The Design of the Diagrammatic and Semantic Models for Process Modelling Language for Digital Triplet

Hideaki Takeda[1], Seiji Koide[2], Sungmin Joo[3], Mizuki Kato[4], Leon Akiyama[4], Jumpei Goto[5], Shinsuke Kondoh[4] and Yasushi Umeda[6]

[1]*National Institute of Informaitcs, 2-1-2, Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan*
[2]*Ontolonomy, LLC, Yokohama, Japan*
[3]*Yamanashi Prefectural University, 5-11-1 Iida, Kofu, Yamanashi, Japan*
[4]*Department of Precision Engineering, The University of Tokyo,7-3-1, Hongo, Bunkyo-ku, Tokyo 113-8656, Japan*
[5]*The University of Tokyo (currently CADDi Inc)*
[6]*RACE (Research into Artifact, Center for Engineering), The University of Tokyo,7-3-1, Hongo, Bunkyo-ku, Tokyo 113-8656, Japan*

Abstract

The paper introduces the ”Digital Triplet (D3)” framework, aiming to integrate human engineers and cyber-physical production systems (CPPS) in Industry 4.0. While existing research focuses on CPPS and digital twins, this framework emphasizes engineers' problem-solving capabilities in enhancing CPPS. It introduces the ”Process Modelling Language for Digital Triplet (PD3)” to represent engineers' intentions, judgments, and rationale in CPPS operations. The PD3 language is designed to be both machine-readable and human-readable, accommodating the needs of knowledge engineers and manufacturing engineers who are not necessarily programmers. The PD3 ontology, based on Semantic Web technologies, underpins the representation of engineering processes and their relationships. The ontology defines classes, properties, and relationships to facilitate the modeling of engineering processes, intentions, rationale, and more. The prototype is built for the validation of PD3 Concept. It consists of the interface, the conversion system between XML and RDF (turtle), the RDF database, and the inference system that enables various inferences like execution simulation.

Keywords

Digital Twin, Digital Triplet (D3), Dicision Making, Process Language, Engineering Knowledge, Ontology

1. Introduction

The rapid digitalization of the manufacturing industry, exemplified by Industry 4.0, has sparked considerable advancements. While existing research primarily concentrates on the automation of production systems through cyber-physical production systems (CPPS)[1] and digital twin technologies[2], a growing emphasis is placed on the role of engineers in designing, maintaining,

Second International Workshop on Semantic Industrial Information Modelling (SemIIM) Co-located with the International Semantic Web Conference (ISWC 2023)

takeda@nii.ac.jp (H. Takeda); koide@ontolonomy.co.jp (S. Koide); s-joo@yamanashi-ken.ac.jp (S. Joo); kato@susdesign.t.u-tokyo.ac.jp (M. Kato); akiyama@susdesign.t.u-tokyo.ac.jp (L. Akiyama); kondoh@race.t.u-tokyo.ac.jp (S. Kondoh); umeda@race.t.u-tokyo.ac.jp (Y. Umeda)
http://www-kasm.nii.ac.jp/ (H. Takeda)
0000-0002-2909-7163 (H. Takeda); 0000-0002-5386-4887 (S. Kondoh); 0000-0003-4688-9602 (Y. Umeda)

and enhancing CPPS. The concept of human-centered CPPS[3] has been explored, including the transfer of human problem-solving capabilities to CPPS and collaborative frameworks[4]. However, the challenge of enabling engineers to harness digitalization for value creation remains under-addressed. This paper addresses this gap by focusing on the enhancement of engineers' problem-solving abilities through digitalization to drive the continuous improvement of CPPS, particularly Kaizen.

To tackle this issue, a novel framework called the 'digital triplet (D3)' has been proposed, designed to support engineers[5, 6]. D3 establishes an 'intelligent activity world' that interfaces with digital twins, envisioning a harmonious interplay between human intelligence and digital tools. Furthermore, a prototype system, D3LF@RACE (digital triplet type learning factory at RACE) has been deployed within a learning factory context. Learning factories, replicated real-world factory setups in academic institutions, serve both educational and research purposes. The development of D3 within this learning factory setting, featuring digital twins, is meticulously detailed. In order to realize the above system, we introduce a structured language called Process Modelling Language for Digital Triplet (hereinafter called PD3) to describe engineering processes, hierarchically mapping engineers' intentions and judgments to CPPS operations. This approach enables the explicit representation of engineering processes within a computer, facilitating the recording of expert engineers' methodologies, knowledge extraction, novice engineer education, and overall task support.

