Quality Matters …

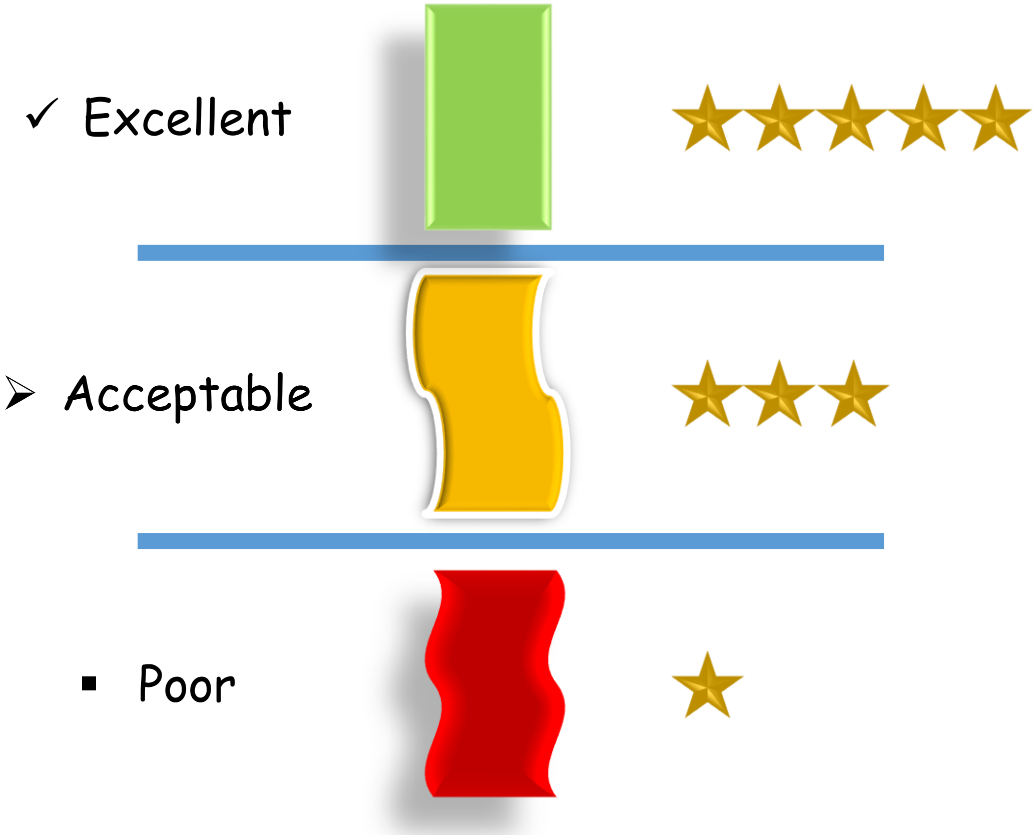
An Introduction

A quick self-assessment:

Looking into the mirror …

A quick self-assessment:

Where is your quality?



- ✓ Excellent ★★★★★
- ➢ Acceptable ★★★
- ▪ Poor ★

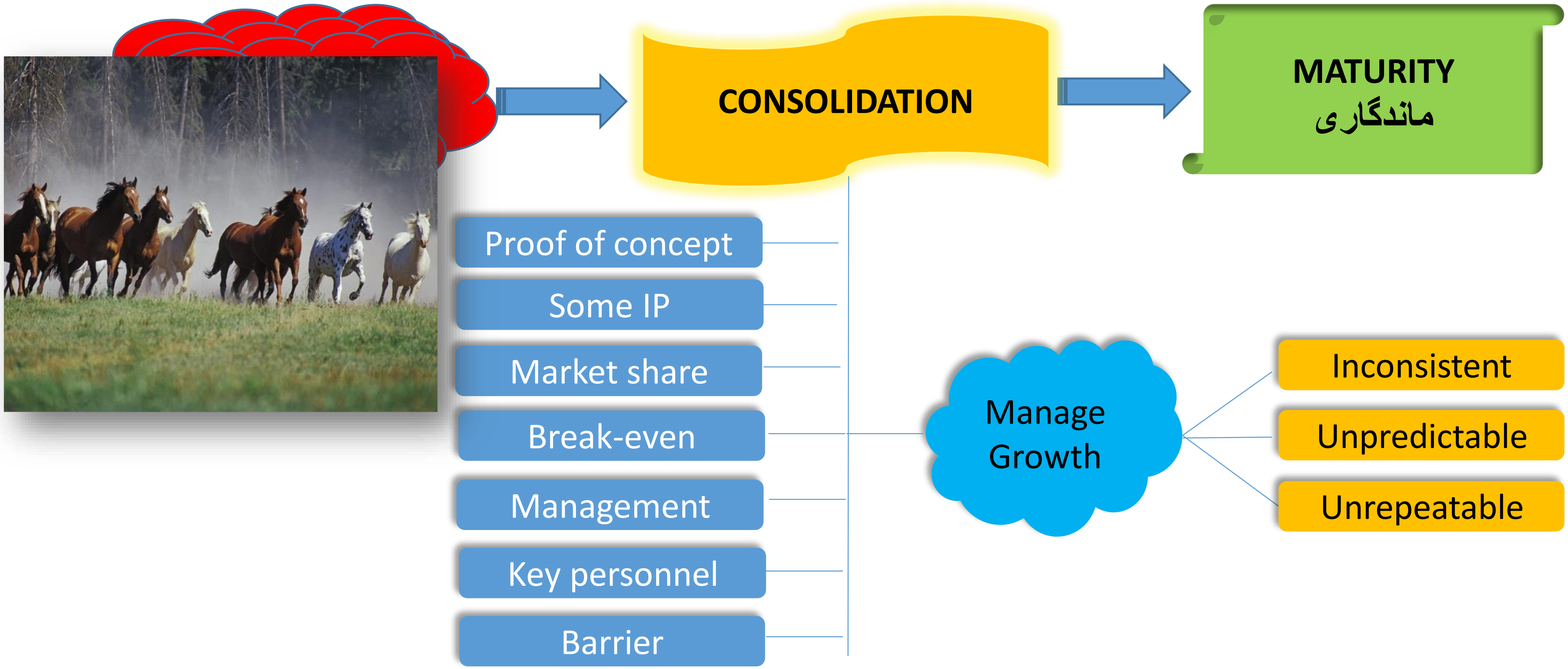
How do you know?

- Feedback from customers/users/competition
- Press reviews/critiques, word of mouth
- Our own internal audits/measurements
- Independent evaluations
- It’s just a hunch!

And why is that?

- Poorly defined (often changing) customer requirements and expectations
- Racing against time
- Inadequate testing (at various levels)
- Lack of a viable quality assurance plan
- Lack of a proper release management process
- It’s the nature of our business (E-Commerce and SaaS in particular)
- Anything else?

Lifespan of a typical software company



CONSOLIDATION

MATURITY
ماندگاری

Proof of concept

Some IP

Market share

Break-even

Management

Key personnel

Barrier

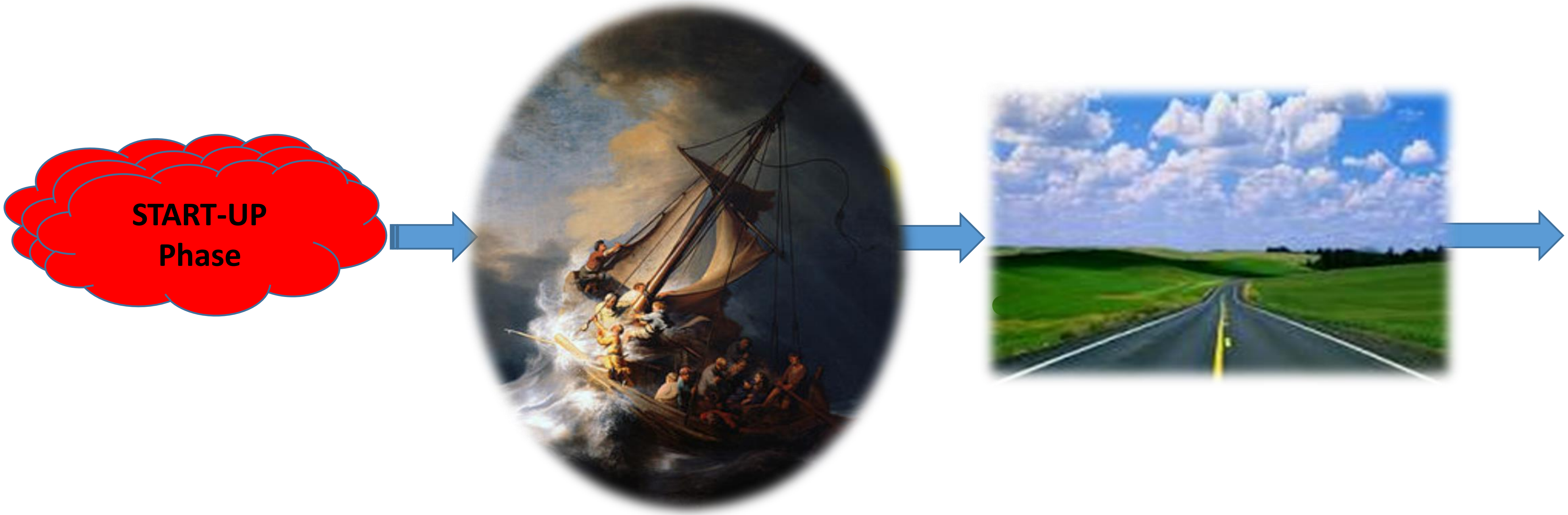
Manage Growth

Inconsistent

Unpredictable

Unrepeatable

Does this feel familiar?



START-UP Phase

Quality Matters …

- What is this ‘thing’ called software quality? Why should you care?
- How do we build quality ‘into’ software?
- What has quality got to do with the survival of your company?

Quality Matters …

- ✓ **What is this ‘thing’ called software quality? Why should you care?**

- How do we build quality ‘into’ software?

- What has quality got to do with the survival of your company?

