International Journal of Engineering Trends and Technology
ISSN: 2231 – 5381 /doi:10.14445/22315381/IJETT-V70I1P215

Volume 70 Issue 1, 139-145, January, 2022
©2022 Seventh Sense Research Group®

Original Article

HMI Development Automation with GUI Elements for Object-Oriented Programming Languages Implementation

Syed Khalid Mustafa[1], Vladyslav Yevsieiev[2], Igor Nevliudov[2], Vyacheslav Lyashenko[3], Adel R. Alharbi[4], Wahid Rajeh[4]

[1] *Department of Chemistry, Faculty of Science, University of Tabuk, Kingdom of Saudi Arabia*

[2] *Department of Computer-Integrated Technologies, Automation, and Mechatronics, Kharkiv National University of Radio Electronics, Ukraine*

[3] *Department of Media Systems and Technology,Kharkiv National University of Radio Electronics, Ukraine*

[4] *College of Computing & Information Technology, University of Tabuk, Tabuk, Saudi Arabia.*

[1] khalid.mustafa938@gmail.com

Abstract – *In this article, the authors propose a new method for developing a user interface for industrial information visualizations within Industry 4.0. A feature of the developed method is the use of structured Microsoft Excel files to simplify the presentation of the parameters of the interface being developed. In the course of the experiments, the authors showed the easy way of creating new elements of the industrial interface for the users who are not experts in the field of software development.*

Keywords – *Industry 4.0, Cyber-Physical Production Systems, Additive Cyber Design, HMI, GUI.*

I. INTRODUCTION

High requirements for the production of the high-tech product in tough economic competition led to the emergence of the production processes implementation new vision, which are reflected in the industry 4.0 concepts [1]-[5].

One of the Industry 4.0 concepts' key elements is the development and cyber-physical production systems (CPPS) implementation [6]-[8].

CPPS is a complex physical and cybernetic components synthesis as a single organizational and technical object with a unique architecture both at the physical and cybernetic levels [9]-[15].

Analyzing the cybernetic component, the following architectural levels can be distinguished: Supervisory Control and Data Acquisition (SCADA), Manufacturing Execution System (MES), and Enterprise Resource Planning (ERP) [16]-[21].

As can be seen, all these levels are implemented directly on the use of Human-Machine Interface (HMI) and Graphical User Interface (GUI) object-oriented high-levelprogramming language elements [22]-[24].

At this point, the SCADA, MES, and ERP levels development is seen as classical methods that are used to develop ordinary software products (PP), which leads to negative consequences inherent in the RAD, SADT and RUP methodologies [25]-[28].

As a result, a complex scientific and practical task arises of HMI presentation data using GUI elements formalization new models developed based on which it is possible to implement automation of the additive cyber design development management with the generating not only a visual component possibility but program code fragments for a given development language per customer CPPS requirements [29].

II. DEVELOPING OF A METHOD FOR HMI DESIGN DEVELOPMENT AUTOMATION FOR CYBER-PHYSICAL PRODUCTION SYSTEMS

To implement the CPPS development management process automation, it is necessary to develop a method for the synthesis of visual components that provides a complete input data description, a logical relationship between them, and the properties of the main parameter that are inherent in object-oriented high-level programming languages elements and use the user interface [29].

Define P - as a decent infinite number of custom forms ($Form_n$) which fulfill the condition:

$$\exists Form_n \in P \text{ where } n = \overline{1, i} \quad (1)$$

Where $i \le 40$ is the maximum number of visual user interface forms that make up the designed software product that interacts with the main (base) form ($Form_1^{master}$)? Based on the analysis of interface constructors for object-oriented programming languages, which are implemented in development environments, as well as on the ISO 9241-12-1988 Ergonomic requirements for office work with visual display terminals (VDTs). P.12. Presentation of information $Form_n$ can be represented as subsets of $Form_n\ Propertied\ and\ evens$ and $Components\ on\ Form_n\ structure$ containing the necessary and sufficient amount of data for this study.