In this paper, we show the design of PD3, in particular, the schematic model and the semantic model. Since PD3 is defined by referring IDEF0, the schematic model is important. On the other hand, the computational model is also important for processing it in computers. We adopt Semantic Web technologies, i.e., the RDF model with the ontology is provided as the semantic model for PD3. Then we implement the prototype system to handle PD3 process data by combination with the interface system, the database system, and the inference system.

2. Digital Triplet (D3) Framework

At its core, the D3 framework (see Figure 1) amalgamates human engineers and CPPS, culminating in an 'intelligent activity world' that complements the physical and cyber worlds in CPPS. This world emphasizes human intelligence-driven problem-solving, diverging from the automated functions of CPPS. This human-centric approach necessitates D3's support across CPPS activities encompassing planning, design, ramp-up, operation, maintenance, continuous improvement, and withdrawal.

Figure 2 delves deeper into the framework, depicting an engineering process as a sequence of 'engineering cycles' involving data collection, information analysis, evaluation, decision-making, and plan execution. While traditional CPPS automates these processes, the framework acknowledges that engineers actively construct diverse engineering processes. 'Knowledge' (depicted as yellow rectangles) is indispensable in this process. The paper emphasizes the importance of describing engineering processes as reusable process knowledge. Achieving this entails recording all CPPS operations, including data collection, software operations, and physical-world actions. The paper's approach connects experts' judgments, intentions, and rationale with CPPS operations through a language called PD3 (Process Modelling Language for

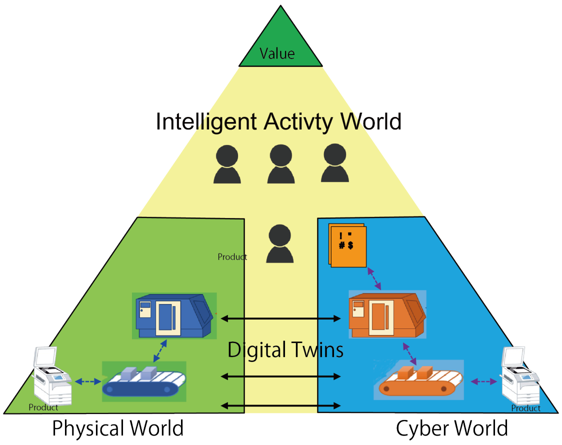


Figure 1: Concept of digital triplet at the manufacturing stage

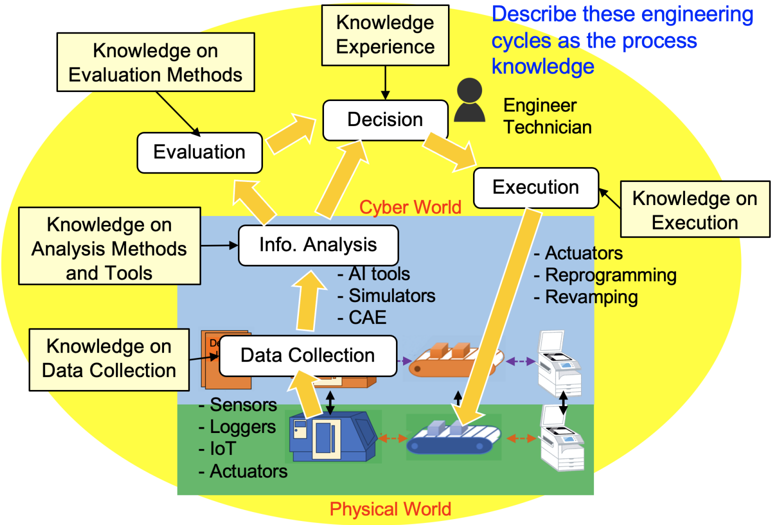


Figure 2: Engineering cycles executed on digital triplet

Digital Triplet), based on the IDEF0 framework[7]. This approach not only links the symbolic representation of human insights to CPPS operations but also the meaning and significance of each operation in CPPS can be understood by the symbolic representation..

The paper's methodology for modelling engineering processes involves several steps. Expert manufacturing engineers execute continuous improvement activities, recorded by knowledge engineers as 'activity logs.' These logs encapsulate observations, data and software usage, physical operations, and results. Subsequently, these logs are refined into structured 'process descriptions,' removing noise and redundancy. Novice engineers can learn from these descriptions, facilitating knowledge transfer. These process descriptions are further enhanced into 'smart process descriptions' by incorporating digital tools. Accumulated process descriptions enable knowledge discovery through data mining and AI techniques.

This paper introduces the D3 framework to empower engineers within the context of CPPS and digitalization. It underscores the 'intelligent activity world' as a nexus of human ingenuity and digital tools. The proposed methodology and prototype system showcase the potential of D3 in bridging human-CPPS collaboration. By harnessing the synergy between human expertise and digitalization, D3 advances problem-solving and continuous improvement in manufacturing.