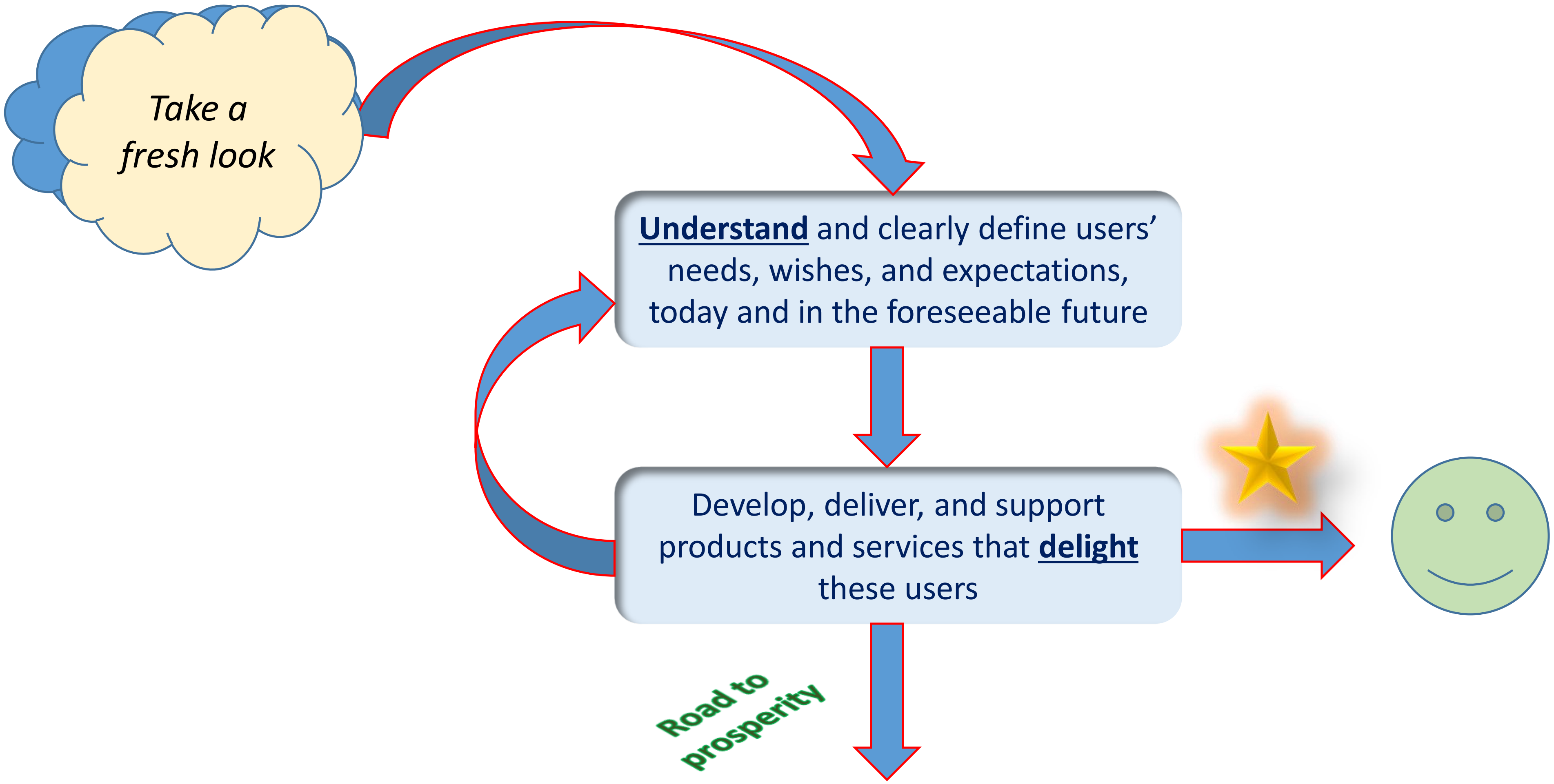
Quality IS NOT …

- An event
- An afterthought (not an additive)
- = Testing
- Magic (it doesn’t just happen)
- Gravy (i.e. not separate from functionality)
- Someone else’s responsibility
- Free

Quality IS …

- Elusive (self-evident)
- Subjective
- Hard to measure
- Incremental
- Inconclusive
- A collective responsibility
- Costly (but well worth the investment)

First, let's settle on a definition …



Take a fresh look

Understand and clearly define users' needs, wishes, and expectations, today and in the foreseeable future

Develop, deliver, and support products and services that **delight** these users

Road to prosperity

In a nutshell …

Open for Business

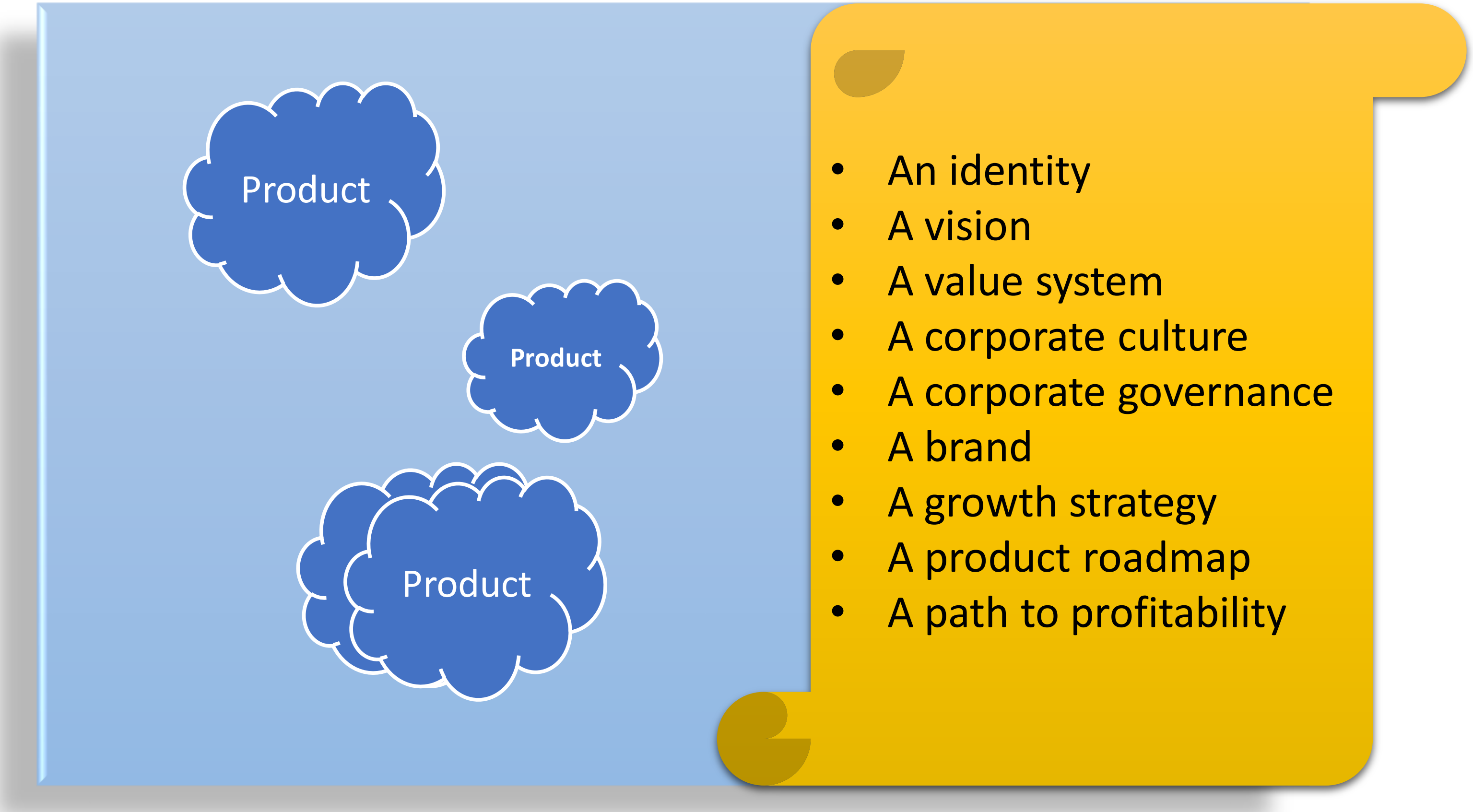
Rock-Solid Dependability

So, why should you care about quality?

- Develop products/services that delight customers and keep them loyal
- Control and efficiently manage cost of business
- Motivated, proud, world-class employees
- Be a market leader (innovative, profitable, mature)
- Be able to compete in the global markets

So, why should you care about quality?

- You want to build a *viable* company:

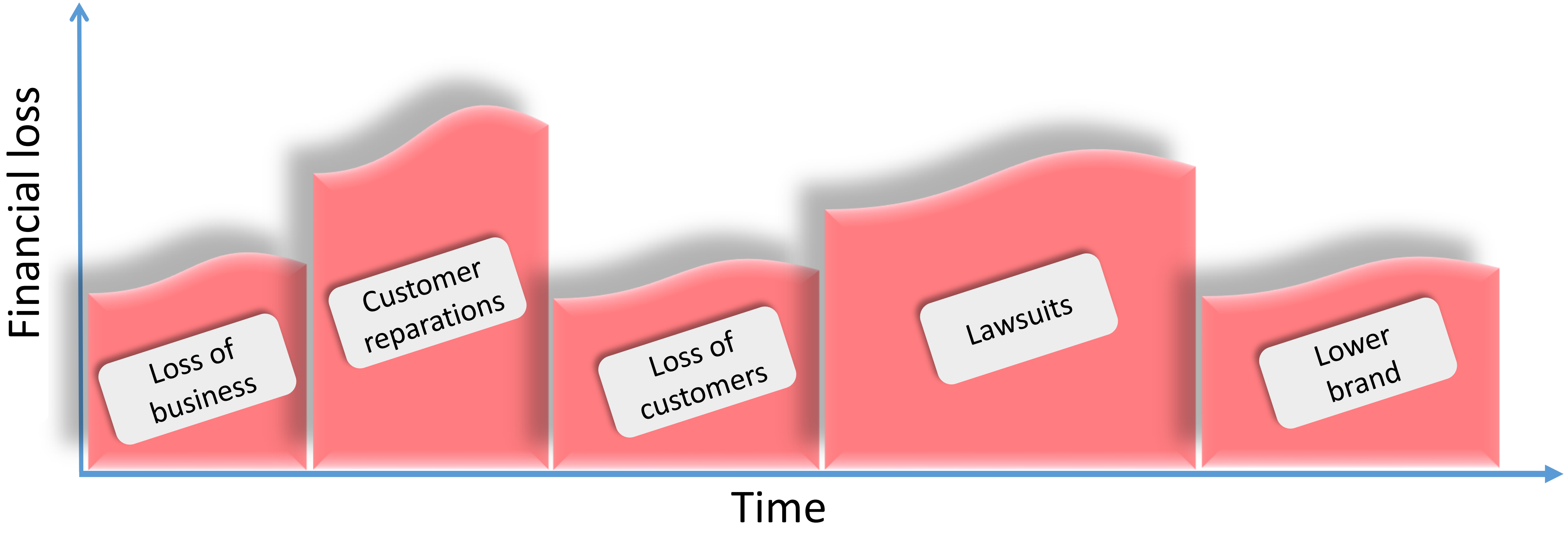


- An identity
- A vision
- A value system
- A corporate culture
- A corporate governance
- A brand
- A growth strategy
- A product roadmap
- A path to profitability

So, why should you care about quality?

Business value of software quality

- Low quality software creates less business value than high quality software



Barriers against achieving quality in software

- Cultural/social baggage or misperceptions – it doesn’t really matter!
- Unclear corporate vision and leadership – a.k.a. management incompetence
- Shortage of competition – won’t last forever
- Poorly defined, ambiguous, unrealistic and/or unachievable goals
- Teamwork dynamics – lack of focus and commitment
- Absence of ‘Do it right the first time’ mentality – confidence and credibility
- Fearful of costs, unclear ROI – such a myth!

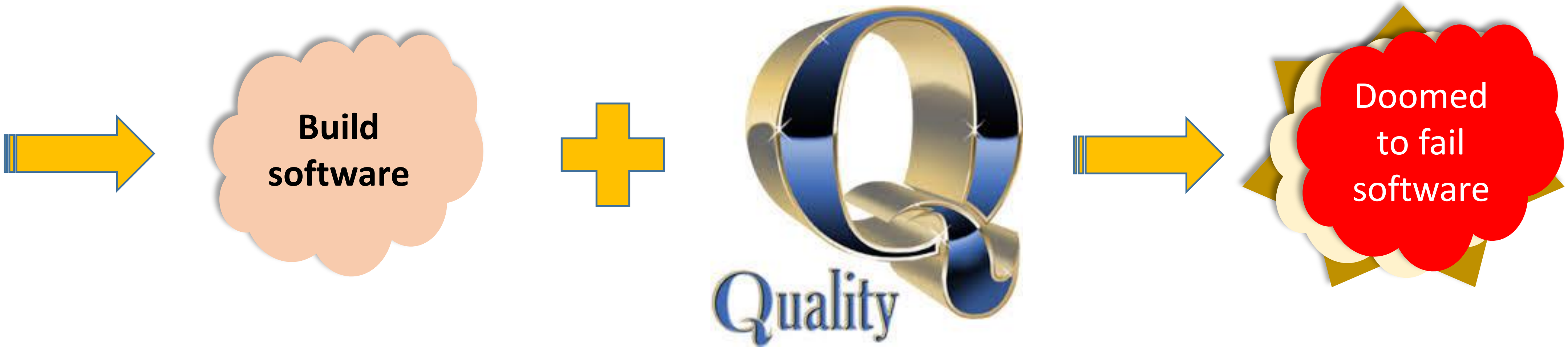
Quality Matters …

- What is this ‘thing’ called software quality? Why should you care?

