

Module 9 - T201 - How to Write a Test Plan Webinar Transcript

Shelley Row

ITS standards can make your life easier. Your procurements will go more smoothly and you will encourage competition, but only if you know how to write them into your specifications and test for them. This module is one in a series that covers practical applications for acquiring and testing standards based ITS Systems. I'm Shelly Row, the director of the ITS joint programs office for USDOT, and I want to welcome you to our newly redesigned ITS Standards Training Program of which this module is a part. We are pleased to be working with our partner, the Institute Of Transportation Engineers, to deliver this new approach to training that combines web based modules with instructor interaction, to bring the latest in ITS learning to busy professionals like yourself. This combined approach allows interested professionals to schedule training at your convenience without the need to travel. After you complete this training, we hope that you will tell colleagues and customers about the latest ITS standards, and encourage them to take advantage of the archived version of the webinars. ITS standards training is one of the first offerings of our updated professional capacity training program. Through the PCB program, we prepare professionals to adopt proven and emerging ITS technologies that will make surface transportation safer, smarter and greener, which improves livability for us all. You can find information on additional modules and training programs on our web site, www.pcb.its.dot.gov. Please help us make even more improvements to our training modules through the evaluation process. We look forward to hearing your comments. Thank you again for participating, and we hope you find this module to be helpful.

Nicola Tavares

Today's module is T-201, How to Write a Test Plan. The target audience for this module are engineering staff, operational staff, maintenance staff, and testing staff. Your instructor is Ken Vaughn, president of Trevalon Corporation. He's a leader in the development of the NTCIP standards and in 2000, Ken founded the travel on corporation.

Ken Vaughn

Hi, thank you for coming today. As Nicola mentioned, I'm Ken Vaughn from Trevalon Corporation, and I'm happy you showed up. I'm going to talk a little bit about testing today, and how to write a test plan. I got started in NTCIP in 1994, and since then we've been able to develop quite a few test plans and get training on that. So will extend that knowledge to you today.

So this first screen describes where we are in the overall training path. The first course that we recommend you take, is, T-101, Introduction to ITS Standards Testing. That's a prerequisite for this course, T-201, How to Write a Test Plan, and the recommended course to follow this is, T-202, An Overview of Test to Design Specifications, Test Cases, and Test Procedures. From there, you could go on to specific courses that we have for specific ITS device standards, such as, T-311, for message signs, and, T-313, for the ESS standard. In addition to the T-101 module that we just talked about we also recommend that you have a knowledge of intelligent transportation systems, the systems engineering process, as well as basic understanding of the acquisition

process for ITS based standards. Many of these topics are covered under another suite of ITS--ITE training modules, sponsored by the Federal Highway Administration, that are also available through this program.

The learning objectives for this particular module include discussing the role of a test plan within the testing lifecycle of the SCP. Summarizing the characteristics of a good test plan, presenting the outline of the test plan, and describing the relationship among different test plans and test design specifications. And that brings us to our first poll of the day, and to respond to this poll, I will bring it up here in a second, the question is going to be, when should you test? So let me go ahead and bring that poll up. So you should be able to respond on your screen. When should you test? When there is a prototype, prior to delivery, in the manufacturers shop, upon installation at a site, all of the above, or, it depends? So we're starting to get some votes in. Wait a few more seconds here, and I will go ahead and close this poll.

As you see, you should see the results now, it says 71 percent, depends on the system, versus 29 percent, all of the above. Well, the real answer is, as we suggest, is, it depends on the system being acquired. The reason it's not all of the above, can be demonstrated by the fact that most people, when they go out and buy, say, a laptop computer, you don't actually go test a manufacturers factory. You-- it's a proven products, so you wait till you acquire it, and then you make sure it works. So the level of testing you do is heavily dependent on the type of system, and how mature that system is. So it kind of depends on the system being acquired, when you test and how you test. We'll have some examples of that later in today's course.

