5WCSQ: The 5th World Congress for Software Quality

Quality Improvement by the Real-Time Detection of the Problems

--- *DevCast* (Development Forecast)
for the Failure Project Prevention ---

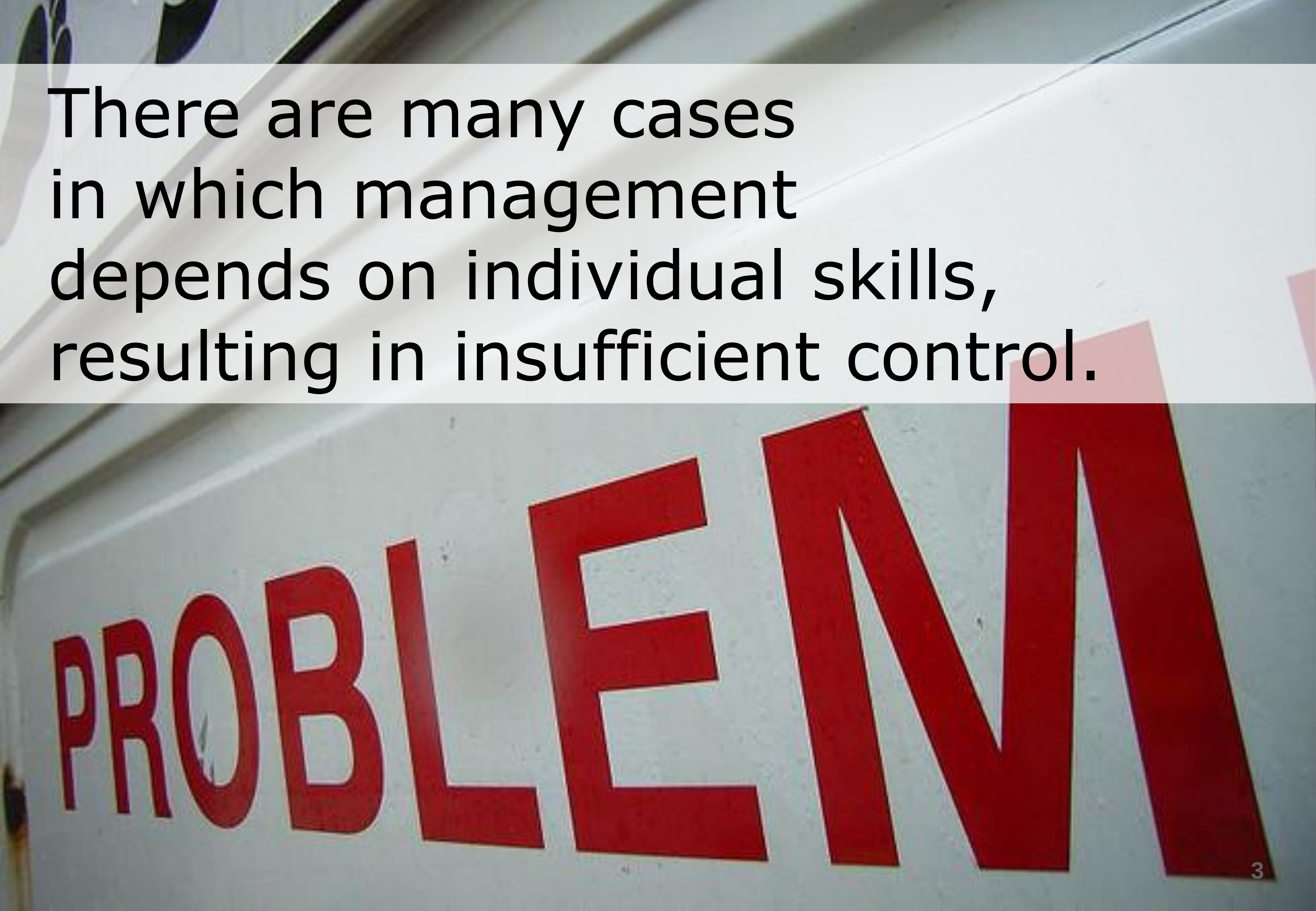
2nd November, 2011

Takanori Suzuki
Acroquest Technology Co., Ltd.

Introduction – SUZUKI Takanori

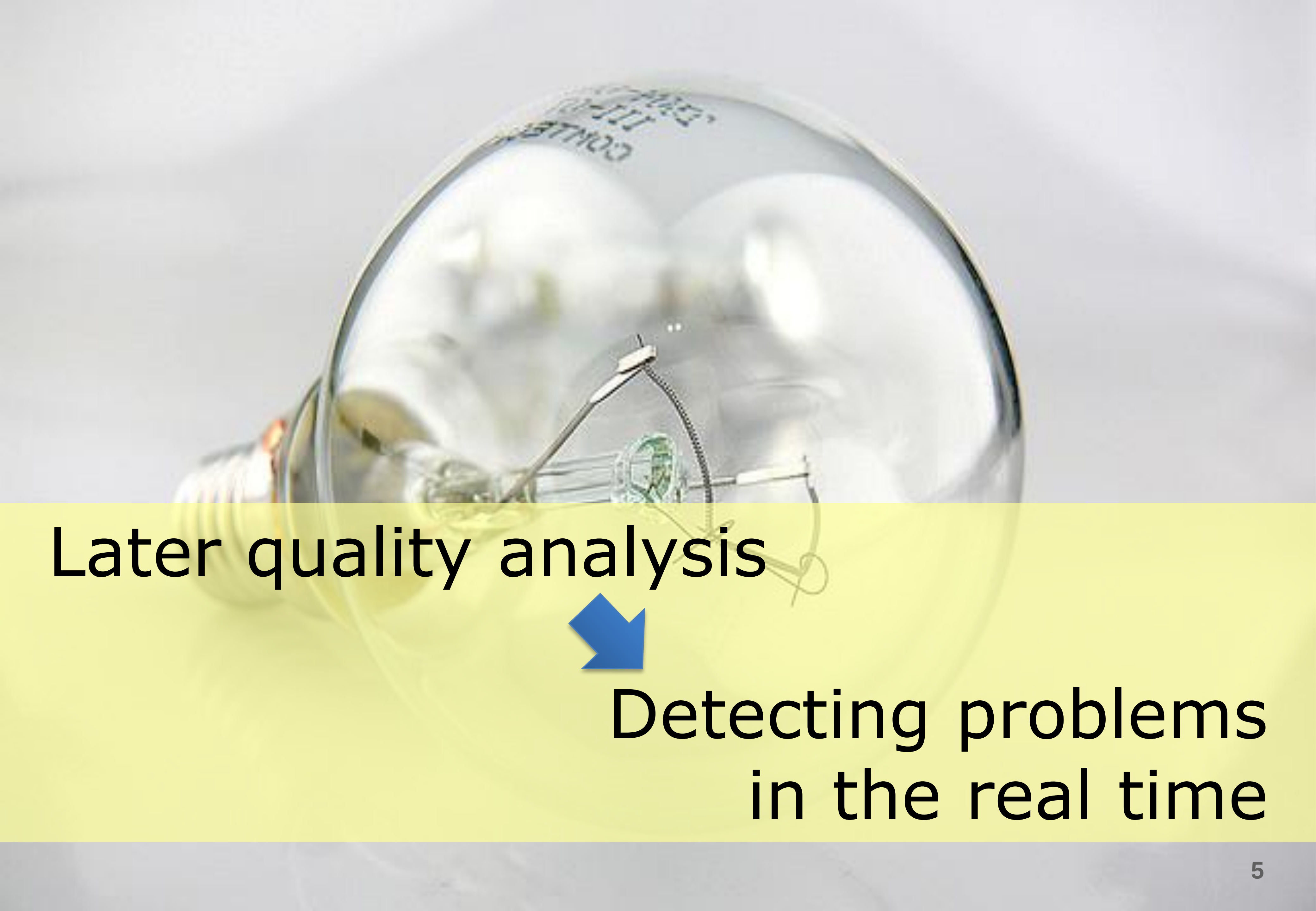
- From
  - Acroquest Technology Co., Ltd.
（http://www.acroquest.co.jp）
  - Technical Consultant
- Specialties
  - SEPG（Software Engineering Process Group）
    - CMMI-based process improvement,
      Quality Assurance activities
    - Development of process-related tools
  - System Development
    - Led several develop projects for frameworks and web systems as the project manager and architect

