APM Center of Excellence

Andreas Reiss
Principle Consultant, Center of Excellence EMEA, *CA Technologies*

agility made possible™

ca technologies

agenda

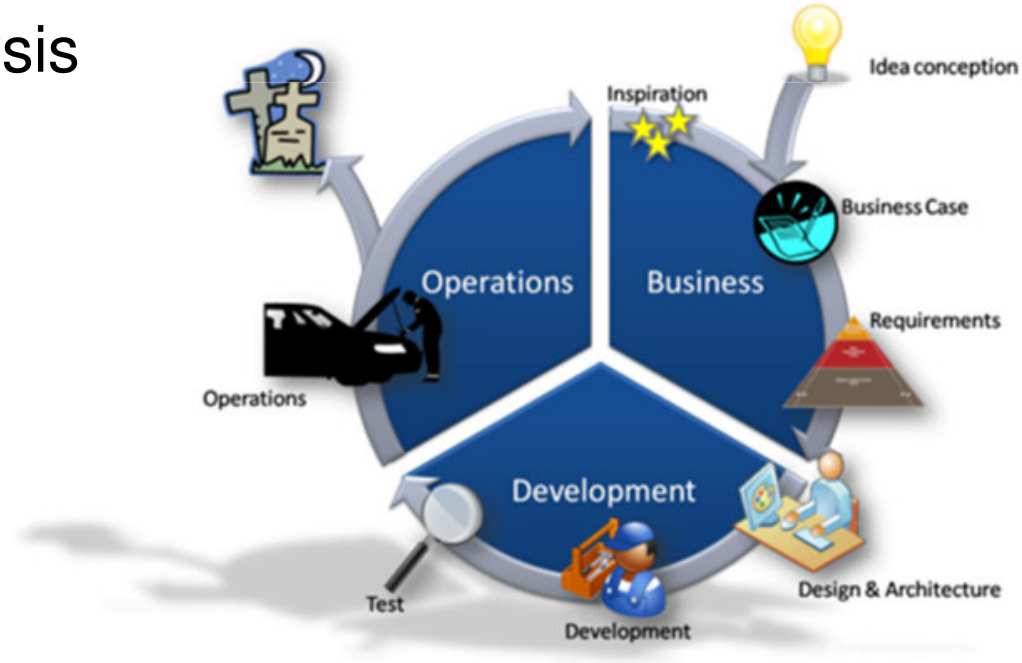
— putting the end user to the center stage

— APM in production

— extending the monitoring to pre-production

— development an analysis

— conclusion

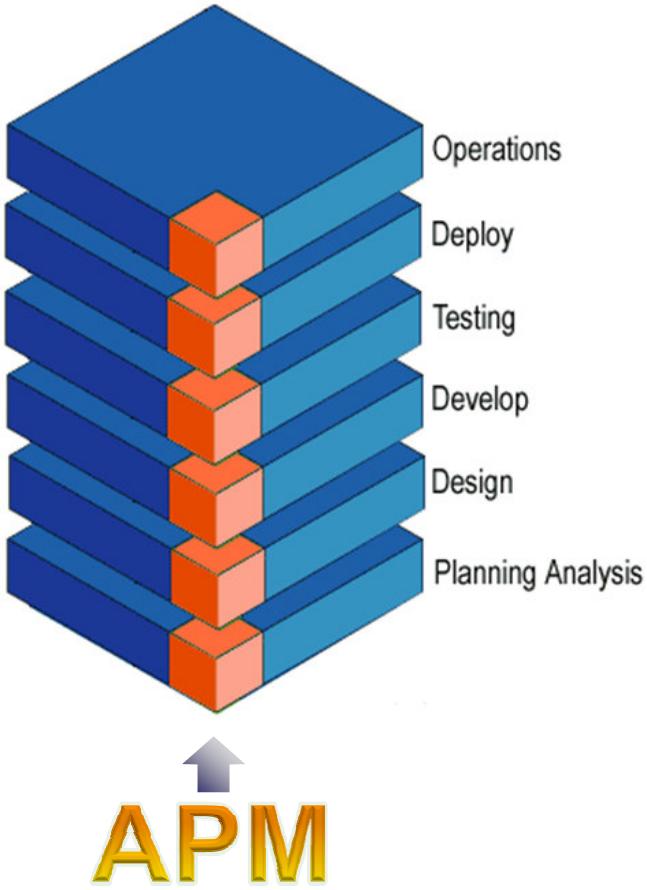


putting the end user to the center stage

the end user in the focus

Common assumptions

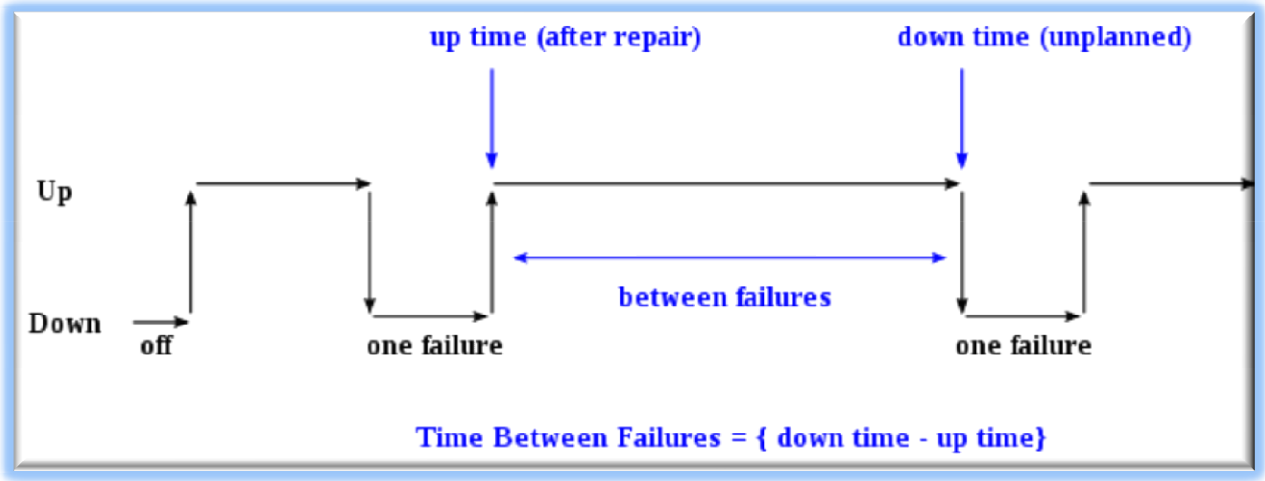
- I need to know what's wrong in case something happened. That's enough!
- I don't need a tool in Pre-Production, I anyway don't find there production problems.
- My developers have their own to tool.
- First we implement and later we will decide whether to integrate into the monitoring.



the end user in the focus – quality metrics

Software Quality metrics:

- Mean time to failure
- Defect density
- Customer problems
- Customer satisfaction



the end user in the focus – quality metrics

Application error definition (ANSI 982.2)

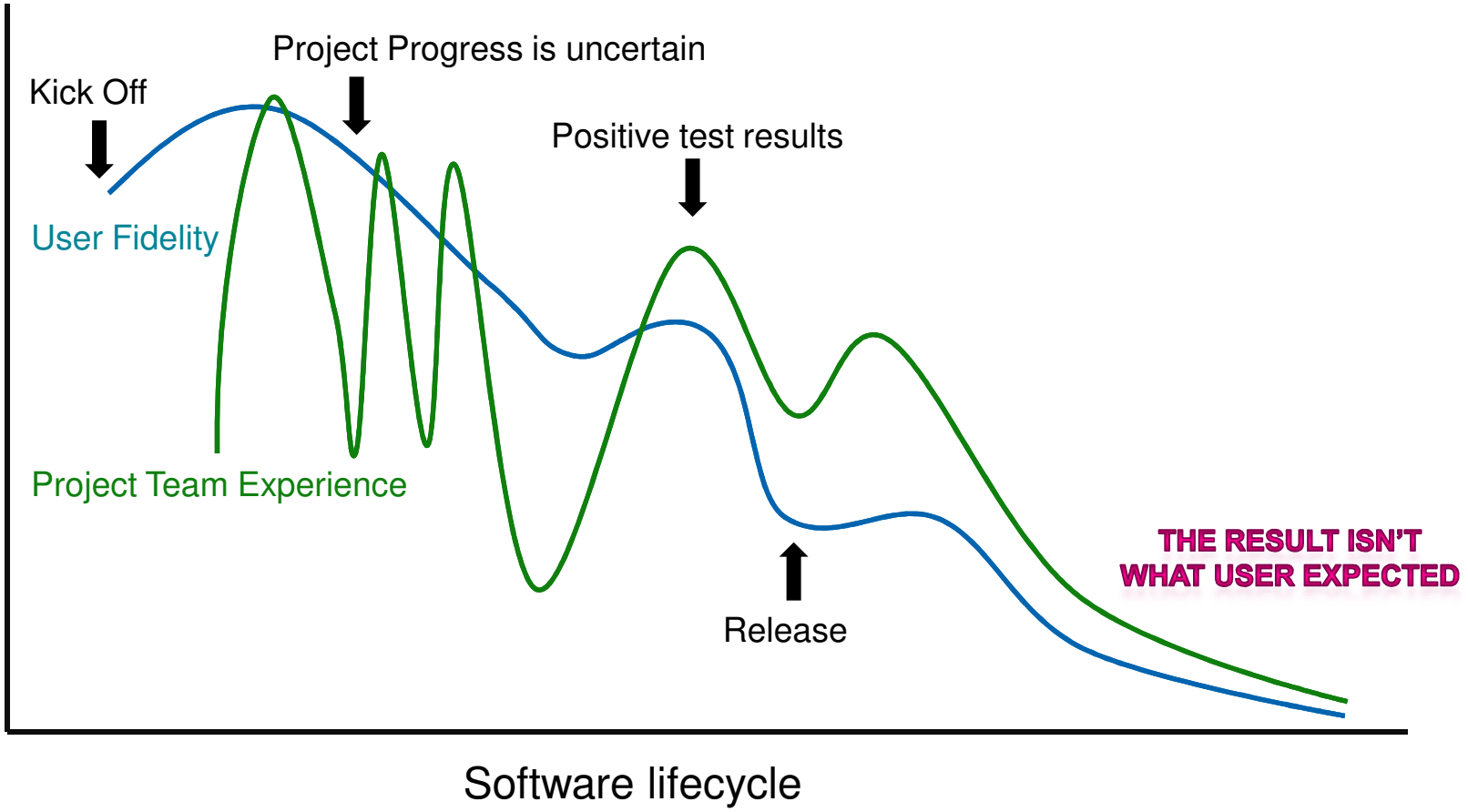
— An error is a **human mistake** that results in incorrect software.

— The resulting fault is an **accidental condition** that causes a unit of a system to fail to function as required.

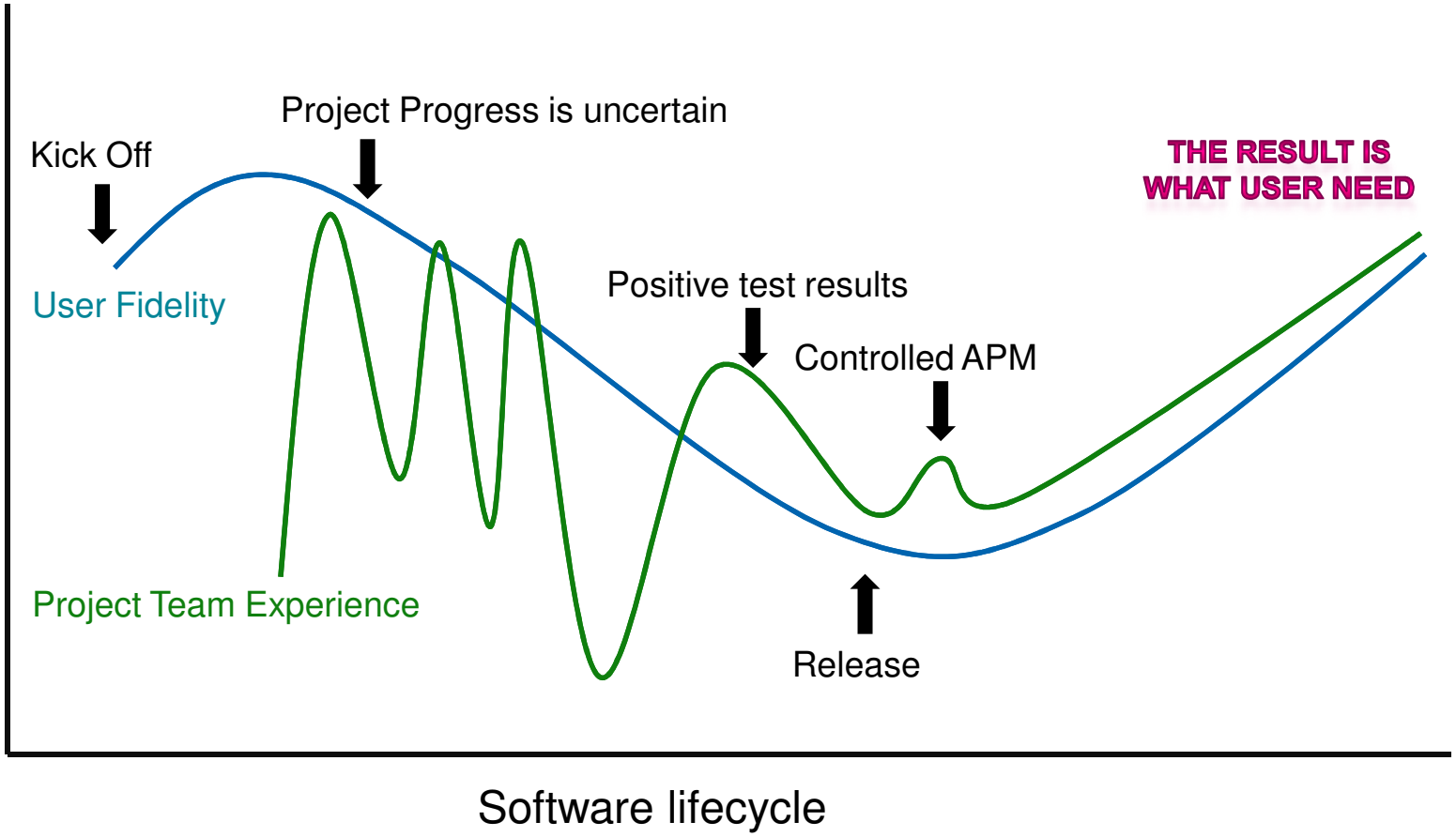
— A defect is an **anomaly** in a product.