3. PD3: A Language for Describing Engineering Process

Process Modelling Language for Digital Triplet (PD3) is designed to realize the Digital Triplet environment described in the previous section. The followings are the requirements for the language to represent Digital Triplet.

1. The process description should be machine- and human-readable.
2. The language should represent engineers' judgment, intention, and rationale along with the engineering process explicitly.
3. The language should relate the symbolic representation to operations in CPPS.

IDEF0 approach is adopted to fulfill the requirement (1) and (2) since IDEF0 is designed as a human-friendly diagram. The expected users of PD3 are not programmers, rather knowledge engineers and manufacturing engineers. So the human-friendly form is crucial. The role of IDEF0 is close to one of PD3, but the specification in detail is not fit with one of PD3, in particular, specific descriptions like intention and the hierarchical structure. PD3 extends IDEF0 diagram for this purpose. Semantic Web approach, in particular, ontology and RDF is adopted to fulfill the requirement (1) and (3). The ontology works as a nexus between the diagrammatic model and the data model. All the elements and relations are defined in the ontology. RDF and RDF Schema are used to represent the data in computers. RDF descriptions with ontology are beneficial for some sort of inference from simple consistency checking to extraction of data with a specific purpose. Another benefit of the approach is the flexibility of data boundary. The engineering process can be revised and sometimes integrated over time. Thanks to linked data nature, data by RDF can be easily associated with each other even though data come from different engineering processes.

3.1. Diagrammatic model for PD3

PD3 diagrammatic model is defined as the extension of IDEF0 diagram (see Figure 3). The followings are modifications and extensions for IDEF0:

- Box represents Action that human can take
- Special Action "Start" and "End" are provided. An engineering process should start with "Start" Action and end with "End" Acton
- Action may have an operation for information, such as parameter assignment. It is included as an attribute of Box.

- Horizontal arrow means information flow, i.e., the flow of control and data just as IDEF0
- Information flow may depict “if” or “else” flow when output from action has choices. It is denoted as label starting with “[if]” or “[else]”
- Upward arrow means resources just as IDEF0, in particular, either Substance, Engineer, Knowledge, Tool or Document
- Downward arrow means Intention of Action
- Left-down arrow means Rationale for Action, i.e., why the action would be taken
- Right-down arrow means Annotation for Action, i.e., additional information for Action.
- Container represents the hierarchical relationship between an Action and Actions as a component of the Action (corresponding to the decomposition structure in IDEF0). It enables an engineering process with a hierarchical structure in a single diagram, while IDEF0 requires multiple diagrams for one with a hierarchical structure.
- Container has a fragment of an engineering process as content. It should also start with “Start” Action and end with “End” Action.
- Container may be invoked iteratively. The condition for iteration should be described as a label of Arrow from Container to Action.

Furthermore, three layers are introduced to distinguish three different Digital Triplet worlds;

- Problem-solving layer to describe engineers’ actions to solve problems (e.g., judgments) with descriptions of intentions and rationales behind the actions. In a diagram, it is located at the top and the elements in it are colored red.
- Information layer to describe engineers’ actions for operating software and handling data in the cyber world. In a diagram, it is located the middle and the elements in it are colored blue.
- Physical layer to describe engineers’ actions for operating the factory in the physical world. In a diagram, it is located at the bottom and the elements in it are colored green.

Container can connect actions in different worlds, i.e., an Action in Problem-solving layer associated with a Container in Information or Physical layers where actions in these layers are located. Thus an engineering process can be represented as a graph of actions across three layers.

3.2. Semantic Model for PD3

The semantic model for PD3 gives an application-independent data structure that various applications can be used. Firstly, the ontology for PD3 is defined (see Figure 4). The elements like actions and arcs are represented as classes and attributes for them as properties. Class “Action” and “Flow” are most primary elements in PD3 while elements such as “Container” and “Intention” are represented as subclasses of Class “Node” and different arrows such as “RationaleFlow” and “IntentionFlow” are those of Class “Arc”. The relationship between box and arrow is represented as property such as “source”, “target”, “input” and “output”. The implicit relationship between a Container and its components is also represented as property (“expansion” and “contraction”). All classes have properties for ID, geometry, layer, and value

