Where's the Interoperability for Asset Management?

Charlie Piper, Invensys Process Systems

THE PROBLEM

Standardization for fieldbuses, such as FOUNDATION fieldbus, Profibus, HART, and others, has delivered on the promise of interoperability for both run time control and device configuration. In mixed vendor environments, with a host system from a different vendor than the device vendors, control loops shared between the host and the devices can readily communicate together in real time to run the plant. Run time interoperability is delivered by standardization of the communication protocol on the fieldbus network, ensuring the ability of the host system to receive process measurement values from the devices and send back control outputs. Interoperability for configuration is also provided via device description languages (DDL) identifying the name, data type, and possible settings of configurable parameters. For some fieldbus types DDL also provides code that can be interpreted by the host in order to execute fundamental device commissioning actions, e.g. to automatically calibrate the travel of a valve positioner to match the actuator and valve it is mounted to.

The problem is neither the bus communication protocol, nor the device description languages, provide several key areas of asset management functions in a cross vendor interoperable way, such as;

- The ability to run advanced analysis and optimization software used to improve the performance and setup of a device.
- The ability to run advanced software to diagnose devices, e.g. diagnostics that can analyze valves for high friction, hysteresis, seat loads, or deadbands.

- Providing standardized APIs and interfaces to host applications handling functions such as advanced diagnostics, predictive diagnostics, maintenance planning, audit trail tracking, database record keeping, and device engineering.

In many cases software for the complete solution can not be completely device resident nor completely host resident. Rather cooperative software in both the host and the device is necessary to achieve the full benefit. The host needs software to interact with the user initiating the diagnostic tests, and software that presents and interprets the test results. Predictive diagnostics needs host software to handle alerting personnel and generation of work orders. The problem is, in most of the systems on the marketplace, these important asset management functions are only possible if devices are bought from the same vendor as the host system, completely defeating the purpose of having standards for fieldbuses in the first place.

Device Description Languages are both part of the solution and part of the problem. Clearly they are vital to define the names and types of the data that reside in the device. Without DDL it would be impossible for hosts to provide a single engineering application usable for all vendors' devices in a consistent way. As a programming language, DDL is also important in delivering to the HMI the menus and dialogs needed to run DD methods. DD methods are interactive routines for a host user to trigger simple interactions with the device, e.g. automatic sensor calibration.

The total solution for advanced asset management and diagnostics requires something more than just DD languages. The application programmed into a DD file is a standalone application, having no ability to interchange or interact with other host programs, since the DD specifications have no definition of interfaces to be used for interaction with other programs. Asset management requires the periodic storage of device status and health, yet without a defined interface for storage and retrieval within the host, the DD can not accomplish that role. Assuming host software for advanced diagnostics should be provided by device vendors, and that it should easily plug into and interact with other software applications on a host system from a different vendor, the DD language specifies no interfaces that help make this happen. There are no interfaces defined to launch the 3rd party software from the asset management

application, and no interfaces to handle the display of the software within the screen space of the asset management package.

The best solution also requires advanced applications that are not constrained by limitations imposed by the technology standard. A proven model for standards is one that defines interfaces between different system components, conceived as blocks or entities of functionality, where the designer has to conform to standards at the interfaces, but has flexibility internal to the block of functionality for innovation and creativity. DD standardization does not use this approach. Rather the DD languages define what you can do inside the block of functionality, i.e. they are a language for use within the functional entity. No matter how far the language is extended, if the capability is not in the specification of the DD language, than you cannot have a feature needing that capability in the application programmed via DD.

THE FOXBORO SOLUTION

On typical host systems, the DD files need additional surrounding software, as indicated in Figure 1. The first layer is an interpreter specific to the computer's operating system. In the case of FOUNDATION fieldbus this interpreter is built from the DD Services developer's kit from the Fieldbus Foundation. Outside this layer, most hosts have proprietary interfaces.

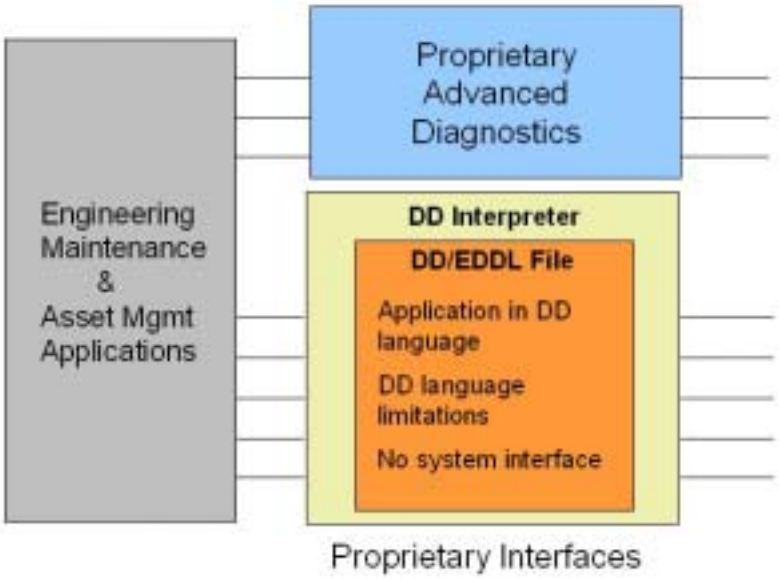

Figure 1 – Typical System
Proprietary Asset and Device Management

However, in the Foxboro system we have used standardized interfaces surrounding the DD interpreter, as shown in Figure 2. In layer 1, Foxboro has built the FF DD interpreter based on the Foundation's services kit. The second layer is software that builds the interfaces to other modules in a complete system having engineering, maintenance, and asset management capability. Foxboro uses another standard in this second layer, the FDT standard.

