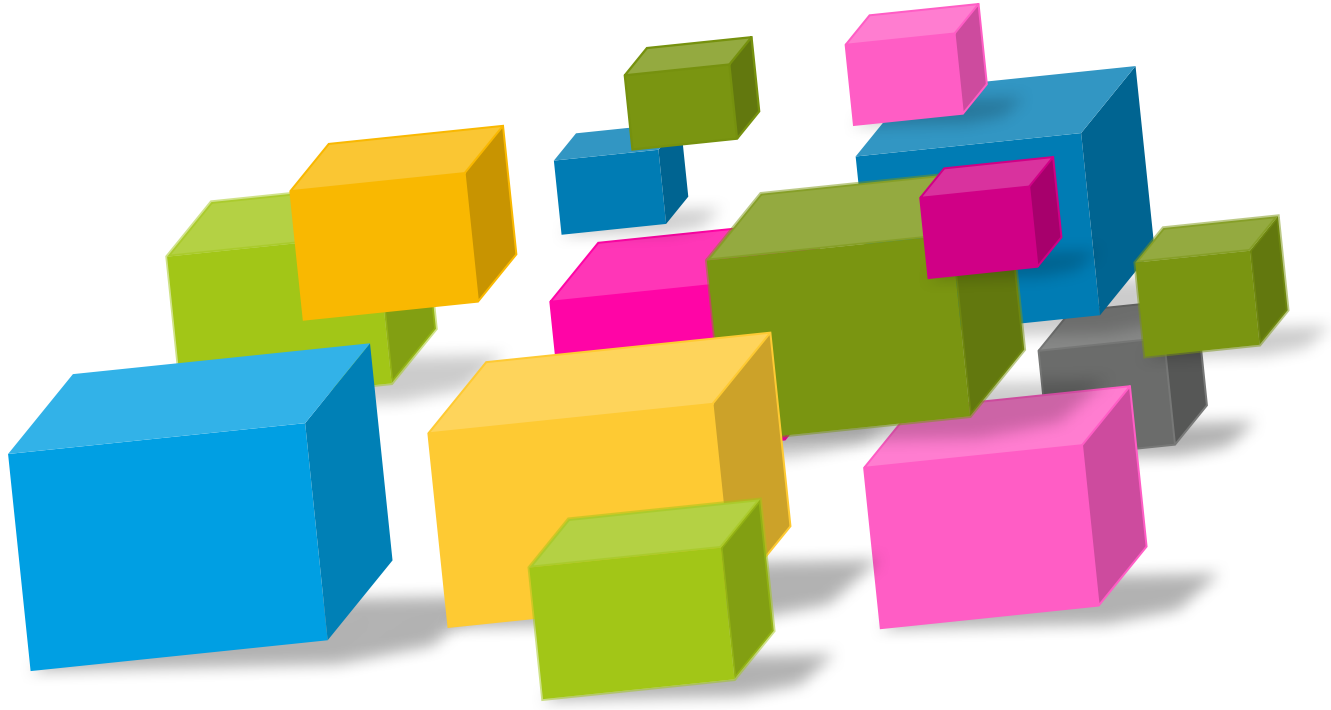
Architecture and NFRs

Kim Horn

Version 0.58 2/8/2016



Contents

- Introduction
- Terminology
- Non Functional Requirements (NFRs) are the Drivers for Architecture
- Results of Quality Survey Review – an nbn set of Qualities.
- The NFR Scenario Template
- Service Example Scenarios
- Questions to prompt NFR thinking
- Using A Quality Tree to Trade Off and Balance
- Systemic Qualities and Constraints
- Future Work

The slides include a lot of examples so its not so big a pack,
Skip those examples and come back later….

NFR Definitions

Introduction

This pack briefly introduces the work done by the Architecture Methods* Team in improving the way we capture and utilise Non-Functional Requirements.

- An extensive review of of Non-Functional Requirements (NFRs) material was done and a pragmatic lean base set adopted as a Version 1.
  - Selected to improve new initiatives such as 24*7, Scalability and Stability (e.g. ITSS) - to improve the quality of our quality;
  - These NFRs and types drive the **nbn** architectural methods and form a core philosophy behind architecure;
  - They start an iterative process through solution options, architectural decisions, tradeoffs, balance and estimates;
- A set of process measures is being developed to allow continuous improvement of these Qualities and Methods.
- **See associated pack on "Business Goals and NFRs"**

* Methods are 'Ways of Working – 'processes'

Often Ambiguity Around Qualities

- Functional requirements describe ‘what’ the users want, and value, but not ‘how well’ the functions perform.
- Non Functional Requirements, the Systemic Qualities, capture the ‘how well’.

Terminology: IIBA Definitions

The International Institute of Business Analysis (IIBA) defines **Functional requirements** as "the product capabilities, or things that a product must do for its users."

- Functional requirements define how software behaves to meet user needs.

The IIBA defines **Non Functional Requirements** (NFRs) as "the quality attributes, design and implementation constraints, and external interfaces which a product must have."

- Quality attributes are often affectionately called the "ilities" because the names of many of them end in "ility."

Terminology: BABOK Definitions

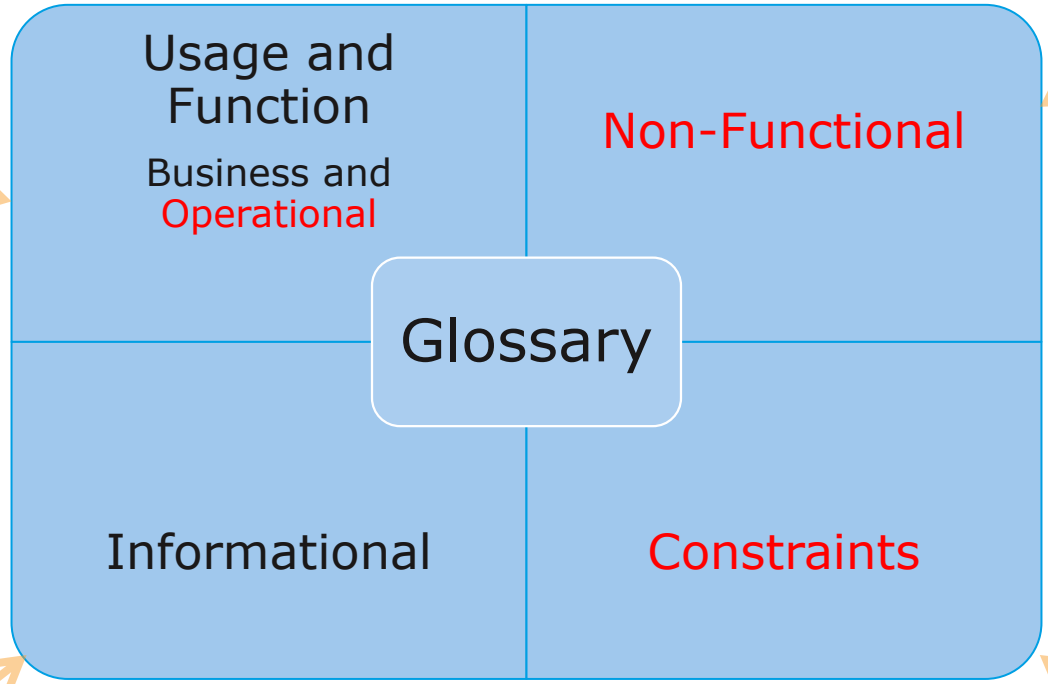
Solution Requirements describe the characteristics of a solution that meet business requirements and stakeholder requirements. They are developed and defined through *requirements analysis*. They are frequently divided into sub-categories, particularly when the requirements describe a software solution:

- **Functional Requirements** describe the behavior and information that the solution will manage. They describe capabilities the system will be able to perform in terms of behaviors or operations---specific information technology application actions or responses.

- **Non-functional Requirements** capture conditions that do not directly relate to the behavior or functionality of the solution, but rather describe environmental conditions under which the solution must remain effective or qualities that the systems must have. They are also known as quality or supplementary requirements. These can include requirements related to capacity, speed, security, availability and the information architecture and presentation of the user interface.

Requirements Types

- Actors, Roles, Personas
- Uses Cases/Scenarios
- User Scenarios
- Journeys
- Epics, Themes, User Stories
- Features
- Each has non functional aspects



- Manifest
- Operational
- Developmental
- Evolutionary

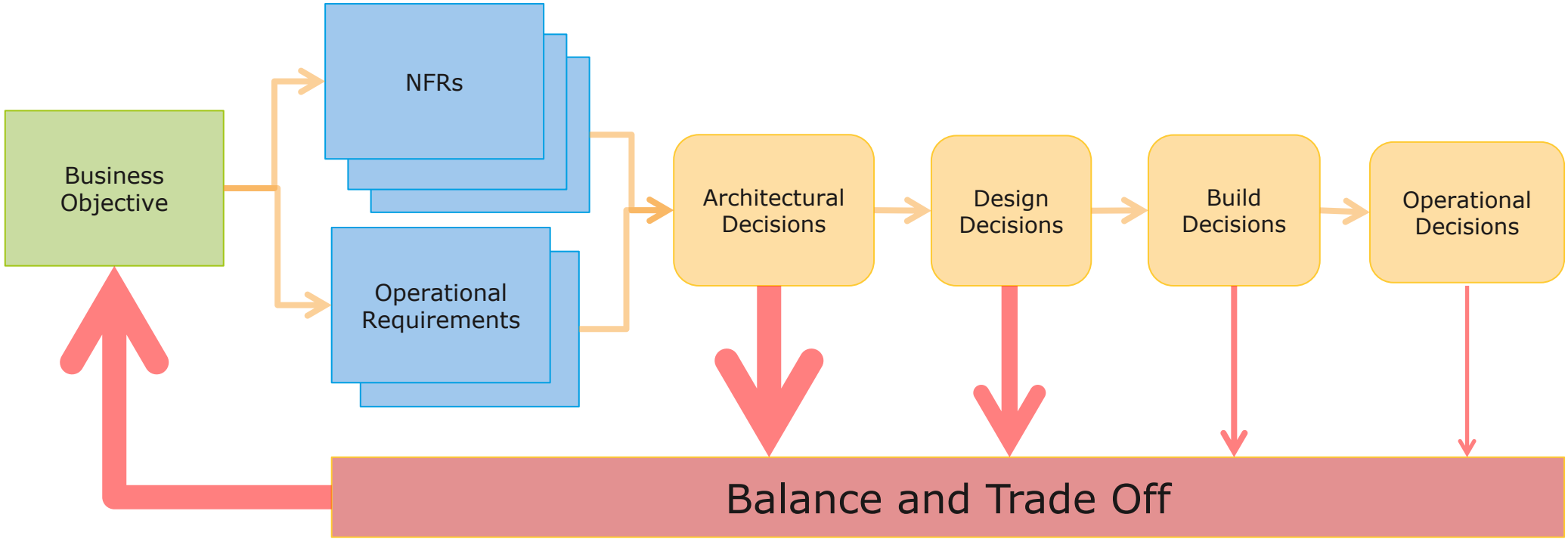
- Entity / Domain Models
- Business Rules

- Development process & team
- Environment & technology
- Delivery & Development

Non-Functional and Operational Traceability

Key is traceability from Business Objectives to the Non-Functional and Functional Operational Requirements and then across the decisions made down to operations.

- Architectural Decisions should catch the main concerns, but then decisions and issues down the path, in design, build and operations also have impacts.
- Balance and Trade off of Requirements is required, that has an impact on the business objectives.
- It is a cyclic E2E feedback process.



What are SLRs and Systemic Quality (SQs) ?

Service Level Requirement (SLR):

- A requirement that specifies desired ‘service’ conditions on a functional requirement at or over a period of time.
- Often a broad statement from a customer to a service provider.
- They are orthogonal to the functional requirements.
- Provide the systemic qualities for a system.

