

Autonomous Business System via Neuro-symbolic AI

Cecil Pang^{1, 2}

¹AI Engineering, USA TODAY Co., Inc., New York, USA

²School of Systems Science and Industrial Engineering, Binghamton University, State University of New York, Binghamton, USA

ORCID: 0009-0003-1725-7873

Hiroki Sayama^{2, 3, 4}

²School of Systems Science and Industrial Engineering, Binghamton University, State University of New York, Binghamton, USA

³Binghamton Center of Complex Systems, Binghamton University, State University of New York, Binghamton, USA

⁴Waseda Innovation Lab, Waseda University, Tokyo, Japan

ORCID: 0000-0002-2670-5864

Abstract—Current business environments require organizations to continuously reconfigure cross-functional processes, yet enterprise systems are still organized around siloed departments, rigid workflows, and hard-coded automation. Meanwhile large language models (LLMs) excel at interpreting natural language and unstructured data but lack deterministic, verifiable execution of complex business logic. To address this gap, here we introduce AUTOBUS, an Autonomous Business System that integrates LLM-based AI agents, predicate-logic programming, and business-semantics-centric enterprise data into a coherent neuro-symbolic AI architecture for orchestrating end-to-end business initiatives. AUTOBUS models an initiative as a network of tasks with explicit pre-/post-conditions, required data, evaluation rules, and API-level actions. Enterprise data is organized as a knowledge graph whose entities, relationships, and constraints are translated into logic facts and foundational rules, providing the semantic grounding for task reasoning. Core AI agents synthesize task instructions, enterprise semantics, and available tools into task-specific logic programs, which are executed by a logic engine that enforces constraints, coordinates auxiliary tools, and orchestrate execution of actions and outcomes. Humans define and maintain the semantics, policies and task instructions, curate tools, and supervise high-impact or ambiguous decisions, ensuring accountability and adaptability. We detail the AUTOBUS architecture, the anatomy of the AI agent generated logic programs, and the role of humans and auxiliary tools in the lifecycle of a business initiative. A case study on subscriber retention in a content subscription business demonstrates how AUTOBUS can be instantiated in data-rich organizational environments and accelerate time to market. A reference implementation of the case study is available in GitHub: <https://github.com/cecilpang/autobus-paper>.

Keywords—neuro-symbolic AI, autonomous business system, LLM, AI agent, enterprise data, logic programming

I. INTRODUCTION

A. Background

In today's markets, businesses must continuously identify opportunities, adjust strategies, and optimize tactics at short notice in order to grow or simply remain competitive. Meanwhile business capabilities increasingly rely on the integration of diverse functions—sales, marketing, finance, logistics, and data—working together toward shared outcomes. Traditional enterprise software, however, is organized around siloed departments and optimizes tasks rather than end-to-end

value creation. Even though Enterprise Resource Planning (ERP) software suites unify data across modular components [1], which helps with cross-module analysis and reporting, it often requires rigid workflows that impede rapid changes. As a result, business organizations struggle to reconfigure capabilities at the pace required by today's markets.

This challenge sits at the core of enterprise digital transformation, which is more than technology adoption: it is a multi-dimensional shift spanning strategy, people, organizational structure, customers, ecosystem partnerships, technology, and innovation. Recent case-based research shows that successful digital transformations depends on coordinated, cross-functional orchestration rather than isolated tools [2].

Decades of business process redesign (BPR) and business process management (BPM) research anticipated this need. Classic BPR argues that companies should “reengineer” work—rethinking processes across functional boundaries instead of automating legacy tasks—because dramatic performance gains come from process-centric, not department-centric, design. Complementary work on process innovation positions technology as the key enabler of such redesign. Together, these perspectives emphasize end-to-end process orientation, explicit ownership of outcomes, and measurement against customer-facing results [3], [4]. More recent research links BPM and BPR directly to digital transformation as its integral part [5], [6]. It requires new BPM logics—lighter-weight processes, infrastructural flexibility, and “mindful actors” who can act on less than ideal situations—that better accommodate continual change and data-driven decision-making [7].

Recent advances in large language model (LLM)-based AI agents offer new opportunities to re-imagine how business operates. Multi-agent systems (MAS), once largely confined to scientific and engineering domains, are becoming viable in business contexts thanks to the increasingly software-centric and data-driven nature of business operations. A recent study by Dennis et al. [8] suggested that AI agents are likely to be accepted as full team members, requiring many classic collaboration questions to be re-examined in the context of human-AI teams. Saghafian and Idan [9] argued that the future application of AI in many domains should focus on combining artificial and human intelligence.

While LLMs excel in natural language understanding, learning and semantic reasoning over unstructured data, they

often struggle with deterministic execution, especially when it involves complex and lengthy logic reasoning [10]. This makes them less effective in business process automation where decisions need to be reasonable and verifiable. This gap can be filled by a traditional branch of AI—predicate calculus-based logic programming [11]. The formal reasoning nature of logic programming provides a simple and compact mechanism to encode business logic, and it is intuitive to reason over. The combining of the strengths of neural network-based LLM and the reasoning power of logic programming, which is a type of symbolic AI, is known as a neuro-symbolic approach.

B. Proposed System

We introduce Autonomous Business System (AUTOBUS)—a system that intelligently orchestrates end-to-end business initiatives by integrating human intelligence, neuro-symbolic AI, and business semantics grounded in business semantic-centric enterprise data. A business initiative represents a concrete objective with defined goals, evaluation metrics, constraints, and expected artifacts. In contrast to software systems organized around fixed and siloed departments, AUTOBUS treats the cross-functional human-AI teams as the primary building block of the enterprise, aiming to deliver end-to-end enterprise level results. *Figure 1* shows an overview of AUTOBUS.

