Service Complex System

Juan Aguilar, Gerd Bottenbruch (Sontheim Industrial Electronic, Inc.)

The complexity of electronic control functions of agricultural machines has been increasing dramatically over the past years. Some of these changes have been driven by statutory provisions like emission related legislation. The introduction of additional sensors and actuators as well as additional control mechanisms has driven the increase of internal complexity of subsystems such as engines. The external interface of such subsystems is becoming more complex as well; even if only a few of CAN messages are required to integrate such subsystems into an agricultural machine. Besides engines, there are other subsystems such as positioning/navigation systems, electric drives, etc. that will create a very similar situation. In addition, the electronic architecture will no longer be determined by the agricultural machine manufacturer. Therefore, the internal communication of each subsystem might be based on different communication protocols and mechanisms.

One of the biggest challenges that these complex subsystems pose is the ever increasing difficulty faced by service technicians to support these systems during their complete lifecycle. Since agricultural machines are typically used in rural areas without a tight network for subsystem service and support, this task has typically become the job of the service department of the machine manufacturer. Therefore the integration of subsystem diagnostics into a machine service tool becomes more important than ever before. In this paper, different approaches to share diagnostic knowledge will be described, which are based on experiences with such complex systems. The main challenge is providing access to a few internal signals on the same or different communication protocols, and sharing test sequences and flash download procedures for integration into the service tool. The integration of these subsystems is limited by the underlying technology of the service tool. The usage of standardized methods like ODX (ISO 22901) for the description of diagnostic data/signals and upcoming standards like OTX (ISO 13209) for the description of test sequences can ease the integration process significantly.

In order to explore the methodology for the integration of subsystem diagnostics into a machine service tool, two different use cases are described. These use cases are examples of real issues that machine manufacturers face regarding their suppliers of subsystems and the task of integrating their respective diagnostics. While these use cases do not encompass every possible scenario, they serve to illustrate the possible complexity of the system and the types of issues that could arise, as well as demonstrate how the described approach and methodology for the solution can be applied to the different situations.

Use Case Descriptions

In the first scenario examined, a machine manufacturer might obtain one type of subsystem from different suppliers depending on the type of machine being developed. For example, a manufacturer of agricultural tractors might have a range of tractors of different sizes and capabilities. The manufacturer uses engines from different suppliers for the different classes of tractors that it produces to best suit their needs and specifications. In this example, the engines, while they might be different and come from different suppliers, represent one type of subsystem.

Since the different engines all represent the same type of subsystem, the goal of the engine manufacturer is to create one diagnostic tool for this subsystem which will be the same to the end user (technician) for every machine that is equipped with this type of subsystem, regardless of the specific supplier. In this example the machine manufacturer wants every vehicle to have the same engine diagnostic tool independent of what kind or brand of engine each vehicle contains. This concept is illustrated in Figure 1.

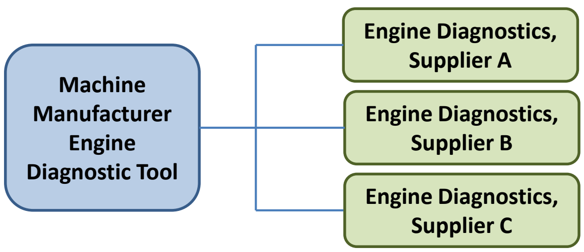

Figure 1: Use case 1, one engine diagnostic tool for multiple engines

Each of the different engines contains a set of diagnostic information that is common among the different suppliers, brands, and models. This is the information that will used by the engine diagnostic tool created by the machine manufacturer. This concept can be further generalized – each type of subsystem contains a set of information used for diagnostics that is common among the different suppliers of the same type of subsystem. Therefore, in theory, a general diagnostic tool for each type of subsystem is possible regardless of the specific supplier of the individual subsystems.

