ready-to-use self-contained functional units, the engineering workload for process plants can be considerably reduced [8]. The reuse of proven module solutions prevents the occurrence of faults in early engineering phases, which in turn leads to cost and time savings [9]. Besides the sought after scalability through the addition or removal of modules, a better data basis is provided for estimating the costs for the planning and construction of the plants, since only known outlays are repeated in a new combination [10].

Decentral Intelligence for Modular Assets (DIMA)

The DIMA concept (Decentralized Intelligence for Modular Applications) [21, 22] describes a method for solving a central challenge of modularization [8]. The latest process control systems (PCS) are not sufficiently prepared for implementing the engineering and flexible operation of modular plants as quickly and as cheaply as possible [12, 14]. DIMA enables the creation of adaptable modular production plants using currently available PCS functionality. This consequently reduces the need for new investments. The DIMA method was developed by Wago, the Dresden Technical University and the Helmut-Schmidt University, Hamburg.

The DIMA method and the results of its practical application are presented in the following.

Like the physical modularization of the plant [18] a function-based modularization is implemented from the process management perspective. A module in this case provides its process engineering function as a service to a higher-level process control system. It thus takes on the role of a service provider. The service offered by the module can be called by the process control system, which is thus a service user. The integration of several modules and their services into a complete system is called integration engineering. In order to keep the effort required to implement the service oriented architecture (SOA), a module must support at least the following basic functions of the process control system:

- HMI functionality: Generation and transfer of data to display and operate the operator screens in the PCS;
- Control and monitoring: Determination, monitoring and transmission of status information for fault-free processing of required services as required.

To provide these functions the following aspects and tasks must be implemented as part of the integration engineering:

- Network engineering − with the aim of mapping the physical communication system and enabling parameterization;
- Implementation of cross-module procedure control for on-time calling and monitoring (orchestration) of module services;
- HMI engineering for implementation of the “Operation” and “Monitoring” functions.

The engineering processes of the module manufacturer (module engineering) and the user (integration engineering) are kept separate.

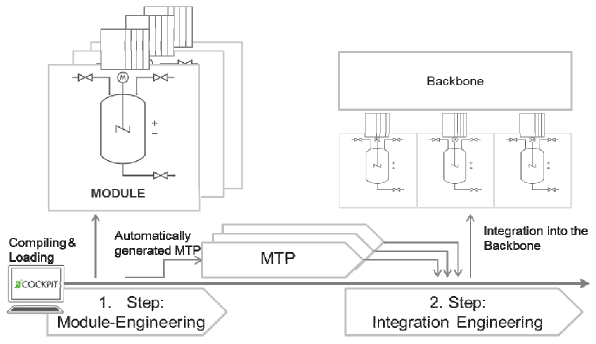

MODULE
Backbone
Compiling & Loading
Automatically generated MTP
MTP
Integration into the Backbone
COCKPIT
1. Step: Module-Engineering
2. Step: Integration Engineering

Figure 1: Engineering a Modular Plant Using Intelligent Modules

The manufacturer of the module carries out the module engineering. This includes the creation and loading of the executable software code of the module control (Figure 1, left) and the definition of the service interfaces. The module software is thus also protected from any later unwanted interventions.

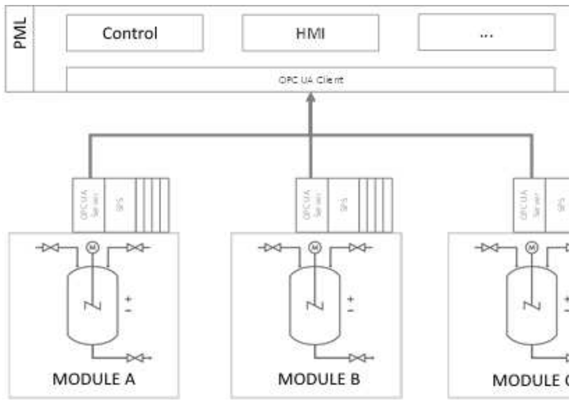


Figure 2: Integration of the Modules in the Process Management Level (PML) via OPC-UA

If several modules are combined into a plant and connected physically to a backbone (Figure 1, right), these modules have to be integrated in a higher-level process management level (PML) as part of the integration engineering (see Figure 2).

A process control system, for example, can be used in the PML. As demonstrated in [22], this PML can also consist of individual components (SCADA, Batch etc.) which each take on subfunctions of a PCS. As part of the further development of the DIMA method described here, the communication media has been revised compared to [22]. OPC-UA is used as the communication technology, by which the OPC-UA server of the module should be located directly on the controller of the module or in the module and created by the module manufacturer. This contains all the information required for the communication.

To organize the services of the individual modules in the sequence required for the production of the product (orchestration), the startup of the reactor in a continuously operated reaction process, as an example, must be adapted to the initial products. As this additional orchestration function is only required when different modules are combined, this function must be handled by an automation instance that can be designed in the integration engineering, e.g., the higher-level control system. The orchestration of the services of several modules requires knowledge of the current states such as Run, Stop or Fault, and corresponding status transitions of the modules and module services. This information is determined through the decentralized intelligence of each module and via the communication interface. The definition of the states must be manufacturer and module independent and thus be carried out uniformly for all modules. IEC 61512 [16] contains an example definition of states and defines the conditions for the possible status transitions for procedure elements (Figure 3). In the current project, this provides a good working basis for the status description and control of modules and services. However, it still has to be examined whether other states, status transitions and/or secondary conditions for status transitions are required for the control of modules and services.