AUTOBUS is best suited for modern data-rich organizations that rely on cross-functional workflows. In such an environment, business initiatives are typically data and software driven, and core systems expose APIs and events so that data can flow across tasks with limited manual intervention.

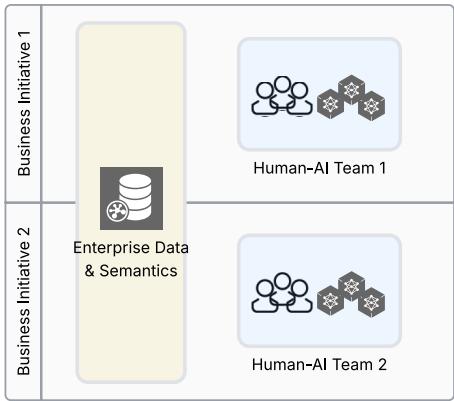


Fig.1. Overview of Autonomous Business System (AUTOBUS). AUTOBUS integrates human intelligence, neuro-symbolic AI, and business semantics to orchestrate end-to-end business initiatives. Teams in which humans and AI are peer members collaborating under shared objectives, metrics, constraints, and expected artifacts—delivering enterprise-level outcomes beyond siloed departments.

C. Related Work

Research on autonomous business and agent-based business process management can be dated back to more than two decades. Jennings et al. [12] described an agent-based business process management system where agents are humans. Flor [13] explored programmable autonomous business built entirely in

software and did not require humans to operate other than the community of customers of the business. More recently, Hughes et al. [14] investigated how AI agents and agentic systems could reshape industries by decentralizing decision-making, reconfiguring organizational structures, and strengthening cross-functional collaboration. Lashkevich et al. [15] studied using LLMs to assist the redesign of business process to reduce waiting time. Regarding having AI agents as team members, Dennis, Lakhial and Sachdeva [8] studied the potentially positive and negative disruptive influence on teamwork. Lemon [16] surveyed the challenges and some of the open research directions in multi-party systems where AI agents interact with more than one human.

Even though LLM-based agentic systems are a very recent development, they have already made tremendous progress. Acharya, Kuppan and Divya [17] surveyed the current and potential applications in various real-world domains such as healthcare finance, supply chain, education, and adaptive software systems. Khamis [18] examined the different architectural components of an agentic system. Hu, Lu and Clune [19] even proposed a new research area of automated design of agentic systems.

While neuro-symbolic AI has been gaining ground in AI research according to these review papers: [20], [21], [22], research on integrating LLM and logic programming for business systems has only recently emerged. Mezini et al. [23] integrated temporal-logic rules with learned neural network models to predict business process outcomes. Manuj et al. [24] and used LLMs to extract facts from legal contracts and then used logic programs for coverage and eligibility reasoning for policy compliance and claims. Neuro-symbolic AI has also been applied in E-commerce recommendation systems [25], [26], [27]. Though not specifically for business systems, Vakharia et al. [28] used logic programming to improve the performance of LLMs in domain specific question answering tasks. Pan et al. [31] used LLMs to translate logical reasoning problems in natural language into a symbolic representations and then used a symbolic logic solver to solve the problems deterministically. Borazjanizadeh and Piantadosi [29] showed that integrating logic programming into the inference pipeline of LLMs yielded robust performance gains. Yang, Chen and Tam [30] used logic programs generated by LLMs to solve arithmetic problems.

D. Our Contribution

Existing neuro-symbolic approaches in literature that utilize LLMs and logic programming tend to focus on narrow reasoning tasks or domains. To the best of our knowledge, no prior work was designed for real-world, data-rich business environments, providing actionable guidance for the design and operation of end-to-end business initiatives. AUTOBUS:

- improves time to market: a coherent data, humans, and neuro-symbolic AI architecture enables business initiatives to iterate rapidly, and
- provides practical guidance: designed for real-world businesses, we provide reference implementation of a case study with open source code in GitHub.

II. AUTOBUS

A. Business Initiative

A business initiative is modeled as a sequence of tasks, often with parallel branches, that transform inputs into measurable outcomes. Each task has explicit pre-/post-conditions, required data, and well-defined actions. Typically these actions are executed by operational software such as API calls. Most importantly, decisions were made throughout the execution of the tasks. Each individual task as well as the business initiative as a whole are evaluated by defined rules and metrics. Parts or the whole of the initiative can be iterative. Any business initiatives that can be modeled in this form, as illustrated in *Figure 2*, are candidates for AUTOBUS.

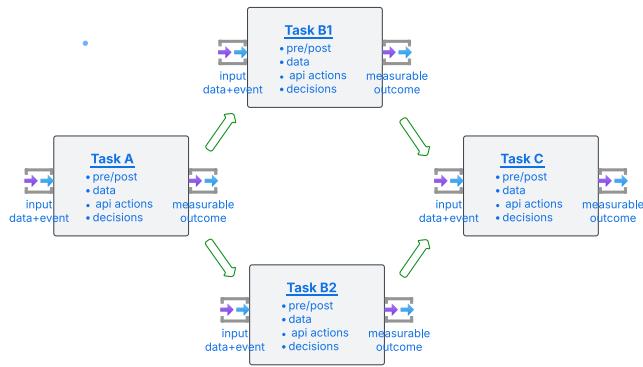


Fig. 2. Business initiative model: inputs traverse a sequence of data-driven tasks—often with parallel branches—to yield measurable outcomes. Each task defines pre/post-conditions, required data, and software-executed API actions, with decisions throughout. The initiative is evaluated by goals and metrics, and sub-flows may iterate.

