Ken Leonard: ITS standards can make your life easier. Your procurements will go more smoothly and you'll encourage competition, but only if you know how to write them into your specifications and test them. This module is one in a series that covers practical applications for acquiring and testing standards-based ITS systems.

I am Ken Leonard, the Director of the US Department of Transportation's Intelligent Transportation Systems Joint Program Office. Welcome to our ITS standards training program. We're pleased to be working with our partner, the Institute of Transportation Engineers, to deliver this 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 your colleagues and customers about the latest ITS standards and encourage them to take advantage of these training modules, as well as archived 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. You can find information on additional modules and training programs on our web site at 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, and thank you again for participating, and we hope you find this module helpful.

Nicola Tavares: Throughout the presentation, this activity slide will appear, indicating there is a multiple-choice pop quiz following this slide. You will use your computer mouse to select your answer. There is only one correct answer. Selecting the Submit button will record your answer, and the Clear button will remove your answer if you wish to select another answer. You'll receive instant feedback on your answer choice.

This module is T204 Part 1 of 2: How to Develop Test Procedures for ITS Standards-Based Test Plan.

Your instructor, Dave Miller, is Principal Systems Engineer at Siemens Industry, Inc. He has more than 36 years of experience in the management, design and development of critical controlled systems, including traffic control, bus rapid transit, connected vehicles, and specific ITS experience includes design and first deployment of ATC 5201 LINUX and ATC 5202 Model 2070 ASCs, as well as the design and first deployment of NEMA TS2 ASCs with NTCIP.

Dave Miller: This Module is primarily for test personnel responsible for verifying actuated signal control design and operations to the user needs and requirements of, typically, purchase contracts. Other stakeholders who might benefit from this module include actuated signal controller manufacturers, system developers, and maintenance personnel who would use the resulting tests for routine maintenance, like for a maintenance staff.

This is the sixth module dedicated to testing, so the prerequisites were, beginning on the screen there with T101 going through 201, 202, 203, and it was a two-part module.

That leads us up to where we are now. This is the curriculum path, so if you did the testing sequence, you would've started with T101, which is the basic introduction to standards testing, kind of an overview, and then we went through how to write a Test Plan, the overview of Test Design Specifications and test procedures, and moving on to T203, the two-part module, we went into how to develop test cases, and the second part of it is test cases with a little bit more detail, going down into actual wording and that sort of thing.

So now we're at T204, Part 1 of 2, so we're going to build on T203, and here we're going to take those test cases that we did in the prior module and we're going to move on to developing test procedures. So we're going to do procedures and this, again, is a two-part module because it's kind of long, and we'll have a natural divide in the middle between the two parts, because there's a tool we're going to be using, which we'll talk about.

Starting off with the acronyms, there's a lot of letters we use here that through this module you will see these things. We'll talk about a Test Plan; Test Design Specification; Test Procedure Specification; Test Plan Generator, which is the tool; Test Case Specification; a piece of roadside equipment called a Dynamic Message Sign, you'll see driving along the road; TMDD is Traffic Management Data Dictionary; and we'll talk about Requirements Traceability Matrix; and Requirements to Test Case Matrix. So those are the acronyms, and when we run into them for the first time in here, we'll go over what the definitions are, again.

Again, this is a two-part module, so you see, actually, a total of eight learning objectives. We're going to be doing the first five here, so it's the top part that's in bold, and then at the bottom of the slide, you can see that the grayed out ones are the ones that we'll be doing in Part 2.

We'll start right off with Learning Objective Number 1. What we're going to do here is to recognize the purpose and structure of a Test Procedure Specification. So what we're going to talk about is what the inputs are to a test procedure and then what we would expect the outputs to be and then, any special needs we'll run into along the way.

This is a quick introduction to the two key definitions using this module, and it'll help you, for those of you that have not taken the other testing modules. We assume you have, but if you haven't, a test procedure provides the detailed instructions for setup, execution and evaluation of a given test case, and again, we're going to be talking about Standard IEEE 829. We'll talk about that a little bit more later. We're not going to go into any real detail and teach you what that standard is, but we're going to be using that format and we're going to be using those steps.

Then, a Test Procedure Specification is a document that defines the steps to execute a test. So remember, from the prior module immediately before this one, we did test cases. Now, we're going to do the procedure. We're going to take that test case and go through a step-by-step procedure to get outputs.

So the Test Procedure Specification is used in a project as a part of a Test Plan. So we had an overall Test Plan that we talked about in earlier modules, and we use it to accept the system. When it's all done, it's all installed, how do we know it's working? As all of the stakeholders that are involved, like the end user, owner-operator, or the taxpayers that are going to pay for it, et cetera, how do you know that it's operating correctly?

This slide states the purpose of a test procedure, and these four concepts are helpful to, kind of, understand the big picture. So we're just going to go over these, briefly. We're not going to go into detail, but a standardized test procedure provides a meaningful set of test results. You get outputs from the test procedure that you can use as real data that says, "Yes or no, did it work correctly?" for evaluating the system, and what we're answering is, "Did we build the system as intended by the project specification?" The stakeholder who wrote the project specification said, "I'm specifying this is what I want it to do in the end." We're talking here about how to verify that it does what it's supposed to do.

In Module 203, we learned how to develop test cases. I think we used a Dynamic Message Sign as an example. And then, going on, in this module we'll learn how to develop the test procedures using the IEEE 829 Standard.

Again, we're not going to really go into depth on the IEEE 829, but this is just an introduction. What it does is, it provides a standard format. So if I'm one of the stakeholders, say, I'm the owner-operator or I'm the person that wrote it or I'm the traffic engineer, whoever, this doesn't tell you how to do the test. It provides you a standard format and definition so that there's no question when I say, "Pass, Fail, Anomaly Report," we all have this common definition of what that means so there's no confusion. It includes-- 829 will have standards for covering Test Plan, the specification, the test cases, the procedures, and the outputs for the test reports. The reason we're using this is the ITS industry, as a whole, is moving towards developing test documentation that conforms to this standard, and it comes from a software and system world. So if you're doing a large software project for, like an Aerospace System, or whatever, that has lots and lots of software pieces to it, IEEE 829 is used extensively, so everybody's on the same playing field and using the same definition of terms and the same steps, and again, we're not going to go into detail. All we're telling you here is we're using this format. So if you hear me use a definition or the steps that we go through or something like I talked about an anomaly report, everybody that's using this standard knows what that means.