Now this diagram is probably very familiar to many of you. It is the V diagram from the systems engineering process. It describes the process by which agencies should be developing and deploying their systems. They go through a concept of operations to find the requirements, develop a high level design, they hire someone, typically, to build the system, and then you go into the testing process, of unit testing, subsystem verification, system verification, and system validation.

So today's course is going to focus on the right path of the Systems Engineering process. It's going to focus on the testing aspect, as opposed to the definition aspect. For information on the definition aspect, there is, once again, another set of ITE course is that describe that path of the V diagram. So we talked a little bit about when to test. It depends on the system being acquired, and you test as needed, which may include, when there's a prototype available, the very first, before there's any real production, once the design is complete, any manufacturer prior to delivery, upon delivery, upon installation at the site, and potentially, even, after all components are integrated together.

Now with many ITS systems, you'll be testing it multiple of these steps, because none of them are all of that mature. There's always development on almost every significant deployment. There are two major types of testing. The first we'll talk about is called verification. This ensures that the system is built right, meaning that it's built according to specifications. Whenever you put in the requirements, those are the tests we're going to test against. So you do this through inspection of the system, demonstration of the software, perhaps, analysis of looking at results

and averaging, making sure the averages look right, and using statistics or testing, which is developing clearly defined test procedures and performing those tests.

That brings us to our second poll. And this question is, who should perform verification testing on behalf of the client? So I will activate this poll. So you should see that poll now, and you can go ahead and submit your votes. The options are, development staff, engineering staff, operational staff, testing staff, or other, and you can use your chat pod to explain that. Once again, this is verification testing so who verifies that the requirements are being met? And I believe you can make multiple selections here. Give you a few more seconds to respond. Verification testing. I'll go ahead and close up the pole. And share the results, and you'll see that most people said testing staff. We had a fair number say operational staff, and a few say engineering and development staff with some others thrown in as well. So, let me hide those results, and we'll go back to the presentation. We'll find out one of the answers is development staff.

The development staff, before they ever deliver the product, should be testing the system before the client ever sees it. That way the client doesn't have to test as much. Whenever you go through testing with development, you will find problems, so it's really the job of the developer to go through that testing effort first, try to find as many of their problems as possible, before turning it over to the agency. The other group that's primarily responsible for verification testing should be your testing staff. This is because the types of tests you're dealing with are very detailed, requirement by requirement throughout the standard. The purpose is not for an operator to say whether or not this is what they want out of the system. The purpose of verification is to make sure that what was delivered meets the written requirements and specifications. And that very low level of testing is best done by people who really focus in on that type of test, and we'll read the requirements and test to the requirements. It's a very detail oriented process that many system operators find tedious. So those are the two answers that we recommend the most. Of course engineering and operational staff can't do verification testing as well, but we primarily focus on development and testing staff.

That brings us to the second type of testing, called validation. Validation ensures you have the right system that's been built. In other words, making sure that the system that's been built actually meets the user needs and the intent for that system. That brings us to our next poll. Who should perform the validation testing? So before, we were talking about verification testing, now we're talking about validation testing on behalf of the client. So I will launch that poll. Once again, your potential answers are the same, but that does not necessarily mean that the correct answer is the same. So go ahead and submit what you think is the right answer, and once again, we're talking about the validation. Who best determines whether or not the system will meet the end user needs? Okay, most of you have voted, so we'll go ahead and close that poll, and share the results. And you'll see that most of the people said operational staff. A few said testing, engineering or development staff. The actual answer is-- most of you are right, it's operational staff. Those are the individuals who best understand what the system ultimately needs to do, how it will be used in a real world environment, and that is the purpose of validation testing. So whereas verification is best done by development and testing staff, it's the operational staff that comes in at the end and says, yes, this system will actually do what I need it to do.

