System Breakdown: What it is, and why you should already be doing it

By Vasily Shishkin and Alyson Gombos

System Breakdowns are an integral part of thorough testing, but they are vastly underutilized in the IT industry, to the detriment of many development projects. Because of the intensive nature of the system map they provide, performing a system breakdown leads to a deeper understanding of the system in production, making both development and testing easier on everyone involved.

What is a system breakdown, and why is it important?

System breakdown is by far the best approach for outlining a system under test, and as said above, it’s one that’s vastly underused and under-discussed. It’s the process of splitting a system into separate blocks and a list of their integrations, which are in turn broken down recursively into smaller and smaller units. These are then tested both on their own and with regard to their integration within the system. The result of the system breakdown is a map of the system’s components from the very broad level to a very specific one, almost down to specific test cases.

A system breakdown also is not necessarily a static thing: it is a breathing process, one which can morph and change as the project it relates to does. Sections may need to be added, removed, or connected in new and different ways over the course of the testing and development effort. At the lowest level, the system breakdown can be nothing more than a list of test cases or exploratory test charters.

However, system breakdowns are not only useful from a testing standpoint; they can be equally relevant, perhaps even more so, to development and design staff. The best-case scenario would be for a system breakdown to be provided about 80-90% finished to the testing team by the designers. This aids in breaking down the testing process while still leaving room for the addition of other information gleaned by the testing team.

When should you do a system breakdown?

This is a process that should be done at the beginning of every relevant process ever. It is systematic and methodical, making it easier to divide the work amongst the testers while also providing a more thorough, demonstrative understanding of the system. System breakdowns don’t need to only relate to the testing effort, either; they don’t need to deal simply with modules and features. Other common uses for a system breakdown, aside from the functional testing aspects, could be a breakdown of business processes, testing steps, user data, and more. Any system breakdown can serve as a baseline for defining testing scope and development, from the very first steps of the process onward. It can be used in relation to scripted testing, exploratory testing, and any combination of the two.

Some characteristics of a successful system breakdown are that they are detailed enough to give a microcosmic idea of the system itself, but not too detailed that the each individual module takes only a
few minutes to run. A good system breakdown will also be totally unambiguous and completely clear; this is indispensable to locating problems directly at their source.

**The downsides of a system breakdown**

A thorough system breakdown has many positives to a project, but what are the negatives? It’s important to remember that this process won’t solve all the problems that you may be testing for, as it does not encompass the entire scope of the testing process. Like we mentioned above, things like usability testing are not included in a system breakdown and must be considered a separate effort. Working in this manner should also require testers to make a conscious effort to consider usability, as it is tempting to focus only on the functionality of the separate components without considering what the users’ experience with the whole product will be. A system breakdown provides a good functional checklist for coverage, but using it as a test plan tends to be inefficient because it doesn’t account for or address negative test cases, which can be just as important to the testing process as positive ones.

The system breakdown should be an integral part of the testing process, but instead it’s often neglected – and by “often” we mean “almost always.” It may come across as overly complicated or seem generally unnecessary to some test managers, but its effects on the testing process are incredibly beneficial to both individual testers and the testing effort in general. There is seriously no other method for ensuring a well-rounded system post-testing as creating a system breakdown early on and assessing the process to ensure that the negatives discussed above are not present in the testing environment.