— A failure occurs when a functional unit of a software-related system **can no longer perform** its required function or **cannot perform** it within specified limits.

user focus – project experience



user focus – project experience



APM in production

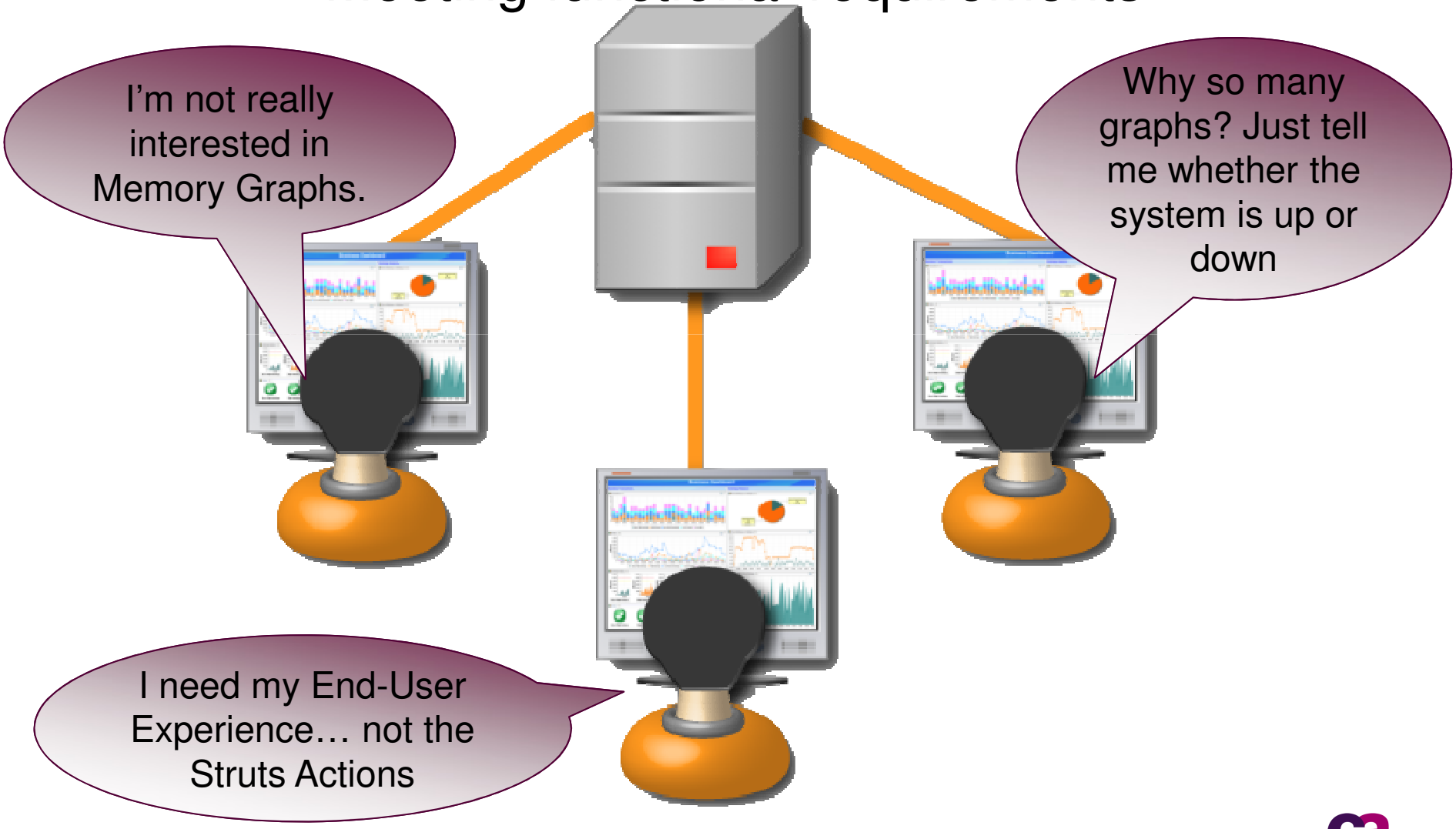
ca technologies

production monitoring – focusing on the problem and isolating it

The right view is key to identify the problem

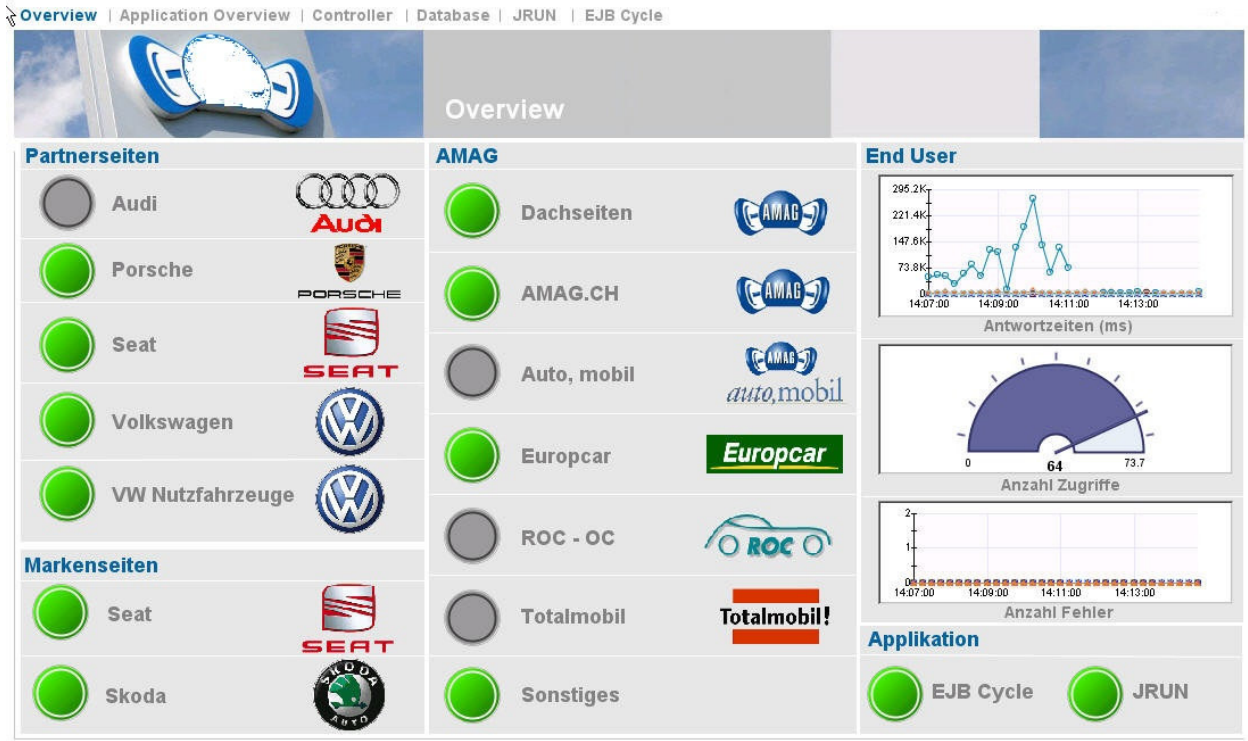
production - different views at the same problem