There are many cases
in which management
depends on individual skills,
resulting in insufficient control.



To discover
whether quality is fully assured

Defect detection density ?
Degree of defect convergence ?



Later quality analysis

Detecting problems in the real time

Agenda

1. Reality of Development Projects
2. Invisible Quality Problems
3. Forecasting Methods of Project Success
4. Case Studies
5. Conclusion

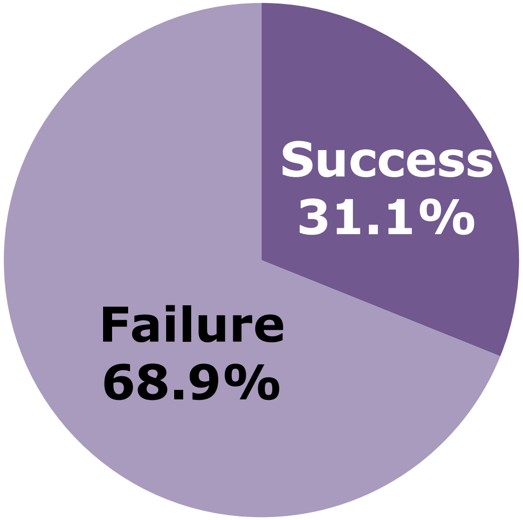
1. Reality of Development Projects

Success rate of projects : 31.1%

||

About 70% of projects end in failure!?

Higher?



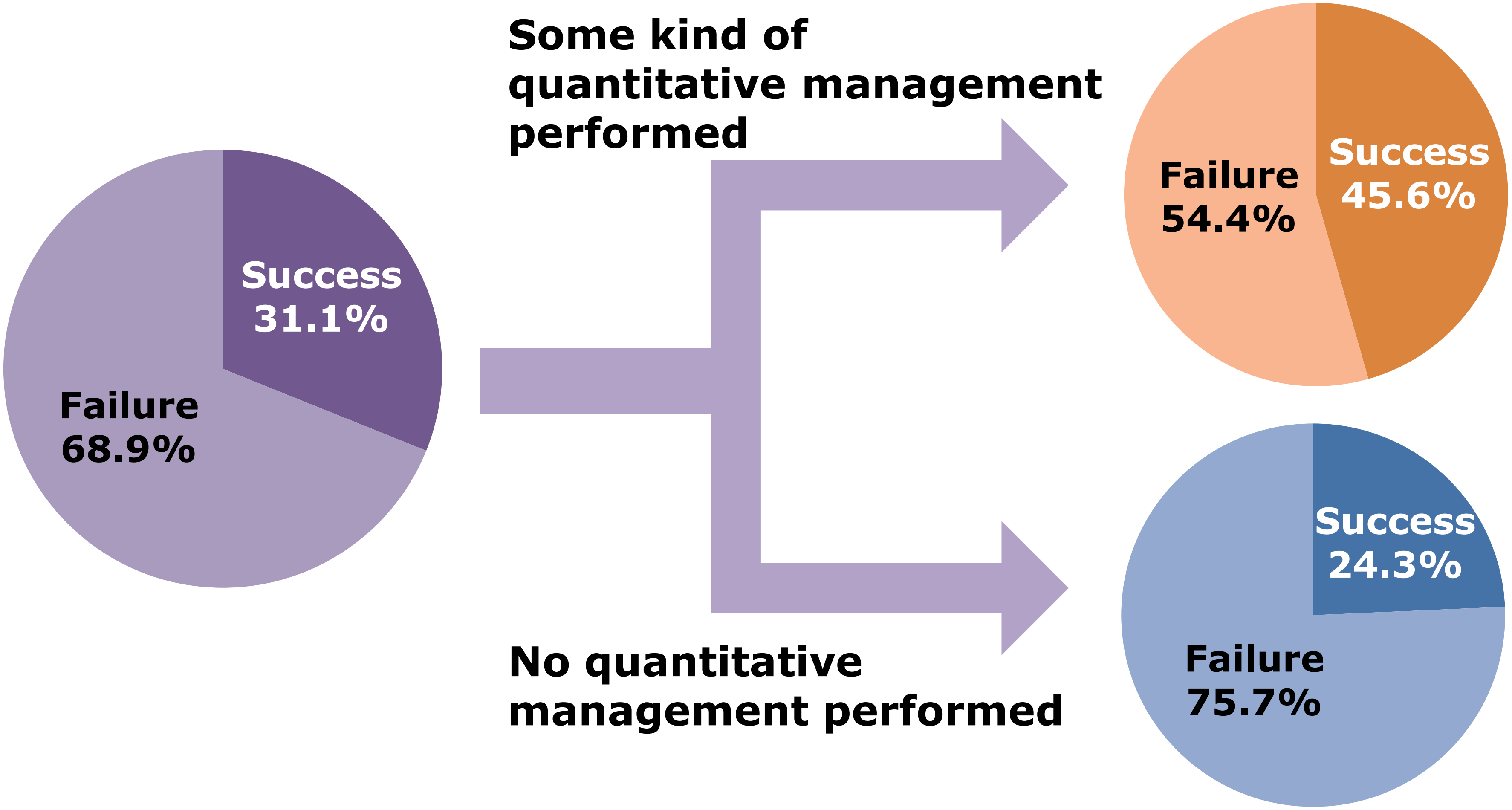
Lower?

Reference：NIKKEI Computer (No.2008-12-1)

Are the profits earned from 30% of successful projects totally consumed by the costs of the remaining 70% of failed projects?

