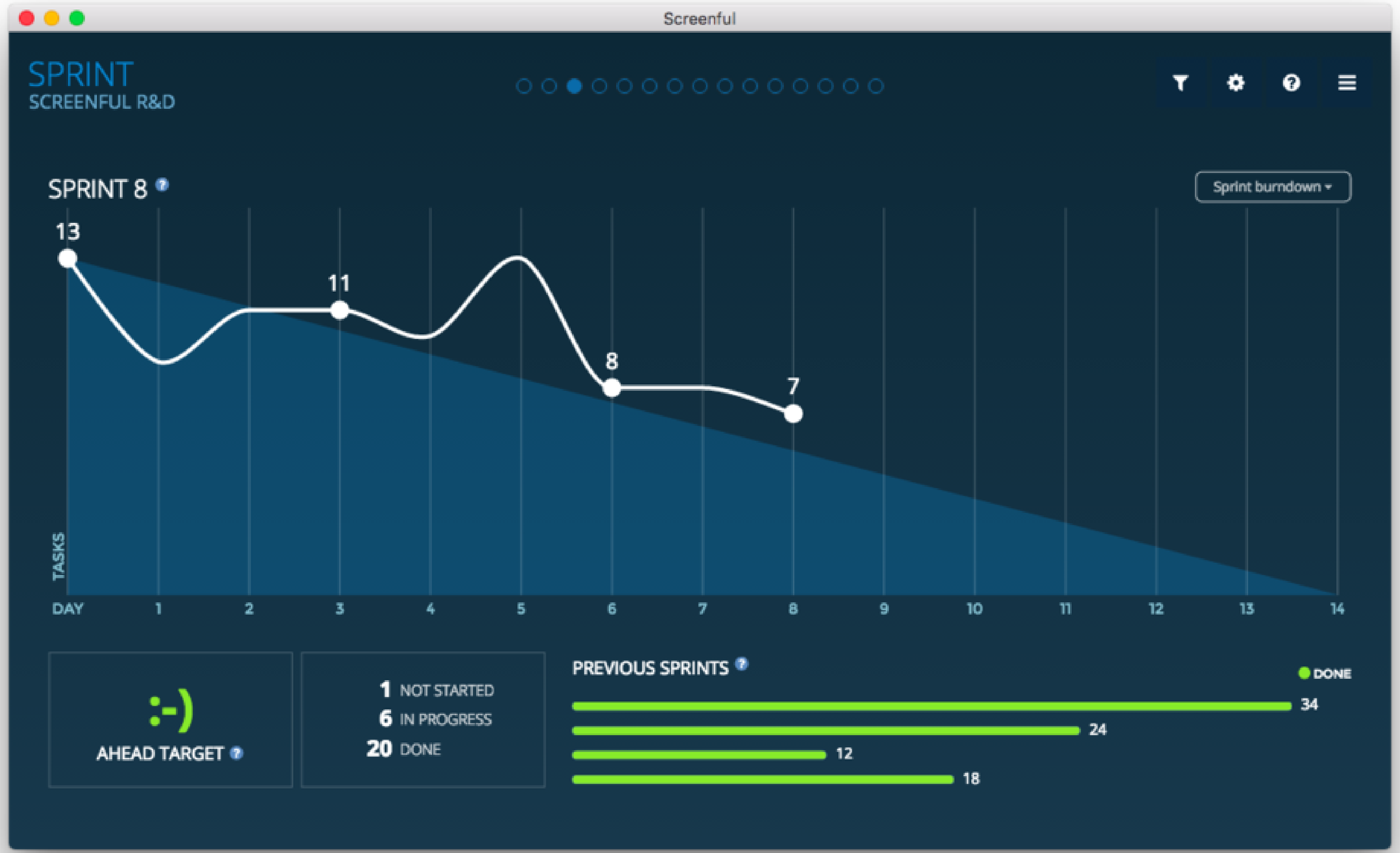

Netflix-Open-Source-Platform - master (last commit: 2017-10-25_00-47-31Z) - Softagram Desktop
File View Tools Share Tours Compare Help
"/Netflix-Open-Source-Platform/servo/servo-core/src/main> java> com> netflix> servo/publish"
Graph View
Netflix-Open-Source-Platform
(repo) servo
servo-core
src
main> java> com> netflix> servo
publish
(file) FileMetricObserver.java loc: 82 functions: 2
(file) MemoryMetricObserver.java loc: 34 functions: 3
(file) PollScheduler.java loc: 47 functions: 6
(file) package-info.java loc: 1
(file) BaseMetricObserver.java loc: 52 functions: 7
(file) MetricTransformObserver.java loc: 23 functions: 3
(file) PollRunnable.java loc: 53 functions: 2
(file) JvmMetricPoller.java
(file) JmxMetricPoller.java loc: 228 functions: 13
(interface) MetricObserver.java> MetricObserver loc: 7
(interface) MetricFilter.java> loc: 5
inherits
typeref
Jump to Source..
Show Dependency Table..
For the selected element:
Structure View
Dependencies within the Scope View
Inbound View
Outbound View
Inbound and Outbound View
Hide Element (PollRunnable.java)
Unhide Previously Hidden Element..
Heatmap
Less
More
external_incoming_dependency_count
Powered by SOFTAGRAM
1 : SmallView
2 : Trees
3 : Search Results
4 : Properties
5 : LegendView
6 : Info
7 : Deps Sandbox