✓ **How do we build quality ‘into’ software?**

- What has quality got to do with survival of your company?

The hard (and very expensive) way …


Build software

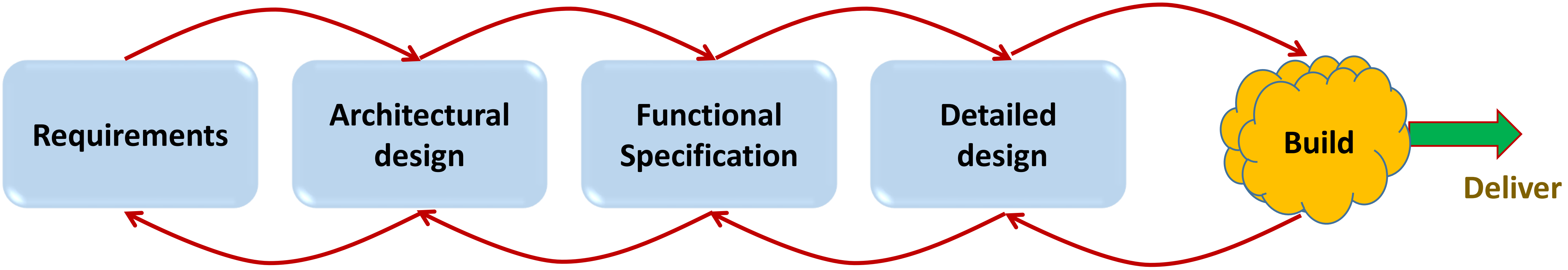
Doomed to fail software

- It costs a lot more to fix a dysfunctional/broken software than to create a software that fully

- It costs even less when you 'do it right the first time'

Software quality as an engineering discipline

- *Engineering* (from Latin ingenium, meaning "cleverness" and ingeniare, meaning "to contrive, devise") is the application of scientific, economic, social, and practical knowledge in order to invent, design, build, maintain, and improve structures, machines, devices, systems, materials and processes.



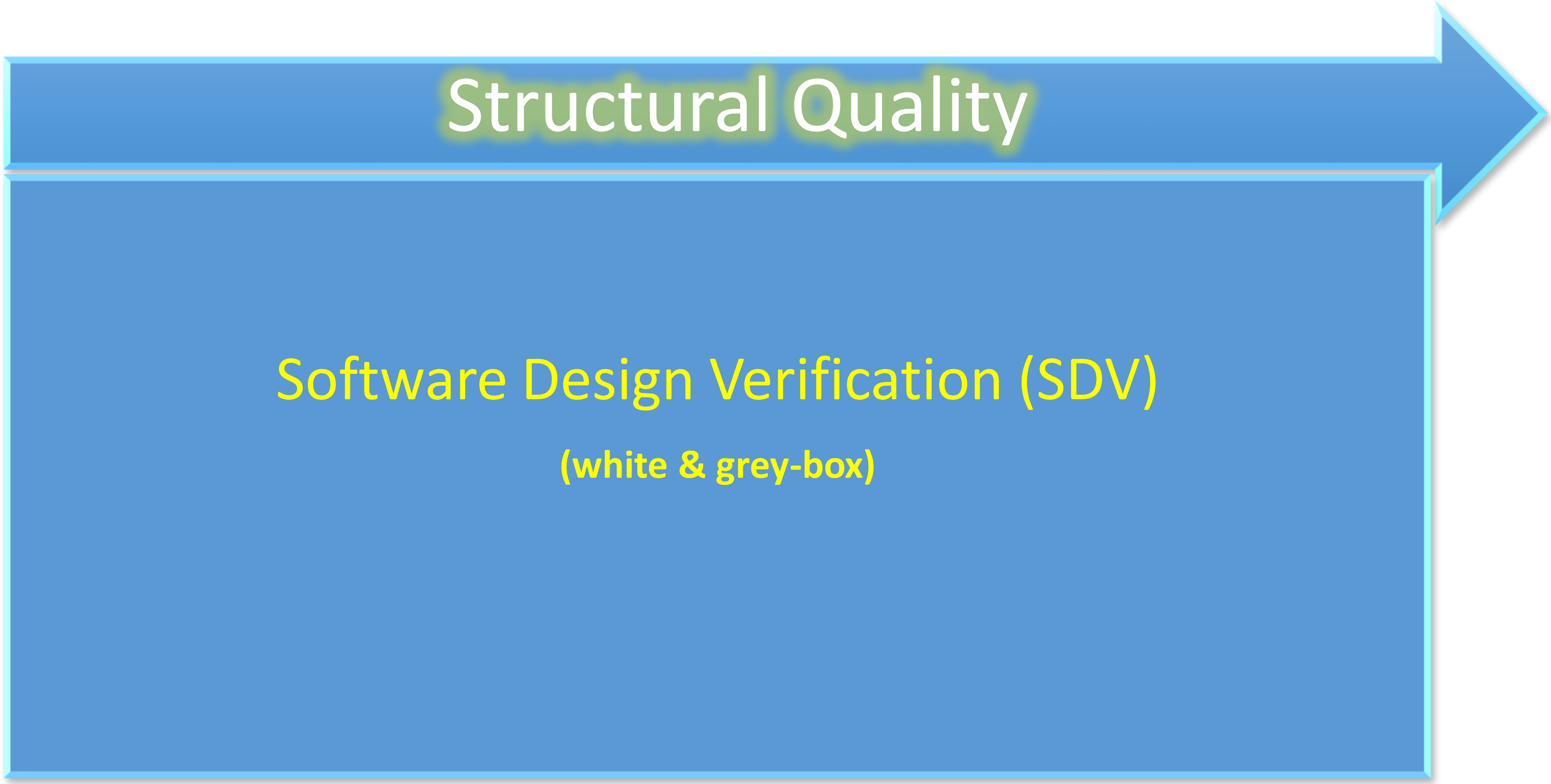
Three aspects of software quality

Functional Quality

Testing

(black-box)

Three aspects of software quality



Structural Quality

Software Design Verification (SDV)

(white & grey-box)

Three aspects of software quality

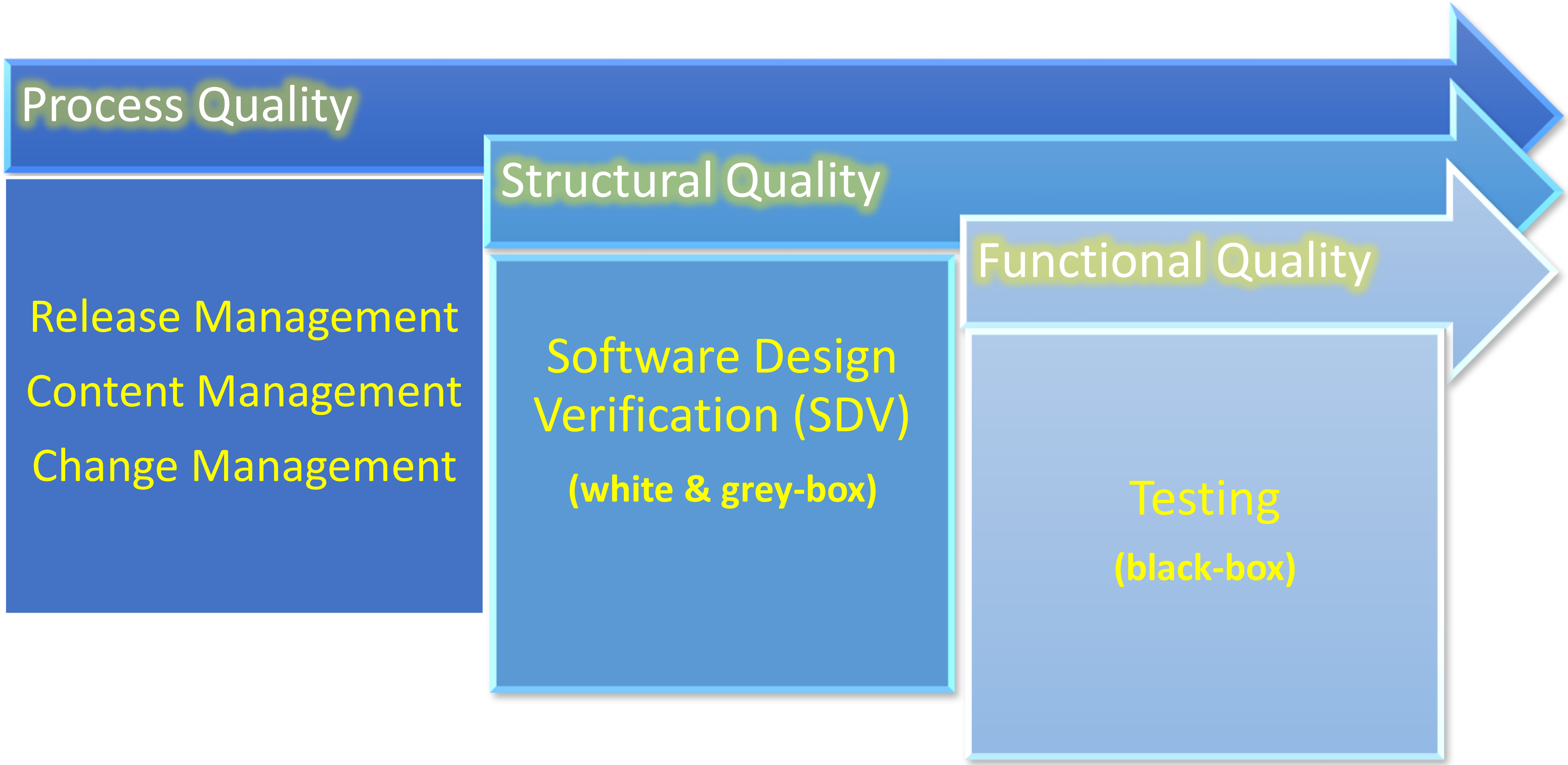
Process Quality

Release Management

Content Management

Change Management

Three aspects of software quality



Process Quality

Release Management

Content Management

Change Management

Structural Quality

Software Design Verification (SDV)

(white & grey-box)

Functional Quality

Testing

(black-box)

Functional quality (Testing)

- Works as specified functionally and non-functionally (aka validation)
- Defect “free”
- Reliability
- Desired performance
- Desired security
- Scalability
- Availability (SLA)
- Failover and disaster recovery
- Ease of deployment, learning, and use

Structural quality (SDV)