1-1. Advantage of Quantitative Management（Overall Total）

Quantitative management is able to double success rate

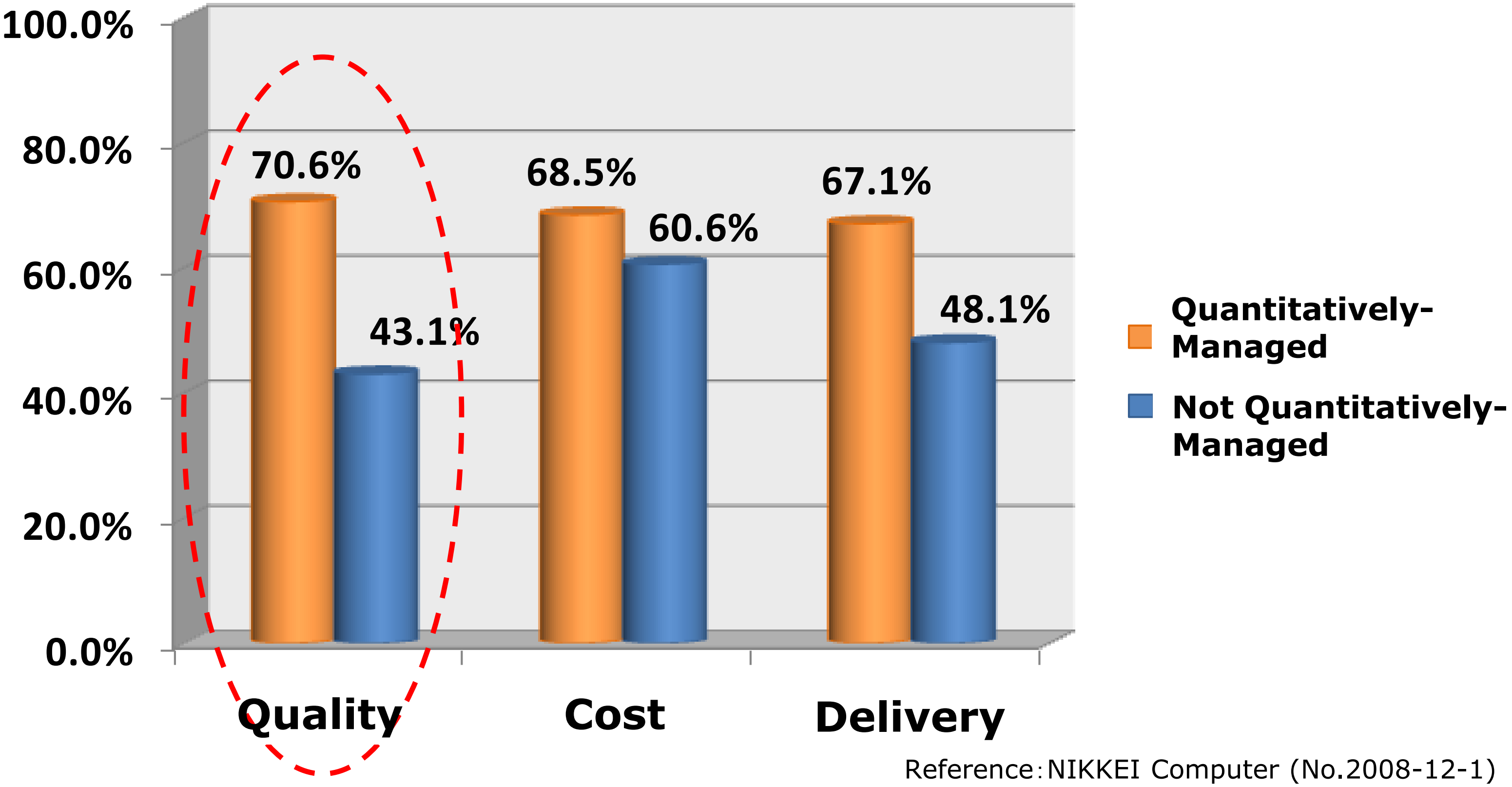


Reference：NIKKEI Computer (No.2008-12-1)

1-2. Advantage of Quantitative Management（in QCD）

Quantitative management is very effective, especially when it comes to quality

Success rates for projects quantitatively managed and those that were not



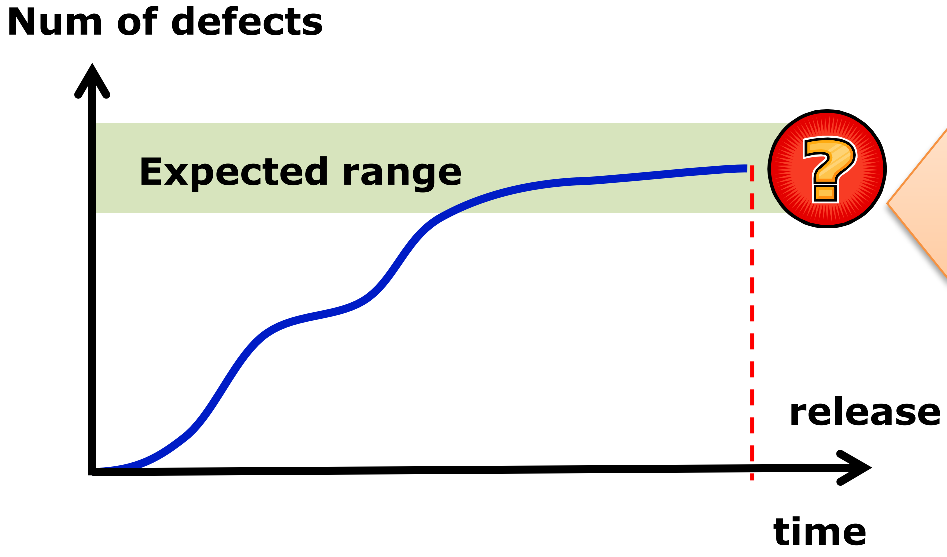
Reference：NIKKEI Computer (No.2008-12-1)

2. Invisible Quality Problems

How do we know whether quality can be fully assured?

- Bug detected.
- Bug detection target satisfied.
- The number of bugs is within the expected range.

- Is the goal satisfied with only superficial errors and without essential problems?
- If the test-operator is insufficiently skilled, how do we know that problems have not been overlooked?