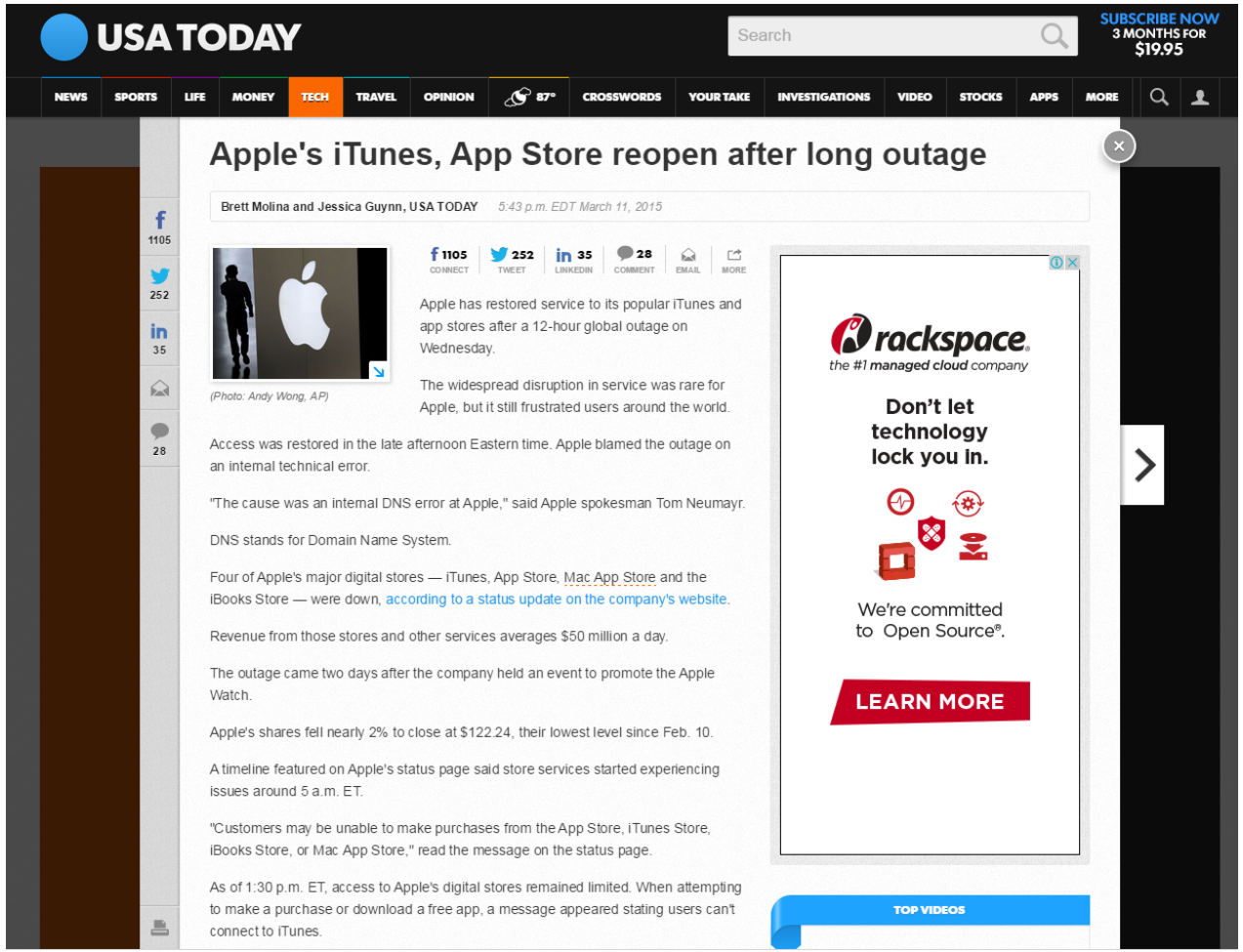
Systemic Quality(SQ):

- A measurable, observable, testable and generally broad characteristic of a system.
- Describe a goal to be achieved. Always relate back to a business goal.
- Criteria that can be used to judge the operation of a system.
- Are Systemic; Broader in scope than a single story, use case, feature, a single stakeholder, or a single point in the lifecycle of the system.
- Are pervasive and cross-cutting over other requirements.
- Rarely independent, with complex interactions.

Qualities and Customer Value

- Unlike Functional requirements, **Architecture and Operational issues are often communicated as having no (or little) externally visible "customer value"** and often seen as a project cost.
- NFRs become 'Invisible Features', and are often forgotten.
- However, visible "customer (or user) value" is not enough to deliver a 'quality' solution.
- Given only Functional Requirements the result is solutions that may not meet the current and future needs of the organisation, e.g. are not always available, too hard to use, can't be operated, do not scale to meet new customer demands, or are unreliable.
- Although not directly impacting the customer, these invisible features end up impacting the customer indirectly, but of more concern is that they will impact the organisation, its strategy, and its reputation;
- **In hindsight, after their impact is felt, organisations find them to be the most important requirements of a system.**

In 2015 Well known ‘Quality’ companies ignored NFRs



USA TODAY

Search

SUBSCRIBE NOW 3 MONTHS FOR $19.95

NEWS | SPORTS | LIFE | MONEY | TECH | TRAVEL | OPINION | 87° | CROSSWORDS | YOUR TAKE | INVESTIGATIONS | VIDEO | STOCKS | APPS | MORE

Apple's iTunes, App Store reopen after long outage

Brett Molina and Jessica Guynn, USA TODAY 5:43 p.m. EDT March 11, 2015

(Photo: Andy Wong, AP)

Apple has restored service to its popular iTunes and app stores after a 12-hour global outage on Wednesday.

The widespread disruption in service was rare for Apple, but it still frustrated users around the world.

Access was restored in the late afternoon Eastern time. Apple blamed the outage on an internal technical error.

"The cause was an internal DNS error at Apple," said Apple spokesman Tom Neumayr.

DNS stands for Domain Name System.

Four of Apple's major digital stores — iTunes, App Store, Mac App Store and the iBooks Store — were down, according to a status update on the company's website.

Revenue from those stores and other services averages $50 million a day.

The outage came two days after the company held an event to promote the Apple Watch.

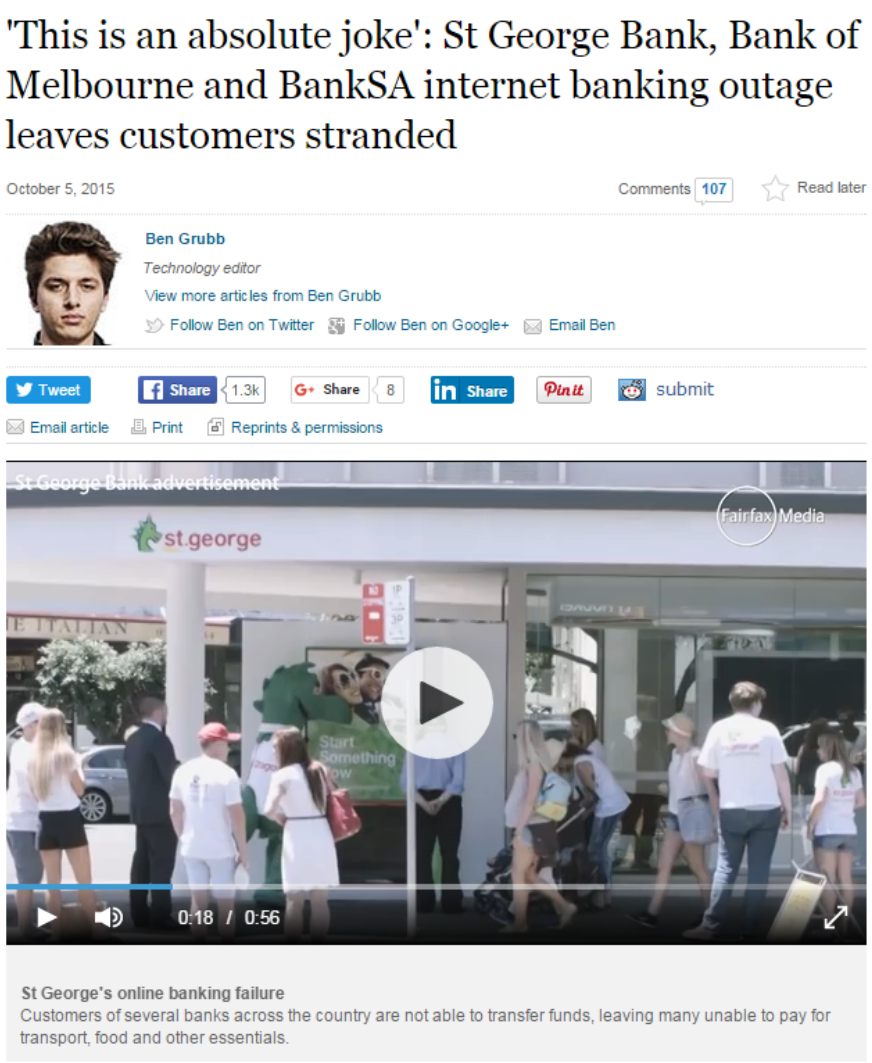
Apple's shares fell nearly 2% to close at $122.24, their lowest level since Feb. 10.

A timeline featured on Apple's status page said store services started experiencing issues around 5 a.m. ET.

"Customers may be unable to make purchases from the App Store, iTunes Store, iBooks Store, or Mac App Store," read the message on the status page.

As of 1:30 p.m. ET, access to Apple's digital stores remained limited. When attempting to make a purchase or download a free app, a message appeared stating users can't connect to iTunes.

TOP VIDEOS



'This is an absolute joke': St George Bank, Bank of Melbourne and BankSA internet banking outage leaves customers stranded

October 5, 2015 Comments 107 Read later

Ben Grubb
Technology editor
View more articles from Ben Grubb
Follow Ben on Twitter Follow Ben on Google+ Email Ben

Tweet Share 1.3k Share 8 Share Pin it submit
Email article Print Reprints & permissions

St George Bank advertisement
st.george
Fairfax Media
0:18 / 0:56

St George's online banking failure
Customers of several banks across the country are not able to transfer funds, leaving many unable to pay for transport, food and other essentials.

2016 known ‘Quality’ companies ignored NFRs

JetBlue planes sit at their gates at John F. Kennedy Airport in April 2014 in Queens, New York City. (Photo by Andrew Burton/Getty Images)

DOWNTIME, VERIZON

Verizon Data Center Outage Delays JetBlue Flights

BY YEVGENIY SVERDLIK ON JANUARY 14, 2016 1 COMMENT

67 Like | Tweet | 93 Share | 16 G+1

A Verizon data center outage Thursday morning brought down JetBlue’s electronic systems, causing flight delays and shutting down the airline’s website, along with its online booking and check-in systems.

Systems outage delays Virgin flights

THE AUSTRALIAN | FEBRUARY 2, 2016 10:10AM SAVE

Mitchell Bingemann
Reporter
Sydney

Follow @Mitch_Hell

Virgin Australia flights were thrown into chaos this morning as a systems outage left passengers unable to check-in. Photo: Marc McCormack.

Virgin Australia flights were thrown into chaos this morning as a systems outage left hundreds of passengers unable to check-in for domestic flights.

The outage — which affected the airline’s Sabre check-in system — affected passengers trying to check-in at most domestic airports including Sydney, Melbourne and Adelaide.

Poor Quality - Seems to be a ~~Weekly~~ Daily Event – 10 Feb 2016

The Sydney Morning Herald
Digital Life

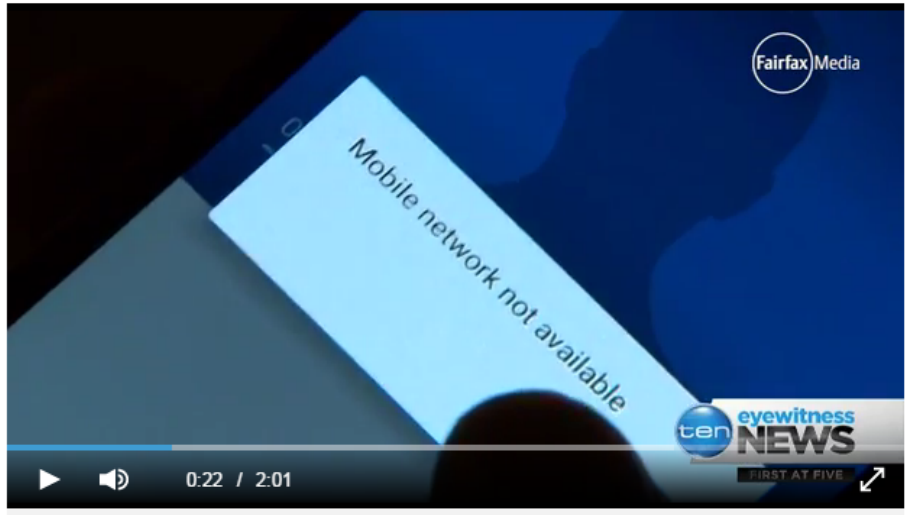
Home Tech | Car Tech | Cameras | MP3s | Mobiles | Computers | Apps | Consumer Security | Games

You are here: Home » Digital Life » Mobiles »

Telstra outage: manager connected customers to faulty node in 'embarrassing error'

February 10, 2016 — 19 reading now — Read later

Kate Aubusson



Telstra network crashes nationwide
"Human error" causes widespread network interruptions affecting millions of Telstra customers across the country.

news.com.au

National | World | Lifestyle | Travel | Entertainment | Technology | Finance | Sport | Video

Telstra hit by second outage in two days

FEBRUARY 11, 2016 11:42AM — 8

Telstra customers back online after outage

Network writer, News Corp Australia Network

THINGS went from bad to worse for Telstra last night after some of its broadband users were unable to access hundreds of websites.