Based on this, we define the object $Form_n$ where $n = \overline{1, i}$, as a set:

$$Form_n \in \{Form_n Propertird\ and\ evens,\ Components\ on\ Form_n\ structure\} \quad (2)$$

Provided that $Form_n Propertird\ and\ evens$ and $Components\ on\ Form_n\ structure \ne \varnothing$.

Theorem 1. About $Form_n$ existence only under the conditions $Form_n Propertird\ and\ evens$ and $Components\ on\ Form_n\ structure \ne \varnothing$.

Proof: Suppose that $Components\ on\ Form_n\ structure = \varnothing$, therefore, on $Form_n$ as the working window of the interface, there will be no visualization elements for working with data (GUI elements: *Button*, *Grid*, etc.), the interface of the software product being developed has no functional purpose, and its development is not expedient, and this is logical. Based on this $Components\ on\ Form_n\ structure \ne \varnothing$. This expression is also liquid for $Form_n Propertird\ and\ evens$ and will avoid the paradox. The theorem is proved.

Define $Form_n Propertird\ and\ evens$ as the set of $Main\ propertied$ parameters and values $Properties\ parameters$, inherent in each $Form_n$, as an integral part of $Form_n Propertird\ and\ evens$. Let us clarify that each parameter of the $Main\ propertied$ set belongs to at least one subset of $Properties\ parameters$ values. However, some values of the subset can be $Properties\ parameters = 0$. As a result, we can write:

$$\exists Properties\ parameters_s \in Properties\ parameters : Properties\ parameters_s \ne 0 \quad (3)$$

Let us describe the $Main\ propertied$ subsets and $Properties\ parameters$ as an ordered rigidly fixed sequence of parameters $mp_x \in Main\ propertied$ where mp_x parameters are an integral part of the subset $Main\ propertied \in Form_n Propertied\ and\ evens$.

Accordingly, the set of values $pp_a \in Properties\ parameters_s$, which in turn satisfies the condition:

$Properties\ parameters_s \in Form_n Propertied\ and\ evens$.

Introduce the set Z with elements $(z_1, z_2, \ldots, z_k)$ as a set of possible values variations (size of $Form_n$ in pix, location "Top", "Left", reserved words "alone - no alignment", and logical "True" "False", etc.), depending on the syntax of the $Form_n$ description for a specific programming language and development environment.

Based on this, we define $\exists z_k \in Z : pp_q \ne 0$. In special cases, it is possible $z_k = 0$, provided that $z_k \in Z \subseteq pp_q \in Properties\ parameters \ne \varnothing$. Also, an integral part of the set is the $Main\ events$ subset, which represents a set of parameters $Form_n Propertird\ and\ evens$ as «*Crate*», «*Close*», «*OnClick*», etc. To describe them, we denote $me_h\ (me_1, me_2, \cdots, me_h) \in Main\ events$ all possible actions on the $Form_n$ set. Many parameters are limited by the rules set in a particular development environment. Let's justify the me_h existence as an element of $Main\ events$ the set, as the rule of existence:

$$Main\ events = \{me_h \in Main\ events: Form_n Propertied\ and\ event \ne \varnothing\} \quad (4)$$

Each me_h element is inherent, the set of $Events\ on\ action\ whith\ Form_n$ which consists of an infinite number of elements ea_z – a possible set of names $LingusticVariable$ that link to $ContainerSolution$ with a certain specific "template" in the form of program code ($code_l$). This code contains all valid procedures/functions that can be performed by the me_h element me_h for each object-oriented programming language. Let's describe it as:

$$Events\ on\ action\ whith\ Form_n \in \{ea_1, ea_2, \ldots, ea_z\} \quad (5)$$

Based on the assumptions made, we represent $Form_n Propertird\ and\ evens$ the set as:

$$\{(\exists pp_a \cap mp_x) \in Main\ properties: : Main\ properties \ne \varnothing)\& \&(\exists ea_z \cap me_h \in Main\ properties \ne \varnothing)\} \subseteq \subseteq Form_n Propertied\ and\ evens \quad (6)$$