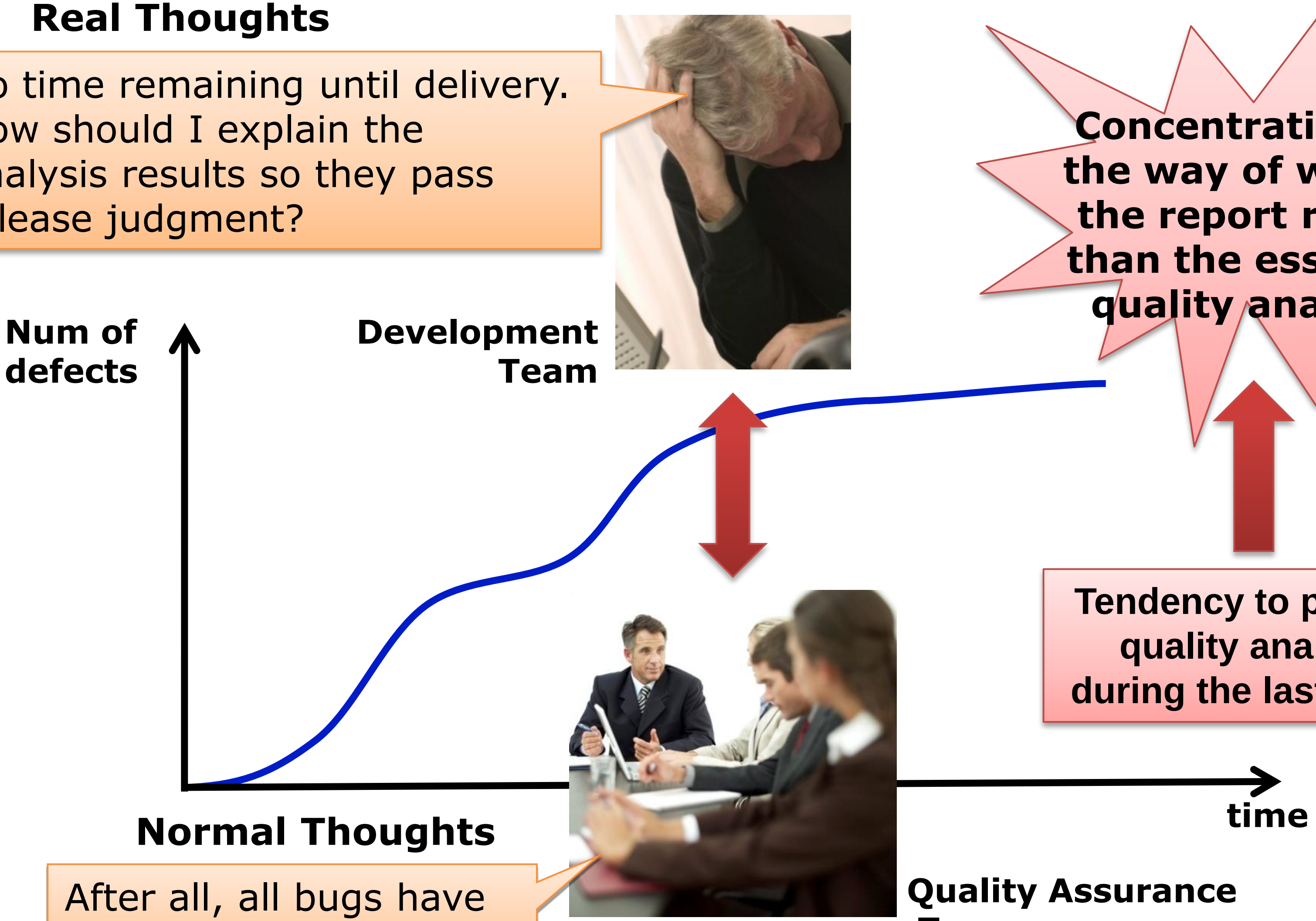
[Judgment factors for test completion]
- Quality of product
- Quality of test cases
- Severity of detected errors
- Skills of test operators

Difficult
to analyze/judge
these factors

2-1. Is it Really Quantitative?

Real Thoughts

No time remaining until delivery. How should I explain the analysis results so they pass release judgment?



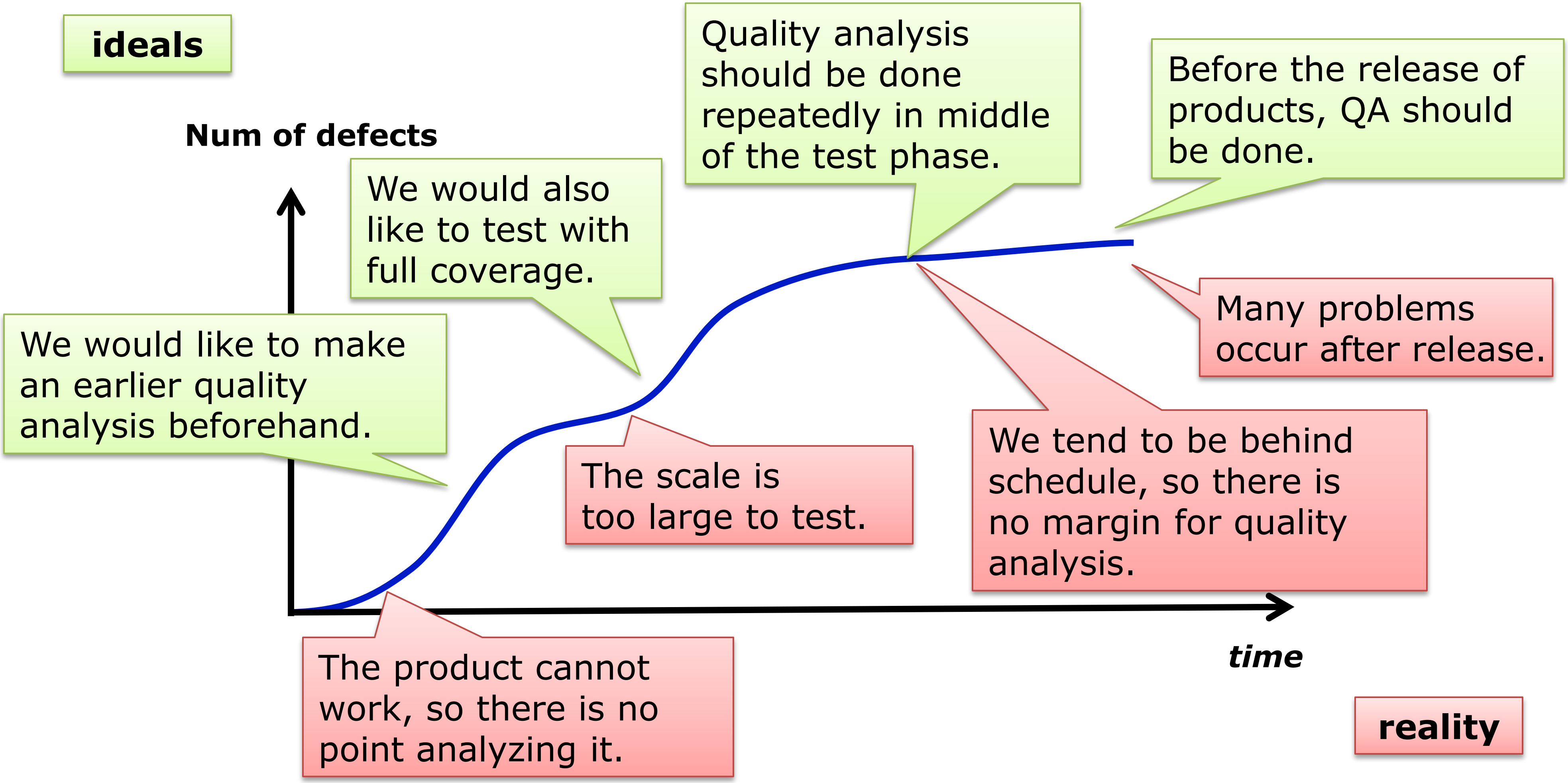
Concentrating on the way of writing the report rather than the essential quality analysis

Tendency to perform quality analysis during the last phase

Normal Thoughts

After all, all bugs have been resolved?

