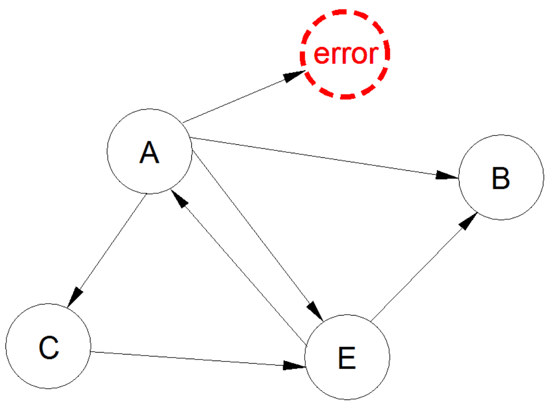
Fixing A Complex System, version 1.3

By R. G. Sparber

Protected by Creative Commons.[1]



I wanted to share with you my general approach for debugging a complex system. This approach is certainly not original or even earth shaking. It just works.

If the failure is resolved by replacing obviously damaged parts, then a technician can complete the job quickly. But if the root causes of the failures are not clear, sometimes even the best technician is unsuccessful. In those rare cases, this engineering approach should work.

In general I have found that the following approach works but it will be slow going. However, moving methodically to success is better than never getting there.

1. Identify people who run the system and invite them to be a part of the trouble shooting team. They likely have insights that even the designers of the system do not possess.

2. Establish that the system is actually broken. If the system never worked, the rules are vastly different. If it used to work and now doesn't, we are looking for one or more things that changed such that the system no longer works. Could it be that the operator changed and the root cause is a person and not the system?

3. If the failures occur often, success is likely. If the failures rarely show, the job becomes more difficult. If you can't see the failures, you can't solve the problems.

4. Until proven wrong, assume there is more than one source of failures. Never assume that the part just taken out of a sealed box is good.

5. How should the system work? This is the centerpiece of finding what is broken. Start with an overview and keep digging. Don't stop until all detail is understood and verified by direct measurement. In most cases the act of learning how something should work leads you to find what isn't working. What are the stimuli? What are the expected responses. Be mindful of the boundaries of the system that fails. Could it be that these boundaries are wrong? Maybe the problem is the interaction of two systems and not just the health of either one. In this case, enlarge the boundaries and define a new system.

6. Making a failure occur more often is just as valuable as making it occur less often. Either way might point you to one of the failure sources. Vindicating a suspected error source is a step forward.

7. Take detailed notes when doing experiments on the system. Review these notes periodically to verify the team is proceeding logically. I find it helpful to do these reviews away from the system and in a quiet place.

8. If experts exists outside of the team, ask them to attend periodic reviews. Often the simple act of preparing for these reviews will shed light on the problem.

9. How do you know the system is fixed? First of all, you should clearly understand how something should be working and observe that it isn't working that way. Say that a new part causes a behavior to work as expected and the overall system stops having failures. It is good practice to put back the old part and verify the failure returns. That is the ideal case and you can declare success. As the rate of failures decreases, it becomes harder to know the device has been fixed. Experience has taught me that Murphy's Law is always present. Just when you think you have fixed the problem, you are unable to reproduce the failure with the old part. Time to take a break and go back to learning how the device is supposed to work.

10. If a number of parts have failed, it is possible that one of them caused the others to fail. Further study should show how they are related. In these cases, do not put the source of these failures back in the system.

Beyond the technical effort of finding the root causes of the failures is the political side. The debugging process can take time, often far more time than the people in power were prepared to spend. Keep an eye on who is paying the bills. They must understand your general approach and support it. Of particular need of explanation is the concept outlined in item 6: making things worse is as useful as making things better.

In some rather frustrating cases, the owner of the system will give up and replace everything. That might be the most cost effective approach but it certainly is not satisfying to me and nothing will have been learned. Where will they be the next time the system fails?

Thanks to John Herrmann for reviewing the list and to Paul Thompson for pointing out the political aspects of fixing complex systems. Thanks to Jerry Feldman for pointing out the case where one bad part fries a number of other parts.

I welcome your comments and questions.

If you wish to be contacted each time I publish an article, email me with just "Article Alias" in the subject line.

Rick Sparber
Rgsparber.ha@gmail.com
Rick.Sparber.org