What we're doing here is, again, a very quick outline of what Standard 829 shows. If you go into that, it'll show for every project when you're going through a test procedure, the test procedure will have an introduction that has a document identifier, a scope, references to any outside standards that you're using, and any relationship to other procedures like, "This one has to be done before--" a different one had to be done first, that sort of a thing. Number 2, the Details. It's going to describe the inputs, outputs, and any special requirements, and then it's going to describe, in order, the steps you need to take to execute the test cases, and finally, it has a glossary, if there's any terms that aren't defined, and, of course, at the very end, there is a document change procedures and history. So every time someone's made a change to the document, it shows who did it and when. So if you're using 829, and we're using it here and we highly recommend that you use it because the entire ITS industry is going in that direction, this is what you'll be doing. Every test case will have a procedure and it'll have this format, so everybody's clear on what we're doing, up front.

This slide has a few additional elements that's not really stated in the outline before but this is typically discussed in test procedures. This simple layout explains that the Test Procedure Specification assigns values to inputs and then creates expected outputs, based on those inputs. So all of the issues, such as resource, schedule, can be pointed out quickly, and the specification also documents any special needs. The Test Procedure

Specification in the format we're talking about always has this stuff, and every stakeholder, these are details that if you have some sort of a question about a test, there would be a flurry of emails or there would be lots of phone calls. This avoids all of that because we're all using the same documents, we're all using the same format, we're all using the same definition of terms, we're all using testing things in the same sequence. So everybody knows how we're doing things, up front.

This shows where test specification test case and shows a document structure. Again, we're not going to go into this in detail. You can see some more of this type of stuff in the supplement, but this is where it fits in. If you start at the top, there's the top level above the first dotted line at the top is Test Plan. We had a prior module that described that and also, the prior training modules before this one, between the two dotted lines, we talked about how to do a Test Design Specification. In the middle is Test Case Specification, and now we're down where the arrow is pointing, the test cases that we did in the prior module. Now we're going to do Test Procedures. The Test Procedure is conducted on test inputs to produce test outputs.

So we have inputs that are test cases, we did in a prior module, training module. Now, we're doing Test Procedures that include the expected outputs, and then, after we conduct the Test Procedures shown in the box, the outputs are the test results that are documented, so that's the flow. Test cases, we do procedures, the procedures include what we expect to go and what we're expecting the test to result in. We get the outputs and we compare, at the end, the test results with the logs. So the test case inputs are used as a stimulus to the procedure. The test procedure operates on inputs to produce outputs in the form, and then, we'll get a summary on the right here that says, "Discrepancy Report." Now, that's really, if we go into 829, the older versions of it call it a Discrepancy Report, but from this point forward, we're going to be calling that an Anomaly Report. So each of the deliverables are documented in the IEEE 829 format using those common definitions, and the outputs will be typically divided into three categories. So we've had our test case, we ran our test, we have outputs, we compared the outputs that we've got in the test to the expected outputs we had in the procedure. So in the two categories, if the outputs that we got when we ran the test match the expected outputs, we would call that a pass. Test passed as designed and runs as expected. Don't need any further action. You just document that you did it and it passed. Unexpected outputs, we got our outputs, the real ones from the test, we looked at the expected ones and they're different. If there's a gap between those, that does not mean that the test failed. That's a common misunderstanding. "It was supposed to do this but it did this, therefore, it's failed. I'm out of here." No. Unexpected outputs isn't a fail. It just means it's not what was expected. Unexpected outputs are documented in an Anomaly Report for further investigation. So it's not really a fail. You make an Anomaly Report that shows what the differences are and it goes to further investigation. For example, if the investigation findings might reveal a flaw in the design, that really means that the test failed. That really is a failure. Or it might reveal a flaw in the Test Procedure. Maybe the design worked but the Test Procedure had a bug in it. Or it might even reveal an error by the test operator. The design was correct. The procedure was correct. The operator made a mistake. So there's other findings from the investigation, but here is where we begin to realize the true value of the common format and understanding it in terms provided by IEEE 829. Because the investigation is normally conducted simultaneously by several divergent stakeholders, using this Anomaly Report, and a lot of us have been there. We run the test. The outputs aren't expected. It gets reported as a failure. Lots of people in the organization, or outside the organization, send a bunch of emails or phone calls, and you explain the same thing over and over again. It's very important to know that if the test wasn't what is expected, it's not a fail. You write this Anomaly Report. Everybody can read it. That's one of the very, very powerful advantages of just following 829 standard. So that's the categories. We said the expected outputs and summary are a pass. The unexpected outputs require further investigation.

The Role of a Test Procedure, the test procedures ensure that the deliverable system is built correctly and will function as expected by the end user. The test procedure ultimately traces back to the user needs through the test cases and requirements. For example, to verify that we built the system correctly, we need to make sure that the design implements the standard objects and the dialogs correctly and fulfills special contract provisions, if there was some special wording in the contract that's outside the standard. Make sure that that is implemented correctly within the framework of the standards, and again, we use the IEEE 829 formats and definitions so everybody's on the same page that reads it and there's no question on what the words mean and what the steps were and why we did it like that.

The second part to the role of a test procedure is, if you recall, in the earlier training modules, we used the "V" Diagram, so we're going to the right part of the "V" here to remind us where we are in this life cycle. The test procedures ensure the deliverable system is built correctly and will function as expected, and again, the test procedure ultimately traces back to the user needs through the test cases. For example, to verify that they've built the system correctly, we need to make sure that the design implements the standard objects correctly and fulfills the contract provisions. So, again, this is, if you go back and look at the "V" model, we're going up the right side of the "V," and you can see the little dotted lines coming in from the left. The system verification would trace back to the left side of the "V," where we were defining what the system would do.

From special needs, this slide points out special needs that may come up during a test procedure, and a lot of these come from outside the standards. So from the standards point of view, we must take into account special needs that are outside the standards. An example would be, say, the standards do not provide a framework for security measures, such as communication encryption or requirements for special access rights to databases. You would expect a system, when it's delivered, to have security for communications and some way to do passwords. Well, there's nothing in the standards to do that, but the user will expect that, for sure. So the user would probably be frustrated by multiple versions of identical documents without revision control, and would expect to be able to configure and operate the test procedure at a later date. So, again, even though we're operating outside the NTCIP standards, if we use 829 and, say, we want to do something that's outside the standards, I'm sure the contracts would say it has to have passwords and has to have security for the communication. We would still use the 829. None of that is defined by the standards, but would certainly be expected so those types of special needs would be documented using the 829 standards and tested and verified.