B. Principal Components

At the core of AUTOBUS are humans, enterprise data, LLM-based AI agents, and a logic programming engine. Each business initiative has its own task- and domain-specific instructions, while auxiliary tools—including other AI agents, machine learning models, and internal or external APIs—provide additional computational and informational support, as well as sending actions to operational systems. *Figure 3* presents the principal components of AUTOBUS.

Operationally, the core AI agents construct logic programs by synthesizing information from business semantics, task instructions and tools availability. These logic programs are then executed by the logic engine. Groundings for facts are typically from enterprise data. In addition, the logic programs invoke tools to get groundings for predicates that cannot be obtained upfront from enterprise data. Some examples are real-time data from the web, data science models, and system events. The logic programs could also invoke other AI agents that in turn utilize other tools or resources to fulfill the request. Tools are also used to perform actions of the logic programs, such as calling APIs or persisting outcome of the logic program. *Figure 4* depicts the following steps:

- translate task instructions into logic predicates,

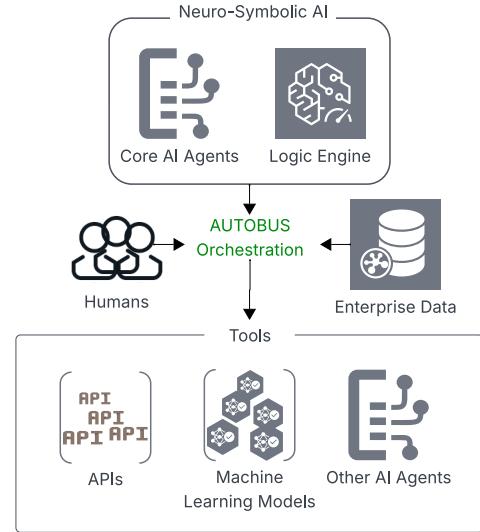


Fig.3. Principal components of AUTOBUS. Neuro-symbolic AI, humans, and enterprise data provide complementary intelligence and context to the AUTOBUS orchestration core, which coordinates decisions and actions. A tooling layer of APIs, AI agents, machine-learning models, and other integrations executes these decisions against operational systems.

- augment these predicates with the relevant facts and foundational rules derived from business semantics in enterprise data, and
- specify groundings from enterprise data and auxiliary tools such as machine learning models, API calls and other AI agents.

The logic engine then executes the generated logic program to drive actions.

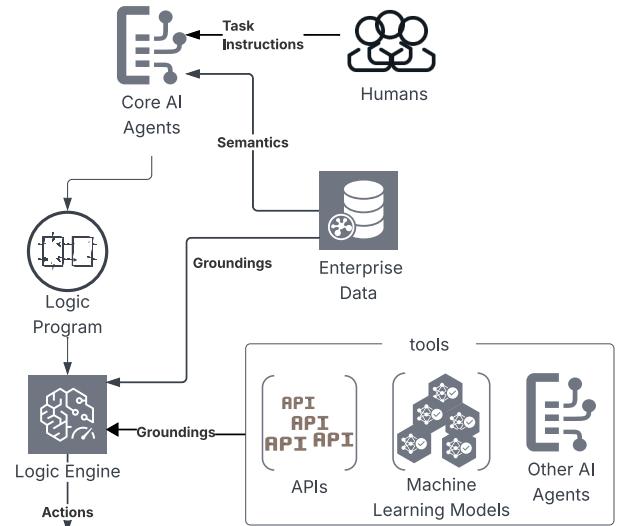


Fig. 4. Logic program generation and execution. AI agents translate task instructions into logic predicates and enrich them with relevant logic facts, foundational rules, and groundings derived from enterprise data and auxiliary tools. The resulting logic program is executed by the logic engine, which determines task feasibility, enforces constraints, and orchestrates actions.

C. Logic Program

Logic programs models tasks using the fundamental constructs of predicate logic—the language of facts, rules, and structured inference. Prolog is used in this work. At a conceptual level, a task can be described by logic predicates such as:

```
task(T) .  
requires(T, Data) .  
precondition(T, P) .  
postcondition(T, Q) .
```

These predicates capture the essential semantics of the tasks, their dependencies, and their expected effects. They allow AUTOBUS to represent an initiative not as ad-hoc procedural code but as a declarative structure that explicitly states what must be true before and after each task, as well as what data and decisions govern its execution. This makes the intent and logic of the tasks transparent, auditable, and directly aligned with business semantics.

Building on these preliminaries, logic programming enables AUTOBUS to orchestrate tasks by reasoning over the evolving state. For example, a task is executable when the logic inference engine can derive that all its preconditions hold. Formally, when predicates such as

```
holds(P, State) .
```

are satisfied, the rule that governs task readiness is satisfied. Parallel branches arise naturally as multiple tasks simultaneously satisfy their preconditions, and iterative structures can be expressed through rules that re-activate a task when certain conditions recur:

```
repeat_until(Task, Goal) .
```

Decisions within a task are represented by rules with alternative heads or guarded conditions, allowing the system to select actions, such as invoking an API to update enterprise data based on the current logical state:

```
invoke(Action, Params) .
```

Evaluation rules and metrics are encoded as higher-level predicates that compute whether an initiative or a task meets defined business criteria, enabling continuous assessment during execution. For example, a customer success evaluation of whether an initiative resolves the customer problem combines operational and customer-satisfaction metrics into a single evaluation rule:

```
resolved(I) :- outcome(I, resolved) .  
success(I) :-  
    resolved(I),  
    customer_satisfaction(I, Score),  
    Score >= 4.0 .
```

A logic program for a task in AUTOBUS consists of three main parts:

1. facts and foundational rules,
2. task specific rules, and
3. predicates for actions and task outcomes.