The second use case deals with a machine manufacturer that requires different components or subsystems from several different suppliers. For example, the manufacturer of an agricultural tractor might use an engine from supplier A, a joystick from supplier B, steering column from supplier C, etc… Each of these different subsystems (engine, joystick, steering column, etc…) is controlled by its own ECU, and each contains its own set of diagnostic subfunctions. The goal for the agricultural machine manufacturer is to be able to integrate all of the different diagnostic subfunctions into one diagnostic tool such that the end user does not need to access each subsystem individually using a different tool for each one. This concept is illustrated in Figure 2.

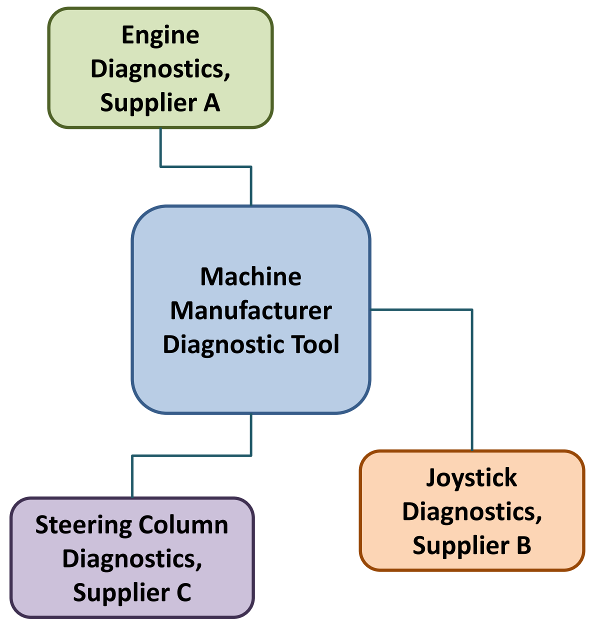

Figure 2: Use case 2, one diagnostic tool for multiple subsystems

In order to be able to integrate the different diagnostic subfunctions from all of the different subsystems on the machine, there needs to be a sort of standardized mechanism to access the diagnostic subfunctions. This could potentially be a problem if the suppliers of the subsystems do not provide information regarding the subfunctions in a standardized way.

Best Case Scenario

The two use cases described illustrate real challenges that are faced by machine manufacturers today. This is especially true as the complexity of these machines increases. One commonality between these two scenarios is the machine manufacturer's goal to be able to combine all diagnostic functions into a single diagnostic tool.

This tool should be able to handle diagnostic subfunctions related to a single subsystem regardless of the supplier, as well as the diagnostic subfunctions of the different subsystems present in the machine. The key to achieving such goal is the use of standardized communication data formats and interfaces. These include D-PDU API, ODX, and, OTX.

The Diagnostic Protocol Data Unit application programing interface (D-PDU API) is specified in the standard ISO 22900 and it is a modular vehicle communication interface (MVCI) protocol module. This standard describes a generic software interface which provides a "plug-and-play" functionality for different communication protocols. The interface connects the diagnostic information gathered through these different protocols to a standard D-server, the MCD-3 D server. This server then uses information regarding the data descriptions found in the ODX files and provides interpreted, symbolic information to the diagnostic application. The purpose of this interface, the D-PDU API, is to ensure that diagnostic and reprogramming applications from any vehicle or subsystem manufacturer can operate on a common software interface [3].

As mentioned above, ODX files contain information regarding the interpretation of diagnostic data pertaining to the different ECU's. Developed by the Association for Standardization of Automation and Measuring Systems, the Open Diagnostic Data Exchange (ODX) specification is used to describe and exchange vehicle and ECU diagnostic information such as diagnostic trouble codes, identification data, input /output parameters, and communication parameters. ODX is a machine-readable data format based on Extensible Mark-up Language (XML), and is independent of specific vehicle diagnostic protocols such as KWP 2000, UDS, or SAE J1939. ODX is designed to describe the following: protocol specifications for diagnostic communication of ECUs, communication parameters for different protocols, data link layers and ECU software, ECU programming data (Flash), related vehicle interface descriptions (information about connectors and pinout), functional description of diagnostic capabilities of a network of ECUs, and ECU configuration data (variant coding) [1-2]. This data format is designed to provide a standard way to describe and communicate diagnostic data independently of manufacturer, testing hardware, and protocol software, making it ideal for communicating diagnostic functions in an agricultural machine which contain multiple subsystems.