And then, there's also the concept of special contract provisions. There may or may not be, but there might be some special needs that are in the contract. We don't want those to show up at the end when we're trying to test things. So there might be something to say, "Well, I'm doing adaptive control system. There's nothing in the standards on how to do adaptive control system. The designer implemented adaptive control system. Now we're at the very end. Now I want to write a test procedure to make sure it works." You do not want to do that. We want to have those all done in test procedures up front so that that can be used by the developer all the way through.

Again, we're not going to go in detail on 829, but if I was doing a test procedure, this is the standard way it would be done, and if you're in the software world and use this standard, this is what would happen. These are the steps and they're called Level Test. So you may have heard things like, well, System Integration Test or Module Test or Hardware Test or Software Test. There's all these things but 829 bypasses all of that. It's just called a Level. So you might be testing a software module or piece of hardware at the hardware test, but they would call that, say, Level 1, Level ID Number 1. Then, we would integrate those together. That might be known, in the world, you may have heard a System Integration Test. That would be a Level 2 test, for example. They just call it a Level and it'd be a different identifier. Then, you'd put all that together into the system, and you might call that an Acceptance Test. Well, that would be another Level Test.

Dave Miller: So what you see here is any sort of a test, just called a level. It has a level ID, so we don't get into a discussion on what to call each test. So it'd be like Level 1, Level 2, Level 3, as we're going up the right side of the "V" module, and it would have a log. At the top, we'd have a Level Test Procedure ID Number 1, for example. The Log would always be included. It would say the tools and methods that would be needed. Those would be listed. Next, Setup; "What is the sequence of actions to prepare for testing?" "Put the unit on the bench, plug it in, key in whatever," whatever we have to do to set it up for the test. There's always, in 829, a Setup listing, and there's how to start the test. This is how you begin the test, Step 1, Step 2, Step 3, Step 4. Measurement, how we collect the data. You go through the test sequence described. Step 1, do this, collect this data, write it down, Step 2, etc. That list. Once we've done the test, we do Shut Down; so we've finished the test. This is what you do to shut the system down. Turn it off, whatever. Then, that usually occurs when you're running a test and you come into some unexpected results. There's always a question. So you might want to suspend the test. "Oh, I got some unexpected results. I've written my Anomaly Report. I need to investigate tis further." So this is a description of how to suspend a test, and then after you've done an Anomaly Report and you've figured out what's wrong or you've resolved the issues, you can restart the test. There's an agreed procedure up front on what constitutes a Shut Down and what constitutes a Restart. And then, your completely finished procedure to orderly halt a test, this is everything, all the data, I have to have before I can finish this test procedure. I want to make sure it's all there, and then, Wrap Up is, this is the next step that needs to be done afterwards, and any Contingencies at the end. If there are any additional anomalies, I mean, you will hear things like Open Issues List, but here we're just going to call it Contingency. So you can see that all the levels, all the different testing that you do in a system is called a Level and these are the steps we always go through. So again, the power of 829 is that everybody does it the same way.

We're going to do an activity here. The question we're going to ask for your polling is which one of these four statements is false? Again, we're looking for the false statement, so the choices are: "A," "ITS standards define objects and dialogs to be tested," is that true or false? "B," "ITS standards provide a format and steps for the Test Procedures," option "C" is "Test Procedures can merge with Test Cases," and "D" is "Test Cases trace to Requirements." Again, we're looking for the false statement here.

We'll go ahead-- and, again, we're looking for the false statement, so the correct answer is "B," standards provide the format and steps. ITS standards don't provide the format and steps for any of the test procedures. IEEE 829 does that, and again, standard objects and dialogs, that would be incorrect. Some of the latest and all of the NTCIP standards define the objects, and even some of the later standards that were developed, have the dialogs for the test. "Test procedures can merge--" That is incorrect; 829 links test cases as inputs to a test procedure, and then, finally, "D," "Test cases trace to requirements," so that one was not false.

We've made it through Learning Objective Number 1. So this was, again, Recognize the Purpose and Structure of a Test Procedure. We discussed the purpose and structure and we showed that it's based on IEEE 829 format and definitions and steps. Secondly, we reviewed the inputs and expected outputs in a test procedure, and then, finally, we reviewed the special needs that arise from outside of the ITS standards and contractual requirements.

We're going to move on to Objective Number 2, and that is to Identify the Role of a Test Procedure Specification within a Test Plan and the Overall Process. What we're going to do is review Test Design Specification, then the role of Test Case Specification, and then test procedure steps.

Recall that when we took Module T202, we learned that the design specification is a document that specifies the details of the test approach and identifies any associated tests, anything that's related, and the design specification also specifies the approach in detail and identifies the associated tests. So the test may apply to a single requirement or a single test may apply to a combination of requirements.

This is an example from NTCIP 1203 Protocol Requirements List, which outlines the requirements we're going to test, and this one we took from Module T202, so you should be familiar with this. If you recall from there, this Protocol Requirements List that you see on the slide is used to link the user needs to the requirements, and again, we're trying to go through this whole process, so at the very end we test it, we know all the way back, it traces all the way back to the original needs of the stakeholders that are going to be using it. Later on in this course, we'll need to test each of these requirements to make sure that each user need was satisfied. But we're bringing this up because this is the link. So far, we've linked back the user needs to requirements. We're going to pick up here and then do the verification.

Again, if we recall from T202, we learned that the Test Case Specification is a document that specifies the inputs, the predicted and expected results, and it also has a set of execution conditions for each identified test item. Again, you should go back to Module T203 Part 2, for some of these examples. We're not going to bring them up again here, but remember that IEEE 829 describes three sections. The Introduction of Section 1 includes a document identifier, references to external documents, the description of the context that's not included in the definition notes, such as context might be numbering systems, things like that, that are standard but probably aren't documented elsewhere.

That was Section 1. Moving on to Section 2 of the Test Case Specification, the specification is a little bit more than just a list of initial condition tests and expected results. Section 2 includes details, so we would include a unique identifier for each case, a description of the objectives, why are we doing this test? The environmental needs for setup, the execution and recording of the results, such as hardware or software that we need to execute the test. Section 2 also includes special procedures, any interdependencies to other test cases, such as, say, the order the tests have to be conducted, "Level Test Number 1 always precedes Level Test Number 2. Don't do 2 before 1 because they're related," that sort of a thing. Section 3 includes just general information such as a glossary for anything that wasn't defined in the 829 standard, and again, just the change procedure, so you know when the document has been changed, to make sure you have an up to date document.

