Non-Functional Requirements
(they’re actually quite important)

Stephen Booth MBCS MIET OLA
Requirements Analyst
https://www.linkedin.com/in/stephenboothuk

This slide deck is adapted from the one I delivered on Tuesday 9\textsuperscript{th} December to Birmingham Branch of British Computer Society based on feedback received.

It can be downloaded from my SlideShare (can be accessed via my LinkedIn profile, https://www.linkedin.com/in/stephenboothuk). I welcome connection requests on LinkedIn and never IDK.

If you have any questions please contact me via LinkedIn.
The obligatory ‘About Me’ slide

Those of you who attended Hannah Dee’s talk on Women in IT may recall her comment that most people in IT have already failed at something else.
Non-Functional Requirements

- What are they?
- What aren’t they?
- Where do you find them?
- Why are they important?
- What should you do with them?
What are Non-Functional Requirements?

- “Anything about a system that isn’t what it does.” (Wikipedia)
- “A property or quality that a product must have, such as appearance, speed, security or accuracy property” (‘Mastering Requirements’, S and J Robertson)
- “Numbers, Nouns, Adverbs and Adjectives, not Verbs”

There’s a lot of literature out there on what a non-functional requirement is, unfortunately there’s a lot of disagreement and differing points of view. The one from ‘Mastering Requirements’ by Suzanne and James Robertson is probably the most correct but still doesn’t really help as it’s quite academic.

A good first step for finding out if a requirement is functional or non-functional is to ask yourself if the solution could ‘Do’ the requirement or if the requirement is more ‘Be’ or a way of ‘Doing’.
What aren’t Non-Functional Requirements

- What it does.
- Anything you’d draw a flow chart for.
  But can be:
  - How it does what is does
  - How it looks
  - How big it is
  - How long it takes to do things
  - How it is restricted or constrained

What is does is the functional requirements, these are usually quite easy to get from stakeholders (the people who will use, manage, support, pay for or be impacted by the item or system).

You can have requirements for anything really: A piece of software, a system, a car, a bridge, a job candidate or even a partner (M, 44, cuddly, NS WLTM F, 34-54, NS with car, GSOH and low expectations/standards for a mate).
Most of us will have a mobile phone, may be one for work and one for personal use, there’s a good chance it’s a smart phone. A lot of us have probably had more than 1 over the years (discounting work phones issued to me by my employer I’ve had 11, work phones brings it up to 16, including like-for-like replacements for stolen or faulty phones). Unless it was bought for us by someone else, we have had the experience of walking into a mobile phone shop to pick a phone.
Most of us will have a mobile phone, may be one for work and one for personal use, there’s a good chance it’s a smart phone. A lot of us have probably had more than 1 over the years (discounting work phones issued to me by my employer I’ve had 11, work phones brings it up to 16, including like-for-like replacements for stolen or faulty phones). Unless it was bought for us by someone else, we have had the experience of walking into a mobile phone shop to pick a phone.
What do you want it to do?
(Functional)

- Make and Get Calls
- Store your address book and use the entries
- Send and Receive Texts
- Take Photographs
- View and Update Facebook/Twitter/Anonomo
- Watch YouTube Videos
- Sync with Outlook and Google
- Read and Send Emails

Functional requirements are usually quite easy to gather as most people know what they want something to do.
Whether you realise it or not you’re also thinking about some of the Non-Functional requirements. For example: Will it fit in your hand comfortably? Will it get too hot to hold when you’re making a long call? How much is the contract, how many minutes and texts does that include, will it include enough data allowance and how much will you be charged if you use more minutes/texts/data? What sort of keyboard you want (physical or onscreen). All good Non-Functional requirements.

You may also think about how many megapixels the camera should be and about how much storage you need (in terms of how many photographs and/or hours of music you can keep on the phone or what apps you absolutely must have). You might also have a view about which brand you want (Apple, BlackBerry, Samsung Galaxy, Nokia Windows Phone &c) due to brand loyalty.

If you are procuring phones for other people to use at work then compatibility with existing systems and ruggedness may be a factor. If you’re buying a phone for your kid then the availability of a Ben10, Disney Princess or Frozen case might be a factor.
Whether you realise it or not you’re also thinking about some of the Non-Functional requirements. For example: Will it fit in your hand comfortably? Will it get too hot to hold when you’re making a long call? How much is the contract, how many minutes and texts does that include, will it include enough data allowance and how much will you be charged if you use more minutes/texts/data? What sort of keyboard you want (physical or onscreen). All good Non-Functional requirements.

You may also think about how many megapixels the camera should be and about how much storage you need (in terms of how many photographs and/or hours of music you can keep on the phone or what apps you absolutely must have). You might also have a view about which brand you want (Apple, BlackBerry, Samsung Galaxy, Nokia Windows Phone &c) due to brand loyalty.