Expression 6 is not sufficient and complete for solving the problem posed in this study since it does not take into account the visual components and work and data display components. As a consequence, we denote the set $Components\,on\,Form_n\ structure \in Form_n$ provided by:

$$(Components\ on\ Form\ structure\backslash$$
$$Form\ \ Propertied\,and\ evens)$$

As a set of elements describing the interface, visual components, necessary and sufficient to implement data management.

$$Components\,description_f \in Components\,on\,Form_n\ structure$$

Where $f=\overline{1,i}$ is the number of visual components presented on $Form_n$ and performing certain functions? In turn, to describe the visual components, we denote the subsets $Preporties\,components_f$ and $Component\,events\,on\,action_f$ as integral parts of the $Components\,description_f$ set – the number of different visual forms with which you can work with information. Then, any visual element can be described as follows:

$$\begin{aligned}&\{Peporties\,components_f,\\&Component\,events\,on\,action_f\,\}\in\\&\in Components\,description_f\end{aligned} \tag{7}$$

To describe the properties of the $Preporties\,components_f$ and $Component\,events\,on\,action_f$ sets denote the parameter $pc_m \in Preporties\,components_f$, where $m=\overline{1,i}$ is the number of parameters that describe the object properties. pc_m Can take values from the set described above. Unlike $Main\,properties_t$, which is the only one for $Form_n$, $Components\,description_f \in Components\,on\,Form_n\ structure$ the amount should be $f \geq 1$, where $f=\overline{1,i}$. Otherwise, if this condition is not met, Theorem 1 is triggered. An endless variety of visual components that can be used to form a $Components\,on\,Form_n\ structure$ set requires restrictions on $Preporties\,components_f$ that the set of parameter values $pc_1 \in Preporties\,components_1 \neq pc_m \in Preporties\,components_f$ and wherein $pc_1 = pc_m$, as well as based on the fact that the $Components\,description_f$ set may contain the same parameter pc_m names in the $Preporties\,components_f$ set, but at the same time different values from the set or vice versa.

Let's describe the $Component\,events\,on\,action_f$ set as an ordered event set ce_w, and you need to perform a mandatory condition:

$$\begin{aligned}&ce_w \in Componenet\,events\,on\,action_f:\\&Components\,description_f \neq 0\end{aligned} \tag{8}$$

Define ce_w as a set of values (a set of "containers" with program code that can be applied to the pc_m parameter), and then it would be the logical statement:

$$\exists ce_w \in \forall pc_m : Components\,description_f \neq 0 \tag{9}$$

Otherwise, this proves that it does not belong $Components\,description_i \notin Components\,on\,Form_n\ structure$ that is, it is absent as a visual element $Form_n$.

For further chosen solutions research and justification in this method, we introduce abbreviations for the mathematical notation convenience: $Form_n Propertid\ and\ evens$ - we will understand as $Form_n PE$, $Main\ propertied$ as MP_n where n is a $Form_n$ number to whom it belongs, and parameters respectively mp_x^n, where n is the number of the form to which the parameter belongs, x – parameter number in the MP_n set. $Properties\,parameters$ – PP_n set of values, which mp_x^n can take in the form of certain pp_a^n. $mp_x^n \rightarrow pp_a^n$, moreover, many variations pp_a^n can take stored in the Z set, which is described above. Allowed actions that can be performed $Form_n$ as a set of $Main\ events_n$ defined as ME_n where n is a form number. ME_n Set may have a unique set of parameters; we denote them as me_h^n, where n is the identifier of belonging to ME_n, and h is a parameter number. To each of ME_n corresponds the $Events\ on\ action\ whith\ Form$ (EA_n) set, which, in turn, contains a set of values ea_z^n, where n – identifier of belonging to EA_n, and z – parameter number. IT is worth considering that $me_h^n \rightarrow ea_z^n$, provided that $n=n$, ea_z^n values for each parameter me_n^n can be selected from a set of "containers". $Components\,on\,Form_i\ stucture$ Denote as CF^n, where n is a $Form_n$ number, which owns many visual components, which are described as $Components\,description_f$ (CD_x^n), where n shows belonging to one or another $Form_n$, and x is component number on $Form_n$ in the structure of CF^n the set. We represent the structure of the CD_x^n set as an orderly set of parameters describing visual components for working with