Meeting functional requirements



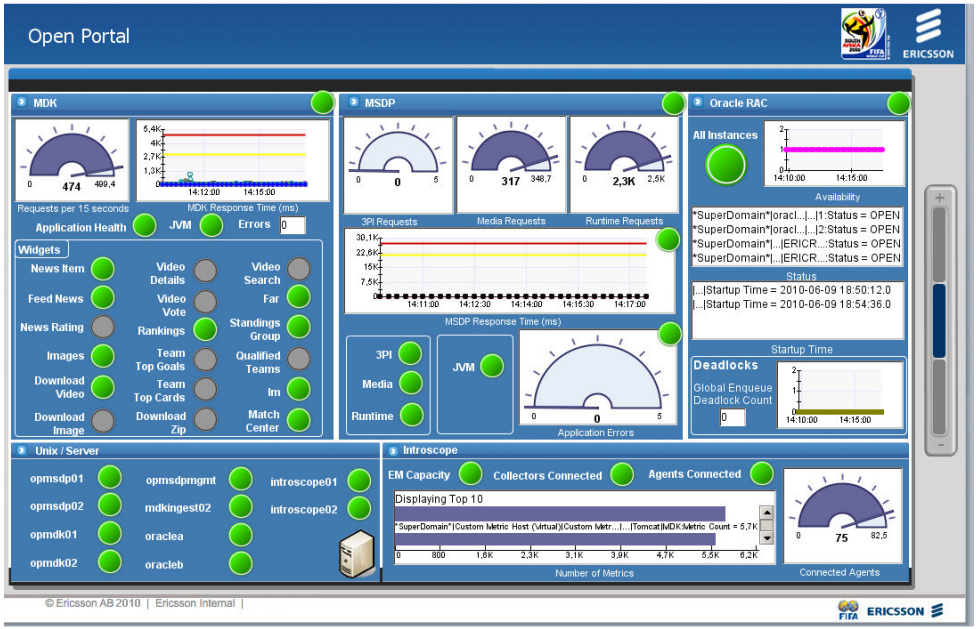
functional requirements – business

- No in deep technical information
- Fast association of the health of the system
- Correlate Evidence with a specific component or use case
- Easy to understand and to map to an incident



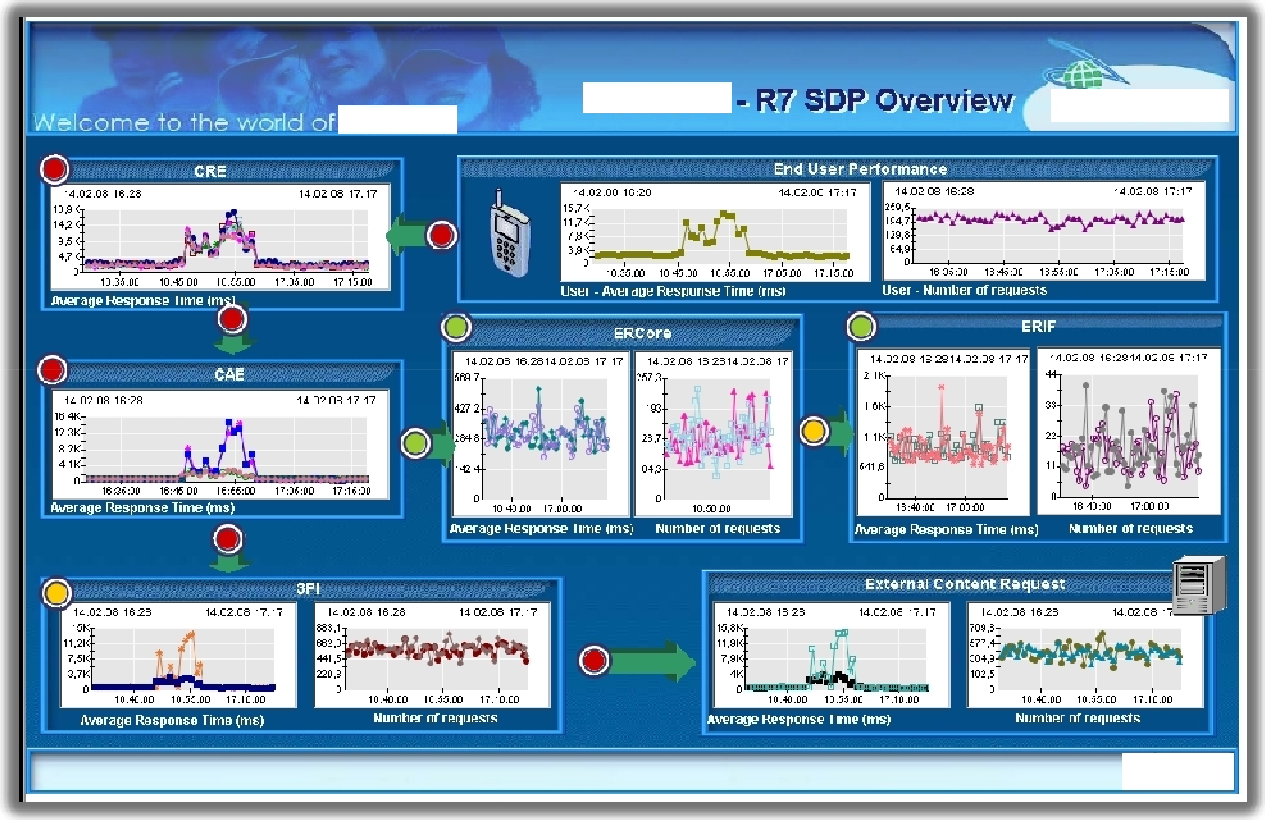
functional requirements – war-room

- Detailed level of instrumentation and visualization
- Strong drill down capabilities.
- Combine resource information with application metrics.
- Getting exact problem time stamps.