2-2. Unfilled Gaps between Ideals and Reality

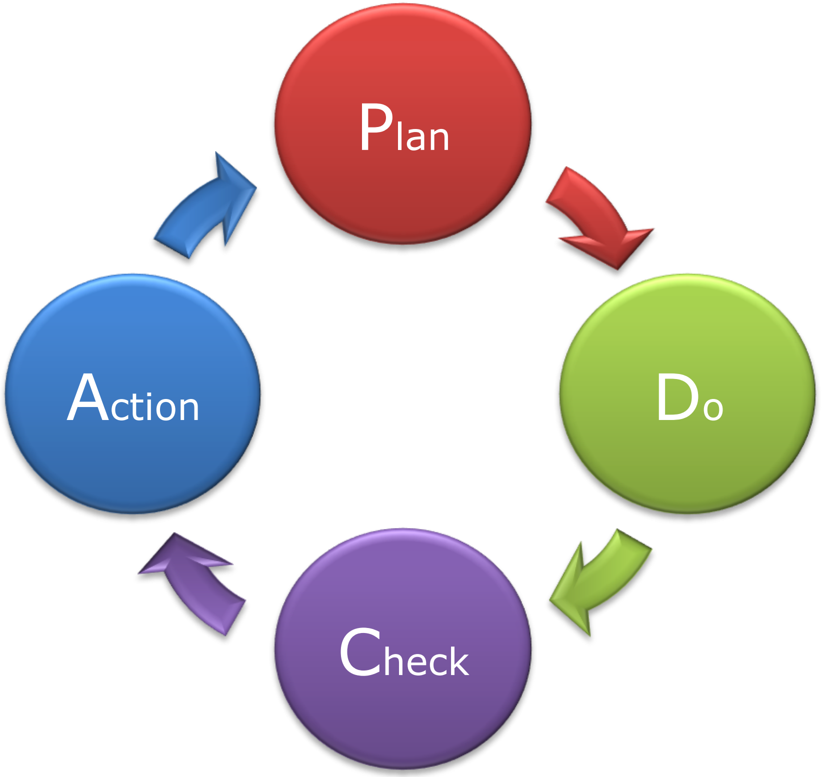

ideals
Num of defects
We would also like to test with full coverage.
Quality analysis should be done repeatedly in middle of the test phase.
Before the release of products, QA should be done.
We would like to make an earlier quality analysis beforehand.
Many problems occur after release.
The scale is too large to test.
We tend to be behind schedule, so there is no margin for quality analysis.
The product cannot work, so there is no point analyzing it.
time
reality

“Quality” is frequently sacrificed for “Cost” and “Delivery”

3. Methods of Forecasting Project Success

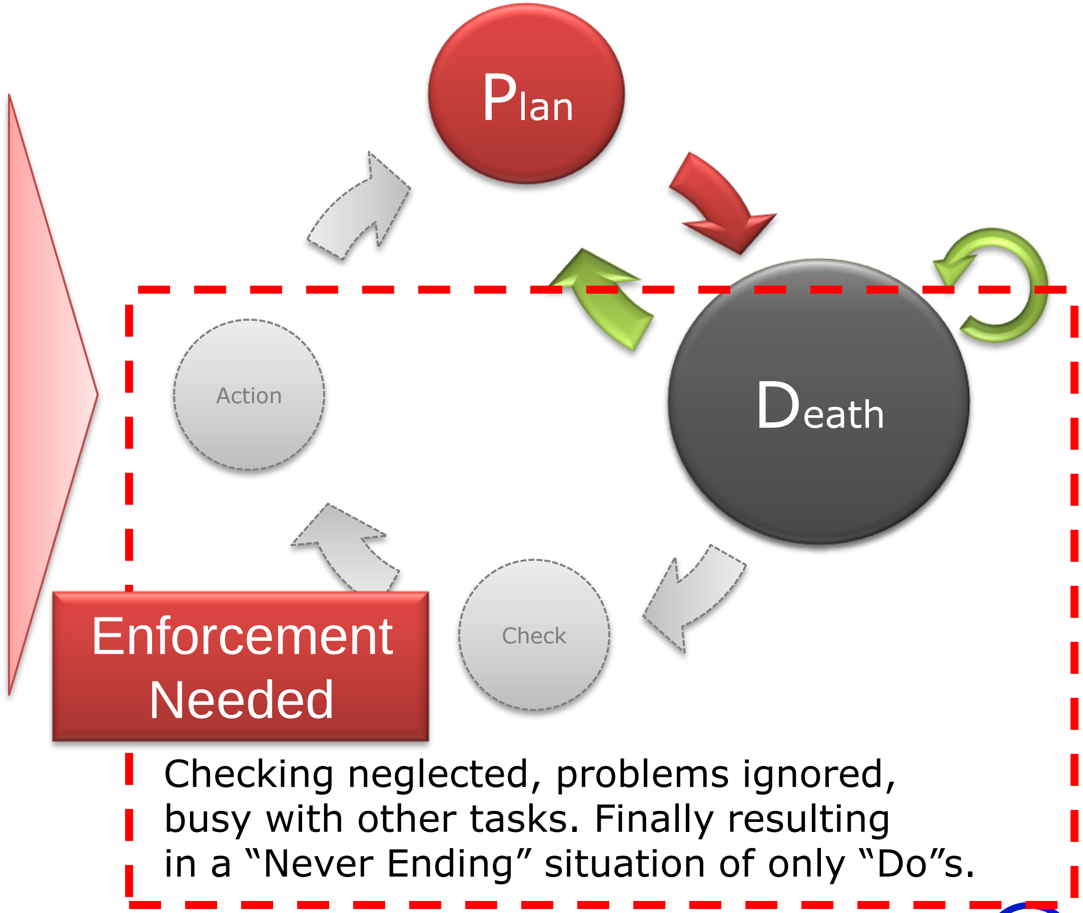
Breaking away from disfunctional PDCA cycles

Ideal PDCA



Based on the plan, working with frequent correction will lead to continual improvement

Real PDCA

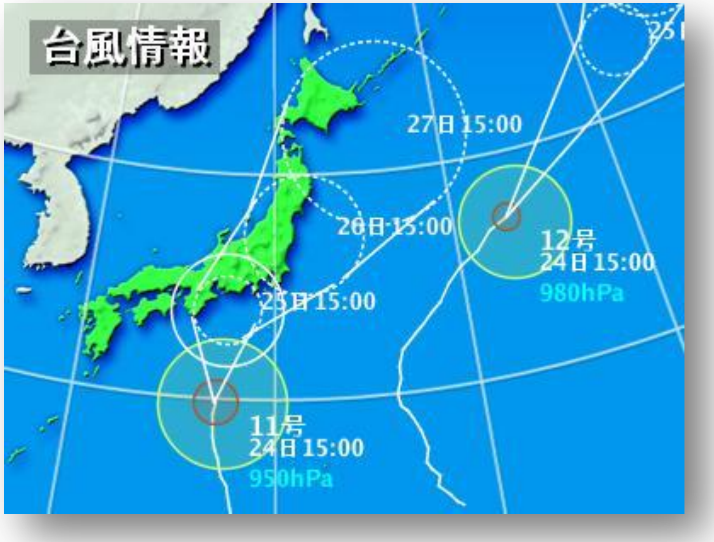


Checking neglected, problems ignored, busy with other tasks. Finally resulting in a “Never Ending” situation of only “Do”s.