Test Procedure Specification is a document that specifies a sequence of actions for the execution of a test, so we're up to what the sequence is.

Here we're showing an example of Requirements to Test Case Matrix, so this would be one based on NTCIP 1204. You see down at the bottom part, that was a table, and you can see that there was a requirement that came from user need, so every requirement has an identification, leftmost column. The requirement has a title, and then, we're moving on to, to verify that, the test case, so one of the tests would be 3.5.1.1.1 Retrieve ESS Characteristics, so the Test Case to verify that is C2.3.1.1, ESS Characteristics, so that would be a test case to verify that requirement. Again, it all seems kind of like a lot of work here to have all these different identifications in this matrix, but this provides traceability. If anybody at the end says, "I had this need and it's not working," you can go back and say, "Oh, yes, it is. We traced it all the way through."

The Test Workflow is shown here. The initial Test Plan is followed by Test Design Specification that details the approach, and then the workflow or testing sequence is broken down into individual test cases that show the inputs, the expected results, and the execution conditions. The Test Procedures document is the step-by-step procedure, so an individual test procedure may cover more than one test case. So this is kind of where we had gone over this before. We started with the plan, the specification, the cases, and then the test procedures, down at the bottom.

We're going to do another activity here. We're going to do some polls. In addition to inputs, outputs and execution conditions, Test Case Specification includes...so which ones of these does a TCS also include, in addition to inputs, outputs and conditions? Answer Choice "A" is "Test objectives to provide guidance to the test operator." "B" would be "Test environment of the hardware and software" needed to run the test. "C," any "Special procedures, such as automated test tools," or option "D" is "All of the above," which means, "A, B and C are true." Go ahead and take the poll.

We're going to go ahead and review the answers. The answer is "D," all of the above are true. A test objective is used to provide guidance to the test operator, the test environment of hardware and software is included in the document, and also included are the special procedures, such as automated tools, and anything else that would be special outside the standards.

Summary of Learning Objective 2. This learning objective has been a brief review of the workflow that was taught originally in T202, as you recall. Plus, we added how that workflow carries forward into developing test procedures that we're going to use later in this module. Then, we learned that the outputs of T202 become inputs to test procedures that we'll cover in the next few slides, going forward here. We also reinforced, in this learning objective, that the resulting documents will conform to IEEE 829 as a framework. Again, the full content of 829 is outside the scope. We're not teaching 829 here. Rather, we're just learning how to use it and tailor what's available in 829 for our workflow and testing steps that we're going to be using in the remainder of this module.

Moving on to Learning Objective Number 3, in this objective, we're going to synchronize the Test Procedure Specification to the contract terms and conditions, so that we can have a successful contract execution and close the contract successfully, and we're going to talk about ITS project background and discuss how to structure contract terms and conditions from the viewpoint of project end.

This is a graphic that came from one of the reference materials on the USDOT website. If you want to go into their references, and you can find this full description, I would very much encourage you to do that. But the message we're saying here is you need to pay attention to how the project unfolds. If you look at this reference article or this reference document, it'll show you that in the surface transportation rules, roads projects are typically awarded to contractors submitting the lowest bids, and they're almost always successful. But you can see that using the same approach in ITS projects usually fails, and this was some hard information, a hard study on 280,000 information Technology, IT, projects that were surveyed, and of the 280,000, 142,000, a little bit over half were late or over budget and 42,000 were failed and were cancelled. So the failure factors usually indicated the lack of user needs, the lack of high quality well stated requirements, and few limits on the increase to project scope, also known as scope creep. "As long as you're doing that, why don't we just add this, too?" But a common failure was the lack of a simple agreement amongst stakeholders of what constitutes a success, in the beginning, so we're going to talk about that in just a little bit here.

The contract terms and conditions should be structured from the viewpoint of the project's end. If you're writing the contract or you can contribute to the wording of a contract that's going to go out for bid, while you're doing that, imagine yourself at the end. Imagine you're not at the beginning. Imagine you're at the end and you're trying to get the project accepted. At the end, you want to interpret vague acceptance terms when the contractor submits invoices. Or would you rather just audit a complete set of IEEE 829 documents, showing the expected results that are mandated by the contract? Projects typically include multiple suppliers. For example, actuated signal controllers from Vendor "A" and controllers from Vendor "B" to be installed in an NTCIP central system from Vendor "C."

You want to put it all together at the end and troubleshoot deployments, trying to figure out while you're out on the street why it's not working, or even in a lab bench. Or would you have each vendor provide test scripts for test equipment that verifies each interface by the vendor before system integration? So you could provide, up front, by imagining yourself at the end, providing the test scripts, the test equipment you want it tested on, and the expected results, up front in the contract. That way, everybody, every manufacturer that's working on it, can test it all the way through the development process. So at the end, you just have an IEEE 829 document submitted by each one, has all the level tests, and has the expected results filled in, certified. So you might say that that's going to be a lot of work up front, and if you go read this reference that I showed in the previous slide, you could easily consume 20 percent of your total project costs doing this up front work. But that study showed that the break over point was the projects that spent 15 percent or more of the total project cost doing the up-front planning in these formats, so that the people that are supplying the equipment can just fill in the test results, if they consumed 15 percent of the total project cost in that up front, it saved 15 percent or more at the back end, so you can see, if you're up to 20 or 25 percent, you're going to have a very smooth project.

Again, the contract should be structured from the project's end. Again, this is just reinforcing that good planning includes good test documentation and really should improve test documentation in the accepted IEEE 829 format in the beginning.

This graphic shows that the terms and conditions should include IEEE 829 formatted documents, just with fill in the blanks for test data, in the beginning. You could go as far in the beginning as to require the exact third-party test equipment you want each supplier to buy and the format you want the test scripts to be in. But again, write your terms and conditions on the left side here as if you're here. So by the time you go up the right side with all the level testing, level, level, level, all the way up, every stakeholder that's doing the design, the manufacturer, the system integration, the system configuration acceptance test, has that test documentation in their contract and they can just fill in the blanks going up the right side of the "V" here.

Again, additional concerns, again, we talk about third-party equipment, test scripts as deliverables, test procedures, and an example, the type of ASCs, ATCs, 2070s, etc. That can all be put in the up-front terms. Takes a little bit longer to do it, but when it's done, it's verified all the way through the whole "V."

