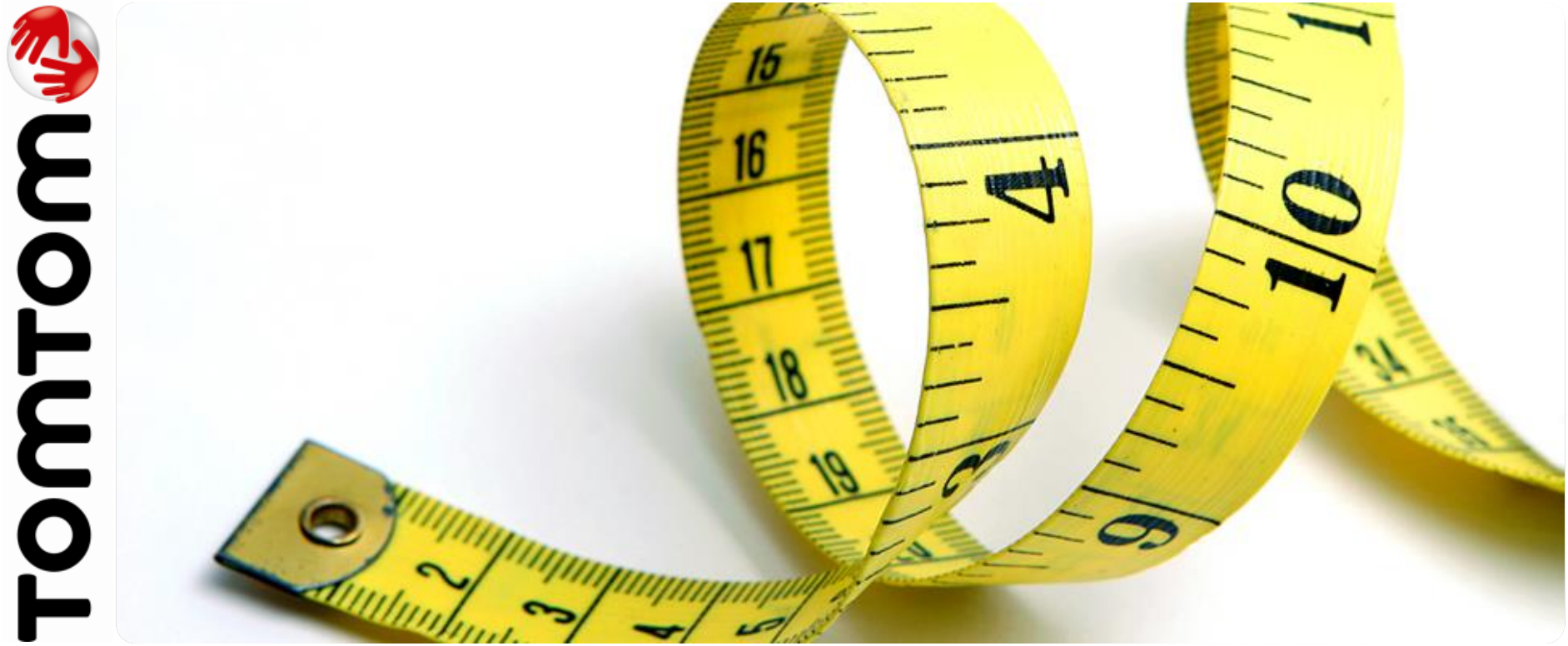


Automotive software quality metrics

Hans Aerts
VP Software Development
TomTom, BU Automotive
Feb 2, 2016

Empower Movement

From A to BE

Simplify complex technology

Innovative solutions

Make available to mass public

Connect people and data

State of the art quality

Maps

Navigation software

Connected services

Consumer

Automotive

Telematics

Licensing

©2015 TomTom International B.V.,. All rights reserved

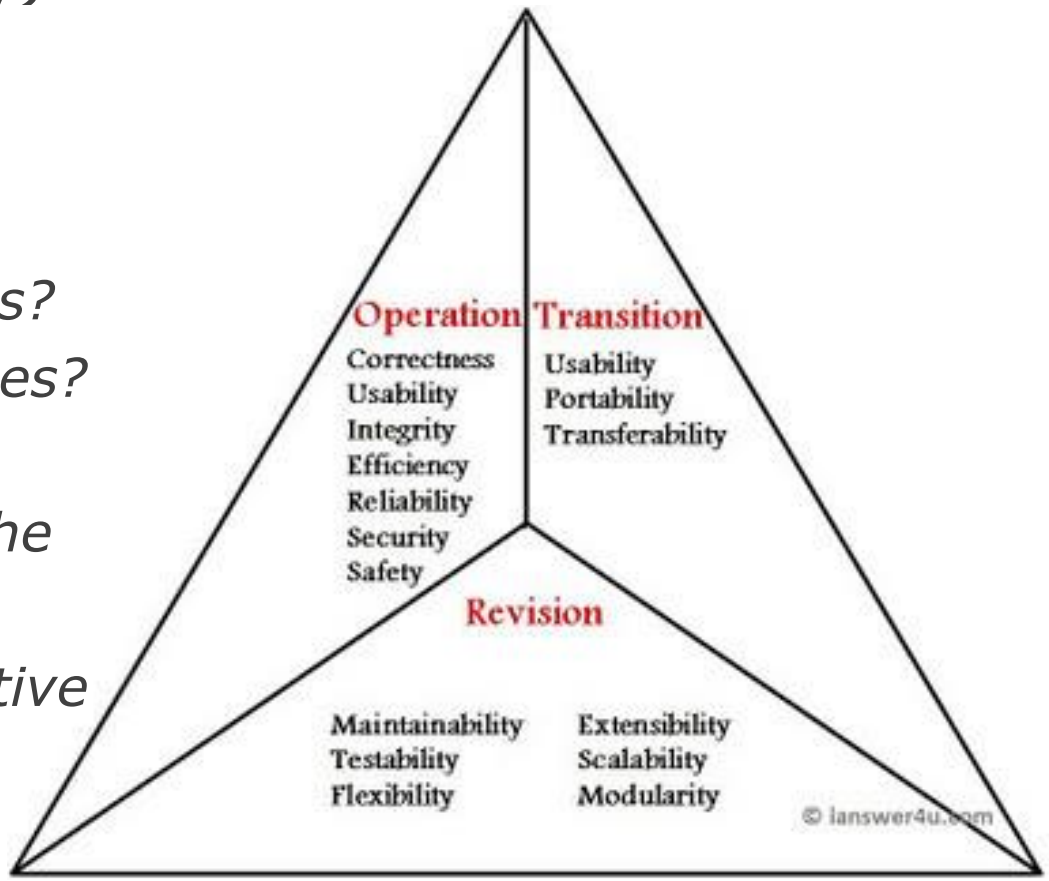
Agenda

- What is SW Quality?
- How do we measure?
- What do we measure?

TOMTOM

What is Software Quality?

- *Fit for use?*
- *Zero defects, no field returns?*
- *According to coding guidelines?*
- *100% [Unit] tested?*
- *Product quality attributes: the „-ilities" of ISO 25010?*
- *Developed conform Automotive SPICE requirements?*
- *Etcetera*



- *Software quality has many aspects, many attributes that can/should be measured.*
- *It depends on what you want to achieve, on what is important for you(r company), your customers and users*

TOMTOM

Types of metrics

In order to measure software quality, we need:

- *Product* metrics, measuring quality attributes of the product -> disadvantage: lagging indicators (however, with continuous integration in place these become also leading)
- *Process* metrics, measuring quality attributes of the process used to develop the product -> typically more leading indicators

We use metrics for:

- *Tactical* application: product development / project management.

With measurement, project leaders have information that leads to early insight of potential problems and that can support decisions. Without measurement, project leaders do not have that insight and as a result they make decisions to react to events and problems rather than to curtail these problems before they surface. Without measurement, architects lack the information that they need for reconsidering architectural design decisions and for checking with the engineers whether the implementation is in line with the intended architecture.

