Problem Analysis
Guidelines for Software Specification Questions

Kepner Tregoe

Questions to Ask to State the Problem

The Problem Statement needs to be clear and concise, using just one object and one deviation. If there is more than one deviation then there are multiple problems, and these should be dealt with separately. Use **Situation Appraisal** to separate and clarify the problems or concerns, and then prioritize them to enable a decision to be made as to which problem to work on first.

Things to Watch Out For

Check that the concern is really a problem.

- Is there a deviation between should and actual?
- Is the cause unknown?
- Do we need to know the cause to take action?

Look for the word “and” in the Problem Statement. You can only include a group of related objects or related deviations if you are convinced that the same cause is operating for everything you include!

Some deviations are not problems because they are desirable, but you may still want to find the cause so you can keep them going, or make them happen in other situations. Positive deviations may have undesirable consequences in other areas. Apply “Extend the Cause” thinking to clarify. Verify that the *Should* is understood. Use data on the ‘should’ and ‘actual’ to formulate a specific statement about the deviation. Ask why the deviation occurred. If the cause is known, ask why the cause happened and repeat until a deviation without a known cause is revealed. Verify the need to know the cause in order to take effective action—a workaround may be a higher priority.

Avoid jargon!

The Problem Specification

Remember the intent behind the questions. They are intended to collect actual data about the problem against which possible causes can be evaluated; you will rarely solve the problem simply by asking these questions. However, asking the customer these questions may trigger a train of thought which enables them to solve the problem!

The questions that are provided are **guidelines** for the types of questions that can be used. The intent is to use the questions that apply to your environment and modify the wording as appropriate, based on your products. The best questions should be developed by a content expert who is knowledgeable regarding KT process.

Question to the void. Ask ’turnaround’ and ’what else’ questions to ensure that you get the most complete answers. If the answer you get does not answer your question, acknowledge it and ask the question again.

 Copyright © 2005-2014 Kepner-Tregoe, Inc. All Rights Reserved.

Problem Analysis
Guidelines for Software Specification Questions

Kepner Tregoe

		KT Process Questions and Intent	Software-Specific Questions
WHAT	IS	***What specific object(s) has the deviation?*** *To specifically identify the object with the deviation.* Before beginning to develop a specification, ask questions to ensure that there is a cause-unknown deviation. The answer to the What question can be very lengthy since there are many dimensions to the product. Be as specific as you can (e.g., version, operating system, patches, etc.). If you are looking at a small part of a larger system, process, or application, describe only the part that has not met the 'should.' If there are any other systems or applications that are integrated, describe them in detail. Avoid the temptation to describe what you think may be the system or application that is at fault. Focus on the thing, component, or product that is exhibiting the deviation.	• Describe what you were looking at when you first noticed the problem. • What is the specific description of the problem that you are seeing or the task that you are trying to accomplish? • How do you know this is a departure from the expected performance? • Which system/process/software/application/component/function/method/part of the process is not working? • If unsure, which area of the product is failing? • Which OS, version, patch level, and platform? Which version, patch level of the integrated application or database? • Is this version an upgrade from a previous version or is it a new installation?
	IS NOT	***What similar object(s) could reasonably have the deviation, but does not?*** Identify the specific deviation on the object. What is (are) the most closely related thing(s) to the IS that you might expect to exhibit this same or a similar type of deviation, but in fact does not? Again, to be helpful, these must be facts as well: 1) That they could logically exhibit the same/similar condition AND, 2) in fact, do not.	• Which other similar objects/processes (with similar dependencies) are working well and do not have the specified problem? • Which other similar/related applications/functions/methods processes/parts of processes do not have the specific problem? (Consider OS, platform, or configuration.) • If you attempt the same task over the web or LAN interface, do you get the same problem? • Do other components have the same problem? • Did you find this problem in previous versions if this object/process is an upgrade? • Have you seen the application performing the same task without customization (if a customized version)?
	IS	***What is the specific deviation?*** *To specifically identify the deviation on the object.* What does the unacceptable condition look, feel, smell, sound, or taste like? Be specific—focus on describing the deviation from the 'should.' Avoid using labels for the type of deviation—try to describe it so that someone else would know exactly what it looks, feels, smells, sounds, or tastes like without seeing it. Include a picture or drawing, if appropriate. Error logs, screen shots, or system reports should be provided if they contain evidence of the deviation.	• What is the specific failure/error/problem? • If the customer had to show you the deviation, what would they show you? • What exactly is displayed on the screen/application, etc.? • What symptoms are present or errors are being reported? • What specifically is it doing that it's not supposed to? • What did you expect the application to do? • What is it NOT doing that it is supposed to? • What evidence do you have of the problem? • Describe the deviation from 'should' exactly.
	IS NOT	***What other deviations could reasonably be observed, but are not?*** Describe some other closely related unacceptable conditions you might expect to see on/with this type of object, but in this case, do not. Be sure not to indicate the 'should.' Consider only other similar types of deviations.	• What related failure/problem is not occurring in the component/area? • What other problems could reasonably be observed, but are not? • What functions, if any, of this application are performing normally? • What else could be going wrong with the application/function/method/process but Is Not? • Describe other closely related fault conditions that might be observed on this object, but are not.