We're going on to another poll here for another activity. The question here is, "Test Procedure Specification should be synchronized to the contract terms and conditions..." Why? Do we do that so the Project ending without unexpected issues, "A?" Or "B," do we synchronize the contract terms to minimize the project planning costs and time? Or do we synchronize the terms and conditions up front so that we can replicate the wording of a similar prior project to save time? Or "D," we would do that as an enforcement, a way to enforce after the project is late and over budget? So why would we spend the up- front time to synchronize the contract terms with the test procedures, up front, "A, B, C, or D?"

The answer is "A," "Contract terms including the tests allow the suppliers to test deliverables throughout the project, eliminating unexpected results at the end." "B" is incorrect, and again, we had our study. I mean, you can not do that. If you don't do it, take the time to synchronize it up front, it's not going to reduce overall project planning and time. It'll make your planning time shorter. But again, the study that we referenced shows that 15 percent of planning saves more than 15 percent, so we would not want to not do it not to save time. And then, "C" is also incorrect. Typically, you'll see good contracts and they just replicate a boilerplate of acceptance wording, really doesn't provide a clear acceptance criteria for differing equipment and software. So just putting the boilerplate in and then getting it at the end and putting it together and say, "I don't think it's working correctly," is going to be the wrong answer here. Then, finally, "D," we do not put the contract terms and conditions in to enforce design changes for an unexpected results. Using the terms to enforce design changes for unexpected results at the end is after all the design effort has been done, so if you're getting clear down to the end and saying, "I'm going to run my test now. I didn't tell you what it was going to be for sure, in any detail, and you didn't have access to the test procedure throughout the design, and I found a problem," that means all that design work, or lots, a big portion of the design work, was in waste, and that's how projects over run. So that is not the correct answer.

We're off to the Summary of this learning objective, and again, the objective was to Synchronize the Test Procedure Specification to the Contract Terms and Conditions for Successful Contract Execution. And again, we learned the challenges of Information Technology content of these projects we're working on as contrasted to their traditional roads projects that a lot of the contracts are written around, and we learned to use the terms and conditions and make those traceable back through the test workflow to user needs. We learned that tailoring the IEEE 829 standard documents to the test workflow and steps, up front in the very beginning, to include those in our contract will avoid ambiguities at the project end. And under the terms and conditions, each vendor is contracted to fill in the test results in the contracted format, using the contracted test cases and the contracted test procedures, while they're doing the design. Finally, we learned that writing the contract from the viewpoint of the project's end will improve the probability of success, so that the terms and conditions should include the test procedures in the proper formats, including the test scripts for the test equipment, in the beginning.

Now we're going to move on to Learning Objective Number 4, and here we're going to Write the Reports Produced at the End of Testing and Understand Their Relationship to Successful Procurement Contracts. In this learning objective, we're going to talk about test logs that include data, information, files, and needs that are captured during the test. We're going to go over the incident report, including a failure description and the investigation process, and we're going to do a summary report, providing a measure of success, based on the stated goals.

Reports produced at the end of testing and the relationship to procurement, here we're going to cover the methodology to create a test procedure from ITS Test. The inputs to the test procedure are one or more of the test cases that we taught in the previous prerequisite. For example, the test procedure might include Test Case 1 and Test Case 2 as inputs, and the procedure outputs are the data, information, and files captured during the test procedure, again, in IEEE 829 format in steps.

This is the familiar graphic from before. Now, we've added the outputs at the bottom, in green. The test workflow again is shown here. Again, Test Plan is followed by design specification, workflow is broken down into individual test cases. We've gone over that before, with the expected results. We talked about the test procedure documents in previous slides of this module, and the step-by-step procedures. Recall, we went through each step. And the individual test procedure may cover more than one test cases, producing multiple output records. For example, a real simple example would be if we're doing actuated signal control and we're looking at our-- making sure we're testing that the red phase is working correctly and that's one test case. You have to make sure the green phase is working correctly. That would be Test Case Number 2, and you might write one test procedure that tests both of those test cases, so it would be very prudent on the design to write the test procedure to say, "Set it up. Do the steps. Test red, yellow, green. Tear it down. Write the results." Test Procedure 2 can be used as Test Case 3, et cetera, on and on and on, like the example I stated. So the test case procedure outputs are shown in green at the bottom because Procedure 1 used two test cases and inputs, as you can see, it creates two output records. So Test Procedure 1 has Output Record 1 and Output Record 2. There's two cases, tests with one procedure that has two records that go one for each test case. Then, likewise, Test Procedure 3 used one test case as an input, so it produced just one output record. So each output record produces multiple outputs, as we saw before, such as logs, data, files, and other information. Again, we're following IEEE 829 here. You will hear things like, "Output records are called artifacts." That's used a lot. Like your output record, there's an artifact, Number 1 Artifact, Number 2, but these are called output records. So again, this is using 829 so that we're all on the same page and we don't have to quibble about what we're talking about.

Say we've run the tests and after populating the outputs for each of these test cases, we look at the outputs and these results are evaluated for anomalies. Again, recall earlier in this module, we said an anomaly is any event that occurs during the test procedure that requires investigation. Test procedure and test case shows what we think the output should be, what's expected. We ran the test, if what we got was not expected, it goes to an Anomaly Report. It's not a failure. It goes to investigation. Because of technical complexity of these ITS system, the test procedure is also likely to include data that's out of bounds for the expected, and again, these are compiled into an Anomaly Report, and the Anomaly Report includes a description of each failure, error, problem or defect that will be needed in doing the defect for investigation.

In IEEE 829 format, this is what an Anomaly Report would look like. Each Anomaly Report begins with an introduction that assigns an identifier to this report. All the identifiers have to have a different identification number. It provides a scope that describes, both, what is covered and what is not covered by this document, along with any references, such as ITS standards. "I'm testing an ASC, per the ATC 5201 Standard," for example. Then, next, the details of the report provide a summary of the expected results and lists any deviations found during the test procedure, along with the date the anomaly was discovered. So these details would include the context within the larger system and a detailed description of the anomaly in very precise terms and objective terms. What should it have been, what is it, in very precise terms. The test operator also provides an impact in urgency assessment, such as ranging-- might range between minor up to do not deploy, if the anomaly could cause a safety risk or a major impact to the success of the project. If the test operator was able to resolve the anomaly, a corrective action is provided. The status typically indicates whether anomaly is open, which means it's still under investigation and hasn't been resolved yet, or closed, it's been resolved and disposed of. The test operator will provide a conclusion based on the data output and a recommendation to fix or further investigate the anomaly. Then, general documentation control is history, so every time this Anomaly Report's been altered or somebody's changed something, you put your revision in the History Report. Again, this may seem like a lot of work when you're going through testing, and every time you find something that didn't match the expected results, you fill one of these out. But again, a lot of us have been in systems where doing testing we did not follow this standard, and a test operator said, "Well, I ran the test and I ran into a problem, so I shut it down," and it's followed by lots of questions and emails and meetings on what happened and what do we want to do. This captures all this by the operator, while the operator is running the test, so this would be all of the information that everyone, all the stakeholders are going to ask, to make a decision on how to dispose of it. The test operator might run the test, say, "This is a very minor thing and this is what I recommend to do," and the test operator may say, "This thing is broken. I would not put it on the street with this thing in it," and you have all the data so that all the stakeholders can read the same thing in the same format to come to general agreement on how to resolve the issue.

