

Uma aplicação embarcada de sistemas multiagentes normativos para operação e regulação de processo de manufatura*

Bruno A. Stefano¹, Jaime S. Sichman¹

¹Laboratório de Sistemas Inteligentes (LTI)
Escola Politécnica (EP)
Universidade de São Paulo (USP)
São Paulo, SP – Brasil.

{bruno.stefano, jaime.sichman}@usp.br

Abstract. *This paper presents the implementation of an embedded normative multi-agent system for the operation and regulation of a manufacturing process. A simulated scenario, based on a real scenario, is used for the implementation. Autonomous agents and norms were developed exemplifying the possibility of use of this technology in an industrial environment. This application enables the automation of more complex industrial processes, collaboration between different stages of the manufacturing system and automated production line regulation.*

Resumo. *Este artigo apresenta a implementação de um sistema multiagentes normativo embarcado na operação e regulação de um processo de manufatura. Uma simulação baseada em um cenário real é utilizada para a implementação. Foram desenvolvidos agentes autônomos e normas que exemplificam a possibilidade de uso dessa tecnologia em um ambiente industrial. Essa aplicação possibilita a automação de processos industriais mais complexos, a colaboração entre diferentes etapas do sistema de manufatura e regulação automatizada da produção.*

1. Introdução

As técnicas mais recentes de manufatura, como a manufatura aditiva e a customização em massa exigem a inserção de novas tecnologias nos chãos de fábrica. A importância de aspectos como sustentabilidade e governança se mostra cada vez mais presente em cada etapa da cadeia produtiva. Processos industriais mais complexos possibilitam a utilização de novas formas de automação e controle. Agentes autônomos aparecem como uma potencial opção para esse tipo de desafio tecnológico. Sua heterogeneidade e capacidade de raciocínio ajuda a adaptar a automação a uma indústria que muda cada vez mais rápido. A utilização de normas que regulam o comportamento dos agentes pode trazer para dentro dos sistemas de manufatura as regulações da indústria. Ainda assim, esse tipo de tecnologia é pouco explorado para aplicações industriais e de manufatura.

Este trabalho apresenta a aplicação de um sistema multiagentes normativo na operação de um processo de manufatura. Uma simulação baseada em uma situação real é

*Copyright © 2024 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

proposta para a implementação da aplicação. Os agentes são embarcados e se comunicam diretamente com os dispositivos da linha de produção. Normas são utilizadas para regular a operação dos agentes e o processo de manufatura como um todo. O objetivo dessa demonstração é, então, mostrar a possibilidade da utilização de um sistema multiagentes normativo embarcado na operação e regulação de um processo de manufatura.

A estrutura deste trabalho segue com a apresentação dos trabalhos relacionados na seção 2. O cenário motivador é exposto na seção 3. A seguir, na seção 4, é descrita a demonstração realizada. A seção 5 conclui o trabalho e apresenta seus possíveis próximos passos.

2. Trabalhos Relacionados

Existem, na literatura, duas principais opções para o desenvolvimento de sistemas multiagentes embarcados. Uma delas é o *Jason Embedded* [Pantoja et al. 2023], uma versão spin-off do framework *Jason* [Bordini et al. 2007]. Essa versão traz ao *Jason* a capacidade de controlar e ler sensores e atuadores, além de um filtro das percepções provenientes dos sensores, proposta inicialmente em [Stabile and Sichman 2015]. Esta funcionalidade é proporcionada pela arquitetura *Argo* [Pantoja et al. 2016], que utiliza um middleware executar a linguagem Java em processadores Arduino. Uma outra capacidade dessa versão é a de se comunicar com outros agentes e outros sistemas multiagentes, com a possibilidade de realizar migração de agentes, o que abre caminho para o desenvolvimento de sistemas multiagentes abertos, ou Open MAS.

A outra principal opção encontrada para o desenvolvimento de sistemas multiagentes embarcados é o framework *Embedded-Mas* [Dias and Brito 2023], cuja versão mais recente foi utilizada em conjunto com o framework de organizacional *Moise*. O *Embedded-Mas* possibilita a integração do *Jason* com o sistema operacional ROS. Dessa forma, os agentes *Jason* são munidos com capacidade de leitura nos tópicos do ROS para percepção e escrita nos tópicos e chamadas de serviços ROS para atuação. O framework *Moise* traz ao sistema multiagentes a possibilidade de programar uma camada organizacional: torna-se possível definir papéis e grupos para os agentes, criar objetivos e missões para os papéis e grupos, além de atribuir obrigações aos papéis e grupos em relação aos objetivos.

