XXI Escola Regional de Redes de Computadores / VIII Workshop de Segurança (ERRC/WRSeg). DOI: 10.5753/errc.2024.4678

ARTIGO COMPLETO/FULL PAPER

DDS-Builder: construção e disponibilização de um *dataset* público para sistemas ciberfísicos baseados em *Data Distribution Service* (DDS)

DDS-Builder: Construction and Release of a Public Dataset for Cyber-Physical Systems based on Data Distribution Service (DDS)

Douglas Fideles • douglas_rodrigues19@outlook.com
Universidade Federal de Uberlândia (UFU)

Diego Kreutz • diegokreutz@unipampa.edu.br
Universidade Federal do Pampa (UNIPAMPA)

Silvio Quincozes • silvioquincozes@unipampa.edu.br
Universidade Federal do Pampa (UNIPAMPA)

RESUMO. Neste trabalho, apresentamos a DDS-Builder, uma ferramenta desenvolvida para sistematizar a coleta, categorização e análise de vulnerabilidades em sistemas baseados no *Data Distribution Service* (DDS), além de introduzir um novo *dataset* público. A DDS-Builder integra-se à base de dados do *Vulners* e utiliza o *Gemini 1.5 Pro* para classificar vulnerabilidades conforme a estrutura *Common Weakness Enumeration* (CWE). Também realizamos uma análise dos dados coletados, oferecendo *insights* sobre a frequência, gravidade e tipos mais recorrentes de vulnerabilidades.

ABSTRACT. In this paper, we present the DDS-Builder, a tool designed to systematize the collection, categorization, and analysis of vulnerabilities in systems based on the Data Distribution Service (DDS), while also introducing a new public dataset. The DDS-Builder integrates with the *Vulners* database and utilizes *Gemini 1.5 Pro* to classify vulnerabilities according to the Common Weakness Enumeration (CWE) framework. We also conduct an analysis of the collected data, providing insights into the frequency, severity, and most common types of vulnerabilities.

PALAVRAS-CHAVE: DDS • Segurança • Vulnerabilidades • Dataset • LLM • Análise Comparativa
KEYWORDS: DDS • Security • Vulnerability • Dataset • LLM • Comparative Analysis

1 Introdução

A proliferação da Internet das Coisas (IoT) e de sistemas ciberfísicos (CPS) aumentou a demanda por soluções de comunicação em tempo real, de alto desempenho e confiáveis. Nesse contexto, o *Data Distribution Service* (DDS) [1], um protocolo de *middleware* padronizado pela *Object Management Group* (OMG)[1], tem se destacado por usar o padrão *publish/subscribe* [2]. Sua adoção em setores críticos como aeroespacial, defesa, automotivo, robótica e saúde exige uma atenção especial à segurança [3–6], tornando essencial a identificação e mitigação de vulnerabilidades associadas ao protocolo.

Com a crescente adoção do DDS em ambientes críticos, aumenta o risco de ataques cibernéticos, pois agentes maliciosos podem explorar vulnerabilidades para comprometer a segurança desses sistemas [7–9]. A detecção e análise de vulnerabilidades são, portanto, cruciais para garantir a integridade, disponibilidade e confiabilidade de sistemas baseados em DDS.

Entretanto, a pesquisa em segurança do DDS enfrenta um desafio significativo: a falta de *datasets* públicos e abrangentes [10]. Essa lacuna dificulta a análise de vulnerabilidades, a comparação entre implementações e o desenvolvimento de soluções de segurança robustas, comprometendo a capacidade de proteger essas infraestruturas críticas.

Para superar as limitações atuais, este trabalho propõe a ferramenta DDS-Builder, que implementa uma abordagem automatizada que combina diversas bases de dados de vulnerabilidades com o processamento de linguagem natural do Google Gemini 1.5 Pro. Essa solução permite coletar, categorizar e analisar vulnerabilidades em diferentes implementações do DDS, criando um *dataset* público. A DDS-Builder facilita a pesquisa em segurança do DDS, possibilitando:

- Análise aprofundada de vulnerabilidades categorizadas pelo CWE.

Douglas é estudante de mestrado da UFU. Diego e Silvio são professores da UNIPAMPA.

[1] https://www.omg.org/

- Melhor cobertura de avaliação de soluções.
- Desenvolvimento de soluções de segurança e modelos de *Machine Learning*.
- Geração de dados sintéticos para aprimorar o treinamento de modelos a partir de um conjunto de dados reais inicial.