$\mathit{Preporties\ components}_f$ data in what follows we denote it by PC_y^x, where x shows belonging to CD_x^n, and n – the component number in this structure. Define PC_y^x as an ordered set of pc_m^y parameters, where $y = y$ the accessory number to PC_y^x, m – as the ordinal number of the parameters in the PC_y^x array. $\mathit{Component\ events\ on\ action}_i$ Define as CE_z^x, where x – number of belonging to a particular CD_x^n, z is a serial number. Let's describe CE_z^x it as a set of events that a visual component ce_w^z can handle, wherefrom the numbering of CE_z^x. A set of events ce_w^z can own a "solution container". Note that the CD_x^n set can be used an infinite number of times, while visual forms can perform different functions and be described by different parameters, but can have the same software solution from the "solution container".

Cybernetic component presentation of the developed CPPS (P) as a mathematical description of structures $Form_1^{master}$ and the connections between them are shown in Figure 1.

To define connections, we define Ξ them as a condition for interaction between sets $Form_n$. In what follows, we consider such a record $Form_1^{master} \xrightarrow{\Xi} Form_n^{slave}$ as interaction (data transfer, call, etc.) $Form_1^{master}$ through the me_h^n event or ce_w^n event belonging to any GUI element that belongs to $Form_1^{master}$ on $Form_n^{slave}$.

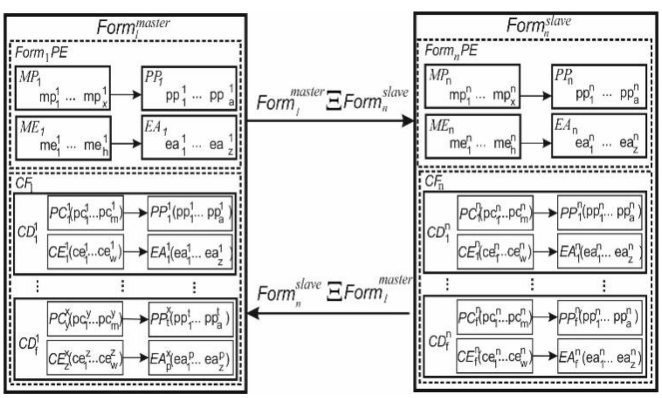

Fig. 1:Graphical representation of connections between $Form_1^{master}$ and $Form_n^{slave}$ through events.

Based on the proposed solutions, within the framework of these studies, we define the following record form Ξ for $(Form_n PE, CF_n) \in Form_n^{slave}$ set and the purpose of each of its subsets:

- mathematical description of $Form_n PE$ the set:

$$Form_n PE \in \underbrace{((mp_1^n, mp_2^n, ..., mp_x^n) \in MP_n}_{\text{Set of parameters}} \xrightarrow{\zeta_p} \underbrace{((pp_1^n, pp_2^n, ..., pp_a^n) \in PP_n)}_{\text{Set of values}} \wedge \wedge \underbrace{((me_1^n, me_2^n, ..., me_h^n) \in ME_n}_{\text{Set of events}} \xrightarrow{\zeta_p} \xrightarrow{\zeta_p} \underbrace{((ea_1^n, ea_2^n, ..., ea_z^n) \in EA_n)}_{\text{Set of "linguistic names"}} \xrightarrow{\varphi_e} \xrightarrow{\varphi_e} \underbrace{((z_1^o, z_2^o, ..., z_q^o) \in Z_o)}_{\text{Set of "solutions containers"}}. \tag{10}$$