So, how do we go about that testing? If we have all of these different types of testing, then how do we manage that? Well, the V diagram specifies several different testing steps. Each project must define when a requirement is going to be tested, where the requirement is going to be tested, how each requirement is going to be tested, and who tests each requirement? That's part of your strategy of saying, we're going to test these items way early because they're very innovative on this particular project, they're new, they're likely to be bug prone, so we're going to test them early, versus other requirements may be well proven, we'll wait to test those, recognizing every project has a limited set of resources, staff, money and time. Where are those resources best invested to give you the biggest bang for the buck on your project? That's part of your testing strategy, how do you go about testing in a planned manner? All the requirements need to be tested. In essence, why have a requirement for your system if you never test it? If you never test it, there's a potential it's not been met, which means, why did you bother having the requirement in the first place? So when you develop your requirements, you need to be thinking about this. How are these requirements going to be tested? You make sure that way, that, when you write your requirements, they are measurable, they're testable, and so at the end of the project you can actually go and test and verify them.

So that brings us to our first case study. If you have a brand new standard, then that's one particular scenario of you may have to consider yourself with, or a new feature. When the DMS standard first came out, Virginia DOT, went out and bought signs. Before they went out and bought the signs, though, they wanted to make sure that the vendors who were delivering those signs, would be able to meet the specifications. But they also recognized that this was a brand new standard, and none of the vendor community had actually implemented fully the standard yet. As a result, what they did is, they came up with a pre-qualification process, and that process included both past history on hardware, review of the sign equipment itself, past performance reviews, in addition to making sure they understood the NTCIP requirements. And the way they checked the NTCIP requirements was, they required manufacturers to deliver a assigned to their facilities to be tested. Now in this case, it was a pre-qualification. These were just signs that were going to be approved so that they could bid on a future project. They did not have to deliver a fully conformant sign on day one at the pre-qualification. So, for the pre-qualification, their process only required the manufacturers to pass 85 percent of the pre-qualification tests.

Now once a pre-qualification, once you're on that list, bid on a project was selected and delivered a sign, as part of that project, you were then required to go through full testing again, and pass 100 percent of the tests, but that pre-qualification process allowed vendors additional time, rather than forcing them to hit 100 percent to even be pre-qualified, which would have limited vendor selection too much. And then finally, they did site acceptance testing for each sign. So they made sure that each sign, when they installed it, would integrate with their essential system. Now you compare that scenario with one of the stable standard, say, where the DMS standard is today. And most systems today, deploying signs, do factory acceptance for hardware requirements, but don't do any NT-SECP requirements at the factory, because most of the vendors have already proven that they know the basics of NTCIP. Instead, what they do, is they wait until the delivery of that first sign. That initial sign, goes through full NTCIP performance testing, making sure that what's delivered, all of the changes and bug fixes and everything else they've done back in their factory have not broken their NTCIP System, but there still fully conformant and compliant with the specifications. And then, once that first sign is approved, they then just integrate every

new sign on the project, making sure that it integrates with their essential system. And that's a typical scenario today. With the management system, though, management systems have to deal with a wide variety of devices and many of these systems have to be customized for the location. The more customization you have, the more likely you are to have bugs arise in your software, so it becomes much more important to go through a rigorous testing process. As a result, most management system procurements require the manufacturer to do their own internal testing before delivery to the client. The client reviews those test reports, before they start any testing. The client then goes and tests the system in its own agency test lab, not in a real environment, but in a separate environment with mock devices. And they start load testing the system. Initially, they only do it with one sign, one mock sign, whatever, but then they will create simulated devices and users to load test the system, to make sure it will handle however many signs and detector stations and everything else it needs. And once it's been proven to be able to handle the full load of a real system, then you start deploying it in real field. But you don't want to deploy it in the field until you know it's actually going to work. So in a separate environment there, your production system will start partial deployment, test it out, and then you slowly scale it up to your full deployment. And that is the typical testing scenario for a significant management system.

That brings us to our next poll. How many test plans should be developed for a project? So let me go ahead and activate this poll. And you should now see that poll. Go ahead and submit your answers. How many test plans should be developed for a project? Is it one, two, one for each test phase, multiple for each test phase, or, it depends? I'll give you a few more seconds here. And most of you have voted, so we'll go ahead and close out that poll. And share the results and you'll see that most of you have selected, it depends, which is always a good option when it's an option there. Some of you suggested one for each test phase, and some of you suggested multiple for each test phase. We'll hide that result, and will show the real answer, which is, it depends. It depends, because once again, you have different products and you have different maturities of each one of those products.