Enterprise data serve as the source of logic facts and foundational rules, which describe business entities, their relationships, and constraints. These are common to all initiatives within an organization and serve as the anchor, providing a shared foundation for tasks within an initiative, and across all initiatives. To enable AI agents to reliably derive these facts, AUTOBUS adopts the business-semantics-centric enterprise data system specified in a recent work [32]. Guided by task instructions, the AI agents extract and construct the facts and foundational rules, grounding them in the enterprise's formal semantic structures. This semantic foundation is essential for ensuring that the generated logic programs are consistent with domain meaning and for maximizing automated logic program generation and minimizing the need for humans to describe the data manually.

These enterprise-data-derived facts can be further augmented with real-time information, including human inputs, live machine learning model inferences, API responses, and outputs produced by other AI agents.

Beyond the facts, there are task-specific rules. These rules are generated by the AI agents based on the task instructions and descriptions of various tools available. Since tasks differ in purpose, scope, and domain, the resulting rules can vary widely. Common categories include rules that:

- identify and filter business entities based on task-specific criteria, and
- provides reasoning from facts via rules to actions.

For predicates that lead to actions or task outcomes, it involves triggering some events. For modern data- and software-driven businesses, it typically means invoking the services of some operational systems, passing in some prepared dataset. It could also simply be persisting the outcomes of the task. *Figure 5* illustrates the anatomy of a logic program in AUTOBUS.

D. Business Semantics Centric Enterprise Data

A foundational component of AUTOBUS is a business semantics centric enterprise data system, as specified by Pang [32]. As the author stated, such a system is an essential and integral element of a modern data-driven business rather than merely a data management utility. We extend this view by positioning this enterprise data system as one of the three pillars of autonomous business, together with humans and neuro-symbolic AI.

Being business semantics centric, the aforementioned enterprise data system organizes information into typed and schema-governed structures around business entities such as consumers and subscriptions, and their relationships. These structured data assets form the basis of an enterprise knowledge graph (KG) in which entities, relationships, and constraints are explicitly represented. For example, consumer-subscription associations and status transitions are captured as KG triples aligned with the business organization's ontology and reference data models. This provides the business semantic from which

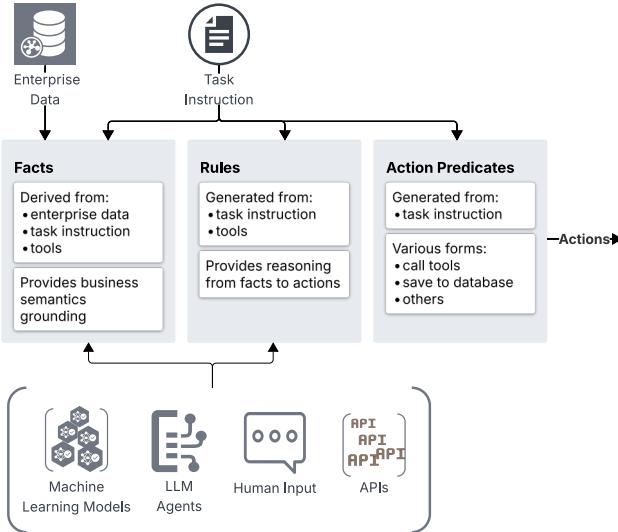


Fig. 5. Anatomy of an AUTOBUS logic program. Each task-level logic program comprises three elements: (1) facts and foundational rules derived from enterprise data, (2) task-specific rules generated from the task instructions and available tools, and (3) predicates that define actions and task outcomes. Task-specific rules identify relevant entities, apply filters, and connect facts to actions, while action predicates trigger operational events.

AUTOBUS derives the facts and foundational rules, as well as their groundings, for the logic programs. KG triples are translated to logic facts, such as the following snippet, which states that `c123` is a consumer who subscribes to an active subscription `s456`:

```
consumer(c123).
subscription(s456).
subscribe(c123, s456).
has_status(s456, active).
```

Schema-level constraints and domain rules in the KG are similarly translated to foundational logic rules, including integrity checks and task preconditions:

```
active_subscription(S) :-
    has_status(S, active),
    subscribe(_, S).

precondition(send_promotion(C)) :-
    consumer(C),
    subscribe(C, S),
    active_subscription(S).
```

The first predicate defines an active subscription as one that is subscribed by any consumer and has an active status. The second one states that promotion is sent to a consumer only if the consumer has an active subscription.

Structured enterprise data represent a semantic knowledge graph. When translated to logic facts and foundational rules, it provides the grounding for reasoning about the business.

Methodologies for generating knowledge graph from relational data are well-established [33]. Typically, table rows are translated to graph nodes, which are then linked to their column values to form bipartite “star” sub-graphs. These sub-graphs are subsequently connected via shared values to form the entire graph.

Moreover, logic inference tools that work directly on large knowledge graphs stored in a database, rather than text based rules, have started to emerge [34], [35]. This is essential for scaling to the potentially very large data volume in today’s businesses.