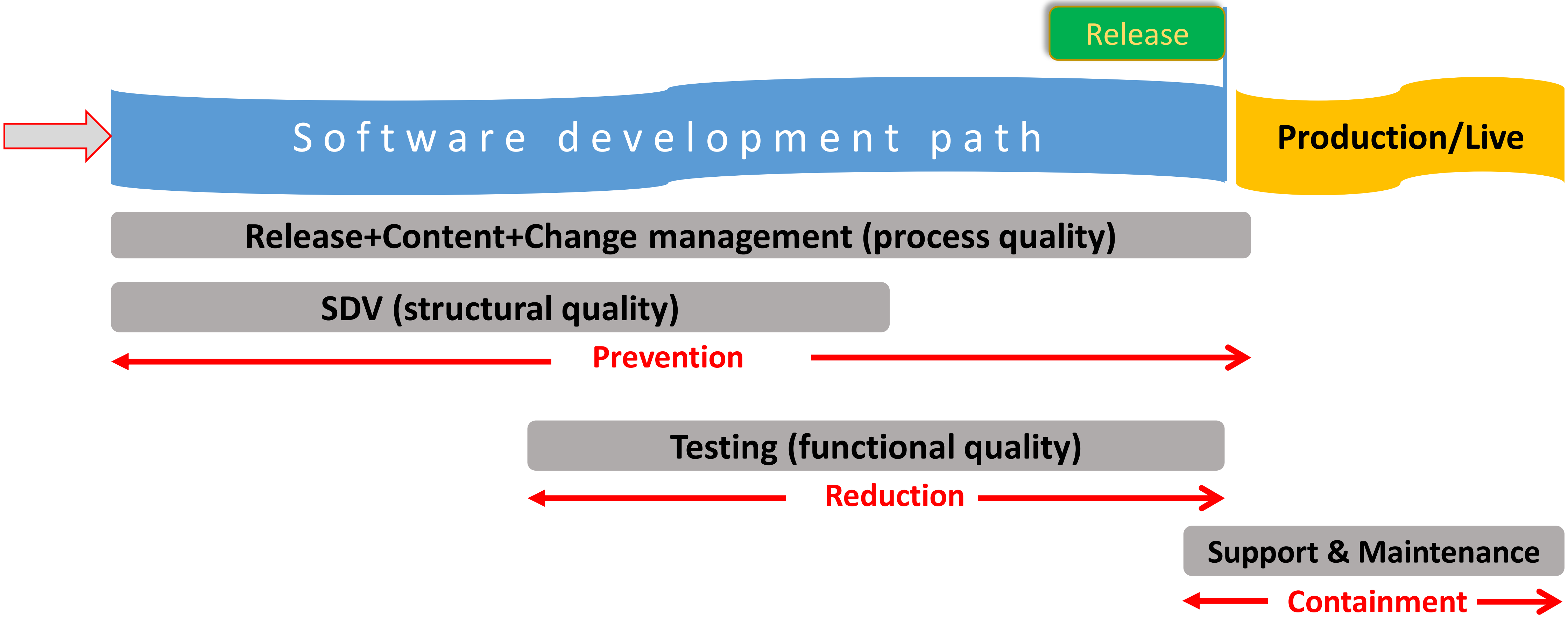
- Testability
- Traceability
- Maintainability
- Portability
- Interoperability
- Code efficiency
- Test coverage
- Code security
- Compliance (standards, best practices, etc.)
- Correctness

Defect management …

- The meaning of a defect

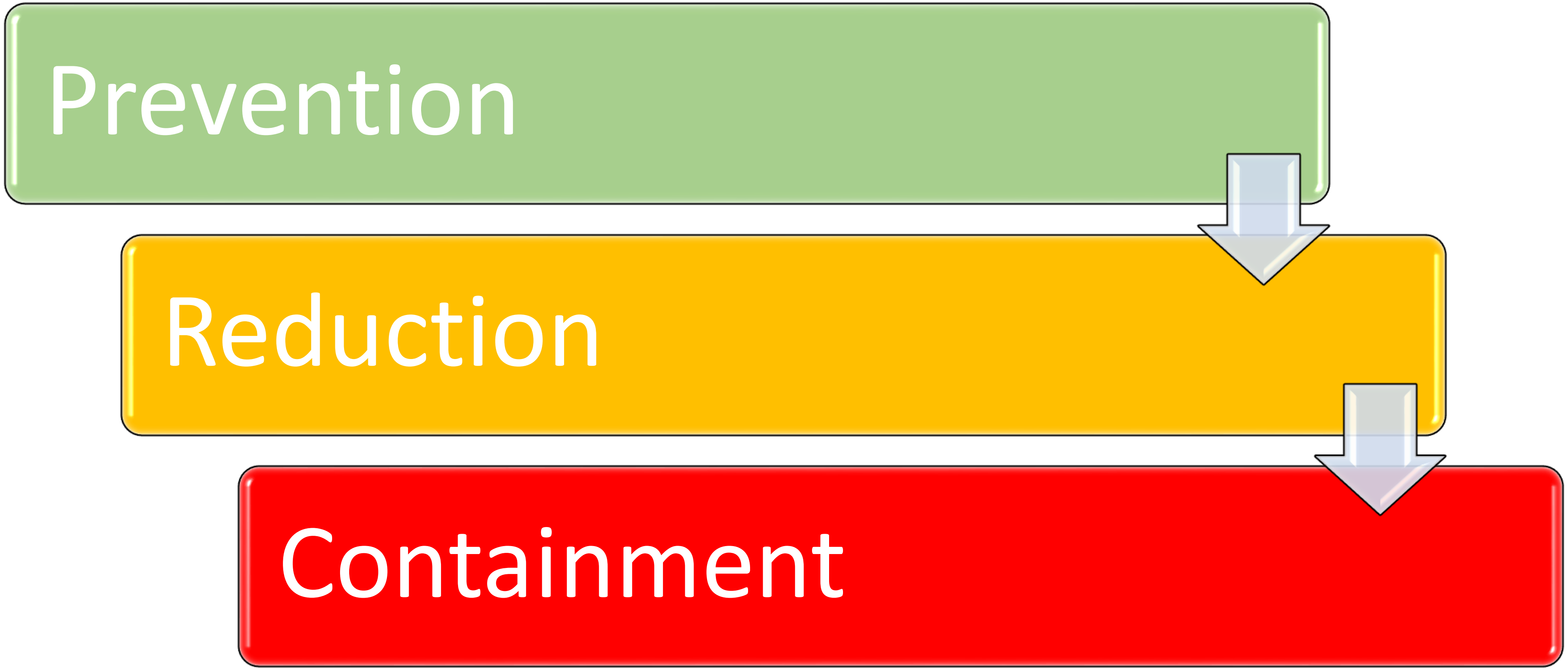
(the ‘defect zone’ illustration from Software QA IIIS slides)

Defect management (aka bug nuisance)



Release
Software development path
Production/Live
Release+Content+Change management (process quality)
SDV (structural quality)
Prevention
Testing (functional quality)
Reduction
Support & Maintenance
Containment

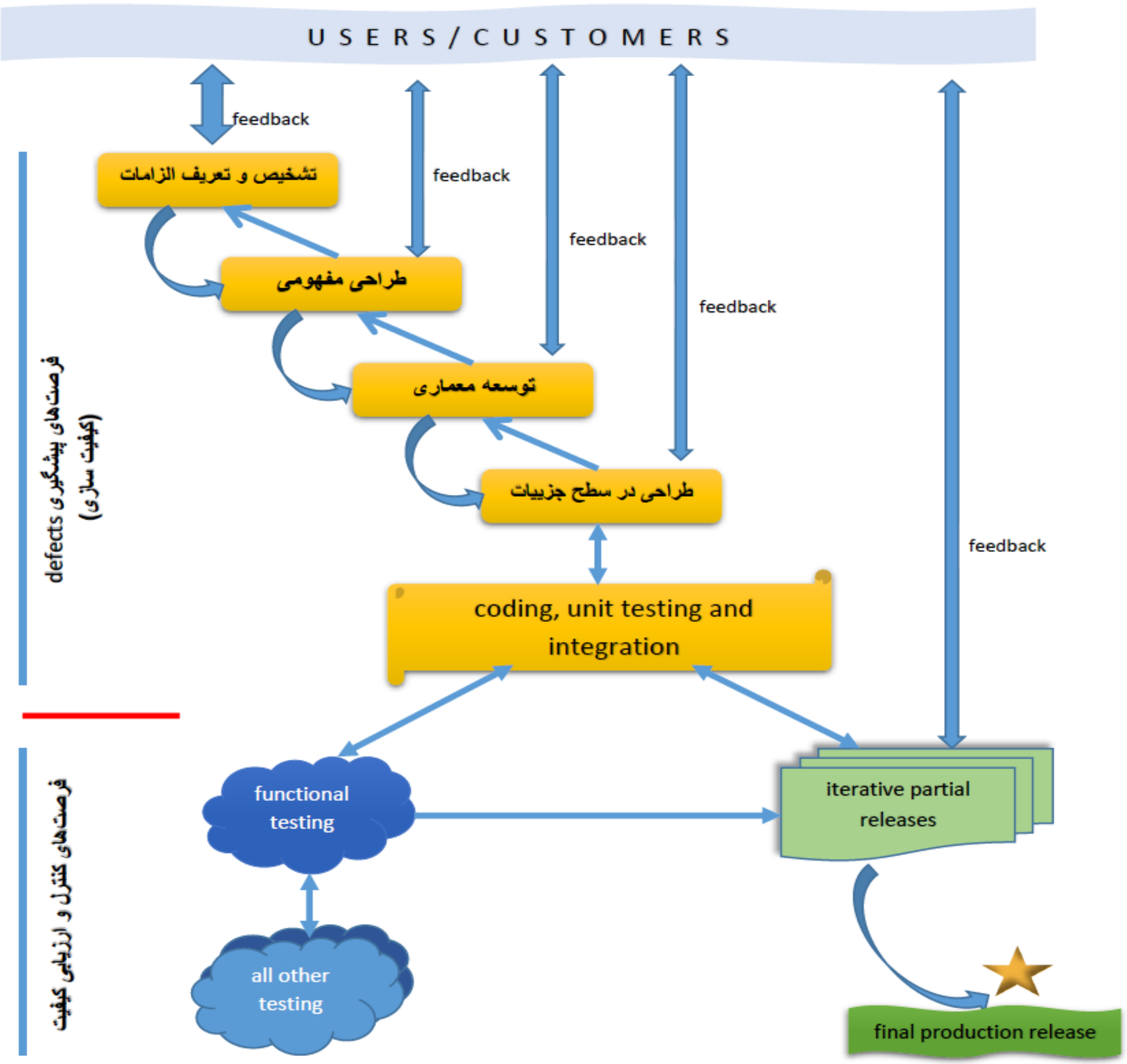
Defect management (aka bug nuisance)



Prevention

Reduction

Containment



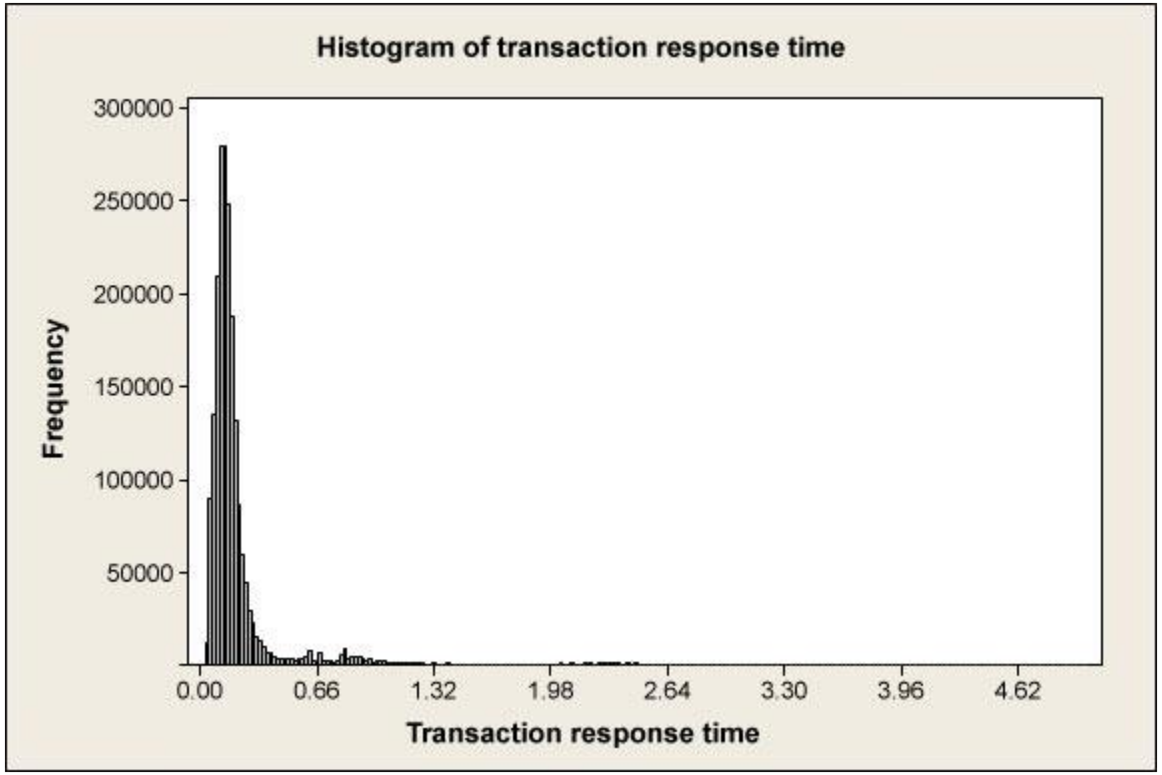
شکل 7. فرصت‌های پیشگیری و کاهش کاستی‌ها و باگ‌ها در قالب یک مدل iterative

Focus on user experience

- **Measuring user experience can produce objective metrics**
- **You can't measure user experience with a ruler, so measuring user experience objectively is generally a matter of aggregating subjective opinions.**
- **What are some good objective methods for measuring user experience on an application?**
- That's a tough question. Let's explore what a good objective measure of user experience might mean and what it could look like.
- User experience is subjective by nature, so finding an objective measure is tough. Surveys can be objective if the questions and results are presented well. Saying, "The majority of our customers rate this as an eight out of ten" is a lot different than, "I think it's an eight."