Foxboro has embraced DD and extended DD languages for their capability as a programming language defining device data structures and methods. We have also embraced technology from the FDT standard, for its part in defining interfaces between different software components, including 3rd party software components. Putting both together we get both the benefit of a standard programming language device vendors can use inside a file delivered to the host, and also the benefit of standardized interfaces between different software applications within the host system. Based on this principle,

our new host software for FOUNDATION fieldbus has the following capabilities, solving the interoperability problem for advanced maintenance and diagnostics.

First, our system can full configure devices from any vendor, using Foxboro software with a completely consistent look and feel, based on interpretation of the DD language supplied by the device vendor.

Second, our system can fully run all methods and programs supplied within the DD and enhanced DD languages, as supplied by the device vendor. The Foxboro engineering and maintenance software provides a consistent look and feel to expose a graphical user interface to those methods, plus offers 1) an area within the window to dynamically watch during method execution, with both tabular or trend line format, a set of user selected dynamic parameters, and 2) another area within the window which shows message transactions between the host and the device during method execution.

Third, our system can fully support plug-in of advanced maintenance, asset management, and diagnostic applications dreamed up and delivered by device vendors, that conform to the interface standards of the FDT specification. Like other function block approaches that standardize the interfaces rather than the functionality inside the functional block, applications already on the market prove this approach supports maximum innovation and complete freedom from functional constraints, useful in the area of building device applications supporting advanced asset management. Best of all, it accomplishes this in a way that the best functionality that device vendors can offer, both non Foxboro as well as Foxboro device vendors, is fully available on the Foxboro host system.

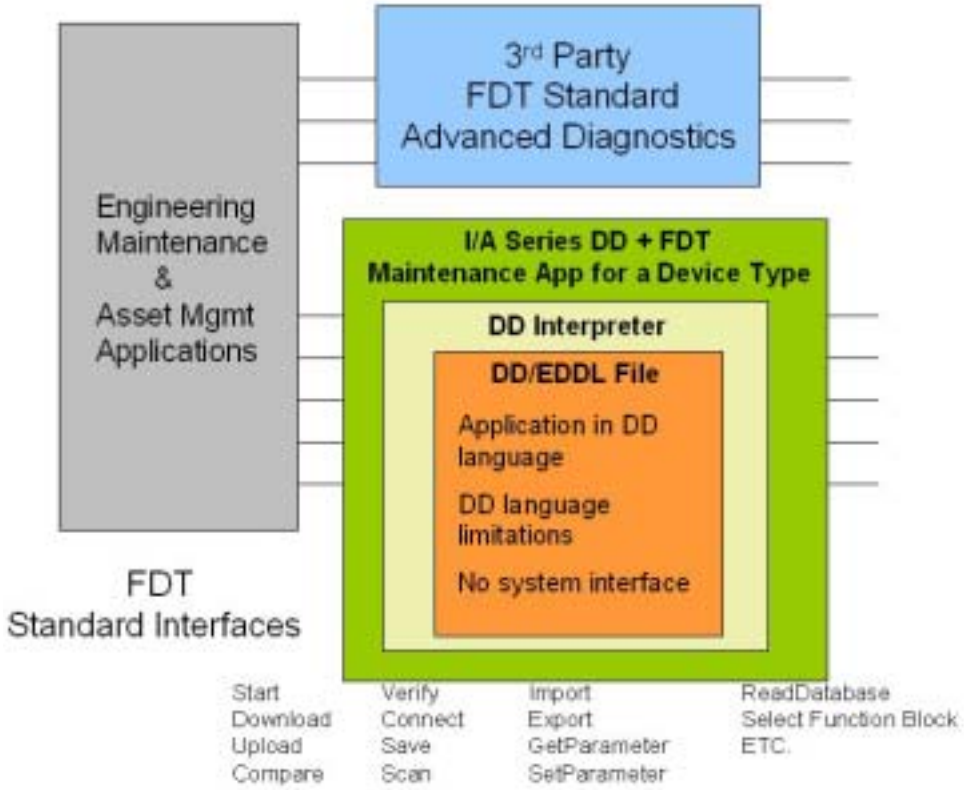

Figure 2 – I/A Series DD + FDT Based Asset and Device Management

For more information contact:

Paul Miller, Invensys Process Systems,

508-549-6540, paul.miller@ips.invensys.com.