UNIVERSITY OF OULU

M3S

[2] https://softagram.com/softagram-cloud

Sprint burn down chart [3]



:-)
AHEAD TARGET

1 NOT STARTED
6 IN PROGRESS
20 DONE

UNIVERSITY OF OULU

M3S

[3] https://screenful.com

Plenty of solutions and opportunities

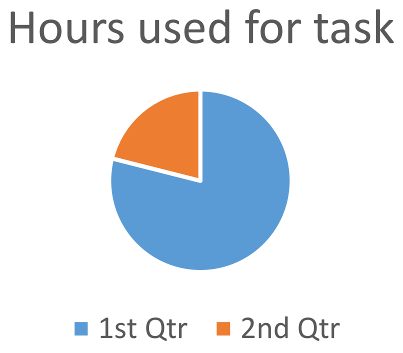
But that is not the point of this talk

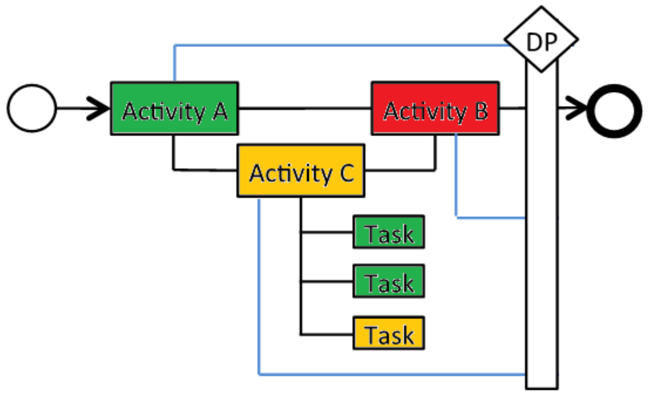
Challenges in information visualisation in software development

In the field, where visualisations are needed and used

New (and old) complex information

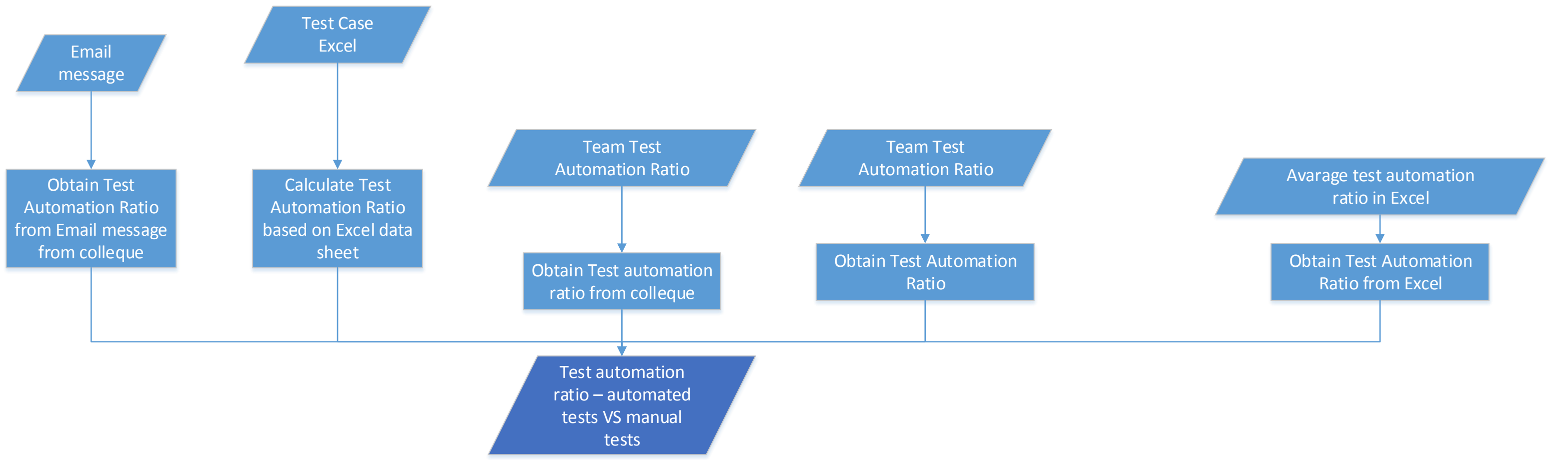
- Certain visualisation techniques and models are well-established for known needs:
  - UML class structure diagrams
  - Code coverage graphs
  - Etc…
- But when asking the needs of the companies they express visualisation needs in decision-making that require inclusion of human activity
  - The special favorite in this category is the "when-something-is-done" (for example when feature is done)

