Non functional requirements

Content

Definition

the semantic definition would be "*any requirement that is not functional*". However, as (for example) data requirements are clearly not functional requirements, and also are clearly not non-functional requirements, this definition is clearly not sufficient!

The fact is that non-functional requirements are any requirements that cannot be categorised in to Functional, Data or Process requirements. We only know

- *They **are** requirements*
- *They are not functional, data or process requirements*.

In that sense they are a 'catch-all' bucket for all those requirements that cannot be categorised in any other way.

Back to contents

Discussion

The International Institute of Business Analysis (IIBA) Business Analysis Body Of Knowledge (BABOK) v1.6 suggests that "Quality of service requirements are most often used to describe some of the attributes of the system or system environment. These requirements are constraints on the solution. They are also known as non-functional requirements."

This may be a useful approach but the important thing is not to get worried about whether a requirement is a non functional or not. If you can't categorise it any other way then categorise it as non-functional. The thing to really worry about is not mis-categorising a requirement, but missing a requirement altogether!

There is an important attribute of non-functional requirements that does differentiates them from other requirements and that is they are optional: Not all solutions will need to specify all *categories* of non-functional requirement. On the other hand, all solutions will need a specification of their functional, data and process requirements.

So what is a requirement that is not functional, process or data? There are a significant number and this is *not* an exhaustive list of categorises:

- **Accessibility Requirements**
- **Accuracy Requirements**
- **Auditing and Reporting Requirements**
- **Availability Requirements**
- **Backup and Recovery Requirements**
- **Capacity Requirements**
- **Compatibility Requirements**
- **Concurrency Requirements**
- **Configurability Requirements**
- **Error-Handling Requirements**
- **Legal and Regulatory Requirements**
- **Licensing Requirements**
- **Localizability Requirements**
- **Maintainability Requirements**
- **Performance Requirements**

Non functional requirements

- **Precision Requirements**
- **Redundancy Requirements**
- **Reliability Requirements**
- **Scalability Requirements**
- **Security Requirements**
- **Stress Requirements**
- **Supportability Requirements**
- **Throughput Requirements**
- **Etc, etc, etc!**

Note that non-functional requirements tend to be the 'ilities" of the system aka availab**ility**, accessib**ility**, etc. Why so many and why the "etc, etc, etc!"? Because as business analysts we need to define all the requirements for a solution and while there are some categories of requirements common to all solutions (functional, data and process) there are a set of categories of requirements that are not common to all solutions and will vary by the particular circumstances of the change project being worked on. As no two change projects ever have the same set of particular circumstances it is likely that there will always be exceptions to the general categorisations of requirements that never-the-less need to be analysed and carried in to design. That is not to say there are not a set of "usual suspects" of non-functional requirements that most change projects will *probably* need to analyse and these are:

- **Availability Requirements**
- **Capacity Requirements**
- **Performance Requirements**
- **Reliability Requirements**
- **Security Requirements**

For example, a solution that provides the user the *functionality* to manage sales through – in part – capturing customer *data* in the 'create customer' *process* may also need to specify some non-functional requirements defining
- Who is allowed to operate the process?
- Are there any restrictions on the data that users can see? (Can users see full credit card numbers for example).
- How many of these transactions per hour must the solution be capable of processing?
- When can user operate this process?
- How many users can operate the process at the same time?

This solution may NOT need to specify
- Licensing requirements (it could be an in-house development).
- Configurability requirements (all users might have to run the process in the same way).
- Scalability requirements (the scope may be fixed user group with no predictions of growth).

A final point: there is no one single correct way to document non-functional requirements. The way you choose should support your objectives of making sure that all requirements are designed in to the solution and whatever way that achieves this best is the correct way. We will look at some of the general issues in this area in the section on documenting non-functional requirements, but typically there are different factors to consider for different non-functional requirements and these will be outlined in the articles looking at each of the non-functional requirements we have identified.
Back to contents

How to find non-functional requirements
Let's face it: it is unlikely that we will sit down with your users and say "today we are going to analyse non-functional requirements." So the question is how do you set about uncovering these non-

functional requirements? You have to 'uncover' them because they already exist whether you and your users know about them, so you just have to find them!

The good news is you have an advantage over your users: You already know that some non-functional requirements do in fact exist (and need discovering) and roughly what categories of non-functional requirements there are. It would be nice if you could just work through a list with your users:
"Ok, please tell me when you need to be able to run this process." (Availability).
"Now tell me how many times you expect to run this process per hour." (Throughput).
"Now tell me how long we need to retain the data that this process generates." (Auditing)
Etc.

