Troubleshooting Theory

# Lesson overview.

In this lesson, we will cover:

* The importance of having a methodology.
* Six-step troubleshooting methodology.

# The importance of having a methodology.

Having a troubleshooting methodology is important, especially in the IT arena. I like this quote by Stone Gossard from Pearl Jam, "My methodology is not knowing what I'm doing and making that work for me." While this is a great quote, not everybody can do this, especially in the technology world.

Due to the complexity of modern computing systems, a wise technician will have and follow a troubleshooting methodology. A formal methodology gives the technician a starting place and a logical sequence of steps to follow. If you do not have a methodology in place, you are much more likely to waste time and effort and create frustration—not only for yourself but also for your end user.

# Six-step troubleshooting methodology.

CompTIA recommends using a six-step troubleshooting methodology. The first step in their methodology is to identify the problem. Step two is to establish a theory of probable cause. The third step is to test that theory of probable cause. Step four is to develop an action plan and implement that plan. The fifth step is to verify complete system functionality. And step six is to document the process. In the section below, each of these steps is covered in more detail.

## The six steps of troubleshooting.

### 1. Identify the problem.

The first thing that you need to be aware of when troubleshooting a problem is that the symptoms are not the problem. When troubleshooting, it is critical that you actually identify the underlying problem—what’s actually causing the symptoms to manifest themselves. To do that, you should question the user. Ask detailed questions about when the symptoms occurred and why they may have occurred. If the user can re-create the issue for you, this can be extremely helpful. Determine what, if anything, has changed, which may have caused the problem. Importantly, before you take any action, make a backup copy of the system, so that you can preserve everything as it is.

#### Highlights:

* Question the user.
* Remember that the symptoms are not the underlying problem.
* Determine what has changed.
* Make a backup of the system before moving on.

### 2. Establish a theory of probable cause.

Once you have identified the base problem that is causing the symptoms, you will establish a list of probable causes. Once you have a list of all of the probable causes, use your technical knowledge to prioritize that list. Your list should have the probable causes listed from most likely to least likely. Incidentally, be sure to question the obvious. For instance, if the symptom is a power situation at the workstation, is the power cord plugged in and, if it is plugged in, is the outlet actually getting power? Additionally, if no probable cause can be determined, you will need to escalate the problem to a higher level.

#### Highlights:

* Make a list of probable causes.
* Using your knowledge, prioritize the list.
* Question the obvious (i.e., if the symptom is a power issue at the workstation, first check to make sure that the power cord is plugged in).
* Escalate to a higher level if a probable cause cannot be determined.

### 3. Test probable cause theory to determine actual cause.

Once you have established your theory of probable cause, you should take a moment to consider whether or not you can troubleshoot the issue on your own, or if escalating it to a higher authority is called for. If it falls within your capabilities, you will need to test your theory to determine if it is, indeed, the actual cause. Your theory was created from the most likely probable cause, so you need to determine how best to test it. If your theory is confirmed, you will move on to the next step. If the theory is disproved, you will need to go back to step two or step one, as needed, and work your way through the troubleshooting methodology.

#### Highlights:

* Create your theory from the most likely probable cause.
* If the theory is confirmed, move on to the next step.
* If the theory is not confirmed, go back to step two or step one (if needed).

### 4. Establish an action plan and execute the plan.

Once you have determined the actual cause by testing your probable cause, you will need to establish an action plan and then execute that plan. Simple problems will probably only need simple plans. However, if it is a complex problem, you may need to write out the plan so that you can be sure to execute it correctly. This is another opportunity to escalate the problem to a more senior level if necessary.

#### Highlights:

* Simple problems probably just need simple plans.
* Complex problems may need written out action plans.
* Escalate to a higher level if required.

### 5. Verify full system functionality.

After you have executed your plan, you will need to verify that the system is fully functional. If everything works—that’s great. Based on your findings and the issue, you may find that you have the opportunity to implement preventative measures so that the problem does not occur again. If full system functionality has not occurred, you will need to go back to step one and continue to work through the troubleshooting methodology.

#### Highlights:

* If everything works, great! If applicable, use your findings to implement preventative measures.
* If not everything works, go back to step one.

### 6. Document the process.

Once everything is fully functional, documenting the process becomes important. This is where you document findings, actions, and outcomes. When the problem occurs again, there will be information available to walk someone through the means of troubleshooting and resolving the issue.

This documentation also captures a history of equipment and users so that perpetual issues become known and recorded. An important aspect of this is that both positive and negative outcomes should be documented. This can save time during future troubleshooting and prevent others from taking the same missteps you may have taken.

#### Highlights:

* Capture your findings, actions, and outcomes.
* Issues that need to be troubleshot may occur again.
* Documentation provides a history of equipment and users so that problem issues are known.

# What was covered.

## The importance of having a methodology.

A methodology gives a systematic approach to solving IT problems. It reduces wasted time and frustration for both the technician and the end user.

## Six-step troubleshooting methodology.

Identify the problem; establish theory of probable cause; test the theory; establish a plan of action and implement it; verify system functionality; and document everything.