E. Humans

Humans play a key role in AUTOBUS, providing the domain expertise, contextual judgment, and strategic oversight that anchor the system’s neuro-symbolic reasoning. Humans supply the nuanced judgements and interpretations—such as ethical considerations, policy intent, organizational priorities, and exception handling—that neither logic programs nor AI agents can reliably infer in isolation. Before setting up tasks for a business initiative, humans:

- define and maintain the business semantics and structure enterprise data accordingly, and
- curate the ontologies and common policies that govern the logic of tasks.

When setting up the business initiatives, humans:

- compose task instructions,
- set the objectives and evaluation criteria, and
- curate tools to be used for the tasks.

During execution of tasks, humans:

- validate ambiguous or high-impact decisions in the generated logic programs, and
- iteratively refine the task instructions in response to task and initiative level outcomes.

Human involvement is critical not only for ensuring correctness, accountability, and alignment with organizational goals, but also for enabling AUTOBUS to adapt to real-world complexity and maintain trustworthiness over time.

III. VALIDATION CASE STUDY

A. Introduction

We validate the effectiveness of AUTOBUS through a case study of implementing a business initiative for an online content subscription business as a proof of concept. The company has well over a million subscribers and is in the early stage of AI transformation. We first describe the business initiative, break it down into three tasks, and describe the generated logic programs. We then compare and contrast with the traditional approach and provide metrics that shows that AUTOBUS accelerates the time to market. We also discuss the factors contributing to this improvement.

As presented in *Figure 6*, the key business entities involved in the case study are consumer, subscription, and product. A consumer is either a subscriber, who subscribes to one or more products, or a non-subscriber. In other words, a consumer subscribes to zero or more products. There are profile attributes attached to consumers, including information about consumers collected from the subscriptions as well as any data from third-party data providers.

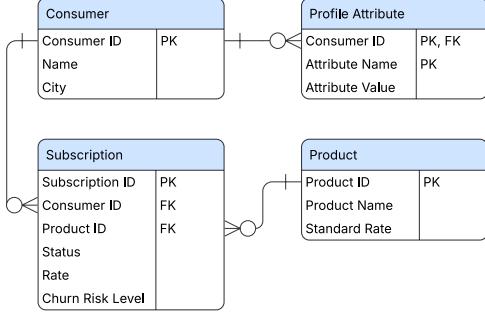


Fig. 6. Entity–relationship diagram for the case study. The primary business entities are consumer, subscription, and product. A consumer may hold zero or more subscriptions. Consumer records include profile attributes obtained from subscription data and augmented with third-party data sources. The relationships among these entities and their associated attributes are formally depicted in the diagram.

B. Business Initiative and Tasks

For a subscription-based business, the top priority is to acquire new subscribers and at the same time retain existing ones. This case study is about subscriber retention. A common tactic is to proactively reach out to high-churn-risk subscribers and offer them perks. The perks can be for certain markets and/or specific products. Concretely, the case study is to send promotions to active subscribers who:

1. subscribe to a certain product, pay a monthly rate above certain dollars, have a certain churn risk level, and
2. have household incomes above the median of the cities residing in.

The initiative is broken down into three tasks, as depicted in *Figure 7*:

1. Identify and retrieve the target subscribers based on Criterion 1 only. That is, without the household income constraint, which will be handled in Task 3.
2. Obtain the median household income for the cities where the subscribers reside. This involves calling a specialized AI agent that fetches median city household income data from the web.
3. Based on the outcomes of Tasks 1 and 2, further filter the subscriptions to include only those whose households have income above the cities' median. Then send promotions to the target subscribers. The latter involves the action of making API calls to the marketing platform.

C. Logic Programs

The logic programs of all three tasks are generated by the same core AI agent of AUTOBUS, with task specific

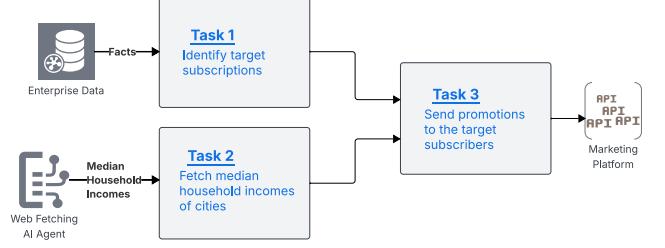


Fig. 7. The initiative targets subscribers who (i) subscribe to Product 1, (ii) have churn-risk level 4, and (iii) have household income above median of the city. It is organized into three tasks: retrieving eligible subscribers from enterprise data, fetching median incomes of cities from the web, and sending promotions to the resulting target subscribers through marketing APIs.

instructions. Each logic program comprises three sections: (1) facts, (2) task specific rules, and (3) actions. These elements are common to data-driven business tasks. The structure of the logic program generated for Task 3 is presented in *Figure 8*. It is quite sophisticated, containing the following elements:

1. Facts and foundational rules: loaded from enterprise data, and the outcomes of Task 1 and 2.
2. Task rules: the logic of targeting subscribers who (i) are savable chuns determined by Task 1, and (ii) have household incomes above the medians of the cities of residence, which are the outcome of Task 2.
3. Actions: (i) persist the target subscriptions into the database, and (ii) send the target subscriptions to a marketing campaign via a tool call.

D. Result and Discussion

The case study was conducted by the first author at the workplace as a proof of concept to mimic a typical business initiative that involves audience segmentation and engagement. While the ultimate goal is to improve retention, a useful metric for the initiative is time-to-market. Whether the team can quickly iterate the initiative with controlled variation of parameters so as to validate hypothesis often determines whether the ultimate goal of minimizing churn can be achieved. This metric is closely aligned with the main value proposition of AUTOBUS: accelerating time to market.