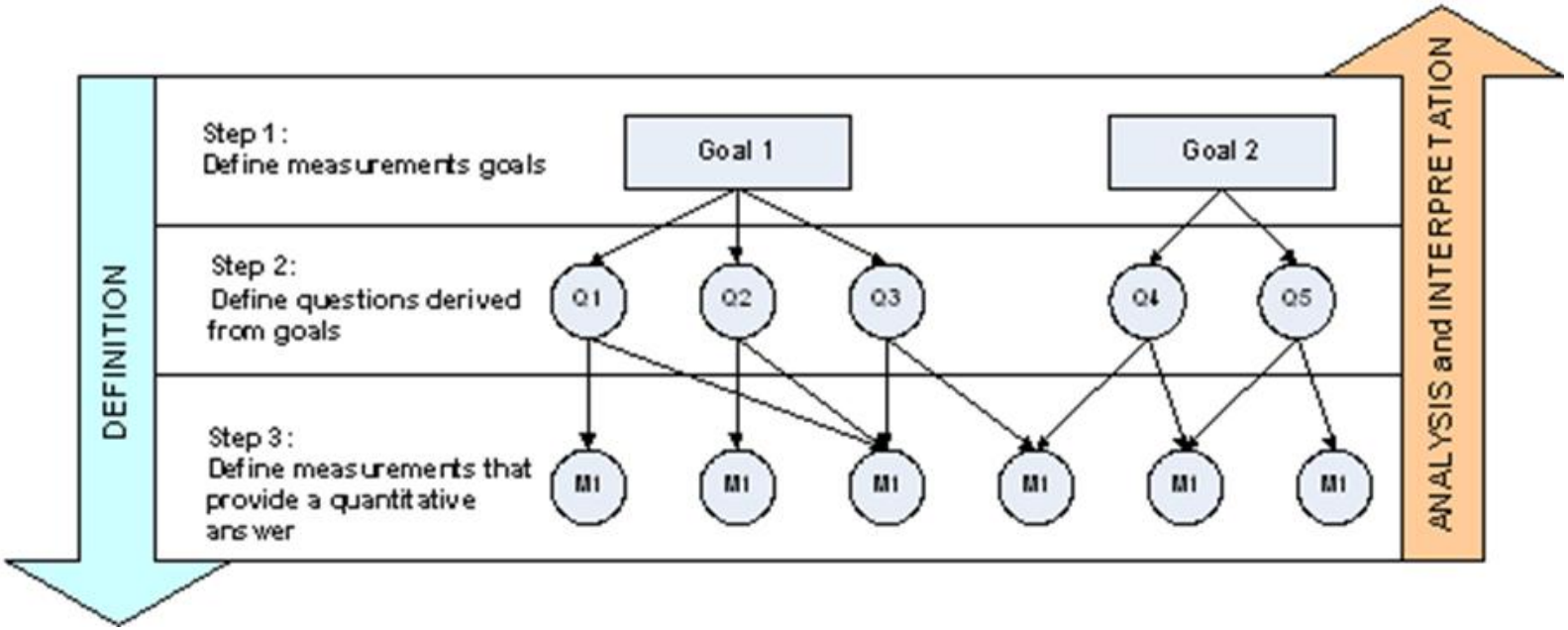
- *Strategic* application: process improvement.

When we consolidate all data in a process database, current practices can be evaluated and opportunities for process improvement will become visible.

TOMTOM

How do we measure?

Goal - Question - Metric approach



Basic measures**:** raw data that is gathered from the various sources is referred to as basic measures. Counting the 'number of lines of code' or 'number of software defects' are examples of basic measures.

Derived measures**:** new measures calculated from the basic measures like 'defect density' i.e. number of software defects per 1000 lines of code.

Automate measurements as much as possible

TOMTOM

Example of a KPI definition

1 KPI definitions

1.1 Coherent action 1: Adopt conventional system integrator best practices

1.1.1 On time delivery

Goal:	On-time delivery
Question:	How many days does a project deviate from its latest agreed upon delivery date?
Metric:	B1: Original delivery date B2: Latest agreed upon delivery date B3: Currently expected delivery date B4: Lead-time of the project (delivery phase) D1: (B3-B2)/B4 * 100% D2: (B3-B1)/B4 * 100% **D3: Σ D1/#projects measured** **D4: Σ D2/#projects measured**
Unit:	%
Frequency:	on a monthly basis
Target:	<5% for D4
Procedure:	The original delivery date (TT2) is taken from the business case at TT4. The latest agreed upon delivery date is the date approved by the MT in the project review meeting. The currently expected delivery date is the forecast of the PjM (e.g. based on burn down extrapolation, risk and dependency evaluation, etc.). The difference between B1/B2/B3 will be a number of days. The lead-time of the project (or delivery phase) is the number of days of development (TT4-TT2) for that delivery. Projects are included in this measurement after they have passed the TT4 milestone until they have passed the TT2 milestone.