In the best case scenario, OTX files are used as the final step, connecting the raw data to the diagnostic application. The Open Test Sequence Exchange (OTX) format, which is described in the ISO 13209 standard, is an XML based exchange format for diagnostic test sequences. The OTX format is used to describe in a formal way the diagnostic sequences which can be executed as part of a diagnostic session. The goal of OTX is to provide a standard way for manufacturers to describe and exchange these test sequences. The structure of these sequences is described in schemas that make it easy to follow, understand, and implement [4].

The use of these standards together forms a modular structure and mechanism to communicate diagnostic information from the lowest level all the way up to the diagnostic tool. This structure, illustrated in Figure 3, allows for the ease of integration of diagnostic information from different subsystems and different manufacturers. This structure forms the basis of the solution to the use cases described above.

The methodology involved is as follows. Information requested by the diagnostic tool is processed by the OTX layer where it is matched to a specific diagnostic sequence. This test sequence then passes through the MCD-3 D server where the appropriate information related to the data required is obtained from the ODX file.

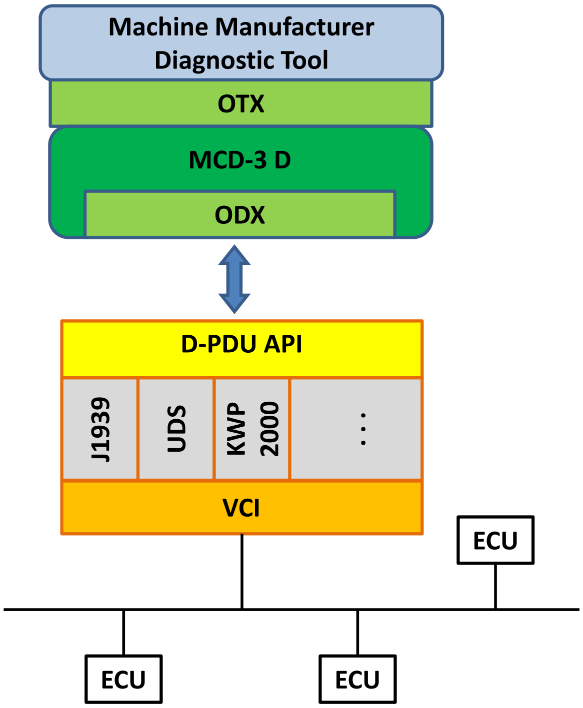

Figure 3: Structure for communication of diagnostic information using standards

Next, the transformed request passes down into the D-PDU API layer where, with the descriptions provided by the ODX file, the request is formatted such that it communicates to the correct ECU on the bus through the appropriate protocol. Similarly, when there is a response from the ECU, the information is passed up through the protocol stack and is collected in the D-PDU API. From here it goes up through the MCD-3 D server where the data is matched to their corresponding descriptions pulled from the ODX file. This information is then sent up to the diagnostic tool.

Applying this methodology to the use cases described above yields very similar models. In the first use case, the machine manufacturer wants to integrate the diagnostic functions of a specific type of subsystem which is obtained from different suppliers into one diagnostic tool. The manufacturer wants to use this tool to be able to handle the diagnostics of that subsystem regardless of which supplier it comes from. The solution for this use case is exemplified with the tractor engine as the subsystem, and is illustrated in Figure 4. In this model, the suppliers of the different engines used by the manufacturer (Suppliers A, B, and C) provide the corresponding OTX and ODX files for their engine ECUs. In these files, the diagnostic data and test sequences are described in a standard, machine readable format. The engine ECU ODX and OTX files are placed in libraries as part of the overall diagnostic software architecture of the machine manufacturer's diagnostic system. These files are then retrieved from the libraries when needed depending on which engine is being used. This way the manufacturer can have a single software architecture that can handle multiple variations of the same subsystem.