And again … 17/3/2016

MERCURY
themercury.com.au

Telstra offers free data after massive mobile and internet outage

March 19, 2016 12:00am
HELEN KEMPTON and NICK CLARK Mercury

A NATIONWIDE Telstra outage has added salt to the wounds of Tasmania's long-suffering internet users as those not affected by the cutting of the Basslink cable were also thrown offline.

Telstra yesterday said April 3 would be a "free data" day for customers as compensation for the outage.

Seventy thousand Tasmanian Internode and iiNet customers have been dealing with slow internet speeds since last Friday due to TPG's failure to secure enough bandwidth from Telstra to keep its customers connected after the cable was cut.

NEWS

Just In | Australia | World | Business | Sport | Analysis & Opinion | Fact Check | Programs

BREAKING NEWS Budget brought forward to May 3

Print | Email | Facebook | Twitter | More

Telstra outages: CEO Andy Penn offers free data day as compensation for four-hour blackout

Updated Fri at 12:37pm

Telstra has apologised for a mass network outage that affected up to 8 million customers, vowing to provide a free data day on April 3 as compensation.

Complaints from customers having trouble accessing their mobile network poured in from across the country on Thursday from about 6:00pm.

Calls, texts and internet services were affected for up to four hours for some customers, while the majority were reconnected within about 90 minutes, Telstra said.

Telstra chief executive officer Andy Penn said the outage was caused by extreme network congestion when a large number of services disconnected at the same time and were then reconnected.

The root of the outage was sourced back to an international roaming problem that had flow-on effects in Australia, Mr Penn said.

VIDEO: Telstra boss promises free data day as widespread outages hamper network (ABC News)

PHOTO: Telstra will offer a free data day on April 3 to compensate for the mass outage. (720 ABC Perth: Emma Wynne)

RELATED STORY: Telstra gave users a free data day, so they downloaded 1,841 terabytes

RELATED STORY: Telstra 'progressively restoring' mobile services after outage

MAP: Australia

And again… 20/5/2016

The Sydney Morning Herald
BusinessDay

Telstra outages hit NBN and ADSL services across Australia

May 20, 2016 - 3:27PM

600 reading now Comments 107 Read later

Patrick Hatch

Reporter for The Age

View more articles from Patrick Hatch

Follow Patrick on Twitter Follow Patrick on Google+ Email Patrick

Telstra was hit by another service outage on Friday. *Photo: Louie Douvis*

Next Month Again… 11/6/2016

The Sydney Morning Herald
BusinessDay

Telstra customers hit by another major outage

June 11, 2016 - 5:18PM

Emily Woods

Telstra hit by another service outage. *Photo: Louie Douvis*

Telstra customers are experiencing widespread network outages across the country.

Internet ADSL and NBN services are down in some areas of NSW, Victoria, Queensland, WA, SA and Tasmania.

Irritated Telstra customers have taken to Twitter, and the hashtag #telstraoutage is trending.

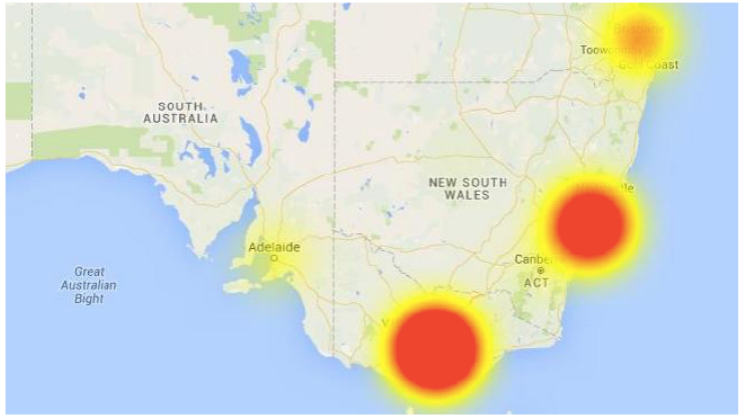
And again …
30/6/2016

news.com.au

online

Another Telstra outage just days after $250m announcement leaves customers furious

JUNE 30, 2016 4:19PM



Aussie Outages showing problems in Victoria, Queensland and New South Wales.

Matthew Dunn, news.com.au @mattydunn11

IT'S another day and another major outage for Telstra customers.

This time customers of the Aussie telco from Victoria are the ones feeling the pinch.

At present, Reece Plumbing, Medibank and Monash University are among the organisations complaining of the outage on Twitter.

RIGHT NOW I

208 READERS The major cap sinking Beijing

49 READERS Australia claw internet speed

57 READERS Mitsubishi rec and Toyota re

47 READERS Audi electric c will blow Tesla

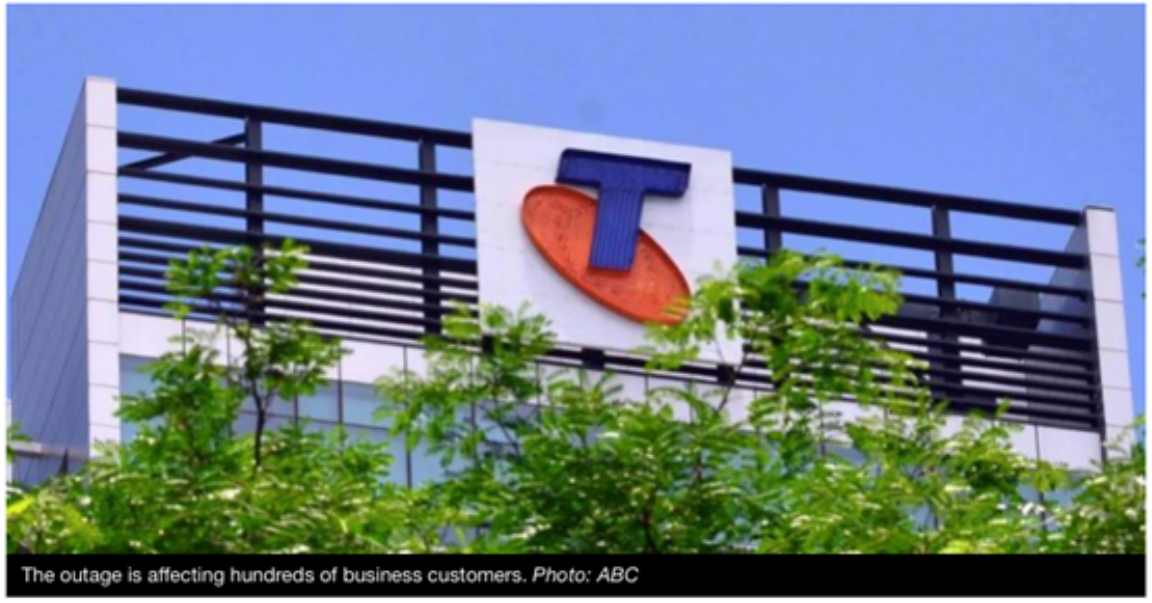
33 Heterosexual Internet outrac

EOFY nightmare: businesses hit by Telstra outage

2:37pm, Jun 30, 2016

NEIL FRANKLAND PM News Editor (EDM)

NSW and Victorian customers thrown into chaos on the last day of financial year.



The outage is affecting hundreds of business customers. *Photo: ABC*

Business across NSW and Victoria were thrown into chaos on the last day of the financial year Thursday as Telstra was hit by yet another network outage.

Major retailer including Officeworks, Myer, The Good Guys and Medibank were all reportedly hit, along with Jetstar airline, ME Bank, Monash University and the Royal District Nursing Service (RDNS).

Another Telco Twice, and Amazon

BT broadband users hit by second UK-wide outage in two days

Caroline Donnelly
Datacentre Editor
21 Jul 2016 9:50

Power supply issues at Docklands datacentre could be behind loss of internet access for more than 5,500 BT broadband users

CW ANZ:
Is Austr
securit
How the

AWS's S3 outage was so bad Amazon couldn't get into its own dashboard to warn the world

Websites, apps, security cams, IoT gear knackered

By Shaun Nichols in San Francisco 1 Mar 2017 at 03:00 122 SHARE ▼

Tuesday's Amazon Web Services mega-outage knocked offline not only websites big and small, by yanking away their backend storage, but also knackered apps and Internet of Things gadgets relying on the technology.

In fact, the five-hour breakdown was so bad, Amazon couldn't even update its own AWS status dashboard: its red warning icons were stranded, hosted on the broken-down side of the cloud.

What makes users really unhappy? Poor Quality

Did you ever realize that when a user is unhappy it's typically not because of what a function does?

Usually users make assumptions (usually undocumented) about how well those functions have to perform, and when they don't, they get very unhappy.

Functionality problems can be dealt with easily, by including the resolution in the next release; this makes the user happy again. But when they are really unhappy it is because some of the quality attribute requirements don't work.

Its not available, reliable, its not fast enough……..

This is what makes the user unhappy. **Qualities are actually much more important than the function itself.**

But they are often treated as second class citizens or not captured at all ?

Understand NFRs Early

"While we must acknowledge emergence in design and system development, a little planning can avoid much waste."

James Coplien.

- **Refactoring Architecture**, in most cases, requires expensive and time consuming re-builds. It is often too big a problem for traditional developmental iterative 'refactoring' or Continuous Integration/Delivery approaches.
  - Indeed, if the system can be easily refactored to solve the problem, then the problem is not architectural.
- **Refactoring Organisations**, e.g. adding new or changed operational procedures requires expensive and time consuming organisational change.
- Therefore, some upfront planning is required, and this needs to be driven by high quality NFRs.
- Tackling these early on, can reduce development and operational costs, provide quality sooner and provide more accurate estimations.
- To do this, the full and realistic business impact of NFRs needs to be understood by stakeholders, the 'value' appreciated; that they are driven by business needs. Their extent needs to be understood to ensure quality is well supported.

NFRs

4 Main Types of Systemic Qualities:

Type of Quality	Description
Manifest	Support end user needs. In most cases they are objectively measurable and have impact at specific release points.
Operational	Impacts those stakeholders that run, operate, monitor or manage the system.
Developmental	Describes desirable qualities of the system from the standpoint of those who will build it.
Evolutionary	Describes future goals of the system to meet the organisation's strategy.

Each type impacts the architecture, across its tiers and layers,
Now and into the future.

4 Types of Systemic Qualities by Who, When and How

Type Of Quality	Who	When	How
Manifest	End user and business stakeholders	After initial release, production.	Test, customer satisfaction, business objectives, analytics.
Operational	IT Operations, IT Service Management, IT Security, Business Security, Maintenance and Support.	Development operations, release-time, after release	Test, customer satisfaction, operational metrics.
Developmental	Developers, Dev-Ops, PMs, Architects.	During and throughout development and delivery.	Productivity metrics, Velocity, delivery frequency.
Evolutionary	Developers, business stakeholders	After initial release	Strategy and Business Performance

Specific List of 30 Qualities by Type - Version 1.0*

Manifest	Performance
	Availability
	Usability
	Reliability
	Accessibility
	Mobility
Developmental	Buildability
	Testability
	Understandability
	Code Quality Measurability
	Conceptual Integrity
	Budgetability
	Planability
	Traceability
	Development Distributability
	Modularity / Packagability

Operational	Throughput
	Security
	Manageability
	Maintainability
	Serviceability
	Deployability
	Reproducibility
Evolutionary	Scalability / Elasticity
	Variability
	Flexibility
	Extensibility
	Reusability
	Portability
	Interoperability

*Definitions in separate document

Quality Definitions– Single Liners

- Usability = Is Easy to use (Note, meeting the users goals is a functional question)
- Accessibility = Is usable by everyone
- Reliability = Doesn't Break
- Performance = Provides expected responsiveness
- Throughput = Handles Volumes of work
- Availability = works even when it breaks
- Scalability = Easily incorporates added capacity
- Flexibility = Easily incorporates new or modified services
- Security = Denies Intrusion and Damage
- Manageability = Is organised and controllable
- Maintainability = Is easy to maintain
- Serviceability = Is easy to repair
- Interoperabilty = Components work together
- Portability = Components can be moved from environment to environment
- Reusability = Components can be used in additional services
- Buildability = the application can be built on time
- Planability = the work can be scheduled across teams

NFRs are the 3rd Dimension of Architecture.

