Available online at www.sciencedirect.com

ScienceDirect

Procedia Technology 15 (2014) 621 – 630

Procedia Technology

2nd International Conference on System-Integrated Intelligence: Challenges for Product and Production Engineering

Evaluating a data distribution service system for dynamic manufacturing environments: a case study

M.S. Essers[a]*, T.H.J. Vaneker[a]

[a]University of Twente, Enschede, Laboratory for Design, Production and Management, Department of Engineering Technology, Drienerlolaan 5 7522 NB Enschede, The Netherlands

Abstract

Small and Medium sized Enterprises (SMEs) in Europe struggle to incorporate industrial robots in their production environments, while large enterprises use these robots for large batch production only. The paradigm shift from mass production to mass personalization decreases batch sizes and changes the approach to implementation of industrial robots in manufacturing environments. It also opens doors for SMEs to further incorporate robots in their production environments. The goal of this research is to evaluate the suitability of a data-centric, distributed, decentralized manufacturing system for cooperation between robots and humans. A case is presented featuring cooperation between robots and humans. A control system is proposed based on distributed intelligence and decentralized control, to handle the rapidly expanding complexity in dynamic manufacturing environments. The communication in such a distributed environment is provided by a Data Distribution Service system; an extendible, flexible approach to communication. Key issues that are encountered in implementing the cooperation into the current industrial environments are identified. The proposed control system is projected on the case and evaluated for application suitability and expected performance.

Keywords: Data Distribution Service (DDS); human robot cooperation; dynamic manufacturing

* Corresponding author. Tel.: +31 53 489 3192
E-mail address: m.s.essers@utwente.nl

doi:10.1016/j.protcy.2014.09.023

1. Introduction

One of the main trends present in both consumer products and manufacturing, is the transition from mass-production to personalization. [1] Production volumes are decreasing and the need for production lines that facilitate low production volumes and fast change-overs emerges. The demand for customized and personalized products poses issues when the targeted goods are traditionally mass-produced. Customizing and personalizing products is generally more expensive; the current gap between mass-produced and mass-customized products can be narrowed by manufacturing environments that are capable of producing products with high variety in small to medium batches. The adaptability of a production line to a high variety of products within a similar product group, is largely dependent on reconfigurability, flexibility in selecting tasks, and flexibility in accepting and performing new tasks. Flexibility is often considered a characteristic of industrial robots in manufacturing context, though industrial robots are only feasible production machinery when batches exceed the small to medium sizes because of their long and tedious work preparation process. In terms of mass personalization, industrial robot control and programming is too rigid for fast, fluent changeovers. Their inter-brand compatibility is low, and their feasibility in manufacturing environments requiring low tolerance manufacturing is very narrow, yet they show the most potential for their large work envelopes, low investment costs, fast work pace, and versatile deployment.

To streamline work preparation processes of small to medium sized production batches, manufacturing systems need to be reconfigurable and flexible. One of the key characteristics of such a high degree of reconfigurability, is the implementability of new and altered manufacturing entities, directly affecting extendibility of manufacturing systems. Physically, the posing problems can be addressed with current industry practices. The real problem in these highly configurable systems is communication with external sub-systems that e.g. sense and distribute information. Reconfigurations of systems require reconfiguration of external sub-systems, or alternatively, communication lines need to be rewired to suit the new situation. A traditional top-down, hierarchic description of system integration is too complex for a dynamically reconfigurable manufacturing system; a more heterarchical perspective towards distributed, decentralized control systems is required to deal with the exponentially increasing complexity in communication. [2-4] This approach complements extendibility of the manufacturing environment, allows for a variety of manufacturing equipment, enables multiple manufacturing environments to connect and communicate, while maintaining much of the original software and hardware.