If you are procuring phones for other people to use at work then compatibility with existing systems and ruggedness may be a factor. If you’re buying a phone for your kid then the availability of a Ben10, Disney Princess or Frozen case might be a factor.
A business analyst can take those and, in discussing them with you, produce detailed definitions of these high level Non-Functional requirements such as:

**Between 50 and 70mm wide, 100 and 1200mm long, 4 and 9mm deep**

**Colour - Plain Black, White or Grey case**

**Price - Free!**

**Keyboard - Physical Keyboard**

**Contract - No longer than 24 months, £20pcm and including 300 minutes, 500 texts and 2Gb data each month**

These are examples of Non-Functional Requirements

These are examples of non-functional requirements.

Maybe your requirements are different.
UX or User eXperience is a bit of a buzz word but does cover the important area of how people use the software. This can include the look-and-feel of forms in an application running on a PC through to what sort of buttons and knobs (do you need them to be operated whilst wearing thick gloves, how rugged do they need to be &c) are needed for an embedded system. It is likely that most systems will need to be usable by people with disabilities who might need to access them in different ways or may use assistive technologies or who need support for different languages. Form design can be key as you want to make sure that users can easily see where to enter data, know what needs to go in each field (labels, explanatory text, ToolTips etc) and have fields for all of the data they need to enter. Where you are replacing an existing system you may have to consider what needs to be imitated from the old system and what can be new.

How will security be handled? Are there any external directories that need to be supported for logons? Single-Sign-On tokens? Two factor or three factor authentication? Do access levels map to business roles? Does data need to be encrypted before storage? What audit logs does the system need to write (GPG13)? Audit levels? What level of disaster recovery do you need (are prepared to pay for)? How often should the system be backed up? Full or incremental? Backup windows?

How many users does it have to support? How many concurrent? How much data will it need to store? How will the amount of data change? How quickly will it need to operate?

Maintainability includes patching schedules, publishing of API calls and interfaces,
compliance with open data formats and access to source code (in particular if the supplier goes out of business or ceases to support the product). Supportability includes access to support documentation and training, access to configuration files and interfaces, diagnostic instrumentation, logging and alerting. Those of you who attended Chris Hills’ talk “Requirements are Required” earlier this year may remember his mentioning a company that installed airport runway lights then needed to patch the firmware, then they discovered they hadn’t put in anything to allow them to install a patch other than by visiting each light individually, removing it, applying the patch then putting it back. All this on a runway that was in near constant use. Reliability may seem like a strange one as, of course, you want the system to be as reliable as possible. Unfortunately high levels of reliability are likely to increase the cost and time to market so a customer (or your own sales team) may settle for a less reliable system that is cheaper and/or gets to market more quickly.

Design constraints might be connected to the hardware the software will be used on (e.g. a PC with 17 inch monitor and separate keyboard or a mobile phone with 4 inch screen of which a third is taken up by the onscreen keyboard), versions of support software (JVM, Browser, Flash &c that will be on the machine) or the valid values for a field. For hardware they might include a size and weight limit or environments it must be able to work in (there’s no point having something that is supposed to be hand held and used outdoors in all weathers that weighs 30kg and breaks down if it gets damp).

There may also be constraints imposed by parallel running with an existing system during the implementation phase so the new system must not interfere with the old. I recall one roll out where a new system had to be rolled out to PCs that operated as tills with attached receipt printers. In test, without the old system on the PCs, it worked fine but no-one had considered parallel running. When we came to install the software on the PCs out in the wild the new system couldn’t print receipts. Two external consultants with a day rate of £1,500 each had spent 3 weeks (that’s £45,000 pounds) failing to fix the problem on the customer site. Had the requirement to parallel run been captured we may have tested for and found the issue much earlier and been able to fix it more quickly, or at least not in full view of the customer.

What information are you going to want to extract from the system? How do you want it summarised and presented? Do you want to report from the transactional data or implement a DataMart? How do you want it output?
Accessibility/UX

UX or User eXperience is a bit of a buzz word but does cover the important area of how people use the software. This can include the look-and-feel of forms in an application running on a PC through to what sort of buttons and knobs (do you need them to be operated whilst wearing thick gloves, how rugged do they need to be &c) are needed for an embedded system.

It is likely that most systems will need to be usable by people with disabilities who might need to access them in different ways or may use assistive technologies or who need support for different languages.

Form design can be key as you want to make sure that users can easily see where to enter data, know what needs to go in each field (labels, explanatory text, ToolTips etc) and have fields for all of the data they need to enter.

Where you are replacing an existing system you may have to consider what needs to be imitated from the old system and what can be new.
Security/Auditability/DR/Backups

How will security be handled? Are there any external directories that need to be supported for logons? Single-Sign-On tokens? Two factor or three factor authentication? Do access levels map to business roles?

Does data need to be encrypted before storage?

What audit logs does the system need to write (GPG13)? Audit levels?

What level of disaster recovery do you need (are prepared to pay for)?