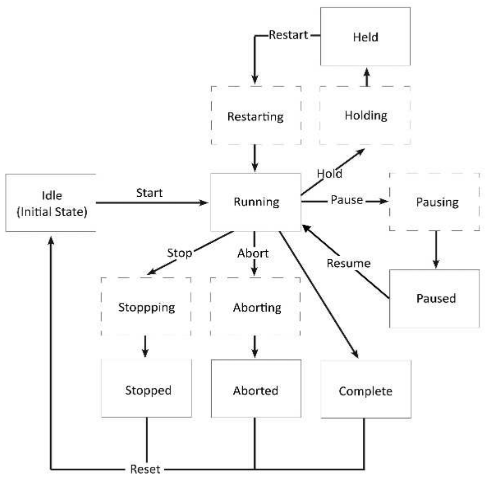


Figure 3: Current Working State of Service States and Transitions Based on IEC 61512 [16]

As already described, the modules of the intended process engineering functions are encapsulated as services. A reactor module with a mixer, for example, could offer the Mix service. As educts must be filled in the reactor, the Fill service is also offered by the reactor, which can be distinguished as FillA and FillB, depending on the number and name of the filling ports. If the reactor has a heating system, the Heating service is likewise executed. An appropriate parameter set of this service could be the target temperature, the temperature rate of increase and the holding time. A detailed view of the service interface and its mapping to the MTP as well as the integration workflow can be viewed in the publications [21] and [22].

The central challenge of the operability and view ability of the process distributed among several modules is both the automatic generation of operating screens as well as the implementation of the uniform “look and feel” of a modular plant as formulated in [13].

As module manufacturers are responsible for the planning, development and programming of the module, they also produce the operating screen(s) of the module. At this time, however, they do not have a knowledge of the project specific operating screen library of the higher-level system mostly used in industrial plant projects. The generation of the module-specific operating screen in the process control system of the overall plant can thus only be completed after the module is integrated in the integration engineering phase. Only at this time can the operating screen library be defined uniformly. In order to implement the module-specific operating screens in those with uniform project operating screen elements, these must be provided in a descriptive form that is independent of the display. This description contains the information on the role of the operating screen element, its position and size information, as well as any color information for dynamic display effects. The information can then be accessed via an algorithm, which places the project-dependent operations graphic elements into the desired form and position on the operating screen in the target system and links them with the corresponding variables to the module controller [19]. The use of a role-based library concept requires the library to be present both in the engineering tool of the module manufacturer and in the target system. An appropriate library is currently being created in NAMUR and ZVEI (see Section 4). A detailed view of this approach as well as the integration work flow and its mapping to the MTP can likewise be viewed in the publications [21] and [22]. All information that is worked out in the module engineering and is required during the integration engineering phase is stored in an “MTP” information carrier.

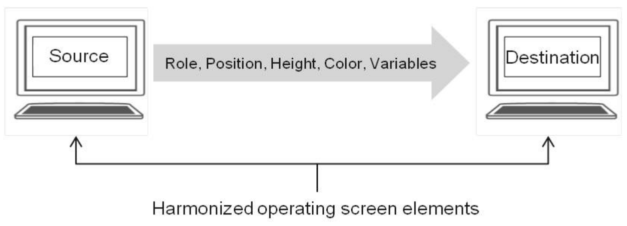

Source
Role, Position, Height, Color, Variables
Destination
Harmonized operating screen elements

Figure 4: Operating Screen Description and Use of an Operating Screen Library

The target system can then be configured accordingly using this information carrier. The following sections present the content of the MTP according to the current state of the discussions held in NAMUR and ZVEI.

The Way to Standardization

DIMA was presented for the first time at the main NAMUR General Assembly in 2014. The prototype implementation at that time used Siemens WinCC for the operating and visualization functions as well as the batch tool Proficy-Batch from General Electric for the orchestration of services. Already in this first presentation of DIMA, the management announced that the company would hand over the results of this prototype implementation to the NAMUR and ZVEI committees. In January 2015, NAMUR set up four ad-hoc working groups.

The content of the MTP was then divided up conceptually (see next section) and the resulting aspects were transferred to these ad-hoc working groups for development. In order to synchronize this and ensure the consistency of the results, NAMUR-AK 1.12 acted as a superordinate instance with AK 1.12.1 founded for this purpose (see Figure 5). As the development of the content had to be solution-based already at an early stage, the companies involved in the ZVEI working group for modular plants followed this structure. Since then, work has continued intensively in these working groups on the further development of the MTP. In all, more than 25 corporate representatives have been involved in drawing up a NAMUR recommendation for the content and structure of the MTP. A diagram of the current work structure is shown in figure 5.

The Way to Standardization

The working groups are each addressing the structure and the content of one aspect of the MTP. The aspects here are derived from the so-called "Tauchnitz PCS pie" [23].