The data-centric approach is a popular research topic about communication between systems. This approach allows for a complex system in which fault-tolerant communication can be guaranteed. Data Distribution Service (DDS) software is a data-centric publish/subscribe communication protocol between system participants, with automatic participant discovery modes and different settings for the quality of service. In [5], a system proposal is explained as envisioned for project SInBot (Smart Industrial Robotics). A DDS communication layer is used to allow manufacturing entities to communicate in an open and flexible environment. The most prominent advantage of using a DDS-based manufacturing system is the possibility of communicating with any entity that has been configured to do so. Extending or constricting the amount of entities in any group does not affect the system, nor any other individual entity. The entities are the system, adding or removing entities merely enlarges or shrinks the system. A major downside of this approach is the necessity for self-awareness (i.e. has information about itself) of manufacturing entities. While this in itself is very feasible, it requires some form of intelligence projected on each entity. For example, a sensing entity needs its global location (to be a truly self-containing sensing entity) for operational entities to request or process the sensing information.

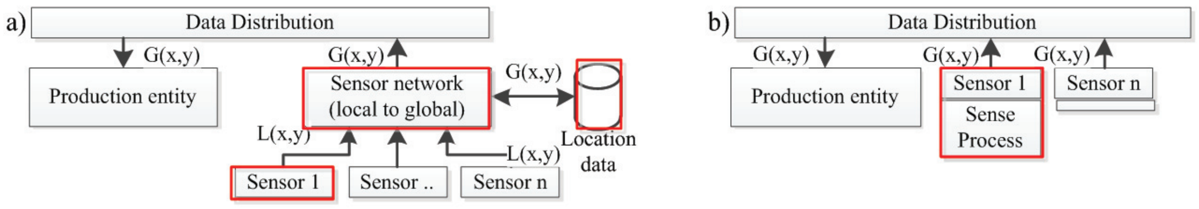

Fig. 1. (a) The difference between a hierarchical system and (b) a heterarchical system.

The same is true for operational entities; to enable a flexible task control architecture, they need to be aware of what it is that they are doing, where they are doing it, and on what they are performing the operational tasks. An example is given for a sensor relocation in Figure 1; in the traditional approach (Figure 1a), a relocation event of a manufacturing entity triggers a modification in a central database containing global coordinate information of the sensor, as well as adapting the system that converts local data (noted as *L(x,y)*) to global data (noted as *G(x,y)*). The heterarchical approach of Figure 1b is no longer dependent on a central database nor is it dependent on central processing. Instead, a generic module senses and processes information itself, directly publishing global data. When the sensor is relocated, there are no changes to any (sub-)system required. The same principle is true for other participants, e.g. production entities. The applicability of a heterarchical structured system is being tested in project SInBot [6]. SInBot is researching methods to improve the efficiency of industrial robots in performing small to medium-sized production runs. Two major research lines are explored within this project; unmanned production and human-robot collaboration production. The remainder of this paper addresses the applicability and expected benefits of the introduction of a DDS system in cases where efficient production demands close cooperation between man and machine.

2. Context

Currently, the problems surrounding robot-robot and human-robot collaboration are a major research area. The aforementioned paradigm shift from mass-production to mass-personalization is one of the key drivers. Another key driver is the reduction of investment costs, in which a switch from large equipment to robots is one aspect. E.g. large gantries often span more than a single work envelope of industrial robots, creating the need for multiple robots per production step. Introducing multiple robots per production step can yield related interference issues between equipment and processes. The most promising generalized solution is to create an environment in which the robots interact and cooperate, inherently solving the interference problem. Important to note, is the distinction between two seemingly cooperating industrial robots and true dynamic interaction between two cooperating industrial robots. Cooperation can be hard-coded through teaching, optimization iterations and manual adjustments to interference. Safety measures in these cases are mostly limited to non-admittance in active operating environments. True dynamic interaction between two cooperating robots can occur in large production volumes, but is most beneficial in small to medium sized productions runs; mass-customization and mass-personalization. In [7], four distinct paradigms regarding industrial robots were presented to aid definitions within manufacturing environments:

1. Two separate robots are working on different products, at different locations (see Figure 1a). There is no interaction nor interference between the two robots.
2. Two robots are working on the same product (see Figure 2b). There is interference but no interaction.
3. Two robots are working cooperatively on the same product, while the robots are online reconfigurable (see Figure 2c). There is both interaction and interference.
4. Two mobile robots are working cooperatively on the same product with a human worker (see Figure 2d). There is both interaction and interference from both robots and human workers.