So, in most cases, you'll have multiple test plans, it may or may not be one for in each test phase, although in most cases you will have every phase of development, you should probably be doing some testing. And in many cases, you'll find that you'll have multiple test plans to test different aspects, and we'll discuss that here in a second. So, at least one test plan per testing phase, may have distinct test plans for different categories of testing, such as functional testing versus interface or communications testing, versus environmental testing. Those are very different types of tests. So for example, you're very likely to separate, if you're going to take your equipment into an environmental test lab, that's probably a completely separate test plan than testing your communications, or your functional aspects of the system. It could be integrated together, but many times they will be separated out. All test plans are developed after the requirements, and that's very key. Recognizing first, you define what you need in your system. Once those requirements are defined, and the contractor is brought on board and all of those have been refined, to meet full agreement with the contractor, exactly what is going to be delivered, then you can start developing your test plans, you're detailed test plans. Each test plan is developed prior to starting tests. So you don't want to be changing your test plan as you are conducting the tests. The intent is, define what you're going to do, based on what the requirements say it should

do, and then verify that the system actually does that. Otherwise, you're kind of cooking your test plans just to make sure it passes, or something.

That brings us to our next case study, the sample test plan. Now there's a student supplement. You should see-- if you've not retrieved it already, you can go to the-- go to training docking station, and under materials, you'll see the student supplement their. If you go to page six of that supplement, there is, in fact, a sample test plan there, and we're going to start walking through the sample test plan. So please go ahead and make sure you have that downloaded and open on your screen, so you can step through that with us. And once again, we're going to go to page six, is where the test plan starts. And you'll see that the test plan starts with a test plan identifier. The purpose of that identifier is just a shorthand notation to identify a particular test plan. Remember, you're going to have, not only one test plan, you're going to have many test plans on a project, and you'll have test results, you'll have test case specifications, test designed specifications, testing logs, all sorts of documentation, so having a test plan identifier allows as a quick way to identify the particular document you talking about. After that, you have objectives, and the objectives identify the types of requirements that are going to be tested, as well as the testing phase. So in the sample, we talk about, upon delivery of the product, and we're talking about the types of requirements, we're talking about the NTCIP requirements. The introduction also gives a background and references, and in particular, the references provide a reference to the test procedures that are being used in the project.

The next section, on page seven of the supplement, talks about identifying the test items. It goes through each item to be tested, the version of the product, the specific version of the requirements. Now in this case, we just simply reference the transmittal report, to identify the exact version of the software. Very frequently, when you're writing the test plan, you don't know the exact version of the product, until it's actually delivered on site. The features to be tested, is the next section. Identifies requirements that will be tested, and we do this within the sample, by referencing the protocol requirements list. Now if you've taken the acquisition courses, you're familiar with this protocol requirements list. This is how you specify the product you want. It identifies all of the requirements that are in the standard, that are being selected for your given deployment. So you compare that long list of requirements, and you select, based off of those requirements, those are the features that you're going to need to test, because remember, we said every requirement for a project should be tested. So that list is as long as necessary, and identifies your requirements that are not tested, is the follow on segment from that, because anything that is selected, you will be tested. Anything that is not selected as a requirement for the project will not be tested.