TOMTOM

What do we measure? (related to software quality)

- Product performance metrics
- Product defect metrics
- Code quality metrics
- Process quality metrics

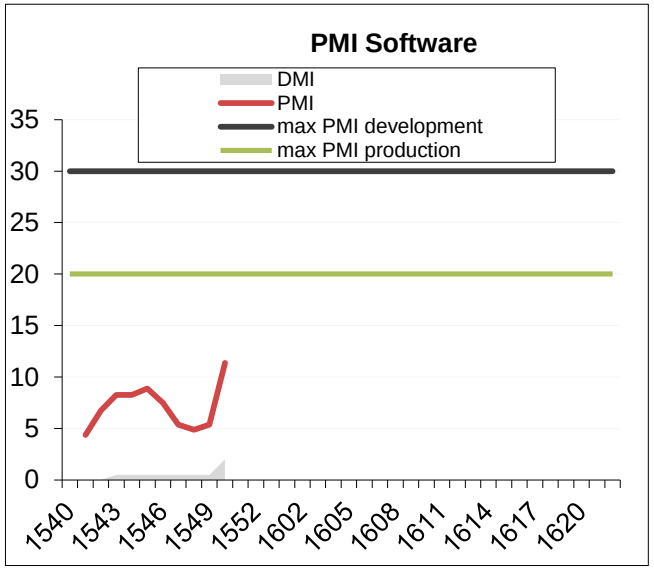
TOMTOM

Product Performance metrics

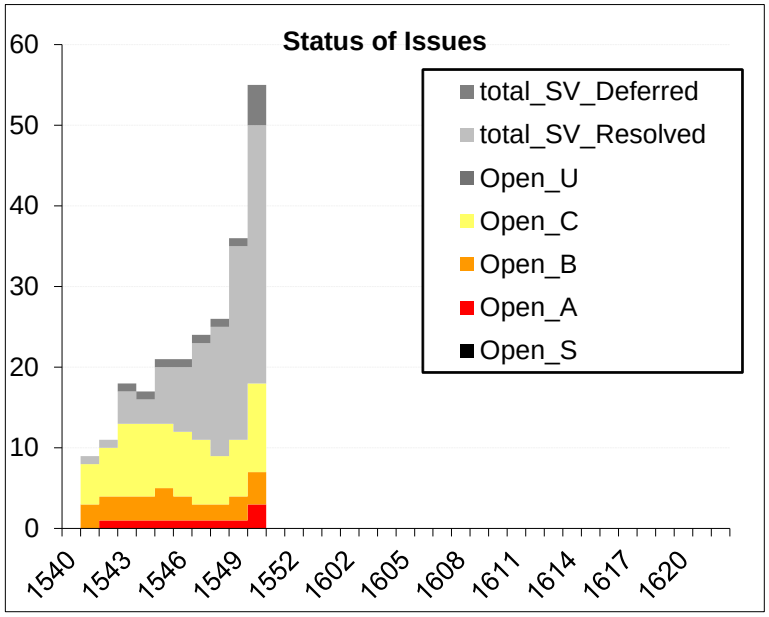
KPI	Details	Key values
Start up	Startup time	Start of NavKit/MapViewer process launching … Map visible to user/Navigation ready for use
Suspend/Resume	Suspend time	Suspend command … Processes have been suspended
	Resume time	Resume command … Navigation ready for use
Map rendering FPS	2D	
	3D	
Map 'white' tiles	Zoom out 2D	Zoom out command … No more white tiles
	Zoom in 2D	Zoom in command .. No more white tiles
	Zoom out 3D	Zoom out command … No more white tiles
	Zoom in 3D	Zoom in command .. No more white tiles
	Switch 2D -> 3D	Switch to 3D command … No more white tiles
	Switch 3D -> 2D	Switch to 2D command … No more white tiles
Route planning	Fast	Start route planning of fastest route … Route calculated
	Economic	Start route planning of economic route … Route calculated
	Shortest	Start route planning of shortest route … Route calculated
	Fast alternative	Start route planning of 1st alternative of fastest route … Route calculated
	Deviation	Deviate on current route (fastest) … Route recalculated
CPU load	Idle (2D)	% of CPU used by NavKit/MapViewer with no route planned
	Idle (3D)	% of CPU used by NavKit/MapViewer with no route planned
	Planning route	% of CPU used by NavKit while planning route
	Demo (2D)	% of CPU used by NavKit/MapViewer when running route demo
	Demo (3D)	% of CPU used by NavKit/MapViewer when running route demo
Memory usage	Idle (2D)	% of memory used by NavKit/MapViewer with no route planned
	Idle (3D)	% of memory used by NavKit/MapViewer with no route planned
	Planning route	% of memory used by NavKit while planning route
	Demo (2D)	% of memory used by NavKit/MapViewer when running route demo
	Demo (3D)	% of memory used by NavKit/MapViewer when running route demo
	Duration	% of increase over time