- However, my clients often find survey data unacceptable or otherwise undesirable. They want a single number pulled out of a computer, something like click-through rates, time spent on the page or page load times. All of those can be aspects of user experience; however, looking at those metrics actually provides a picture of application performance.
- The next challenge is to come up with the right questions to ask. The questions should be precise because company performance will be measured against the questions as asked. Questions for measuring user experience should address a customer's likelihood to use the software again and to recommend it to others.
- James Surowiecki, author of *The Wisdom of Crowds*, claims in his book that large groups tend to come to the right answer more often than individuals do. Imagine a company with tens of thousands of customers that could question any block of one thousand at a time and the answer always came out about the same -- somewhere between 8.3 and 8.4. The organization could reasonably say the objective measure of the application's user experience was about 8.35.

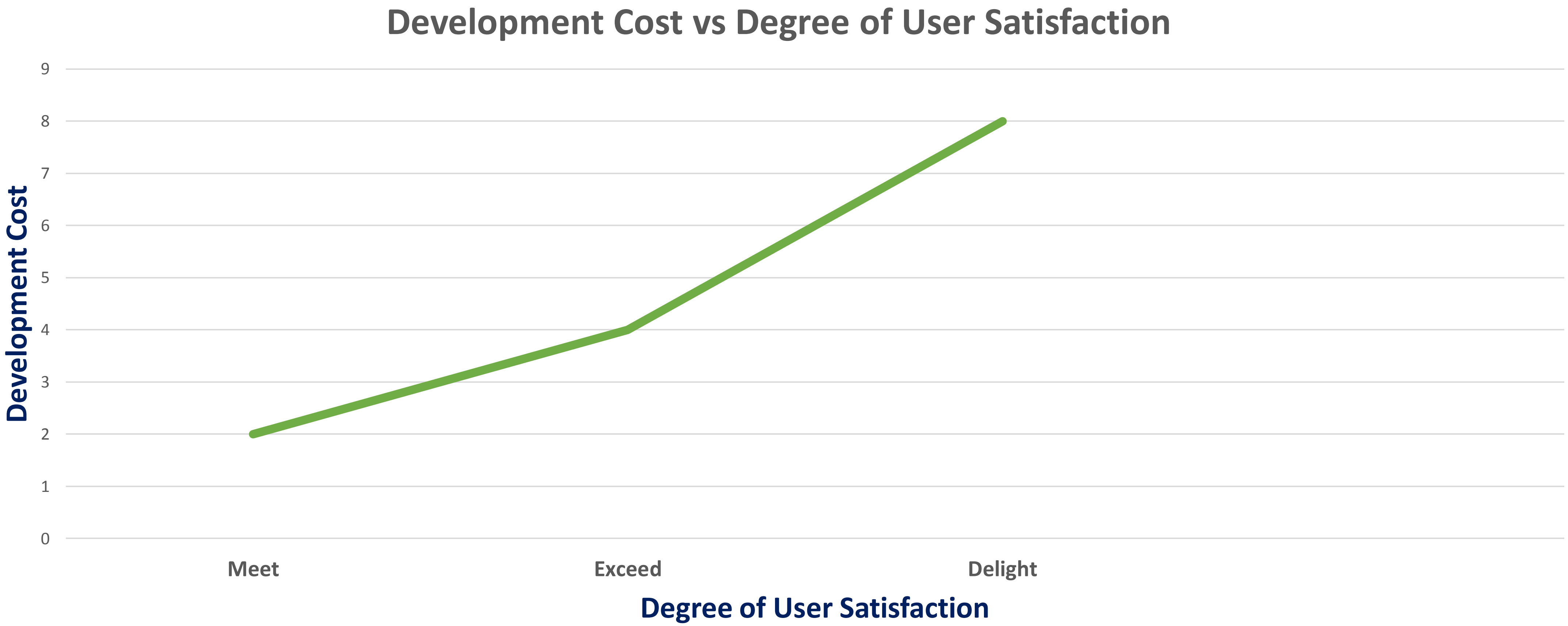
- Still, averages don't always tell the whole story. Imagine measuring user experience by analyzing response time for library software. The software has two broad categories of users: inner-city librarians and those in rural areas with slow connections. Use is split almost exactly evenly. If we only measure the average, the speedy performance of the urban libraries might offset and hide rural library performance issues.
- Even worse, the slow performance could vary by screen, with signup and login taking so long the users abandon the process, while everything else is blazing fast. Again, it is important to recognize that user experience is not the average of what people feel. Each user has an individual experience. When looking at large populations, it might be better to look at, say, the percentage of people (or pages) at 0-1.99 seconds to serve, the percentage at 2.0-3.99 and so on. A histogram is a graph that can help with visualizing these numbers.


Histogram of transaction response time
Frequency
300000
250000
200000
150000
100000
50000
0.00
0.66
1.32
1.98
2.64
3.30
3.96
4.62
Transaction response time

- A histogram of hypothetical transaction response times.
- It is tempting to look at a histogram, see that most of the users have good responses and throw out the outliers. Avoid this temptation. Those outliers are often the best source of new product ideas or potential extensions of the software.
- We started off with an attempt to gather a single number for measuring user experience that stands up to scrutiny. It may be possible to look at the user base as a whole, to let the users tell you what matters to them. However, if those users care about different things or are trying to get different things done, then I suggest getting past the averages by looking at the data using a histogram.
- When opinion surveys are not acceptable, then response times might be your best bet from a numbers perspective. Where numbers aren't attainable, sometimes Goldilocks's measurements ("too big," "too little" and "just right") are a better fit.

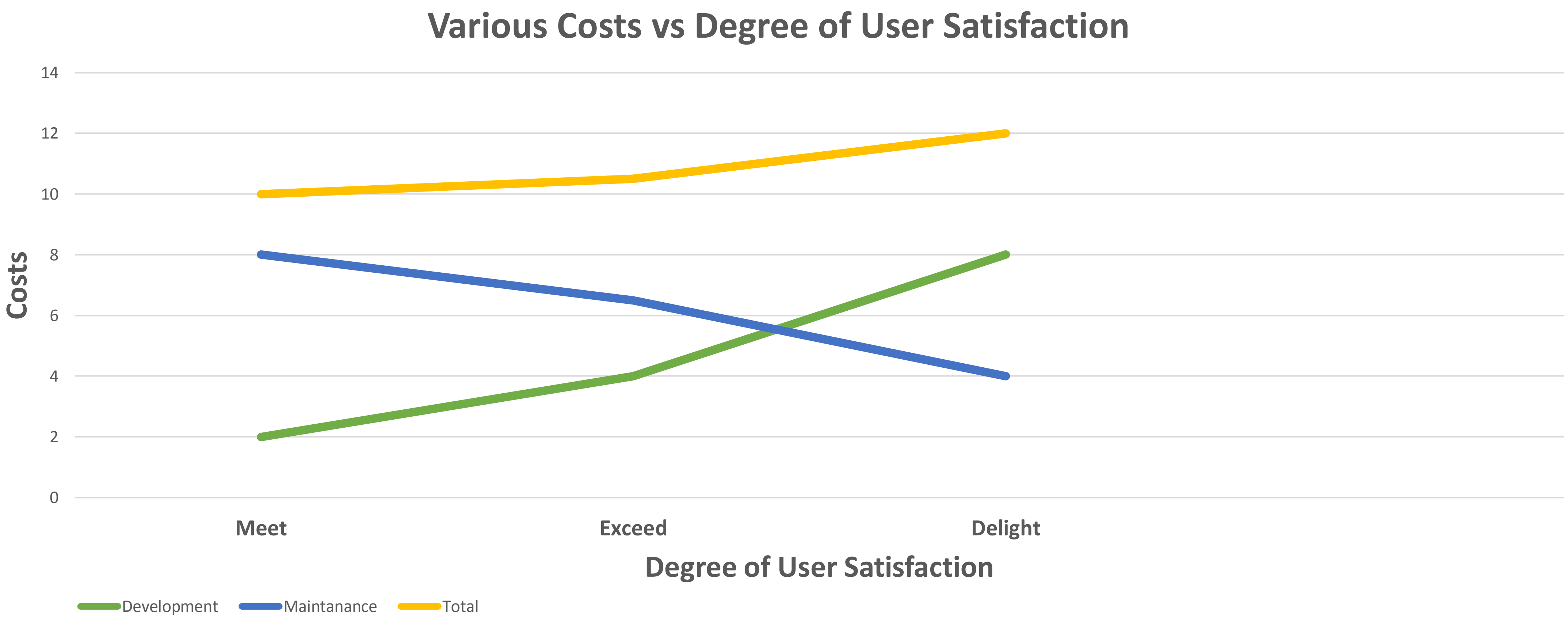
Focus on delightful user experience

Cost projection (development)