That brings us to our next question. Where do you find the requirements list when the standard does not include a CP content. So let me open up that poll. And you'll see the options are, define them in the test plan, refer to project requirements, refer to design specification, or refer to the user guide. So go ahead and make your selection. And the results are slowly coming in. Once again, when you are looking for requirements for your test plan, and the standard does not include this information, where do you find this information? Okay, a few more seconds here. I'll go ahead and close the poll results, and will launch-- or share the results. You see, most people said, refer to the project requirements. That is, in fact, the correct answer. You also need to refer to-- some people said refer to the design specification. Well, really, the design specification is

what's developed after the project requirements. You have project requirements, and that says what I want. The design specification really only is designed to say, this is how I'm achieving those requirements. When you're doing testing, you really don't care about how they achieve the requirement, you're really only concerned that they met the requirement itself. So referring to the project requirement is the right answer. That brings us to our next item, which is the approach. And once again, we'll have a couple of different ways here. The standards with test cases, if they've already been defined within the standard, then it's pretty easy. You refer to requirements- - to the requirements to test case traceability table, within the standard. All of the standards which include the definition of test cases, include this traceability table from requirements to the specific test case. And there's a sample of this within the participant supplement, on pages 20 and 21. It also identifies activities to be performed, identifies the tools that are needed to perform these test cases, and the key, really, is to identify enough detail so that a user or a planner can identify the amount of work that will need to be performed, how much time will this take, how many resources will I need, and how much is this going to cost? That's the goal of developing his test plan.

Now, if the test cases are not included in the standard, then you'll have to develop them yourself. They will have a high level overview of how an item will be tested, and you will have to specify that in your own test case, and then identify the activities to be performed. Identify the tools that are needed, once again, it's going to be more difficult because you're doing the work yourself, but your goal is to identify how much work will be needed? And, with every test plan, there needs to be clear definition of what constitutes passing the test or failing the test. In most cases, this is a 100 percent requirement, although there are some cases, as we discussed earlier, where, maybe 100 percent isn't the right marking tool. Most cases, though, will be 100 percent success, but it's important to call that out, to identify the fact up front, that no exceptions will be allowed. You also need to identify suspending the test. Many of these tests take quite a bit of time, maybe several days, so how do you pause the test at the end of a day, so that you can restart the next day? What steps do you have to take to pause the testing procedures, what steps do you have to take to restart the testing? Are there places where you cannot stop, and are there places that are better to stop? So all of that needs to be considered when you're writing your test plan. Now, you also need to consider all of the deliverables that are going to be provided during this process.

Lots of different documents, as we described before, to manage. First off, you have your requirements that are part of your specification package, you then have your test design specification, and in addition to this test plan. The test designed specification is a very high level overview of each individual test, and then that's more detailed in your test case specification that provides a detailed description of inputs and outputs for the test. That's then supplemented with the test procedure specifications that identify the exact process that is required to get those expected inputs and outputs defined in your test case. Finally, once you have that whole set of test documentation done, you then are ready to receive the equipment to be tested. And along with that equipment to be tested, you should also receive a test item, or prepare a test item transmittal report saying exactly what was received on what date, version number of software, everything, serial number of hardware, whatever. Then you can perform your tests. And as you perform your tests, you produce another set of documentation. That includes your test log, you want to capture as much information as possible, as you're performing this test. Making sure your capturing all of the data that is exchanged over the wire, as well as capturing as much

information off of the screens you're seeing, to make sure that you know what's happening every step of the process.

It's very frequent that people go through testing, they identify some bug, but then they have a real challenge trying to recreate how they got that bug. And the more you're capturing information as you go along, the easier it is to recreate that scenario and identify exactly how to reproduce those bugs. So the test logs are very important. But also, it allows you then to produce your test instant reports. For every single flaw or bug you see in the system, you should be preparing a test instant report that identifies exactly what the problem is, so that if it's properly reported, then the developer can take that and make sure that that particular report is addressed and repaired. And then, finally, once you've done all of that, you compare your test summary, which, as it says, summarizes the results, and that's a final report that most of your managers will look at and identify, okay, did it pass, how big are the problems, how many times do we need to go through this again, and make the management decisions on how to precede.

