Agents

Part I: What is an Agent?

A Conceptual Primer and History

Michael J Bommarito II · Jillian Bommarito · Daniel Martin Katz

November 18, 2025

Working Draft Chapter

Version 0.1

This chapter is Part I of a three-part series from the textbook *Artificial Intelligence for Law and Finance*, currently under development. Individual chapters are being drafted and refined independently before integration into the complete book. Part II: *How to Build an Agent* and Part III: *How to Govern an Agent* are forthcoming.

You can find the most current copy of the textbook project here:
`https://github.com/mjbommar/ai-law-finance-book/`

Contents

How to Read This Chapter

This chapter answers a deceptively simple question: *What is an agent?*

The term "agent" appears everywhere in industry and academia today—from "AI agents" to "agentic workflows"—yet rarely with clear definition. This creates confusion, miscommunication, and inevitably, disappointment.

We hope to help you avoid this disappointment, but **this chapter is not short and your time is valuable**. You don't need to read everything, especially on first reading. Three reading paths correspond to different goals:

Path 1: Get the working definition. If you need practical clarity fast, read Sections 1–2 and stop. You'll get the three-level hierarchy, the six operational properties, and a practical evaluation rubric. This is enough to use the term correctly, evaluate vendor claims, and participate in informed discussions. You can always return for deeper context later.

Path 2: Understand where this came from. If you want to understand why we define agents this way and how the concept evolved, continue through Section 4. You'll learn how agency emerged from 1950s philosophy through 1990s distributed systems to today's LLM-powered tools (Section 3), and why eight different disciplines—from law to cognitive science—emphasize different aspects of agency (Section 4). This historical and disciplinary context explains current debates and grounds our synthesis in established scholarship.

Path 3: Master the analytical framework. If you're writing research, crafting policy, or need comprehensive understanding, read everything. Sections 5–6 provide analytical dimensions (autonomy spectrum, entity frames, goal dynamics), formal specifications, boundary cases, and professional

deployment requirements. Section 7 synthesizes key takeaways and connects to subsequent chapters. This path gives you the theoretical foundations needed for rigorous analysis.

This chapter provides conceptual foundations and analytical frameworks. Part II (*How to Build an Agent*) covers architectures, protocols, and technical evaluation. Part III (*How to Govern an Agent*) addresses regulation, controls, and deployment.

---

1 Introduction

Agent. *Agentic.*

Few terms generate more confusion despite widespread use. While these words appear everywhere—from marketing copy to academic papers—their meanings remain contested and often unclear. Yet despite this definitional chaos, the underlying concepts are deeply intuitive and accessible.

At heart, agents are simply **"doers" with a to-do**. As we unpack this accessible starting point, we will discover more explicit conditions for identification. But this four-word formulation captures something essential: agency requires both goals and the capacity to act toward them.

1.1 Motivation and Approach

The proliferation of "agentic AI" makes definitional clarity urgent. Existing work remains fragmented across purpose and discipline: computer scientists cite Russell and Norvig (Russell and Norvig 2020), philosophers reference Bratman (Bratman 1987), legal scholars consult the Restatement of Agency (American Law Institute 2006), and commercial vendors seem untethered by anything other than sales.

Some of this fragmentation reflects genuinely different perspectives, such as whether we recognize agents by their *internal properties* (mental states, intentions) or *external manifestations* (observable behavior, delegated authority)—a spectrum we explore in Section 4. While theoretical considerations like these can be useful, it is now most critical that we **establish a practical framework** to guide communication and coordination.

The stakes for getting this right are tangible. In *Mata v. Avianca*, ChatGPT generated plausible but fictitious case citations that an attorney submitted in a court brief (Mata v. Avianca, Inc. 2023). The attorney relied on a single-shot text generator lacking the tools to search legal databases, the iteration to verify citations, and the mechanisms to escalate uncertainty. An agentic legal research system would have used search tools to validate citations, iterated to confirm case holdings, and escalated when unable to locate a source. The failure stemmed not from lack of sophistication

but from using a system without the operational properties required for reliable legal research. Distinguishing genuinely agentic systems from sophisticated chatbots is essential for professional practice, regulatory compliance, and client protection.

For legal and financial applications, the six operational properties are *necessary but not sufficient*. Professional deployment demands additional safeguards—attribution to authoritative sources, auditable provenance, escalation protocols, and confidentiality controls—that augment rather than replace the six-property framework. Section 6 addresses these professional deployment requirements in detail.

To do so, we organize agency into three levels that correspond to answering the following questions:

Level 1: What makes *something*—biological or computational—an agent?

Level 2: What makes computational systems agentic?