To evaluate time-to-market, we compare the time required to complete the three tasks shown in *Figure 7* with a baseline reflecting the organization's existing approach. In the baseline, each task requires building task-specific data pipelines and deploying them into the enterprise data infrastructure. The estimate is derived from observations of similar efforts conducted in the past, which typically involve multiple siloed departments. Coordination across these groups often requires several rounds of discussions before roles, responsibilities, and technical details are fully aligned. This overhead is repeated for subsequent initiatives—even when they are similar—because team composition often changes and time is again spent re-establishing shared understanding, down to minor details.

With AUTOBUS, task logic programs are generated by a consistent set of core AI agents using task instructions, shared enterprise data, and standardized tools. This approach largely

eliminates the unproductive coordination cycles inherent in the baseline workflow. The reduction in redundant planning and

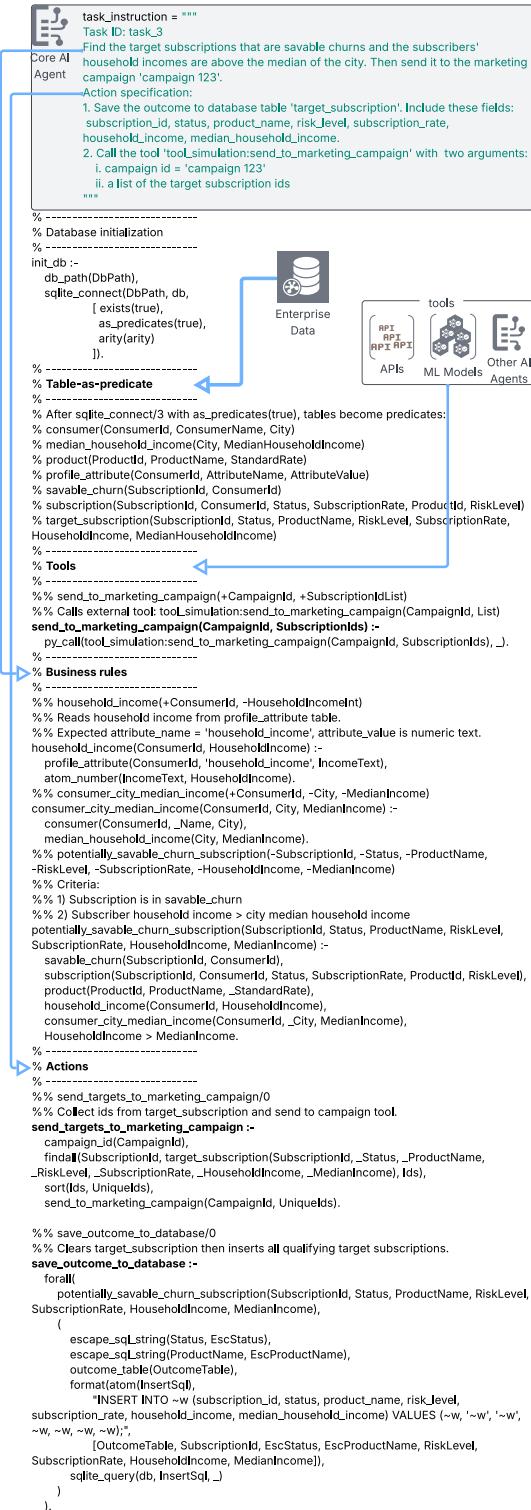


Fig. 8. Structure of the logic program for Task 3, generated by the AUTOBUS core AI agent from task-specific instruction. The program consists of three sections—facts, task-specific rules, and actions—where facts and foundational rules are derived from enterprise data and the outputs of Tasks 1 and 2; task rules encode the targeting logic; and actions persist the results and invoke a marketing campaign via tool call.

alignment effort, together with the consistent execution of the logics, constitute the primary contributor to improved time to market.

The results are recorded in *Table 1*. Since Tasks 1 and 2 can be executed in parallel, the time to market is two days for AUTOBUS versus 2 weeks with the existing approach. There is flexibility in structuring the initiative. Breaking it down into three tasks in this case study is a design choice that follows a good system design practice of grouping logically related activities in separate modules for ease of reasoning, maintenance and reuse. Functionally, it would work just as well if the three tasks are combined into one.

TABLE I
CASE STUDY METRIC: TIME TO MARKET

Task	Existing Approach	AUTOBUS
1 ^a	Retrieve savable churns based on product, rate and risk level	1 week
2 ^a	Obtain the median household income for cities from the web	1 week
3	Further filter by household income and send result to marketing platform	1 day

End-to-end case study execution time of AUTOBUS, with benchmark against the existing approach.

^a Tasks 1 and 2 are run in parallel.

IV. CONCLUSION, LIMITATION & FUTURE WORK

Autonomous Business System (AUTOBUS) provides a logical, clear, and well-organized architecture for businesses to focus their resources on end-to-end business initiatives that bring measurable values, rather than siloed departmental operations. It is also a practical system for humans to team up with neuro-symbolic AI in a data-rich business environment, with a reference implementation for a case study.

Enterprise data volume can be very large in today's data driven business. It could be a challenge to effectively transform business entities their attributes and relationships into predicate logic facts usable in logic programs. This work does not specifically address this issue. There are tools to cope with this challenge, such as PyReason [34], which can operate directly on a graph database, and Logica [36], which compiles to SQL for execution in big data databases.

There is a critical factor for the success of AUTOBUS: effective human-AI teaming. It demands very different skills than traditional roles. While AUTOBUS provides a practical system, it does not address the organizational issue of transforming existing workforces to team with AI. This is a research topic on its own.