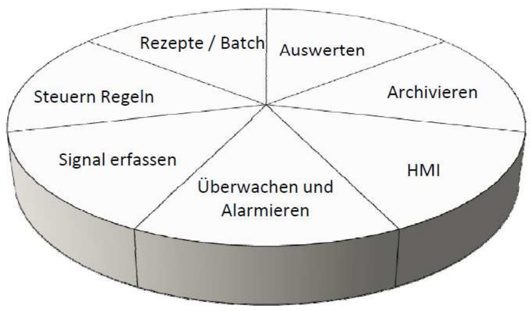
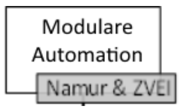


Figure 6: "PCS Pie" According to [23]

The aspects addressed by the working groups are as follows:

- HMI – Operation and monitoring/alarming
- Control – Status-based process management and coupling processes
- Service/maintenance – Determination of the processing of diagnostic information in and from the module

These place requirements on the



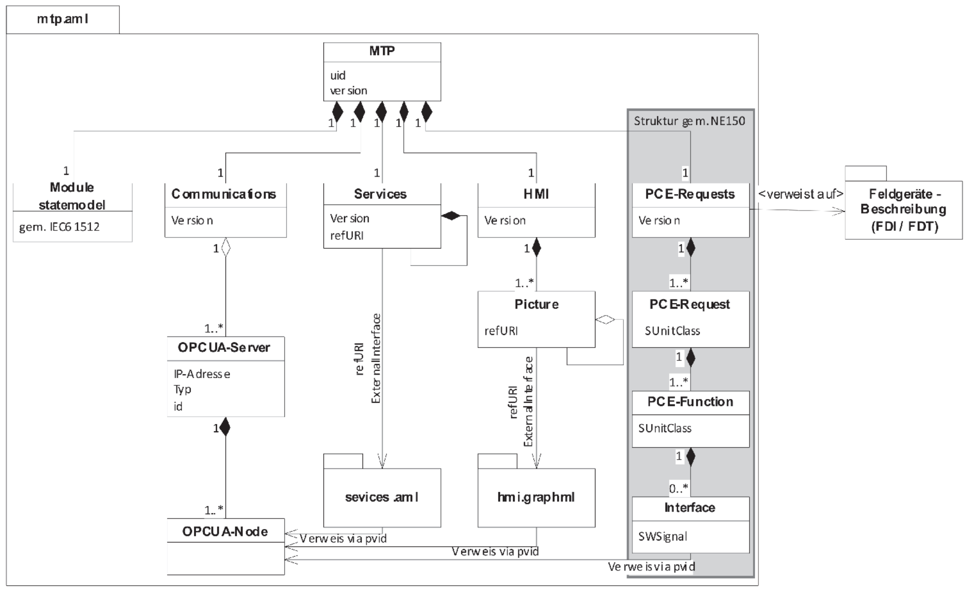


Figure 7: Structure of the MTP.aml

information management that is to be exchanged with the MTP between the module and integration engineering. How this content can be organized in a suitable structure and effectively modeled with descriptive media in the MTP is the task of AK 1.12.1. several relevant descriptive media are considered and assessed for this purpose. The descriptive media used has to be XML based in all cases. It was also determined that there are considerable differences between the innovation cycles of individual aspects. As a result, it was decided to consider the aspects by object and also keep them separate in the model. This enables a simplified exchange of the relevant areas without changing the entire MTP. As a solution, a simple folder structure with several XML-based files was selected. The MTP manifest is the entry point in the structure.

The MTP manifest (mtp.aml – see Figure 7) was modelled in AutomationML (AML). This descriptive media is standardized in IEC 62714 [20] and is suitable through its multi-library approach for describing a large number of aspects. Besides the version, the manifest contains manufacturer information and a unique MTP-ID, organizational features of the following aspects:

- Status model of the module
- Communication part in the communication technology supported by the module
- Services with the services offered by the module
- HMI with the operating screen hierarchy within the module and references to the operating screen descriptions
- PCS point information in case these are required in the PML

Services and HMI reference one or several files in the folder structure of the MTP. For example, the services of a module in the file services.aml. Each service has an independent status model. A Dependency area is reserved for the description of dependencies between different services (e.g. simultaneous exclusion, sequences etc.), in which behavior descriptions are formally represented with PLCopenXML [source (Homepage of PLCopen)] (Figure 8).

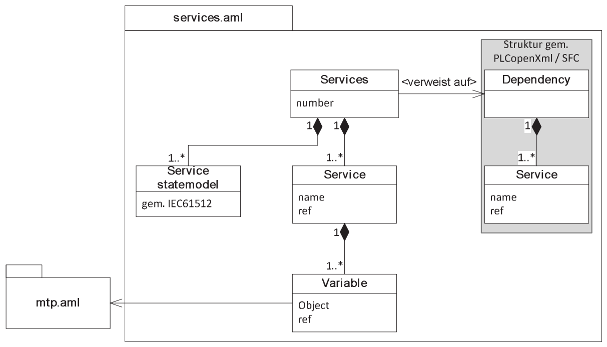


Figure 8: Class Structure Diagram of the Service Description Modelled in AutomationML