- mathematical description of CF_n the set:

$$CF_n \in (\underset{element}{CD_x^n} \in [\underbrace{((pc_1^x, pc_2^x, ..., pc_m^x) \in PC_y^x)}_{\text{Set of elements parameters}} \xrightarrow{\varepsilon_v} \underbrace{((pp_1^x, pp_2^x, ..., pp_a^x) \in PP_t^x}_{\text{set of values}} \wedge \wedge \underbrace{((ce_1^z, ce_2^z, ..., ce_w^z) \in CE_z^x)}_{\text{set of events}} \xrightarrow{\varepsilon_v} \xrightarrow{\varepsilon_v} \underbrace{((ea_1^p, ea_2^p, .., ea_z^p) \in EA_p^x)}_{\text{Set of "linguistic names"}} \xrightarrow{\varphi_e} \xrightarrow{\varphi_e} \underbrace{((z_1^o, z_2^o, ..., z_q^o) \in Z_o)}_{\text{Set of "solution containers"}}]) \tag{11}$$

III. ADDITIVE CYBER DESIGN DATA REPRESENTATION STRUCTURE AND EXPERIMENTAL RESEARCH

The presented concept is based on the graphical interface elements used as basic information carriers about the developed CPPS.

$Form_1^{master}$ Is the main form of the CPPS cyber component is understood, which is characterized by a basic block set: $Form_1 PE$, CF_1. $Form_1 PE$ Block consists of interconnected structural elements MP_1: parameters $mp_1^1, ..., mp_t^1 \rightarrow PP_1$: values $pp_1^1, ..., pp_q^1$. Each mp_t^1 corresponds to one pp_q^1, which can take the values of digital, logical operations (false, true) or reserved describing values parameters pp_q^1 for a certain high-level programming language (Top, Batten, etc.), the set of such parameters is strictly limited and carries information about the $Form_1^{master}$ conditions of visual display for the user (centered on the desktop, maximized to full desktop, etc.). ME_1 The block is an events collection $me_1^1, ..., me_h^1$ that can be superimposed on $Form_1^{master}$ when processing actions because of «*Create form*», «*Close form,*» etc. The set of events and parameters in the form of program code is strictly defined for each language and development environment. To each of the events $me_1^1,..., me_h^1$ correspond $ea_1^1,..., ea_z^1$ EA_1 block which can take values as functions and user interaction procedures described as program code fragments.

Each $Form_1^{master}$ has an endless CF_1 set which is the structure of $Form_1^{master}$ in the construction tree form, where each tree element is a set of visual components designed to work with data and grouping elements (input elements (*Edit*) and data display (*Grid*), interface elements (*Menu*), grouping elements (*GroupBox*)). Each element is presented in a block CD_1^1 that represents the relationship $PC_1^1 \rightarrow PP_1^1$. Based on the proposed structure PC_1^1 has the same properties as the element MP_1 but at the same time CE_1^1 has an endless variety of actions EA_1^1 which may coincide with EA_1 (access to the database, calculations, closed forms, etc.) when a certain event is called (clicking on the "Button" component, hovering the mouse, etc.). Depending on the general structure of building the CPPS graphical interface (how many elements of type $Form_1^{master},\ldots, Form_n^{slave}$ will be used), it is necessary to take into account the transfer of global variables and functions of transition between interface windows. As a result, the interaction between elements $Form_1^{master} \Xi Form_n^{slave}$ should be taken into account within the developed CPPS; on average, the type elements number $Form_1^{master}$ and can range from 1 ... 25-30 and higher, they can be called as floating inside the main $Form_1^{master}$ and refer to it.