3-1. Can the Future of Projects be Foreseen?

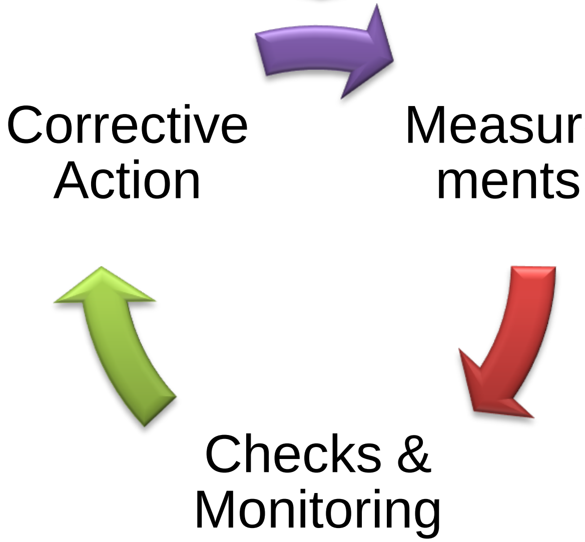
Weather Forecast

Weather is a nonlinear phenomenon that is difficult to predict.
But measuring / evaluating past and present data enables the future to be predicted (Weather Forecast).



Software Development

Software development is also difficult to predict the future.
But in the case of development situations, forecasting the future can be aimed at through the evaluation of past and present events.

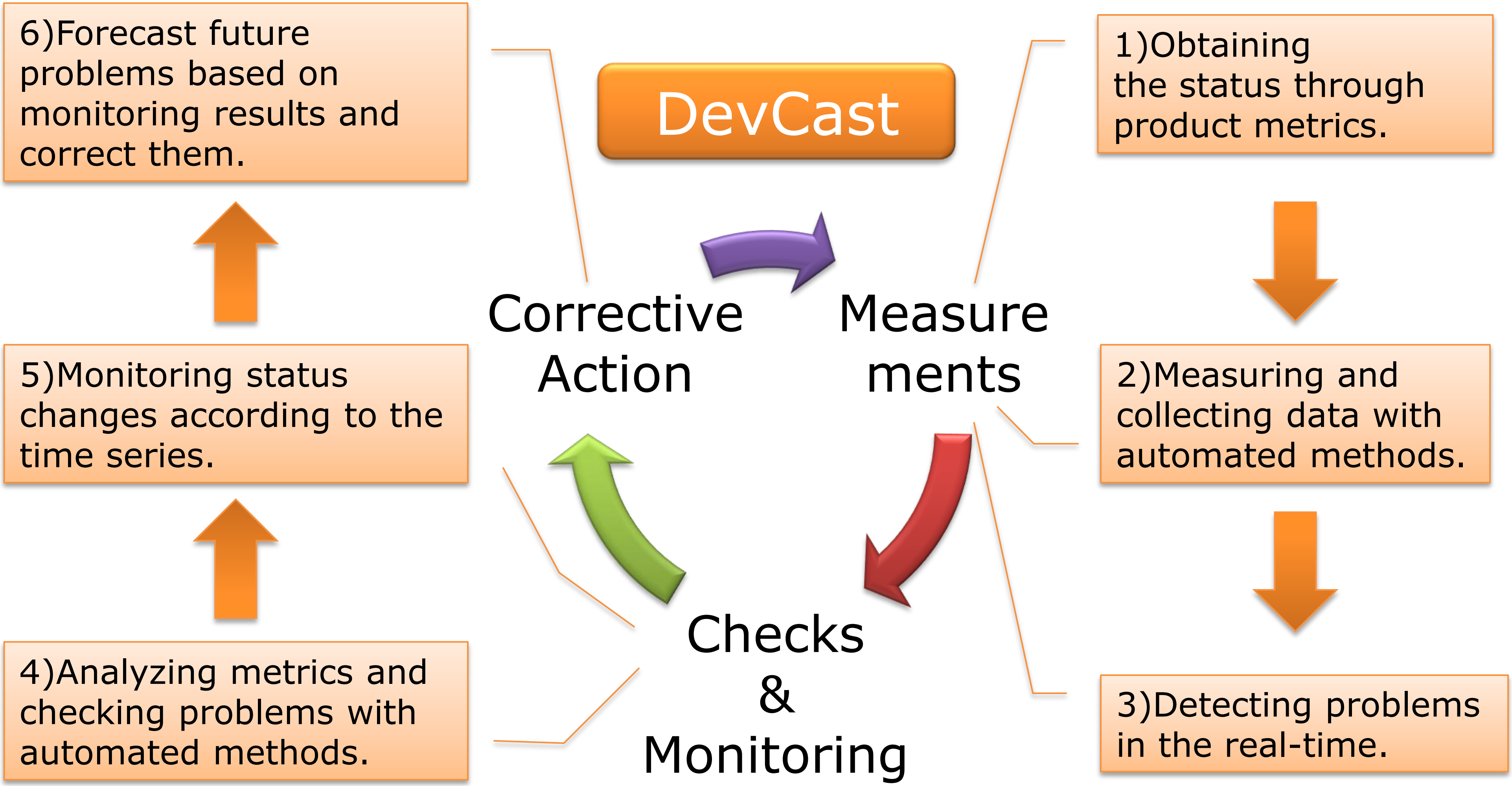


Corrective Action

Measure ments

Checks & Monitoring

3−2. “Development ForeCast” Approach to Preventing Failure



DevCast

1)Obtaining
the status through
product metrics.

2)Measuring and
collecting data with
automated methods.

3)Detecting problems
in the real-time.

4)Analyzing metrics and
checking problems with
automated methods.

5)Monitoring status
changes according to the
time series.

6)Forecast future
problems based on
monitoring results and
correct them.

Measure
ments

Checks
&
Monitoring

Corrective
Action

3−3. “Development ForeCast” Main Features

Point (1)

It is not a plan-based project management, with action taken based on the current status. It is the Automated Project Monitoring Approach.

Point (2)

By using tools that don’t rely on the effort of managers and developers, data from facts (products) are automatically corrected and analyzed.