 Copyright © 2005-2014 Kepner-Tregoe, Inc. All Rights Reserved.

Problem Analysis
Guidelines for Software Specification Questions

Kepner Tregoe

		KT Process Questions and Intent	Software-Specific Questions
WHERE	IS	***Where is the object when the deviation is observed (geographically)?*** *To locate the objects with the deviation as on a map.* Where specifically is the affected object when the deviation is observed? List all locations at which the object is and has been observed to have the deviation, not simply the first place it appeared. Be as specific as possible—use turnaround questioning to pinpoint exact location(s). The exact location is particularly important if you have a good 'is not' on the same site, or a similar site—same customer. Be sure to list all the locations where the problem has been seen. Be as detailed as possible. The answer to this question could start with "at the customer site on the …" Consider whether it is a production system, test system, etc.	• Where is the hardware (that runs the system/process/application) located physically? • If this is an application/process, please draw a diagram showing where it fits in to the whole system. • What is its environment? Which building (address), floor, room, rack, shelf, disk, network card, cable, network port, router port, LAN, server, client? • Where in the topology/network is the component failing? • Which customer, network, access level, and interface is experiencing the problem? • Where is the router in the network/system? • What is the specific area, location, user or users, group, machine and/or activity? • Where at that site? Which machines?
	IS NOT	***Where else could the object be when the deviation is observed, but is not?*** What are other locations at which the affected object (or a closely related object) has actually been and has not exhibited the defective condition? Again, be specific here—for each answer you got when specifying the IS, look for a logical IS NOT.	• Where else could the hardware be when the problem is observed, but is not? Name the places where you could reasonably expect to see the problem, but do not. • Is there a place where a similar setup exists, but you do not see the problem? Describe this location. • List other locations where the object, or a similar/same setup has actually been, but has not exhibited the same fault symptoms. • Where else is the application/function/method/process not failing? • Where are there users that do not experience this problem? What specifically are these user accounts, locations, etc.? • Are you capable of creating a similar environment where the problem does not exist? Describe the environment.
	IS	***Where is the deviation on the object?*** *To identify the specific locations on the object where we can see, hear, feel, taste, or smell the deviation.* What part(s) or facet(s) of the object is (are) affected with the deviation? Is it localized? If so, where? Be as specific as you can—provide specific measurements and/or physical descriptions if possible. This is sometimes hard for a software problem, but try and locate the part of the object affected by the deviation. You will likely have a variety of useful information, based on the type of problem. If we rephrase the question to "If you took me by the hand and led me to where you see the problem, where would you take me?" the answer can be "on the console," or "in the messages file," or "in the application log file," or "in the network analyzer printout," and at the same time the answer can be "disk c0t0d0s1." Both kinds of answers maintain the intent, and both are of use in resolving the problem.	• Where is the defect on the object? • Where is the problem in the software? Which module/component/function/screen? • Where in the process are you observing this problem? • In the process, which function/process step has the problem? • Where are you looking when you observe the problem? • Which specific line of the output are you reading that shows the problem? • Which specific screen of the browser/GUI are you looking at when you see the problem? • When decompiling or debugging, at what specific procedure or function do you see it fail?
	IS NOT	***Where else could the deviation be located on the object, but is not?*** What other part(s) or facet(s) of the object might you expect to be affected and in fact is (are) not? If the entire object is affected, is it possible that this type of deviation could be localized? If so, which part(s) or facet(s) would you expect to be affected, but in fact is (are) not? Describe all parts of the object that could reasonably be affected, but are not.	• Where else could the defect be located on the object, but is not? • Where else in the process could this problem be observed, but is not? • What process steps executed correctly since the application had the problem? • Which parts of the output could reasonably exhibit this problem, but do not? • Which screen of the browser/GUI could reasonably have this deviation, but does not?

 Copyright © 2005-2014 Kepner-Tregoe, Inc. All Rights Reserved.

