Validation of PERFoRM reference architecture demonstrating an automatic robot reconfiguration application

N.Chakravorti, E.Dimanidou
Manufacturing Technology Centre
Pilot Way, Coventry, CV7 JU
U.K.

G. Angione
Research for Innovation
AEA s.r.l. – Loccioni Group
Ancona, Italy
g.angione@loccioni.com

J. Wermann, F. Gosewehr
Faculty of Technology
University of Applied Sciences
Emden/Leer, Germany
jeffrey.wermann@hs-emden-leer.de,
frederik.gosewehr@hs-emden-leer.de

***Abstract*—The PERFoRM project aims to develop a common reference architecture for Agile Manufacturing Control systems for true plug-and-produce devices, robots and machines. The aim of the work described in this paper is to validate the concepts of the reference architecture in the context of the production of home appliances. A demonstrator has been designed and implemented to illustrate the automatic reconfiguration of the path of a robot equipped with a probe for the detection of microwave leaks coming from an oven.**

Keywords—architecture; adaptor; robots, middleware, demonstrator;

I. INTRODUCTION

Manufacturing as a sector is under increasing pressure to have the ability to produce more customised, cheaper and higher quality products. Furthermore, delays or shortages from suppliers or breakdown of production assets can have a high impact on the overall company performance. There are also needs for overall cost reduction.

These demands have motivated the introduction of new paradigms such as process integration, digitisation of production, access to production information, modularity and re-configurability into the manufacturing domain [1]. Additionally, technologies such as the Internet of Things (IoT), cloud computing and embedded systems are being utilised within the manufacturing domain, which are key components of the 4th industrial revolution [2]. In response to manufacturing industry demands, several national and internal strategic programs have been initiated e.g. "Industrie 4.0" in Germany and "Made in Sweden 2030" in Sweden. The European Commission has been encouraging research in these areas by launching the H2020 programme which funds work in areas such as the Factories of the Future.

The Production harmonized Reconfiguration of Flexible Robots and Machinery (PERFoRM) project is focussed on the requirements of increasing flexibility and configurability in manufacturing. The overall aim of the PERFoRM project is to develop a common reference architecture for Agile Manufacturing Control system for true plug-and-produce devices, robots and machines. The proposed system also integrates tools that enable scheduling, simulation and intelligent decision support [3]. This project is using the innovative results of the previous collaborative projects (such as SOCRADES [4] and IMC-AESOP [5]).

The validation of the PERFoRM architecture will be conducted by applying the architectural concepts to four diverse use cases, such as the: (1) production of compressors, (2) assembly of low cost full electric vehicles with high variants and high quality on low budget assembly lines, (3) production of aerospace components within high variants and (4) production of consumer white goods. The requirements of the use cases were collected via focussed interviews with the stakeholders. The requirements can be classified into *General Requirements*, covering aspects such as flexibility and re-configurability and *Other Requirements* including requirements such as traceability, automatic data acquisition of machine condition and integration of different departments and functions (e.g. scheduling system and maintenance, production and process planning etc.).

This paper will describe the architecture for the white good use case, the set-up of the demonstrator to validate the technology and present the results of the validation. The aim of this paper is to validate that the proposed architecture is able to satisfy the re-configurability requirement. The rest of the paper are organised as follows: the PERFoRM architecture and brief description of the architectural elements are presented in Section II, the user requirements and the corresponding architecture for one of the use cases is presented in Section III, the test cases for the evaluation of the re-configurability requirement are presented in Section IV, details about the demonstrator for the validation is presented in Section V and the results and conclusions are presented in Sections VI and VII respectively.

II. PERFORM ARCHITECTURE

The PERFoRM project is envisaged to introduce a next generation of agile manufacturing systems that are dynamically reconfigurable to enable self-organisation and adaptation along

the system life-cycle. These systems are targeted to be based on modular plug-and-produce components within the manufacturing system life cycle [6]. Prior work [6] has reported that the proposed architecture is composed of a diverse network of hardware and software assets, within the realm of different levels of the ISA-95 standard enterprise architecture. These assets are interconnected via the use of industrial middleware [7] and the assets expose their functionalities as services following the Service-Oriented Architecture (SOA) principles.