If one dimension of a systems architecture is its

Tiers

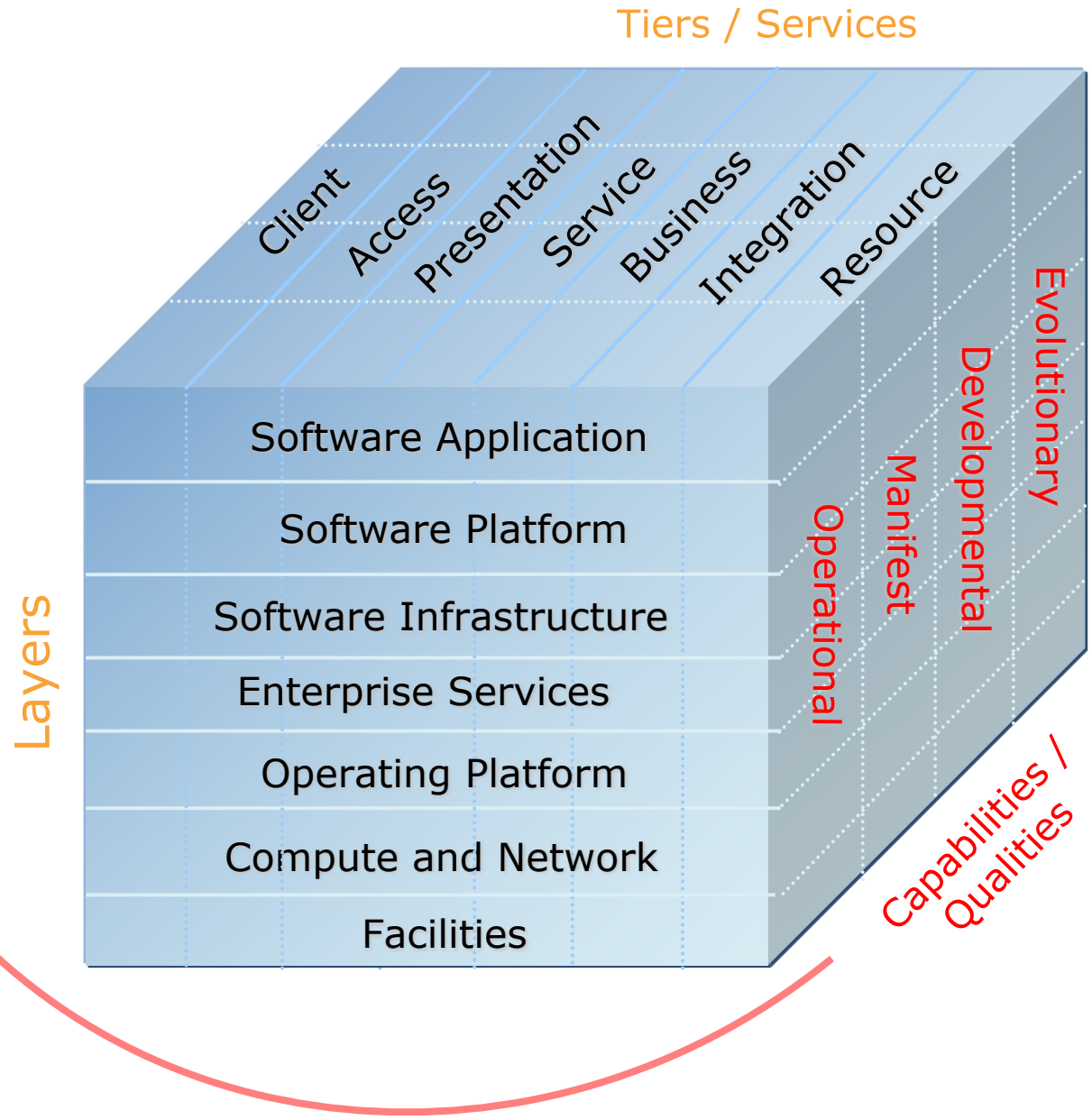
and another its

Layers

the third is the

Qualities

The 4 types define the Quality of Architecture.



Example: What is needed to capture a Systemic Quality Need ?

A Performance Example, Internet startup, with limited initial funds releasing new product. Market research suggest users want <1 second response times 24*7, but, business needs to demonstrate initial adoption before spending more.

A poorly captured statement:

"Response times will average 5 seconds or less"

But 10 responses of 20,20,5,0.1,0.5,0.2,0.2,1,2,1 seconds meet this !!!!

Improvements for a better goal statement:

- Provide a usage profile and time characterization to make the goal more specific and meaningful.
- Make it specific, measureable and testable.
- Provide traceability to a business goal or driver.
- Describe evolutionary (future) needs.

Better Statement:

"Business owner requires response times for web transactions to take no longer than 5 seconds between the hours of 9am and 5pm on a weekday. It is essential to provide a good experience to customers so they do not go to a competitor."

Example: Needs now and into the Future – Things Evolve

Better:

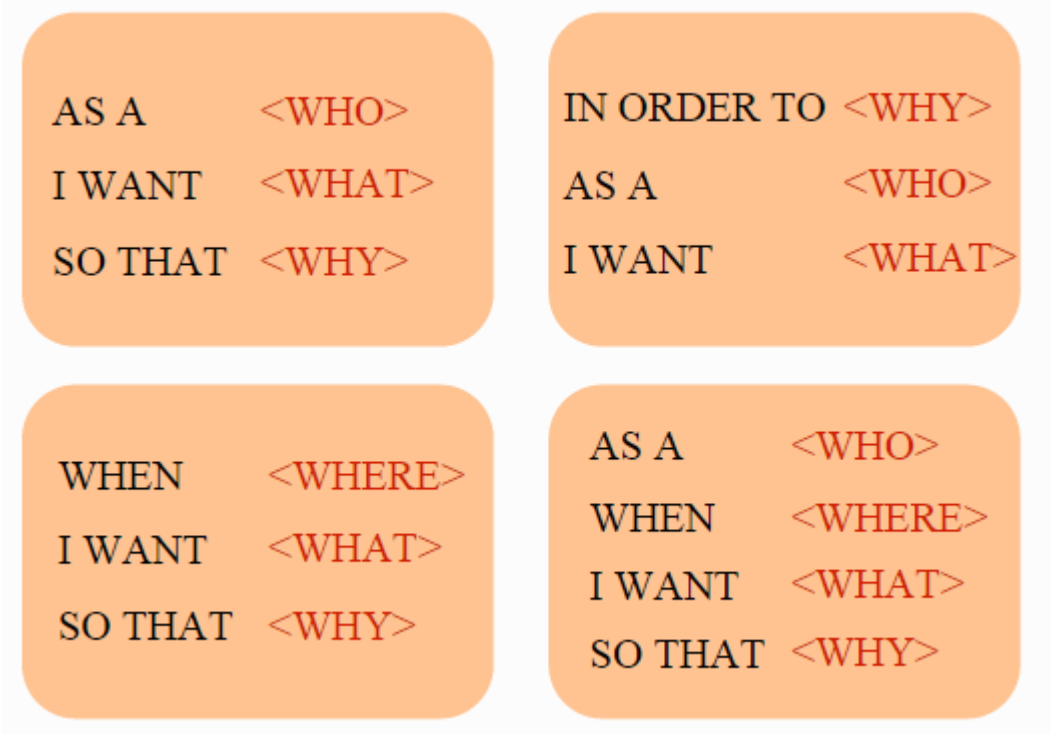
"In the first 6 months of service the Business owner requires response times for transactions on the web site to take no longer than 5 seconds between the hours of 9am and 5pm on a weekday. It is essential to provide a good (enough) experience to customers so they do not go to a competitor."

"After 6 months of service the Business owner requires response times for web transactions to take no longer than 2 seconds between the hours of 7am and 11pm on a Weekday and Weekends. It is essential to improve the customers experience and create a demand outside working hours."

As a User Story ?

Mike Cohen : A User story is a description of a feature told from the perspective of the person that desires that new capability.

Varieties of User Stories:



Example: As a User Story ?

As A	Business owner
I Want (Now)	to have response times for web transactions to take no longer than 5 seconds between the hours of 9am and 6pm on weekdays only.
So That	customers use the service but do not churn to a competitor.

As A	Business owner
I Want (in 6 months)	to have response times for web transactions to take no longer than 2 seconds between the hours of 9am and 11pm all days
So That	customers get an improved experience and I get an increased customer business outside working hours.

Issues:

- There 2 Stores are not independent. Ignoring the second story now may cost more in 6 months.
- Is this a 'User' Story, or a 'Business' Story ?
- Who is responcible, accountable, impacted by the story ?

"Users of a system don't care about the problems of the system's creators; such stories are not Valuable to them (in the sense of the INVEST model for user stories)." Bill Wake

Example: NFRs are not Acceptance Tests

Maybe a very simple 'I want' statement, and put the requirements in an Acceptance Test, fixes the problem ?

As A	new web customer
I Want	the web site responses to be less than 1 seconds all the time
So that	I don't get frustrated and leave the site
Acceptance Test	However, In the first 6 months, business accepts response times up to 5 seconds between the hours of 9am and 6pm on weekdays only. **So That** Customer finds service adequate but do not churn to competitor. After 6 months response times for web transactions to take no longer than 2 seconds between the hours of 9am and 11pm all days. **So That** customers get an improved experience and increase customer business outside working hours.

Confusing ! in a similar way placing the NFR in every Web Related Functional User Story as an Acceptance Test leads to management and traceability issues. The NFR Requirement knowledge is now hidden, it is not available on a backlog.

Mike Cohen on Using the common User Story Template for NFRs:

"But, you should be careful not to get obsessed with that template. It's a thinking tool only. Trying to put a constraint into this template is a good exercise, as it helps make sure you understand who wants what and why. If you end up with a confusingly worded statement, drop the template. If you can't find a way to word the constraint, just write the constraint in whatever way feels natural."

Need more knowledge about the requirement in context?

The business owner needs to understand what they have to invest, to ensure there are no issues in building, operating, distributing and owning their system.

- When could performance be impacted, what are the risks and impacts my business faces?
- What or who could cause a problem and how; what are the triggers?
- Where could impacts occur, in my system, or the legacy mainframe?
- What happens if something goes wrong and how does it get fixed?
- Who is responsible to mitigate the problem and keep things going?

More Knowledge is required to understand what is 'really required' to provide a quality business solution and estimate.

Need a richer knowledge approach and template.

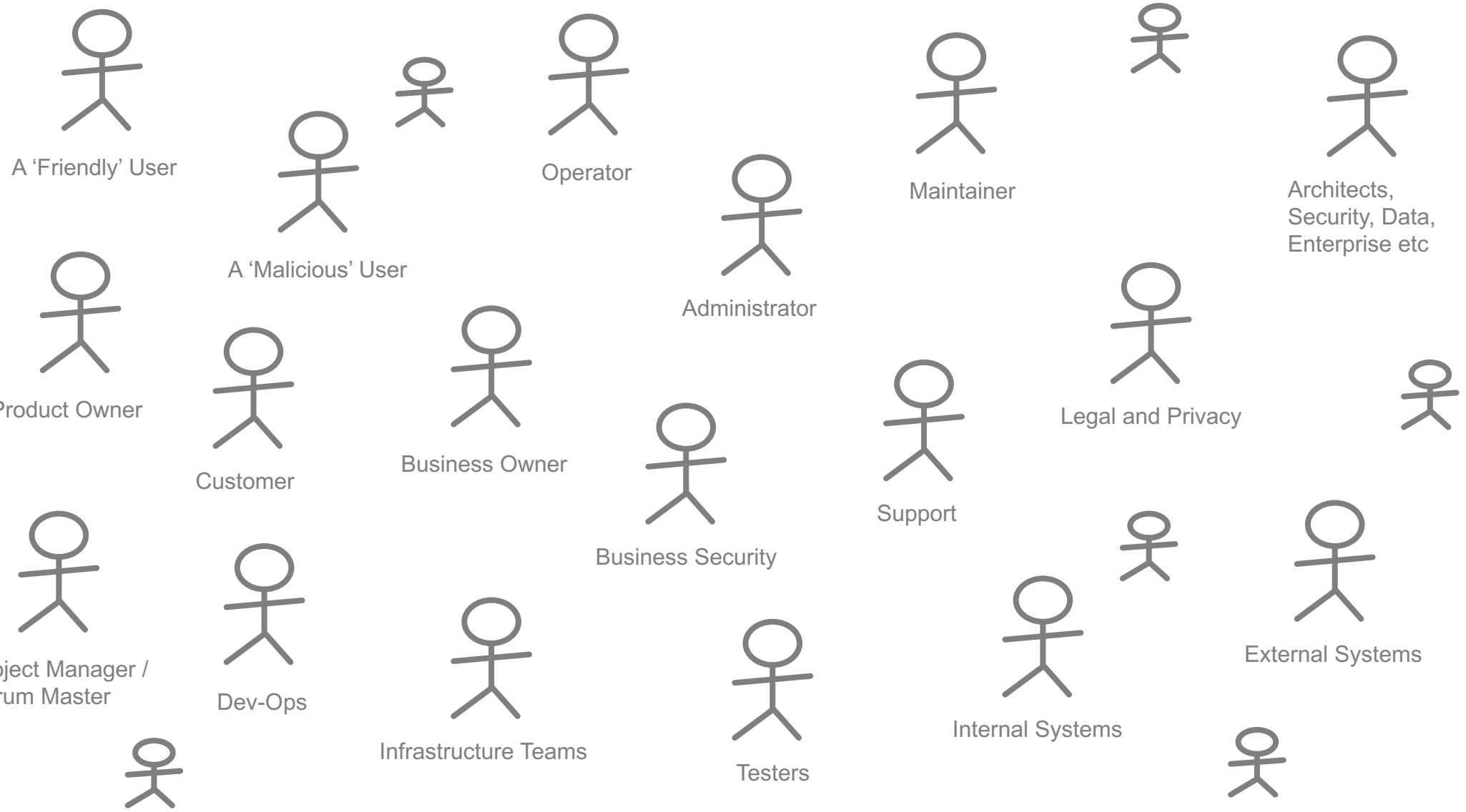
Its not just the ‘As a’ Person Involved its ‘We Want’.