Problem Analysis
Guidelines for Software Specification Questions

Kepner Tregoe

		KT Process Questions and Intent	Software-Specific Questions
WHEN	IS	***When was the deviation observed first (in clock and calendar time)?*** *To identify a specific point in time when we know we had a problem (deviation).* On what date and at what time on that date was this problem first observed? Be as specific as possible. Avoid the tendency to list events which seem to correspond with the timing of the problem—this could restrict your thinking when developing possible causes. The answer to this question can be crucial and can lead the customer to discover the answer. Push hard for specifics. If you cannot get an exact date and time, an approximate one is better than nothing. You'll know how painful it is scanning a 30MB log file for a deviation when you have no idea what time the problem happened. Having the time also enables you to ensure that the log files you request relate to the correct time period, and are therefore useful.	• What date and time did the customer/user first see the problem? • When exactly was the problem first documented or reported? • Did it start at all locations at the same time? When did it start at the other locations? • Is this a non-working new install or a startup problem? Has it ever worked correctly?
	IS NOT	***When else could the deviation have been observed first, but was not?*** On what other date(s)/time(s) might this problem have started, but in fact did not? The question determines how long the system or application has been running without a problem in its current configuration.	• When was the last time it worked correctly? • When else could this problem possibly have started, but did not? • When was the last time you checked, prior to that? • When would have been the next time you would have checked? • Has this application/function/method/process ever worked?
	IS	***When since that time has the deviation been observed? Any pattern?*** *To understand the frequency of the problem.* What is the frequency of occurrence of this problem on this object? Since the first occurrence, when else has it appeared? Be specific, using date(s) and time(s) at which the problem starts and stops each time. Refer to charts or logs if available. View for a long enough timeframe to identify patterns if they exist. Does it exhibit any of the following patterns? • Continuous: the problem is in existence every time the object is observed; • Periodic: the problem comes and goes with a predictable frequency; • Sporadic: the problem comes and goes with no predictable frequency. If the deviation is periodic or sporadic, note the start and end date, time of recurring deviations, and frequency of occurrence.	• When since then has the problem occurred. List all occasions exactly, using date and time? • What subsequent times has the problem been seen or reported/documented? • How often is the problem reflected in the logs or traces? • How often do users report the problem? • What pattern exists, if it can be identified at this point? • What evidence do you have of this pattern?
	IS NOT	***When since that time could the deviation have been observed, but was not?*** When else, since the first occurrence, might you have expected the problem to be in existence, but in fact was not? What other patterns in timing could you have expected to see, but did not? Could it have been continuous, periodic, or sporadic?	• When since could the problem have been seen, but was not? • What times has the system/application worked properly since the first error? • What timing patterns could have been expected, but were not seen?

 Copyright © 2005-2014 Kepner-Tregoe, Inc. All Rights Reserved.

Problem Analysis
Guidelines for Software Specification Questions

Kepner Tregoe

		KT Process Questions and Intent	Software-Specific Questions
WHEN (cont.)	**IS**	***When, in the object's life cycle or history, was the deviation observed first?*** *To understand the specific functional or operational considerations under which the deviation occurs.* At what point(s) in the purchase, install, decommissioning, boot, run, shutdown, etc., of the object does the deviation first appear? Be as specific as you can here—focus on the exact point(s) in the life cycle: What is being done to the object or what is the object doing when the problem appears? The intent of the life cycle question is dependent upon the nature of the object in question—what are the various stages/steps in the life cycle of the object? Observe the object and study its life cycle carefully when describing this aspect of the problem. There can be more than one answer to this question—depending upon the nature of the object. IMPORTANT: Users who are asked, "What were YOU doing?" may feel threatened (or guilty if they have done something stupid) and reply, "Nothing," so the question always needs to be impersonal: "What was happening…?"	• What is the system or application doing or having done to it when the problem is observed? • What is the exact sequence of activities (boot up sequence, keystrokes, etc.) preceding the problem? At what point was the problem noticed? • Have you seen the same application performing the same task before an upgrade or patch was applied? • At what point in the internal communication between processes or applications is the failure seen (as recorded through traces and log files)? • When in the software's history or life cycle was the problem observed first (consider protocol life cycle, operational life cycle (e.g., booting, loading OS, loading configuration, initialization, starting applications)?
	IS NOT	***When else, in the object's history or life cycle could the deviation have been observed first, but was not?*** At what other point(s) in the production, use, testing, shipping, etc., of the object might you expect the problem to first appear, but in fact does not? Again, the more you study the life cycle of the object, the better you will be able to describe this part of the problem. Be sure and try to describe an IS NOT for each IS you have identified in the life cycle.	• When else in the object's life cycle could the error have been seen, but was not (e.g., warm boot versus cold boot, read only versus read/write)? • At what point in your sequence of activities are you sure that the problem was not happening? • At what point in the internal processes is the application/function/method/process working where you would think it should also fail? • When in the software's history or life cycle could the problem have been observed, but was not?