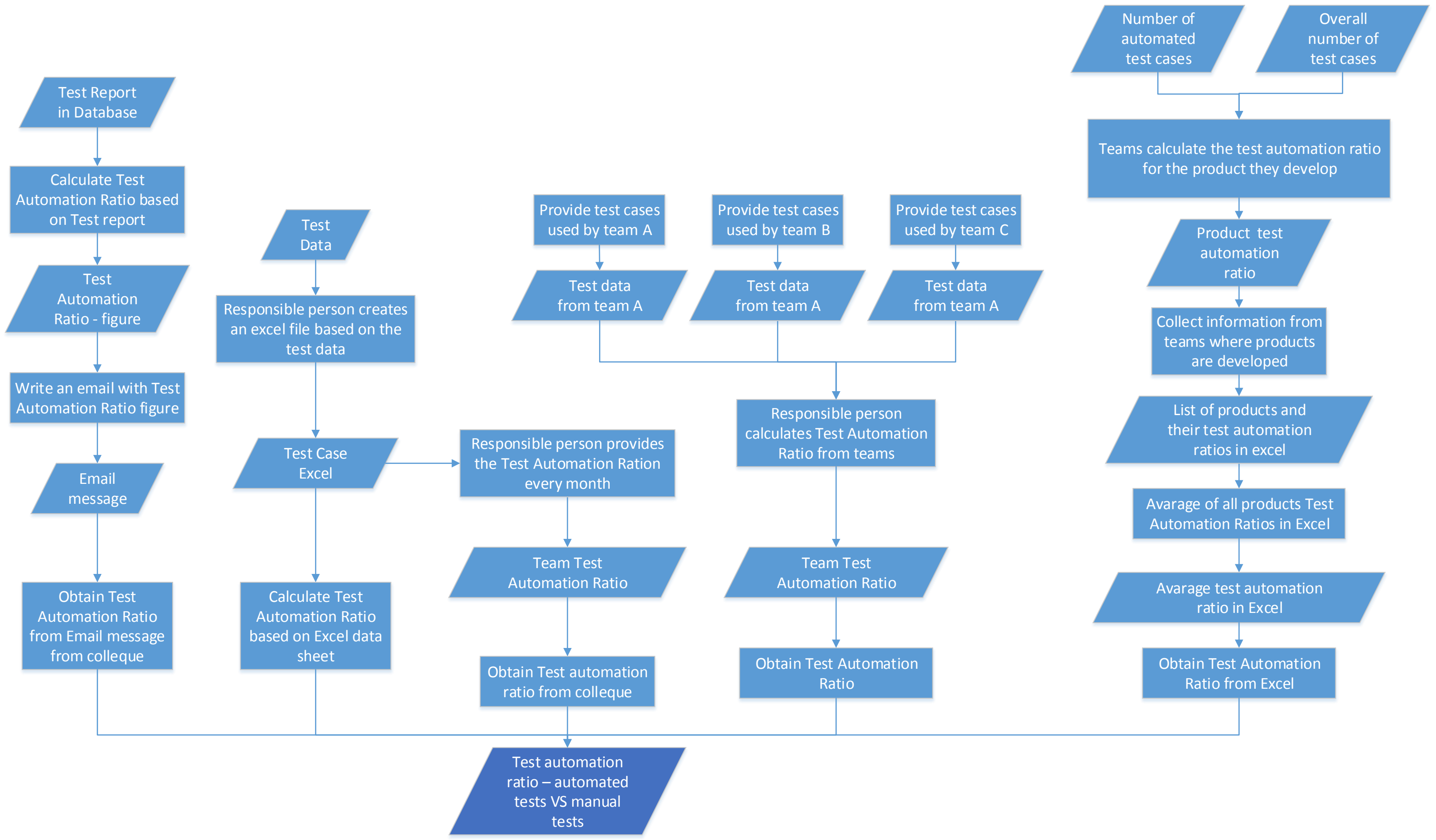




Easy to understand

- "Intuitive" seems to be the keyword when discussing with companies
  - Often they want simplicity as there is limited amount of time available for the team
  - This can be also a problem of perceived value of the visualisation itself
- Information overflow
  - Cannot feed too much information as it takes then too much time and resources to understand what the visualisation wants to tell


Email message
Test Case Excel
Obtain Test Automation Ratio from Email message from colleque
Calculate Test Automation Ratio based on Excel data sheet
Team Test Automation Ratio
Obtain Test automation ratio from colleque
Team Test Automation Ratio
Obtain Test Automation Ratio
Avarage test automation ratio in Excel
Obtain Test Automation Ratio from Excel
Test automation ratio – automated tests VS manual tests