Product Defect metrics

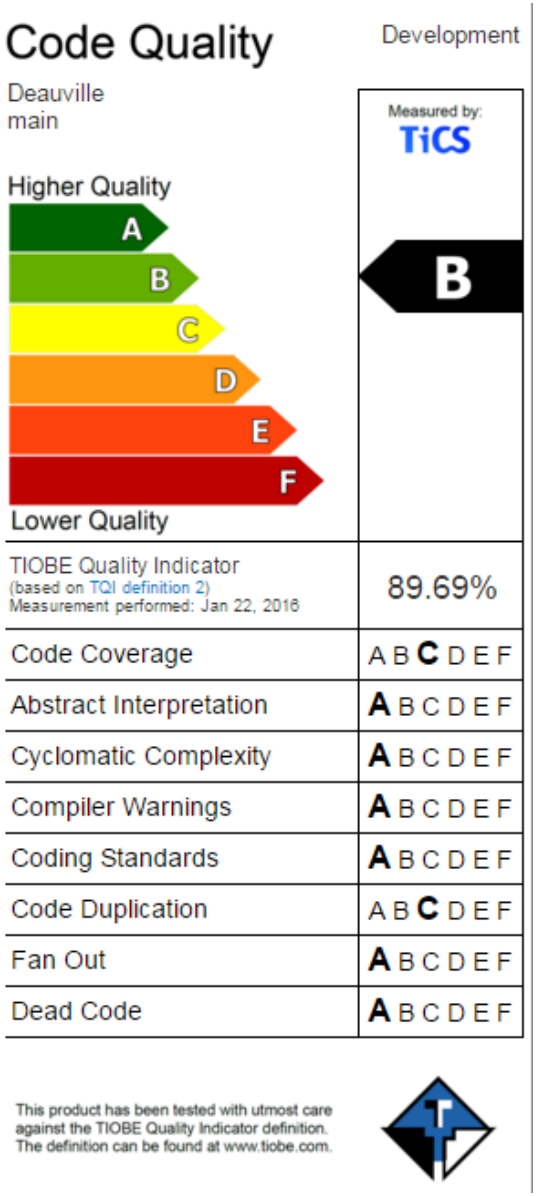
PMI Calculation						
Severity/Priority code		S	A	B	C	U
Problem status		4	2	1	0.5	0
New_Problem	1.00	4.0	2.0	1.00	0.500	0.0
Cause_Known	0.75	3.0	1.5	0.75	0.375	0.0
Solution_Known	0.50	2.0	1.0	0.50	0.250	0.0
Solution_Implemented	0.25	1.0	0.5	0.25	0.125	0.0
Solution_Verified	0.00	0.0	0.0	0.00	0.000	0.0



- All defects are recorded in Jira
- All defects have a severity and a status
- The 'weight' of a defect is determined by severity / status (see table)
- Product Maturity Index (PMI) is the sum of all weights of all defects
- PMI is tracked over time
- If PMI approaches the pre-defined threshold values, then measures can be taken (e.g., stop developing features, first solve defects)



Code Quality metrics



- **Code Coverage**

path's covered during [unit] tests

- **Abstract Interpretation**

static code checking; „coverity"

- **Code Complexity**

nr of path's through the code (*mcCabe's cyclomatic complexity nr)*

- **Compiler Warnings**

nr of compiler warnings in the code

- **Coding Standard**

deviations from the coding standard

- **Code duplication**

amount of copy-pasted code

- **Fan out**

number of includes

- **Dead code**

amount of unreachable code

TOMTOM

Process Quality metrics

All our processes (should) have KPIs to measure effectiveness and efficiency of the process. Examples include KPIs like:

- On-time delivery (current forecast versus agreed upon with customer)
- Gross margin
- Attrition rate
- #escalations with internal suppliers
- Customer returns

but also include the following leading indicators related to software quality:

- User story test coverage
- User story cycle time
- % of user stories with acceptance criteria specified

Thank you

Any questions?

TOMTOM