2 Trabalhos Relacionados

Diversos estudos se dedicam à análise de segurança em DDS, como resumido na Tabela 1, focando na identificação de ameaças, vulnerabilidades, proposição de mecanismos de segurança e verificação formal do protocolo. Esses estudos, como o relatório da TrendMicro [7], apresentam uma visão abrangente, destacando ameaças e vulnerabilidades específicas, além de sugerir práticas de mitigação. Tais pesquisas evidenciam a crescente preocupação com a segurança do DDS em setores críticos, demonstrando a necessidade de abordagens mais robustas e detalhadas para proteção.

Tabela 1. Sumário de Trabalhos Relacionados

Referência	(1)	(2)	(3)	(4)	(5)
[11]	X	X	V	X	X
[10]	V	V	X	X	X
[12]	V	V	X	X	X
[13]	V	V	X	X	X
[14]	V	V	X	X	X
[7]	V	V	V	X	X
[15]	X	V	V	V	V
Esse trabalho	V	V	V	V	V

Legenda para células: (V) Tratado no trabalho; (X) Não mencionado;.

Legenda para Colunas: (1) DDS, (2) Análise de Vulnerabilidades, (3) Categorização de Vulnerabilidades, (4) Análise Comparativa de Implementações, (5) Dataset.

A pesquisa em [10] utiliza redes de *Coloring Petri Net (CPN)* para modelar e analisar formalmente a segurança do protocolo, identificando vulnerabilidades e propondo um esquema para aumentar a robustez do DDS. Similarmente ao nosso trabalho, os autores destacam a necessidade de um *dataset* abrangente para pesquisas em segurança de DDS, mas focam principalmente na modelagem formal. Em contraste, [12] investiga ataques direcionados ao lado do cliente de sistemas baseados em DDS, sem abordar a construção de *datasets* ou a categorização sistemática dessas vulnerabilidades.

O uso de métodos formais para a análise da segurança do DDS continua sendo amplamente explorado. Em [13], os autores combinam verificação formal e modelagem para examinar a segurança do DDS em sistemas robóticos. Já em [14], uma análise formal da segurança do *DDS Security* é conduzida utilizando a ferramenta *ProVerif*, para modelar os objetivos de segurança, identificar vulnerabilidades e sugerir resoluções para problemas comuns no protocolo.

No campo de geração de *datasets* e análise de vulnerabilidades, [11] propõe a ferramenta *VIET*, que aplica técnicas de *Natural Language Processing (NLP)* para extrair informações essenciais de descrições de vulnerabilidades. Essa ferramenta automatiza a avaliação de severidade e auxilia na atribuição de pontuações CVSS. Similar à nossa abordagem, o trabalho reconhece a ineficiência da avaliação manual de vulnerabilidades e explora soluções automatizadas com o uso de *LLM*, otimizando o processo e os resultados. Em trabalhos futuros, nossa ferramenta DDS-Builder poderá ser adotada para extrair e classificar vulnerabilidades em sistemas DDS.

Além disso, [15] propõe uma taxonomia para classificar vulnerabilidades em Python, utilizando uma abordagem automatizada para coleta e categorização de dados. O trabalho destaca a importância da categorização CWE e a necessidade de *datasets* abrangentes para a análise de segurança, de forma semelhante à nossa proposta para o DDS. A utilidade de *datasets* de vulnerabilidades para o treinamento de modelos de *Machine Learning* e a geração de dados sintéticos também é explorada, o que poderá ser aplicado ao *dataset* de vulnerabilidades em DDS que propomos.

A análise da literatura evidencia uma crescente preocupação com a segurança do DDS, mas aponta para a falta de *datasets* públicos e abrangentes na área. A maioria dos trabalhos se concentra na análise do protocolo, com pouca ênfase na comparação entre implementações de segurança. A ferramenta DDS-Builder busca preencher essa lacuna, fornecendo recursos valiosos para a comunidade de pesquisa e promovendo o desenvolvimento de sistemas baseados em DDS mais seguros e resilientes.

3 Arquitetura e Implementação

A ferramenta DDS-Builder foi projetada para automatizar a coleta, categorização e análise de vulnerabilidades em diversas implementações do DDS, estruturando-se em quatro módulos principais, conforme ilustrado na Figura 1. Cada módulo desempenha funções específicas, aplicando tecnologias e bibliotecas apropriadas para aprimorar o processo de construção do *dataset*. Essa arquitetura modular assegura uma coleta de dados eficiente e possibilita uma análise detalhada das

vulnerabilidades encontradas, oferecendo informações essenciais para fortalecer a segurança e a robustez das implementações baseadas no DDS.