Perhaps that would work! That is for you to judge based on what you know about your users. However, what we must recognise as Business Analysts is that humans are not usually quite so methodological. What they tend to do is tell us Business Analysts a whole bunch of things all in one go: "Hey, this process is terrible: it takes 10 minutes to process an order! What we need is a system that can retrieve the customer details as soon as the operator enters the customer name and address – oh and the operator must validate that data with the customer – and we need to keep the data for 6 years (because of financial regulations)"

As Business Analysts we need to categorise all the information they have just given us in to a structure that allows us to analyse their requirements:
1. There is an *objective* to reduce the time taken to process an order from 10 minutes to…(well, we don't know and we would need to ask).
2. There is a *process requirement* to retrieve customer details.
3. There is a *non-functional requirement* to retrieve customer details as soon as the operator enter the customer name and address.
4. There is a *process requirement* to validate customer details.
5. There is a *non-functional requirement* to keep customer details for 6 years (starting when? We don't know and would have to ask.)
6. There is an *objective* to maintain compliance with financial regulations (which ones? We don't know and would have to ask to see what other objectives we might have in order to maintain compliance).

As Business Analysts we also know that for the above processes (retrieve customer details and validate customer details) and data (customer details) there will almost certainly be the following set of "the usual suspects" non-functional requirements as well as any we have already discovered:
- **Availability Requirements** – when do they want to be able to operate these processes?
- **Capacity Requirements –** how many times do they operate the process per day? How many customers are there (over 6 years)?
- **Performance Requirements** – they asked for the data to be returned "as soon as" the operator entered the customer name and address. Does this really mean instantly (which could have a significant impact on costs of the solution) or is there a time period that they would deem acceptable (e.g. 1 second).
- **Reliability Requirements** – do the users really need the process and data to be available 100% of the time? Again, cost impacts if they do and if not then what would they be prepared to accept (e.g. is it acceptable if the system goes down for no more than 1 day per year).
- **Security Requirements** – who can use these processes? Just 'operators' or can 'system administrators' (for example) use them as well? Can all operators access any customer details or are there some that they are not allowed to access? E.g. operator teams who manage customer segments – perhaps they should not be allowed to see other operator team customers?

Then we might start thinking about *other* candidate non-functional requirements: perhaps the operators in different teams need to **configure** what information gets retrieved for their customers…?
Back to contents

How to document non-functional requirements

It depends.

It depends on what **type** of non-functional requirements you are documenting and at what **level** they apply.

The basic **types** of non-functional requirements are process, data or both.

The basic **levels** that non-functional requirements can be applied at are
- Whole solution
- All automated (or all manual) components of the solution
- Functional requirement
- Whole process
- Any level within a process hierarchy
- An individual process step
- All data
- An individual data entity
- An individual attribute on an entity

Regardless of type and level that they are documented at, non-functional requirements all involve a definition of what they are and some values (targets) they must achieve.

We can imagine a matrix:

<u>Type</u> Level	Process	Data	Both
Whole solution			
All automated or all manual components			
Functional requirement			
Whole process			
Any level within a process hierarchy			
An individual process step			
All data			
An individual data entity			
An individual attribute on an entity			

At each intersection can be any combination of non-functional requirements. This is – potentially – a lot of non-functional requirements!

We can restrict the number we document by applying 2 rules:

Non functional requirements

1. only document the non-functional requirements that apply to the solution – not all solutions will need to specify all non-functional requirements.
2. Given that we want to document the minimum number necessary to define the non-functional requirements it follows that we should document the non-functional requirements at the highest row we can based on the principle that *a non-functional requirement applies to all the components it logically contains.* It also follows by the same rule that if we can put it in to the 'both' column we should.

Examples:
- "the solution should be available to users during normal working hours" could apply to the whole solution for both processes and data (top right in the matrix).
- "full history of all changes to the date of birth of a customer must be maintained" applies to one attribute used by process steps within a processes within functional requirements within the automated components within the whole solution.

So, back to the question of how to document functional requirements.

We have seen that non-functional requirements can be documented in text as they all involve a definition of what they are and some values (targets) they must achieve.