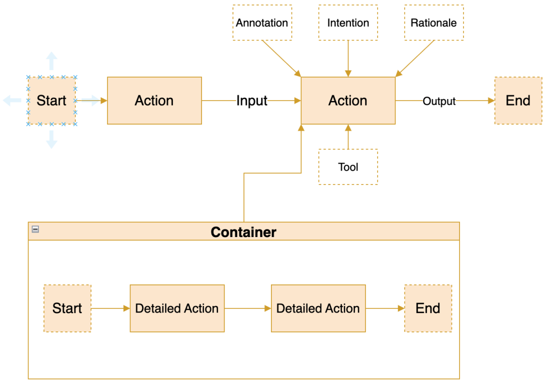


Figure 3: The diagrammatic model for PD3

to describe basic information. Another simple ontology describes meta-relations between engineering processes (see Figure 5). Class “Engineering process” is provided to represent each engineering process and the elements of an engineering process such as actions and arrows are depicted with Property “includes” (and “isIncludedin” as reverse relation). Engineering processes can be associated with each other. An engineering process can be used by another one (Property “epUses” and “epIsUsedBy”) and can be derived by others (Property “epDerives” and “epIsDerivedFrom”). Such relationship between engineering processes can be got down to the relationship between elements in engineering processes, i.e., properties from Entity to Entity such as “uses”, “isUsedBy”, “derives” and “isDerivedFrom” are also provided (see Figure 4). All elements in the ontology are defined as RDF schema.

4. PD3 Platform Prototype

A prototype system designed to facilitate PD3-based data management has been constructed to assess the functionality of PD3, considering both user and computational aspects. The prototype system consists of primarily four key components: the interface system, the data conversion system, the database system, and the inference system (refer to Figure 6 for an overview). In order to ensure sustainability and scalability, the prototype system uses established systems and technologies wherever feasible. This strategic integration of existing resources serves to fortify the system’s long-term viability and potential for expansion.

4.1. The Interface System

The interface system is realized as draw.io system[1] with the specialized library for PD3. draw.io is a cross-platform drawing software that can be used to create various diagrams such as UML and flowchart. The specialized library is provided in which elements correspond to those defined in Section 3.1 (see Figure 3). There are several types of boxes and arrows for three layers

[1] http://www.drawio.com/

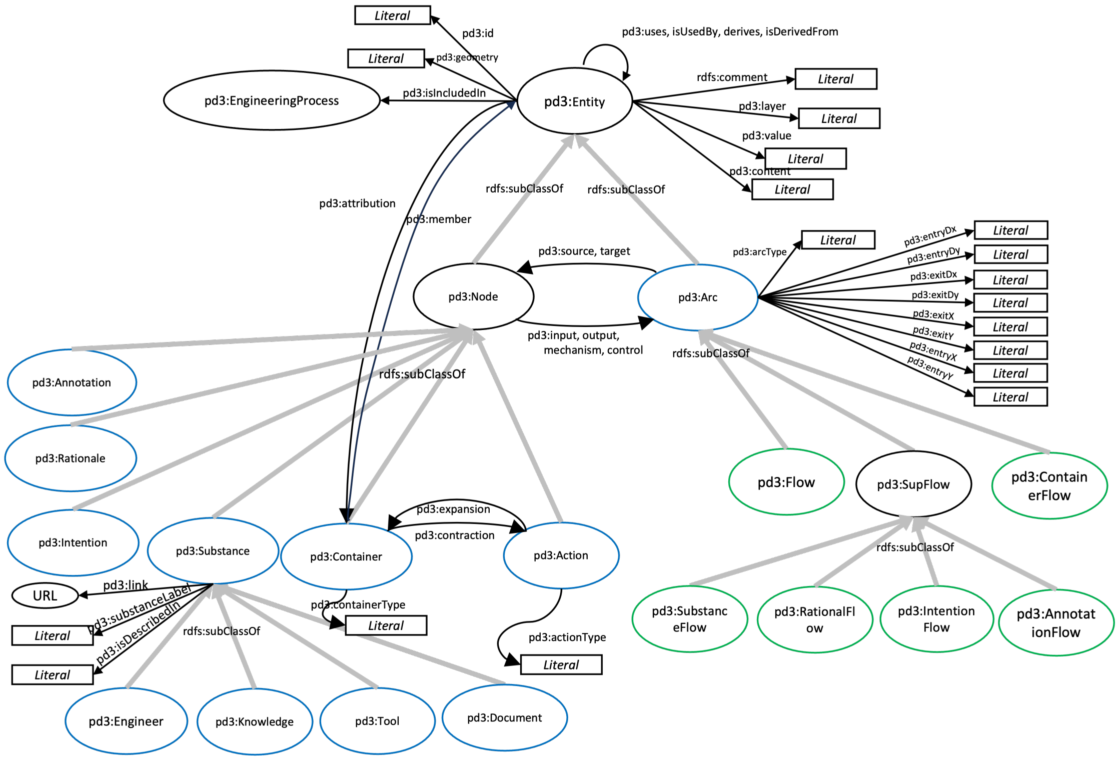


Figure 4: PD3 Ontology

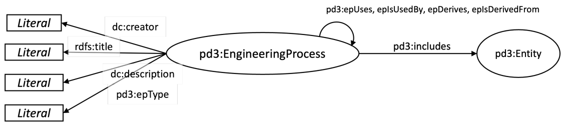


Figure 5: PD3 Meta Ontology (Ontology for Engineering Process)