Now the next section in your test plan identifies the testing tasks. Each task should have a description, predecessors, a responsible party, identification of the skills required and the effort required. Remember, the focus of the test plan, really, is to identify how much effort is required overall, and what resources will be needed, so this is all part of the process to identify how large of an effort, how much do we need to budget for this process? That brings us to our next activity. Tasks involving testing. What are some of the tasks involved in testing? So you can go ahead and use your chat module to go ahead and provide some responses to this. So just think of what are some of the tasks that you have related to testing? Go ahead and type those into the chat module and as I see them come in, I'll let you know about it. So you've identified some. We actually perform the test, would be one. Some of the other tasks would be, preparing documentation and getting some other suggestions in here. Results analysis, finalizing the test plan, and we'll go ahead, go on to the next slide which will identify some of the ideas. Developing the test plan itself, and all of these other documents we've talked about, receiving the required equipment, setting up the test environment, actually performing the tests, recording the test results, and summarizing the test results. And of course, analysis would be as part of that summary and then finalizing the test plan is also on there as well.

So, the other-- the next section of the document talks about your environmental needs. This is very important to make sure that before you go about testing anything you've considered all of the equipment you're going to need to make sure you have all of your major components, and the connections between those components, the right dongles, and anything else you need to connect various pieces together, making sure you have the right test software, and make sure you have that recorded. Configuration of each piece of equipment, how is each piece of equipment supposed to be configured to work together. Finally, practical and logistical needs of the test environment, such as making sure you have the proper electrical outlets making sure you have tables and chairs, you have protection from the elements, you have safety considerations if you're out on the side of the road, that you have your orange safety vests and other things.

The next section of the document talks about roles and responsibilities. You need to identify each major stakeholder involved in the test, and the responsibilities of each of those stakeholders. Then you need to talk about the schedule. Define the expected start and end of each testing task,

identify the dependencies on other project tasks. This is critical, because if you're dependent on other projects, in order to perform your tests, that could potentially be a huge impact on your schedule, so there may be other projects dependencies within the project, dependencies on resources, so you may have a dependency on getting communications installed to your field equipment, it may be making sure that you have permission to use a particular sign in the field at a particular time of day. And you may have dependencies on when you can get staff time to do the work. All of these need to be considered. Because there are so many dependencies on when it can start, the schedules are often shown as weeks from the start of testing, so that regardless of what other project constraints may occur, and delays, you can simply shift your testing schedule.

You should always count on a certain amount of time for your testing. And, in fact, you also need to make sure that you have time and appreciation for the purpose of the test, which is to identify problems so it's not just simply going through your test procedures once and everything checks out. The plan should be, we're going to go through this process once, we will more than likely find problems, the developer will then have to go back and fix those problems, and then we will have to retest again. And depending on how new your system is, you may have to count on multiple cycles to test and retest and retest, before you can expect a final deployment. The schedule defines the length of each relevant step related to the V diagram.

That brings us to our next activity. During testing, what are some of the problems that may arise on a project? Can anyone give concrete examples that may have happened to you or your colleagues? So what problems can arise, and if you can give examples, particularly real world examples from the user community are always very valuable. So go ahead and type those in, using your chat pad, and we'll talk about those. And there's, of course, a variety of projects or problems that can arise during testing. You may not have the equipment available, the people you plan on doing the tests may become incapacitated for some reason or unavailable, drawn off on to other projects. You can also have communication problems hardware problems, all sorts of problems. So all of these things need to be considered as you're developing your project. What are all of the risks that you can have, and if those risks materialize, the plan really needs to identify how will we respond to these situations. So someone suggested software bugs which, of course, are another big problem. So all of these things, whether they're problems with the system your testing, problems with having access to the resources, problems that, you know, if you had a project designed to start testing at 9-11, obviously, and you're in New York, or something, people are going to be pulled away. So what happens when unexpected events occur, you need to consider those in your plan. And it may delay your development, because if you can't test, you don't know what the bugs are, they can't be fixed. And you may also have a problem with delay in other projects. So you can't start testing until the other projects are ready. Resources being unavailable, defects found during testing, all of those are common risks that you have to consider.

