Functional Testing Heuristics

A Systems Perspective

Joris Meerts, Testing References

In functional testing it is tempting to use test techniques to cover the paths through a program or function based on specifications. It is also tempting to assume that once these paths or combinations have been covered functional testing is more or less complete. By the heuristics displayed below I intend to show that functional testing can - and perhaps should - include a large number of system aspects that are not regularly stated explicitly in requirements or in the design. I am certain that with these heuristics or points of view a number of interesting functional aspects of a system can be highlighted.
This list is derived from my own experiences in testing a range of software products. I am also indebted to James Bach, who wrote the SFDPOT heuristic, and Elisabeth Hendrickson, James Lyndsay and Dale Emery, who wrote the Test Heuristics Cheat Sheet.

Heuristic	Definition	Description
Sequence	Order of succession	Functions can be executed in a sequence. Vary the execution sequence of the functions, also in situations where it does not seem logical to execute the functions in that sequence.
Concurrence	The simultaneous occurrence of events or circumstances	Cause things to happen at the same time. For example in a web store try to purchase an item and at the same time set the status of that item to 'unavailable'. In this case that would require two users acting upon the same set of data.
Confluence	A coming or flowing together, meeting, or gathering at one point	Think of a system as a set of flows that diverge and meet. Interesting things may happen where flows meet.
Synchronization	To represent or arrange (events) to indicate coincidence or coexistence	In larger systems information may be stored in several places and has to be synchronized. Synchronization can be done in batch jobs, on demand or online. Test what happens when information is synchronized or not.
Share	To partake of, use, experience, occupy, or enjoy with others	Functions may share data resources or, for example, hardware resources. Test what happens when functions require or manipulate the same things at the same time.
Interaction	Mutual or reciprocal action or influence	Many functions interact. They pass on results, wait for eachother or are called by another function. Learn the ways in which functions interact and test the interactions.Funtions ins systems usually also have interfaces and therefore interaction with the outside world.
Continuity	Uninterrupted connection, succession, or union	Continuity is often assumed in the functioning of a system but in real life continuity, or uninterrupted functioning, might not always be the case. Break continuity at interesting points in the system.
Hierarchy	A graded or ranked series	Not all functions are equal. It may be that some functions take precedence over others or that functions are executed according to a strict hierarchy or organisation. Use or break the hierarchy to manifest faults.
Priority	Superiority in rank, position, or privilege	Priorities in systems may depend on a lot of things and may be shifting continuously. Ask yourself which functions have priority over others at a certain point and use that information.
Dependency	Determined or conditioned by another	Systems usually contain a lot of depencies of different forms. Some of these forms are mentioned in this document. Dependencies can easily be exploited to find faults. The more dependencies, the merrier.
Repetition	Renewed or recurring again and again	Execute a function of a set of functions more than once, perhaps many times successively. Execute the same function a number of times with slight variations.
Loop	A series of instructions (as for a computer) that is repeated until a terminating condition is reached	Systems and funtions contain loops and maybe loops within loops. Shift your mind by thinking of functions and systems in terms of (feedback) loops (systems thinking) and see where it leads.
Parameter	Any of a set of physical properties whose values determine the characteristics or behavior of something	Many functions are parameterized, meaning that the behaviour of a function is governed by (external) parameters or settings. Find out how the settings of a function can be manipulated or are manipulated by the system.
Prerequisite	Something that is necessary to an end or to the carrying out of a function	Some systems or functions require certain preconditions. More often than not these preconditions are silent assumptions. A web application may, for example, require a certain browser. Or a system may require certain system settings, connections or versions of other software programs installed on that system. Tinkering with the prerequisites may lead to some interesting findings.
Configuration	Relative arrangement of parts or elements	Many things in a system can be configured, many more things than you'd guess at first sight. Testers usually do not go too deep into configuration settings but configuration may matter a great deal for some specific functions. Learning about the configuration of a system can reveal relevant things.
Rule	A prescribed guide for conduct or action	In many systems there is some degree of separation of business rules and functionality. There is often a blurry line between what is governed by rules and what is governed by the function itself. Think of priorities, preconditions and parameters. Look for things in this area.