The overall system architecture of the PERFoRM project can be seen in Figure 1. Different components such as Programmable Logic Controllers (PLCs), Computer Numerical Control (CNC) machines, industrial robots and Human Machine Interfaces (HMIs) can be seen in Figure 1. Different Information Technology (IT) assets (see Figure 1) such as the Enterprise Resource Planning (ERP), Manufacturing Execution Systems (MES), System Control and Data Acquisition (SCADA) and databases (DB) also need to be integrated within any typical production environment. The ERP, MES, SCADA and DB are proprietary elements (seen in blue boxes within Figure 1) and will be used in the industrial use cases. The boxes in green colour (i.e. *Simulation* and *Data Analytics*) are being developed as part of this project. The validation of the *Simulation* and *Data Analytics* tools is beyond the scope of this project.

The different elements of this proposed architecture are presented in the following sub-sections.

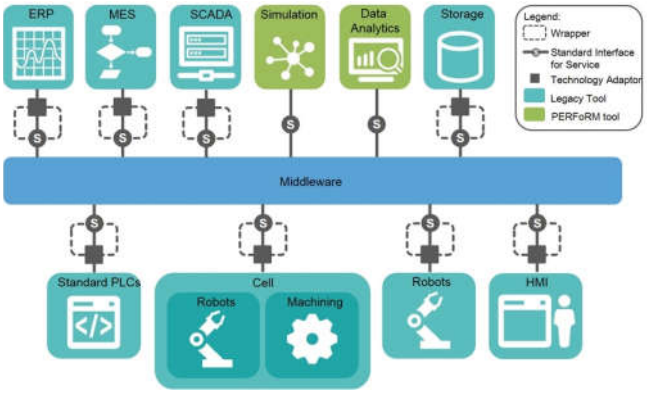

Figure 1. PERFoRM Architecture

A. Middleware

In order to enable each component within PERFoRM's architecture to recognise and communicate with each other, a common integration layer is necessary. This is achieved by implementing a Middleware as a platform, which resides in between two communication partners and provides a channel to send and receive data.

Within the PERFoRM project, a common data model is specified, which covers a semantic way of standardising data and interfaces, but does not restrict the user to a specific communication technology. The Middleware has to ensure that different protocols are supported and allows the transport of the standardised data. The protocols should use a Service-oriented approach, therefore protocols such as REST, SOAP or technologies like OPC-UA are especially of importance. A client application should not have to consider which exact protocol is used, as long as the data follows the standardised data model described before. The Middleware will be responsible for translating the protocol to a form that the server is capable of processing. Furthermore, the Middleware needs to provide a platform, where different components can register their services and which then can automatically be detected by other components within the architecture. The middleware should also support the configuration of how specific data should be handled and where the data needs to be sent. It also provides a way to transform data, if necessary.

Many different distributors already offer solutions for achieving these functionalities to varying extents. This is dependent on the domain application (e.g. for industrial systems or for business level integration) the supported features and protocols can be very different. Since it is necessary to find a common integration layer for the PERFoRM architecture, a Middleware approach which consists of two main components has been selected.

The first component is the Middleware core, which can be any existing Middleware solution. An assessment of potential solutions which can be used to implement the core functionality has been carried out in the first phase of the Middleware specification [7]. Many of these solutions already provide the most important features for a certain application, often with graphical user interfaces and easy configuration.

The second component is the Middleware shell, which is used to provide a common integration interfaces for all the core solutions. As different solutions are used, the core component sometimes misses important features or use their own non-PERFoRM-compliant method to handle the data, this additional layer is used to fill in the missing gaps and provide standardised interfaces to the outside. To achieve this, Apache ServiceMix is used as the main toolset to implement the necessary components for this shell. ServiceMix is an open-source container including various solutions from the Apache software family to build a SOA-based Enterprise Service Bus, such as message broker (Apache ActiveMQ), routing and integration patterns (Apache Camel) and Web service interfaces (Apache CXF). ServiceMix can run on its own as a Middleware solution, but lacks the industrial support and ease of use of many of the above mentioned commercial solutions. But since it provides all the necessary functionality, it is used within the demonstrator presented in this paper.

B. Standard Interfaces

The PERFoRM architecture uses Standard interfaces as the main drivers for plug-ability and interoperability and is used to enable the connection between the different system elements in a seamless and transparent manner. Through these interfaces each component is capable of fully describing and exposing its services in a standardised way, by means of a clear specification of the semantics and data flow involved in

these interactions. A data model has been developed to serve as a data exchange format to be adopted by the different components within the PERFoRM architecture.