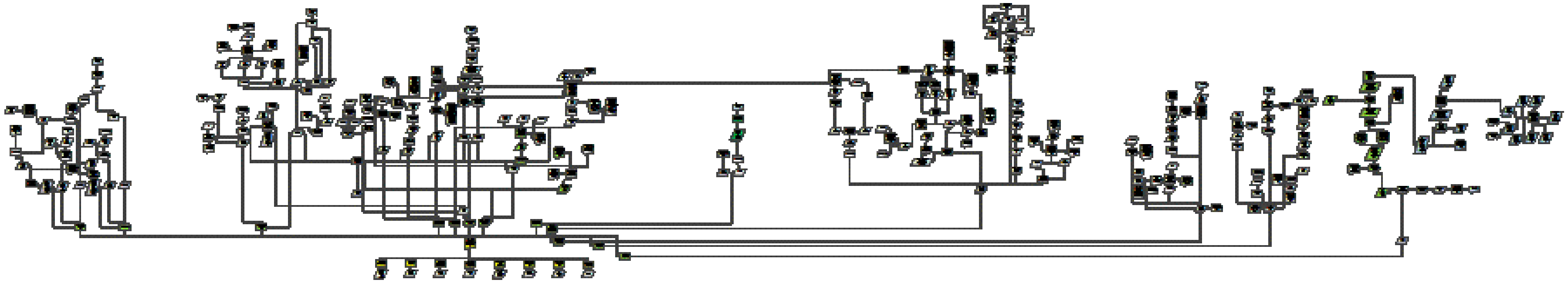
Test Report in Database
Calculate Test Automation Ratio based on Test report
Test Automation Ratio - figure
Write an email with Test Automation Ratio figure
Email message
Obtain Test Automation Ratio from Email message from colleque
Test Data
Responsible person creates an excel file based on the test data
Test Case Excel
Calculate Test Automation Ratio based on Excel data sheet
Responsible person provides the Test Automation Ration every month
Team Test Automation Ratio
Obtain Test automation ratio from colleque
Provide test cases used by team A
Provide test cases used by team B
Provide test cases used by team C
Test data from team A
Test data from team A
Test data from team A
Responsible person calculates Test Automation Ratio from teams
Team Test Automation Ratio
Obtain Test Automation Ratio
Number of automated test cases
Overall number of test cases
Teams calculate the test automation ratio for the product they develop
Product test automation ratio
Collect information from teams where products are developed
List of products and their test automation ratios in excel
Avarage of all products Test Automation Ratios in Excel
Avarage test automation ratio in Excel
Obtain Test Automation Ratio from Excel
Test automation ratio – automated tests VS manual tests

UNIVERSITY OF OULU

M3S

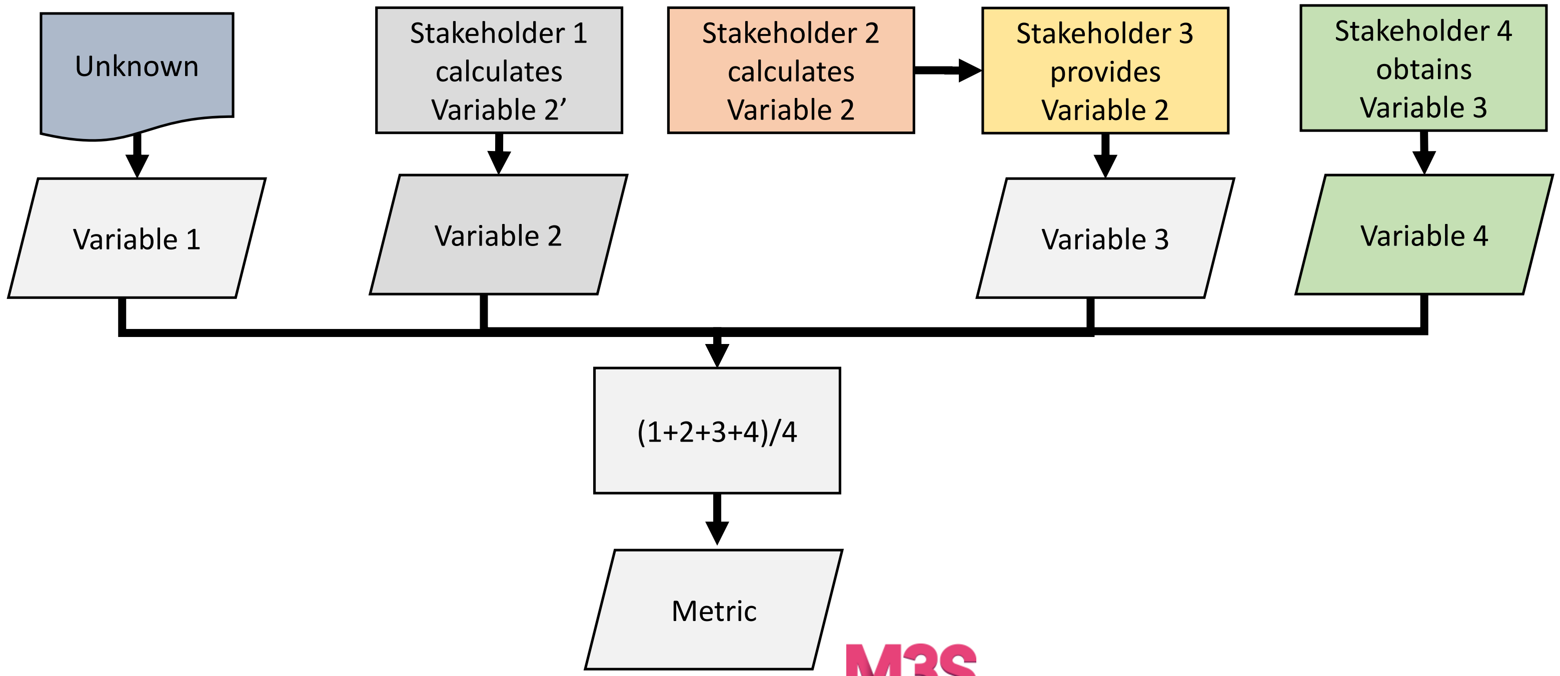
But still comprehensible

- The problem of the simple visualisation, it hides too much information and increases the change of wrong conclusions
- The need has been for a visualisation that can explain itself, usually with different abstraction levels (and viewpoints)



Validity of visualisation

- It is often assumed that information and its formulation is valid, users often perceive the visualisation which is the end result
- However, the validity of visualisation is tied to the data it visualises and how it has been formulated before it is seen by user