 Copyright © 2005-2014 Kepner-Tregoe, Inc. All Rights Reserved.

Problem Analysis
Guidelines for Software Specification Questions

KT Kepner Tregoe

		KT Process Questions and Intent	Software-Specific Questions
EXTENT	IS	***How many objects have the deviation?*** *To determine the magnitude of the problem.* What is the exact count of the number of object(s) affected with the defective condition? The answer to this question depends on the nature of the object—it could be an absolute count, percentage of functions, etc. Where the object is an application, this could mean the number of instances of the application. Where the object is a process (e.g., a query) it could be the number of queries returned with out-of-spec. responses.	• How many objects have the problem? • How much of the software is affected? • How many systems/processes/applications/ connections have the problem? • How many instances of this application or process have the fault or error? • How many sessions are getting dropped/freezing?
	IS NOT	***How many objects could have the deviation, but do not?*** What other counts, percentages, etc., of the object might you expect to have been affected with the deviation, but in this case were not? If 100% of the objects are affected, is it possible that only some of them could have been?	• How many objects could have the problem, but are not impacted? (Note whether factual or estimated.) • How much of the software could be affected, but is not? • What is the total number or percentage of systems/ processes/applications/connections that could have the problem, but do not? • How many sessions could be getting dropped or freezing, but are not?
	IS	***What is the size of a single deviation?*** What is the most appropriate measure of the level or degree of severity of the defective condition on/with any one object? The answer to this question depends on the nature of the object and deviation in question. It could be a physical size, degree of severity, percentage of surface area affected, percent/amount of time/$ off 'should,' etc. Study the defective condition carefully to answer this question. In the software environment, this question is meaningless for a crash, or any other failure which is 'total.'	• What is the size of the problem (severity)? • How many problems do you see with the system/ process/application? • How much above normal is the CPU load? • How long was the system/process/application frozen or 'hung'?
	IS NOT	***What other size could the deviation be, but is not?*** What other level(s) or degree(s) of severity might you expect this problem to exhibit, but in this case does not? Again, study the object and deviation carefully and look for other appropriate measures of the defective condition that might be happening under similar conditions, but in this case are not.	• What other size of defect could be expected, but is not seen? • How many problems could you see within the system/process/application, but do not? • How long could the system/process/application have been frozen or 'hung,' but was not?
	IS	***How many deviations are on each object?*** Does the deviation appear in multiple locations on/with the object? If so, provide a specific count of the number of deviations on any one object. If the count is a range, provide the extremes and as specific an accounting as possible of the number or percentage of objects at each level of the range.	• How many entries are in the error log?
	IS NOT	***How many deviations could there be on each object, but are not?*** What other counts of the deviation might you expect to see on/ with any one object, but do not? If the answer to the IS question is a range, what other range(s) might you expect to see, but do not? Is it possible that it could always be the same number of deviations per object? What is the most likely number of deviations you would expect to see under similar conditions, but in this case do not?	• How many entries could there possibly be in the error log, but are not seen?

 Copyright © 2005-2014 Kepner-Tregoe, Inc. All Rights Reserved.

		KT Process Questions and Intent	Software-Specific Questions
EXTENT (cont.)	**IS**	***What is the trend?*** Over the entire existence of the problem, what has been the growth or shrinkage of the problem's symptoms, in terms of: • The number of affected objects? • The severity of the defective condition? • The number of deviations on any one object? Be as specific as you can with the trend. Request any appropriate graphs or charts that might describe the trend of the problem.	• What is the change, if any, over time in the number of applications/users/processes affected? • What is the change over time of the degree or size of the failure, where there is a measured extent? • What is the trend in terms of the number of errors on this application? • How does increased up time affect the occurrence of the problem? Does it get better, get worse, or stay the same? • How do fluctuations in usage affect the problem? Does the problem happen more often, less often, or stay the same as usage declines/as fewer users are connected/as usage increases?
	IS NOT	***What could be the trend, but is not?*** Given what you know about the trend of the problem over time, what other trends might you have expected to see, but in fact did not? Try to provide an IS NOT for each IS in your description of the trend. In general, the IS NOT is any other possible trend than the one actually seen.	• What other types of trends could possibly occur, but are not seen?

 Copyright © 2005-2014 Kepner-Tregoe, Inc. All Rights Reserved.