We also suggest the following two avenues for future work:

1. Conduct broader case studies of applying AUTOBUS, and refine the architecture based on the findings.
2. Autonomous business system planning and optimization driven by business priorities. For example, representing the logic programs of business initiatives as graphs, the research can be guided by complex systems and adaptive network theories [37].

REFERENCES

- [1] M. Gupta and A. Kohli, "Enterprise resource planning systems and its implications for operations function," *Technovation*, vol. 26, no. 5, pp. 687–696, May 2006, doi: 10.1016/j.technovation.2004.10.005.
- [2] L. Ivančić, V. Vučkić, and M. Spremić, "Mastering the Digital Transformation Process: Business Practices and Lessons Learned," *Technol. Innov. Manag. Rev.*, vol. 9, no. 2, pp. 36–50, 2019, doi: <http://doi.org/10.22215/timreview/1217>.
- [3] M. Hammer, "Reengineering Work: Don't Automate, Obliterate," *Harvard Business Review*. Accessed: Oct. 15, 2025. [Online]. Available: <https://hbr.org/1990/07/reengineering-work-dont-automate-obliterate>
- [4] T. H. Davenport, *Process innovation: reengineering work through information technology*. USA: Harvard Business School Press, 1993.
- [5] R. Ivanišević, D. Horvat, and M. Matić, "Business process redesign as a basic aspect of digital business transformation," *Strateg. Manag. - Int. J. Strateg. Manag. Decis. Support Syst. Strateg. Manag.*, vol. 30, no. 3, Sept. 2025, doi: 10.5937/StraMan2300040I.
- [6] A.-M. Stjepić, L. Ivančić, and D. S. Vugec, "Mastering digital transformation through business process management: Investigating alignments, goals, orchestration, and roles," *J. Entrep. Manag. Innov.*, vol. 16, no. 1, pp. 41–74, 2020, doi: 10.7341/20191612.
- [7] A. Baiyere, H. Salmela, and T. Tapanainen, "Digital transformation and the new logics of business process management," *Eur. J. Inf. Syst.*, vol. 29, no. 3, pp. 238–259, May 2020, doi: 10.1080/0960085X.2020.1718007.
- [8] A. R. Dennis, A. Lakhial, and A. Sachdeva, "AI Agents as Team Members: Effects on Satisfaction, Conflict, Trustworthiness, and Willingness to Work With," *J. Manag. Inf. Syst.*, vol. 40, no. 2, pp. 307–337, Apr. 2023, doi: 10.1080/07421222.2023.2196773.
- [9] S. Saghafian and L. Idan, "Effective Generative AI: The Human-Algorithm Centaur," *Harv. Data Sci. Rev.*, no. Special Issue 5, Dec. 2024, doi: 10.1162/99608f92.19d78478.
- [10] C. Gershenson, "AI's errors may be impossible to eliminate – what that means for its use in health care," The Conversation. Accessed: Dec. 14, 2025. [Online]. Available: <http://theconversation.com/ais-errors-may-be-impossible-to-eliminate-what-that-means-for-its-use-in-health-care-251036>
- [11] R. Thomason, "Logic-Based Artificial Intelligence," in *The Stanford Encyclopedia of Philosophy*, Spring 2024., E. N. Zalta and U. Nodelman, Eds., Metaphysics Research Lab, Stanford University, 2024. Accessed: Dec. 19, 2025. [Online]. Available: <https://plato.stanford.edu/archives/spr2024/entries/logic-ai/>
- [12] N. R. Jennings, T. J. Norman, P. Faratin, P. O'Brien, and B. Odgers, "Autonomous agents for business process management," *Appl. Artif. Intell.*, vol. 14, no. 2, pp. 145–189, Feb. 2000, doi: 10.1080/088395100117106.
- [13] N. V. Flor, "An analysis of the design of a programmable autonomous business," *J. Syst. Inf. Technol.*, vol. 7, no. 1–2, pp. 111–128, June 2003, doi: 10.1108/1328726038000776.
- [14] L. Hughes *et al.*, "AI Agents and Agentic Systems: A Multi-Expert Analysis," *J. Comput. Inf. Syst.*, vol. 65, no. 4, pp. 489–517, July 2025, doi: 10.1080/08874417.2025.2483832.
- [15] K. Lashkevich, F. Milani, M. Avramenko, and M. Dumas, "LLM-Assisted Optimization of Waiting Time in Business Processes: A Prompting Method," in *Business Process Management*, A. Marrella, M. Resinas, M. Jans, and M. Rosemann, Eds., Cham: Springer Nature Switzerland, 2024, pp. 474–492. doi: 10.1007/978-3-031-70396-6_27.
- [16] O. Lemon, "Conversational AI for multi-agent communication in Natural Language: Research directions at the Interaction Lab," *AI Commun.*, vol. 35, no. 4, pp. 295–308, Sept. 2022, doi: 10.3233/AIC-220147.
- [17] D. B. Acharya, K. Kuppan, and B. Divya, "Agentic AI: Autonomous Intelligence for Complex Goals—A Comprehensive Survey," *IEEE Access*, vol. 13, pp. 18912–18936, 2025, doi: 10.1109/ACCESS.2025.3532853.