The HMI aspect is divided into operating screen hierarchy and operating screen description. The operating screen hierarchy is part of the MTP manifest (mtp.aml – see Figure 7). The individual operating screen descriptions themselves are stored in easy to exchange files in the folder structure of the MTP. The GraphML [Source] format, which was well supported by tools and libraries, was selected as the modeling language for the operating screens consisting of Nodes and Edges. Nodes correspond to the operating screen elements and contain position, size and variable information as well as their meaning. Nodes can be connected by Edges. An Edge can thus be interpreted as a pipe or an information flow on the operating screen. To make diagnostic information of the module also accessible to the operator, each operating screen also contains corresponding status displays with connections to the relevant variables (figure 9)

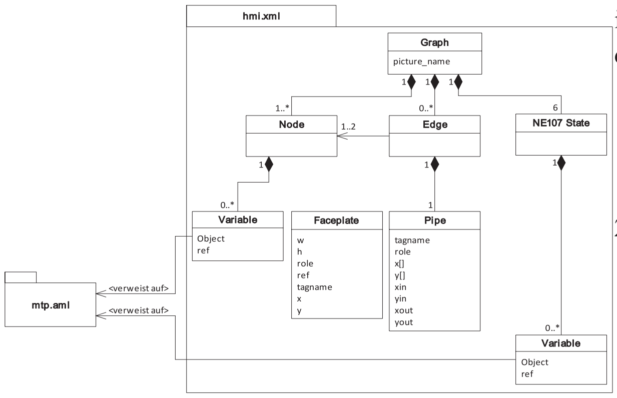


Figure 9: Class Structure Diagram of the Operating Screen Description Modeled in GraphML

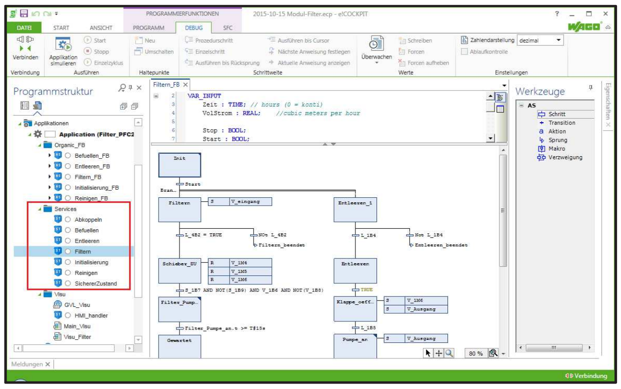
Both the Service.aml and the hmi.xml reference variables which are described in the manifest file.

Current Implementation of the Plant Demonstrator

A technical implementation was created in order to validate the described method. The implementation was completed here with two different software tools:

1. The e!COCKPIT1 engineering tool from WAGO to implement the module engineering and the creation of the MTP,
2. The zenon2 process control system from COPA-DATA for integrating several modules via MTP as well as for orchestrating the module services in the integration engineering.

A plant demonstrator was also planned and created, which shows the benefits of the concept and already covers the requirements of users today. All parts of the implementation to date are presented in the following.



```
VAR_INPUT
    Zeit : TIME; // hours (0 = konti)
    VolStrom : REAL;    //cubic meters per hour

    Stop : BOOL;
    Start : BOOL;
```

Figure 10: Services in the corresponding folder using e!Cockpit

I. Module Automation with e!COCKPIT

The task of module engineering is to design the module logic as a self-contained unit, provide suitable interfaces in the form of services, as well as to present the information required for the visualization of the module. These activities were carried out as part of the DIMA project with the new e!COCKPIT engineering tool. The application development of the module control is carried out in the IEC 61131-3 programming languages. Services are created as functions or function blocks in a

[1]http://www.wago.de/produkte/neuheiten/uebersicht/engineering-software.jsp
[2]*www.copadata.com/de/**zenon**/*

"Services" folder provided for it (see Figure 10).

The services contain their phase logic, parameter variables for assigning parameters, as well as function blocks and function calls. The called function blocks and functions contain the actual control logic. After creating the control application and the services which the module is required to offer, the user interface can be developed. Operating screens are created using templates and linked to variables from the application. (see Figure 11 – left). This completes the creation of the program code of the module control.

The automatic generation of the MTP is the final step of the module engineering (see Figure 11 – right). For this the user describes the meta data of the MTP (manufacturer, version, screen, short description) and selects the services and operating screens which the MTP is to contain (see Figure 11 – right). This last step enables the creation of a variant management system for the module manufacturer. This manufacturer can prepare all conceivable services in the program code and only publish those through the MTP which were also ordered by a customer.

Integration Engineering in the Process Management Level with the zenon PCS

The integration engineering phase is completed via the plant builder/operator. This person uses the MTP generated by the manufacturer in e!COCKPIT to integrate it in the process control system (PCS). The zenon PCS from COPA-DATA is used for this project. The modules are integrated here exclusively via the MTP which are read in using an import wizard (see Figure 12). The wizard creates all required variables and operating screens as well as their links. This enables direct communication between the control system and module control to be set up.

The display of the operating screens is specific to the particular control system, as the screen objects are taken from a template library of the control system that was created beforehand. The information concerning name, size, position and variable link come from the MTP. This ensures that operating screens of modules from different manufacturers, which were read in via different MTPs, comply in the process control system with the "look and feel" specified by the manufacturer.