So what happens if delays occur? While many times, the delay simply delays the deployment of the project. It delays testing, and everything else slips behind that. But sometimes, you don't have that luxury. You know, if you're installing a brand new system for a special event like the Olympics, then that date is fixed, it's not going to change. So under those situations, it becomes really critical that you really start considering what are our contingency plans if we can't get this deployed without confidence that it's going to work, what happens then? Is there a way to scale

back the project if we'd need to, in order only to install a subset of features rather than the whole thing. And that just has to be decided on a case by case basis.

That brings us to the final section which is approvals. The approvals simply is, each of the major stakeholders approving the test plan, and saying yes, this is how we should go about testing, and if this product passes this test, then it should be accepted. And all of your major stakeholders should be involved in that. Obviously your agency is going to be a major stakeholder the developer is going to be a major stakeholder, and typically you also have a separate tester who also wants to be a part of this process. Well, that covers what we're intended to learn in this course. We'll take a quick summary of what we've discussed. Testing occurs throughout the right side of the V diagram. We also talked about testing should follow an overall strategy. Test plans should follow the IEEE-829 outline, so that's the outline that we discussed here today. The test plan is one of several documents, or testing documents that are produced. The detailed steps are defined using test design specifications, test case specifications, and test procedure specifications. The results are then reported in the test summary, and instant reports and the test log.

That concludes this portion of the course. But the follow on presentation is T-202, Overview of Test Design Specifications, Test Cases and Test Procedures. From there you can go on to individual courses for specific standards, such as the one for message signs, or the ones for weather stations. If you want to learn more, we have a module supplement, as well as the other ITS module learning courses. And with that, we can have questions from the audience. And you can type your questions into the chat window, and I'll happily answer any questions you might have.

Thank you for participating today, and, but please stay on the line and I'm sure the questions that come up will be very valuable to listen to. So any question you have, go ahead and type it into the chat window. One of the first questions I've seen come in, is, who is best to write the test plan? Well, generally speaking, you want someone who has some testing experience, they're going to be the most knowledgeable of what to look out for, so that could be a-- someone who is like a developer, although I think it's generally very important to also have an independent test agency. So I would say in addition to the developer, you also need someone else, and that may be a contractor or consultant, it may be someone internal to the department. The key is that they really have experience testing Software Systems, and what we're talking about here, or testing the types of equipment that you're testing on your project. So if you're testing hardware, they have experience in testing hardware. That experience is very valuable.

So if there's any other questions, please go ahead and type them in, and we will address those. Hearing no others-- Oh, here we go. So how is SEP, and non-SEP standards-- or how is testing affected by SEP, and non-SEP based standards with good information available, versus not available. So testing is, with the SEP standard, if it has all of your SEP content, including test procedures, then you can simply use those test procedures and implement them straight. It's a very easy process. The standards that have test procedures defined in them, there are typically off the shelf tools. They have automated those procedures and therefore, it's very easy to implement those tests procedures and relatively inexpensive. The standards that don't have test procedures, but do have the remainder of the SEP content, it's still fairly easy to develop your own test

procedures. Requirements are clearly spelled out, and you go through identify a test case for each requirement, making sure every requirement is being tested. And once again, there's off the shelf tools that facilitate much of this. You'll have to be developing your own test cases, but the tools are there to assist you.

The final aspect is with those standards that do not include any SEP content, at which point, you're relying on your interpretation of what the requirements are to be met, and then write test procedures according to those requirements. It us a little bit more effort, but any project has requirements and every project, every single requirement needs to be tested. So-- are there certified testing labs available for ITS standards? Unfortunately no. Generally speaking, the way the industry handles this is, there are a variety of firms, including my own, that offer testing, third party independent testing, and they will issue letters of certification saying that they performed a certain set of test procedures. That does not necessarily certify that the device is conformant in every respect, because there are an infinite number of possible scenarios that can happen to a device. It says that we have performed a test suite against the device and it passed that test suite, or, alternatively, here are the problems we found. But there's no single certification lab, it is more of an open market solution. Firms that are willing to issue those sorts of letters.

So that's all of the questions that I have received, and with that, we conclude this module of ITS training program, and with that, I pass it back to Nicola.