IEEE P2848 Design Review Meeting

Teradyne, North Reading, MA

2024-05-02

Attendee list

Anand Jain, NI anand.jain@ni.com

Chris Gorringe, Spherea chris.gorringe@spherea.co.uk

Chris Walds, USAF christopher.walds@us.af.mil

Damien Willemet, MBDA France damien.willemet@mbda-systems.com

Eric Gould, DSI Int egould@dsiintl.com

Gilberto Garcia, US Navy gilberto.garcia1@navy.mil

Gokul Subramanian, Lockheed Martin gokul.subramanian@lmco.com

Ion Neag, Reston Software ion.neag@restonsoftware.com

Jack Schuster, Teradyne jack.schuster@teradyne.com

Jean-Christophe Hertzog, MBDA France jean-christophe.hertzog@mbda-systems.com

Jordan Geeson, Lockheed Martin jordan.geeson@lmco.com

Larry Adam, USAF william.adams.41@us.af.mil

Mike Seavey, mike.seavey@gmail.com

Sylwester Sobolewski, USAF sylwester.sobolewski@us.af.mil

Teresa Lopes, Teradyne teresa.lopes@teradyne.com

Record of Discussions

Ion Neag gives an introduction

> 2848 is the working group for a standard for prognostics and health management in automatic test systems.
>
> This is not a formal meeting of the working group.
>
> John Sheppard is the working group chair.

What we're trying to achieve here he is to review some of the work that we've been doing, to use existing ATML elements to support the PHM scenarios.

We don't need quorum and we don't take votes.

Ion displays patent slides that were also shown yesterday in the plenary.

Ion shows a few slides with an introduction.

We are talking about 2 main use cases.

One is UUTs that we normally test on ATE. We look at test results and prognose future failures of those units as we test them. The units passed, but we're looking at their graduation and we can see that something is degrading. And we prognose that it that many hours it may fail, with some level of confidence.

The other one is the failures of the AE itself as we run self test and calibration we collect data. That's actually a lot larger in volume and frequency than what you get with UUTs. We support algorithms that look at that data and see some degradation in switches, accuracy of instruments, things like that.

The standard does not aim to specify any algorithm that can do that. Whether it can be done or not, that's a question that we're not trying to answer now in this meeting.

After some discussions in the working group meetings, it was decided that this would be a normative standard. It will use existing ATML and SIMICA component standards.

To represent data that are either inputs or outputs of prognostics, it will specify extensions to the schemas for data that are specific to prognostics.

The outcomes will be one clause that mandates the use of specific ATML and SIMICA components for specific types of data in the prognostic use cases.

The second should be a schema that defines prognostics-specific data items like prognostic models and prognostic results.

As I said before, we assume that there are algorithms that can prognose failures. We do not attempt to specify those algorithms; we represent them as mostly black-box prognostic procedures.

There is an entity called prognostic procedure. It has some inputs. Has an Outcome. We do not mandate the algorithm. However, we need to be aware of algorithms that exist, in order to be able to define meaningful inputs and outputs.

The prognostic procedure is assumed to have two operating modes. One when it is developed or improved in time, which basically does the algorithm for a specific prognostic subject. We could have an algorithm that is generic in nature, that follows some kind of curve, but the threshold of failure for that is specific to a given instrument or a function of that instrument. The algorithm calculates the threshold from historical data. It looks at historical data and it calculates some parameters.

The second mode is execution. That's when the algorithm would run. For instance, running at the end of a self test or calibration. Now I have the data from this TPS. I'm running the algorithm to see what I can predict a failure. The communication between the first operating mode, the development/maturation, and the execution are parameters. Development calculates them. Execution uses them.

Ion presents the design documentation that has been developed

The prognostics subject is the failure am I trying to prognose. And it's different in the two use cases.

In the first use case the subject of prognosis is either the UUT, the which isn't very helpful, or a specific component of the UUT. An even better subject would be a failure mode of that component, or some function. We could also prognose failures of the ITA, looking at things like connectors, pins, relays, maybe active components.

In the second use case we prognose the ATE, so the subjects could be any instrument, a component in an instrument, the failure mode of that component, or a function. And we can look at prognosing failures in the ITAs for self-test and calibration.

The inputs that the tuning of the algorithm uses are typically historic measurement results, but there is a case to be made of also using historic maintenance actions, because looking at those will tell me first when a component was replaced. So the life of the new components now starts at zero. I can also look at the results before the actual failure and see how much they degrade.

in ATE testing, we have units that actually failed. This is generally difficult to achieve in the general prognostics use case, because we need to run the units to failure to verify the prognoses.

The outputs of the development and maturation mode are the parameters of the prognostic model and we looked at different cases. You could have a data series; horizon or distance to failure plus a confidence. We looked across different types of

algorithms. Ion points out that this part of the design needs to be reviewed by domain experts.

In execution mode, when we run the prognostic procedure on, the inputs are the parameters calculated in development mode, the current measurement results, and possibly what was measured in the past, to determine some kind of trend. It depends on the algorithm. The other input is maintenance actions, to know when that component was last replaced. So I know where the life starts at zero.