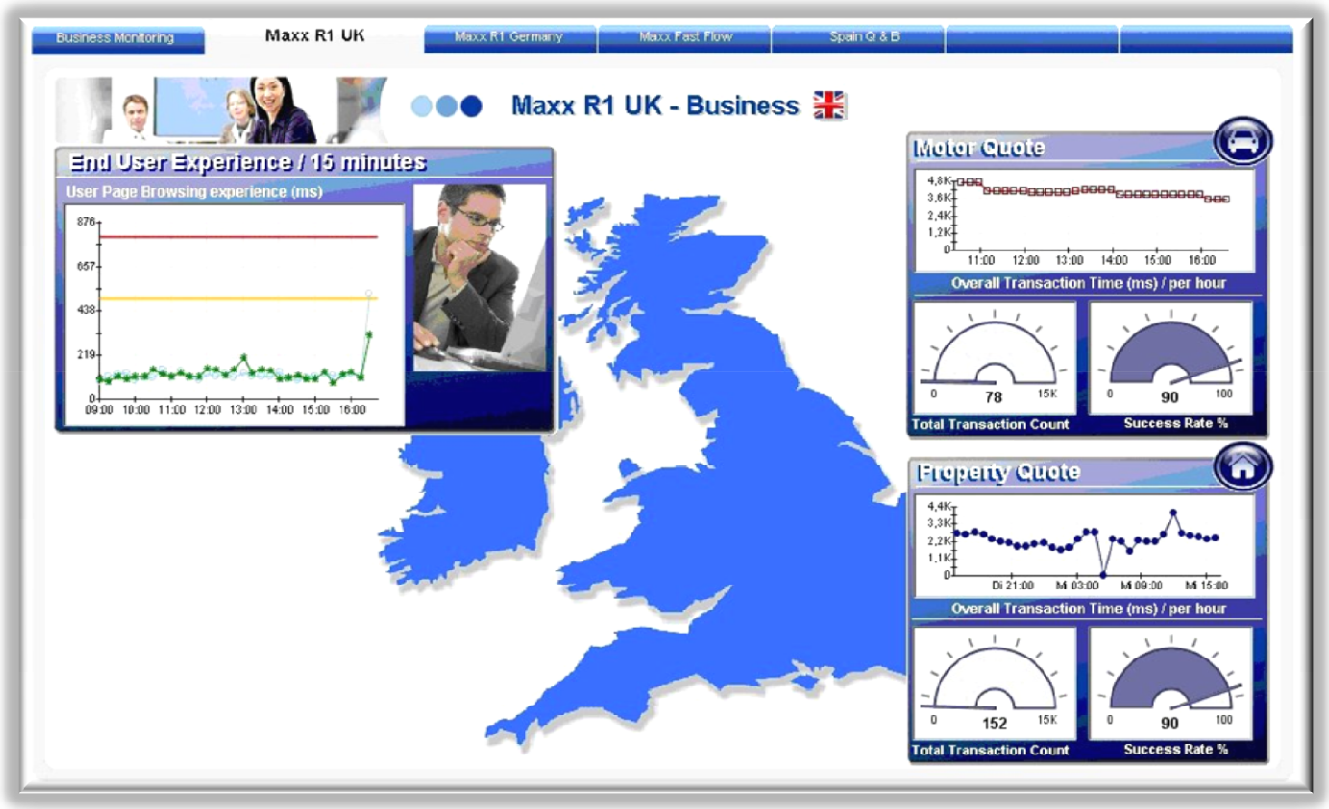
functional requirements – error and triage

- Technical captions and expressions.
- Drill down capabilities into components.
- Visualize the context of the calling component.



functional requirements – SLA management

- Focus on long term information
- End-User Experience
- Peak times



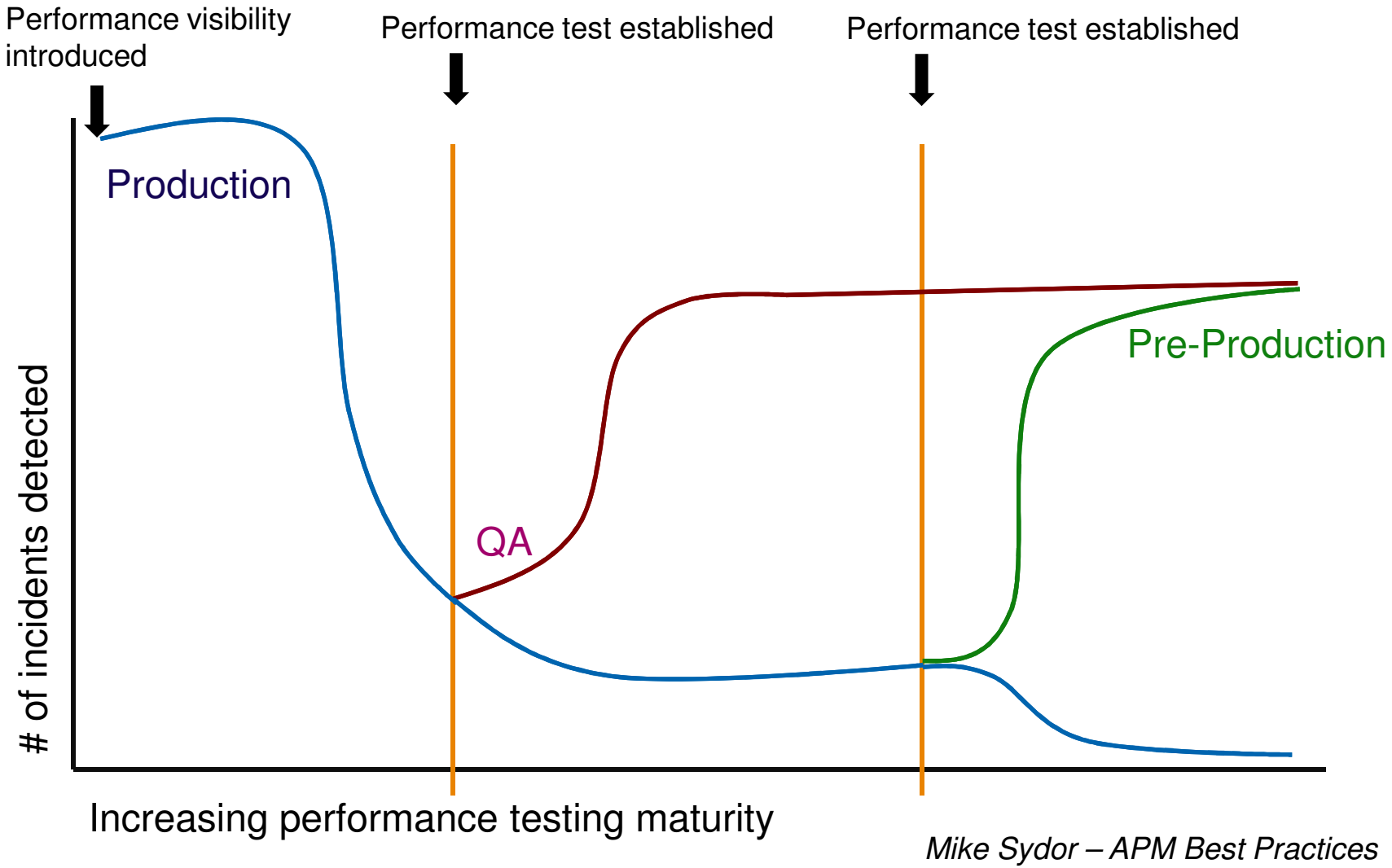
extending the monitoring to pre-production