Unknown
Stakeholder 1 calculates Variable 2'
Stakeholder 2 calculates Variable 2
Stakeholder 3 provides Variable 2
Stakeholder 4 obtains Variable 3
Variable 1
Variable 2
Variable 3
Variable 4
(1+2+3+4)/4
Metric

Stakeholders require visualisation from their viewpoint

- When one makes a decision, a specific set of information is needed in a specific format allowing one to make a decision
- The information is often "standardised" in different domains
    - For example agile sprint burn chart
- However, in practice the set of information and format vary depending on practices, experience, organisation and other factors
    - Often individual stakeholders need to have a visualisation on the same data as others but in different way to support their work
- Ignoring this need causes stakeholders to find an other way to find the information they need
    - It can lead to situations where the visualisation tool is modified, abandoned or simply replaced with something else

Implementation of a visualisation tool

- Companies often use a set of visualisation tools bought for their teams
  - Selection criteria varies, recently tools have been selected more based on developer needs
- On the other hand, some tools are selected and used based on personal preference of the user
- The use of these tools depend on availability on resources
  - The problem is especially visible with “1-person teams” where the project is small and resources should be invested on actual code development
  - There is a need for visualisation tools and methods that can be implemented and run as automatically as possible (cf. Docker)

Cost of visualisation

- Visualisation costs are easily absorbed in large companies
  - Allows smaller teams to utilise powerful tools in development
- The bar rises to apply visualisations as the companies get smaller, unless:
  - Obtained value overcomes the cost of visualisation
  - Implementation and maintenance requires minimal effort
- Essentially we can visualise almost anything but the companies want to adapt those that are effective in cost and resource -vise

Comments?

Università degli Studi dell’Insubria
Dipartimento di Scienze Teoriche e Applicate

Trends in software development and verification

Management of code smells and technical debt in the software development process

Luigi Lavazza

Dipartimento di Scienze Teoriche e Applicate

luigi.lavazza@uninsubria.it

Situation

- There is an increasing attention to the quality of code.
- In the past, quality was mainly checked during testing.
- Now quality is increasingly evaluated and improved during development
- Probable reasons
  - because of the increased familiarity with concepts like refactoring and Test-Driven Development, made popular by agile processes
  - Good marketing: the term ***technical debt*** suggests that you lose money with bad quality and managers are quite sensible to money-related arguments.

Caring for quality is OK, but ...

- In general, the results you get depend on *how* you manage technical debt.
- Key issues:
  a) identification and quantification of debt
  b) management of debt

Identification and quantification of debt

- Problems observed
  - Ill-defined measures
  - Measures of dubious utility
  - Code smell misinterpretation and misuse

Ill defined measures

- Example: “cognitive complexity”, an alternative to McCabe’s Cyclomatic Complexity
- Problems:
  - Defined via a few examples. No rigorous definition based on syntactic and semantic properties of programs.
  - No experimental evidence given that cognitive complexity
    - is not correlated with McCabe’s complexity (or any other code measure)
    - provides the sought advantages with respect to McCabe’s complexity

Measures of dubious utility

- SonarQube provides over 200 types of measurement-based “violations”.
  - The so-called squids
- There is no evidence that these violations are actually dangerous

- Similarly, code smells (proposed by Beck and Fowler) are widely used, but empirical research on their actual effects on code quality provides no conclusive indications
  - Some researchers even found that some smell in some cases appear to increase software quality

Code smell misinterpretation and misuse

- Code smells were introduced to let programmers recognize code that needs refactoring
- Originally, they were meant to be used in “manual” code inspections
  - So that programmers could recognize the occurrence of the “anti-pattern” and the existence of some danger at the same time

- Now they are often identified automatically using static code analysis
- Problem: models and thresholds used to decide that a piece of code smells are not based on meaningful relations with external qualities
  - GodClass(c) iff (($WMC(c) \geq 47$) and ($TCC(c) < 1/3$) and ($ATFD(c) > 5$)
    - WMC = Weighted Methods per Class
    - TCC = Tight Class Cohesion
    - ATFD = Access To Foreign Data
- As a result, many false positives are obtained

A conceptual error

- Let’s take definition of God Class
  - GodClass(c) iff ((WMC(c)≥47) and (TCC(c)<1/3) and (ATFD(c)>5)
- In this way, we are just transforming measures of internal code properties (WMC, TCC, ATFD) into yet another internal property of code.
- Suppose that God Classes are actually dangerous for maintainability

$$GodClass \xrightarrow{affects} Maintainability$$

- If being a god class actually depends on WMC, TCC and ATFD, it is

$$\{WMC, TCC, ATFD\} \xrightarrow{define} GodClass$$

- Well, we do not need the “God Class” concept at all:

$$\{WMC, TCC, ATFD\} \xrightarrow{affects} Maintainability$$

Management of debt

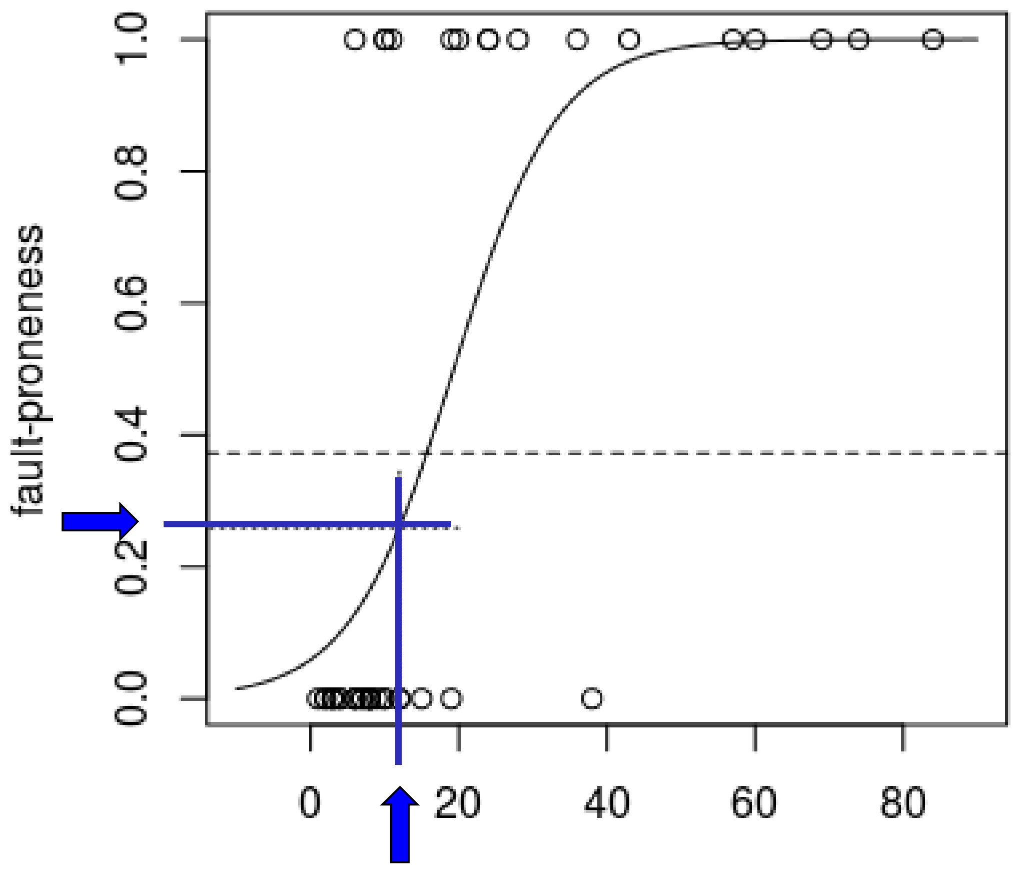
- Problems
  - No methodologies
  - No measure of the actual effectiveness
- Several organizations impose that
  - Smells and “violations” be detected using some (usually commercial) tool
  - Some types of smells and violations be removed asap, so as to keep the code “clean” with respect to the supposedly dangerous smells
- However, these organizations often do not have any evidence that these directives are actually effective
  - It is possible that they are dedicating time and effort to deal with "issues" of no consequence
  - It is possible that the process of dealing with these issue is not as efficient as it could be

Suggestions

- Measure rigorously
  - internal qualities (e.g., code size, complexity, cohesion, coupling, etc.)
  - external qualities (faultiness, maintainability, etc.)
- Build models that relate external qualities to internal qualities
  - Using your own data (i.e., do not trust what happens elsewhere)
  - Using rigorous statistical analysis
- Define thresholds for internal qualities based on the minimum acceptable value of external quality measures
- Recommend developers to keep internal quality compliant with the thresholds
- Refactor code when internal measure thresholds are violated

Thresholds accounting for fault-proneness

Maximum acceptable probability that the class is faulty (*based on business-relevant considerations!*)



Threshold on WMC

TECHNISCHE UNIVERSITÄT DRESDEN

DRESDEN concept

Center for Information Services and High Performance Computing (ZIH)

Panel Presentation: Outlier Detection

ICSEA 2018, Nice France

October 17, 2018

Martin Zinner (martin.zinner1@tu-dresden.de)

Definition of Outliers

An exact definition of an outlier depends on assumptions regarding the data structure and the applied detection method:

- Hawkins: An observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism.
- Barnet and Lewis: Outlier is one that appears to deviate markedly from other members of the sample in which it occurs.

Outliers can be of two kinds**:**

- Data entry errors (and similar).
- Legitimate data that is unusal.

Bibliography:

Hans-Peter Kriegel et al.: “Outlier Detection Techniques” (2009), https://www.siam.org/meetings/sdm10/tutorial3.pdf

Extreme Value Analysis - 1

Method for Multivariate Outlier Detection:

— Often indicate those observations that are located relatively far from the center.