Based on the proposed graphical structure of relationships between properties and parameters and events, we will develop a data representation structure in Microsoft Excel 2003 to create HMI CPPS prototype for the Pascal programming language in Red Studio X5, which is shown in Figure 2.

Fragment of the completed data representation structure for prototyping $Form_1^{master}$ in $Form_1 PE$ the block with connections $MP_1 \rightarrow PP_1$ и $ME_1 \rightarrow EA_1$ shown in Figure 3.

Depending on the chosen language and the CPPS cybernetic component development environment, the name and purpose of the elements in the $MP_1(mp_1^1,..., mp_t^1)$ and $ME_1(me_1^1,..., me_h^1)$ blocks will change, and this statement is valid for block elements $CD_1^1,\ldots,CD_f^1$. But at the same time, the data representation structure for creating a prototype of the CPPS cybernetic component graphical interface will be the same under the conditions that object-oriented programming languages will be used.

Fig. 2 Example of a data presentation structure in Excel 2003 for Pascal in the Red Studio X5 IDE for prototyping a user interface.

32	Structure	Button1	Properties		Events	
33			Align		OnClick	Close
34			Cancel		OnExit	
35			Caption	Close		
36			Font			
37			Height	25		
38			Left	552		
39			Top	367		
40			Width	75		
41			DropDownMenu			
42		GroupBox1	Properties		Events	
43			Align		OnClick	
44			Cancel		OnExit	
45			Caption	Test		
46			Font			
47			Height	343		
48			Left	8		
49			Top	8		
50			Width	619		
51			DropDownMenu			
52			TabOrder	1		
53			Edit1	Properties		Events
54				Align	alNone	OnClick
55				Cancel		
56				Text	Search	
57				Font		
58				Height	21	
59				Left	16	
60				Top	24	
61				Width	585	

Fig. 3 Fragment of the completed block data representation structure CF_1 which contains the elements CD_1^1 «*Button1*» and CD_2^1 «*GroupBox1*».

To test the proposed method was developed "Management processes for the complex CPPS development automation" system was, one of the functions in which is data import of * .xls format, the structure of which is shown in Figure 2. Based on this fragment, the developed system " Management processes for the complex CPPS development automation" will generate the following HMI presented in Figure 4.

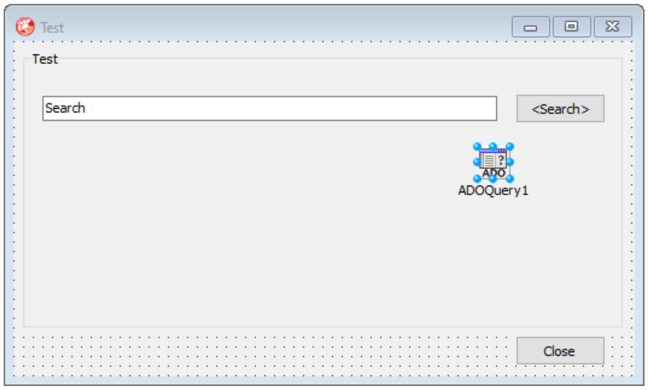

Fig. 4 Generated HMI in Red Studio X5 Form Designer.

To work with a test search form, the user selects the Edit1 element and enters the query parameters from the keyboard and presses the <Search> button. This event activates the following procedure, shown in Figure 5.

```
procedure TForm1.Button1Click(Sender: TObject);
begin
Close;
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
 ADOQuery1.SQL.Clear;
 ADOQuery1.SQL.Add('Select * from operation');
 ADOQuery1.SQL.Add('where name like '''+'+'''+'''+Edit1.Text+'''+''+'');
 ADOQuery1.Active:=true;
end;

procedure TForm1.Edit1Click(Sender: TObject);
begin
Edit1.Text:='';
end;
```

Fig. 5 Generated program code (Solution container) for "Linguistic variables" Buttion.OnClick = Close and Edit1Click = Search_ADO_BD

As you can see from Figures 4 and 5, the developer receives the generated files: *.dfm – the graphical form of the HMI, *.pas – the generated fragment of the program code, and *.dpr – common project file in Red Studio X5.