- [18] A. Khamis, "Agentic AI Systems: Architecture and Evaluation Using a Frictionless Parking Scenario," *IEEE Access*, vol. 13, pp. 126052–126069, 2025, doi: 10.1109/ACCESS.2025.3590264.
- [19] S. Hu, C. Lu, and J. Clune, "Automated Design of Agentic Systems," Mar. 02, 2025, *arXiv*: arXiv:2408.08435. doi: 10.48550/arXiv.2408.08435.
- [20] B. C. Colelough and W. Regli, "Neuro-Symbolic AI in 2024: A Systematic Review," Apr. 05, 2025, *arXiv*: arXiv:2501.05435. doi: 10.48550/arXiv.2501.05435.
- [21] A. d'Avila Garcez and L. C. Lamb, "Neurosymbolic AI: the 3rd wave," *Artif. Intell. Rev.*, vol. 56, no. 11, pp. 12387–12406, Nov. 2023, doi: 10.1007/s10462-023-10448-w.
- [22] U. Nawaz, M. Anees-ur-Rahaman, and Z. Saeed, "A review of neuro-symbolic AI integrating reasoning and learning for advanced cognitive systems," *Intell. Syst. Appl.*, vol. 26, p. 200541, June 2025, doi: 10.1016/j.iswa.2025.200541.
- [23] A. Mezini, E. Umili, I. Donadello, F. M. Maggi, M. Mancanelli, and F. Patrizi, "Neuro-Symbolic Predictive Process Monitoring," Aug. 31, 2025, *arXiv*: arXiv:2509.00834. doi: 10.48550/arXiv.2509.00834.
- [24] M. Kant, S. Nabi, M. Kant, R. Scharrer, M. Ma, and M. Nabi, "Towards Robust Legal Reasoning: Harnessing Logical LLMs in Law," Feb. 24, 2025, *arXiv*: arXiv:2502.17638. doi: 10.48550/arXiv.2502.17638.
- [25] Y. Xian *et al.*, "CAFE: Coarse-to-Fine Neural Symbolic Reasoning for Explainable Recommendation," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, in CIKM '20. New York, NY, USA: Association for Computing Machinery, Oct. 2020, pp. 1645–1654. doi: 10.1145/3340531.3412038.
- [26] M. Wu, B. Chen, S. Zhu, B. Zheng, W. Peng, and M. Zhang, "Neuro-symbolic recommendation model based on logic query," *Knowl.-Based Syst.*, vol. 284, p. 111311, Jan. 2024, doi: 10.1016/j.knosys.2023.111311.
- [27] G. Spillo, C. Musto, M. de Gemmis, P. Lops, and G. Semeraro, "Recommender systems based on neuro-symbolic knowledge graph embeddings encoding first-order logic rules," *User Model. User-Adapt. Interact.*, vol. 34, no. 5, pp. 2039–2083, Nov. 2024, doi: 10.1007/s11257-024-09417-x.
- [28] P. Vakharia, A. Kufeldt, M. Meyers, I. Lane, and L. H. Gilpin, "ProSLM: A Prolog Synergized Language Model for explainable Domain Specific Knowledge Based Question Answering," in *Neural-Symbolic Learning and Reasoning*, T. R. Besold, A. d'Avila Garcez, E. Jimenez-Ruiz, R. Confalonieri, P. Madhyastha, and B. Wagner, Eds., Cham: Springer Nature Switzerland, 2024, pp. 291–304. doi: 10.1007/978-3-031-71170-1_23.
- [29] N. Borazjanizadeh and S. T. Piantadosi, "Reliable Reasoning Beyond Natural Language," Dec. 01, 2025, *arXiv*: arXiv:2407.11373. doi: 10.48550/arXiv.2407.11373.
- [30] X. Yang, B. Chen, and Y.-C. Tam, "Arithmetic Reasoning with LLM: Prolog Generation & Permutation," May 28, 2024, *arXiv*: arXiv:2405.17893. doi: 10.48550/arXiv.2405.17893.
- [31] L. Pan, A. Albalak, X. Wang, and W. Wang, "Logic-LM: Empowering Large Language Models with Symbolic Solvers for Faithful Logical Reasoning," in *Findings of the Association for Computational Linguistics: EMNLP 2023*, H. Bouamor, J. Pino, and K. Bali, Eds., Singapore: Association for Computational Linguistics, Dec. 2023, pp. 3806–3824. doi: 10.18653/v1/2023.findings-emnlp.248.
- [32] C. Pang, "Toward Data Systems That Are Business Semantic Centric and AI Agents Assisted," *IEEE Access*, vol. 13, pp. 113752–113762, 2025, doi: 10.1109/ACCESS.2025.3583260.
- [33] J. f. Sequeda, S. hamid Tirmizi, O. Corcho, and D. p. Miranker, "Review: survey of directly mapping sql databases to the semantic web," *Knowl Eng Rev*, vol. 26, no. 4, pp. 445–486, Dec. 2011, doi: 10.1017/S0269888911000208.
- [34] D. Aditya, K. Mukherji, S. Balasubramanian, A. Chaudhary, and P. Shakarian, "PyReason: Software for Open World Temporal Logic," *arXiv.org*. Accessed: Nov. 11, 2025. [Online]. Available: <https://arxiv.org/abs/2302.13482v3>
- [35] E. Skvortsov, Y. Xia, and B. Ludäscher, "Logica: Declarative Data Science for Mere Mortals," OpenProceedings.org, 2024. doi: 10.48786/EDBT.2024.84.
- [36] E. Skvortsov, Y. Xia, S. Bowers, and B. Ludäscher, "The Logica System: Elevating SQL Databases to Declarative Data Science Engines,"
- [37] H. Sayama *et al.*, "Modeling complex systems with adaptive networks," *Comput. Math. Appl.*, vol. 65, no. 10, pp. 1645–1664, May 2013, doi: 10.1016/j.camwa.2012.12.005.