Heuristic	Definition	Description
Customize	To build, fit, or alter according to individual specifications	Customization of functions will occur when the system has users who have different roles. This is always interesting. More often than not users can also customize systems to their own needs and thus move away from default values.
Constraint	The state of being checked, restricted, or compelled to avoid or perform some action	Constraints, for example in the form of value boundaries, can be found all across systems and functions. It may be interesting to see what happens at boundaries at different levels of the system. Constraints also tell us something about the capabilities of the system and what happens when we stretch those capabilities. Another way of looking at it is that the test environment is a constraint.
Resource	A source of supply or support	A resource can be anything that is used by the system. Commonly the system uses processors and memory. What happens when those resource change or become less accessible?
Access	Freedom or ability to obtain or make use of something	Functions have, and may require, access to other functions, data sources, hardware etc… The capabilities of a function to access things may change.
Lock	To fasten in or out or to make secure or inaccessible	Locks are often applied to data in order to avoid manipulation of that data by other functions. Locks are usually, at one time or another, released. Investigate what locks do with the system.
State	Mode or condition of being	A systems is always in some condition or state. Try to identify conditions and try to find out how certain conditions affect functionality.
History	A chronological record of significant events	Systems have a history of things that happened in the past. Testers usually like to start from a clean sheet, a system without history. Users always work with a system that has a history. Try to see how the history of a systems influences its behavior.
Roll back	To reduce to or toward a previous level	Roll backs in a system are usually procedures that are executed when something has gone wrong. A roll back intends to 'reset' the system to a valid and functional state. But does it do that and how does a system behave after a roll back?
Restore	To bring back to or put back into a former or original state	Restore aligns with roll back but keep in mind the functions and the data that are affected by a restore. Is the whole systems restored to a former state or just parts of that system? What about the functions that are not supposed to be affected by a restore?
Refresh	To update or renew	The most obvious refresh function is the Refresh or Reload function in any internet browser. This may lead to some interesting observations. In some applications, not necessarily in web applications, there are functions that automatically refresh, or explicitly do not refresh, the content that is displayed. How does refreshing information affect other functions?
Clone	One that appears to be a copy of an original form	Cloning is a concept in software design. Among other things systems, functions and data sets can be cloned for a number of purposes. If you have a separate test environment that can be considered a clone of (parts of) a system. A view on one or more database tables could also be regarded as a clone. In testing with clones beware of discrepancies and adjustments to the system or function that was cloned.
Temporary	Lasting for a limited time	Many things in the system are not there forever and do not last forever. Common examples of things that are temporary are temporary tables that support, for example, the copying of information. But also files and objects may be temporary. One way to look at this is to find out if temporary stuff is cleaned up after use.
Trace	A mark or line left by something that has passed	Traces in software are important in functional testing. They can be found, among other places, in logging and audit trails. Use traces or trails for root cause analysis. But hey can also be used to gain more information about the inner workings of a system.
Batch	The quantity produced at one operation	In many systems large portions of data are processed in batches. The testing of batches and the functionality contained in batches may be a discipline in itself. As a functional tester you should be aware that there is functionality sheduled in batch operations that may do surprising things. It is not uncommon that batches run without the tester being aware of such processes. It is also not uncommon to switch off batch jobs in systems testing, leaving options open for suprises in a later phase of testing.
Void	Empty space	For example; in a database 'empty' is not the same as NULL. Functions may deal with empty values in an entirely different way than with NULL values. Explore how functions deal with empty fields.
Absent	Not present or attending	For many different reasons data, certain parts of data, functions, applications, resources and other things may be missing at one time. Examinate how functions handle missing things.

Heuristic	Definition	Description
Feedback	The transmission of evaluative or corrective information about an action, event, or process to the original or controlling source	Functions provide feedback in one form or another. In testing this feedback is essential in verifying the result of a test. Beware that the feedback provided by a function may not reflect the actual results of that function. Whenever you use feedback to verify functionality, you may want to use other sources to verify the feedback.
Saturate	To treat, furnish, or charge with something to the point where no more can be absorbed, dissolved, or retained	Saturation may cause functional errors in systems at many levels. Try to evaluate functional behavior with regards to large amounts of input data, opening many screens in a GUI, using more than one user at the same time or other aspects that can vary form zero to one to many.
Sort	A group set up on the basis of any characteristic in common	Sorting algorithms are everywhere in a system. They may or may not be written down in specifications. Start from the point of view that data is always offered to a function or operation in a sorted order. Try to break the sorted order, mess up the order or try to sort in different ways and see what happens.
Scale	A distinctive relative size, extent, or degree	Size does matter. Scale is usually important in non-functional testing such as performance testing. But by changing the scale of certain aspects of a system functional errors can be found. Scaling for example the number of records in a data collection or installing the system on much slower or quicker hardware may reveal the limitations of certain functions.
Corrupt	Characterized by improper conduct	A system will, after having been used for a while, contain corrupt data or settings. As testers we have a tendency to test with clean data. It will be hard to assess in what ways data can become corrupt. Yet in our functional testing strategy we may want to test with corrupt data.
Integrity	The representational faithfulness of information to the true state of the object that the information represents	A systems is a model, a representation of aspects of a real world. Explore how objects or, for example, data collections, in a system evolve due to the operations that are executed. Try to find out if these objects represent aspects of the real world over time and what may corrupt them.
Invoke	To put into effect or operation	A function may be invoke in different ways. When using a GUI the most obvious distinction we can make is invoking functions by using strokes on a keyboard or using mouse clicks. Functions may behave differently based on such approaches.
Timing	Placement or occurrence in time	Time is always a part of a system. In functional testing we often tend to forget that the time at which an operation is executed may contribute significantly to its functioning and its result. Find functions that could to some degree depend on, for example, system time or internal clocks. Also try to travel in time.
Delay	To stop, detain, or hinder for a time	Delays are part of many systems. In some systems delays in, for example, response may be deliberate. But a unplanned delay may also be the result of operations running on a system. Use delays to explore what happens if you execute certain funtions within that delay. Also explore delays from a systems thinking point of view to see how balances in system evolve.