IV. CONCLUSION

The proposed method of data formalization for additive CPPS cyber design management development automation was implemented as a function in the system "Management processes for the complex CPPS development automation". To check the correctness of the scientific decisions taken, a computer experiment was carried out to automate the control process, the creation of the simplest HMI form for creating an additive cyber design CPPS. The results obtained were compared with the classic RAD software development method. Based on this, it was found that the proposed method reduced the development time by 1.25 times, due to the use of "Linguistic variables" and the generated program code using "Solution containers".

REFERENCES

[1] Ugur M. Dilberoglu, BaharGharehpapagh, UlasYaman, MelikDolen. The Role of Additive Manufacturing in the Era of Industry 4.0, Procedia Manufacturing, 11 (2017) 545-554.

[2] Rami Matarneh, SvitlanaMaksymova, ZhannaDeineko, Vyacheslav Lyashenko. Building Robot Voice Control Training Methodology Using Artificial Neural Net, International Journal of Civil Engineering and Technology, 8(10) (2017) 523-532.

[3] Lucas Santos Dalenogare, Guilherme Brittes Benitez, Néstor Fabián Ayala, Alejandro Germán Frank. The expected contribution of Industry 4.0 technologies for industrial performances, International Journal of Production Economics, 204 (2018) 383-394.

[4] Oleksandr Kuzomin, Mohammad Ayaz Ahmad, HryhoriiKots, Vyacheslav Lyashenko, MariiaTkachenko. Preventing of technogenic risks in the functioning of an industrial enterprise, International Journal of Civil Engineering and Technology, 7(3) (2016) 262-270.

[5] Yang Lu. Industry 4.0: A survey on technologies, applications and open research issues, Journal of Industrial Information Integration, 6 (2017) 1-10.

[6] N. Jazdi. Cyber physical systems in the context of Industry 4.0, in 2014 IEEE International Conference on Automation, Quality and Testing, Robotics, Cluj-Napoca, Romania, (2014) 22-24.

[7] László Monostori. Cyber-physical production systems: Roots, expectations and R&D challenges, Procedia CIRP, 17 (2014) 9-13.

[8] P.J. Mosterman, J. Zander. Industry 4.0 as a Cyber-Physical System study, Softw Syst Model, 15 (2016) 17-29.

[9] Jin Ho Kim. A Review of Cyber-Physical System Research Relevant to the Emerging IT Trends: Industry 4.0, IoT, Big Data, and Cloud Computing, Journal of Industrial Integration and Management, 02 (03) (2017).

[10] Jay Lee, Behrad Bagheri, Hung-An Kao. A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems, Manufacturing Letters, 3 (2015) 18-23.

[11] Behrad Bagheri, Shanhu Yang, Hung-An Kao, Jay Lee. Cyber-physical Systems Architecture for Self-Aware Machines in Industry 4.0 Environment, IFAC-PapersOnLine, 48(3) (2015) 1622-1627.

[12] SergiiIarovyi, Wael M. Mohammed, Andrei Lobov, Borja Ramis Ferrer, Jose L. Martinez Lastra. Cyber–Physical Systems for Open-Knowledge- Driven Manufacturing Execution Systems, Proceedings of the IEEE, 104(5) (2016) 1142 - 1154.

[13] SvitlanaSotnik, Rami Matarneh, Vyacheslav Lyashenko. System model tooling for injection molding. International Journal of Mechanical Engineering and Technology, 8(9) (2017) 378-390.

[14] M. Ayaz, T. Sinelnikova, S. K. Mustafa, V. Lyashenko. Features of the Construction and Control of the Navigation System of a Mobile Robot, International Journal of Emerging Trends in Engineering Research, 8(4) (2020) 1445-1449.