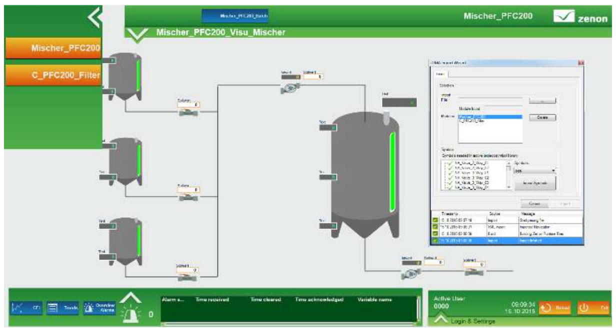
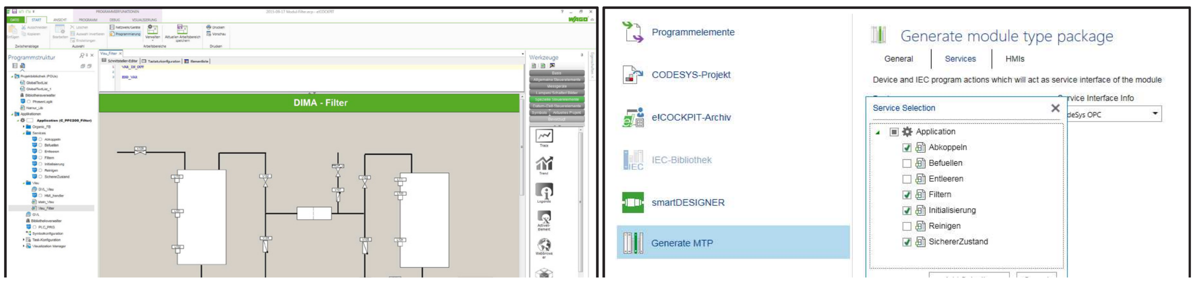


Figure 13: Operating Screen of a Module after MTP Import



Device and IEC program actions which will act as service interface of the module

Service Selection

- Application
  - ✓ Abkoppeln
  - ☐ Befuellen
  - ☐ Entleeren
  - ✓ Filtern
  - ✓ Initialisierung
  - ☐ Reinigen
  - ✓ SichererZustand

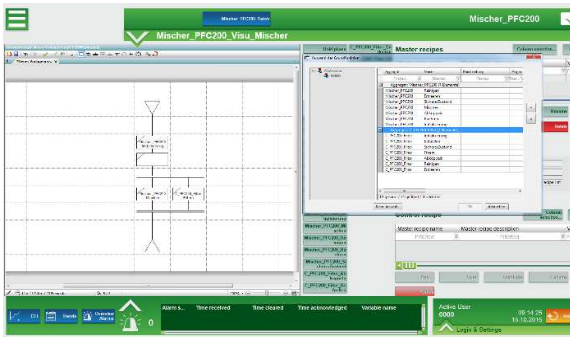


Figure 14: Orchestrating the Services in the Batch Tool

Besides creating and connecting variables and operating screens, the batch tool of the control system also creates the services contained as basic functions in the MTP. The basic functions can be parameterized as required and interlinked in the form of recipes (see Figure 14). After this step, the production process of the plant is ready for operation. If required, the variable management created can be used to implement interlocks in the control system between modules and also create additional messages.

Validation on the Plant Demonstrator

As part of the project, a plant demonstrator was developed for examining the software components. This simulates the requirements of the process industry in the laboratory (see Figure 15). The plant consists of 4 different modules:

1. A mixing module which mixes the materials of three containers according a ratio which can be assigned parameters
2. A distiller module
3. A filter module with a particle filter
6. A filling module for filling liquids in containers

A backbone has also been developed, which provides the required power and compressed air supply, as well as the network infrastructure for communicating between modules and the process management level.

The control logic and the operating screens of all modules were developed in e!COCKPIT. Each module provides different services and all module information was stored in generated MTPs. The implementation of the process illustrated in Figure 16 is planned as a first step.

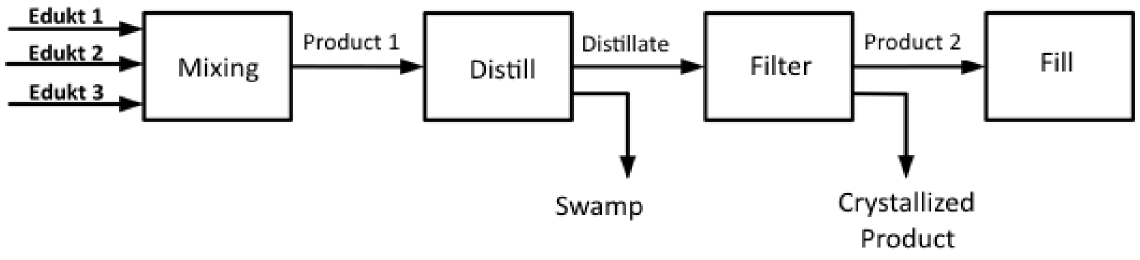


Figure 16: Basic Process of the Plant Demonstrator

Three educts are mixed in the mixing module in the ratio 1:1:1. The module provides a "Mixing" service for this, in which the mixing ratio can be parameterized by the process management level. The resulting product is then distilled ("Distilling" service of the distilling module) and the distillate is cleaned of any impurities resulting from the piping. This is performed by running the "Filter" service in the filter module. The filtered product is then filled ("Filling" service). The creation of the required recipes was completed in zenon and corresponds to the stated services being called simultaneously.