There are many more responsible and accountable stakeholders (Actors) involved in this scenario, that have requirements too, than just the person who desires the requirement:

- The Monitoring team is responsible to instrument the new application, work out how to measure its performance and set up alarms.
- The Operations and Support staff need to know what to do when performance alarms ring.
- The Maintenance team needs to specify how they get involved when things go wrong and what to do to fix the issue.
- The Dev Ops team needs to deploy the apps, monitoring services.
- The Call Center has to handle calls, when performance is poor.
- Infrastructure team needs to know what hardware and software is required to run the system at right level of quality.
- The Architects have to create a design to achieve the performance, and explain to the business owner if its not possible, or requires a trade off, e.g. “we can achieve high performance but the system won’t be as reliable.”

Knowledge about these actors, how they are involved, and what they do, needs to be understood and communicated. Each stakeholder needs to be satisfied.

Many Actors - Who creates a problem and who has their Bacon in the Pan?



Attribute Scenario Template

Attribute Scenario Template

Attribute	Description
Goal	A description of the goal; what the goal owner wants (As A) and actors are trying to achieve (We Want) and why; relate to the business goal or driver. (see Goal Template – pack)
Evolution	Describe how the goal evolves over time; the future needs; these can influence the current solution. For example, what is required in 3, 5 years time.
Impact	The business impact (So That) if not met, e.g. business will loose 30% of orders, customer dissatisfaction will rise. Usually felt by user, then operators, system owner or whole enterprise.
Actors	The various stakeholders: source of the trigger, who reacts to the trigger and who mitigates the impact, who is responsible and accountable. General types are Internal or External.
Trigger	A condition or stimulus that starts the impact and the Response flows.
Response Flows	The response to the trigger, the sequence of actions (scenarios) required to mitigate and resolve the impact. These flows keep the business running and involve multiple actors cooperating. 3 Types: Prevention, Recovery, Backup
Environment	Conditions and context in which the trigger occurs, e.g. high load in production.
Artefact	What things, or part of the system, is impacted, e.g. web site / server.
Measures	How the goal is Measured and Tested and where in the Flows.

Template Attributes - used to trade off quality

Attribute	Description
Risk - Likelihood	Risk if NFR not met (L, M, H), and likelihood the impact occurs (L, M, H).
Importance	The importance of the scenario to the success of the system, and thus its importance to the business function; (L, M, H).
Difficulty	The degree of difficulty in achieving the scenario. (L, M, H).

Process is iterative and not all attributes are always required; depends on architectural significance and context.

For example start with:

- Goal
- Evolutionary
- Impacts
- Actors
- Importance

Then

- Environment
- Triggers
- Response
- Artifacts
- Measures

Then

- Importance (refine)
- Difficulty
- Risk
- Likelihood

Environment can have 5 main Context Types

1. **Usage Context** describes subsets of functionality
   - Functional Area, e.g. billing
   - Sets of stories or use cases
   - Specific transactions
2. **Source Context** describes categories of systems (e.g. internal or external) or users (e.g. type, skill, service purchased).
3. **Feature Context** describes usability feature sets, such as Online help, graphical layout, wireless or mobile access.
4. **Data Context** describes subsets of business data:
   - Functional area, e.g. billing data,
   - Data owned by specific users, e.g. Telecom data
5. **Configuration Context** describes different system configurations, e.g. client, server, test.

3 Main Types of Response

1. **Prevention** describes steps, processes, capabilities that should be put in place to prevent the impact, e.g. the ability to automatically scale systems, or provide failover.
2. **Recovery** can be automated or manual, e.g. the system will initiate a manual response, for example sending an Alert, or providing various capabilities to assist manual recovery.
3. **Back Up Plan** specify a less desirable response in case full recover is not achievable. Often called graceful degradation.

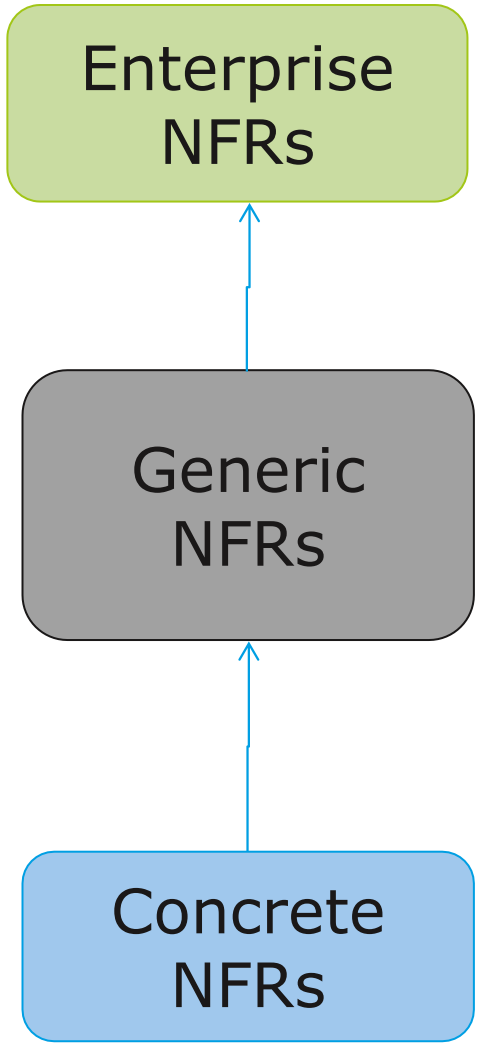
Throughput Example

Cause	Prevention	Recovery	Backup
Internal	All major system operations will be load tested in a simulated production environment prior to deployment	Operations staff will be alerted during periods of high activity	New user sessions will be rejected when system response times degrade
External	100% excess system capacity (storage and processing) should be used to handle infrequent spikes in demand.	Transactions in excess of 50 per hour should be queued and batched with delay up to 24 hours	Update operations will be rejected when system is overloaded.

Reliability Example – data corruption in accounts

Cause	Prevention	Recovery	Backup
Internal	• Affected account balances totals will be validated before commit. • Update Code Quality will be measured. • Update Component will be tested by independent team. • Update code will also be reviewed by independent team.	Starting account balances will be validated before each update operation begins	• Database snapshots will be made daily. • Code can be rolled back from production.
External	All updates will be checked for invalid data parameters	An undo log of updates will be maintained for 2 weeks.	Update operations will be denied to specified systems.

Each Generic Scenarios can provide 100s of Unique Scenarios



Enterprise: agreed qualities the organisation strives to meet for governance or strategy, e.g. 24*7 availability, scalability, PII, GDPR.

Generic: system independent, pertain to many systems. Capture domain knowledge, and summarise a range of possibilities for re-use.

Concrete: specific requirements for the system under consideration. Can be captured textual or graphically.

Reduce the number of options

Each generic scenario can result in 100s of specific concrete instances

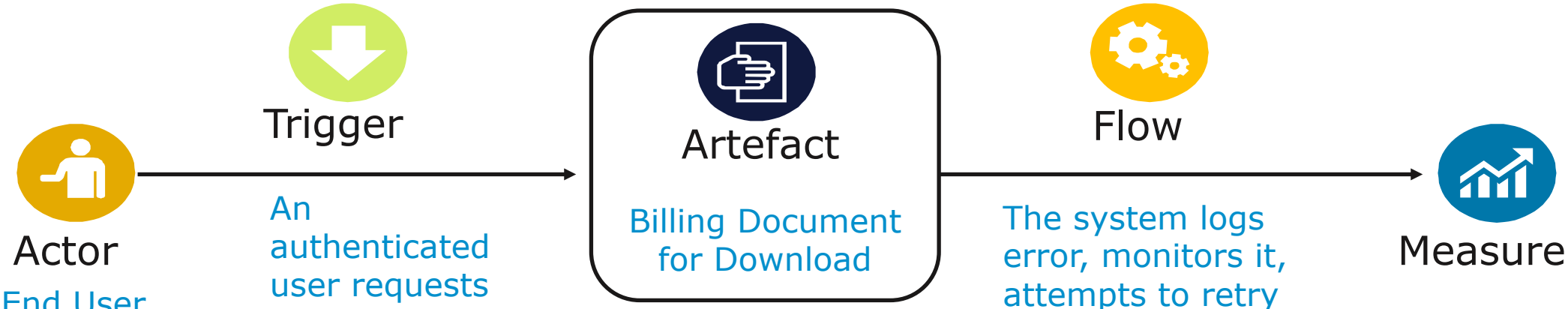
Generic NFs - Knowledge Capture

Generic ‘Availability’ Scenario - Example

Attribute	Possible Values
Goal	Ensure the system (Hardware/Software/Network/Security/Facilities) is available for users within the agreed SLAs.
Impact	Detrimental to business reputation, impacts customer processes and experience.
Actor	Internal or external to system, too many to list.
Trigger	Crash, fault, omission, timing, no response, incorrect response, security breach, e.g. DDOS attack.
Environment	Normal operation; degraded (failsafe) mode
Artifact	Applications, Servers, Network, VMs, ESBs, communication channels, storage etc
Response Flow	System should detect the trigger and do one of: Log the failure, notify users/operators, disable source of failure, continue (normal/degraded) or be unavailable for pre-specified period
Measures	Time interval available, availability%, repair time, unavailability time interval, degraded mode time interval

A Concrete 'Availability' Example - Graphical Form

The requirement is not just a generic statement about Availability but an agreed set of specific measureable scenarios



Tactic

Monitors and Retry

Generic ‘Performance’ Scenario - Example

Attribute	Possible Values
Goal	Ensure the system is architected to perform within the SLAs so users can effectively achieve their goals.
Impact	User frustration, users can’t work, loss of business, reputation
Actor	Internal and External systems
Trigger	Periodic events, sporadic events, stochastic events, with measureable features
Environment	Normal mode; overload mode
Artifact	System, or possibly a component
Response Flow	Process triggers; change level of service in environment
Measures	Latency, deadline, throughput, jitter, miss rate, data loss

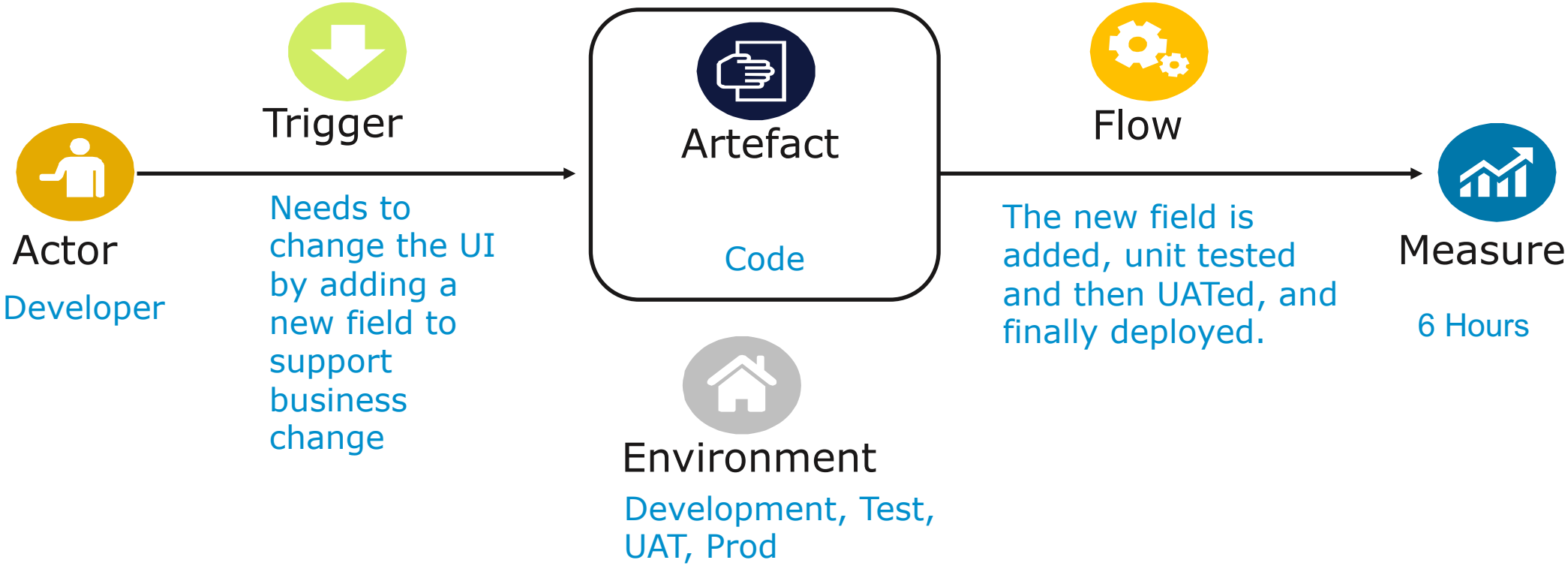
Generic ‘Extensibility’ Scenario - Example

Extensions can be through the addition of new functionality or through modification of existing functionality.