pre-production monitoring

"I define proactive monitoring as **prevention**: reducing the number of *operational incidents* by **preventing** problematic or unstable **apps** from **reaching** the production environment. You become **proactive** because you simply **do not** let badly behaving apps into **production**."

Mike Sydor – APM Best Practices

pre-production monitoring – integrating APM



Mike Sydor – APM Best Practices

pre-production monitoring – integrating APM

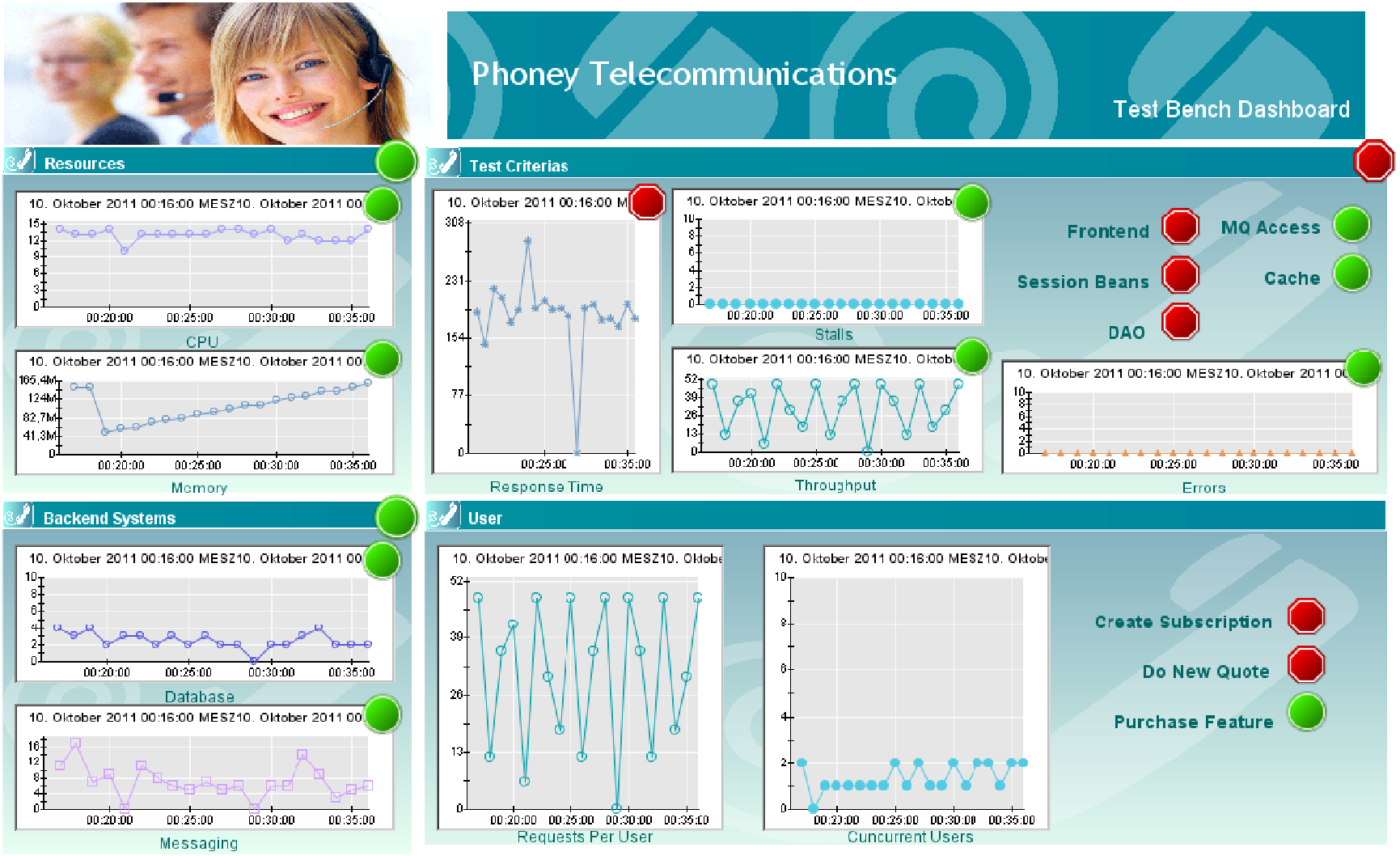
— *According to Unit Driven Testing and all agile modern software processes, every bug leads to a unit test.*

— *Unit Tests are part of deployment mechanisms and support the software quality*

How about APM?

