SJST
sjst.scst.edu.ly

Surman Journal for Science and Technology
ISSN: Online (2790-5721) - Print (2790-5713)

مجلة صرمان للعلوم والتقنية
Vol 6, No.1.Jan – May. 2024
Pages: 288 ~ 293

Bridging Intentional and Formal Models: An Agent-Oriented RE Framework Integrating i and CASL

Salama Eisa Ali Abudalrasul
Al-ahli higher institute of computer technologies and administrative and financial sciences
Ssfngits@gmail.com

Abstract
The growing complexity of software systems necessitates rigorous analysis during the early stages of requirements engineering (RE). While Goal-Oriented RE (GORE) frameworks like i∗*i*∗ are excellent for capturing stakeholder intentions and social dependencies, they lack formal semantics for precise verification. Conversely, formal methods often operate at a level of detail removed from high-level organizational reasoning. This paper presents a novel agent-oriented RE methodology that bridges this gap by integrating the informal, intentional modeling of the i∗*i*∗ framework with the formal, mental-state-based specifications of the Cognitive Agents Specification Language (CASL). We introduce Intentional Annotated Strategic Rationale (iASR) diagrams, an extension of i∗*i*∗'s SR models, augmented with composition and link annotations to provide the precision needed for formal mapping. We define a systematic mapping from iASR diagrams into CASL, enabling the formal analysis of agent goals, knowledge, and interactions. The framework's utility is demonstrated through a detailed meeting scheduler case study, showing how it supports the verification of requirements consistency, goal decompositions, and epistemic feasibility. The key contribution is a seamless, traceable path from high-level, intentional models to verifiable, executable agent specifications.

Keywords: *Requirements Engineering, Agent-Oriented Software Engineering, Formal Methods, i∗i∗, CASL, Goal-Oriented Analysis.*

الملخص
تستلزم التعقيدات المتزايدة للأنظمة البرمجية إجراء تحليل دقيق خلال المراحل المبكرة لهندسة المتطلبات. بينما تتفوق أطر عمل هندسة المتطلبات الموجهة بالأهداف - مثل إطار - i* في التقاط نوايا أصحاب المصلحة والتبعيات الاجتماعية، فإنها تفتقر إلى دلالات شكلية تتيح التحقق الدقيق. على العكس من ذلك، تعمل الأساليب الشكلية غالبًا على مستوى من التفصيل منفصل عن المنطق التنظيمي عالي المستوى. تقدم هذه الورقة منهجية جديدة لهندسة المتطلبات موجهة للعوامل لسد هذه الفجوة من خلال دمج النمذجة القصدية غير الشكلية الخاصة بإطار i* مع المواصفات القائمة على الحالة الذهنية الشكلية الخاصة بلغة تحديد المواصفات للعوامل المعرفية (CASL).ونقدم مخططات المبرر الاستراتيجي المشروح بالقصد(iASR) ، وهي امتداد لنماذج المبرر الاستراتيجي (SR) الخاصة بإطارi* ، معززة بشروحات التركيب والروابط لتوفير الدقة اللازمة للتعيين الشكلي. ونحدد تعيينًا منهجيًا من مخططات iASR إلى لغةCASL ، مما يمكن من التحليل الشكلي لأهداف العوامل ومعرفتها وتفاعلاتها. يتم إثبات فائدة الإطار المقترح من خلال دراسة حالة مفصلة لمنظم الجدولات، موضحة كيف يدعم التحقق من اتساق المتطلبات، وتحليل تفكيك الأهداف، والجدوى المعرفية. المساهمة الرئيسية هي توفير مسار سلس وقابل للتتبع من النماذج القصدية عالية المستوى إلى مواصفات العوامل القابلة للتحقق والتنفيذ.

الكلمات المفتاحية : هندسة المتطلبات، هندسة البرمجيات الموجهة بالعوامل، الأساليب الشكلية، إطارi* ، لغةCASL ، التحليل الموجه بالأهداف.

1. Introduction

Modern software systems are increasingly open, distributed, and complex, often modeled as Multiagent Systems (MAS) where autonomous agents interact to achieve their objectives. Requirements Engineering for such systems must capture not only *what* the system should do but also *why*, by understanding stakeholder goals and the organizational context [1, 2]. Goal-Oriented RE (GORE) approaches, such as i∗*i*∗ [3], excel at this by modeling actors, their goals, and the strategic dependencies between them.

However, i∗*i*∗ models are informal and lack the precision for rigorous analysis of properties like consistency, completeness, and the epistemic feasibility of agent plans. While some approaches have added formal analysis to i∗*i*∗ (e.g., Formal Troops [4] for model-checking SD models, or the i∗*i*∗-ConGolog framework [5] for animation), they abstract away the agents' goals and knowledge—the very concepts central to i∗*i*∗'s philosophy and agent-oriented systems.