corresponding to the diagrammatic model in the left panel in the window (see Figure 7. When the element in the library is used, information necessary for PD3 elements is automatically embedded in data in draw.io. The right panel of the windows is provided for metadata of the engineering process. Metadata including title and creator can be specified in the right panel in the window. The base URI can be also specified which is used by RDF converting, otherwise, a unique URI is automatically generated. It is important for uniqueness and accessibility when PD3 data is published as RDF. The data created with draw.io is stored as XML or compressed XML. The specific information for PD3 is also included in draw.io generic XML.

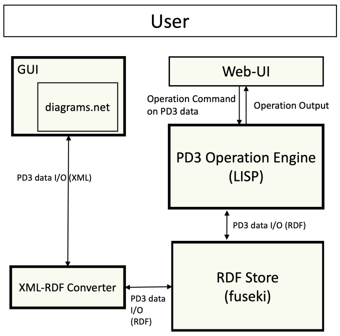


Figure 6: The overview of PD3 Platform Prototype

4.2. The Conversion System between XML and RDF

XML data generated with draw.io is converted to RDF data and vice visa by using PD3 ontology. The most of conversion is done straightforwardly, .i.e, elements in XML are converted to RDF statements one by one. There are some notions in the conversion process. The base URI is given by the user or generated automatically. In the latter case, a unique ID string for XML data is used as a part of URI. URI for each element is also generated from ID in XML elements. Thus uniqueness of the URI is primarily preserved. Geometrical information such as position of a graphical element is encoded and attached as a value of Property "geometry". The information is decoded when RDF-to-XML conversion is executed. Thus geometry information is preserved. The conversion program also checks the inconsistency of data and adjusts it if possible. The system expects users to use the elements in PD3 library but they may add elements with the original drawing functions of draw.io. In such cases, information necessary for PD3 elements is missing. For example, s/he draws an arrow from one Action to the other by the drawing function. The information about arrow type and layer is missing. The conversion system infers and adds such information by referring to information on the connected actions.

4.3. The Inference System

The inference system can read RDF data (turtle) and make some different types of inference. One example of inference is tracing the engineering process. In the inference, a sequence of actions is generated with parameter assignment, i.e., one action is invoked and parameters are changed, then the other action is invoked and so on. As mentioned, there are "if/else" and iterations by Container. In such cases, The system is implemented by LISP.

4.4. The database system

Apache Jena Fuseki[2] is adopted as the database system for PD3. It is just standard use of Jena. All PD3 data are stored and accessed with unique base URIs that are assigned to individual engineering processes. The relationship like revision and variation of engineering processes is also stored and accessed as the RDF statements including IDs for target and source engineering

[2] https://jena.apache.org/documentation/fuseki2/

processes that are generated from their base URIs. Thus, an RDF database works as that both for engineering processes themselves and for their relations.

4.5. An example

An artificial small example is shown in Figure 8 that demonstrates how a PD3 process is represented, converted, and used for inference. The PD3 process is a calculation of parameters with if and loop conditions. Note that such a numeric calculation is not the main function of PD3 but is represented as PD3 to show how the execution is realized as the inference.

In figure 8(a), the PD3 process is represented as the diagrammatic model. There are five actions including ”Start” and ”End”. There are five flows connecting the actions. When an action is executed, the content of the action is evaluated and then another action connected by Information Flow is called and evaluated. Note that the content of an action is only visible on mouse-over but is shown as a box below in the figure for convenience. The converted PD3 data is shown in Figure 8(b) where only prefixes and an Action (Action1) are extracted.

In the example, The process starts with ”Start” Acton assigning some parameters, and then go to Action1 and Action2 in order. The output of Action2 has a choice. One choice is to to Action3 and the other is to go back to Action1. It is determined by the value of a parameter. Finally, the process stops at ”End” Action. Figure 8(c) shows how the inference system can execute the given PD3 process as mentioned above.

