# ChatBDI: Think BDI, Talk LLM

Andrea Gatti
DIBRIS – University of Genoa
Genoa, Italy
andrea.gatti@edu.unige.it

Viviana Mascardi
DIBRIS – University of Genoa
Genoa, Italy
viviana.mascardi@unige.it

Angelo Ferrando
DSFIM – University of Modena-Reggio Emilia
Modena, Italy
angelo.ferrando@unimore.it

## ABSTRACT

This paper describes ChatBDI, a framework for extending Beliefs-Desires-Intentions (BDI) agents with the ability to communicate with humans exploiting Large Language Models (LLMs). The ChatBDI integration of BDI agents and LLMs relies on the Knowledge Query and Manipulation Language (KQML) as the intermediary language between humans and agents, and Jason – inside the JaCaMo framework – as the implementation language. The purpose of ChatBDI is not only to create brand new 'BDI speakers', but also to add communication capabilities to existing BDI agents without altering their source code. This 'chattification' serves a double purpose: by exploiting the BDI model, it provides an 'intentional brain' to LLMs, hence addressing one of their major limitations as speakers - the lack of intentionality; by exploiting the generative power of LLMs, it adds a creative and fluent 'language actuator' to BDI agents.

## KEYWORDS

ChatBDI, Beliefs-Desires-Intentions, BDI, Large Language Models, LLM, Chattification, General purpose, Domain independent

**ACM Reference Format:**
Andrea Gatti, Viviana Mascardi, and Angelo Ferrando. 2025. ChatBDI: Think BDI, Talk LLM. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 3 pages.

## 1 INTRODUCTION

Large Language Models (LLMs) are excellent in generating sentences in natural language, but they are neither a cognitive architecture nor reasoners and planners [16]. LLMs are not even speakers because they lack goals and intentions [12, 19, 24].

Conversely, Rao and Georgeff's Belief-Desire-Intention (BDI) architecture [20] is by design a cognitive architecture, and is one of the most well known implementations of strong agency [15]. Still, BDI agents are not speakers as well: they were not born with communication with humans in mind, and works that extend them with speaking capabilities are still few [2–7, 9, 10, 13, 17, 26].

In this paper we describe the design of ChatBDI and some preliminary experiments. Driven by the vision '*think BDI, talk LLM*', ChatBDI allows a BDI agent to act as the intentional brain of an LLM and the LLM to be the 'language actuator' of the BDI agent. The ChatBDI working prototype[1] uses Jason [**?** ] inside JaCaMo [1] for 'thinking', and CodeGemma [28] as LLM for 'talking'. CodeGemma is also used with Nomic-embed-text [18] for 'understanding': embeddings of messages from humans are computed and compared against embeddings of the agent's context. The Knowledge Query and Manipulation Language (KQML [8, 25]) plays a relevant role in ChatBDI because it is the message format adopted by Jason.

*ChatBDI allows multiagent systems (MAS) users to interact with Jason BDI agents in natural language, via LLMs. Thanks to a general purpose 'chattification' mechanism based on meta-programming, with the 'plans to understand and talk' injected in the MAS agents without even accessing their source code, the ability to interact in natural language is also available for legacy systems.*

The work closest to ours is by Frering et al. [9], who integrate Jason with an LLM to process natural language commands and leverage BDI reasoning. However, our approach is open-source and general-purpose, since it passes through KQML, whereas theirs is closed-source and tailored to a specific, closed MAS with domain-specific prompt engineering. In particular, their method only translates natural language sentences into a `goto(X,Y)` predefined command, while ChatBDI is meant to translate any kind of sentence into the closest literal in the receiver's code that might trigger a plan execution upon reception, with no further constraints. Other than [9], no works integrating LLMs and Jason agents exist. In their AAMAS 2024 Blue Sky paper [22], Ricci et al. envision (but neither design, nor develop) generative BDI architectures, namely architectures based on the BDI model and integrating generative AI technologies. Ichida et al. [14] exploit LLMs and reinforcement learning to bootstrap the reasoning capabilities of NatBDI agents. They target developers, while ChatBDI targets MAS users.

## 2 DEVELOPMENT AND USE OF CHATBDI

In order to operate, ChatBDI relies upon (i) a graphical 'chat-like' user interface to allow interaction with human users; (ii) a *kqml2nl* function from KQML messages produced by agents to sentences in natural language to be shared with human users; (iii) an *nl2kqml* function from users' sentences in natural language to KQML, for allowing humans to be understood by agents; (iv) one intermediary BDI agent that acts as an interpreter between natural and KQML languages, and hence between humans and BDI agents. The ChatBDI interpreter agent, ChatBDInt for short, is not tailored for any specific MAS domain and purpose. It is able to call the *nl2kqml* function, obtain the KQML message, and – being ChatBDInt a Jason agent itself – send it to the right receiver in the MAS. In the same way, ChatBDInt can receive KQML messages from the agents in the MAS, call the *kqml2nl* function on them, get sentences as



*Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA.* © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

[1] https://github.com/VEsNA-ToolKit/chatbdi.