This paper presents a methodology that overcomes these limitations by combining i∗*i*∗ with the Cognitive Agents Specification Language (CASL) [6]. CASL is a formal language based on the situation calculus that supports reasoning about agents' knowledge and goals, and includes ConGolog for specifying agent behavior.

Our main contributions are:

1. The Intentional Annotated Strategic Rationale (iASR) Diagram: An extension of i∗*i*∗'s SR diagrams with annotations to disambiguate process flows and add formal details.
2. A Formal Mapping to CASL: A set of rules to systematically transform iASR elements into a formal CASL specification, preserving intentional concepts.
3. Support for Mental State Reasoning: The mapped model allows for formal verification of agent goals, knowledge, delegation, and interaction protocols.
4. Introduction of Capabilities: A new construct to model guaranteed achievement of goals or performance of tasks under specified context conditions.
5. A Comprehensive Methodology: A guided process from early requirements (using i∗*i*∗) to late requirements and verification (using iASR and CASL).

2. Background & Related Work

2.1 The i∗*i*∗ Framework

The i∗*i*∗ framework [3] consists of two main models:

- Strategic Dependency (SD) Models: Represent a network of dependencies between actors (agents, roles, positions). A dependency involves a *defender*, a *depended*, and a *dedendum* (a Goal, Task, Resource, or soft goal). These models help analyze opportunities and vulnerabilities in an organization.
- Strategic Rationale (SR) Models: Explain the internal reasoning of each actor. They show how goals are decomposed into sub-goals/tasks (means-ends links) and how tasks are broken down (task-decomposition links), often using soft goals to evaluate alternatives.

2.2 The Cognitive Agents Specification Language (CASL)

CASL [6] integrates a theory of action and mental states (based on the situation calculus) with the ConGolog agent programming language. It provides predicates:

- Know (agt, φ, s): Agent *agt* knows formula φ in situation s.
- Goal (agt, ψ, s): Agent *agt* has goal ψ in situation s. Communicative actions like request and inform change these mental states. ConGolog provides high-level constructs (sequence, concurrency, non-determinism, interrupts) to specify complex agent behaviors.

2.3 Related Work

- Formal Troops [4]: Adds formal specification and model-checking to i∗*i*∗ concepts but uses a temporal logic that is not agent-oriented and does not support reasoning about goals as mental states.
- i∗*i*∗-ConGolog [5]: An inspiration for our work. It maps annotated SR diagrams to ConGolog for animation. However, it procedurally encodes goals and removes them from the agent's mental state, preventing formal reasoning about goal delegation and knowledge.
- KAOS [7]: A goal-oriented RE framework with formal analysis, but its goal decompositions are central and design-time, unlike the subjective, actor-centric, and runtime-friendly decompositions in our approach.

Our work distinctively uses CASL to retain goals and knowledge as first-class mental entities within the formal specification, enabling a deeper analysis of agent-oriented systems.

3. The Proposed Framework: Integrating i∗*i*∗ and CASL

The core challenge is bridging the abstraction gap between informal i∗*i*∗ models and precise CASL specifications. Our solution is a two-step process: first, refining SR models into iASR diagrams, and second, mapping them to CASL.

3.1 Intentional Annotated Strategic Rationale (iASR) Diagrams

iASR diagrams add precision to SR models through annotations:

- Composition Annotations: Specify how sub-elements are combined. They include:
  - ; (Sequence), || (Concurrency), >> (Prioritized Concurrency), | (Alternative).
- Link Annotations: Applied to individual decomposition links to specify execution conditions. They include:
  - if(condition), while(condition), for (variable, list), π (vars, condition) (pick),
  - guard(condition) (blocks until condition is true),
  - whenever (vars, trigger, cancel Condition) (interrupt).

Furthermore, iASR diagrams enforce specific constraints to streamline the mapping to CASL's declarative-procedural paradigm:

sjst.scst.edu.ly
Surman Journal for Science and Technology
ISSN: Online (2790-5721) - Print (2790-5713)
مجلة صرمان للعلوم والتقنية
Vol 6, No.1.Jan – May. 2024
Pages: 288 ~ 293

- Goal Nodes must be children of Task Nodes and connected via an interrupt or guard link annotation. This reflects how an agent's procedural behavior monitors its declarative mental state for newly acquired goals.
- Means-Ends Links can only connect a Goal (the end) to a Task or Capability (the means). This clarifies that goals are achieved by executing specific procedures.
-

3.2 Mapping iASR to CASL

We define a mapping function m that translates iASR elements into CASL constructs.