That text needs to written against the highest element within the matrix they can be incorporated in to and here is a table showing the analysis deliverables that could be used to document different types of non-functional requirements at different levels:

Type / Level	Process	Data	Both
Whole solution	Terms of reference or Requirements Spec	Terms of reference or Requirements Spec	Terms of reference or Requirements Spec
All automated or all manual components	Terms of reference or Requirements Spec	Terms of reference or Requirements Spec	Terms of reference or Requirements Spec
Functional requirement	Requirements Spec or Requirements catalogue	N/a	N/a
Whole process	Process spec	Process spec	Process spec
Any level within a process hierarchy	Relevant level process spec	Relevant level process spec or Entity spec or Attribute spec	Relevant level process spec
An individual process step	Process step spec	Process step spec or Entity spec or Attribute Spec	Process step spec
All data	Process spec	Data model overview	Process spec Data model overview
An individual data entity	Process spec or Entity spec	Entity spec	Process spec or Entity spec
An individual attribute on an entity	Process spec or Attribute spec	Attribute spec	Process spec or Attribute spec

Non functional requirements

smart·BA

Suppose you have different names for your analysis deliverables or maybe different analysis deliverables? You should still apply the rules of documenting the non-functional requirements you need to at the highest level you can, regardless of the analysis deliverable they end up in.
Back to contents

Example non-functional requirements

Non functional requirement category	Typically applies to Non-functional Type	Example
Accessibility Requirements	Process	Help text will be provided in English, French and German.
Accuracy Requirements	Both	Process: All comment fields will be spell checked. Data: Date of Birth must be in the past.
Auditing and Reporting Requirements	Both	Process: A record of which users access or try to access process "Update Customer" is required. Data: A record of which user updates the Date of Birth attribute is required.
Availability Requirements	Process	Process "Update Customer" is available 08:00 to 18:00 daily excluding Sundays and Public Holidays.
Backup and Recovery Requirements	Both	Process: All processes can be made available after unplanned system downtime within 1 working day. Data: All data will be backed-up daily.
Capacity Requirements	Both	Process: Up to 500 users in total can use "Update Customer". Data: Up to 1,000,000 customers can be stored.
Compatibility Requirements	Both	Process: "Update Customer" can integrate Word 2003 onwards. Data: Customer data can be exported in XML format.
Concurrency Requirements	Process	Up to 300 users may be using "Update Customer" at any one time.
Configurability Requirements	Process	"Update Customer" users may choose whether to display previous name the customer has been known by.
Error-Handling Requirements	Process	In the event of the user cancelling or quitting the process "Update Customer" any changes made by the user will be reversed.
Legal and Regulatory Requirements	Both	Process: The user must confirm that they have notified the customer of the updates they have made before saving changes made during "Update Customer". Data: All changes to Customer data will be held for 6 years from the date of change.
Licensing Requirements	Both	Process: The "Update Customer" process will be licensed for 300 concurrent users. Data: The Postcode Address File will be licensed for 1 year starting each April.
Localizability Requirements	Both	Process: For American users of "Update Customer" all dates will be displayed in

Non functional requirement category	Typically applies to Non-functional Type	Example
		MM/DD/YY format. Data: For American users all currency will be stored as USD.
Maintainability Requirements	Both	Process or data: Changes required by law will applied at least 3 months before the law becomes enforceable.
Performance Requirements	Process	During the process "Update Customer" system responses should be no more than 1 second.
Precision Requirements	Data	Time of changes to data must be recorded to the nearest second.
Redundancy Requirements	Process	In the event of an unplanned exist from "Update Customer" the user can choose to restore the update to the customer they were working on at the time of the event.
Reliability Requirements	Process	"Update Customer" will be available to users 98% of normal working hours.
Scalability Requirements	Process	Up to 200 new sites per year may start to use "Update Customer".
Security Requirements	Both	Process: Only users holding the role "Customer Advisor" or "Supervisor" can access "Update Customer". Data: Only users holding the role "Supervisor" can update customer Date of Birth.
Stress Requirements	Process	Up to 10 users may access the customer details concurrently during "Update Customer".
Supportability Requirements	Process	The Customer Advisor Help desk will support users of "Update Customer" from 09:00 to 17:00 daily on weekdays only excluding public holidays.
Throughput Requirements	Process	"Update Customer" may apply up to 3,000 updates per working day.
Etc, etc, etc!	Both!	Process: Blah, blah, blah! Data: Ya-de-ya!

References & further reading

1. Business Analysis Body of Knowledge, Release 1.6 ©2006, International Institute of Business Analysis http://www.theiiba.org/. There is a v2.0 of this document.
2. Most books deal with Functional AND Non-Functional Requirements such as "Writing Better Requirements" by Ian Alexander and Richard Stevens (Paperback - 17 Jul 2002)
3. There are some specialist books on non-functional requirements such as "Methodologies for Non-functional Requirements in Service-oriented Architecture" by Junichi Suzuki (Editor) (Hardcover 2009) or Non-functional Requirements in Software Engineering (International Series in Software Engineering) (Hardcover) by Lawrence Chung, Brian A. Nixon, Eric Yu , John Mylopoulos (1999)

smart·BA