In one scenario the filter module is required to be removed from the plant. A possible reason for this would, for example, be the need to use the module in another plant, together with the finding that it is unnecessary in this process for the currently required product quality. To remove the module, the process is terminated by stopping the recipe. To remove and reuse the filter module it must first be cleaned and emptied. For this the module offers the services "Disconnect" and "Empty". An appropriate recipe is created for this and executed. The module can then be removed both physically as well as from the process management level. A pipe connection is then established between the distiller module and the filling module.

The "Distilling" service is removed from the recipe in the PCS and the plant can then be operated again.

Figure 17: Picture of the used DIMA-demonstrator

II. Outlook

It was possible to use the plant demonstrator with the e!COCKPIT engineering tool and the zenon process control system to validate the results so far of the DIMA project and the stated NAMUR and ZVEI working groups as effective in the field of modular process automation. The aim now is to build on this and push forward the standardization with finalizing in 2019. The method firstly has to be further developed and validated. This particularly involves the development of a generally valid status modeling of the services and modules. This requires the definition and validation of states, state transitions and secondary conditions for changing state based on the IEC 61512 state model. Only after this step has been completed is a fully standardized integration and process management of the modules of different manufacturers possible. Secondly, the current and future results of the working groups must be laid down in the standardization documents. Besides the publication of a NAMUR recommendation an international standard should also be worked towards.

III. Customer applications

[1] CEFIC The European Chemical Industry Council: The european chemical industry - Facts and Figures. http://www.cefic.org/Facts-and-Figures/, 2014

[2] Bramsiepe, C.; Schembecker, G.: Die 50 %-Idee: ModularisierungimPlanungsprozess. ChemieIngenieurTechnik, vo84(5), S. 581–587, 2012

[3] ZVEI e.V: Life-Cycle-Management fürProdukte und Systeme der Automation: EinLeitfaden des ArbeitskreisesSystemaspekteim ZVEI Fachverband Automation. 2010.

[4] Forschungsunion/acatech: Umsetzungsempfehlungenfür das ZukunftsprojektIndustrie 4.0 - Abschlussbericht des ArbeitskreisesIndustrie 4.0. 2013

[5] PAAT Team Tutzing, ProcessNet: Die 50% Idee: VomProduktzurProduktionsanlage in der halbenZeit. ThesenTutzing, ProcessNet, 2009

[6] Bramsiepe, C.; Sievers, S.; Seifert, T.; Stefanidis, G.D.; Vlachos, D.G.; Schnitzer, H.; Muster, B.; Brunner, C.; Sanders, J.P.M.; Bruins, M.E.; Schembecker, G.: Low-cost small scale processing

technologies for production applications in various environments—Mass produced factories. Chemical Engineering and Processing: Process Intensification 51, S. 32–52, 2012

[7] Obst, M.; Holm, T.; Bleuel, S.; Claussnitzer, U.; Evertz, L.; Jäger, T.; Nekolla, T.: Automatisierungim Life Cycle modularer Anlagen: WelcheVeränderungen und Chancensichergeben. atp edition – Automatisierungstechnische Praxis 55(1-2), S. 24-31, 2013

[8] Urbas, L.; Bleuel, S.; Jäger, T.; Schmitz, S.; Evertz, L.; Nekolla, T.: Automatisierung von Prozessmodulen: Von Package-Unit Integration zumodularen Anlagen. atp edition – Automatisierungstechnische Praxis 54(1-2), S. 44-53, 2012

[9] Hady, L.; Wozny, G.: MultikriterielleAspekte der Modularisierungbei der Planungverfahrenstechnischer Anlagen. ChemieIngenieurTechnik 84(5), S. 597–614, 2012

[10] Hady, L.; Dylag, M.; Wozny, G.: Kostenschätzung und KostenkalkulationimchemischenAnlagenbau. CzasopismoTechniczne z.(5-M), S. 159–176, 2008

[11] Lier, St.: Entwicklung einer Bewertungsmethode für die Modularisierung von Produktionssystemen in der Chemieindustrie. Dissertation Ruhr-Universität Bochum, 2013

[12] Fay, A.; Drumm, O.; Eckardt, R.; Gutermuth, G.; Krumsiek, D.; Löwen, U.; Schertl, A.; Schindler, T.; Schröck, S.: Anforderungen an LeitsystemedurchIndustrie 4.0. In: Tagungsband Automation 2014, [CD]. VDI, 2014

[13] NE 148: Anforderungen an die Automatisierungstechnikdurch die Modularisierungverfahrenstechnischer Anlagen. Namur 2013.