O *Jason Embedded* originalmente não traz nenhum aspecto normativo em sua concepção, ao menos até a escrita deste trabalho. O *Embedded-Mas* tampouco aborda questões normativas. Quando integrado ao *Moise*, entretanto, permite a definição de obrigações através dos aspectos organizacionais de papéis e missões. Entretanto, em ambas as abordagens o aspecto normativo é incompleto ou inexistente até o momento, pois a definição de normas é utilizada diretamente relacionada ao aspecto organizacional. Por outro lado, faltam mecanismos de regulação e inexistente a possibilidade de expressar a regulação nas definições do sistema.

3. Cenário Motivador

O cenário proposto para demonstração é baseado na Fábrica do Futuro (USP)¹, um ambiente de manufatura próximo do real. É um laboratório de pesquisa, ensino e demonstração

¹<https://sites.usp.br/fabricadofuturo/>.

de conceitos e tecnologias da Indústria 4.0. O laboratório utiliza uma montagem de skate para ilustrar os processos industriais.

Este estudo foca na etapa de controle de qualidade por visão computacional do processo de montagem do skate. Para realizar customização em massa, cada skate produzido tem alguns pontos de customização. Um cliente pode, por exemplo, escolher a cor de cada roda da prancha. Após a colocação das rodas do skate, um sistema de visão computacional faz a identificação de suas cores e valida a customização [Zancul et al. 2020].

Na demonstração a ser apresentada, o processo de validação da customização será coordenado por agentes autônomos. O sistema de visão computacional será simulado por um software que gera os dados das cores das rodas do skate. A demonstração, portanto, não será executada em um ambiente de manufatura real. Com os dados de visão computacional gerados, um sistema multiagentes normativo então é responsável por analisar o pedido do cliente e o que de fato foi executado na fábrica e concluir se a montagem das rodas foi feita corretamente. Além do raciocínio dos agentes, normas serão utilizadas para ilustrar a possibilidade de regulação desse processo ².

4. Demonstração

O sistema de visão computacional, baseado no da Fábrica do Futuro (USP), é aqui simulado por um microprocessador Arduino Mega. Um código específico para esse propósito foi desenvolvido. Tal código gera pseudo aleatoriamente uma dupla de dados (cor predominante, percentual da área da imagem ocupada por essa cor) para cada roda do skate, identificadas como frente, trás, esquerda e direita. As cores são escolhidas pseudo-aleatoriamente em uma lista com as opções vermelho, verde, azul, amarelo, preto e branco. A porcentagem da área ocupada pela cor também é gerada pseudo-aleatoriamente, porém sempre maior que 80%. Através da biblioteca Javino [Pantoja and Lazarin 2015], o Arduino consegue se comunicar com os agentes do sistema multiagentes. O Arduino tem então o papel de enviar as leituras do sistema de visão computacional geradas para serem interpretadas como percepções pelos agentes.

O sistema multiagentes normativo, responsável pela validação da customização, é formado por agentes de arquitetura BDI implementados na plataforma JaCaMo [Boissier et al. 2020]. Esse sistema é executado em um processador Raspberry Pi modelo 3B, que possui conexão serial com o Arduino. Especificamente, dois agentes são utilizados para essa tarefa. Um agente *validador* é responsável por se comunicar com o sistema de visão computacional simulado, raciocinar a partir dos dados obtidos e informar suas conclusões. O outro agente de *controle de qualidade* informa as especificações dos pedidos dos clientes e aguarda a conclusão da validação realizada. Uma norma que regula a operação do validador é implementada para incentivar que os pedidos sejam validados dentro de um limite de tempo. A Figura 1 apresenta uma visão geral da demonstração.

O agente validador tem o importante papel de se comunicar com o sistema de visão computacional simulado pelo Arduino. Os dados gerados pela visão computacional são integrados como crenças do agente através da arquitetura de agentes Argo [Pantoja et al. 2016]. A partir das preferências de cor recebidas do controle de qualidade

²Disponível em: <https://github.com/brunoastefano/demonstrations/tree/master/smart-factory-simulation>.