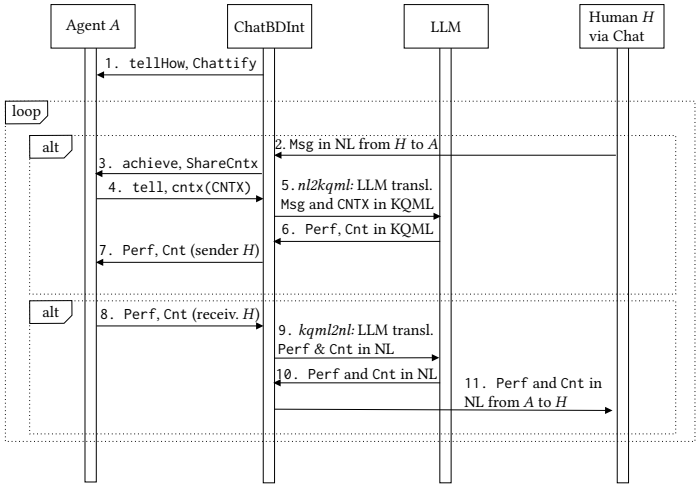


**Figure 1: Human-agent interaction via ChatBDInt: Msg (user message), NL (natural language), `Perf` (KQML performative), `Cnt` (KQML content), `CNTX` (context).**

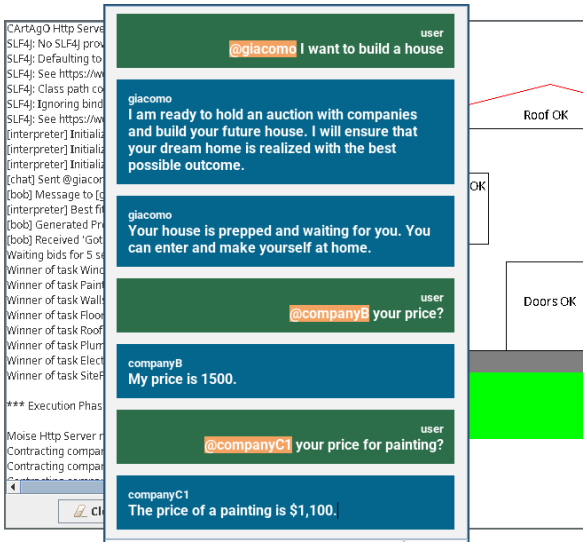


**Figure 2: Example of use of ChatBDI.**

result, and display them on the chat interface, in order for human users to read and understand them.

**From KQML to natural language.** In the LLM era, a very natural and coherent approach to implement *kqml2nl*, and *generate fluent sentences from structured messages*, is via an LLM. The *kqml2nl* function feeds the LLM with a carefully engineered - but domain-independent - prompt that provides the LLM the means to elaborate the request, and with the KQML message to be translated. The return value is the sentence generated by the LLM.

**From natural language to KQML.** This direction of the translation is much more complex and can be faced using different approaches. For example, the Natural Language Understanding (NLU) black box component available in intent-based chatbots like Rasa [21] and Dialogflow [11] may serve the purpose, as well as ad hoc trained classifiers. In the ChatBDI prototype, LLMs are employed also for tackling this second problem, in cascade after the embedding creation step. The *nl2kqml* function needs the LLM to understand the illocutionary force (or 'speech act', or 'communicative act', or 'performative'; examples are `tell`, `achieve`, `askOne`) of the user's sentence first. Then, based on the recognized performative, the message content is translated into a logical atom, to serve as proper content to a KQML message. A context consisting of the current beliefs and other useful information about the chattified agents, stored by ChatBDInt, is added to the prompt to the LLM. The context is not hard-wired: rather, it is asked to the agents in the MAS at execution time, for the sake of generality.

**From humans to agents, and back.** Figure 1 outlines the interaction between the human $H$, ChatBDInt, the LLM, and the agents ($A$ represents one among all the agents in the MAS). First, ChatBDInt 'chattifies' the agents by teaching them how to provide the context and to handle unrecognized messages via `tellHow` (message 1). Once chattified, agents and users can start communicating bidirectionally: user messages are processed (arrow 2), enriched with the context on the domain that chattified agents can provide (arrows 3 and 4), and are translated by the LLM into KQML (arrows 5 and 6); the KQML message is sent to the intended receiver $A$ as if it came from the human $H$. Conversely, agent messages for humans (arrow 8) are converted into natural language (arrows 9 and 10), and sent by ChatBDInt to $H$ as if they came from $A$ (arrow 11).

