

**DESENVOLVIMENTO DE SOFTWARE PARA EXECUTAR E/OU SIMULAR A  
EXECUÇÃO DE SISTEMAS BASEADOS EM BLOCOS DE FUNÇÕES  
ADERENTES A IEC 61499**

Pedro Henrique Flenik, André Bittencourt Leal, Herbert Albérico de Sá Leitão, Yuri  
Kaszubowski Lopes, Roberto Silvio Ubertino Rosso Junior

**INTRODUÇÃO**

Com o aumento da automação nos mais diversos setores da indústria, surge a necessidade do desenvolvimento de formas de controlar e organizar as entidades que fazem parte de uma linha de produção. Com isso surgiram diversas tecnologias que possibilitam controlar uma linha de produção, porém o uso delas frequentemente requer um conhecimento mais aprofundado sobre programação. Nesse contexto, surge a norma IEC 61499 [1], [2], [3] que facilita a programação do controle de linhas de produção com a introdução de blocos de função que proporcionam uma modelagem orientada a objetos que suporta interoperabilidade, portabilidade e configurabilidade, características importantes para uma arquitetura utilizada em sistemas de automação distribuídos. Além disso, os algoritmos implementados nos blocos de função podem ser escritos em diversas linguagens como texto estruturado e Ladder, conhecidas na indústria, ou ainda em linguagens populares de propósito geral, como C e Java, entre outras, o que garante maior flexibilidade no momento da implementação. Os blocos de função e seus algoritmos são armazenados em arquivos XML (*eXtensible markup language*), o que é outra vantagem por ser um formato de arquivo definido pela W3C, com um padrão já estabelecido. Após a criação da norma diversas implementações surgiram para o funcionamento dos blocos de função, como o uso da máquina virtual Java para realizar a execução de uma aplicação de controle, porém a JVM pode acabar consumindo muitos recursos do hardware de pequeno porte presente em uma linha de produção, com isso surge a solução proposta por [4], que cria uma máquina virtual pequena capaz de fazer a execução de uma aplicação de controle em um hardware com poucos recursos de processamento e memória.

**DESENVOLVIMENTO**

Esse desenvolvimento tem como base o trabalho de mestrado desenvolvido por Pinto[5] e o artigo dele derivado [4], portanto após observação e entendimento do ambiente de execução que nele foi desenvolvido, o primeiro passo do desenvolvimento foi a procura de atualização nas tecnologias que foram utilizadas. A primeira atualização feita foi a adaptação dos códigos implementados em Python. O código foi originalmente desenvolvido com Python 2 e não podia ser utilizado por interpretadores mais recentes que utilizam o Python 3, assim foi feita uma refatoração nos códigos de Python 2 adaptando para a sintaxe do Python 3, além da mudança na sintaxe a forma da comunicação por soquetes também foi adaptada para funcionar com a nova versão do Python. Indicar a metodologia utilizada de modo que o leitor entenda os procedimentos utilizados no desenvolvimento do trabalho de pesquisa proposto pelo orientador.

O segundo passo foi levar em consideração possíveis mudanças na norma quanto a definições dos blocos de função. Para resolver isso foram feitas mudanças no pré-processador do ambiente de execução, pois ao fazer com que os arquivos XML sejam corretamente transformados em arquivos de texto estruturados, é possível garantir a compilação correta desses para o código de máquina da máquina virtual e portanto o funcionamento correto do ambiente de execução.

## RESULTADOS

Após realizar as mudanças no código em Python e no pré-processador as alterações foram testadas. Nos testes foram realizadas transformações, utilizando a nova versão do pré-processador, de arquivos XML de blocos de função desenvolvidos no 4diac, em arquivos de texto estruturados. O 4diac é um ambiente de desenvolvimento *open source* aderente à IEC 61499. A partir disso, também foi possível realizar a compilação correta para o código da máquina virtual do ICARU-FB.

## CONSIDERAÇÕES FINAIS

As modificações realizadas no ambiente de execução possibilitaram a atualização do código legado para Python 3 e a adaptação do pré-processador, garantindo a correta transformação de blocos de função definidos no 4diac para a máquina virtual do ICARU-FB. Dessa forma, foi possível restabelecer o funcionamento do sistema em plataformas de hardware simples.

Entretanto, algumas limitações permanecem. O pré-processador ainda apresenta restrições ao lidar com implementações de algoritmos escritos em diferentes linguagens como C, linguagem usada pela indústria para implementar o algoritmo definido no XML, o que limita a flexibilidade do ambiente, já que atualmente o pré-processador aceita apenas algoritmos definidos em texto estruturado. Além disso, o compilador não possui suporte a tipos de dados complexos, como data e tempo, que podem aparecer nas definições de blocos de função. Essa limitação, no entanto, exige reflexão, pois ampliar o suporte a tipos complexos pode aumentar a utilidade do ambiente, mas também pode comprometer sua principal característica de ser leve e adequado a dispositivos de recursos reduzidos. Uma solução para esse dilema poderá ser alvo de trabalhos futuros.

### Palavras-chave:

IEC 61499, hardware de baixo custo, pré-processador, automação industrial.

## REFERÊNCIAS BIBLIOGRÁFICAS

- [1] IEC 61499-1. **Function blocks - Part 1: Architecture**. Geneva: International Electrotechnical Commission, nov. 2012.
- [2] IEC 61499-2. **Function blocks - Part 2: Software and Tool Requirements**. Geneva: International Electrotechnical Commission, nov. 2012. Constar apenas autores e obras mencionados no texto, obedecendo-se às normas da ABNT.
- [3] IEC 61499-4. **Function blocks - Part 4: Rules for Compliance Profiles**. Geneva: International Electrotechnical Commission, jan. 2013.
- [4] PINTO, Leandro Israel et al. **Icaru-fb: An IEC 61499 compliant multiplatform software infrastructure**. IEEE Transactions on Industrial Informatics, v. 12, n. 3, p. 1074-1083, 2016.
- [5] PINTO, L. I. **Icaru-fb: uma infraestrutura de software aderente à norma iec 61499**. Dissertação. 2014. Centro de Ciências Tecnológicas, Universidade do Estado de Santa Catarina.

---

**DADOS CADASTRAIS**

---

**BOLSISTA:** Pedro Henrique Fleink

**MODALIDADE DE BOLSA:** Voluntário (IC)

**VIGÊNCIA:** 10/2024 a 08/2025 – Total: 11 meses

**ORIENTADOR(A):** Roberto Silvio Ubertino Rosso Junior

**CENTRO DE ENSINO:** CCT

**DEPARTAMENTO:** Departamento de Ciência da Computação.

**ÁREAS DE CONHECIMENTO:** Engenharia/Engenharia Elétrica

**TÍTULO DO PROJETO DE PESQUISA:** CONTROLE TOLERANTE A FALHAS EM SISTEMAS INDUSTRIAIS BASEADOS NA IEC 61499 (2023-2025).

**Nº PROTOCOLO DO PROJETO DE PESQUISA:** NPP3293-2023