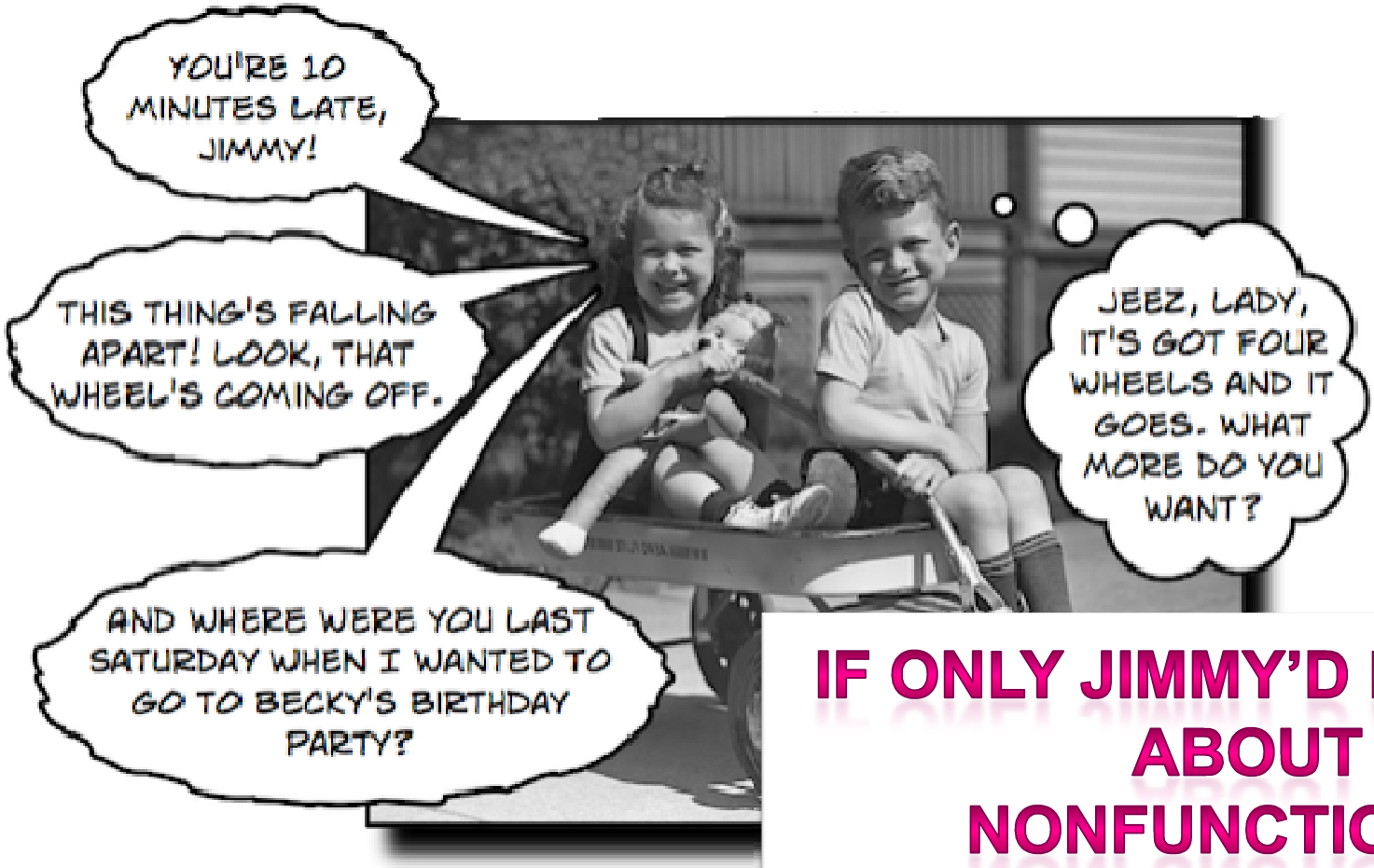
- Every non functional problem and defect must be covered by a test and verified with the monitoring solution
- Introduction of a performance workbench
- ProActive APM part understands itself as contract between Development and Operations
- Provide feedback to QA and Developers

pre-production monitoring – integrating APM



development an analysis

development and analysis – non-functional requirements



IF ONLY JIMMY'D LEARNED ABOUT NONFUNCTIONAL REQUIREMENTS …

development an analysis – developer view

Developers …

— Every developer wishes to build quality software

— Developers are proud of the code

— The developers are the firsts to know the disappointments of their products

— If it would depend on developers, the applications would never be considered completed

— However ... Software is not for its creators, it is for its users

— Developers need the right feedback from production to integrate non-functional intelligence in production

development an analysis – developer view

Successful APM starts in development

— Give developer the chance to control what information will be shown in production … they anyway know it best..

— Integration in the development process by using existing technologies (Versioning / IDE Integration)

— Good software is good testable – Common unit test rule

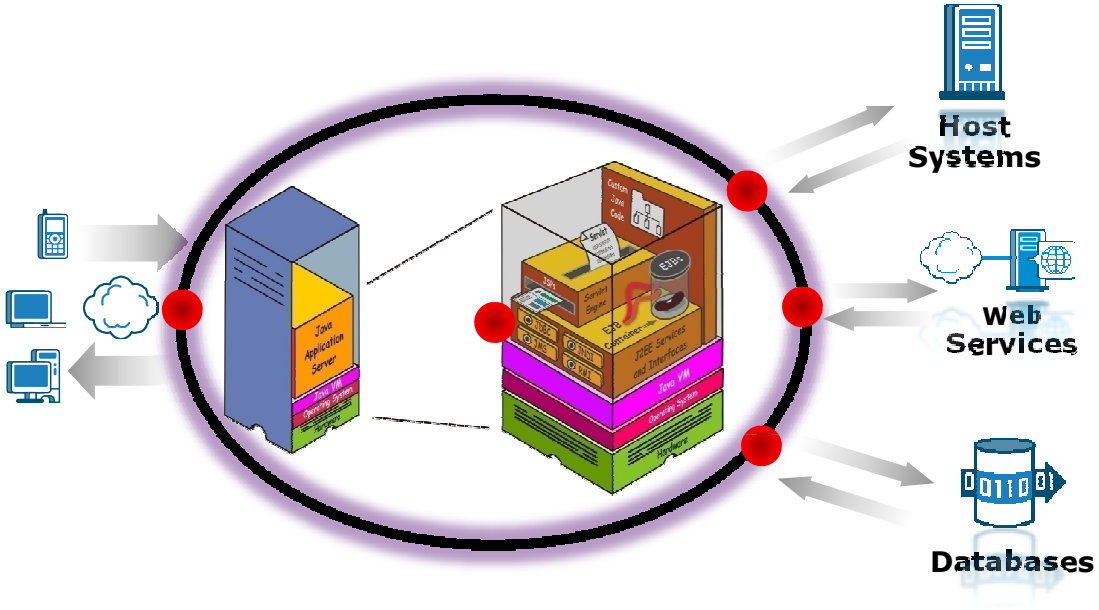
— Good software can be good integrated in APM