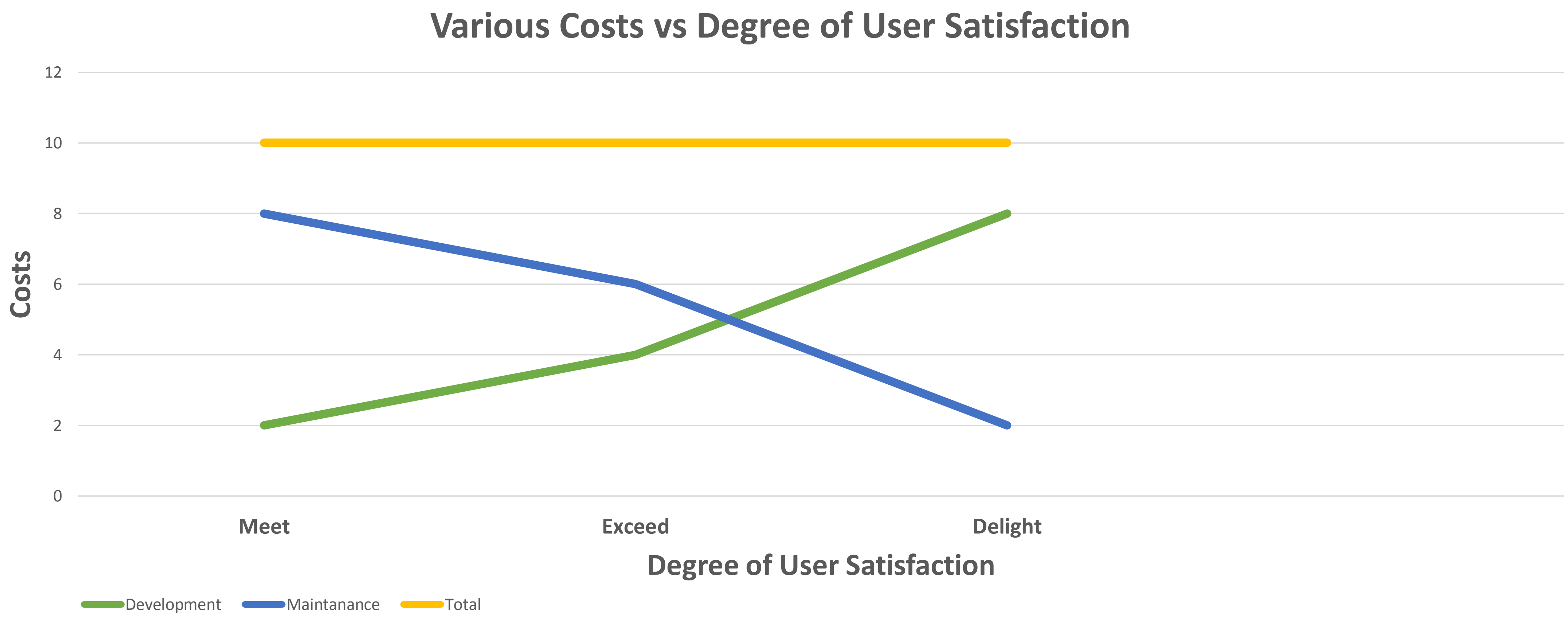
Focus on delightful user experience

Cost projection - realistic



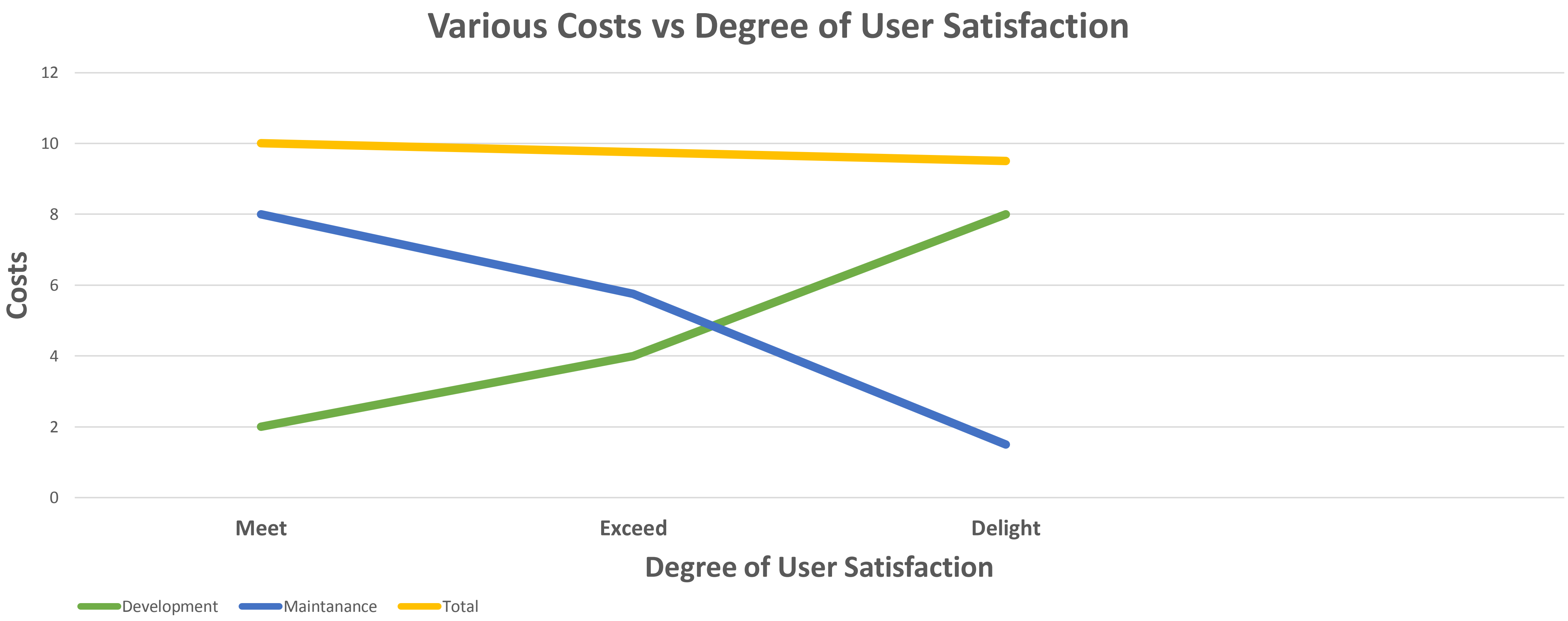
Focus on delightful user experience

Cost projection - achievable

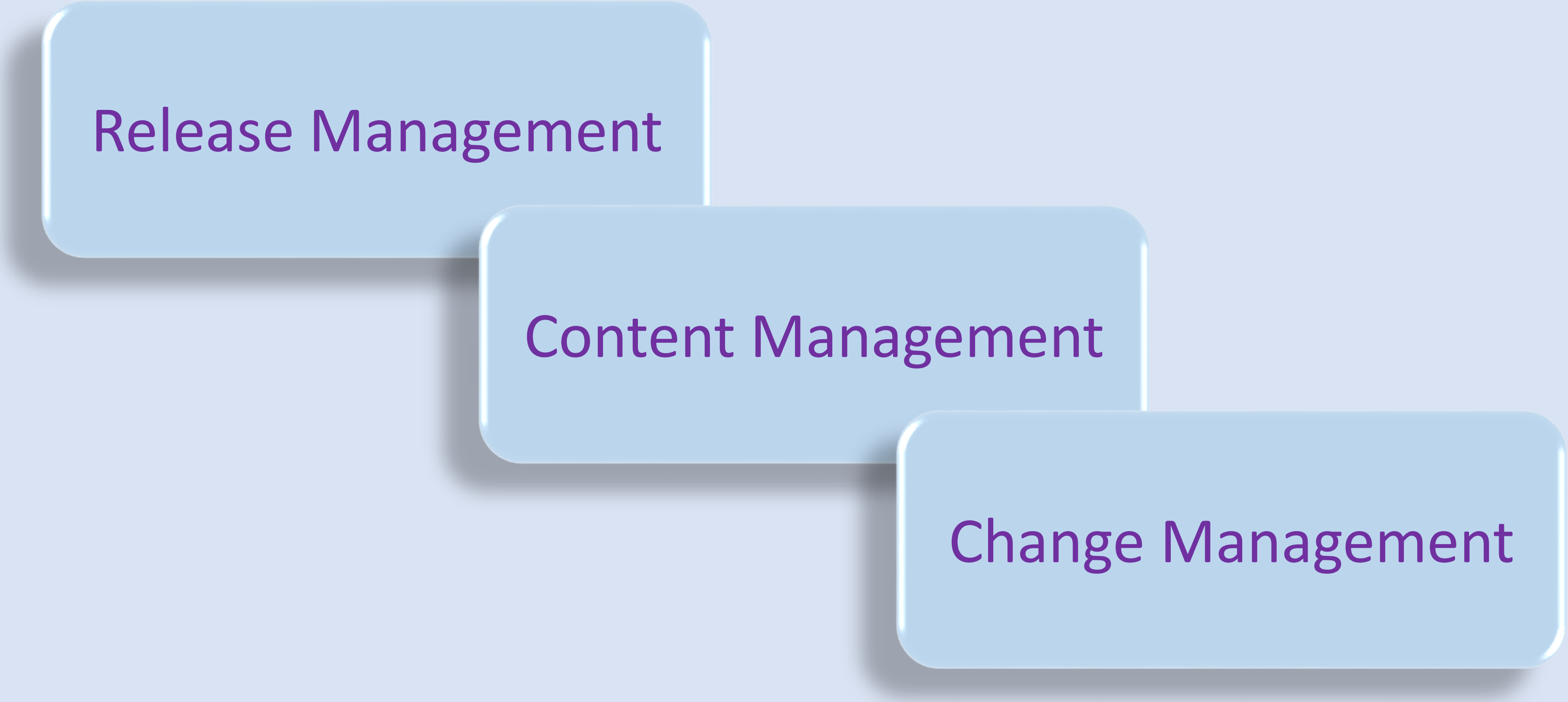


Focus on delightful user experience

Cost projection – possible



Process quality



Release Management

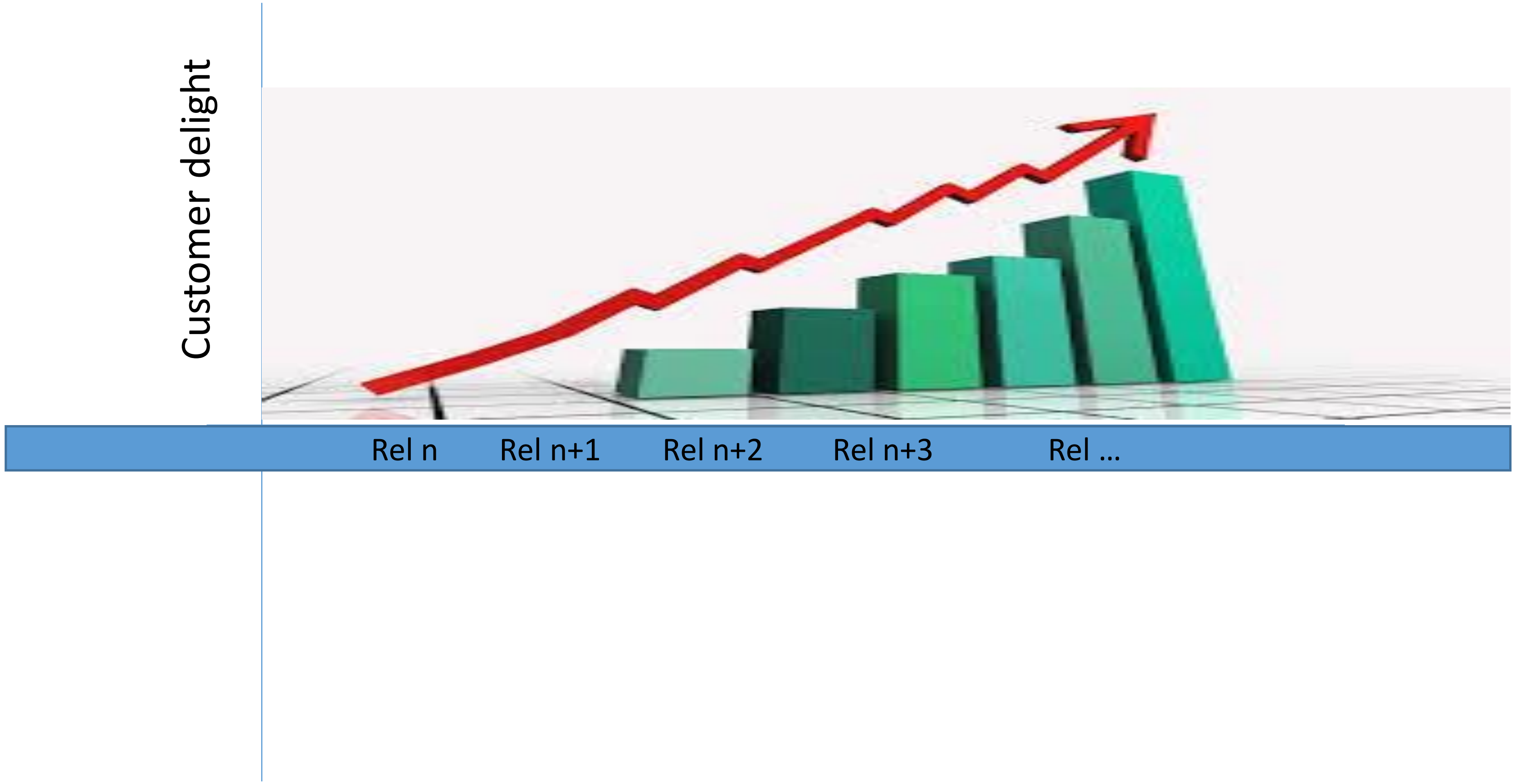
Content Management

Change Management

Process quality

- Consistency
- Predictability
- Repeatability

Consistency of production releases



Predictability

- Is this doable?
- Can I do it?
- When can I deliver it, and at what cost?

