15º Congresso de Inovação, Ciência e Tecnologia do IFSP - 2024

MECANISMO DE IMPLEMENTAÇÃO DE BLOCOS FUNCIONAIS O-PAS DEFINIDOS PELO USUÁRIO

Guilherme S. da Silveira[1], Rodrigo P. Pantoni [2]

[1]Graduando em Engenharia Elétrica no IFSP, Campus Sertãozinho, Bolsista IT/IFSP, silveira.s@aluno.ifsp.edu.br.
[2]Professor Doutor do IFSP, Campus Sertãozinho, rpantoni@ifsp.edu.br.

Área de conhecimento (Tabela CNPq): 3.04.05.02-5 Automação Eletrônica de Processos Elétricos e Industriais.

RESUMO: O padrão O-PAS™ vem sendo especificado desde 2016, com o objetivo de levar os sistemas de automação de processos industriais rumo à chamada Indústria 4.0. O "coração" de um sistema de controle e automação em conformidade com o padrão O-PAS™ consiste em um conjunto de blocos funcionais (O-PAS™ function blocks) padronizados, cada um realizando cálculos específicos. Mais do que isso, a especificação O-PAS permite que o usuário do sistema possa programar um bloco funcional, de modo a resultar em blocos funcionais definidos pelo usuário. Neste sentido, o objetivo geral do projeto é o desenvolvimento de um mecanismo de implementação de blocos funcionais O-PAS™ definidos pelo usuário, que incluem o algoritmo interno, a definição dos parâmetros de entrada, saída e de configuração e, por fim, o mapeamento dos parâmetros com as variáveis do algoritmo. A metodologia contou com a implemetação de um PID simplificado como exemplo. Os resultados obtidos permitem que os arquivos resultantes do referido bloco funcional seja portável para qualquer outro sistema de automação industrial que segue o padrão O-PAS™.

PALAVRAS-CHAVE: *Blocos funcionais definidos pelo usuário*; *O-PAS*; *Modelo de informação*; *IEC 61131-3*.

USER DEFINED O-PAS FUNCTION BLOCK IMPLEMENTATION MECHANISM

ABSTRACT: The O-PAS™ standard has been under specification since 2016, with the aim of guiding industrial process automation systems towards the so-called Industry 4.0. The "core" of a control and automation system compliant with the O-PAS™ standard consists of a set of standardized function blocks (O-PAS™ function blocks), each performing specific calculations. Furthermore, the O-PAS specification allows the system user to program a function block, resulting in user-defined function blocks. In this context, the overall goal of the project is to develop a mechanism for implementing user-defined O-PAS™ function blocks, which include the internal algorithm, the definition of input, output, and configuration parameters, and finally, the mapping of parameters to the algorithm's variables. The methodology included the implementation of a simplified PID as an example. The results obtained ensure that the resulting function block files are portable to any other industrial automation system that follows the O-PAS™ standard.

KEYWORDS: *User-defined function blocks*; *O-PAS*; *Information model*; *IEC 61131-3*.

INTRODUÇÃO

O padrão O-PAS™ para sistemas de automação industrial está em desenvolvimento desde 2016 e tem como intuito ser o padrão dos padrões, pois define uma arquitetura aberta, segura e interoperável (Qamsane *et al.*, 2022). Apesar de promissor, o padrão ainda possui lacunas em sua especificação que inviabilizam sua implementação prática no contexto deste trabalho, como observado nas partes 6.2 (The Open Group, 2023a) e 6.6 (The Open Group, 2023b) da norma O-PAS™.

Neste sentido, o problema técnico-científico deste trabalho contempla o desafio de transpor as especificações da norma O-PAS™ em um mecanismo funcional específico proposto pela empresa financiadora deste projeto. Com isso, este trabalho poderá ser utilizado como referência para a padronização, uma vez que não existem informações suficientes para que o mecanismo funcione na prática.

Mais do que isso, embora não explicitamente declarado no conjunto de especificações O-PAS™, os padrões atualmente estabelecidos permitem que os usuários programem seus blocos funcionais em vez de usar apenas os blocos funcionais padronizados, especificando entradas, saídas, parâmetros de supervisão/configuração e lógica de processamento. Portanto, ao aderir ao padrão O-PAS™, o bloco funcional definido pelo usuário (UDFB) desenvolvido poderia ser portado por qualquer outro sistema de automação industrial.