5. Conclusion

Based on the innovative Digital Triplet (D3) Concept depicted in the novel, this paper introduces a process modelling language for D3, referred to as PD3. Furthermore, a prototype system is developed to validate the practical application of PD3. Notably, the foundation of this system’s construction hinges upon a semantic approach, prominently featuring the PD3 Ontology and RDF-based data derived from the said ontology. The intrinsic value of the ontology lies in its ability to establish a seamless connection between the graphical and semantic models, all while preserving the essential flexibility of the modelling process. In contrast to XML, RDF stands out for its exceptional capacity to enable data reuse, even when confronted with modifications to the underlying model. The establishment of identity is achieved through the allocation of a distinct Uniform Resource Identifier (URI) for each engineering process, thereby facilitating straightforward data publication and accessibility. RDF further supports versatile data utilization, accommodating scenarios such as the integration of data from multiple engineering processes. Moreover, the RDF framework lends itself to the facilitation of inference through the utilization of the logical structure inherent in the ontology. Though the prototype system remains a work in progress, its ongoing development seeks to unlock the full potential of PD3 data utilization, delving into more intricate applications. Foreseen as an integral component, this system is anticipated to be seamlessly integrated into a broader platform responsible for the comprehensive management of engineering process information, synergizing with AI-driven technologies.

Acknowledgments

This research is based on results obtained from a project, JPNP18002, commissioned by the New Energy and Industrial Technology Development Organization (NEDO). We are grateful to all of the project members, in particular, Jun Ota (University of Tokyo) as the project leader, Masahiro Nakamura (Lexer Research) for advice on implementation, and Toshinori Yasui (Denso Corporation) for advice on engineering knowledge.

References

[1] L. Monostori, B. Kádár, T. Bauernhansl, S. Kondoh, S. Kumara, G. Reinhart, O. Sauer, G. Schuh, W. Sihn, K. Ueda, Cyber-physical systems in manufacturing, CIRP Annals 65 (2016) 621–641. doi:https://doi.org/10.1016/j.cirp.2016.06.005.

[2] F. Tao, Q. Qi, L. Wang, A. Nee, Digital twins and cyber–physical systems toward smart manufacturing and industry 4.0: Correlation and comparison, Engineering 5 (2019) 653–661. doi:https://doi.org/10.1016/j.eng.2019.01.014.

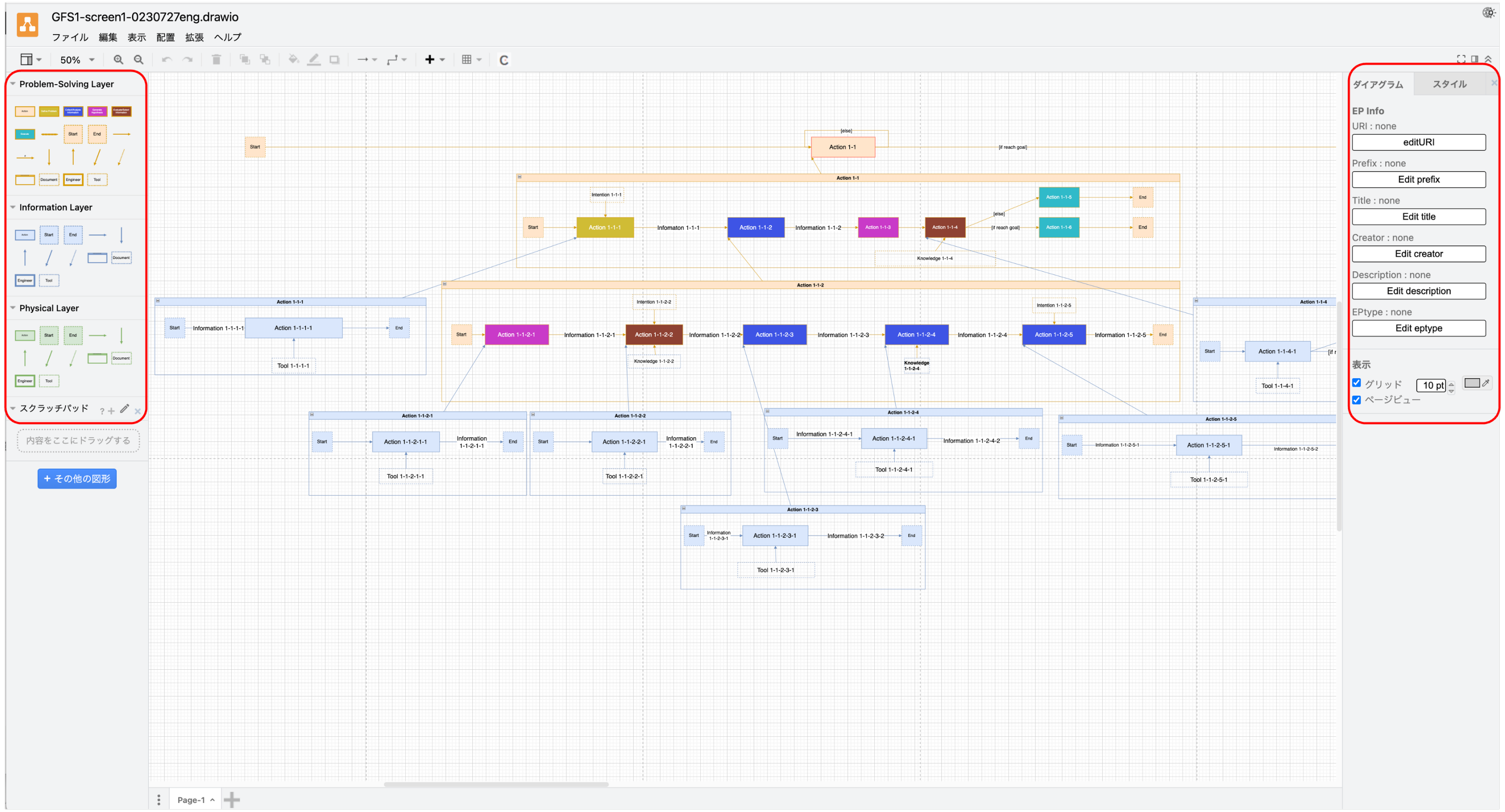
[3] B. Hadorn, M. Courant, B. Hirsbrunner, M. Courant, B. Hirsbrunner, Towards human-centered cyber-physical systems: a modeling approach, Départment d'informatique Université de Fribourg, 2016.