Point (3)

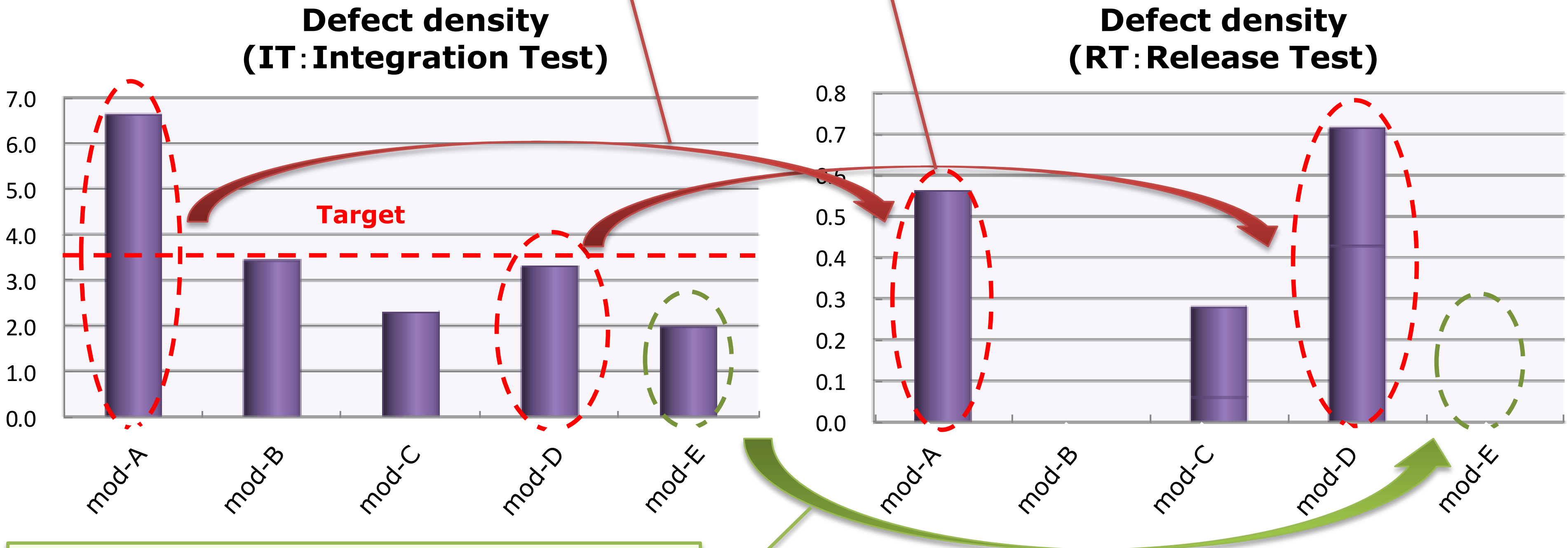
In the real-time feed forward of risks, possibly leading to project failure is reduced due to early problem-detection and correction.

4. Case Studies（Bugs Appearing during the Testing Phase）

Difficult to judge with testing alone

Module A, with many density defects detected in both the IT and the RT, is low in quality.

mod-D achieved its target in the IT, while additional defects were detected in the RT. It can therefore be assumed that the defects were passed across from the IT.



Defect density (IT：Integration Test)

Defect density (RT：Release Test)

Module E, with less density defects detected in both the IT and the RT, can be said to be of high quality.

4-1. Multilateral Analysis of Source Code（Static Quality Evaluation）

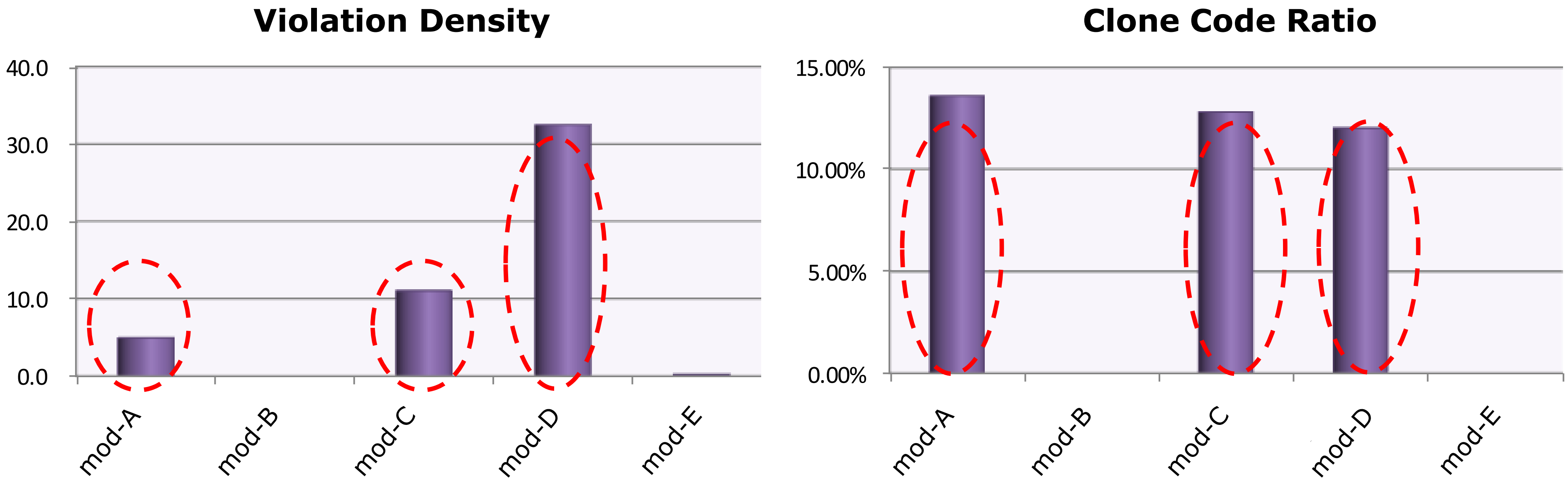
Evaluating quality conditions before the testing phase is needed

Multilateral Analysis of Source Code

No	Metrics	Tool	Description
1	Coding standard violation	Checkstyle	Check source code and count the coding standard violations.
2	Static analysis violation	FindBugs	Check source code and count the static analysis violations.
3	Cyclomatic complexity number	JavaNCSS	Count number of methods having a cyclomatic complexity (McCabe’s) greater than 30.
4	Clone code lines	CPD	Count duplicate code.