Repeatability

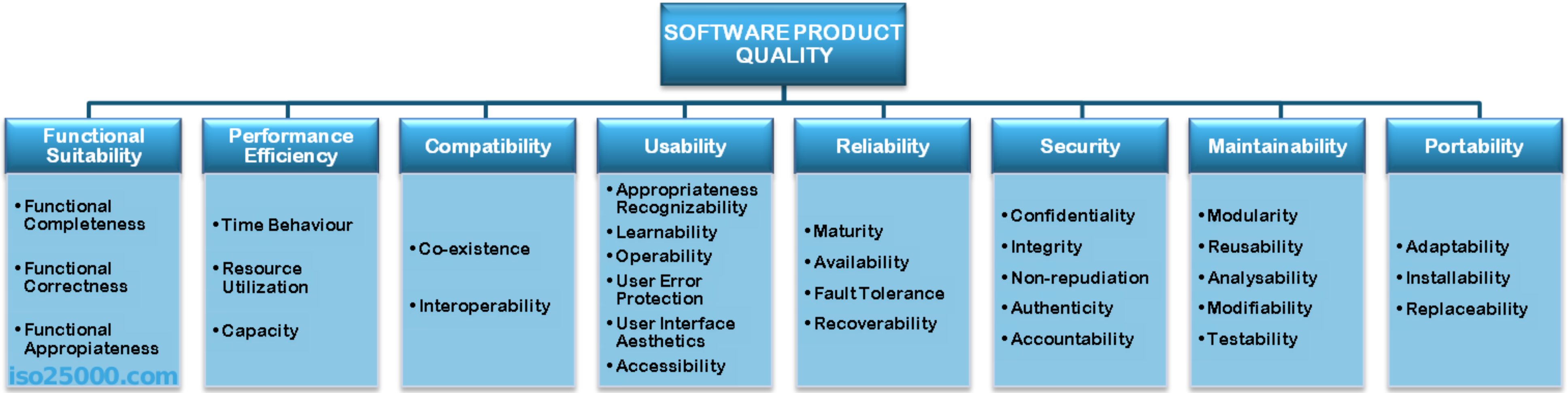
- Repeat what works
- Avoid what doesn't work
- Learn and move on

A few quality models

- ISO 25010
- SEI CMMI
- EFQM
- RUP

Software quality as defined by ISO

- ISO 25010 (2011), a way to 'measure software quality':



SOFTWARE PRODUCT QUALITY

Functional Suitability
- Functional Completeness
- Functional Correctness
- Functional Appropiateness

Performance Efficiency
- Time Behaviour
- Resource Utilization
- Capacity

Compatibility
- Co-existence
- Interoperability

Usability
- Appropriateness Recognizability
- Learnability
- Operability
- User Error Protection
- User Interface Aesthetics
- Accessibility

Reliability
- Maturity
- Availability
- Fault Tolerance
- Recoverability

Security
- Confidentiality
- Integrity
- Non-repudiation
- Authenticity
- Accountability

Maintainability
- Modularity
- Reusability
- Analysability
- Modifiability
- Testability

Portability
- Adaptability
- Installability
- Replaceability

iso25000.com

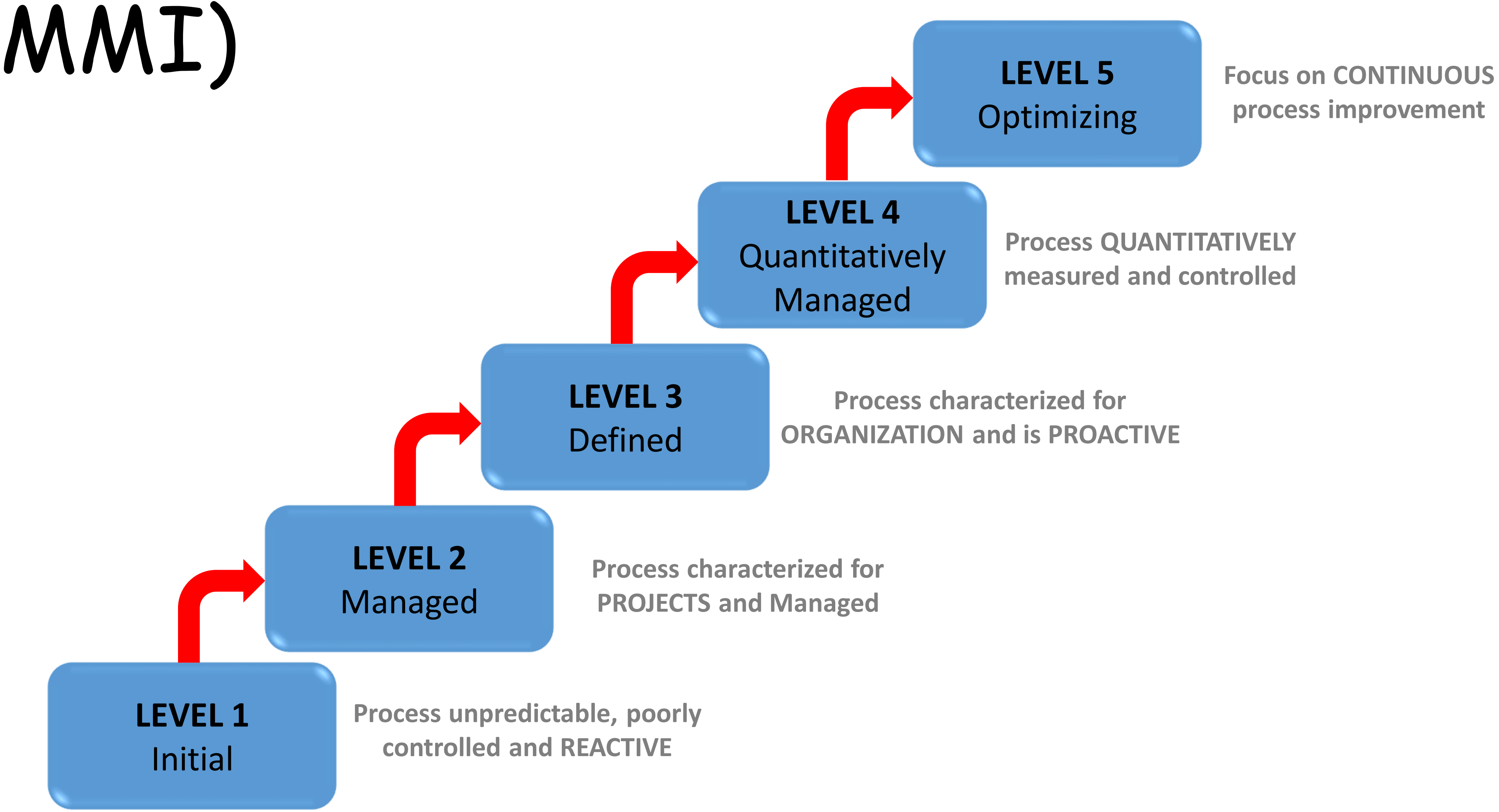
SEI's Capability Maturity Model (CMMI)

- CMMI does not provide a single process. Rather, the CMMI framework models what to do to improve your processes, not define your processes
- CMMI is designed to compare an organization's existing processes to proven best practices developed by members of industry, government, and academia
- Reveals possible areas for improvement; and provide ways to measure progress

RUP

- Rational Unified Process

SEI's Capability Maturity Model (CMMI)



LEVEL 5
Optimizing

Focus on CONTINUOUS process improvement

LEVEL 4
Quantitatively Managed

Process QUANTITATIVELY measured and controlled

LEVEL 3
Defined

Process characterized for ORGANIZATION and is PROACTIVE

LEVEL 2
Managed

Process characterized for PROJECTS and Managed

LEVEL 1
Initial

Process unpredictable, poorly controlled and REACTIVE

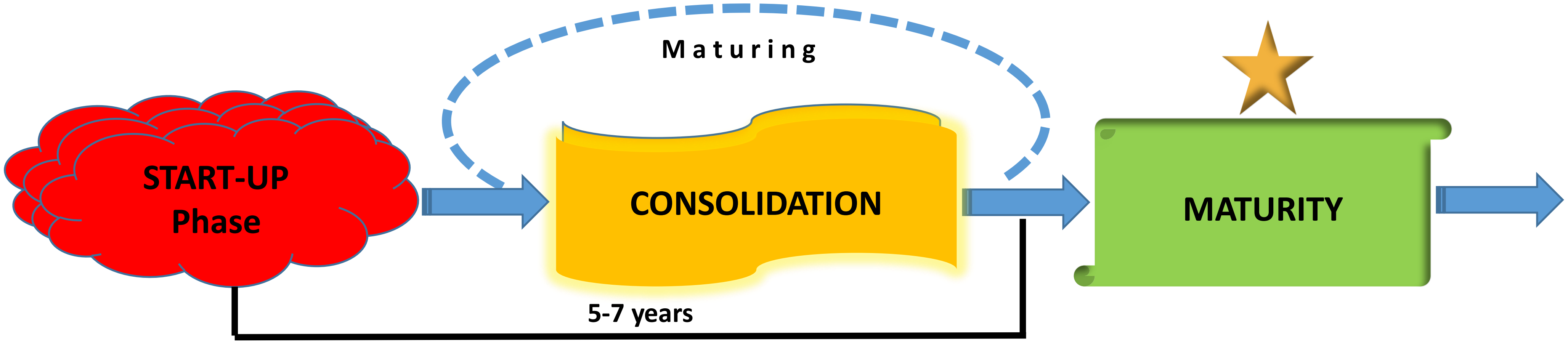
Quality Matters …

- What is this ‘thing’ called software quality? Why should you care?