How often should the system be backed up? Full or incremental? Backup windows?
Capacity/Volumetrics/Performance

How many users does it have to support?
How many concurrent?
How much data will it need to store? How will the amount of data change?
How quickly will it need to operate?
Maintainability/Supportability/Reliability

Maintainability includes patching schedules, publishing of API calls and interfaces, compliance with open data formats and access to source code (in particular if the supplier goes out of business or ceases to support the product).

Supportability includes access to support documentation and training, access to configuration files and interfaces, diagnostic instrumentation, logging and alerting. Those of you who attended Chris Hills’ talk “Requirements are Required” earlier this year may remember his mentioning a company that installed airport runway lights then needed to patch the firmware, then they discovered they hadn’t put in anything to allow them to install a patch other than by visiting each light individually, removing it, applying the patch then putting it back. All this on a runway that was in near constant use.

Reliability may seem like a strange one as, of course, you want the system to be as reliable as possible. Unfortunately high levels of reliability are likely to increase the cost and time to market so a customer (or your own sales team) may settle for a less reliable system that is cheaper and/or gets to market more quickly.
Design Constraints

Design constraints might be connected to the hardware the software will be used on (e.g. a PC with 17 inch monitor and separate keyboard or a mobile phone with 4 inch screen of which a third is taken up by the onscreen keyboard), versions of support software (JVM, Browser, Flash &c that will be on the machine) or the valid values for a field.

For hardware they might include a size and weight limit or environments it must be able to work in (there’s no point having something that is supposed to be hand held and used outdoors in all weathers that weighs 30kg and breaks down if it gets damp).

There may also be constraints imposed by parallel running with an existing system during the implementation phase so the new system must not interfere with the old.
Reporting/MI/Printing

What information are you going to want to extract from the system?
How do you want it summarised and presented?
Do you want to report from the transactional data or implement a DataMart?
How do you want it output?
Where do you find them?

- Talk to the users
- Talk to the support teams
- Talk to the enterprise architects
- Talk to the lawyers
- Read industry journals
- Read the Standards
- ...

Talk to people, read up. Same sort of process as functional requirements.

Most of them will come from the people who will be using the system and operating the business process. Or at least the managers of the people who will be using the system and operating the business process. If this is something new then you can talk to people who might be using something similar or might be potential customers for your product.

Support teams and enterprise architects can tell you the environment the system needs to fit into, other systems it needs to talk to, the support interfaces needed, monitoring tools used &c. Support teams in particular may be able to tell you the issues that cause both the users and themselves the biggest problems.

Lawyers can help you to understand any legal implications, data retention or auditing requirements.

Industry journals can tell you what others are doing for similar systems.

Standards quite often impose Non-Functional Requirements, especially around security, auditability and maintainability.
Talk to people, read up. Same sort of process as functional requirements.

Most of them will come from the people who will be using the system and operating the business process. Or at least the managers of the people who will be using the system and operating the business process. If this is something new then you can talk to people who might be using something similar or might be potential customers for your product.

Support teams and enterprise architects can tell you the environment the system needs to fit into, other systems it needs to talk to, the support interfaces needed, monitoring tools used &c. Support teams in particular may be able to tell you the issues that cause both the users and themselves the biggest problems.

Lawyers can help you to understand any legal implications, data retention or auditing requirements.

Industry journals can tell you what others are doing for similar systems.

Standards quite often impose Non-Functional Requirements, especially around security, auditability and maintainability.
A lot of the time your non-functional requirements are buried in your functional requirements, the how it does to the what it does.

If a step in a process is to collect some information from the user how will it do that? If a form how will it look? What are the valid values for fields?
Functionally most of those will work as a swing, only one is what the customer wanted. The difference is in the Non-Functional Requirements.
What should you do with them?

- Non-Functional Requirements are:
  - As important to delivery as Functional
  - Validated as thoroughly as Functional
  - Vital for proper testing
  - Included in requirements documentation
  - Signed off
  - Incorporated into the design

Record them, validate them, decide how to test them and get them signed off then included in the design. Don’t figure you’ll add them in later.
This presentation is available on my SlideShare (can be accessed via my LinkedIn profile, see first slide for the URL). I welcome connection requests on LinkedIn and never IDK.

I do encourage anyone on LinkedIn who is a member to BCS to join the members group on LinkedIn: https://www.linkedin.com/groups?home=&gid=80608&trk=anet_ug_hm

If you have anything to do with Oracle (DBA, Developer, Business Analyst, Applications DBA etc) and live in the Midlands you may want to check out the Midlands Oracle Users Group: http://oraclemidlands.com/

At the end of the talk one of the audience asked me about how to find out more, I do recommend the S & J Robertson book “Mastering Requirements” although I do find it rather academic and dry. “Business Analysis” by Debra Paul et al is a good introduction. Both of these can be bought through Amazon and most other major booksellers, they are also both on Safari if you have an account.