Nesse contexto, este trabalho tem como objetivo projetar e desenvolver um mecanismo de software para criar um UDFB segundo a especificação do padrão O-PAS™. Os objetivos específicos são: 1) programar a lógica interna de um PID simplificado em texto estruturado da IEC 61131-3 (International Electrotechnical Commission, 2013), 2) definir os parâmetros de entrada, saída e de supervisão/configuração e 3) realizar a ligação lógica ou associação dos parâmetros com a lógica programada.

METODOLOGIA

A metodologia deste trabalho é detalhada com o alinhamento aos objetivos específicos estabelecidos. Assim, em relação ao objetivo específico de número 1, programar a lógica interna do bloco com Texto Estruturado da IEC 61131-3, foi utilizada a ferramenta OPEN PLC Editor (Autonomy, 2022).

A especificação define que a lógica de programação deve ser realizada com Texto Estruturado (ST) da IEC 61131-3. No entanto, isso não implica exclusividade no desenvolvimento dessa linguagem. A IEC 61131-3 define cinco linguagens em sua especificação (Texto Estruturado (ST), Lista de Instruções (IL), Diagrama de Blocos (FBD), Linguagem Ladder (LD) e Diagrama de Fluxo (SFC)) e existem ferramentas disponíveis com tradutores que convertem qualquer linguagem definida na IEC 61131-3 para ST.

Por exemplo, o tradutor de código aberto Matiec (2001) é utilizado pela ferramenta também de código aberto OPEN PLC Editor. Com essa ferramenta, os usuários podem usar qualquer uma das cinco linguagens ou uma combinação delas, que ao final, serão traduzidas em código ST. Contudo, há várias outras ferramentas capazes de desenvolver lógica na linguagem ST, como CODESYS (2024), Siemens SIMATIC STEP 7 (2024) etc.

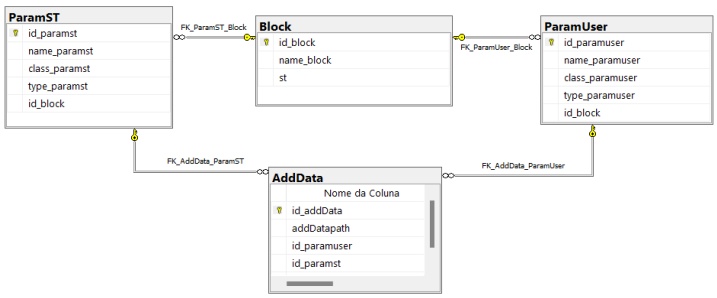
Em relação aos objetivos específicos de número 2 e 3, definir os parâmetros de entrada, saída, de supervisão/configuração e realizar a ligação lógica ou associação dos parâmetros com a lógica programada, foi preciso desenvolver uma ferramenta para geração dos arquivos necessários de acordo com o padrão que representam o UDFB. Essa ferramenta foi chamada de "O-PAS™ Function Block Builder".

Para o desenvolvimento desta ferramenta utilizou-se o IDE (ambiente de desenvolvimento integrado) Visual Studio (Microsoft, 2024) e a linguagem de programação C#. Ademais, utilizou-se o banco de dados relacional SQLServer (Microsoft, 2022) para o armazenamento das informações dos UDFB criados, permitindo que fossem acessados posteriormente para edição, consulta ou exclusão.

O modelo do banco de dados desenvolvido para poder persistir as informações contidas ao atendimento dos objetivos 2 e 3 é apresentado na figura 1, onde observa-se a existência de quatro tabelas: *Block*, *ParamUser*, *ParamST* e *AddData*, em todas existe um campo para identificação único que começa com o prefixo *id* e termina com o respectivo nome, esse campo também é utilizado para associar as diferentes tabelas.