[15] Jay Lee, Behrad Bagheri, Chao Jin. Introduction to cyber manufacturing, Manufacturing Letters, 8 (2016) 11-15.

[16] Armando W., Bangemann Thomas, Karnouskos Stamatis, DelsingJerker, Stluka Petr, Harrison Robert, Jammes Francois, Martinez Lastra, Jose L. Industrial Cloud-Based Cyber-Physical Systems, Springer International Publishing Switzerland, Springer, Cham. (2014) 245.

[17] Igor Nevliudov, Vladyslav Yevsieiev, Jalal Hasan Baker, M. Ayaz Ahmad, Vyacheslav Lyashenko. Development of a cyber design modeling declarative Language for cyber physical production systems, J. Math. Comput. Sci., 11(1) (2021) 520-542.

[18] SangSu Choi, Gyhun Kang, Chanmo Jun, , Ju Yeon, Lee, Seukjoo Han. Cyber-physical systems: a case study of development for manufacturing industry, International Journal of Computer Applications in Technology, 55(4) (2017) 298-307.

[19] Omid Givehchi, Klaus Landsdorf, Pieter Simoens, Armando Walter Colombo. Interoperability for Industrial Cyber-Physical Systems: An Approach for Legacy Systems, IEEE Transactions on Industrial Informatics, 13(6) (2017) 3370-3378.

[20] Chunyang Yu, Xuanlin Jiang, Shiqiang Yu, Cheng Yang. Blockchain-based shared manufacturing in support of cyber physical systems: concept, framework, and operation, Robotics and Computer-Integrated Manufacturing, 64 (2020).

[21] Armando W. Colombo, StamatisKarnouskos, OkyayKaynak, Yang Shi, Shen Yin. Industrial Cyberphysical Systems: A Backbone of the Fourth Industrial Revolution, IEEE Industrial Electronics Magazine, 11(1) (2017) 6-16.

[22] Paola Fantini, Marta Pinzone, Marco Taisch. Placing the operator at the centre of Industry 4.0 design: Modelling and assessing human activities within cyber-physical systems, Computers & Industrial Engineering, 139 (2020).

[23] Albert T. Jones, David Romero, Thorsten Wuest. Modeling agents as joint cognitive systems in smart manufacturing systems, Manufacturing Letters, 17 (2018) 6-8.

[24] Marie-Pierre Pacaux-Lemoine, Quentin Berdal, Simon Enjalbert, Damien Trentesaux. Towards human-based industrial cyber-physical systems, in 2018 IEEE Industrial Cyber-Physical Systems (ICPS), (2018).

[25] Edward A. Lee. Cyber Physical Systems: Design Challenges, in 2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC), Orlando, FL, USA, (2008).

[26] Amy J. C. Trappey, Charles V. Trappey, UsharaniHareesh Govindarajan, John J. Sun, Allen C. Chuang. A Review of Technology Standards and Patent Portfolios for Enabling Cyber-Physical Systems in Advanced Manufacturing, IEEE Access, 4 (2016) 7356-7382.

[27] Moshev E., Meshalkin V., Romashkin M. Development of Models and Algorithms for Intellectual Support of Life Cycle of Chemical Production Equipment, Cyber-Physical Systems: Advances in Design & Modelling. Studies in Systems, Decision and Control, 259 (2019) 153-165.

[28] Urcun John Tanik. Cyberphysical Design Automation Framework for Knowledge-based Engineering, The International Journal on Multidisciplinary Approaches on Innovation, 1(1) (2013) 158-178.

[29] Igor Nevliudov, Vladyslav Yevsieiev, SvitlanaMaksymova, Inna Filippenko. Development of an architectural-logical model to automate the management of the process of creating complex cyber-physical industrial systems, Eastern-European Journal of Enterprise Technologies, 4 (3(106)) (2020) 44-52.