Attribute	Possible Values
Goal	Provide an architecture able to respond to requests to extend the system to meet new business or operational strategic needs. Extensions may include: New channel, device, product feature, service level, changed COTS.
Impact	Business strategy cannot be met, new opportunities lost.
Actor	End-user, developer, system-administrator, architect, operator.
Trigger	Add/delete/modify functionality or quality attribute.
Environment	At runtime, compile time, build time, design-time.
Artifact	System user interface, application, platform, environment.
Response Flow	Locate places in architecture for change, make change, test changes and deploy.
Measures	Cost in effort, money, time. Extent impacts other system functions or qualities.

A Concrete 'Extensibility' Example - Graphical Form

It is not enough to just say a system is Extensible, specific measureable scenarios need to be agreed. What kind of extensions and how long to deploy ?



Note: this is not a requirement to add a specified field, not a functional requirement.

Generic 'Testability' Scenario - Example

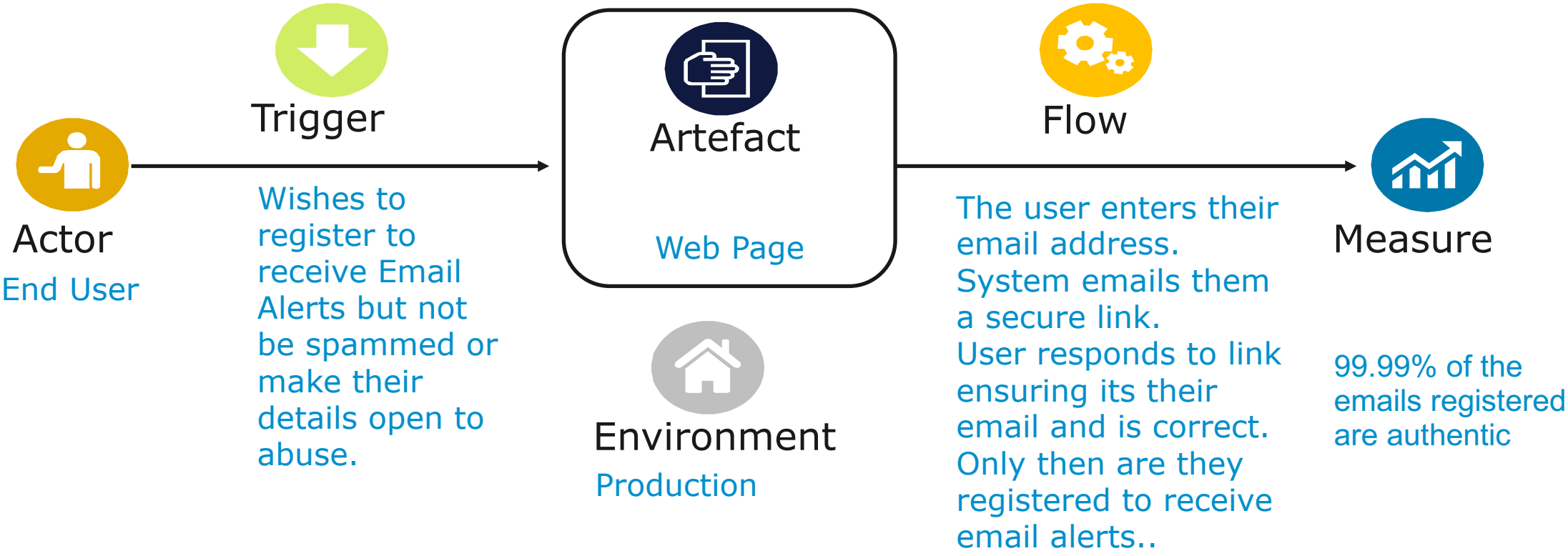
Attribute	Possible Values
Goal (Generic)	Provide a system that is able to be tested in the required environments.
Actor	Unit developer, increment integrator, system verifier, client acceptance tester, system user
Trigger	Analysis, architecture, design, class, subsystem integration, system deployment and delivery
Environment	At design time, at development time, at compile time, at deployment time
Artifact	Component of design, code, configuration or complete system
Response Flow	Provide access to state data values, provides computed values, observes results, compare
Measures	% executable statements executed; probability of failure if fault exists; time to perform tests; length of longest dependency chain in a test; length of time to prepare test environment

Generic ‘Security’ Scenario - Example

Attribute	Possible Values
Goal	Protect data and information from unauthorised access while providing service to those authorised. Maintain Confidentiality, Integrity, Availability, Assurance, Auditing, Non-repudiation
Actor	User/system who is legitimate/imposter/unknown/malicious with full/limited access. Internal or External. Authorised or Not Authorised, with access to limited or vast resources
Trigger	Unauthorised attempt to display/modify data (CRUD); access services, reduce availability or performance.
Environment	Normal operation; degraded (failsafe) mode
Artifact	System services and/or data within system or in transit.
Response Flow	• Authenticate user; hide identity of user; grant/block access to data and/or services; allow access to data and/or services; grants or withdraws permission to access data and/or services; • Records access/modifications or attempts to access/modify data/services by identity; stores data in an unreadable format; • Recognize unexplainable high demand for services, and informs a user or another system • Restricts availability of services
Measures	Time /effort/resources to circumvent security measures with probability of success, probability of detecting attack; probability of identifying individual responsible for attack or access/modification of data and/or services; percentage of services still available under denial-of-services attack; restore data/services; extent to which data/services damaged and/or legitimate access denied

A Concrete (real) 'Security' Example - Graphical Form

The requirement is not just a generic statement about Security but an agreed set of specific measureable scenarios. Agreed requirement was traded off with usability.



The User (Owner) wants 100% but sorry this is not possible so we ask the question:
"Would it be acceptable if in one out of 1000 cases (0.01%) the authentication was wrong in this situation?"

Generic Usability Scenario - Example

Attribute	Possible Values
Goal	Ensure the ease at which users can accomplish their tasks, and learn how to tuse the system. Concerns: Understandability, Learnability, Operability, Attractiveness, Recover-ability, Personalisation
Actor	End-user
Trigger	Wants to: learn system, use system, recover from errors, adapt system, feel comfortable, personalize their interface.
Environment	At runtime, or configure time, install-time
Artifact	System user interface and system (e.g. services)
Response Flow	To learn system: • Help system is context sensitive • Interface familiar, consistent To use system efficiently: • Aggregation of data and/or commands, Reuse of command or data already entered • Efficient Navigation and Comprehensive searching • Distinct views with consistent operations; • Multiple simultaneous activities To recover from errors: • Undo, cancel, recover from system failures • forgotten passwords To adapt system: customise and personalise the system to user liking To feel comfortable and productive
Measures	Task time, number of errors, number of tasks/problems accomplished, user satisfaction, gain of user knowledge, ratio of successful operations to total, amount of time/data lost

SOA/Service Example

Service Interoperability Example

Scenario I1

A new business partner that uses platform „X“ is able to implement a service consumer module that works with our available services in platform „Y“ in two person-days.

Scenario I2

A transaction in a legacy system running on platform „X“ is made available as a Web service to an enterprise application that is being developed for platform „Y“ using Web services technology. The wrapping of the legacy transaction as a service with proper security verification, transaction management, and exception handling is done in 10 person-days.

Service Performance Example

Scenario P1

The service provider can process up to „X“ simultaneous requests during normal operation, keeping the response time on the server less than „Y“ seconds.

Scenario P2

The roundtrip time for a request from a service consumer in the local network to service „X“ during normal operation is less than „Y“ seconds.

Service Scalability Example

Scenario S1

Marketing landed several new high-volume accounts that will increase service request volume by a factor of 10. During normal operation, the service requests are processed without affecting the current quality of service.

Scenario S2

Marketing landed several new high-volume accounts that will increase service request volume by a factor of 10. During normal operation, the service requests are processed according to the Service-Level Agreements negotiated with each account.

Service Security Example

Scenario 1

An attack is launched attempting to access confidential customer data. The attacker is not able to break the encryption used in all the hops of the communication and where the data is persisted. The system logs the event and notifies the system administrators.

Scenario 2

A request needs to be sent to a third-party service provider, but the provider“s identity cannot be validated. The system does not make the service request and logs all relevant information. The third party is notified as well as with the system administrator.

Service Availability Example

Scenario A1

An improperly formatted message is received by a system during normal operation. The system records the message and continues to operate normally without any downtime.

Scenario A2

Unscheduled server maintenance is required on server „X.“ The system remains operational in degraded mode for the duration of the maintenance.

Service Reliability Example

Scenario R1

A sudden failure occurs in the runtime environment of a service provider. After recovery, all transactions are completed or rolled back as appropriate, such that the system maintains uncorrupted, persistent data.

Scenario R2

A service becomes unavailable during normal operation. The system detects the problem and restores the service within two minutes.

Service Modifiability Example

Scenario M1

A service provider changes the service implementation, but the syntax and the semantics of the interface do not change. This change does not affect the service consumers.

Scenario M2

A service provider changes the interface of a service that is publicly available. The old verson of the service is maintained for 12 months, and existing service consumers are not affected within that period.

Sample NFR Questions

Questions to Prompt NFR thinking

- A set of **Questions** that help prompt NFR thinking is available and is currently being extended.
- A set of **Guidance** material to help explain technical terms, provide check lists etc, is being created.
- The following are some examples…….

Availability SLR Questions (example, not full list)

- What are the expected hours of operation ?
  Does it vary by use case ?
- Can the system become unavailable for scheduled maintenance?
  Any restrictions on the hours ?
- What is the acceptable downtime on a failure ?
- Are there times when downtime is more costly than others;
- Are different functions to be made available on different client devices?
- How soon after a disaster does this system's functionality need to be available ?
- How quickly must data in the system be restored from backup or rollback in the event of data loss or corruption ?
- Examples
  - Availability is required between 6 AM and midnight Mon-Sat. A 6 hour maintenance window exists between midnight and 6 AM and on Sundays.
  - If the connection is down to mainframe the User Interface should still function and allow changes to be submitted later.

Reliability SLR Questions (example, not full list)

- Are there operations about whose success or failure clients absolutely must be notified in an apparently synchronous fashion ?
- In the event of failure of one of the platform layers (e.g. losing a server), is it acceptable to lose the state of active client sessions? In other words, must platform failures be "transparent" to end-users?
  - Is it acceptable for them to have to re-login?
- In the event that corrupt or incorrect data has been sent from the system to other systems, what is the obligation to correct the data in the external system (e.g. by cancel/correct protocols)?

Example, a payment, or legal agreement, or trade submission.. Since nothing is 100% reliable, the real issue is "apparent ACID semantics" - if the communications link goes down, for example, at the next successful communication the success/failure of the operation must be immediately available.

Guidance: Availability & Reliability Aspects

Availability = (1 - MTTR/(MTBF + MTTR)) * 100%

Reliability:

- Measure: Mean Time Between Failures (MTBF) or number of failures per week;
- Need to define a ‘failure’

Recoverability:

- Recovery from failure
- Measure: Mean Time To Repair: (MTTR)
- Tactics: checkpoint restarts; auto-saves, etc.

Guidance: Availability and the NINES

Percentage Uptime	Percentage Downtime	Downtime Per Year	Downtime Per Week
98%	2%	7.3 days	3 hours, 22 minutes
99%	1%	3.65 days	1 hour, 41 minutes
99.8%	0.2%	17 hours, 30 minutes	20 minutes, 10 seconds
99.9%	0.1%	8 hours, 45 minutes	10 minutes, 5 seconds
99.95%	0.05%	4 hours, 23 minutes	5 minutes
99.99%	0.01%	52.5 minutes	1 minute
99.999%	0.001%	5.25 minutes	6 seconds
100.00%	0.0001%	31.5 seconds	0.6 seconds

Note: cost to implement increases exponentially as you go down the page

Scalability / Elasticity SLR Questions (example, not full list)

What is number of concurrent users at deployment?
- In 6 months? In 12 months? In 24 months?

What is the total user base at deployment?
- In 6 months? In 12 months? In 24 months?