C. Technology Adaptors

Interoperability is one of the key challenges whilst designing systems-of-systems, coping with the representation and seamless exchange of data originating from a wide array of entities, often from different assets. Technology adaptors are being used to convert legacy data (non-PERFoRM compliant) into the PERFoRM's data model and to implement the PERFoRM's standard interfaces. The PERFoRM middleware together with the proper adaptors permit the interconnection of heterogeneous legacy hardware devices, e.g. robots and the respective controllers, and software applications such as databases, SCADA applications and other management, analytics and logistics tools.

Adaptors for legacy systems such as databases (MySQL, Microsoft SQL Server and Oracle), PLCs (Siemens S7), robots (UR3, UR5 and UR10) and measurement sensors (Mitutoyo roughness tester SJ-210) have been developed as a part of the project. This paper focuses, in particular, on the deployment of robot adaptor within the test bed.

D. Analytics and Visualisation

The Data Analytics system uses various sources of machining and process data and involves processes such as data acquisition, pre-processing, analysis and interpretation. The data transportation must be ensured by the middleware such that relevant data from machine / ERP / MES level is available to the data analytics module (as shown in Figure 1). In general, data regarding the actual machining behavior (e.g. active power, reactive power, power factor, vibration) recorded by external sensor nodes, process data (e.g. actual process parameters, use of specific tools) and maintenance information, are required with a predefined data rate. The Middleware must satisfy these specific requirements for data transfer rates as well for latency needs.

E. Simulation Environment

Within the PERFoRM project, plant simulation tools (e.g. Technomatix PlantSimulation™ and AnyLogic™) can be used to simulate the material flow through the workstations regarding logistic aspects of production process like lead time, mean time to failure (MTTF), scheduling (could also be optimised with an evolutionary algorithm). These evaluations can be done offline for planning issues, like reconfiguration or maintenance, and online in parallel to operation (e.g. for KPIs) taking current conditions (e.g. change request in current production line, machine break down) into account.

III. USE CASE OBJECTIVE AND ARCHITECTURE

A leading European producer of consumer white goods (e.g. microwaves for this specific use case) has been selected to validate the PERFoRM architectural concepts within this paper. This company, produces medium sized, high quality components with an average lot sizes between 100 and 500. The company's motivation is to minimise change-over effort and allow fast integration of automation systems into the wider company Information and Communications Technology (ICT) infrastructure. Additionally, the visibility of real-time production system status for decision makers was also identified as a key driver. The overall objectives of this use case are: (1) monitoring of the Key Performance Indicators (KPI) to detect performance degradation and (2) fast reconfiguration of the path of a robot equipped with a probe for the detection of microwave leaks coming from a microwave oven.

The validation of the second objective has been selected for illustration within this paper. The use case aims to achieve an automatic reconfiguration of the path of a robot equipped with a probe for the detection of microwave leaks coming from the oven. The robot, Universal Robot (UR) will be used to perform an electromagnetic compliance (EMC) leakage test based on special markers on the CAD file of the microwave. The system consists of a UR10 robot, equipped with a microwave probe to enable the detection of microwave leakages. The second objective (as seen above) has been mapped to the overall PERFoRM architectural concepts as seen in Figure 2.

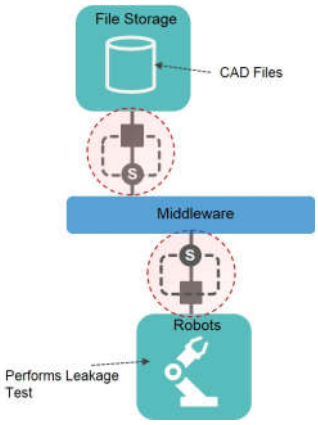

Figure 2. Leakage test architecture

The implementation of the robot adaptor considers the requirements coming from the leak robot station in the microwave oven manufacturing process. The robot moves the probe following a predefined path around the microwave oven. The CAD file contains the information for the probe path and is stored in a file repository. This CAD file is automatically transferred to the adaptor via the middleware. The markers in the CAD file are translated to a UR path file via the adaptor. Finally, the adaptor automatically transfers the UR path file to the UR10 robot for conducting the leakage testing.