A primeira tabela serve para o armazenamento dos dados de um bloco, tendo como elementos o nome do bloco (*name_ block*) e o código (st). A segunda tabela armazena os parâmetros criados pelo usuário e é preenchida com o nome do parâmetro (*name_ paramuser*), sua classe (*class_ paramuser*) de acordo com a norma 61131-3 (International Electrotechnical Commission, 2013), seu tipo (*type_ paramuser*) de acordo com a norma O-PAS™ (The Open Group, 2023a) e por fim o id de um bloco (*id_ block*) a quem o parâmetro está associado. A terceira tabela é análoga à segunda, porém serve para armazenar os dados das variáveis oriundas do código st, além disso seu campo *type_ paramst* armazena seu tipo de acordo com o texto estruturado, e não de acordo com a norma O-PAS™. Por fim, a última tabela associa um elemento da tabela *ParamUser* com outro da tabela *ParamST* tendo, portanto, dois campos de *id*, cada um referente a uma dessas tabelas, além disso possui um campo que armazena o caminho da associação (*addDatapath*).

Figura 1: Modelo do banco de dados desenvolvido no SQLServer



Assim, nesta ferramenta desenvolvida, foi criado um mecanismo para criar os parâmetros O-PAS™ do UDFB, chamado de "Define Parameters", onde o usuário pode selecionar se o parâmetro é de entrada, saída ou de configuração e ou supervisão. Além disso, ele deve escolher o tipo, que pode ser booleano (digital) ou *float* (analógico). Ao final do processo, a ferramenta exporta as informações dos parâmetros definidos no formato XML chamado de *Nodeset* especificado nos padrões OPC UA (Open62541, 2024) e O-PAS™ (The Open Group, 2023b).

A implementação da escrita e leitura dos parâmetros do arquivo nodeset.xml foi realizada com a biblioteca CESMII NodeSet Utilities, disponível em (CESMII, 2024). CESMII NodeSet Utilities é um conjunto de bibliotecas .Net para leitura, validação, manipulação e criação de *NodeSets* OPC UA. Ele é construído com base na biblioteca OPCFoundation.NetStandard.Opc.Ua.Core da OPC Foundation. Além disso, a validação do *NodeSet* foi realizada com um Servidor OPC UA real (Open62541, 2024).

Em relação ao objetivo específico de número 3, foi implementada na ferramenta a seleção para associar os parâmetros definidos e as variáveis da lógica no padrão IEC 61131-3, e o produto final desta operação é um arquivo de formato XML chamado de *AddData* (The Open Group, 2023b).

RESULTADOS E DISCUSSÃO

Assim, em relação ao objetivo específico de número 1, foi implementada uma lógica de programação de um UDFB PID (proporcional-integral-derivativo) simplificado. O controlador ajusta um valor de controle (CV) com base no erro entre o valor desejado (SP) e o valor atual do processo (PV). Ele calcula os componentes proporcional, integral e derivativo do erro para ajustar o sinal de controle de maneira a minimizar a diferença entre o SP e PV. O bloco também verifica se o sinal de entrada está em estado "bom" antes de executar os cálculos e atualiza diversas propriedades de status, nível de acesso, intervalo e unidades de engenharia do sinal de saída para refletir as características do sinal de entrada. O programa foi encapsulado num bloco funcional da IEC 61131-1, cuja programação é apresentada no código 1.

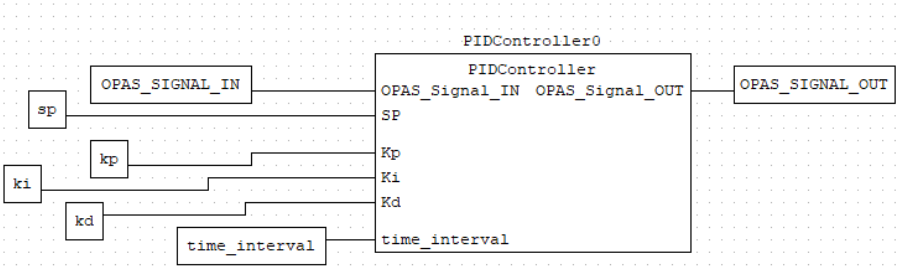
Código 1: Programa na linguagem ST do PID simplificado

```
IF (OPAS_Signal_IN.status AND 16#c0000000) = 16#00000000 THEN
    error_value := SP - PV;
    CV := Kp * error_value;
    integral_value := integral_value + (Ki * error_value * time_interval);
    CV := CV + integral_value;
    derivative_value := (error_value - e_prev) / time_interval;
    CV := CV + (Kd * derivative_value);
    e_prev := error_value;
END_IF;

OPAS_Signal_OUT.status := OPAS_Signal_IN.status;
OPAS_Signal_OUT.accessLevel := OPAS_Signal_IN.accessLevel;
OPAS_Signal_OUT.euRange := OPAS_Signal_IN.euRange;
OPAS_Signal_OUT.engineeringUnits := OPAS_Signal_IN.engineeringUnits;
OPAS_Signal_OUT.timeStamp := OPAS_Signal_IN.timeStamp;
```