Again, the process must identify the person responsible to set the status, especially who is authorized to close it. I might say, "Well, you know, I don't think that's a big problem. Close it." But now, in the beginning, everyone, all the stakeholders must agree on who's authorized to poll everyone on the stakeholder team and who's authorized to say, "Yes. We tested it. We found this problem, this anomaly. We changed it and ran the test again. We resumed the test. We think it's correct," and there is a person that's authorized to sign off and close an anomaly. So the test operator, in the end, will provide a conclusion based on the data input and recommendation to fix or further investigate, and history is included. So that's an Anomaly Report. It's very important that those are used and the format of 829 is followed, and again, the whole ITS industry is moving towards this methodology.

Q: The Anomaly Report could have some additional information. It probably could have a description like the test inputs used, expected outputs. Now, this stuff has already came from the test cases. So the test cases already said these were the inputs, these were the expected outputs. So the test operator would be filling in the actual results observed. The test operator would say, "This is what it was supposed to be. This is what I saw. This is the gap, difference between expected and actual. What step in the test procedure did I see the anomaly? What was the environment? I was out in the lab bench or I was out in the field or I was--" whatever the environment was, "I was in the test chamber. I was doing hot and cold tests," whatever. "How many times did the operator attempt to repeat the test?" "I tested. I got the bad results, unexpected results, and I ran it again, and I ran it two more times to make sure I was doing things right." Because we all know, somebody would say, "Well, did you try it again?" "Well, that's all included in this Anomaly Report. Yes, I tried it three times. You don't have to ask me about that." And it identified the test operator who conducted it and if there were any observers. So maybe the test operator was there and the person who designed it, like the engineer, was sitting and watching and said, "Yeah, I observe it. I agree with this and I contributed to this Anomaly Report."

Now, we're going to move on to Level Test Reports that are produced at the end of testing. So the Level Test Report identifies gaps between the stated goals and the actual results, and the results are objectively quantified with very certain metrics, such as irresolvable anomalies within the available timeframe or budget, and those that are outside the defined scope. Level Test Report might include the readiness for deployment or an estimate of further effort, and should state the recommendations. So in the Level Test Report, this would be the output, so we talked about it in the previous slide and what the operator's going to do. But this would be the report that this is the output, so when the test is done, this is the Level Test Report and this is what would be included, so this is a document that's archived.

Again, Level Test Report, this would be the standard format for each level of testing. We have an introduction that has the ID, scope, references, details, overview, detail, rational, recommendations, and general, like the document control. So again, this would be the standard format.

Example of a Level Test Report, this would be like for ASC Timing. For example, we would have an introduction. We would have an ID, a scope, references. We would probably refer to the ATC 5201 Standard overview, detailed, rational, recommendations, and general. So if we were doing it for actuated signal control, we would fill that in.

This is a Master Test Report of the Level Test, so the Master Test Report is above the Level Test. It includes the various tests at the various levels. This might also be known as an Integration Test Report, but if we're following 829, that would be a Master Test Report. So a Master Test Report includes all the levels that are tested. So if you had seven different levels of a system we were testing at seven levels, the Master Test Report would say to complete the testing of the system, and in the Master Test Report, it would state things like, "You must complete Level Test 1, Level Test 2, and Level Test 3," so again, it has a document identifier. Overview of the aggregate tests, rationale, conclusions and recommendations, and glossary, and revision control again. One thing to know about the Master Test Report, it's not always required to have a Master Test Report. Small projects, you can just have one or two or three level tests. Master Test Reports are for complex projects where there's lots and lots of levels and the test operator really needs to know how all this stuff is related to each other, so you would have a Master Test Report showing how the level tests go together.

We're going to do an activity. "Which of the following statements is FALSE?" And again, we're looking for a false statement, so is it false that "Only one Test Case is used in input to each Test Procedure," or is it not true that "A Test Procedure can use multiple Test Cases as inputs?" Answer either "A" or "B."

Correct answer is "This statement is false," "A" is false. "Test cases that are independent of each other will likely be the single input to a test procedure. However, several dependent test cases are often used as multiple inputs to one procedure," and we talked about that in Signal Controller having testing the different colors. So that makes "B" incorrect. "This statement is true." Test cases that require similar test equipment with related functions can be tested with one procedure.

Summing up Learning Objective Number 4, in Learning Objective 4 we studied the methodology used to create a test procedure for ITS tests and its relationship to IEEE 829 Standard. We also learned to execute the test procedure and to populate the data to create output reports. We also learned to identify incidences that require further investigation in an Anomaly Report and the description of the investigation process. Any incident that occurs becomes an anomaly, so there's an Anomaly Report that investigates anything that's out of bounds. Then, finally, we learned how to write a summary report that identifies any gaps between your results and the stated and expected inputs.

Moving on to our final Learning Objective Number 5 for this half of the training course, we're going to use a Test Procedure Generator to develop the test procedures for a sample Test Procedure Specification structure. The Test Procedure Generator is a tool and it's available from USDOT. We're going to talk a little bit on how to get the distribution of this tool. It runs on a personal computer and we're going to talk about just here, the purpose of the Test Procedure Generator. We're going to talk about how you can obtain a copy so that you can use it between the end of this course and the beginning of the next course, where we'll be using it more. Right here, we're going to, kind of, do a quick overview of how to install it. We're, kind of, going to explain each step and the results, and we're going to go through a very simple how to handle an error in a sample Test Procedure Specification structure, and then how to understand pre and post conditions and different types of steps.