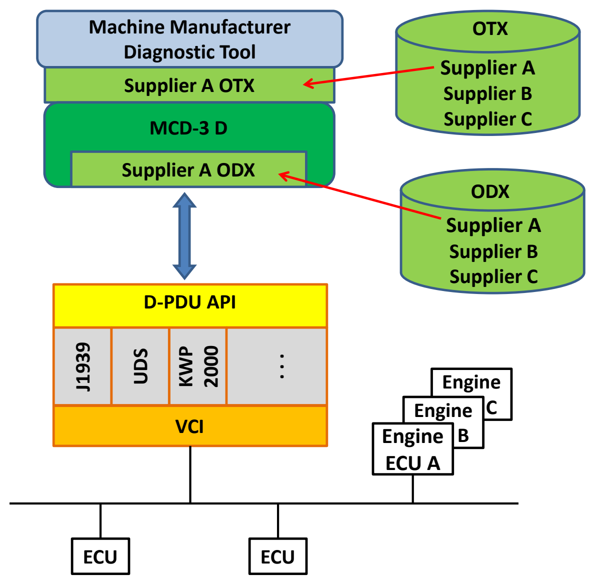

Figure 4: Schematic of best case scenario solution for use case 1

The solution for use case 2 is similar to that of use case 1. In the second use case, the machine manufacturer wants to integrate the diagnostic subfunctions that correspond to the different subsystems used in the machine into a single machine diagnostic tool. For example, the machine might include an engine, a joystick, and a steering column from different suppliers, all equipped with their respective ECUs. Similar to the first use case solution, the solution to this use case involves the suppliers of the different subsystems providing the corresponding ODX and OTX files. Figure 5 illustrates the architecture of this solution.

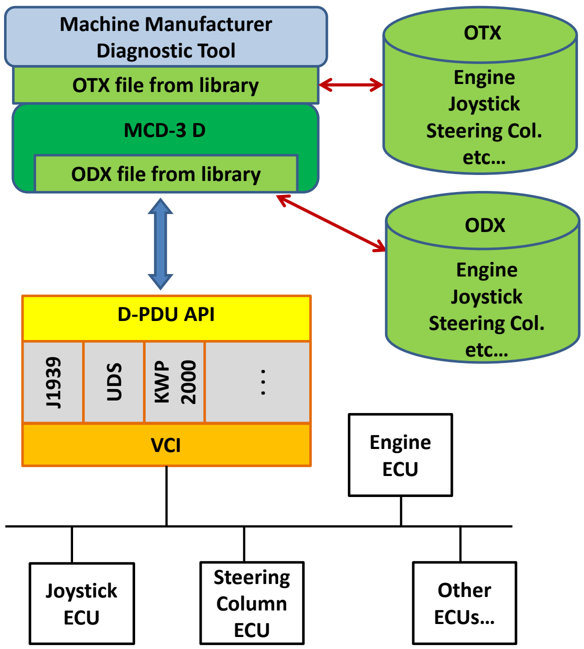

Figure 4: Schematic of best case scenario solution for use case 2

The ODX and OTX files are again stored into libraries that become part of the software architecture of the diagnostic tool. From these libraries, the ODX and OTX files associated with each ECU are retrieved when required. The use of these libraries in this use case builds on the concept described in the first solution in that these libraries not only contain ODX and OTX files related to different subsystems, they also contain the ODX and OTX files for the different variations of each subsystem. This way, the machine manufacturer is able to create one diagnostic tool that can be used for the entire machine. For example, using ODX and OTX files provided by the different suppliers, a manufacturer of agricultural tractors is able to create a single diagnostic tool which can be used for all of the different tractors he produces.

Real World Case

The solutions for the use cases described thus far are the best case scenario solutions. This is because they assume that for each subsystem, a corresponding ODX and OTX file is provided. In the 'real world' however, this is not always the case. It is very common for subsystem suppliers not to provide diagnostic information in these standard formats. The reasons for this depend on the supplier; several examples are detailed as follows.