[14] Urbas, L.: ZentralesLeitsystemjaodernein?. process-online: http://www.process.vogel.de/automatisierung_prozessleittechnik/articles/451796/?cmp=nl-98; zuletzt: 21.07.2014

[15] Stern, W.: AllgemeinePsychologie auf personalitischerGrundlage. Haag: MartinusNijhof 1935

[16] DIN EN 61512-1: ChargenorientierteFahrweise. 2010

[17] Brandl, D.: Design patterns for flexible manufacturing. ISA 2007

[18] Hawkins, W.; Brandl, D.: Applying ISA-88 in discrete and continuous manufacturing. Momentum Press, 2010

[19] Urbas, L.; Doherr, F.: autoHMI: a model driven software engineering approach for HMIs in process industries. IEEE Int. Conf. Computer Science and Automation Engineering, 2011

[20] DIN EN 62714 DatenaustauschformatfürPlanungsdatenindustriellerAutomatisierungssysteme-AutomationML, 2013

[21] Obst, M.; Hahn, A.; Urbas, L.: Package Unit Integration for Process lndustry-A New Description Approach. IEEE Int. Conf. Emerging Technologies and Factory Automation, 2014

[22] Obst, M.; Holm, T.; Urbas, L.; Fay, A.; Kreft, S.; Hempen, U.; Albers, T.: Beschreibung von Prozessmodulen. atp edition - Automatisierungstechnische Praxis 57(1-2), S. 48-59, 2015

[23] Tauchnitz, T.: Die "neuenProzeßeitsysteme - wohingeht die Reise?.atp edition -

Automatisierungstechnische Praxis 38(11), S. 12–23, 1996

Authors:

Dr.T. Holm and U. Hempen are with WAGO Kontakttechnik GmbH & Co.KG, Germany.

Dr. Thomas Albers is with MINDA Anlagenbau, Germany.

Dr. M. Obst is with GlaxoSmithKline Pharma, Germany.

Prof.Dr.L. Urbasis with TechnischeUniversität Dresden, Germany.

Dr.J. Ladiges is with Nordex SE, Germany

Prof.Dr.A. Fay is with Helmut-Schmidt University / University of Federal Armed Forces Hamburg, Germany

BIOGRAPHY OF THE PRESENTER

Dipl.-Ing. Ulrich H. Hempen was born in Germany. After two technical apprenticeships as a power plant electronics technician, he studied electrical engineering and graduated as a graduate engineer. He then worked from 1990-1995 at Hartmann&Braun AG as Product Manager for HART and Profibus transmitters, from 1995-2001 first as Product Manager and afterwards as Director for System Technology at Endress+Hauser AG, from 2001-2007 as Managing Director of the system integrator Endler&Kumpf and since 2007 as Head of Market Management for WAGO Kontakttechnik in Germany.

SMART CONTRACT - A TRUSTED SYSTEM FOR EMPOWERING PROCESS INDUSTRIES

Arupjyoti Saikia, Rajiv Gupta

Engineers India Limited-New Delhi

ABSTRACT

The Blockchainisadecentralized trust network which has wider applications other than cryptocurrency transactions. Block chain based Smart contracts is an important enabler foran autonomous system. This paper examines mainly on the Smart Contract concepts,underlying technology, its applications specially in process industries and futuristic trends. Two important domain of areas like Supply Chain and IoT are discussed which are relevant to the present development in process industries in terms of autonomous system. It isimportant to keep an active watch on the developments happening in this field of technology when the Blockchain based technologies are the disruptive forces round the corner.

KEYWORDS

Smart Contract, Autonomous System, Blockchain, IoT, Supply Chain

INTRODUCTION

During a routine diagnostic check, a sensor system in a Oil refinery detects that, the hydrocracker reactor catalyst needs replacement, and it automatically triggers a purchase order for procurement of catalyst! Not only that, it executes the entire supply chain management till the completion of contracts which includes payment and contract closure. Sounds too futuristic, too unrealistic? Not any more! The future is here and that is Autonomous System (AS) powered by Smart Contract.

AUTONOMOUS SYSTEM

When the business processes are becoming automated through their relationship with suppliers,key vendors, and customers, that means it necessarily involves the technology, both hardware and software. For the first time, an Autonomous System has evolved which is capable of capturing data, interpreting it and rendering a decision with no human intervention. The shift from humans making all the decisions to machines making decision has begun.

Autonomous systems define our future. They will become embedded in our lives. Autonomous System will also cause disruptions to existing business models, forever changing how supply chains work and how humans work.

The three important building blocks of AS are Artificial Intelligence (AI), Analytics and Internet of Things (IoT). Blockchain and Smart Contracts are by far the most important enablers of an AS. In fact the potential to link Smart Contracts to data science is perhaps one of the first steps towards creating a new world of opportunities. Now, with advances in IoT that connects to cloud computing, wireless sensors and physical objectives that include small computer chips, Smart Contracts can provide that, these inanimate objects achieve a certain level of artificial intelligence to work in the 21st century.

SMART CONTRACT IN AUTONOMUS SYSTEM

Smart Contracts are important part of Blockchain Network. A Smart Contract can be seen as self-executing algorithmic code that's stored and replicated on a distributed ledger system-the Blockchain ecosystem. The algorithm is executed by a network of nodes that run the Blockchain and can result in regular updates of the ledger. The general objectives of Smart Contract design are to satisfy common contractual conditions (such as payment terms, liens, confidentiality, and even enforcement) minimizes exceptions both malicious and accidental, and minimize the need for trusted intermediaries. Related economic goals include lowering fraud loss, arbitration and enforcement costs, and other transactioncosts. Three elements of Smart Contracts that make them distinct are **autonomy, self-sufficiency, and decentralization**. *Autonomy* means that after it is launched and running, a contract and its initiating agent need not be in further contact. Second, Smart Contracts might be *self-sufficient* in their ability to marshal resources-that is, raising funds by providing services or issuing equity, and spending them on needed resources, such as processing power or storage. Third, Smart Contracts are *decentralized* in that they do not subsist on a single centralized server: they are distributed and self-executing across network nodes.