[4] F. Ansari, M. Khobreh, U. Seidenberg, W. Sihn, A problem-solving ontology for human-centered cyber physical production systems, CIRP Journal of Manufacturing Science and Technology 22 (2018) 91–106.

[5] Y. Umeda, Y. Hongo, J. Goto, S. Kondoh, Digital triplet and its implementation on learning factory, IFAC-PapersOnLine 55 (2022) 1–6. doi:https://doi.org/10.1016/j.ifacol.2022.04.160, 14th IFAC Workshop on Intelligent Manufacturing Systems IMS 2022.

[6] Y. Umeda, J. Ota, S. Shirafuji, F. Kojima, M. Saito, H. Matsuzawa, T. Sukekawa, Exercise of digital kaizen activities based on 'digital triplet' concept, Procedia Manufacturing 45 (2020) 325–330. doi:https://doi.org/10.1016/j.promfg.2020.04.025, learning Factories across the value chain – from innovation to service – The 10th Conference on Learning Factories 2020.

[7] M. N. Lakhoua, B. Salem, L. El Amraoui, The need for system analysis based on two structured analysis methods sadt and sa/rt, ACTA TECHNICA CORVINIENSIS–Bulletin of Engineering 11 (2018) 113–118.



EP Info

URI : none

Prefix : none

Title : none

Creator : none

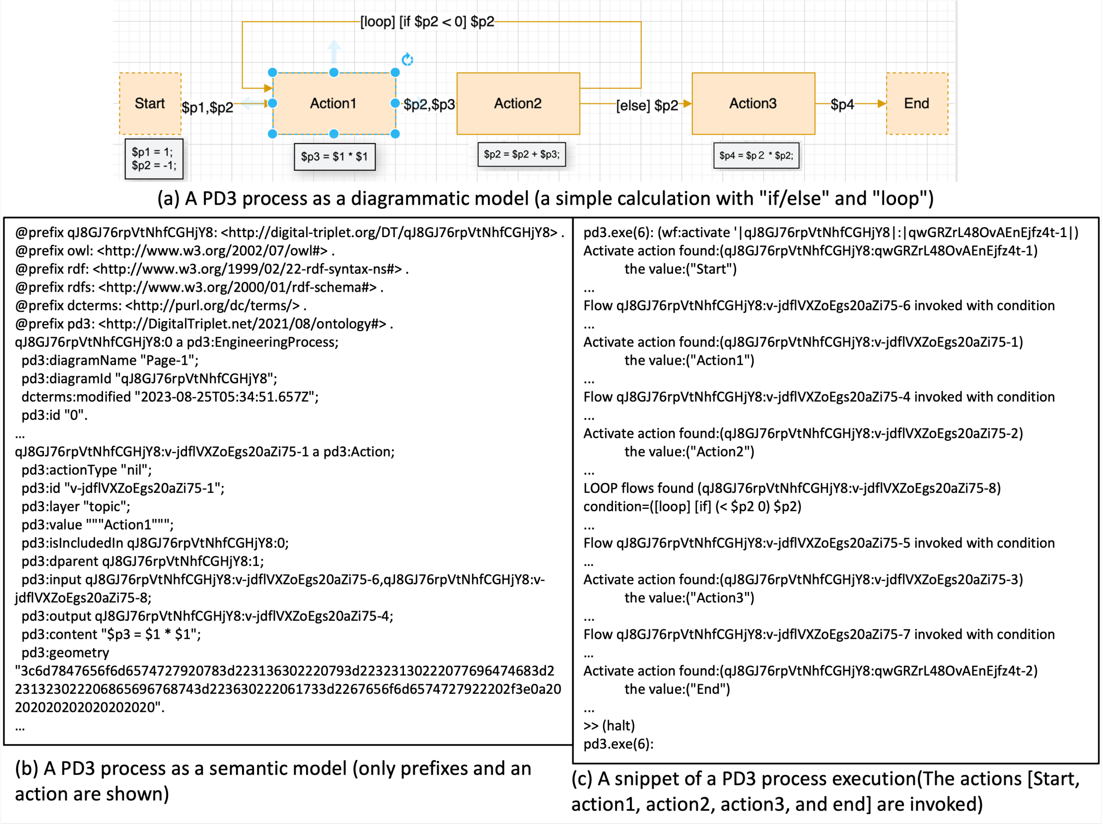
Description : none

EPtype : none

☑ グリッド 10 pt

☑ ページビュー

Figure 7: A snapshot of the interface system



(a) A PD3 process as a diagrammatic model (a simple calculation with "if/else" and "loop")

```
@prefix qJ8GJ76rpVtNhfCGHjY8: <http://digital-triplet.org/DT/qJ8GJ76rpVtNhfCGHjY8> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix pd3: <http://DigitalTriplet.net/2021/08/ontology#> .
qJ8GJ76rpVtNhfCGHjY8:0 a pd3:EngineeringProcess;
  pd3:diagramName "Page-1";
  pd3:diagramId "qJ8GJ76rpVtNhfCGHjY8";
  dcterms:modified "2023-08-25T05:34:51.657Z";
  pd3:id "0".
...
qJ8GJ76rpVtNhfCGHjY8:v-jdflVXZoEgs20aZi75-1 a pd3:Action;
  pd3:actionType "nil";