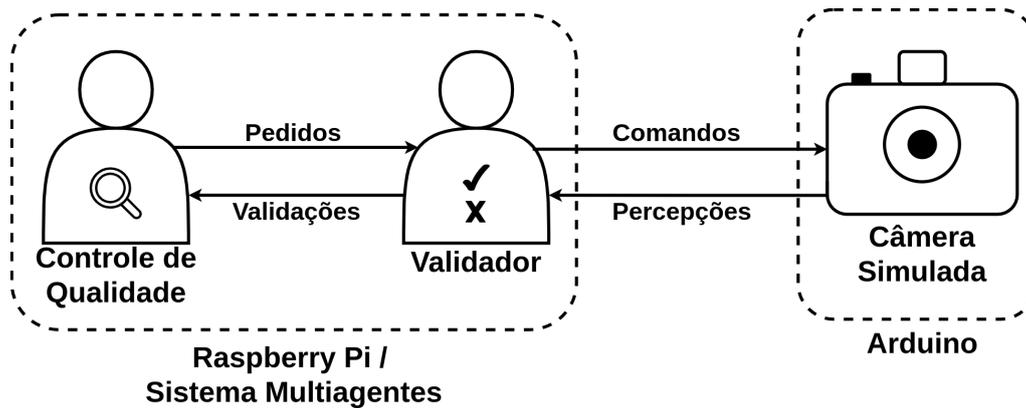


Figura 1. Visão geral da demonstração.

e de um nível de confiança mínimo previamente estipulado, que indica o mínimo percentual de área coberto pela cor para que ela seja aceita, o agente validador determina a conformidade de cada roda do skate. Com as atualizações das crenças sobre a conformidade de cada roda do skate, o agente validador consegue raciocinar sobre a conformidade geral do skate. Com a conclusão sobre a conformidade do skate, o agente validador pode então informar ao agente de controle de qualidade sobre a conformidade do skate com as preferências do cliente. A Figura 2 traz um trecho do código do raciocínio do agente sobre a conformidade do skate.

```

1 +frontLeftMeasure (ColorMeasure, ConfidenceMeasure) :
2 frontLeftSettings (ColorSetting) & minimumConfidence (MinCon) &
3 ColorMeasure == ColorSetting & ConfidenceMeasure > MinCon <-
4   +compliant (frontLeft);
5   !updateCompliance;
6   .
7
8 +!updateCompliance: compliant (frontLeft) & compliant (frontRight) &
9 compliant (backLeft) & compliant (backRight) <-
10  +compliant (skateboard);
11  !communicateCompliance;
12  .

```

Figura 2. Fragmento do código do raciocínio do agente sobre a conformidade do skate.

O agente de controle de qualidade gera as preferências do cliente de modo pseudo-aleatório. Esses dados são crenças do agente que serão comunicadas ao agente validador à medida que forem sendo atualizados. Como se trata de um ambiente simulado, essa lógica representa os pedidos dos clientes que vão chegando à linha de produção. Junto das especificações do pedido, o agente de controle qualidade também informa que um pedido está ativo e incrementa um contador de pedidos. O controle de qualidade deve então ser informado sobre a conclusão da validação, através do resultado. Assim, o agente de controle de qualidade incrementa um contador de resultados e informa ao agente validador que não há mais um pedido ativo.

A regulação do sistema é feita através de uma norma implementada utilizando a linguagem de programação normativa NPL(s) [Yan et al. 2024], ilustrada na Figura 3. O

agente de controle de qualidade utiliza uma abordagem agente-cêntrica para o processamento de normas através de uma nova arquitetura de agentes capaz de processar a NPL(s) [Yan et al. 2024]. A cada novo pedido ativo, a norma representada em NPL(s) exibida na Figura 3 entra em vigor e ela exige que o controle de qualidade receba um novo resultado de validação em até 10 segundos. Caso o resultado não chegue, uma sanção é aplicada e o agente de validação perde 5% de reputação. Essa perda de reputação pode ser usada para desqualificar este agente, ou até para agilizar seu processo de validação.

```

1 norm n1: orders(N) & N > 0
2   -> obligation(quality_agent, n1, results(N), '10 seconds')
3     if unfulfilled: srl(camera_agent, 5).
4 sanction-rule srl(Ag, Value) -> sanction(Ag, loseReputation(Value)).