**Implementation.** ChatBDI is designed for flexibility and adaptability across diverse MAS. To ensure modularity, ChatBDInt is decoupled from the translation process, which is managed via CArtAgO artifacts [23]. This allows seamless replacement of the translation implemented by *kqml2nl* an *nl2kqml* without modifying the core framework, accommodating the evolving nature of MAS.

**Example.** To show the generality of our approach, we chattified the house building JaCaMo MAS available at https://github.com/jacamo-lang/jacamo/tree/main/examples/house-building (Figure 2). We only added two plans to the `giacomo` code, to make it able to receive and answer requests to build a house. Via the ChatBDI interface, the human user may ask `giacomo` (note the `@` in `@giacomo` to address a specific agent) to build a house. Following the original JaCaMo code, `giacomo` starts an auction, that successfully completes with the house being built. The human user may also directly ask the building companies for the price of their services.

## 3 CONCLUSIONS

A key advantage of ChatBDI's chattification, besides letting the user enter the MAS, is improved MAS inspectability: developers can track agent interactions in natural language via chat, aiding debugging. Messages can also be customized per user [27]. A drawback of ChatBDInt's plan injection into legacy agents - instead - is potential security risks in sensitive domains: MAS owners should approve this process before deployment. In our prototype, CodeGemma is used as one possible instantiation for natural language translation, leveraging LLMs for flexibility and reusability. However, in high-security environments, frameworks like Rasa and Dialogflow may be preferred. Finally, the *kqml2nl* and *nl2kqml* translations may be wrong or inaccurate, due to the big challenge of their domain-independency: fail-safe mechanisms must be foreseen.

**Credits.** This work was supported by the *ENGINES* project funded by the Italian MUR program PRIN 2022, grant 20229ZXBZM.

## REFERENCES


[1] Olivier Boissier, Rafael H. Bordini, Jomi F. Hübner, and Alessandro Ricci. 2020. *Multi-agent oriented programming: programming multi-agent systems using JaCaMo.* MIT Press.

[2] Louise A. Dennis and Nir Oren. 2022. Explaining BDI Agent Behaviour through Dialogue. *Auton. Agents Multi Agent Syst.* 36, 1 (2022), 29.

[3] Débora C. Engelmann, Lucca Dornelles Cezar, Alison R. Panisson, and Rafael H. Bordini. 2021. A Conversational Agent to Support Hospital Bed Allocation. In *BRACIS (1) (Lecture Notes in Computer Science, Vol. 13073)*. Springer, 3–17.

[4] Débora C. Engelmann, Juliana Damasio, Tabajara Krausburg, Olimar Teixeira Borges, Lucca Dornelles Cezar, Alison R. Panisson, and Rafael H. Bordini. 2021. Dial4JaCa - A Demonstration. In *PAAMS (Lecture Notes in Computer Science, Vol. 12946)*. Springer, 346–350.

[5] Débora C. Engelmann, Juliana Damasio, Tabajara Krausburg, Olimar Teixeira Borges, Mateus da Silveira Colissi, Alison R. Panisson, and Rafael H. Bordini. 2021. Dial4JaCa - A Communication Interface Between Multi-agent Systems and Chatbots. In *PAAMS (Lecture Notes in Computer Science, Vol. 12946)*. Springer, 77–88.

[6] Débora C. Engelmann, Alison R. Panisson, Renata Vieira, Jomi Fred Hübner, Viviana Mascardi, and Rafael H. Bordini. 2023. MAIDS - A Framework for the Development of Multi-Agent Intentional Dialogue Systems. In *AAMAS*. ACM, 1209–1217.

[7] Zeinab Namakizadeh Esfahani, Débora Cristina Engelmann, Angelo Ferrando, Massimiliano Margarone, and Viviana Mascardi. 2023. Integrating Ontologies and Cognitive Conversational Agents in On2Conv. In *EUMAS (Lecture Notes in Computer Science, Vol. 14282)*. Springer, 66–82.

[8] Timothy W. Finin, Richard Fritzson, Donald P. McKay, and Robin McEntire. 1994. KQML As An Agent Communication Language. In *CIKM*. ACM, 456–463.

[9] Laurent Frering, Gerald Steinbauer-Wagner, and Andreas Holzinger. 2025. Integrating Belief-Desire-Intention Agents with Large Language Models for Reliable Human-Robot Interaction and Explainable Artificial Intelligence. *Eng. Appl. Artif. Intell.* 141 (2025), 109771.

[10] Andrea Gatti and Viviana Mascardi. 2023. VEsNA, a Framework for Virtual Environments via Natural Language Agents and Its Application to Factory Automation. *Robotics* 12, 2 (2023), 46.