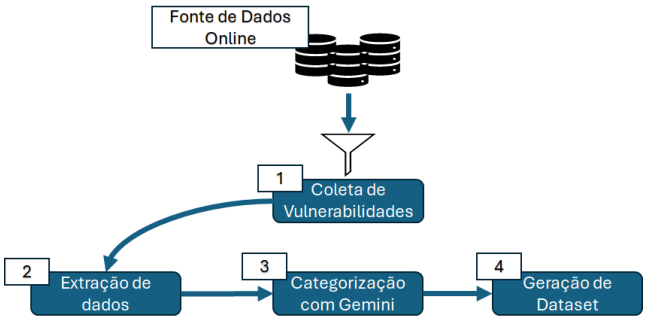

Figura 1. Arquitetura DDS-Builder

3.1 Módulo 1: Coleta de Dados

Este módulo coleta informações brutas sobre vulnerabilidades no DDS a partir da API do *Vulners* [16], escolhida pela sua abrangente cobertura, que abrange desde sistemas operacionais até *middleware* como o DDS. A API do *Vulners* permite acesso programático ao seu extenso banco de dados de vulnerabilidades, agregando informações de diversas fontes, como o NVD [17] e bancos de dados de *exploits*. Ela oferece detalhes como ID da vulnerabilidade, descrição, gravidade, pontuação CVSS e datas de publicação. As consultas são realizadas de forma iterativa, utilizando termos predefinidos como implementações populares de DDS. Os dados são armazenados em formato JSON e enviados ao próximo módulo para processamento.

As consultas à API são realizadas iterativamente usando uma lista predefinida de termos de pesquisa, incluindo nomes de implementações populares de DDS, como Fast DDS, Cyclone DDS, OpenDDS, RTI Connext DDS, Coredx DDS, Gurum DDS e InterCom DDS. A cada requisição, o módulo coleta dados como ID da vulnerabilidade, descrição, vendor afetado, pontuação CVSS, gravidade, entre outros. As informações são armazenadas em formato JSON e transmitidas para o próximo módulo.

3.2 Módulo 2: Extração de Dados

Este módulo processa os dados brutos recebidos em formato JSON do módulo de coleta, extraindo informações essenciais para a análise de vulnerabilidades. Utiliza-se a biblioteca JSON do Python para analisar o JSON e a biblioteca *re* para realizar expressões regulares, quando necessário, para formatar ou limpar os dados. As informações extraídas incluem o ID da vulnerabilidade (e.g., CVE-2023-1234), título, descrição detalhada, fabricante afetado, data de publicação, pontuação CVSS, gravidade (e.g., alta, média, baixa), categoria CWE, explicação da CWE e a fonte dos dados (*Vulners*). Essas informações são organizadas em uma lista de dicionários, onde cada dicionário representa uma vulnerabilidade com suas características específicas, facilitando o processamento e análise nos módulos subsequentes.

3.3 Módulo 3: Categorização com Gemini

Este módulo categoriza as vulnerabilidades extraídas em categorias CWE, utilizando o Google Gemini, um modelo avançado de linguagem natural. A categorização CWE é essencial para a análise de segurança, pois classifica as vulnerabilidades com base nas fraquezas de software mais comumente exploradas, de acordo com o dicionário mantido pela MITRE Corporation [18]. Para cada vulnerabilidade, a descrição é enviada ao Gemini, que retorna a categoria CWE mais relevante e uma explicação. Esses dados são analisados e adicionados ao dicionário correspondente à vulnerabilidade, facilitando a compreensão e mitigação dos riscos associados.

3.4 Módulo 4: Geração do *Dataset*

A etapa final do processo consolida todos os dados coletados e processados nas fases anteriores para gerar o *dataset* em formato CSV, escolhido por sua acessibilidade e compatibilidade tanto para leitura humana quanto para sistemas automatizados. O *dataset* inclui informações como ID, descrição, categoria CWE, explicação, causa e impacto, obtidos via Google Gemini. Para garantir a unicidade das entradas, verificamos vulnerabilidades duplicadas com base no ID. Por fim, realizamos uma análise manual para revisar a qualidade e precisão, corrigindo eventuais erros de categorização e eliminando dados irrelevantes.