  pd3:id "v-jdflVXZoEgs20aZi75-1";
  pd3:layer "topic";
  pd3:value """Action1""";
  pd3:isIncludedIn qJ8GJ76rpVtNhfCGHjY8:0;
  pd3:dparent qJ8GJ76rpVtNhfCGHjY8:1;
  pd3:input qJ8GJ76rpVtNhfCGHjY8:v-jdflVXZoEgs20aZi75-6,qJ8GJ76rpVtNhfCGHjY8:v-
jdflVXZoEgs20aZi75-8;
  pd3:output qJ8GJ76rpVtNhfCGHjY8:v-jdflVXZoEgs20aZi75-4;
  pd3:content "$p3 = $1 * $1";
  pd3:geometry
"3c6d7847656f6d6574727920783d223136302220793d2232313022207769647468 3d2
231323022206865696768743d223630222061733d2267656f6d6574727922202f3e0a20
202020202020202020".
...
```

(b) A PD3 process as a semantic model (only prefixes and an action are shown)

```
pd3.exe(6): (wf:activate '|qJ8GJ76rpVtNhfCGHjY8|:|qwGRZrL48OvAEnEjfz4t-1|)
Activate action found:(qJ8GJ76rpVtNhfCGHjY8:qwGRZrL48OvAEnEjfz4t-1)
        the value:("Start")
...
Flow qJ8GJ76rpVtNhfCGHjY8:v-jdflVXZoEgs20aZi75-6 invoked with condition
...
Activate action found:(qJ8GJ76rpVtNhfCGHjY8:v-jdflVXZoEgs20aZi75-1)
        the value:("Action1")
...
Flow qJ8GJ76rpVtNhfCGHjY8:v-jdflVXZoEgs20aZi75-4 invoked with condition
...
Activate action found:(qJ8GJ76rpVtNhfCGHjY8:v-jdflVXZoEgs20aZi75-2)
        the value:("Action2")
...
LOOP flows found (qJ8GJ76rpVtNhfCGHjY8:v-jdflVXZoEgs20aZi75-8)
condition=([loop] [if] (< $p2 0) $p2)
...
Flow qJ8GJ76rpVtNhfCGHjY8:v-jdflVXZoEgs20aZi75-5 invoked with condition
...
Activate action found:(qJ8GJ76rpVtNhfCGHjY8:v-jdflVXZoEgs20aZi75-3)
        the value:("Action3")
...
Flow qJ8GJ76rpVtNhfCGHjY8:v-jdflVXZoEgs20aZi75-7 invoked with condition
...
Activate action found:(qJ8GJ76rpVtNhfCGHjY8:qwGRZrL48OvAEnEjfz4t-2)
        the value:("End")
...
>> (halt)
pd3.exe(6):
```

(c) A snippet of a PD3 process execution(The actions [Start, action1, action2, action3, and end] are invoked)

Figure 8: A Running Example of PD3 Process