O programa principal (myPOU) é mostrado na figura 2. Foi criado um POU na linguagem FBD, que possui como entrada e saída sinais O-PAS do tipo real definidos pela estrutura “AnalogUnitRange”, de acordo com a especificação O-PAS, que possui *status*, *timeStamp*, *value* e *accessLevel*, *euRange* e *EngineeringUnits*. Além disso, possui cinco variáveis do tipo real para parâmetros de configuração e/ou supervisão (kp, ki, kd, sp e time_interval). O intuito desta aplicação é de projetar um controlador PID simples em ST, que exige que sejam empregados valores de sinais e de variáveis do tipo Real.

Figura 2: Lógica do bloco funcional em FBD



Internamente, ao compilar a lógica desenvolvida, o OPEN PLC Editor gera um arquivo de extensão ST, que contém o código traduzido, conforme mostrado no código 2.

Código 2: Programa na linguagem ST do PID simplificado

```
FUNCTION_BLOCK PIDController
  (*Declaracao das variaveis do Codigo 1*)
  (*Codigo 1, citado anteriormente*)
END_FUNCTION_BLOCK

PROGRAM myPOU
  VAR_INPUT
    OPAS_SIGNAL_IN : AnalogUnitRange;
  END_VAR
  VAR
    kp : REAL;
```

```
    ki : REAL;
    kd : REAL;
    sp : REAL;
    time_interval : REAL;
  END_VAR
  VAR_OUTPUT
    OPAS_SIGNAL_OUT : AnalogUnitRange;
  END_VAR
  VAR
    PIDController0 : PIDController;
  END_VAR

  PIDController0(OPAS_Signal_IN := OPAS_SIGNAL_IN, SP := sp, Kp := kp, Ki := ki, Kd := kd, time_interval :=
    time_interval);
  OPAS_SIGNAL_OUT := PIDController0.OPAS_Signal_OUT;
END_PROGRAM
```

Os resultados consistem em uma descrição dos arquivos xml *NodeSet* e *AddData*, de acordo com as informações citadas na seção de Metodologia. Assim, em relação ao objetivo específico de número 2, o código 3 apresenta dois trechos do *NodeSet* onde está a declaração do parâmetro O-PAS™ do sinal de entrada "input".

Código 3: Trecho do *NodeSet* onde está a declaração do sinal de entrada "input"

```
<UAVariable NodeId="ns=1;i=1003" BrowseName="1:input" DataType="Number">
<DisplayName>input</DisplayName>
<References>
<Reference ReferenceType="HasTypeDefinition">OPASAnalogSignalType</Reference>
<Reference ReferenceType="HasModellingRule">i=78</Reference>
</References>
</UAVariable>
```

Em relação ao objetivo específico de número 3, realizar a ligação lógica ou associação dos parâmetros com a lógica programada, o Código 4 apresenta um trecho do *AddData* onde está a associação do parâmetros O-PAS™ do sinal de entrada (input.ActualValue) do UDFB com a variável da IEC 61131-4 `OPAS_SIGNAL_IN`.

Código 4: trecho do *AddData* onde está a associação do parâmetros O-PAS™ do sinal de entrada (input.ActualValue)

```
<InputVars>
<Variable name="OPAS_SIGNAL_IN" orderWithinParamSet="1">
<Type>
<TypeName>AnalogUnitRange</TypeName>
</Type>
<AddData>
<Data name="http://www.opengroup.org/open-process-automation/O-PAS/61131-10/schema">
<UAObjectVariableLink permittedAccessLevel="3" permittedAccessLevelScope="VariableAndProperties">
<BrowsePathSegment nameSpaceURI="https://opcua.rocks/UA">
<BrowsePathSegmentName>input</BrowsePathSegmentName>
</BrowsePathSegment>
<BrowsePathSegment nameSpaceURI="https://www.opengroup.org/open-process-automation/O-PAS/Base">
<BrowsePathSegmentName>ActualValue</BrowsePathSegmentName>
</BrowsePathSegment>
</UAObjectVariableLink>
</Data>
</AddData>
</Variable>
</InputVars>
```

CONCLUSÕES

Este trabalho apresentou um mecanismo com projeto, implementação e validação utilizando tecnologias previstas para o padrão O-PAS™ para criar um UDFB.