3.5 Tecnologias

A ferramenta desenvolvida neste trabalho utiliza um conjunto de tecnologias e bibliotecas para automatizar a coleta, categorização e análise de vulnerabilidades em diferentes implementações do DDS. A implementação foi feita em Python 3.10 [19], uma linguagem de alto nível amplamente utilizada em análises de dados e segurança. O conjunto de bibliotecas padrão do Python que forma usados incluem *json* para o processamento de dados em formato JSON, *csv* para armazenar o *dataset*, *os*, *time* para tarefas adicionais como manipulação de datas, interação com o sistema operacional e controle do tempo entre as requisições. Também foram usadas bibliotecas adicionais como: *re* (versão 2.2.1) para manipulação de expressões regulares, *requests* (versão

2.30.0), para requisições HTTP à API do *Vulners*, *re* e google.generativeai (versão 0.8.3) que fornece acesso ao Google Gemini para a categorização de vulnerabilidades.

O Google Gemini 1.5 Pro [20], um modelo de linguagem de larga escala, foi selecionado para categorizar vulnerabilidades nas categorias CWE, fornecendo explicações contextuais e auxiliando na identificação de padrões e fraquezas comuns. Cada vulnerabilidade é submetida ao Gemini através de um *prompt* estruturado, e a resposta do modelo é processada para incluir a categoria CWE correspondente e uma breve explicação. Embora o Gemini tenha sido a escolha inicial, a arquitetura da ferramenta foi projetada para ser adaptável a outros modelos de linguagem, exigindo apenas ajustes no processamento de *prompts* e nas respostas, caso uma troca seja necessária.

4 Resultados

A ferramenta de coleta e análise de vulnerabilidades em DDS, usando o Google Gemini, foi aplicada a dados extraídos do *Vulners*, abrangendo diversas implementações do protocolo. Ao todo, 283 vulnerabilidades foram identificadas, distribuídas entre os principais fabricantes, conforme a Tabela 2. A maior quantidade foi detectada em "Fast DDS" (116), "RTI Connext DDS" (64) e "OpenDDS" (34), refletindo sua popularidade no mercado.

Fabricante	Quantidade
Fast DDS	116
RTI Connext DDS	64
Open DDS	34
Cyclone DDS	24
Intercom DDS	20
Coredx DDS	13
Gurum DDS	9
Todos os Fabricantes	3

Tabela 2. Número de Vulnerabilidades Por Fabricante

A análise temporal, que pode ser vista na Figura 2, revela um aumento significativo de vulnerabilidades a partir de 2021, com pico em 2022, atribuído ao uso crescente do DDS em áreas como robótica [3, 21–24]. Isso reflete a importância crescente da segurança em sistemas DDS, especialmente em implementações amplamente adotadas.

A média da pontuação CVSS das vulnerabilidades é de 5.8, com desvio padrão de 3.5, variando de 0 a 10, com uma mediana de 7.5, sugerindo um número

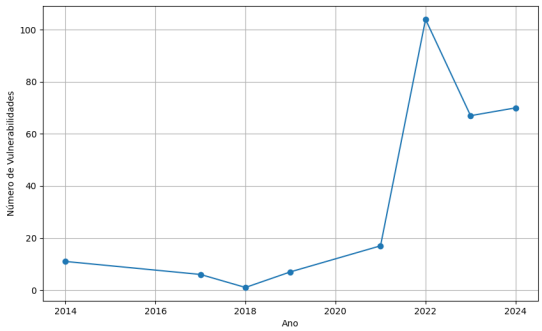

Figura 2. Gráfico de Distribuição de Vulnerabilidades DDS por Ano

considerável de vulnerabilidades de impacto elevado, principalmente em "Fast DDS" e "RTI Connext DDS".

As categorias CWE mais frequentes no *dataset* incluem CWE-400 (consumo de recursos descontrolado), CWE-120 (estouro de *buffer* clássico), CWE-122 (estouro de *buffer* baseado em *heap*), CWE-20 (validação de entrada imprópria) e CWE-787 (escrita fora dos limites). Essas categorias representam falhas recorrentes em software e podem comprometer a segurança e a confiabilidade de sistemas DDS.

A análise conduzida pelo Google Gemini revelou padrões nas causas das vulnerabilidades, como negação de serviço (DoS), execução arbitrária de código (RCE) e vazamento de informações. As causas mais comuns identificadas incluem manipulação inadequada de entradas, processamento incorreto de mensagens e falha na verificação de limites, confirmando os achados de pesquisas anteriores, como [7] e [10], que apontaram as mesmas causas para vulnerabilidades recorrentes em implementações de DDS.