The first paradigms is the current industry standard and, although time consuming in (re)programming, poses no additional issues to the already complex process of programming and reconfiguration. The second paradigm can be found mostly in assembly lines and already shows some difficulties in programming due to (predictable) interference. The third and fourth paradigms pose interference and interaction problems, but have great potential in the manufacturing industries. These paradigms are the focus of many research studies to improve the efficiency of deploying industrial robots [8].

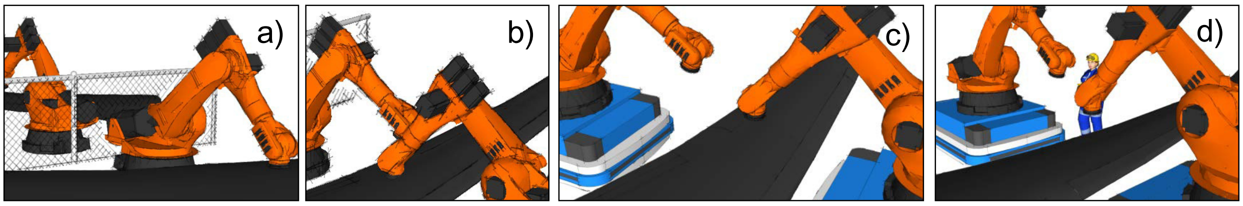

Fig. 2. (a-d) Industrial robot production planning paradigms.

The case that is used in this research focusses on paradigm four; introducing a human worker. Focusing on paradigm 4 in the case of small to medium sized production runs, four interesting generic issues related to interference and changeovers can been identified. For each issue, a solution is proposed in line with the SInBot system forming the case proposal.

2.1.1. Issue 1: Detection of unpredictable interference

Current unpredictable interference in the industry can be caused by unexpected behavior of human workers, faulty hardware, and malfunctioning software. Free-roaming (or path-guided) Automated Guided Vehicles (AGV) detect obstacles in their path, which is an example of interference detection. Their safety is based on unidirectional optical sensing [9-11]. Using similar technology based on solely internal sensors for human- (industrial) robot collaboration, poses an efficiency and costs problem; the safety sensors should monitor all the robot's joints, as well as the proximity environment of the entire arm. The monitoring system would require a reconfiguration on each changeover, making the already complex changeovers even more time-consuming. The costs of such a safety system is high due to the redundant sensors, and the efficiency is very low. A more efficient solution can be found in monitoring the environment, instead of the robot in question. Multiple objects may use the same sensors, and the focus of the safety system shifts from a single entity towards the production environment. A distributed sensor network based among others on bioptical vision, is used to monitor safety of robots, moving objects, and human workers.

2.1.2. Issue 2: Predictable interference and inter-system interaction

The current practice in the industry is to hardcode inter-system interaction between production machines. Path planning of industrial robots, for example, is hardcoded individually on both robots, often sharing a single programmable logic controller (PLC). This divisionary and centralized approach is rigid and required both robots to be reprogrammed on a changeover. The use of a single PLC also limits the redeployment capabilities of both robots, further limiting the dynamic possibilities of industrial robots. Programming interactions is a long tedious task; the programming of a single robot to do its task adequately may take up to 360 times the targeted cycle time [12]. The second industrial robot needs to be programmed to not interfere with the first, while performing its task as required. The programming is an already comprehensive task if the tasks of the robots are not of a dependent nature. If the two tasks are simultaneous *and* dependent, a programming and optimization circle starts in which each minor change in one of the robots will affect the task performance of the other. The latter statement is the biggest problem, and the most difficult to achieve in a production line. The main advantage of using hardcoded interaction and aforementioned physical restraints, is the consistency in production and some quality assurance during large production runs, (>50.000 cycles).

The problem that is addressed in this paragraph, is the interaction coding problem. First, dynamic safety zones are introduced to counter the hardcoded interaction. Each industrial robot can publish a zone in which it will work the next (e.g.) 5 seconds. Its path is not hardcoded, but its task partly is. If a task is unachievable (e.g. its path is blocked by a coworker's safety zone), it will attempt to perform another task first. This particular solution is not new [13, 14]. The combination with a DDS system allows for the creation of a specific domains to decrease network and processing load. The advanced dynamic extendibility enables safety zones publishing of products, machinery, infrastructure, and even human workers.