Are there metrics for the data volumes which will need to be stored?

Does the system need to scale automatically or is this a manual operations process ?

Examples (not well stated):
- The proposed system must comply with the companies channels scalability standard.
- The proposed system must support, at peak, up to 40 External Customers and 100 Internal users simultaneously.
- The proposed system must be able to handle double the current connections per day, in one year.
- The proposed system must be able to support deployment to 4 more globally distributed cloud DC's in 6 months.

Performance SLR Questions (example, not full list)

What is the average desired response time or **latency** ?

- Does this vary by use case /story?
- Does this vary by timeframe?
- Does it vary across the day/week/month/year?

What is the maximum allowable response time?

- Does this vary by use case?

Do these acceptable/expected response times change with different types of connectivity (e.g. Intranet/Internet-broadband/Internet-dialup)?

How many user requests should the system handle at one time ?

Will the system be expected to deliver very large amounts of data, e.g. multi-media data?

- Does this vary by timeframe?

Are there regular circumstances (e.g. backups) under which performance is allowed to degrade?

Usability

Usability is concerned with how easy it is for the user to accomplish a desired task and the kind of user support the system provides. It can be broken down into the following areas:

- **Learning system features.** If the user is unfamiliar with a particular system or a particular aspect of it, what can the system do to make the task of learning easier?
- **Using a system efficiently.** What can the system do to make the user more efficient in its operation?
- **Minimizing the impact of errors.** What can the system do so that a user error has minimal impact?
- **Adapting the system to user needs.** How can the user (or the system itself) adapt to make the user's task easier? For Example personalisation.
- **Increasing confidence and satisfaction.** What does the system do to give the user confidence that the correct action is being taken?

Is Usability Architectural – in some a cases, Yes ?

Consider an Order Web site that requires the user to enter their address.

Version 1 of the system had users type in their address fully. The system then searched for an exact match. Often due to typos and spelling mistakes addresses were not found. The searches on the backend system could take 40 seconds. The company address data base is home grown and has many missing addresses

New Requirement to improve usability: provide a type ahead address lookup capability. As the user types, the system looks ahead and suggests matching addresses. The matching is fuzzy, and copes with spelling errors.

Architectural Concerns:

- The company has no backend service that provided fast address searched, a new system is required.
- A reliable source of address data will have to be procured and maintained.
- Network caching is required to speed up searches.
- An API is required so that the web page can call from JavaScript to a new service to look up partial addresses and return a list of matches, the time for these is 20-50 milliseconds per search. The service will be required to be secure and immune from DDOS attacks.

To improve usability a number of Architectural Decisions have to be made. It may be the case that the ROI is not good enough.

Usability Questions

How does the system support the users in performing their tasks effectively and efficiently?

How can the system provide users with feedback on progress as they interact with it?

Does the system draw on real-life concepts, language and relationships that are familiar with the targeted users?

Is the use of language and patterns consistent within the system and with related systems?

Is the navigation scheme structured according to the users' mental model?

Does the system facilitate learning and recognition so that the users can become proficient?

Does the system support users by minimising the risk of making an error?

Does the system support users in recovering from errors quickly and effectively?

Does the system offer users the freedom to make choices on how they wish to use the system to achieve their goals?

Other Usability – Sub Qualities

Useful	Is the product or system useful to the targeted users? Does the product or system improve the efficiency of the targeted users in performing their tasks, as measured by time taken to perform a task, number of manual steps etc.?
Desirable	Will the users have a desire to switch from current process/system to using the new process/system?
Findable	Will users be able to find the system so they can use it?
Credible	Will the user trust and believe the data, and what the system can do?
Valuable	Does the system deliver value to the users and the organisation?
Compliance	Does the system comply with relevant standards and legal obligations? Does the look and feel comply with the company's brand guidelines?

Accessibility Questions

Is the system designed to be inclusive so that the majority, if not all, users can access and use it?

Is there an alternate available for users who are unable to access or use the system, in cases where some users are unable to use it?

Does the system comply with WCAG 2.0 or accessibility?

Can the system be accessed across the mainstream browsers and operating systems?

Can the system be used with assistive technologies / Screen readers?

Can the system be used by users in geographic locations with low bandwidth and poor connection?

Does the system have accessible content? (e.g.: meaningful context for links, changing labels to be more meaningful)

Government Enterprise. It is a mandatory requirement that all government websites comply with WCAG 2.0. Under the Disability Discrimination Act 1992 agencies must ensure that people with disabilities have the same fundamental rights to access information and services as others in the community

Security SLR Questions (example, not full list)

- What is the user's expectation for confidentiality of data transmitted, manipulated or stored by the system?
- Is there personal identifiable information stored or transferred ?
- Does the organization have legal liability if this confidentiality is compromised?
- Does authorization for capabilities/functions need to be based on data (e.g. age of user) or simply role?
- Is there an obligation to timeout inactive systems?
- Do you have an obligation to prevent users from having multiple sessions active simultaneously?
- What are the end-user's expectations for protecting access to this system?
- What are IT's expectations for protecting access to this system?
- Does the organization have legal liability if this access is compromised?
- Is there a requirement to support delegated security administration?
- Are there operations which must be auditable?
- To what extent is the organization responsible for data security on the client device?

Manageability SLR Questions (example, not full list)

Manageability is a major area where ‘functional’ requirements are often not collected. The focus on ‘customer’ users often results in overlooking the many other actors involved with the system. Operational Use Cases / Stories cover these.

- Have the operators and managers of the system been identified as actors?
- Have the standard "operational use cases/stories" been considered and included in the inventory?
- Are there functional tasks which will need to be performed by the data center operations staff, for example batch jobs or reports?
  - If so, have they been expressed as use cases, stories?
- Are there functional tasks which will need to be performed to keep the business running, the system, that are not performed by end users but by maintenance, operational or key business personnel.

Flexibility / Extensibility SLR Questions (example, not full list)

- Is there a "single data image" that must be consistent across all users/accesses of the system at all times (complete consistency for ACID)? Is this true for some data and not others?
- What is the acceptable latency for a content change (e.g. an image or descriptive text), once released, to be propagated to end-users?
- What is the acceptable latency for a business data change (e.g. a price change), once released, to be propagated to end-users?
- What is the acceptable latency for a business rule or policy change to be propagated to end-users (e.g. a discount program)?
- What is the expected/desired development time for new features for the system?
  - Feature "enhancements"?
  - Non-static UI changes?
  - Database schema changes?

Testability SLR Questions (example, not full list)

- Are TDD practises employed?
- Is testing performed continuously or once with a release?
- Is testing fully automated?
- Is testing a precursor to code submission?
- What tests are performed: security and PEN, smoke, acceptance (UAT), unit, component, interface/API, Alpha/Beta, Performance, Load, Failure, usability, other NFR tests…
  - When are these tested, incrementally or at releases?
- Is it clear what to test; requirements are of good quality and traceable to components?
- Are systems tested manually via their devices and user interfaces?
- How much user interface testing is performed automatically?
- Are component/service interfaces and APIs tested independently of the system as a whole?
- Is mocking employed to test systems isolated from interfaced systems?

Traceability SLR Questions (example, not full list)

- Are the requirements traceable to the business goals and drivers, the stakeholders rationales and objectives?
- Are the development and maintenance changes traceable to requirements, architectural decisions, work products, code, configuration and test cases. Are these traceable through to the deployable components.
- Are the above supported by tools, e.g. APM, Issue, Change management. tools?
- Are Software Configuration Management practices used to manage software change?
- Is it important to trace the flow of data across transactions, processes and systems?

Modularity / Packagability SLR Questions (example list)

- Is the system code componentised logically for development?
  - How is the component quality considered, e.g. coupling, cohesion?
  - Are the reasons for component decomposition understood?
- Are the system components for deployment logically packaged?
  - How do the development components relate to the deployment packages?
  - Are the packages uniquely identified?
  - Are the packages versioned?
  - Are they deployed in an immutable manner?
- Can individual components be deployed or the whole system requires re-deployment with small changes?
- How easy is it to roll back deployed components?

Deployability SLR Questions (example, not full list)

- How often does the business need to change the product?
  - Continuously or incremental larger releases?
  - What is the expected frequency of deployments ?
- Can the product be changed via configuration or does it require partial or full re-build?
- What is the impact on the customer when the system/product is changed?
  - How often can customers cope with a changed system?
- Are their timely, or legal, business processes that must be completed before a new product or feature is released?
- What is the time required to deploy a change?
- What is the time required to roll back a deployed change?
- Should the system be able to be deployed while users are accessing the system, with no visible impact?
- Can the new release be deployed automatically?
- Is precisely the same mechanism used to deploy to every environment ?
- Have the change management issues, such as operational, maintenance and servicing changes, been addressed with more frequent releases, e.g. training, operations manuals, service manuals and processes?

Variability SLR Questions (example, not full list)

- Can certain features of the system be configured without requiring re-deployment or rebuild?
- Can configuration items be deployed independent of the whole system?
- How many different platforms does the system need to run on with different configurations?
- What are the different environments the system needs to run on with different configurations?
- Are there different contexts or markets that require different configurations?

Code Quality Measurability

- Does the development process include steps to measure code quality?
- What features of the code quality are measured?
- Is code continuously inspected?
- Are standard metrics defined for ‘good’ quality?
- Is the quality measured automatically, e.g. static analysis tools?
- Is the measurement part of the build/check in process?
- If code does not meet quality what happens?
- Is code reviewed by hand?
- Is complexity considered as a measure?
- Are security vulnerabilities tested in code?
- Are introspection tools used early on to look into running code for bottlenecks and performance issues?

Plannability

- Is this a novel development with new technologies that the team has not experienced ?
- How much of the system is a Proof of Concept ?
- Can the system be decomposed into deliverable components ?
- Is the delivery of key components out of the project managers control, e.g. third party supplier ?
- Is the estimation accuracy known and risks understood ?
- Are all the dependencies understood ?
- Are there parts of the system that just cannot be estimated with any reliability ?
- Can a set of increments be envisioned and an initial iteration plan provided.
- Can major risks be mitigated early on ?
- Are all the resources required willing to work at the required or known times ?

Buildability

- Can all the services, mechanisms and components be sourced, constructed or integrated with?
- Are there separate teams that may not be able to work together due to distance, legal issues, security clearance etc
- Is the risk around resourcing impacting development?
- Is the customers expectation for build completion too short, unrealistic?
- Are there decisions that need to be moved forward that are high risk to the build?

Reusability

- What artefacts are re-used: Requirements, NFRs, software components, services, infrastructure, test plans, document templates, views, etc ?
- Are components of the system to be used by Stakeholders other than those:
  - Paying for the system ?
  - Building the system ?
  - Accessing the initially deployed system ?
- Is there an accounting model for shared services, for build cost, operational and maintenance cost ?
- How are re-usable components re-used:
  - White box – modified, multiple copies to maintain;
  - Black Box – used unmodified;
  - As Example – used for ideas only;
- Are boundaries of re-use clearly established.
- Are there clearly defined interface/contracts ?
- Does organisation have a re-use lifecycle and governance model ?
- Are there mature re-use metrics ?
- Are existing artefacts harvested for re-use ?

Interoperability - Guidance

Achieved by:

Standard Implementation – agreement to use an industrial, national, open or international standard.

Industry/community partnership - sponsor standard workgroups with the purpose to define a common standard.

Common Technology and IP - reducing variability between components from different sets of separately developed software products, e.g. 3rd party libraries, open source or using a common vendor product.

Product Testing - Products produced to a common standard, depend on clarity of the standards, but there may be discrepancies in their implementations that system or unit testing may not uncover. Interoperability testing is required, which is different to conformance testing, as conformance to a standard does not necessarily engender interoperability with another product which is also tested for conformance.

Product Engineering - Implements the common standard, or a sub-profile thereof, as defined by the industry/community partnerships with the specific intention of achieving interoperability with other software implementations also following the same standard or sub-profile thereof.

Interoperability - Questions

- Is a standard used to define interoperability ?
  - Are they industrial, national, open or international standard ?
- Are the same products or common technology used to achieve interoperability?
- Are industry/community partnerships used to govern interoperability ?
- Is testing carried out amongst industry/community partnerships , e.g. an Open Source Forum?
- What integration standards are employed, e.g. XML, Web Services, EDI ?
- Is search required across components ?
- How is change and versioning managed across interoperating components?

Serviceability - Guidance

A **service tool** is defined as a facility or feature, closely tied to a product, that provides capabilities and data so as to service (analyze, monitor, debug, repair, etc.) that product. **Service tools** can provide broad ranges of capabilities. Regarding diagnosis, a proposed taxonomy of service tools is as follows:

- **Level 1**: Service tool that indicates if a product is functional or not functional. Describing computer servers, the states are often referred to as 'up' or 'down'. This is a binary value.
- **Level 2:** Service tool that provides some detailed diagnostic data. Often the diagnostic data is referred to as a problem 'signature', a representation of key values such as system environment, running program name, etc. This level of data is used to compare one problem's signature to another problem's signature: the ability to match the new problem to an old one allows one to use the solution already created for the prior problem. The ability to screen problems is valuable when a problem does match a pre-existing problem, but it is not sufficient to debug a new problem.
- **Level 3:** Provides detailed diagnostic data sufficient to debug a new and unique problem.