The outputs will be predictions for the subject of prognosis, and here we have several options. One is an estimate that the failure will occur at some time – for example what's the confidence that this will not fail through the next mission. The next is an estimate that failure will occur before a specified time horizon, with a specific confidence, and there could be multiple estimates and the horizon moves farther. The confidence increases as you get closer to failure.

Eric Gould indicates that he thinks of horizon as the time to failure. This needs to be clarified.

Ion: The other one type of output is the estimated remaining useful life.

Larry Adams: This is very interesting because maybe the formula will be thrown off, because you may be using an instrument that was used before in other stations, and think you are putting a new instrument. The formula is going to be skewed. Ion indicates that there is a way to track the instrument instance across all stations. This will require a pretty complex infrastructure, you will have to track your station instances and all your instrument instances and all your maintenance data, which is hopefully complete; this is going to show that an instrument was moved but was not repaired or renewed.

Discussion on prognosing instrument failures.

Ion continues the presentation of slides. Looking at “Design: Referenced Instance Documents”

We are now looking at which ATML components can we use. In ATML, instance documents are typically XML files that conform to one of the standards.

Test results - that's where your input data for your execution mode are. Also you will see that we proposed to extend test results to also be able to store the prognostic results; we are trying to have some commonality between the way we store diagnostic results and the way you store prognostic results.

Discussion on Test Results and Test Description.

> Ion: We are trying to define an extension type that describes a prognostic procedure in the similar way that we can describe a diagnostic procedure. There is a test group for Serial, a test group for sequence, maybe a new test group for prognostic. Also in the test description we can identify the prognostic targets.

Discussion on MAI.

Chris Gorringe: MAI should be a minimum requirement, along with Test Results; without that, trends in Test Results cannot be interpreted.

Ion presents slides on prognostics subject and parameters. Discussion.

Ion presents the concept design document; the use case of the UUT on ATE.

> What we propose here is storing prognostic results in an extension of the Test Results schema. Similarly, Test Description is used to describe test and diagnostic procedures; we propose an extension to it where we describe prognostic procedures.
>
> Ion expects the prognostic model to be a separate document.

Chris thinks there's one more item in the minimum requirements - the instance documents. without those you cannot translate.

Discussion on model maturation. Ion thinks this will run infrequently. The results need to be evaluated before being deployed for run time.

Discussion on using estimated RUL when selecting a used instrument from a set of spares, for installation in ATE. It is important to keep track by unique serial numbers.

Ion presents the concept design document; the use case of prognosing the ATE. Indicates that the documents will be distributed with the minutes,

Larry Adams points out that the accuracy of measurements on the UUT is influenced by the state of the ATE, so the UUT algorithms may need to also consider the state of the ATE. There could be noise, of measurements drifting, while remaining within calibration limits.

Group discussion. We may get slight difference from switches. We may see noise from maintenance. We may see noise from calibration changes that has to be accounted for somewhere, in order to make things work. Environmental changes are causing problems too. Prognostics of the ATE could help here, because it is degradation within tolerance.

Returning to the slides on “Design: Prognostic Subject”

We're looking at how do we specify the use of existing ATML elements. For the prognostic subject, we have UUT components and the failure modes of the components. We also have functions that are available at the UUT port and we can describe the failures of these functions.

Ion presents the UML design document; the use case of the UUT on ATE.

What I propose is that similar to the test groups for diagnostics, have a test group of type prognosis. The group will have steps. Those steps represent the execution of prognostics, and they will have outputs.

Returning to the slides on “Design: Prognostic Subject”

Discussion on prognostic subjects. The UUT Description defines capabilities. It is specified as being available at UUT ports. Has an optional 1641 signal description. Chris recommends adding this as a subject because it works better when extending the set of TPS measurements for the purpose of prognostics.

There are also functions under connector pins; they can have signal information, but this is not a 1641 signal description, it's just a signal name and a signal type.

Chris suggests that we could extend Instrument (ex. through xsi:type) also represent a UUT Description. That will be able to describe Faults and Failures.

Ion presents the slide on “Standard Contents”.

How to turn this into a normative standard. There are two there are going to be 2 parts to this. The UML diagrams and the schemas.

Probably 2 schema files. One will extend test description the other one test results.

Because we create the schemas, we have to reference a namespace for ATML and a namespace for SIMICA and I think it will be a challenge to deal with changes. But practically I think what we can do now is referenced the latest versions.

In the text, reference the component standards without version, which means the latest version available. In the schema, reference current versions. Reference 2018 SIMICA version that uses ATML Common. A future amendment to reference revised ATML schemas will only impact the 2848 schema files.

The second part is how to write down the clause the standard that describes the use of existing ATML components. Take the UML diagram, split it into small diagrams that are self-consistent.

Looking at the conformance section from the root 1671 standard. Emulate that because it has some prescriptive statements. For example: this instance documents conforming to test results shall be utilized to store the test results for the prognostic procedure. Or: should maintenance data be provided, the format specified in 1636.2 shall be utilized.