2.1.3. Issue 3: Fast change-overs and reconfigurations of manufacturing lines

As aforementioned, current industry practice is to use industrial robots rather statically. Some robots are situated on tracks, yet by far the most robots are mounted directly on a floor mount. Their flexibility (and potentially efficiency) can be increased by introducing dynamic reconfigurability, compared to the current scarce and manual reconfiguration standard. Both transporting and installing are manual processes, followed by time-consuming online calibrations. These manual processes can be replaced by deploying industrial robots on a transport-support structure, enabling AGVs to reconfigure the industrial robots. An important precondition is to ensure automated online calibration for industrial robots [15], AGVs capable of transporting industrial robots [16], and accurate positioning systems for fault-tolerant positioning of the robot transport-structures. Many examples of fault-tolerant positioning are already used in the industry [17, 18], most of these can be made more accurate by simple, fault-preventive (poka yoke [19]) positioning solutions.

2.1.4. Issue 4: Fast, flexible, redeployable, and brand-independent programming of cooperating robots

As mentioned in issue 2.1.3, the current approach to programming industrial robots is individual, centralized and therefore time-consuming and rigid. Every changeover requires an optimization iteration process, if the robot-location-product combination is new, also offline and online programming. Manufacturing tasks can be programmed in G-Code, which is the most common numerical control (NC) programming language. One of the problems with G-Code is its perspective, which is machine-oriented. When the raw-material / product is leading, true task flexibility is only reached through task definition from the product's perspective; product-oriented. An intermediate yet feasible approach that would be suitably generic is the use of intuitive commands; drill a hole, mill a protrusion, or grab an object. These commands can be seen as groups containing G-Code with dynamically adjustable coordinates and process settings based on the parameters of (e.g.) 'drill a hole'. Currently, each robot needs a program for every robot-location-product-task combination. A generic task description would exclude robot, location and product from the equation, leaving a single program per task.

3. Approach

The main issue with a flexible manufacturing environment that allows human-robot interference and cooperation is that a traditional top-down description of system integration is too complex for a dynamically reconfigurable manufacturing system,. Recent developments steer towards distributed, decentralized control systems in which speed and reliability are gaining momentum to assist in safety systems [20]. The main advantage of the distributed, decentralized approach is the reduction of single points of failure, potentially greater efficiency, flexibility and quality of manufactured goods. The decentralized aspects allows systems to break free from their hierarchical structure by assigning tasks to specific production entities in some form of deliberation. The extendibility of the system is the main advantage of using specifically a DDS system to communicate the bulk of necessary data across multiple production entities. Each participant can join a communication domain if necessary, in both a publishing and subscribing role. Each entity can supplement the information regarding a specific topic thus creating a relatively fault-tolerant architecture (Figure 3).

When looking at the static complexity of a system as defined in [21], there are three main aspects that describe said complexity: the structure of the system, the variety of subsystems, and the strengths of interactions. Each subsystem in a manufacturing system is a system itself, comprised of components or other subsystems. Therefore, the complexity of as system is at least comprised of the total complexity of individual entities. Complexity in this paper is targeting static complexity assuming that dynamic complexity adds uncertainties of events and behavior, and both subsystem-variety and degree of interaction are captured in interface design and interface abundance. The main issue with decoupled system is the required added awareness of entities, or artificial intelligence and environment sensing. In the traditional approaches, the total static complexity of a system ($\sum c$), approaches the complexity of a single entity (c_{s1}), to the power of the number of manufacturing or sensing entities (n). In another perspective; the entire system is affected and requires modifications when the system is extended with a single production entity.