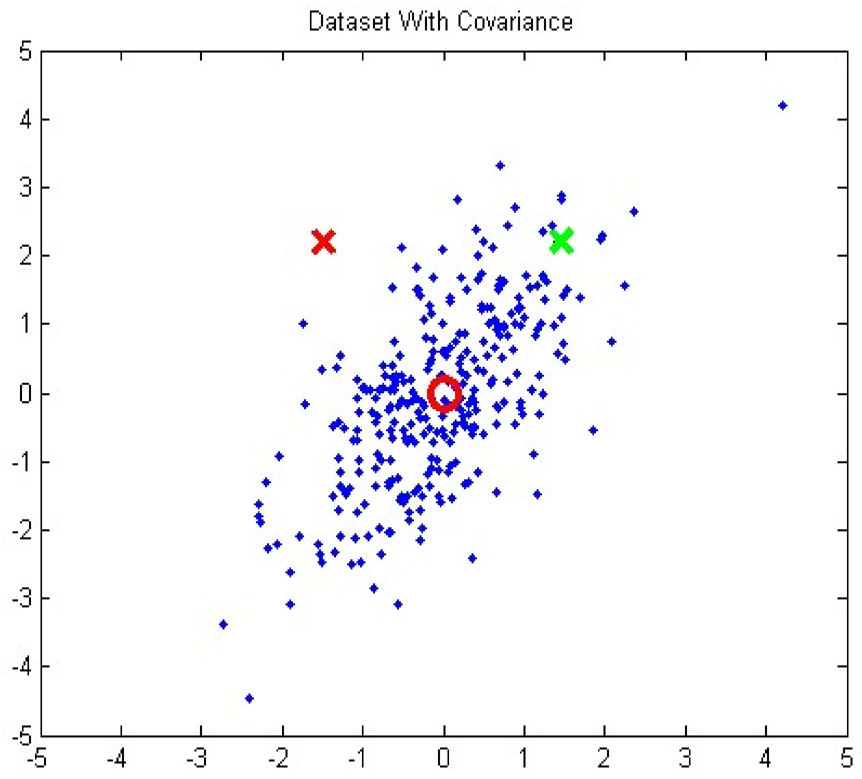
— Several distances can be implemented.

Question:

What is the difference between the red point and the green one?

Intuition:

The red point is less likely to belong to the cluster than the green one.



Bibliography:

Chris McCormick: “Mahalanobis Distance“ (2014), http://mccormickml.com/2014/07/22/mahalanobis-distance/

Extreme Value Analysis - 2

Normalization of the Data:

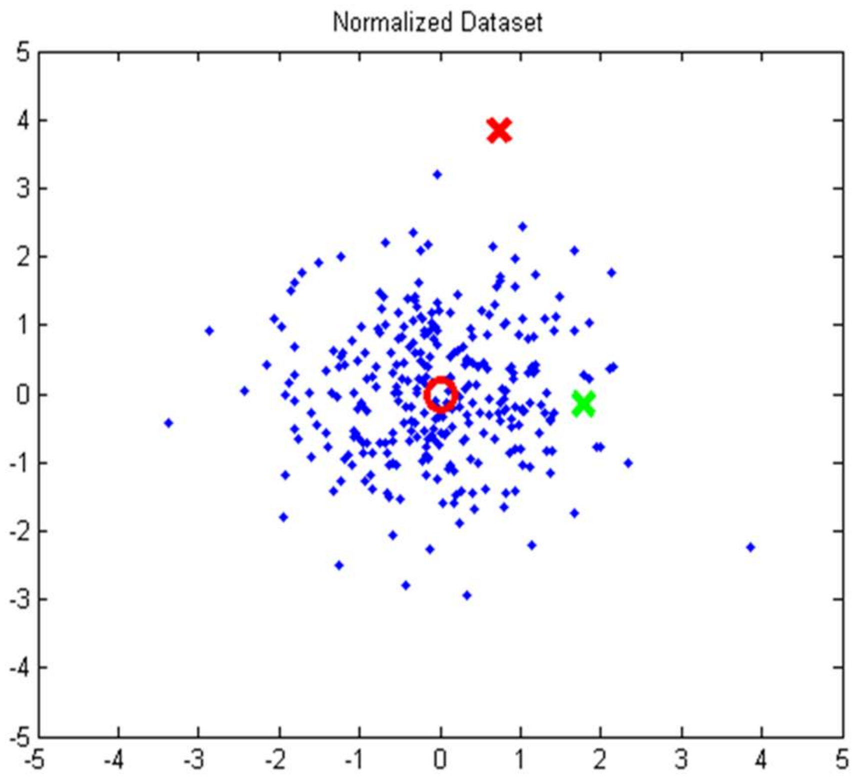
— The principal directions of variation are aligned with the axes.

— Normalize to have unit variance (by dividing the components by the Standard Deviation).

— Turns the data cluster into a sphere.

Result:

The green point is closer to the mean (red circle) than the red point.