5 Limitações

A pesquisa apresenta algumas limitações que impactam a interpretação dos resultados. Em primeiro lugar, a qualidade e abrangência do *dataset* dependem da precisão dos dados fornecidos pela API do *Vulners*, que podem conter informações incompletas ou imprecisas, como descrições de vulnerabilidades que não detalham o método de exploração ou o impacto exato. Apesar da validação manual amostral, a categorização das vulnerabilidades com o Google Gemini 1.5 Pro pode apresentar vieses, como a dificuldade em diferenciar falhas de implementação de configurações incorretas do protocolo. Por exemplo, o Gemini pode classificar erroneamente uma vulnerabilidade relacionada a uma configuração inadequada de um recurso do DDS como uma falha na implementação do protocolo. Além disso, a análise

comparativa de segurança se limita às vulnerabilidades documentadas no Vulners, não abrangendo as não descobertas ou não divulgadas publicamente.

6 Conclusão

A ferramenta desenvolvida para a coleta, categorização e análise de vulnerabilidades em DDS, utilizando o Google Gemini 1.5 Pro, mostrou-se eficaz na automação do processo de análise de segurança e na construção de um *dataset* abrangente e atualizado. A análise dos dados revelou vulnerabilidades significativas, especialmente em áreas críticas como estouro de buffer, consumo de recursos descontrolado e falhas de validação de entrada. A categorização automática realizada pelo Gemini destacou as categorias CWE mais comuns e permitiu uma análise detalhada das causas e impactos das vulnerabilidades identificadas.

O *dataset* gerado, disponibilizado publicamente[2], representa um recurso valioso para a comunidade de segurança de DDS. Pesquisadores podem usá-lo para conduzir análises detalhadas, desenvolver novas técnicas de detecção e mitigação de vulnerabilidades e, assim, aprimorar a segurança de sistemas baseados em DDS.

Finalmente, como trabalhos futuros, podemos destacar algumas melhorias para o *DDS-Builder*:

- Expandir o *DDS-Builder* com dados de outras fontes, como o NVD [17], para aumentar a abrangência e a precisão do *dataset*.
- Aprimorar a análise comparativa, utilizando técnicas mais avançadas de visualização de dados e conduzindo análises estatísticas mais robustas.
- Avaliar a capacidade do Gemini em gerar descrições detalhadas de *exploits* para simulações de ataques e desenvolvimento de contramedidas.
- Integrar o *DDS-Hunter* com *scanners* de vulnerabilidades para automatizar o processo de detecção e correção de falhas.

Declarações complementares

Agradecimentos

A pesquisa contou com apoio parcial da CAPES (Código de Financiamento 001).

Disponibilidade de dados e materiais adicionais

Os dados e/ou materiais adicionais poderão ser disponibilizados mediante solicitação.

[2] https://github.com/douglasfideles/DDSBuilder

Referências

1 Pardo-Castellote, G. OMG data-distribution service: Architectural overview. *In*: IEEE. 23RD International Conference on Distributed Computing Systems Workshops, 2003. Proceedings. 2003. P. 200–206.

2 (OMG), O. M. G. *Data Distribution Service (DDS) Specification*. 2015. https://opendds.org/. Accessed: 01 oct. 2024.

3 Macenski, S. *et al.* Robot operating system 2: Design, architecture, and uses in the wild. *Science robotics*, American Association for the Advancement of Science, v. 7, n. 66, eabm6074, 2022.

4 Scordino, C.; Mariño, A. G.; Fons, F. Hardware acceleration of data distribution service (dds) for automotive communication and computing. *IEEE Access*, IEEE, v. 10, p. 109626–109651, 2022.

5 Du, J.; Gao, C.; Feng, T. Formal safety assessment and improvement of DDS protocol for industrial data distribution service. *Future Internet*, MDPI, v. 15, n. 1, p. 24, 2022.

6 Wagner, P. G.; Birnstill, P.; Beyerer, J. DDS Security+: Enhancing the Data Distribution Service With TPM-based Remote Attestation. *In*: PROCEEDINGS of the 19th International Conference on Availability, Reliability and Security. 2024. P. 1–11.

7 Maggi, F. *et al.* A Security Analysis of the Data Distribution Service (DDS) Protocol. *In*: TREND Micro Research, Inc., Japan. 2022. P. 15–20.