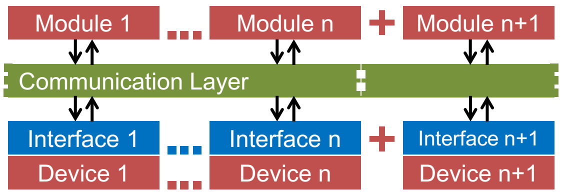

Fig. 3. Abstract view of the proposed system architecture.

$$\sum_{i=0}^{n} c \sim c_{s1}^{n} \tag{1}$$

The system does not require modifications after adding or removing entities in the distributed, decentralized approach. The only affected entities in a system extension are those that the system is extended with. The complexity of a system ($\sum$ c) thus has a linear relation to the complexity of a single entity (c_{s2}) and the number of entities (n), with $c_{s1} < c_{s2}$.

$$\sum_{i=0}^{n} c \sim c_{s2} \cdot n \tag{2}$$

If the complexity increases exponentially (traditional approach, equation 1), manufacturing systems reach their extendibility limit relatively fast. The distributed, decentralized approach has a much higher extendibility limit, enabling the system to combine data from many different manufacturing, sensing, and processing entities (equation 2). The method to propose a solution for the case, is to determine which entities are connected (interface in hardware and software) and are to be altered or modified after a reconfiguration of a manufacturing environment. The complexity of these interfaces define the need for distributed intelligence, while the type of connected entities determine the level of decentralized control. The solution should encompass a control system that is easy to implement, easy to extend, easy to maintain, and can handle growing complexity.

4. Results

4.1. Application requirements and interface specification

The general application of DDS systems, in particular OpenDDS, contains a few characteristics that determine the complexity of implementation, and performance during use. The most prominent issue is incompatibility of software, e.g. incompatible programming languages or protected software environments. A programming language that is not natively supported by the DDS entities can be shaped into an Application Programming Interface (API) wrapper, i.e. 'wrapping' C++ functions in a .NET language to allow the DDS API to be controlled through the wrapper [22]. Protected software and hardware require client/server bus-type communication, since manufacturing hardware and software often offer communication sockets / busses. An integrated approach in implementing the DDS API in manufacturing entities yields the smallest message latencies. Navigating the messages through e.g. a socket or communication bus is the least reliable and increases message latencies. The DDS-standard message structure 'Interactive Data Language (IDL)' cannot be maintained in these cases and is swapped for a restructuring process native to the target software. For example, an IDL can be translated to a 'Structure' in the .NET framework. These issues are also encountered in the presented case. The required level of DDS integration can be acquired based upon the need for fast message transport. Requirements for fast message transport can be either safety or cycle time related. From [7], interfaces can be designed upon these requirements. Fast, high risk, mostly safety related processes require an direct implementation in the targeted software. Moderately demanding processes, mostly cycle time related, require a fast yet flexible approach for which an indirect implementation with parallel processing of data is ideal. The last option has no direct influence on schedule based processes and can be indirectly implemented with simple serial processing of data. Case solution proposal

The cooperating robot – case is a puzzle the industry is trying to solve for many years. However, before the industrial robot can be truly flexible, it should adhere to Offline Programming (OLP) to a larger extend than it currently does. Acquiring an accuracy near repeatability specifications is one of the preconditions of a dynamically reconfigurable system, and is still a major research topic [23]. Self-awareness of production entities becomes increasingly important when dealing with unpredictable interference issues. Safety measures are only reliable when the potential dangerous entity has reliable information regarding its location, position and information regarding the nature, location, and position of its work space invader. Optic camera systems that capture both image and depth (e.g. Kinect2, or similar industry stereovision camera) can be used to dynamically define the work space [24]. The system can be divided into several active participants regarding the need for communication. The case is specified as containing two industrial robots, sensors and actuators, defined in Table 1.

Table 1. Hardware overview: A (incomplete) list of hardware with which the case study is performed.