The Test Procedure Generator is used to guide-- and again, this is used as a guide to the development of test procedures in a manner that's uniform and generates XML scripts that can be consistently interpreted for use on automated test equipment. So any NTCIP standards that have a MIB, have standard dialogs, requirements, and a Requirements Traceability Matrix can be typically used on the Test Procedure Generator. If you've been keeping up with the NTCIP standards, some of the ones that have been around for a while started out by not really going through the full system engineering process, and you'll notice that some of the later ones have been through the system engineering process, and some of the earlier ones are undergoing modifications to make them fully conformant with NTCIP 8002 Annex B, so they can be used with the Test Procedure Generator. Now, that sounds like a lot, but we'll show you that as we go along here. It's a software tool developed by USDOT, guides in the development-- so that if you have a Management Information Base and standardized dialogs and a Requirements Traceability Matrix, defining the performance, we can use the tool. And the powerful value of this tool is when you run the test, it automatically creates XML scripts so that the output is the test results in a standard format. So again, that would go along with having a consistent interpretation among stakeholders who can all know that these scripts that came out, we all agree that that's the standard format and we understand what they mean. It eliminates errors of manual creation of scripts. I mean, I have created scripts where I've checked it myself, but someone runs the test for the first time and comes back and said, "Did you really mean this?" "Oh, yeah, I made an error when I keyed that in." So this Test Procedure Generator, if you try to do things like that, it'll flag errors so that if you're trying to put something in that's non-conformant, it'll flag that so the data is entered correctly. And then, again, the output can be used on automated test equipment so you can rerun the equipment over and over again, whenever you want to, which is really great for regression testing later on. You've got this system that's been on the street for a long time, you add to the system, the test operator can just install these scripts again on his test equipment and run it again and look at the output. It goes back to the engineer. She can look at it and say, "Yeah, that looks right." Automated can be done very quickly so it eliminates a lot of the manual processes.

The Test Procedure Generator generates uniform test procedures based on a common understanding so, again, if you try to put in keywords that it doesn't understand, it's going to say, "I don't understand that." Although the ITS standards were developed with great attention to detail, there's a lot of differing interpretations, resulting in differences in message content by multiple vendors. Test procedure generates consistently uniform procedures based on the understanding of the ITS standards, and what we mean by that, that standards have a common understanding of the standards that have been through the system engineering process. So the standards went through user needs, requirements, a Requirements Traceability Matrix, requirements go into test cases, test cases go into test procedures, test procedures are run by Test Procedure Generator automated tool, TPG puts out XML files that everybody understands and can go into automated test equipment.

A copy of the TPG can be obtained. We'll show you where that is in a later slide here. If you want to get a copy, it must be initiated with a request. Identify as a requester; who are you? For their records. They want to record who you are and your email address for, as this goes along, you can get notifications for updates. When a new version comes out you automatically get a notification that there's a new update. This is what it includes. These are bugs found in the old one, whatever. The request is acknowledged and you can get with a method to download a copy of the TPG with the installation instructions and operations manual. So again, this little graphic down in the right hand corner, we're not going to go into this in any detail. But we're just going to, very quickly, here at the end of this training module, as Part 1, we're just going to, kind of, show you what this looks like. So after you conclude Part 1 here, this is what you will be doing. You will request a copy of this, and what you will do is you'll register yourself. Your email will get instructions on how to obtain it. You'll download it. It goes through an Install Shield wizard so most everyone that runs a Windows computer is familiar with this. It goes through an install wizard, as shown here.

What you will do is you'll open a new. So once it's installed, it'll say, "I want to open a new session." This one, I know you can't see it and there's no reason to see it right now. Once you get a copy of it, you can play around doing what I'm-- we're just giving you an overview of what to expect when you get it. So this particular one, they're opening a session that's NTCIP center to field device interface. So it gives you a selection of different standards, so that's really nice. First of all, it says, "Okay. I got this piece of equipment. I'm trying to do center to field," or "I'm going to do a test procedure for an actuated signal controller, so I can select NTCIP 1202." So you can select a standard and from there, it's really good because it gets the standard. It knows the conformance groups. It knows the dialogs, if they're there. So you don't have to go through all that manual stuff of searching through the standard trying to find the dialog. So if you give it a standard, it knows what things it needs to test within that standard. So when you get your copy, it's going to look like this, so you can start a new session, as shown here.

Then, again, this is probably too small, but again, this is just to go over, kind of, the workflow of how we would go through this. This shows opening a standard and it shows the commands being executed, in the progress window. So we said we want to open a new standard. It's going to go out and find the standard so it's going to process-- I opened the standard here and you can see its monitor, light sensing error, monitor controller, software operation, climate control systems. So it's bringing in all the tests. I said, "I want to look only at open a new session. It's testing this piece of equipment, and I opened it and then, it starts bringing in all the tests. You'll see them come in, in this installation wizard window.

Then, the next thing you do, you have a menu where you can pull down and say, "The next thing I want to do after I've opened a session, I've identified the standard, I want to start writing a set of new test procedures." So when you do that, it actually brings up a copy. So it brings up the copy of NTCIP 1203 Version 3. So that's really nice. You don't have to go around and search through all the website and try to find the version. The Test Procedure Generator has the version in it. You say, "I want to open a new session and I want to start writing a set of test procedures that pulls the correct standard up."

As we learned, test procedures have an identification number, so this is going to do the IEEE 829 format for you and the steps. Not really a format, it's really the steps. As you recall, we had IEEE 829 says you have to have a test procedure identification number, you'd have to have a description, requirements, pass/fail criteria. So the Test Procedure Generator is going to bring that up for you, and again, this is too tiny to see, but we're just going through. When you get your copy of TPG, between the end of this training session and the beginning of the next one, you can go play with this. This is the type of stuff you will see, so it'll actually bring up 829 formats, like shown. Then, once we bring up a test procedure, it's assigned a number so that they're all different. The next thing we're doing here in the wizard is define the test procedure header, so it's Test Procedure Number 1. We fill in the header.

This shows a header definition and window. Again, it's too small, but you can get the idea. When you bring it up yourself, you'll see what this is doing. Test Procedure Number 1, and the header is determine sign, type, and technology. So that's the header and the description. My first test procedure is to determine the sign, type, and technology, make sure that works, and the description of the test is, "This test case verifies that a Dynamic Message Sign indicates that it is the sign type and uses the technology as required by the specification." So you fill that in.

Then, the next thing we do after we define the test procedure header, we select a requirement. So here's a requirement. We select a requirement. Okay. One requirement might be to configure the logging service, and we select that from a drop down menu, and once you select that, recall we already had populated the test procedure, the header, the description of the test, and now the requirement. You can see how that's been filled in. So to run that test, the Test Procedure Generator from the standards is going to know that it has to test how to retrieve the data, how to deliver the data, explore the data, determine the current configuration of logging service, and configure the logging service. So it already knows that those five requirements have to be tested to make sure that it meets that description.