4-2. Relationship between Static Quality Evaluations and Violations

Comprehensive analysis of source code quality with static quality evaluations

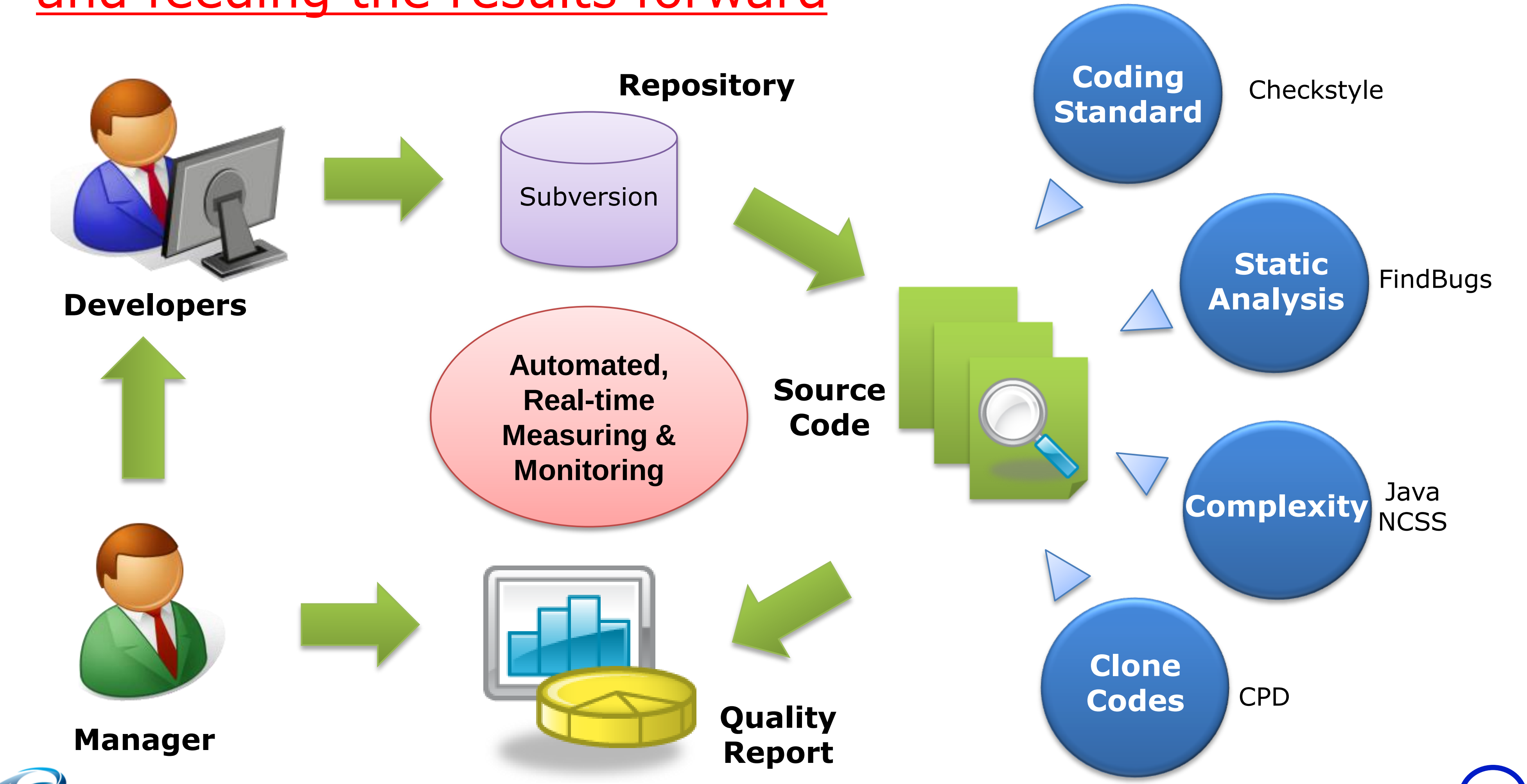


mod-A
mod-C
mod-D ▶ It matches up with the modules for which many defects were detected in the release test.

⇨ Multilateral static analyses make it possible to specify risky functions from the quality point of view before testing.

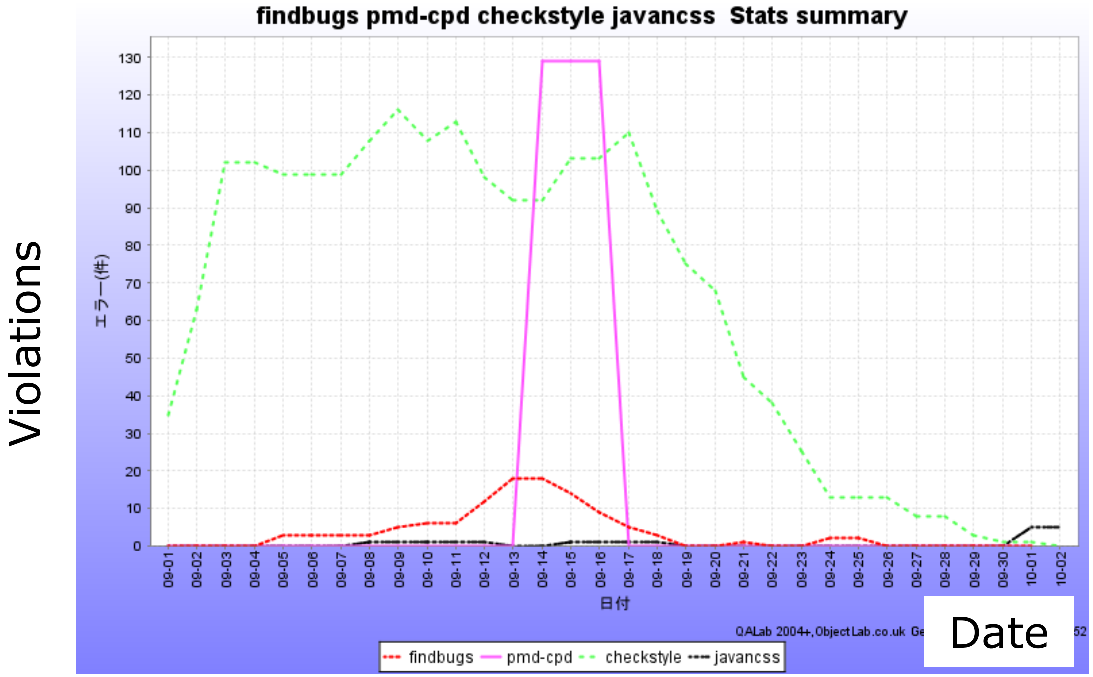
4-3. DevCast Approach

Evaluating quality level in the real-time and feeding the results forward



4-4. Quality Evaluations during the Coding Phase

Detecting and fixing violations in the real-time



- The X-axis is the date
- The Y-axis is the violation count detected by the tools
- Automatically checked and converted into a graph every day
- Modify defects in the real-time to improve quality

Be able to forecast the quality risks in modules for which many violations were detected.
Check the modules in more detail during later testing phases!

4-5. Effects of Improvements

Improved Quality

- •Multilateral and cyclopaedical quality checks are done
- •There is reduction of quality improvement costs
- •Removals of simple bugs are possible

Early Risk Specification

- •Deliverables-based risk specification are done
- •Insufficient skill of person in charge can be overlooked
- •Members will be adherence to the status of the process

Enlightenment of Developers

- •Recognition of bug patterns
- •Motivation toward solving errors

5. Conclusion

1. Current problems
   ① 70% failure rate in projects
   ② Quantitative management for quality may be insufficient due to individuals
2. Action to be taken
   ① Using “Measurements – Check & Monitoring – Corrective Action” cycle
   ② Applying this to multilateral analysis of source code to detect quality problems in the real-time
3. Merits of real-time detection
   ① Quality level raised without increasing the burden on managers and developers
   ② Risks specified early based on fact (product)-based evaluations
   ③ The awareness of developers improved with regard to product quality

Thank you