Sub-system	Type	Focus	Implementation	Amount	Interface
Production cell	Sub-system coll.	Factory	External	1... n	Complex
Safety sensor	Optic	Work space	External	4... n	
Safety sensor	Force	Robot	Internal	2... n	
Safety actuators	Active, electric	Work space	External	4…n	
Manufacturing sensor	Optic	End-effector	Internal	2	
Manufacturing actuator	Industrial robot	Part	External	2… n	Complex
Manufacturing actuator	Tool rack	Robot	External	2… n	Complex
General actuator	AGV	Factory floor	External	1… n	Complex

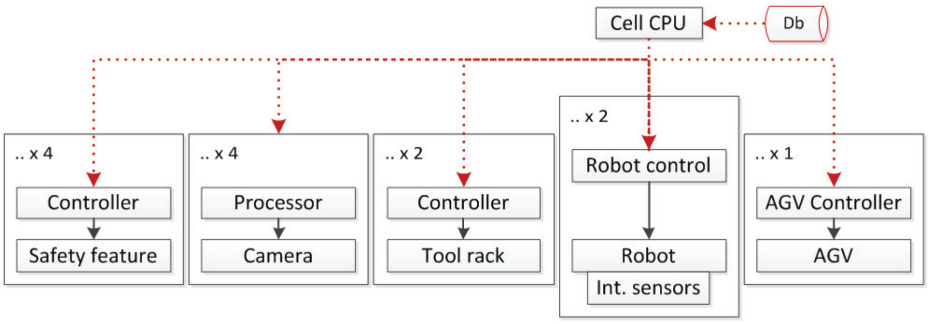

Fig. 4. Current situation architecture sketch of case control system.

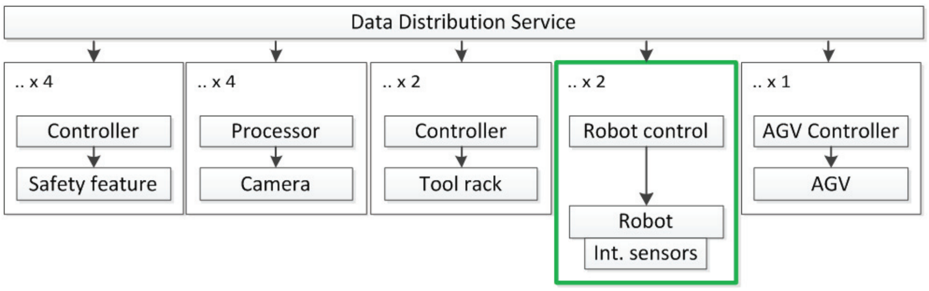

Fig. 5. New situation architecture sketch of proposed control system.

At the time of writing, there were no records of online or dynamic reconfigurable industrial robot implementations in the industry. The current situation would deal with this paradigm hierarchically (sketched in Figure 4). The red dotted lines indicate interfaces (both hardware and software) that would inherently alter on reconfigurations. The information is gathered in the robot controllers, resulting in approximately 13 interfaces that would need to be rebuild on reconfiguration in this simplified architectural view. The new situation changes a lot in the area of interfaces and running services on production entities. The amount of interfaces remains 13, without the need for manual reprogramming of interfaces on reconfigurations. Instead of interfacing different types of hardware and software, the interface is target is always the DDS system; simpler and less prone to human error. The required intelligence increases to a point that all data should be interpreted within each individual entity, while the more sophisticated hardware (robots, AGVs) require some form of preprogrammed strategies for performing tasks and safety maneuvers. A more detailed view of the industrial robot (highlighted in bold green in Figure 5) is given in Figure 6. A total of 9 (DDS) connections are defined in this simplified detailed view. The inherently dynamic DDS system gives the robot the opportunity to switch from one (e.g. sensor) interface to another. There is no need to manually redefine interfaces between components on reconfigurations; the robot can use provided global location information to re-establish a connection with relevant data publishers (i.e. subscribe to information relevant entities). The DDS participants can be integrated on different levels in the robots controller of Figure 6. Specified previously, interface design can be linked to risk factors; high risk, human safety related information should be implemented directly into controller software, while task related non-safety related information can be routed through a layered processing system. These decisions will dictate the effort required to link a new production entity to a distributed, decentralized system. Lastly, communication that does not have a relationship with cycle times nor safety can be interfaces at the highest abstraction level. The system distinguishes the latter two interfaces as parallel and serial indirect interfaces; parallel processing enables speed, serial processing has the lowest required effort/costs. The interface design of the example provided in Figure 6 is presented in Table 2. Recommendations on safety

The proposed system is by no means a complete view of a manufacturing system. It is recommended to introduce safety protocols throughout the different software levels of the system. For example, the lowest level will contain the sensing and actuation on a safety breach. Using redundant sensor systems and safety processing systems is one of the strongest aspects of a DDS based system. On a higher level, failing of sensor systems or absence of sensor information should be noticed, upon which a supervisory system should implement a safety strategy. Each level in the communication layer should be reliable, for example by performing a handshake to ensure data transition. DDS systems (e.g. OpenDDS) can use reliability settings in the transport types and Quality of Service settings to ensure message transport. Both publishing and subscribing entities should be able to spot failed connections to and from DDS participants. By integrating safety protocols in different levels, reliability of safety systems increase.