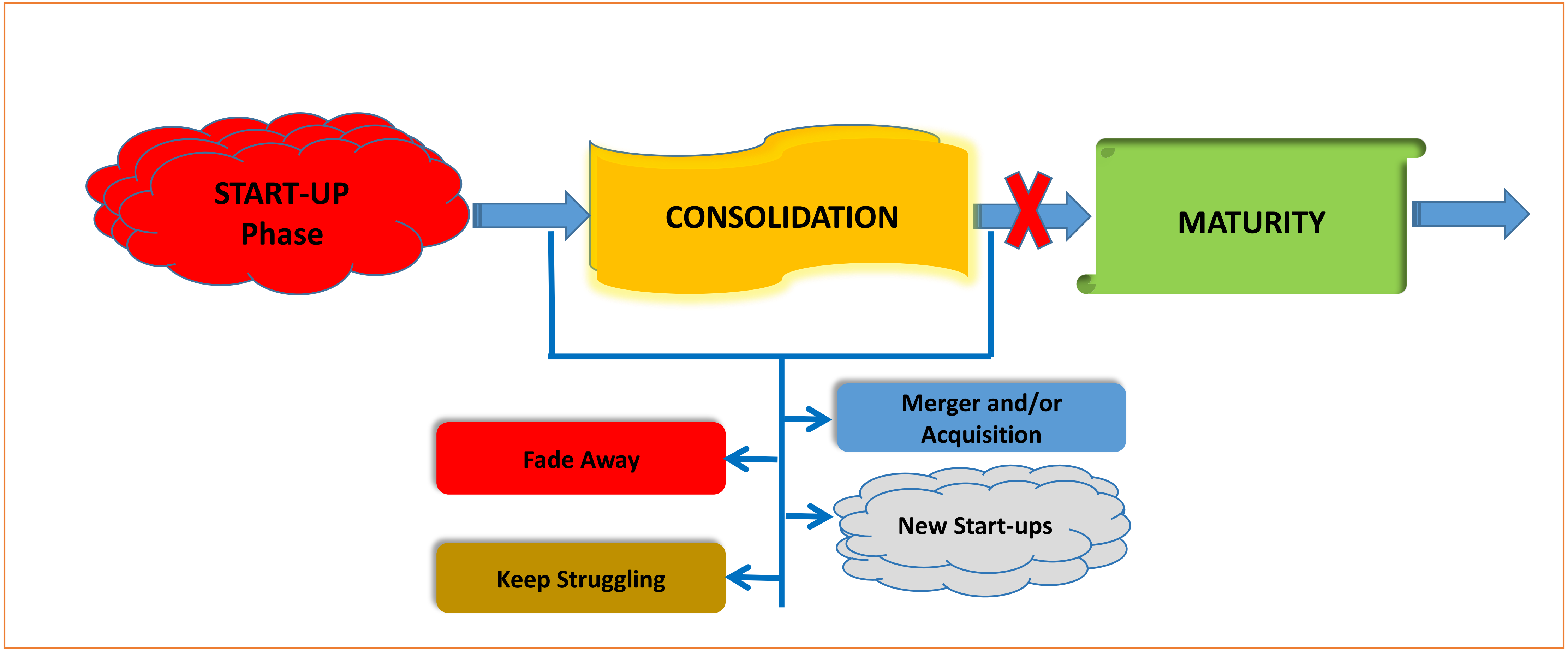
- How do we build quality ‘into’ software?

✓ **What has quality got to do with the survival of your company?**

Software company lifespan
The silicon Valley model



Software company lifespan
The Silicon Valley experience



START-UP Phase

CONSOLIDATION

MATURITY

Merger and/or Acquisition

Fade Away

New Start-ups

Keep Struggling

Hallmark of a mature software company

Consistency

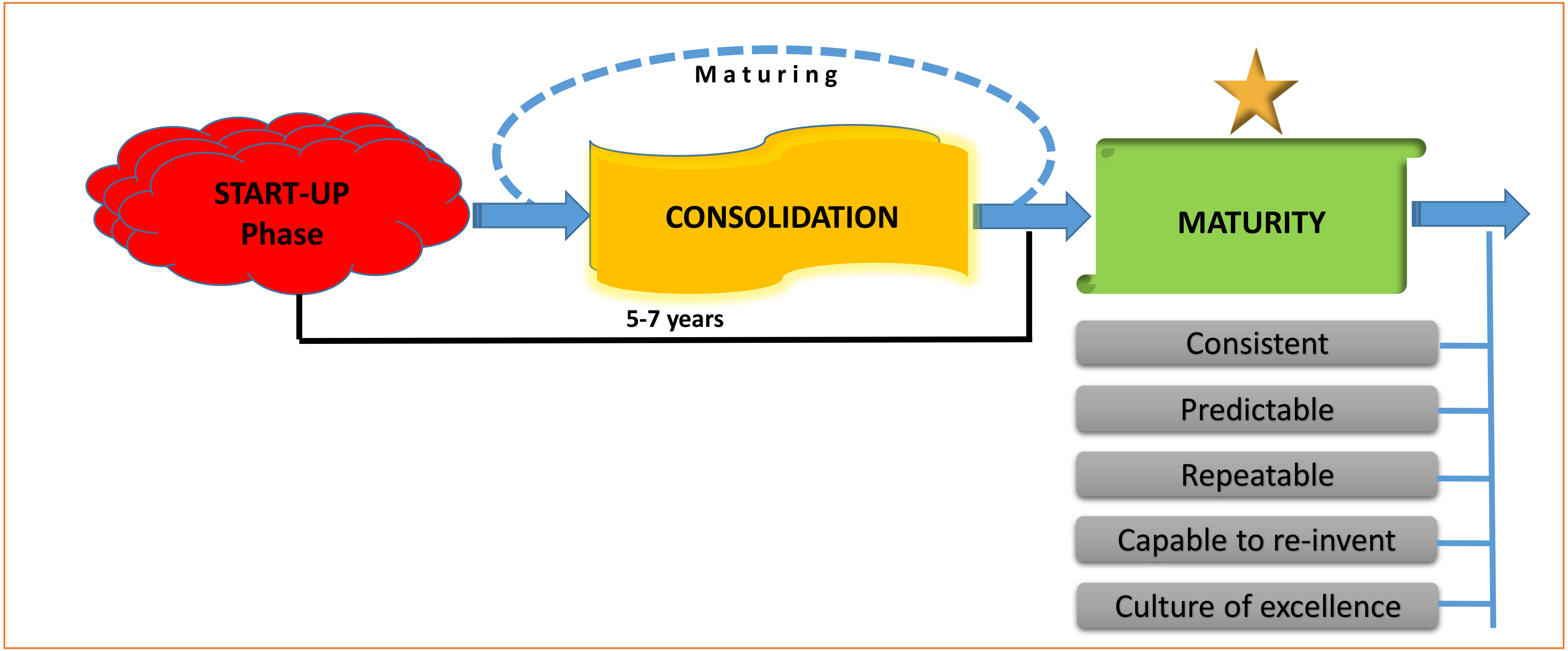
Entrenched Culture of Excellence

Predictability

Capable to re-invent

Repeatability

Software company lifespan



START-UP Phase

Maturing

CONSOLIDATION

5-7 years

MATURITY

Consistent

Predictable

Repeatable

Capable to re-invent

Culture of excellence

Capable to re-invent

- Recognize ‘the writing on the wall’ (end of the road, existential threats)
- Able to ‘peel’
- Radically change direction of business
- Take audacious risks
- Manage exceedingly tough transition

Capable to re-invent

IBM
AMERICAN EXPRESS
WESTERN UNION
moving money for better
Nintendo
NOKIA
NATIONAL GEOGRAPHIC
Shell

Culture of excellence

- Andy Grove: “only the paranoid survive”
- I say: “only the best survive” So, don’t be a mediocre, don’t be content with the ordinary, strive for excellence

Culture of excellence

- Customer service
- Employees
- Quality as a stratgey

Culture of excellence
Delightful user experience, every time

- Products and services that ***meet***

- Products and services that ***delight*** our customers

★ ★ ★ ★ ★

Culture of excellence

Exceptional, extraordinary employees

- Most talented
- Most competitive
- Most creative

SURVIVE

Culture of excellence

Quality is a strategic business imperative

- A top corporate priority
- Competitive advantage
- Survival strategy

Culture of excellence

Practically speaking …

- Quality is everyone’s responsibility
- Poor quality work (by anyone, at anytime, in any shape or form) is disallowed
- Really, really, really try to get things done right the first time around

Quality Matters …

- ✓What is this ‘thing’ called quality? Why should you care?
- ✓How do we build quality ‘into’ software?
- ✓What has quality got to do with the survival of your company?

Last word …

Remember:

Quality is not a goal,

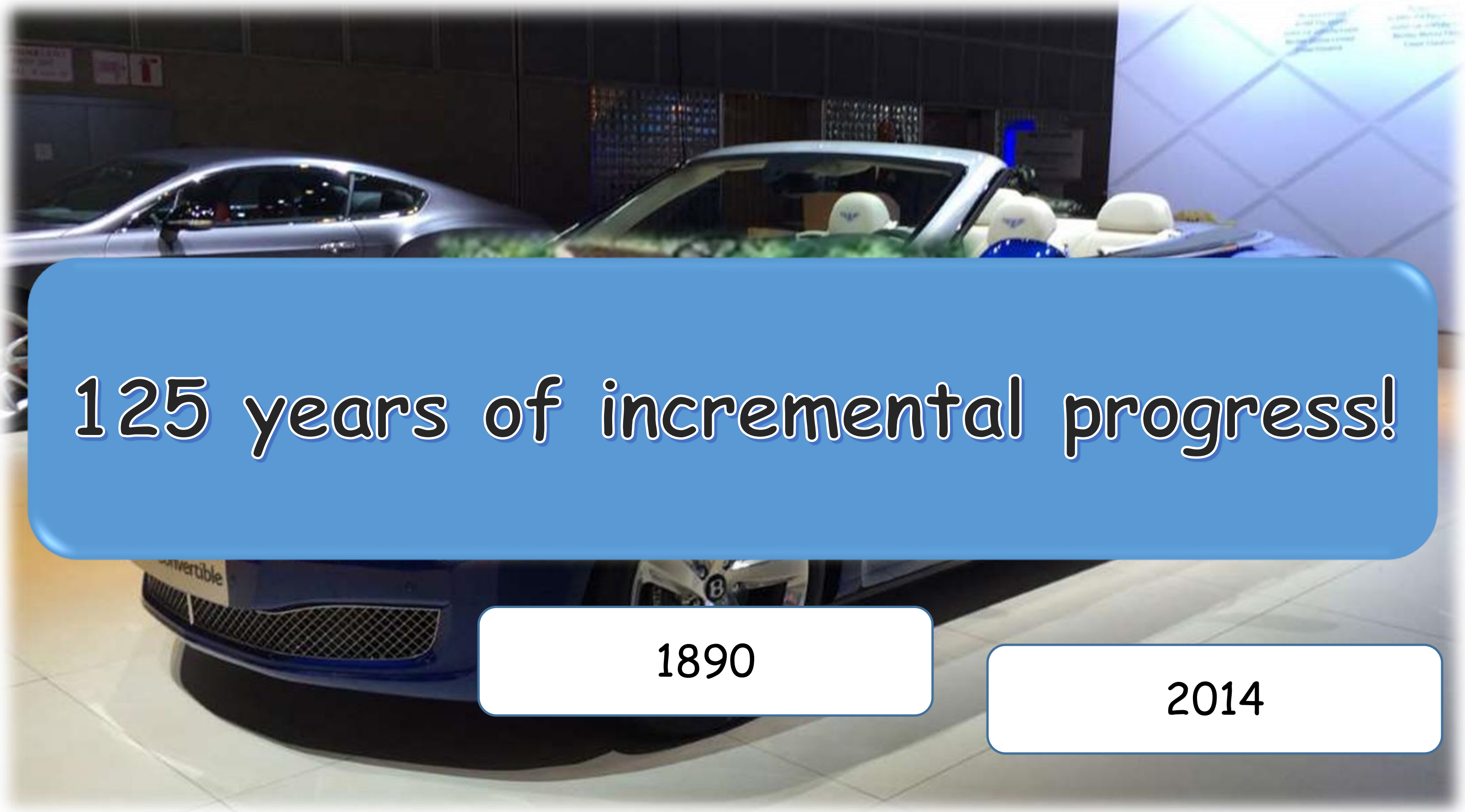
it is a strategy, tactfully implemented,

it is small increments of progress

to help us get better at what we do,

and then, serve our customers better …

This is what I mean …



125 years of incremental progress!

1890

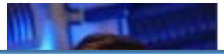
2014

The story of Silicon Valley (in 1 slide)

And why it's so hard to replicate …

Dave Hitz

Co-Founder & Chief Strategist, NetApp



What do you want to do?

I want to change the world.