[11] Google. 2025. *Dialogflow web site.* https://cloud.google.com/dialogflow Accessed on February 14, 2025.

[12] Reto Gubelmann. 2024. Large Language Models, Agency, and why Speech Acts Are Beyond them (for Now)– a Kantian-Cum-Pragmatist Case. *Philosophy & Technology* 37, 1 (2024), 32.

[13] Alexandre Yukio Ichida and Felipe Meneguzzi. 2023. Modeling a Conversational Agent using BDI Framework. In *SAC*. ACM, 856–863.

[14] Alexandre Yukio Ichida, Felipe Meneguzzi, and Rafael C. Cardoso. 2024. BDI Agents in Natural Language Environments. In *AAMAS*. International Foundation for Autonomous Agents and Multiagent Systems / ACM, 880–888.

[15] Nicholas R. Jennings, Katia P. Sycara, and Michael J. Wooldridge. 1998. A Roadmap of Agent Research and Development. *Auton. Agents Multi Agent Syst.* 1, 1 (1998), 7–38.

[16] Subbarao Kambhampati. 2024. Can Large Language Models reason and plan? *Annals of the New York Academy of Sciences* 1534, 1 (March 2024), 15–18. https://doi.org/10.1111/nyas.15125

[17] Aida Mustapha, Mohd Sharifuddin Ahmad, and Azhana Ahmad. 2013. Conversational Agents as Full-Pledged BDI Agents for Ambient Intelligence. In *ISAmI (Advances in Intelligent Systems and Computing, Vol. 219)*. Springer, 221–228.

[18] Zach Nussbaum, John X. Morris, Brandon Duderstadt, and Andriy Mulyar. 2024. Nomic Embed: Training a Reproducible Long Context Text Embedder. arXiv:2402.01613 [cs.CL]

[19] Paul Piwek. 2024. Are Conversational Large Language Models Speakers?. In *Proc. of the 28th Workshop on the Semantics and Pragmatics of Dialogue - Poster Abstracts.*

[20] Anand S. Rao and Michael P. Georgeff. 1995. BDI Agents: From Theory to Practice. In *ICMAS*. The MIT Press, 312–319.

[21] Rasa technologies. 2025. *Rasa web site.* https://rasa.com/ Accessed on February 14, 2025.

[22] Alessandro Ricci, Stefano Mariani, Franco Zambonelli, Samuele Burattini, and Cristiano Castelfranchi. 2024. The Cognitive Hourglass: Agent Abstractions in the Large Models Era. In *AAMAS*. International Foundation for Autonomous Agents and Multiagent Systems / ACM, 2706–2711.

[23] Alessandro Ricci, Mirko Viroli, and Andrea Omicini. 2005. Programming MAS with Artifacts. In *PROMAS (Lecture Notes in Computer Science, Vol. 3862)*. Springer, 206–221.

[24] Zachary P. Rosen and Rick Dale. 2024. LLMs Don't "Do Things with Words" but their Lack of Illocution Can Inform the Study of Human Discourse. In *Proceedings of the 46th Annual Meeting of the Cognitive Science Society*. 2870–2876.

[25] Renata Vieira, Álvaro F. Moreira, Michael J. Wooldridge, and Rafael H. Bordini. 2007. On the Formal Semantics of Speech-Act Based Communication in an Agent-Oriented Programming Language. *J. Artif. Intell. Res.* 29 (2007), 221–267.

[26] Wilson Wong, Lawrence Cavedon, John Thangarajah, and Lin Padgham. 2012. Flexible Conversation Management Using a BDI Agent Approach. In *IVA (Lecture Notes in Computer Science, Vol. 7502)*. Springer, 464–470.

[27] Elena Yan, Samuele Burattini, Jomi Fred Hübner, and Alessandro Ricci. 2023. Towards a Multi-Level Explainability Framework for Engineering and Understanding BDI Agent Systems. In *WOA (CEUR Workshop Proceedings, Vol. 3579)*. CEUR-WS.org, 216–231.

[28] Heri Zhao, Jeffrey Hui, Joshua Howland, Nam Nguyen, Siqi Zuo, Andrea Hu, Christopher A. Choquette-Choo, Jingyue Shen, Joe Kelley, Kshitij Bansal, Luke Vilnis, Mateo Wirth, Paul Michel, Peter Choy, Pratik Joshi, Ravin Kumar, Sarmad Hashmi, Shubham Agrawal, Zhitao Gong, Jane Fine, Tris Warkentin, Ale Jakse Hartman, Bin Ni, Kathy Korevec, Kelly Schaefer, and Scott Huffman. 2024. CodeGemma: Open Code Models Based on Gemma. *CoRR* abs/2406.11409 (2024).