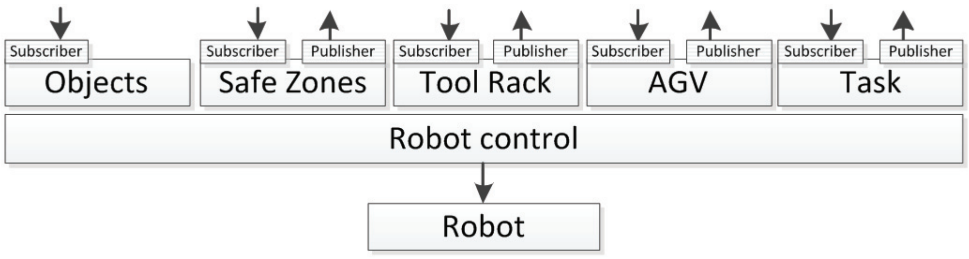

Fig. 6. Detailed view of the industrial robot

Table 2. Interface specification of industrial robot example

Interface / data processing	Direct / Serial	Indirect / Parallel	Indirect / Serial
Latency	< 1 ms	< 16 ms	< 70 ms
Process / Operation Freq.	750 Hz / 20 - 50 Hz	75 Hz / 0 – 10 Hz	35 Hz / 0 – 5 Hz
Reliability	Excellent	Good[†]	Good[b]
Interface of	Objects, Safety zones	Tool rack	Task, AGV

5. Conclusion

A case was presented that featured cooperating industrial robots, interfering and interacting human workers, focusing on small to medium batch size production. One of the main problems in applying traditional manufacturing systems to a production environment with rapidly recurring reconfigurations, is that the system becomes too complex. A general perspective on growing complexity in traditional manufacturing environments was proposed, yielding a complexity growth nearing a power function. From the simplified system architecture in the traditional approach emerged a quadratic growing interface-specific complexity, much in line with our expectations. In contrast, the proposed distributed, decentralized approach yields the expected linear increase in interface-specific complexity. Another great advantage is the fact that on initial install of a single entity all possible interface connections are created. There are no additional or modified interfaces required on reconfigurations caused by changeovers in production. The traditional approach is normally configured for a specific configuration, requiring incremental additional or modified interfaces on each reconfiguration.

Four distinct issues were presented that block the industrial application of the case in the traditional approach. The four issues were focused on interference detection, interaction of production entities, fast changeovers, and generic robot programming. The distributed approach enables the use of a distributed sensor network for detection of unpredictable interference (issue 1) and the sharing of restricted, constrained (and derivative safe-) zones to ensure interaction between production entities (issue 2). Using AGVs to automate the dynamical reconfiguration processes is supported by the system, but is by no means a directly implementable solution in the industry, providing a recommendation for issue 3. The last issue points towards the problems with combining production entities of different brands, types, or purposes. The proposed system supports and recommends brand-independent programming, enabling a much more efficient deployment of a distributed, decentralized system in the industry. Currently, G-Code can be used as a brand-independent medium, but only solves the interoperability of manufacturing machines. In upcoming research, the system as proposed is build and tested on laboratory scale. Manufacturing scenarios are investigated on the potential benefit they will obtain from a distributed, decentralized system to further define the parameters that determine the break-even point from which this approach is preferred.

References

[1] Hu SJ. Evolving Paradigms of Manufacturing: From Mass Production to Mass Customization and Personalization. Procedia CIRP 2013; 7(0): pp. 3-8

[2] Vrba P, Marik V. Capabilities of Dynamic Reconfiguration of Multiagent-Based Industrial Control Systems. IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans 2010; 40(2):213-223.