- Mapping Tasks: A leaf task maps to a primitive action or simple procedure. A complex task node is mapped to a ConGolog procedure built by applying the mapped composition and link annotations to the mappings of its subtasks.
- Mapping Goals: A goal node G is mapped to a tuple: m(G) = <Goal Formula, Achieve Proc>, where Goal Formula is a situation calculus formula defining the goal, and Achieve Proc is a ConGolog procedure that encodes the alternative means of achieving it (from its means-ends decomposition).
- Mapping Intentional Dependencies: A goal dependency from Actor A to Actor B is operationalized. A task in Actor A executes a request (A, B, eventually(φ)) action. In Actor B, a monitoring task uses an interrupt with a trigger condition like Goal (B, eventually(φ), s) ∧ Know (B, ¬φ, s) to execute Achiever for φ.
- Mapping Self-Acquired Goals: An agent can give itself a goal using a commit action, typically followed by a guard that waits for the goal to be adopted before trying to achieve it. This is used for modeling internal goal decompositions.

3.3 Modeling Capabilities

We introduce Capability Nodes to model behaviors that ar guaranteed to succeed if their context condition holds and the environment behaves as specified. A Goal Capability is mapped to a tuple <Goal Formula, Achieve Proc, Context Cond, Env ProcSpec>, with the formal guarantee: -
∀s. Context Cond(s) ⊃ AllDo (Subj (agt, Achieve Proc; Goal Formula?) || Env ProcSpec, s) This ensures all executions of the capability's procedure, under the specified conditions, will terminate successfully
with the goal achieved.

4. Case Study: Meeting Scheduler

We applied our methodology to a Meeting Scheduler system, a classic example in RE literature.

- Early Requirements (SD Models): We identified stakeholders (Meeting Initiator, Participant, Room Booking System) and their dependencies (e.g., Initiator depends on Scheduler for Meeting Scheduled).
- Late Requirements (iASR Models): We developed detailed iASR diagrams for each agent. For example, the Meeting Scheduler agent has a top-level task that monitors for the Meeting Scheduled goal via an interrupt. Upon acquisition, it decomposes into self-acquired sub-goals

like Available Dates Known and Participants Notified, each with their own alternative achievement procedures (e.g., emailing or phoning participants).

- Formal Specification & Verification: The iASR diagrams were mapped into a complete CASL specification. This model was then checked using the CASLve tool. We were able to verify properties such as:
    - Liveness: If a meeting is requested, the scheduler eventually attempts to schedule it.
    - Safety: A room is not double-booked.
    - Knowledge-based Correctness: The scheduler does not attempt to book a room before it knows the participants' availability.
    - Epistemic Feasibility: The plans for achieving sub-goals were checked to ensure the agent had the necessary knowledge to execute them (e.g., it knows the participant list before trying to notify them).

5. Results and Discussion

The case study demonstrated the practical benefits of our framework:

1. Formal Verification of Intentional Concepts: Unlike previous approaches, we could formally verify properties related to agent goals and knowledge, such as the correct propagation of goals via requests and the consistency of goal decompositions.
2. Improved Requirements Traceability: The direct mapping from iASR to CASL creates a clear traceability link from high-level stakeholder goals to detailed agent procedures, simplifying the impact analysis of requirements changes.
3. Support for Flexibility and Runtime Reasoning: By preserving goals as mental states in the formal model, our approach supports the design of systems where the choice between alternative means to achieve a goal can be deferred until runtime, increasing system adaptability.
4. Enhanced Confidence via Capabilities: The use of capabilities allows a designer to specify and verify critical components that are guaranteed to work under known conditions, improving the overall reliability analysis of the system.

The primary challenge observed was the increased complexity of iASR diagrams compared to standard SR models. However, this is a necessary trade-off for achieving formal precision, and tool support can mitigate this by automating parts of the annotation and mapping process.

6. Conclusion and Future Work

This paper presented a novel, integrated framework for agent-oriented requirements engineering that successfully combines the intuitive, high-level modeling of i*$i*$ with the rigorous, formal analysis of CASL. The introduction of iASR diagrams and a systematic mapping to CASL provides a concrete pathway from organizational modeling to verifiable multiagent system specifications.

Future work includes:

- Developing a software tool to automate the creation of iASR diagrams and their mapping to CASL.
- Extending the framework to handle more complex types of goals (e.g., maintenance goals) and belief revision.

- Exploring the compilation of capabilities into reusable, plug-and-play agent components.
- Conducting larger-scale case studies to further evaluate the scalability and effectiveness of the methodology.

Our approach provides a solid foundation for building complex, reliable agent-based systems where the rationale behind requirements is preserved and formally verified throughout the development process.

References

[1] van Lamsweerde, A. (2001). Goal-Oriented Requirements Engineering: A Guided Tour.

[2] Jennings, N. R. (1999). Agent-Oriented Software Engineering.

[3] Yu, E. (1995). Modelling Strategic Relationships for Process Reengineering.

[4] Fuxman, A., et al. (2001). Formal Tropos: Language and Semantics.

[5] Wang, X. (2001). Verifying Strategic Properties of Multi-Agent Systems with ConGolog.

[6] Shapiro, S., & Lespérance, Y. (2001). The Cognitive Agents Specification Language and Verification Environment.

[7] Dardenne, A., van Lamsweerde, A., & Fickas, S. (1993). Goal-Directed Requirements Acquisition.