A principal lacuna do padrão O-PAS™ é a falta de exemplos reais e funcionais na especificação, como observado nas partes 6.2 (The Open Group, 2023a) e 6.6 (The Open Group, 2023b) da norma O-PAS™. Portanto, este trabalho contribuiu ao fornecer um exemplo completo de um UDFB, desde a

sua concepção e implementação, passando pela seleção de tipos e pela estrutura dos arquivos XML, utilizando ferramentas existentes para programação em IEC 61131-3, desenvolvendo uma nova ferramenta e validando o modelo de informação via um servidor OPC UA.

CONTRIBUIÇÕES DOS AUTORES

Rodrigo P. Pantoni contribuiu com a concepção, orientação e estudo de viabilidade do projeto. Guilherme S. da Silveira, por outro lado, desenvolveu o projeto em sua implementação, realização e testes completos. Assim, Rodrigo P. Pantoni e Guilherme S. da Silveira contribuíram com a curadoria e análise dos resultados e procederam com a metodologia e experimentos. Todos os autores contribuíram com a revisão do trabalho e aprovaram a versão submetida.

AGRADECIMENTOS

Os autores agradecem à empresa Nova Smar pela colaboração e apoio no desenvolvimento desse projeto (APPDI - PD&I Nº 05/2023 - IFSP e Nova Smar S.A.).

REFERÊNCIAS

AUTONOMY. *OpenPLC Editor 2.01*. 2022. <https://autonomylogic.com/>. Acesso em: 26/03/2024.

CESMII. *CESMII NodeSet Utilities*. 2024. <https://github.com/cesmii/CESMII-NodeSet-Utilities>. Accessed: 16 Feb 2024.

CODESYS GROUP. *CODESYS Group*. 2024. <https://www.codesys.com>. Accessed: 26 de mar. 2024.

INTERNATIONAL ELECTROTECHNICAL COMMISSION. *IEC 61131-3: Industrial automation systems - Programmable controllers - Part 3: Programming languages*. [S.l.], 2013. Disponível em: <https://webstore.iec.ch/publication/6182>.

MATIEC. *IEC 61131-3 Compiler*. 2001. Https://github.com/nucleron/matiec. Acesso em: 26/06/2024.

MICROSOFT. *SQL Server 16.0.1000*. 2022. <https://www.microsoft.com/pt-br/sql-server/sql-server-downloads>. Acesso em: 15/04/2024.

MICROSOFT. *Visual Studio Community 2022 17.9*. 2024. <https://visualstudio.microsoft.com/vs/community/>. Acesso em: 17/04/2024.

OPEN62541. *Open Source OPC UA licensed under the MPL v2.0*. 2024. <https://www.open62541.org>. Accessed: 22 April 2024.

QAMSANE *et al.* Open process automation- and digital twin-based performance monitoring of a process manufacturing system. *IEEE Access*, v. 10, p. 60823–60835, 2022.

SIEMENS SIMATIC STEP 7. *Siemens SIMATIC STEP 7*. 2024. <https://www.siemens.com/global/en/products/automation/industry-software/automation-software/tia-portal/software/step7-tia-portal.html>. Accessed: 01 april 2024.

THE OPEN GROUP. *O-PAS™ Standard, Version 2.1: Part 6.2 – Information and Exchange Models: Basic Configuration*. Apex Plaza, Forbury Road, Reading, Berkshire, RG1 1AX, Reino Unido, 2023.

THE OPEN GROUP. *O-PAS™ Standard, Version 2.1: Part 6.6 – Information and Exchange Models: IEC 61131*. Apex Plaza, Forbury Road, Reading, Berkshire, RG1 1AX, Reino Unido, 2023.