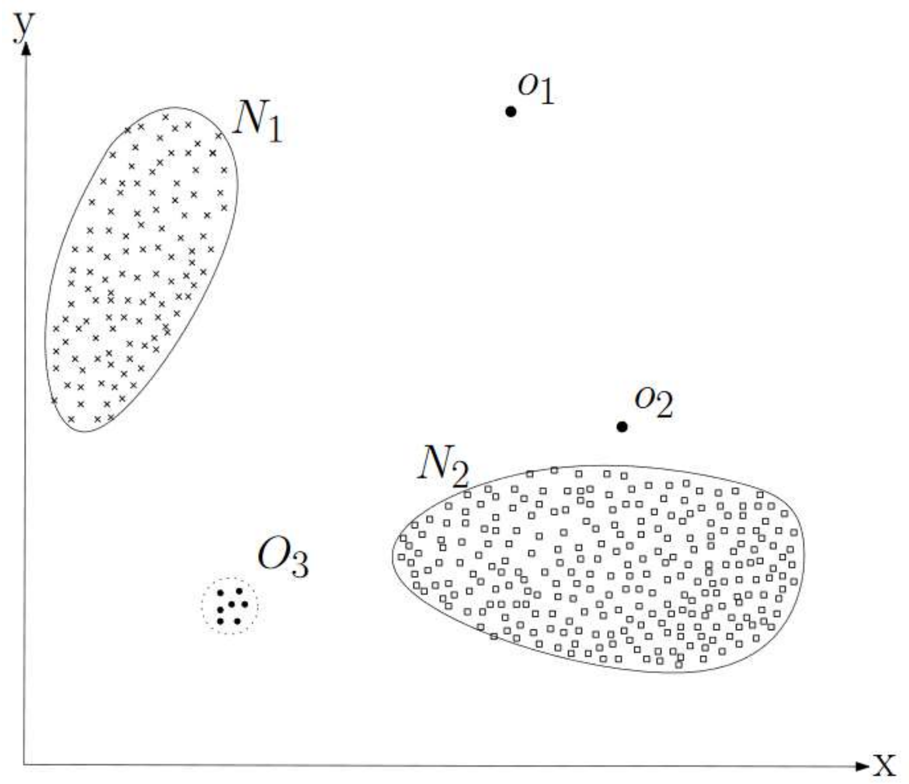


Bibliography:

Chris McCormick: “Mahalanobis Distance“ (2014), http://mccormickml.com/2014/07/22/mahalanobis-distance/

K-Means Cluster Analysis

K-means Clustering:
— Aims to partition n observations into k cluster.
— Each observation belongs to the cluster with the nearest mean.
— Has a inherent element of randomness.



Outliers:
Big distance to their cluster center.

Bibliography:

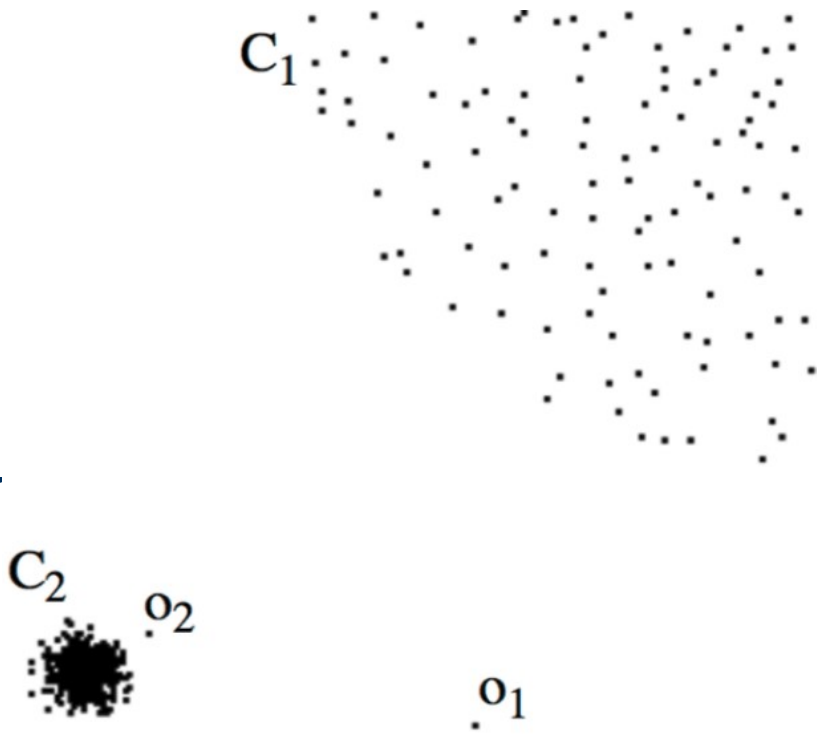
Varun Chandola: "Anomaly Detection: A Survey" (2009), http://cucis.ece.northwestern.edu/projects/DMS/publications/AnomalyDetection.pdf

Density Based Outlier Detection

Clusters: both C1 and C2.

Outliers: both o1 and o2.

Distance or clustering based methods
can detect o1 but cannot detect o2 as outlier.



Bibliography:

Jian Pei: “CMPT 741/459 Data Mining – Outlier Detection“ (Lecture notes 2015),
http://www.cs.sfu.ca/CourseCentral/741/jpei/slides/Outlier%20Detection%202.pdf

Tools for Outlier Detection

An excerpt:

- **RapidMiner**: Formerly YALE, is a data science software platform.
- **Weka**: Waikato Environment for Knowledge Analysis - Data Mining Tool.
- **Knime**: Konstanz Information Miner – Data Mining and Machine Learning Tool.
- **Spark**: Unified analytics engine for big data processing, with built-in modules for machine learning, etc.
- **R**: Programming language for statistical computing.
- **Python**: Programming language with libraries for Data Science, Machine Learning, Data Mining, etc.

Applicability of Outlier Detection

Used in a variety of domains:

— Intrusion detection,
— Fraud detection,
— Fault detection,
— System health monitoring,
— Event detection in sensor networks,
— Ecosystem disturbances,
— Preprocessing to remove anomalous data from the dataset.

Removing the anomalous data from the dataset often results in a statistically significant increase in accuracy.

Bibliography:

https://en.wikipedia.org/wiki/Anomaly_detection

Thank you

Thank you for your attention

Questions ?