Then, the next thing, we would drop down, we'd look at variables. So then, we would run the test from the variables, so we would populate the variables, like the test case test procedure said. In the test procedure, we want to put the variables as inputs. We want to run the test and then see the outputs and then, we save this procedure.

Then, once we're done, after we've saved it, we can close it. Then, after we've closed it, the next we do, after we run the test, we would save the set of XML test procedures. So you can see this would be the output. Again, we're running through this very quickly and we're not training this, so this is the training you're going to get and we expect those of you that are going to move on to the next half of this course to get the download for this and do what I'm showing here in these slides, yourself, and it's pretty simple. You get the instructions, and it's very user friendly, and you should be able to do this. This is the output so what we would get is the test output, and recall earlier in this course, part of 829 is we had our test cases, we did our test procedures, we had our inputs, we ran the procedure, and then there's output. So this puts the output test in a report that's in standard XML format, so you can see it shows the test procedure, shows the version, the TPG version, the ID, the standard major, the standard minor, test procedure, and then lists what the tests were and populates the results. That's what the outputs look like in XML format and, again, this is a format that you can use by an automated, in TCIP Tester, for example, to put it in and it will run the test and give you the results to see if your input data matches what's expected.

In the TPG, you can also do other functions not shown in these screenshots, and again, once you get your copy you will be able to explore this in more detail. So it allows you to delete a test procedure step, if you're in the test procedure and you want to delete a step. You can resort the procedure steps. Like, you decided, "Well, I need to really do this in a different order," you can go back and pull down a menu and resort them in a different order. You can renumber the test procedure steps. You can take any test procedure identification and modify it to a different ID number. You can go into a test procedure and edit it and you can print it out and you can print it out onto paper or you can print it out into a PDF file, if you want that for your records.

As seen in previous slides, each test procedure step is constructed of keywords. Because the syntax inputs is checked by the TPG against a uniform set of keywords, all the ambiguity and chances for error are eliminated. So each keyword embodies the common understanding of the standard or the defacto correct, quote/unquote "interpretation of the standard." Because the TPG enforces consistencies of keywords, the consistent test procedures will be created as a result. For the example we used, an application of keywords will be shown and explained. The keywords will be used both with and without the syntax to see the effect. Again, the TPG has what is the most common understanding of what the standards interpretation means. So if you have a different understanding, it's going to flag it as an error, so you have to put in the keywords correctly.

We're going to have a final activity here. Again, here we're looking for the true statement, so which of these four statements is true? The Test Procedure Generator "A," "Takes Test Cases directly as inputs," or "B," The TPG "Guides Test Procedures having MIBs, Dialogs, and a Requirements Traceability Matrix, or "C," The TPG "Creates outputs in commonly understood XLS" spreadsheet format in files, or "D," The TPG "Executes Test Procedures automatically," which of those four is the true statement?

Review of the answers; correct statement is "B." The Test Procedure Generator is used to develop test procedures with uniform keywords, dialogs, and a Requirements Traceability Matrix to eliminate manual entry errors and to enable test procedures to be reused."A" is not correct. The TPG does not read in test cases. It pulls information from various sections of the standards and reads non-TPG created test procedures that you may have entered in those fields, after you've installed it. That's not correct. "C" is also not correct. The TPG creates outputs in commonly understood XML scripts. It doesn't do outputs as XLS spreadsheet files. It's XML. The XML scripts are the ones that can be read by automated test equipment. Spreadsheets cannot. Then, finally, "D," is also incorrect. The TPG does not execute test procedures. The TPG XML scripts can be used as inputs to automated test equipment. So we've stated that several times.

Our Summary of Learning Objective 5, in this learning objective, we were introduced to the Test Procedure Generator. TPG is a tool developed for USDOT that is available for download at no cost. We learned how to obtain a copy--we'll go into that in a little bit here--of the TPG and how to install and use the TPG. We learned the great value of the Test Procedure Generator producing consistent procedures that are interoperable among vendors.

As a summary of Part 1 of this course, a Test Procedure Specification inputs are test cases that are used to create outputs of expected results and anomaly reports of unexpected results in IEEE 829 standard format. Number 2, A Test Design Specification details what a test is to demonstrate, a Test Case Specification is a specific example that assigns values, while a test procedure defines the steps to perform the test. Number 3, Contract Terms and Cods should be viewed from the project's end, including Test Case Specifications and test procedures. Number 4, A master test report measures project success to stated goals, and a Test Procedure Generator is an automated tool that generates XML scripts using consistent key words for interoperability.

This is a list of our resources. You can look through all of these. The one that I took the graphics on, the studies of the pie chart and the projects that had failures and how to make your projects not be failures, is the 1, 2, 3, fourth bullet down, "Systems Engineering for Intelligent Transportation Systems. It's on the FHWA website. I encourage anyone to read the first 15 pages of that, at least.

Again, this is a two-part module. Part 2 is How to Develop Test Procedures for ITS standards-based Testing. That would be Part 2 of 2 and that will cover Learning Objectives 6, 7 and 8 that we would lay out in the beginning slide.

The next course module, we're going to have some homework for you, so between the end of this one, Part 1, and Part 2, we will use the TPG to create test procedures for different roadside equipment using the standards. We'll probably do an actuated signal controller and maybe a message sign, things like that. Between now and then, just between now and the start of Part 2, please request a copy of the TPG and register email address to receive updates. The link is shown in this slide, right here that you're seeing, as well as in the student supplement. After you download it, install it on a computer, and go ahead and create a simple test procedure to become familiar with what the TPG is for inputs and its operations and take a look at the outputs. Just do a few of them before the next one. Please have it installed and running at the beginning of Part 2, as we're not going to spend any time at the beginning of Part 2 taking any questions on the TPG installation and operation. So please don't download it an hour before and say, "I can't get it to run and I have a bunch of questions on that." We want to see if you can have it running when it starts and up and running and all that installation problems behind you. I've personally done it and I really didn't run into any problems. It has an installation wizard and you can, pretty simply, select your standard and write some procedures and get some XML output, so please be prepared to do that at the start of the next one. So any questions, we should try to resolve them before Part 2 begins.

Dave Miller: That's the end of this one. We'll go ahead and take questions at this point.

End of T204_1_1.m4v