[3] Duffie NA, Piper RS. Non-hierarchical control of a flexible manufacturing cell. Robotics and Computer-Integrated Manufacturing 1987;3(2): 175-179.

[4] Duffie NA, Chitturi R, Mou J-I. Fault-tolerant heterarchical control of heterogeneous manufacturing system entities. Journal of Manufacturing Systems 1988; 7(4):315-328.

[5] Essers MS, Vaneker THJ. Developing Concepts for Improved Efficiency of Robot Work Preparation. Procedia CIRP 2013;7(0):515-520.

[6] Intereg, SInBot. [cited Access 10-11 - 2013]; Available from: http://www.smartbot.eu/project-information/sinbot/.

[†] Excellent, as long as operation frequency is lower than process frequency

[7] Essers, M.S. and T.H.J. Vaneker, Evaluating a prototype approach to validating a DDS-based system architecture for automated manufacturing environments. The 8th International Conference on Digital Enterprise Technology (DET2014) (in press). Stuttgart, Germany; 2014.
[8] Nilsson, K., et al., Productive robots and the SMErobot project. Book of Abstracts of Third Swedish Workshop on Autonomous Robotics. 2005.
[9] Ronzoni, D., et al., AGV global localization using indistinguishable artificial landmarks. Robotics and Automation (ICRA), 2011 IEEE International Conference on. 2011.
[10]Martinez-Barbera, H. and D. Herrero-Perez, Development of a flexible AGV for flexible manufacturing systems. Industrial robot: An international journal 2010;37(5): 459-468.
[11]SICK, Home. [cited Access 11-02 - 2013]; Available from: http://www.sick.com/nl/nl-nl/home/Pages/Homepage1.aspx.
[12]Johansson R et al. Sensor integration in task-level programming and industrial robotic task execution control. Industrial Robot 2004;31(3): 13.
[13]Yared R, Defago X, Wiesmann M. Collision prevention using group communication for asynchronous cooperative mobile robots. Advanced Information Networking and Applications, 2007. AINA '07. 21st International Conference on. 2007.
[14]Rodriguez A, Zhang C, Hammad A. Feasibility of location tracking of construction resources using UWB for better productivity and safety. The Conference on Computing in Civil and Building Engineering. 2010, Nottingham, United Kingdom. 2010.
[15]Nubiola, A. and I.A. Bonev, Absolute calibration of an ABB IRB 1600 robot using a laser tracker. Robotics and Computer-Integrated Manufacturing 2013;29(1):236-245.
[16]Kim J, Park J, Kim S. Inertial Navigation System for Omni-Directional Agv with Mecanum Wheel. Advances in Mechanical Engineering 2011;2.
[17]Wang Z, Liang M, Maropoulos PG. High accuracy mobile robot positioning using external large volume metrology instruments. International Journal of Computer Integrated Manufacturing 2011;24(5):484-492.
[18]Muller R. et al., Reconfigurable handling system. Production Engineering 2011. 5(4), pp. 453-461
[19]Dudek-Burlikowska M, Szewieczek D. The Poka-Yoke method as an improving quality tool of operations in the process. Journal of Achievements in Materials and Manufacturing Engineering 2009;36(1):95-102.
[20]Hancke GP, Gungor VC. Guest Editorial Special Section on Industrial Wireless Sensor Networks. IEEE Transactions on Industrial Informatics 2014;10(1):762-765.
[21]Deshmukh AV, Talavage JJ, Barash MM. Complexity in manufacturing systems, Part 1: Analysis of static complexity. IIE transactions 1998;30(7): 645-655.
[22]Calkins C. Code Generation with OpenDDS, Part II. 2010.
[23]de Graaf M et al. Real-time seam tracking for robotic laser welding using trajectory-based control. Control Engineering Practice 2010; 18(8): 944-953.
[24]Henry P. et al. RGB-D Mapping: Using Depth Cameras for Dense 3D Modeling of Indoor Environments, 33. O. Khatib, V. Kumar, and G. Sukhatme. 477-491; 2014.