Very often subsystem suppliers do not use these standardized formats because they do not want the machine manufacturer to have access to their proprietary knowledge and technology; they want their knowledge to stay hidden. They do this by using proprietary communication protocol .dll's which are not supported by the D-PDU application programming interface. Some suppliers do not provide ODX and OTX files because their diagnostic subfunctions use specialized communication formats. For example, the diagnostic subfunctions of the subsystem might communicate via a serial interface such as RS232 instead of CAN. Lastly, some suppliers fail to use standards such as ODX and OTX simply because they do not have enough time during the development process or because they have been using a specific way of describing data and they do not want to change the way things have always been done.

Regardless of the reason, when suppliers do not provide ODX and OTX files, the task of integrating the subsystems' diagnostics into a machine diagnostic tool becomes more difficult, although still possible. The solution in these cases involves constructing ODX files from the data descriptions provided by the subsystem supplier, and by making 'wrappers' or specialized interfaces which allow for proprietary or non-standard communication formats to interface with a standard protocol such as UDS. Hardcoded within these wrappers is information regarding diagnostic test routines which can be used with the built-in UDS function routine controls.

To explore this methodology, the solutions to both use cases are again examined. Use case 1 is modified in this scenario to encompass two engine ECUs (from suppliers B and C) which communicate using proprietary protocol .dll's. The engine ECU from supplier A in this scenario communicates through a serial interface such as RS232. For the sake of exploring a real world solution, none of the engine suppliers provide ODX or OTX files. The solution to this modified use case is illustrated in Figure 5.

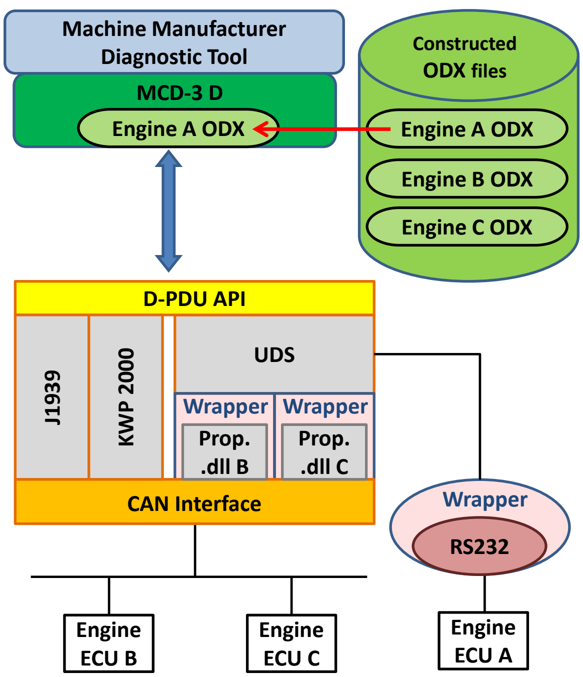

Figure 5: Schematic of real world solution for use case 1

The architecture of the solution for use case 1 includes ODX files which are created by the machine manufacturer from the description data of the ECUs provided by suppliers A, B, and C. Wrappers are also created for each proprietary protocol .dll, as well as for the RS232 interface, which will allow for these non-standard formats to interface with a standard protocol – i.e. UDS.

Use case 2 is also modified to demonstrate how the solution architecture must be changed in the case that the suppliers of different subsystems do not provide ODX and OTX files. This solution is illustrated in Figure 6.