Serviceability

To Fix

- Help desk notification of exceptional events (e.g., by electronic mail or by sending text to a pager)
- Network monitoring
- Documentation
- Event logging / Tracing (software)
- Logging of program state, such as
  - Execution path and/or local and global variables
  - Procedure entry and exit, optionally with incoming and return variable values (see: subroutine)
  - Exception block entry, optionally with local state (see: exception handling)
- Software upgrades
- Graceful degradation, where the product is designed to allow recovery from exceptional events without intervention by technical support staff
- Hardware replacement or upgrade planning, where the product is designed to allow efficient hardware upgrades with minimal computer system downtime (e.g., hotswap components.)

Maintainability – Guidance

Various experts have asserted that most of the cost of software ownership arise after delivery, i.e. during “maintenance”. (E.g. > 90%, Erlikh, L. (2000). Leveraging legacy system dollars for E-business. (IEEE) IT Pro, May/June 2000, 17-23 !)

But software doesn’t wear out? No, but it gets

- fixed (corrective maintenance),
- adapted to changing needs (adaptive maintenance),
- improved in performance or maintainability (perfective maintenance)
- improved by fixing bugs before they activate (preventive maintenance)

[ISO/IEC 14764, following Swanson]

NFR Quality Tree

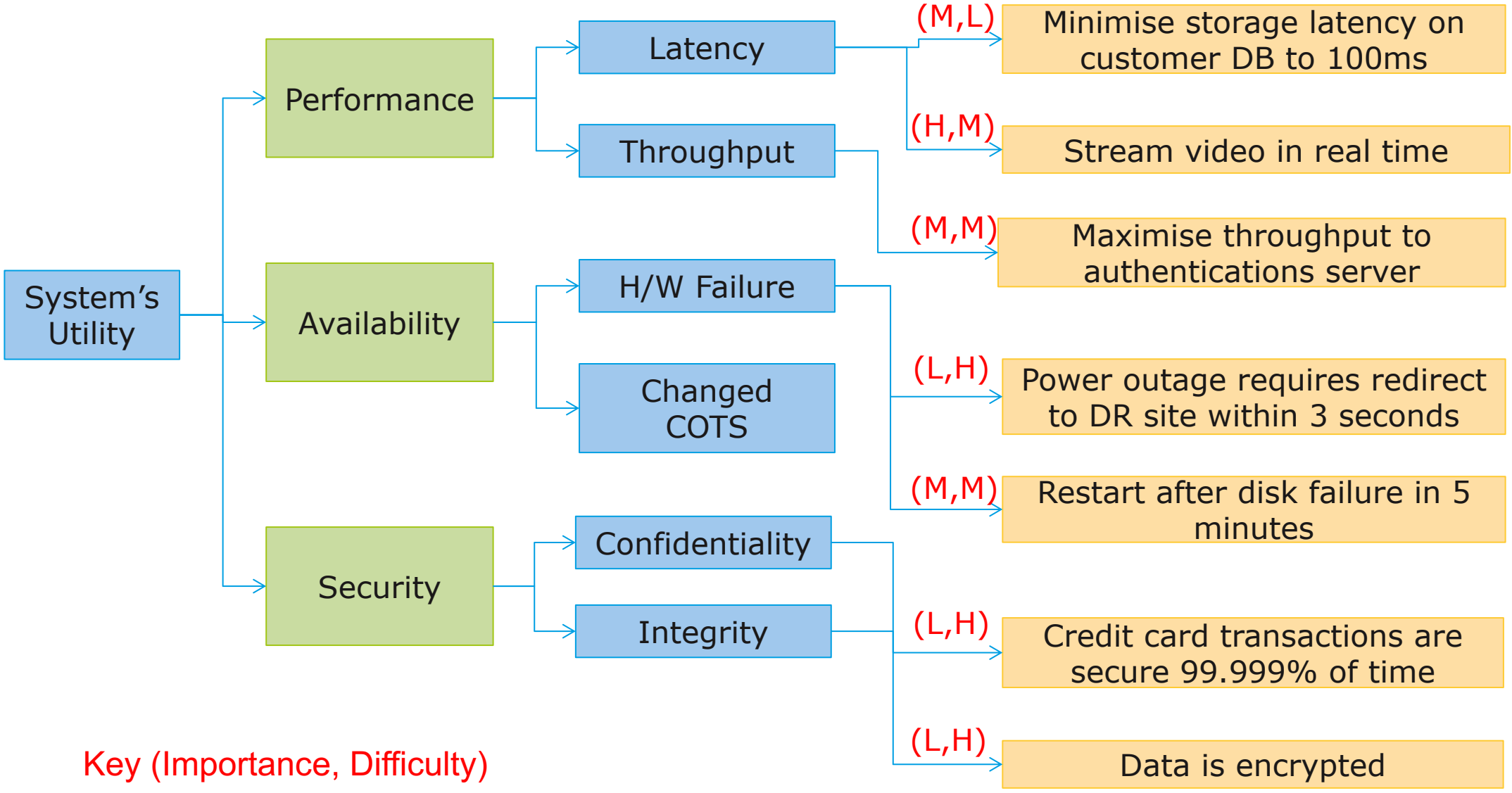
Systemic Qualities Interact - Example

These (across) impact those below, Positive (+) and Negative (-) interactions.	Scalability	Serviceability	Availability	Manageability	Security
Scalability		+	-/+	+	-
Serviceability	+/-		-/+	+	-
Availability	+/-	+		+	-
Manageability	+	+	+/-		-/+
Security	+	+/-	-	+	

Its all about Balance and Trade Off

- The fact that qualities interact means that ***trade off and balance*** is required. Its often just not possible to deliver one quality at 100% without reducing others.
- **A systematic 'method' is required to analyse these trade offs, record and manage the iterative process.**
- BaaS, Clements and Kazman claim that system quality stems from architectural quality and describe an approach:
  - Qualities can be organised in a 'utility' tree structure, the nodes describe Quality Requirements;
  - At the leaves of the tree, scenarios (The flows) are employed as the atomic elements to describe the detail that are concrete enough for prioritisation and analysis.
  - Each Node can have its Importance and Difficulty noted, to help start the trade off and prioritisation conversation.

Quality Tree – a way to visualise the Forest



NFRs: Constraints Verses SQs

A constraint is a limitation, dependency or assumption on the "solution", to the functional and non functional requirements.

- Constraints are beyond your control, they are not negotiable.
- If the constraints are not properly managed, they could limit the project success.
- The system is to be designed, built and architected within the required constraints, **that is, they are non-architectural decisions.**

It is often suggested that NFRs are 'constraints'. An important distinction is that most **NFRs require 'balance' and 'trade off'**, and some can be negotiated, these are the SQs.

If this is not so, and the requirement cannot be negotiated or 'traded off' then the NFR Statement is a Constraint.

Constraints – 3 Main Types

- Development Process and Team
  - Describe the manner in which a project may be formed and operated in order to build the proposed system.
- Environment and Technology
  - Describe the environment in which this system will be built and operated in.
- Delivery and Deployment
  - Describe criteria the system must meet in order to provide a complete solution.

Constraints - Examples

- The project development process must be in-line with the Enterprise Methodology;
- It is required that operational actors be skilled in Oracle App Server in order to operate the system;
- The system must reuse the following Business Services and Frameworks…
- The system will depend on a specified legacy system for report generation;
- The system functionality is delivered to the user via two communication channels: a browser running on a desktop and a browser displayed on a mobile phone;
- The following service contract must be signed before the system can operate…

NFR Iterative Approach

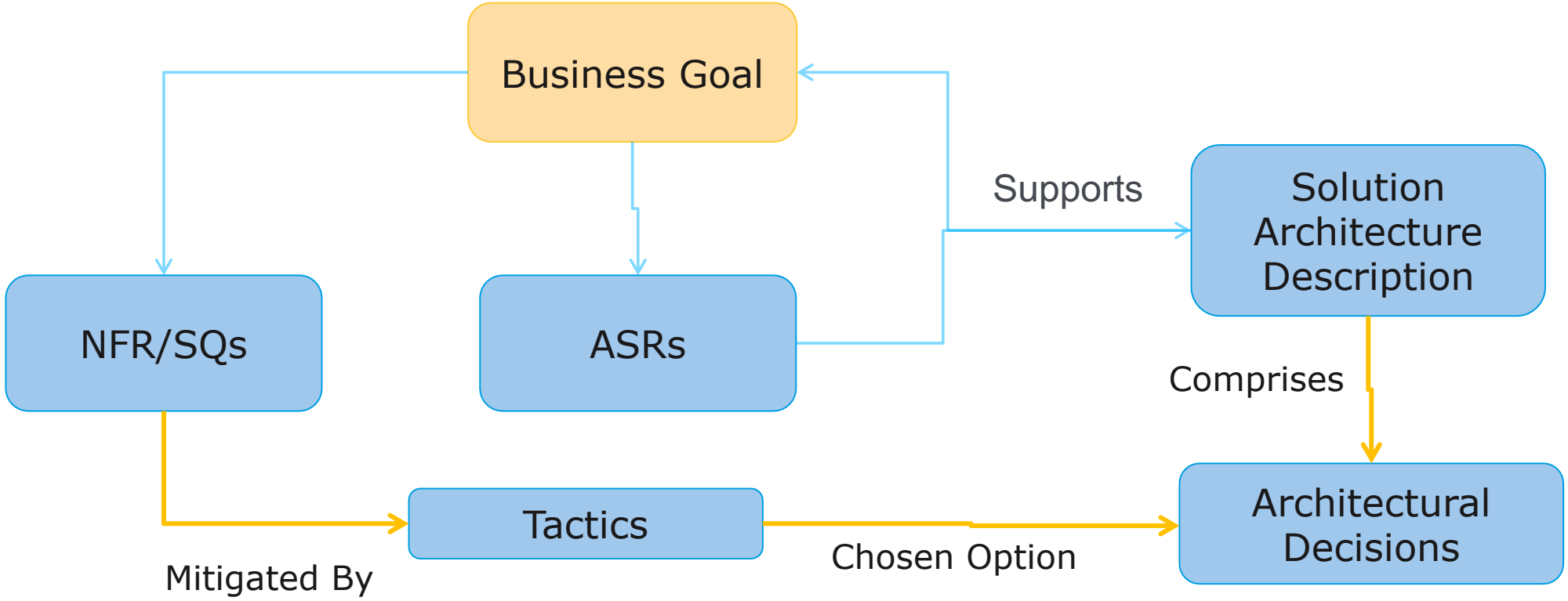
1. Starts right at the beginning of the project concept. The vision should contains a list of the 3 most important ‘Qualities’ for the Business Stakeholders, e.g.
    - *To support our customer base in 3 years the system must be able to ramp up from 100s of customers per hour in the first years to 1000s per hour in third year.*
    - *Initially the system can be available during business hours but after 1 year we need to extent the hours to 23 / day.*
    - *Customer data privacy is critical, and legally required.*
2. Iteratively refine these statements and consider other ones beyond the 3 most important.
    - Ensure they are measureable and testable.
3. Focus on the operational impacts to support the often extensive change required.
4. Gather the template attributes iteratively and focus on the Important and architecturally significant ones first.

Are Stories right for NFRs – Is all knowledge the same ?

Although on the surface User Stories seem OK, NFRs are very different to Functional Requirements and the fit is not great. Some of the issues are:

- INVEST just does not hold :
  - (I) they are not independent, are mostly pervasive and cross cutting across many functional stories.
  - (S) they are not small, they impact the whole architecture and many stakeholders
  - (N) they are often not negotiable as solution requires trade off and balance. For example, security is often not open to negotiation.
- Knowledge Attributes Missing.
- Acceptance Tests are misleading as they are the Goals. The GOAL is what is acceptable, and measureable. It is the GOAL that needs to be clearly specified, not the test, and not in a vague manner , e.g. "As a product owner I what high performance" is not good enough.
- NFRs require a richer structure, they are not generic statements, and always require business to understand the triggers and mitigation flows, the many actors and the extent across the enterprise.

Future Work – To Do



- Socialise current materials with wider teams.
- Further complete the Enterprise, Generic Templates and Questions material.
- Complete the Catalogue of Tactics, options process.
- Integrate into Architectural Decisions and Governance process.
- Integrate with volumetric modeling work.
- Complete Architectural Methods description.
- Add Measures/Metrics for Continuous Improvement.
- Review with Stakeholders…….

Enterprise NFRs

Generic NFRs