— Object Oriented Programming aligns with successful APM Monitoring

development an analysis – IDE integration

YATT

development an analysis – 3 columns of successful monitoring



- Boundaries
- Container
- Controller

development an analysis – how does APM align with the roles

Developer jobs in the modern world

— A developer doesn't develop a complete use case

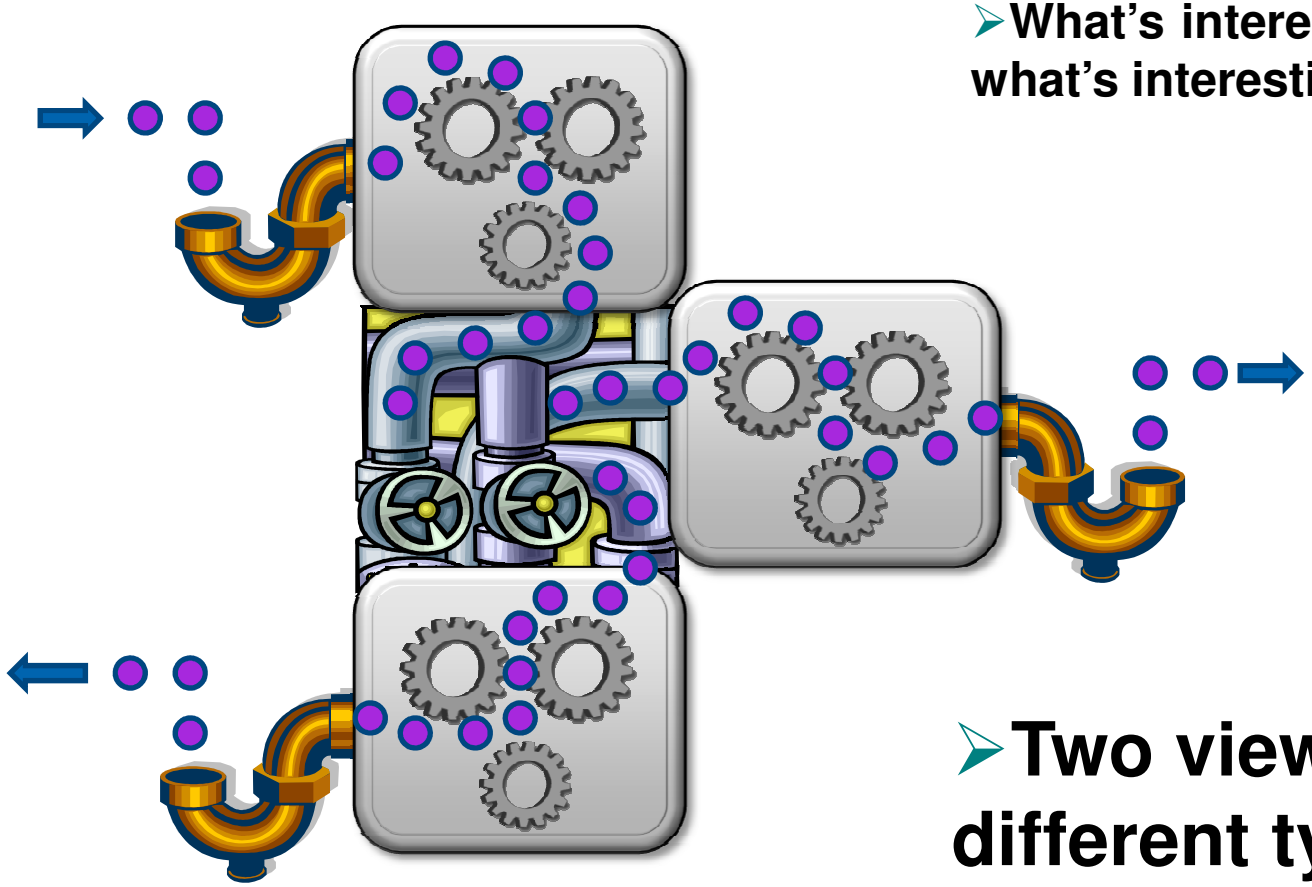
— Developer think in components and units

— Developer roles:

- Bean Developer
- Frontend Developer
- EAI Developer
- DB specialist
- MQ specialist
- Framework developer

— **It fully aligns with successful APM monitoring**

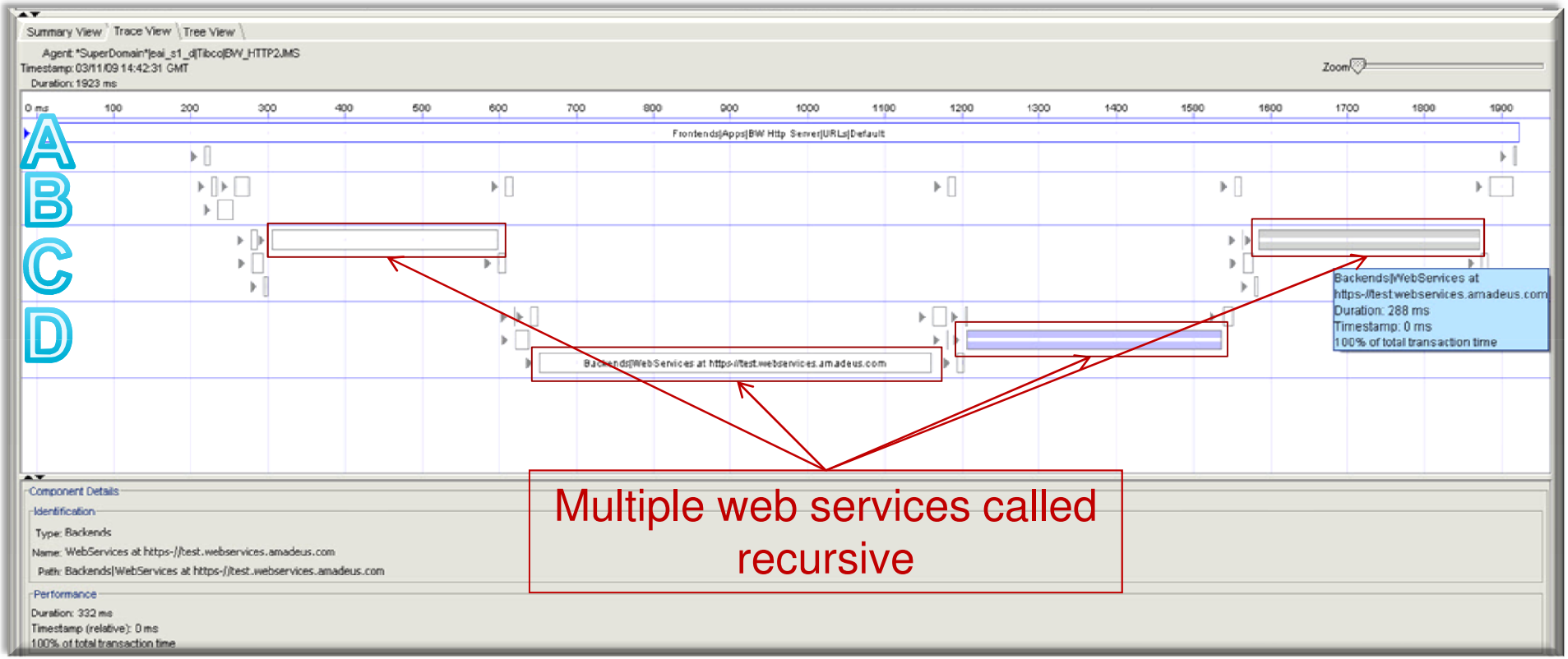
development an analysis – application flow monitoring



- What's interesting in development vs. what's interesting in production.

- Two views to two different type of problems

call stacks – information overflow



- **30 Threads across 4 JVM's for one transaction.**

conclusion

conclusion

- Successful APM starts in development
- Developers can influence “views” on the production environment
- APM is best used as a contract between development and production
- APM helps to verify KPI and non-functional verifications in pre-production
- APM in production
  - Provides feedback to developers
  - Defines test cases in pre-production
  - Verifies pre-production

thank you