```

Figura 3. Norma escrita em NPL(s) para regulação da operação dos agentes.

5. Conclusão

A aplicação implementada, uma simulação baseada em caso real, demonstra o uso de sistemas multiagentes normativos embarcados em um cenário de manufatura. A utilização dessa tecnologia em cenários industriais abre possibilidades para automação de processos de raciocínio e tomada de decisão. A implantação de normas permite a regulação dos processos automatizados, facilitando a integração dos sistemas computacionais com diretrizes da indústria, legislação e outros tipos de regulação. A comunicação entre os agentes simplifica a conexão entre diferentes etapas do processo de produção, que normalmente são feitas por interfaces entre diferentes sistemas ou até por intervenção humana.

Como essa primeira demonstração foi realizada em um ambiente simulado, trabalhos futuros devem explorar a aplicação dessas tecnologias em um cenário real. Assim é possível extrair seus reais benefícios para os processos de manufatura. Para enriquecer a implementação, é viável a maior exploração do impacto das sanções no comportamento dos agentes. Isso pode ser feito com o uso de sanções positivas (recompensas) e implementação das reações dos agentes às sanções. Além disso, o desenvolvimento de mais agentes e mais normas pode deixar mais completos a representação e o controle do sistema de manufatura.

Agradecimentos

O presente trabalho foi realizado no âmbito do projeto FAPESP/ANR NAIMAN, financiado pela Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), Processo Número 2022/03454-1 e também contou com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001.

Referências

- Boissier, O., Bordini, R. H., Hübner, J. F., and Ricci, A. (2020). *Multi-Agent Oriented Programming*. MIT Press.
- Bordini, R. H., Hübner, J. F., and Wooldridge, M. (2007). *Programming multi-agent systems in AgentSpeak using Jason*, volume 15. John Wiley & Sons.

- Dias, P. and Brito, M. d. (2023). Coordenação de robôs ROS com Agentes BDI e MOISE. In *Anais do XVII Workshop-Escola de Sistemas de Agentes, seus Ambientes e Aplicações - WESAAC 2023*, pages 67–78, Pelotas, RS.
- Pantoja, C. and Lazarin, N. (2015). A Robotic-agent Platform for Embedding Software Agents Using Raspberry Pi and Arduino Boards. In *Proceedings do 9o. Workshop-Escola de Sistemas de Agentes, seus Ambientes e aplicações (WESAAC 2015)*, pages 13–20.
- Pantoja, C. E., de Jesus, V. S., Lazarin, N. M., and Viterbo, J. (2023). A spin-off version of Jason for IoT and embedded multi-agent systems. In Naldi, M. C. and Bianchi, R. A. C., editors, *Intelligent Systems - 12th Brazilian Conference, BRACIS 2023, Belo Horizonte, Brazil, September 25-29, 2023, Proceedings, Part I*, volume 14195 of *Lecture Notes in Computer Science*, pages 382–396. Springer.
- Pantoja, C. E., Jr., M. F. S., Lazarin, N. M., and Sichman, J. S. (2016). ARGO: an extended Jason architecture that facilitates embedded robotic agents programming. In Baldoni, M., Müller, J. P., Nunes, I., and Zalila-Wenkstern, R., editors, *Engineering Multi-Agent Systems - 4th International Workshop, EMAS 2016, Singapore, Singapore, May 9-10, 2016, Revised, Selected, and Invited Papers*, volume 10093 of *Lecture Notes in Computer Science*, pages 136–155. Springer.
- Stabile, M. F. and Sichman, J. S. (2015). Evaluating perception filters in BDI Jason agents. In *2015 Brazilian Conference on Intelligent Systems, BRACIS 2015, Natal, Brazil, November 4-7, 2015*, pages 116–121. IEEE Computer Society.
- Yan, E., Nardin, L. G., Hübner, J. F., and Boissier, O. (2024). An Agent-Centric Perspective on Norm Enforcement and Sanctions. In *COINE - International Workshop on Coordination, Organizations, Institutions, Norms and Ethics for Governance of Multi-Agent Systems*, Auckland, Nova Zelândia.
- Zancul, E., Martins, H. O., Lopes, F. P., and Da Silva Neto, F. A. (2020). Machine Vision applications in a Learning Factory. *Procedia Manufacturing*, 45:516–521.