Smart Contracts do not make anything possible that was previously impossible; rather, they allow common problems to be solved in a way that minimizes the need for trust. Minimal trust often makes Things more convenient by taking human judgment out of the equation, thus allowing complete automation.

PLATFORM& RESOURCES

Smart Contracts are digital contracts that are self enforcing or make it prohibitively expensive to break contract. Since a Blockchain can be used as a distributed state machine without a trusted third party, the technology is well suited to support Smart Contracts. While Bitcoin already supports a limited set of Smart Contracts, Ethereum was the first Blockchain to support arbitrary code execution on the Blockchain, allowing any kind of Smart Contract. Depending on the setting and the requirements, a permissionlessBlockchain or a permissioned Blockchain can be used. If details of the contract should remain hidden and the contract is only concerned with a limited set of known participants, a permissioned Blockchain such as Hyperledger Fabric may be the best fit. Otherwise – since Smart Contracts can also be established between multiple anonymous parties – a permissionlessBlockchain such as Ethereum can be used.

Ethereum is an open source Blockchain project that was built specifically to realize Smart Contracts. This new type of Smart Contracting was first introduced in VitalikButerin's white paper, "A next Generation Smart Contract and Decentralized Application Platform". This vision is about applying business logic on a Blockchain, so that transactions of any complexity can be coded, then authorised (or denied) by the network running the code. As such, Ethereum's primary purpose is to be a platform for Smart Contract code, comprising of programs controlling Blockchain assets, executed by a Blockchain protocol, and in this case running on the Ethereum network. In Ethereum there are 3 different languages available to write Smart Contracts i.e. Solidity, Serpent and LLL.

Hyperledger is an open source collaborative platform hosted by the Linux Foundation, and its members include practically all the big pioneers in Blockchain technology development today – from IBM, Intel, Accenture, SAP, J.P. Morgan, EY, Deloitte and Samsung and Huawei.

ADVANTAGES OF SMART CONTRACT

The current legacy contracts suffer from many problems including delay, environment harming, storage and backup, transparency, insecurity, extra cost, and mistrust about the execution.

The block chain technology enabled Smart Contracts has paved the road to address all these concerns.

Fast Settlement: The fact that Smart Contracts are automated by computer code and connected to a globalBlockchain means digital tasks can be performed much faster and more frequently. For example, if a Smart Contract is used to pay a contractor for work, the payment could be executed almost immediately once the work is Complete.

Real time &High Accuracy: Provided that Smart Contracts are written properly, actions performed using them will be very accurate because they adhere strictly to a computer code, which almost entirely eliminates the possibility of mistakes.

Transparency: The terms and conditions of these contracts are fully visible and accessible to all relevant parties. There is no way to dispute them once the contract is established. This facilitates total transparency of the transaction to all concerned parties.

Clear communication: The need for accuracy in detailing the contract results in everything being explicit. There can be no room for miscommunication or misrepresentation. ThusSmart Contracts can drastically cut down on efficiency lost to gaps in communication.

Less Risk, security: The main objective of using Blockchain technology in Smart Contracts is to enforce “trust” through consensus and hashing algorithm. Thus, using a Blockchain to execute Smart Contracts makes them more secure than regular contracts that require centralized counterparties. In addition, they would be virtually unhackable, immutable and not prone to manipulation.

Efficiency: A natural byproduct of the speed and accuracy of these contracts is the efficiency with which they operate. Higher efficiencies result in more value engineering transactions processed per unit time.

Fewer intermediaries: There are no intermediaries involved in Smart Contract settlements, which removes the need for trust in any individual or company.

Trust: Smart Contracts generate absolute confidence in their execution. The transparent, autonomous, and secure nature of the agreement removes any possibility of manipulation, bias or error. Once solemnized, the contract is executed automatically by the network.

Lower Costs, Savings: Since Smart Contracts do not rely on third party intermediaries, the associated costs are reduced or even removed. Legal costs, especially related to dispute resolution, frequently impact on traditional contracts but these can be significantly reduced if different scenarios or contingencies are written into Smart Contracts as code. This process could mean introducing legal participation before a project begins to ultimately save money on legal costs post-handover. Also, as repetitive tasks could be automated, this means that the main costs associated with using Smart Contracts for this purpose would just be the actual creation of the contracts themselves.

Guaranteed outcomes: another attractive feature of these contracts may be the potential to reduce significantly or even eliminate the need for litigation and courts. By using a self executing contract, parties commit themselves to bind by the rules and determinations of the underlying code.

APPLICATION IDEAS OF SMART CONTRACT-PROCESS INDUSTRY

The possible uses of Smart Contracts extend far off past the transfer of digital cash. Smart Contracts can be applied to business activities that involve purchases and exchanges of virtually any types of goods, services or rights.**IF** the Product/service/ information