If leakages are detected the oven is sent for repair. The role of the adaptor is to automatically generate the robot commands starting from a path drawn with a CAD tool and send them to the robot. This permits the product designers to directly draw the path which is more appropriate for detecting possible microwave leakages outside the oven. Moreover, it promotes no stoppages to the production line when a new model of oven is being produced and hence teaching the robot to follow the new path.

IV. TEST CASES

The test cases designed for validating the automatic robot reconfiguration system for the EMC leakage test can be seen in Table 1. Each test case involves a definition, pre-requisite, input and expected outcome. It is to be noted that the validation of the plug-and-produce mechanisms via the use of the standard interfaces is out of scope for this paper.

Table 1. Test Cases

Test Case ID	*Definition*	*Pre-requisite*	*Input*	*Expected Outcome*
TC-1	Verify that the UR script generated by the adaptor is correct	CAD file translated to UR Script by the adaptor	URScript	The robot should be able to act upon the command
TC-2	Verify that Adaptor can take a CAD file as an input, translate it to a UR Script and send it to the robot	The CAD file should be placed on the correct file location	CAD file	1. The adaptor should be able to acquire the CAD file and translate it to UR script. 2. The adaptor should be able to send the UR Script to the robot. 3. The robot should follow the correct path.
TC-3	Verify that the robot follows the correct path via interaction with the middleware	The middleware should be instantiated. The CAD file should be placed on the correct file location	CAD file	1. The middleware should be able to acquire the CAD file and send it to the adaptor for translation to UR script. 2. The adaptor should automatically send the script to the robot. 3. The robot should follow the path specified in the CAD file.

The first test case (see TC1 in Table 1) involves the validation of the output of the technology adaptor. The prerequisite for this test case is that the CAD file has already been translated to an URScript by the adaptor. This test in essence tests the validity of the URScript. For validating the script, each of the commands were independently sent to the robot. After each commands was executed by the robot, the test involved validation of whether the robot was following the defined path or was not acting as expected. After validating the commands individually, the whole script was sent to the robot through the adaptor in order to ensure that the controller is able to receive all the commands at the same time and execute the subsequent command after the current command was executed.

Test case TC2 (see Table 1) involves the validation of the adaptor via retrieving a CAD file from a file server. Whilst TC3 validates the vertical integration of the different elements of the PERFoRM framework (i.e. adaptors, middleware) within the context of the current use case.

V. DEMONSTRATOR SET-UP

The test items required for the validating test cases TC2 and TC3 are listed below:

- The Universal Robot for performing the microwave oven leakage test.
- The Adaptor, which acts as a meta-layer between the robot and the middleware. The adaptor is implementing the mechanism to access the path pattern file.
- The Middleware, which is used to transfer (text) files between above mentioned assets, i.e. the robot and the adaptor.

The first iteration of the test bed (referred to as demonstrator in the rest of the paper) has been set up at the Manufacturing Technology Centre (MTC), Coventry, U.K. including an UR10 robot (Machinery Level, as seen in Figure 3) and the PERFoRM middleware (Apache Service Mix) as seen in Figure 3. This demonstration is aligned to the Leak Test Scenario from the domestic appliances use case. The scenarios involve a file repository which stores CAD files, and the UR adaptor which translates a particular CAD file to a UR Script and transfers this script to the robot. For the purpose of the demonstrator, four virtual machines (VM) have been set-up to replicate the de-coupling between the individual systems. The VMs also allow a user to directly copy the VM image to individual host machines as well, and this can result in four different host PCs running the File Server, Operator Graphical User Interface (GUI), Robot Service and the Middleware respectively. The function of each of the VMs are listed below:

- File Server (VM): hosts the CAD files for performing the robot leak test
- Operator GUI VM: reads the list of CAD files from the file storage location (File Server, in Figure 3) and displays them such that an operator can select a particular file
- Robot Service VM: is used to pull a particular CAD file from the File Server and push it to a pre-determined folder that the UR adaptor can access
- Middleware VM: comprises of an Apache Service Mix

The sequence of operations for this demonstrator can be seen below:

- An operator selects a CAD file from the list displayed on the Operator GUI. This results in the Operator GUI service sending a trigger to the Robot Service, which then calls the FTP service requesting the file selected by the operator.
- The selected CAD file from the File Server VM is being downloaded, communicating via the middleware, to a folder that the Robot service can access
- Once the Robot service gets the file, it pushes this file to the Adaptor
- The Adaptor translates the CAD file to an URScript and pushes the script to the robot.