8 Kim, H.; Kim, D.-K.; Alaerjan, A. ABAC-based security model for DDS. *IEEE Transactions on Dependable and Secure Computing*, IEEE, v. 19, n. 5, p. 3113–3124, 2021.

9 Abdulghani, R. M. *et al.* Vulnerabilities and security issues in IoT protocols. *In*: IEEE. 2020 First international conference of smart systems and emerging technologies (SMARTTECH). 2020. P. 7–12.

10 Du, J.; Gao, C.; Feng, T. Formal Safety Assessment and Improvement of DDS Protocol for Industrial Data Distribution Service. *Future Internet*, v. 15, n. 1, p. 24, 2023. DOI: 10.3390/fi15010024. Disponível em: https://doi.org/10.3390/fi15010024.

11 Zhang, S.; Zhang, M.; Zhao, L. VIET: A Tool for Extracting Essential Information from Vulnerability Descriptions for CVSS Evaluation. *In*: DATA and Applications Security and Privacy XXXVII: 37th Annual IFIP WG 11.3 Conference, DBSec 2023, Sophia-Antipolis, France, July 19–21, 2023, Proceedings. Berlin, Heidelberg: Springer-Verlag, 2023. P. 386–403. ISBN 978-3-031-37585-9. DOI: 10.1007/978-3-031-37586-6_23. Disponível em: https://doi.org/10.1007/978-3-031-37586-6_23.

12 Michaud, M. J.; Dean, T.; Leblanc, S. P. Attacking OMG data distribution service (DDS) based real-time mission critical distributed systems. *In*: PROCEEDINGS of the 13th International Conference on Malicious and Unwanted Software (MALWARE). Nantucket, MA, USA, out. 2018. P. 68–77.

13 White, R.; Caiazza, G.; Jiang, C. Network reconnaissance and vulnerability excavation of secure DDS systems. *In*: 2019 IEEE European Symposium on Security and Privacy Workshops (EUROS&PW). Stockholm, Sweden, jun. 2019. P. 57–66. DOI: 10.1109/EuroSPW.2019.00013.

14 Wang, B.; Li, H.; Guan, J. A Formal Analysis of Data Distribution Service Security. *In*: ACM. ACM Asia Conference on Computer and Communications Security (ASIA CCS '24). New York, NY, USA: ACM, jul. 2024. P. 12. DOI: 10.1145/3634737.3656288. Disponível em: https://doi.org/10.1145/3634737.3656288.

15 Bogaerts, F. C.; Ivaki, N.; Fonseca, J. A Taxonomy for Python Vulnerabilities. *IEEE Open Journal of the Computer Society*, IEEE Computer Society, n. 01, p. 1–12, 2024.

16 Vulners. *API Documentation*. 2024. Disponível em: https://vulners.com/api/. Acesso em: 4 out. 2024.

17 NIST. *National Vulnerability Database*. 2024. Disponível em: https://nvd.nist.gov/. Acesso em: 1 out. 2024.

18 MITRE CWE. *Common Weakness Enumeration*. 2024. Disponível em: https://cwe.mitre.org/. Acesso em: 2 out. 2024.

19 Python Software Foundation. *Python Language Reference*. 2024. Disponível em: https://docs.python.org/3/reference/. Acesso em: 2 out. 2024.

20 Google AI. *Gemini API*. 2024. Disponível em: https://blog.google/technology/ai/google-gemini-next-generation-model-february-2024/. Acesso em: 4 out. 2024.

21 Xiaowen, Z. *et al.* Design and Implementation of Robot Middleware Service Integration Framework Based on DDS. *In*: IEEE. 2022 IEEE International Conference on Real-time Computing and Robotics (RCAR). 2022. P. 588–593.

22 Jeong, S. *et al.* Behavior tree driven multi-mobile robots via data distribution service (DDS). *In*: IEEE. 2021 21st International Conference on Control, Automation and Systems (ICCAS). 2021. P. 1633–1638.

23 Lu, Q. *et al.* Modeling and Analysis of Data Flow-Oriented ROS2 Data Distribution Service. *International Journal of Software & Informatics*, v. 11, n. 4, 2021.

24 Lienen, C.; Middeke, S. H.; Platzner, M. fpgaDDS: An Intra-FPGA Data Distribution Service for ROS 2 Robotics Applications. *In*: IEEE. 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2023. P. 6261–6266.