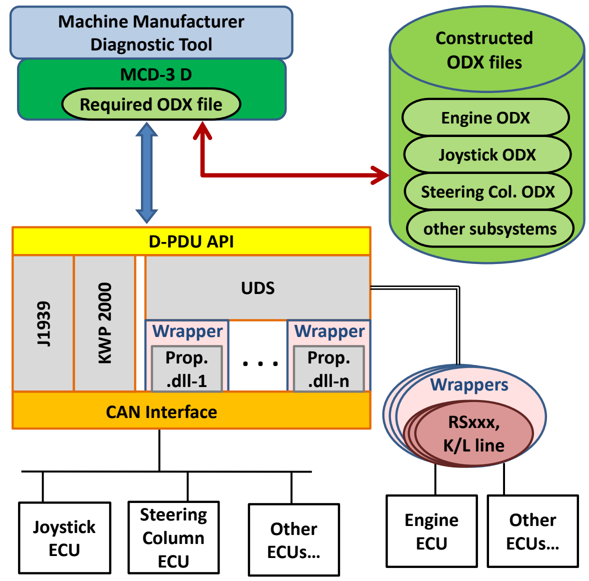

Figure 6: Schematic of real world solution for use case 2

In the modified use case 2, the machine manufacturer's diagnostic tool has to be able to integrate subfunctions from different subsystems provided by the same or different suppliers without having been provided with ODX or OTX files. Some of the ECUs might use proprietary protocol .dll's, others might use other forms of communication such as RS232. The solution for their integration is based on the same principals as in the solution for use case 1. ODX files are constructed from the information provided for each ECU and are stored in an ODX library. The MCD-3 D server then retrieves from this library the appropriate ODX file as required. Furthermore, wrappers which include hardcoded information regarding diagnostic test routines are created to interface proprietary protocol .dll's and other communication formats with the UDS standard protocol.

Conclusion

The complexity of agricultural machines has been and continues to increase allowing them to accomplish more tasks in a more efficient manner. Unfortunately, as the machine complexity increases, so does the complexity of the electronic control functions within the machine. This poses an ever growing challenge for service technicians to provide support for the machine throughout its lifecycle. One way to facilitate this task is for the machine manufacturer to create a diagnostic tool which is capable of carrying out all of the diagnostic subfunctions related to the different subsystems on the machine. This way the service technician can use one tool to diagnose any problem on the machine rather than having to use a variety of specialized tools for the many different types and/or suppliers of subsystems.

To this end, two different use cases were described which exemplify common integration problems that machine manufacturers are faced with. The solutions to these use cases demonstrate the underlying software architecture which allows for the integration of the diagnostic functions of all subsystems into a single diagnostic application. These solutions have been divided into two situations: the best case scenario, and the less favorable, real world scenario.

In the best case scenario, the suppliers of the subsystems provide the machine manufacturer information describing the diagnostic information and routines associated with their ECUs in formal, standardized ways – ODX and OTX. The use of these standards along with standardized communication interfaces allows for a “plug-and-play” architecture built for ease of integrating ECUs. Unfortunately, in the real world, the circumstance is not always the best case scenario. It is common for subsystem suppliers to fail to provide ODX and OTX files. Reasons for this include, among others, the supplier wanting to hide proprietary information, the subsystem diagnostic functions of the supplied ECUs use non-standard forms of communication, or the supplier does not have enough time/incentive to conform to these standards. In these situations, standardize communication interfaces are still used as part of the solution. ODX files and wrappers are created using the information provided by the suppliers in order to communicate via the standard communication interfaces.

These solutions illustrate that the real key to integrating different subfunctions into one diagnostic tool is to use standardized communication interfaces and methods of describing diagnostic information. Using standards allows for a modular architecture which is capable of being adapted to different situations. By having a modular schema, the machine manufacturer is able to incorporate subsystems from any supplier into one diagnostic application without having to completely rework his solution each time a new ECU is added. Not only is this system is much easier for the manufacturer to work with, it is also robust and future-proof.

Juan Aguilar
Sontheim Industrial Electronic, Inc.
One West Court Square, Suite 750, Decatur, Georgia, GA 30030, USA
+1 (678) 896-5446
+1 (770) 934-3384
juan.aguilar@s-i-e.de
http://www.sontheim-industrie-elektronik.de/en/

Gerd Bottenbruch
Sontheim Industrie Elektronik, GmbH
Georg-Krug-Straße 2
D-87437 Kempten, DE
+49-831-575 900 50
+49-831-575 900 72
E-mail: gerd.bottenbruch@s-i-e.de
Website: www.s-i-e.de

References

[1] http://www.asam.net/nc/de/home/standards/standard-detail.html?tx_rbwbmasamstandards_pi1%5BshowUid%5D=523&start=
[2] http://automotiveiq.wordpress.com/2011/06/03/automotive-diagnostic-systems-from-obd-to-open-diagnostics-exchange-format-odx/
[3] ISO 22900-2
[4] http://www.otxcentral.com/about-otx/