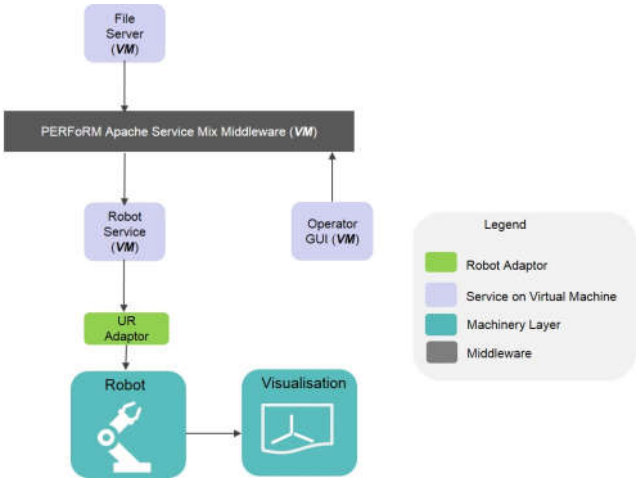

Figure 3. Demonstration of the Leakage test of microwaves using the PERFoRM compliant middleware (Apache Service Mix)

The linear movements and the rotation in x, y and z directions are recorded via using a test script. The actual and the planned movements are then visualised using a Matlab script.

VI. RESULTS

The validation of the various elements within the PERFoRM architecture was conducted incrementally. The first test case (TC1, see Table 1) involves the black box testing of the adaptor. The input for this test is the UR Script. For this test it is assumed that a CAD file has been translated to an URScript by the adaptor. This test was done manually by sending the commands in the script to the robot to test the validity of the script. The UR Script was validated to assess if the adaptor was behaving correctly.

The second test case (TC2, see Table 1) validates if the adaptor can take a CAD file as an input, generate a corresponding URScript and finally sent the URScript to the robot. The CAD file is manually placed in a pre-defined file location. It was observed that the robot was able to receive a script from the adaptor.

The third test case (TC3, see Table 1) is similar to the second test case, but includes the middleware. The actual flow of messages is illustrated by Figure 4. The file server and the robot adapter are connected to the Middleware, via a REST interface implemented with Apache CXF. Additionally, the Operator GUI is connected. All components offer various Web services, which are used to communicate between each other. An explanation of Figure 4 can be seen below:

- Step 1: The Operator GUI invokes the *updateRobotProgramTable*( ) service. This service is used to display the list of available robot programs and their status (e.g. "buildmicrowave" has a status indicating that it is running, as seen in Figure 4).
- Step 2: The Middleware uses the *getCurrExecProgID*( ) service provided by the Robot Service to get the current program, which is in execution.
- Step 3: The current executing program is available to the middleware.
- Step 4: The middleware uses the *getProgramVersion*( ) service provided by the File Server to retrieve the programs stored on the File Server.
- Step 5: The programs stored on the File Server are now available to the middleware.

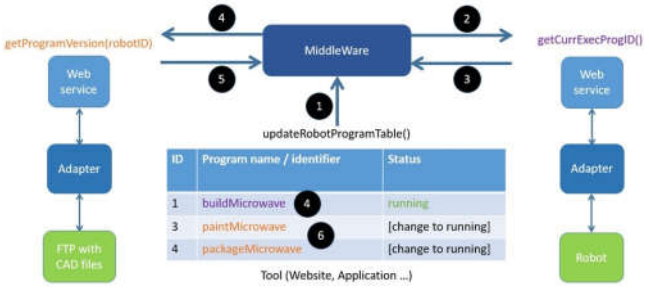

Figure 4. Flow of messages in the demonstrator

In a similar way, the GUI can be used, to then select a program from the list and send it to the robot adapter to load the selected robot program, using the *updateRobotProgram( )* service.

It is to be noted that this demonstrator can be extended to include further production assets. Using a setup of four different virtual machines, the files incoming from the host via C:\temp\ftp were successfully transmitted via FTP to the outgoing robot service, which writes the output file to C:\temp\robot. The adaptor was then able to consume the file to drive the UR10. This showcases the vertical integration going from the FTP via the middleware down to the robot service and the adapter, which finally transmits the new programming to the actual robot. A program was written to record the actual positions reached by the robot arm. A comparison of the path specified in the CAD file in comparison to the actual recorded positions can be seen Figure 5. The calculated error in the expected vs. reported positions were ±12 mm (see Figure 5). This calculated error is much higher than observed during the trials, a possible reason for the erroneously

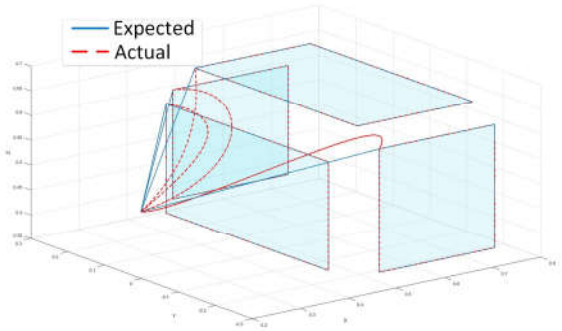

Figure 5. Comparison of path in CAD file with actual movements by robotic arm

large arises because the path comparison does not account for the robot's acceleration. In the comparison the robot was assumed to be travelling at a constant speed with zero acceleration between points. In practice the arm will accelerate to the commanded speeds and this the most likely reason for the large reported deviation by the algorithm. To get a more accurate figure the robot's acceleration profile will need to be known and accounted for in each move. Another approach is to ignore the temporal characteristics of the robot and calculate a spatial-only error.

Further work needs to be conducted to determine if this error can be minimised. In order to eliminate the danger of the robot crashing into the oven whilst moving from one point to the next, the robot returns to a pre-defined position every time the robot scans a side of the oven. This is visualised in Figure 5 at the far left corner of the plot.

VII. CONCLUSIONS

This work has implemented a system that enables the automatic reconfiguration of the path of a robot equipped with a probe for the detection of microwave leaks coming from the oven. The paper presents details regarding a demonstrator to validate the automatic reconfiguration of the robot. The demonstrator includes a technology adaptor for translating a CAD file to a script for a Universal Robot and a middleware for providing a common communication platform and translation of different communication protocols. Further work needs to be conducted to integrate the standard interfaces for supporting the pluggability of diverse production assets. Future work also needs to be conducted to the inclusion of additional assets, the validation of the architectural elements and the overall objectives of the PERFoRM project in the context of all the use cases.

ACKNOWLEDGMENT

The PERFoRM project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 680435.

REFERENCES

[1] H. Bauer, C. Baur, G. Camplone, and et. al., "Industry 4.0: How to navigate digitization of the manufacturing sector," tech. rep., McKinsey Digital, 2015.

[2] S.Wang, J.Wan, D.Li and et.al., "Implementing Smart Factory of Industrie 4.0: An Outlook", International Journal of Distributed Sensor Networks, vol. 2016, Article ID 3159805, January 2016.

[3] P.Leitao, J.Barbosa, A.Pereira, and et. al., "Specification of the PERFoRM Architecture for Seamless Production system reconfiguration" Industrial Electronics Society, 42nd Annual Confeernce of the IEEE IECON, October 2016.

[4] A. Colombo and S. Karnouskos, "Towards the Factory of the Future: a Service-oriented Cross-layer Infrastructure," ICT Shaping the World: a Scientific View, European Telecommunications Standards Institute (ETSI), pp. 65–81, 2009.

[5] A. W. Colombo, T. Bangemann, and S. Karnouskos, "IMC-AESOP Outcomes: Paving the way to Collaborative Manufacturing Systems," Proceedings of the 12th IEEE International Conference on Industrial Informatics (INDIN'14), pp. 255–260, 2014.

[6] P. Leitão, J. Barbosa, M. Foehr and et. al. "Instantiating the PERFoRM System Architecture for Industrial Case Studies". Service Orientation in Holonic and Multi-Agent Manufacturing. Studies in Computational Intelligence, vol. 694. Springer, Cham, 2017.

[7] F. Gosewehr, J. Wermann and A. Colombo. "Assessment of indurtrial middleware technologies for the PERFoRM project". Proceedings of the 42nd Annual Conference of Industrial Electronics IECON, October 2016.