Level 3: How do traditional and AI-powered agentic systems differ?

Answering these progressive questions establishes a nested hierarchy with three levels, as illustrated in Figure 1.

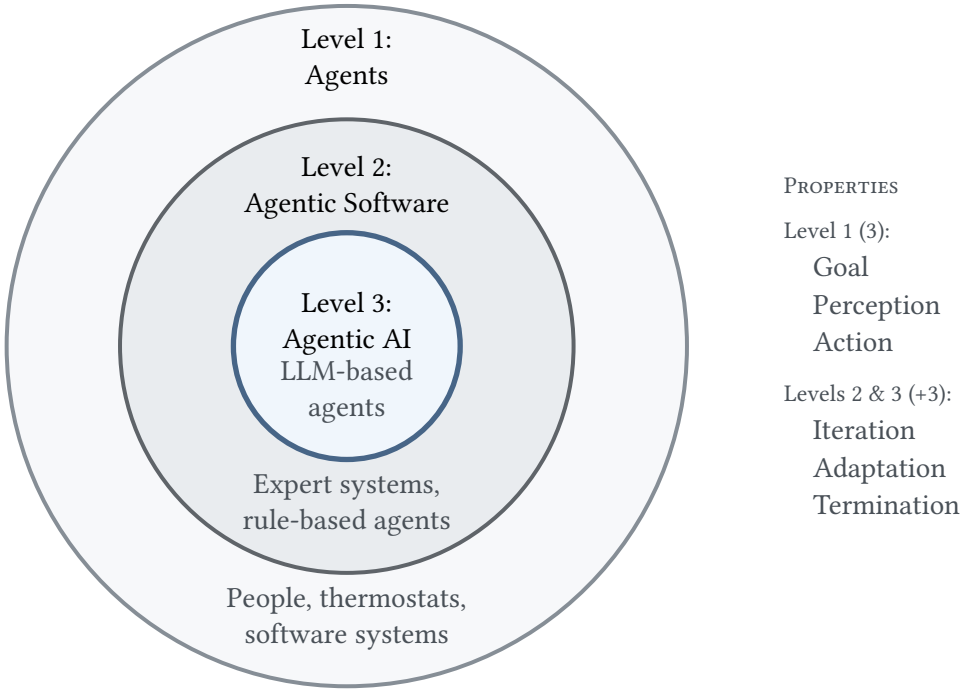

Figure 1: The three-level hierarchy of agency. Each level is a subset of the one above it: all agentic AI is agentic software, and all agentic software consists of agents.

1.2 Level 1: Minimal Agency

We begin with the conceptual foundation. What is the absolute minimum required for something to qualify as an agent—whether human, organizational, or computational? Level 1 establishes this

baseline, applicable across all domains and technologies.

Level 1: Agent (Mnemonic: GPA)

An **agent** is any entity that pursues goals through perception and action, with at least minimal discretion over which action to take in response to what it perceives.

Examples: A paralegal researching case law, a thermostat maintaining temperature, an organization pursuing objectives.

Minimal properties (3):

- **G**oal — Clear objective or performance criterion
- **P**erception — Awareness of environment through sensing
- **A**ction — Capability to affect environment

Property Definitions and Falsification Tests:

Goal (G): A clear objective, task specification, or performance criterion that directs behavior. Goals may be simple (maintain temperature) or complex (maximize portfolio returns), provided by external principals or internally generated, fixed or dynamic.

Falsification: If the entity responds identically regardless of desired outcomes, or transforms inputs mechanically without reference to success criteria, it lacks goals. Examples: compilers (execute predetermined transformations), pure lookup tables (no optimization target).

Perception (P): Awareness of environment through sensing capabilities—sensors, APIs, database access, document reading. The environment need not be physical; abstract spaces (contract negotiations, market data) qualify. *Legal example:* An agent perceives via EDGAR/Westlaw APIs to read regulatory filings and case law, observes which queries return hits, and uses those observations to refine subsequent searches.

Falsification: If the entity operates identically regardless of environmental state, or cannot observe consequences of its own actions, it lacks perception. Examples: open-loop controllers (no feedback), write-only systems.

Action (A): Capability to affect environment through actuators—physical forces, variable modifications, tool invocations, API calls, command execution. Requires minimal discretion: selecting among at least two possible actions contingent on perceptions.

Falsification: If the entity cannot modify its environment, or executes exactly one predetermined sequence regardless of circumstances, it lacks action. The discretion threshold is "$\geq$2 policies contingent on perceptions." Examples that fail: pure sensors (read-only), fixed scripts with zero conditional logic, fully deterministic single-action transformers that apply one complex transformation regardless of

input content (even if the transformation itself is sophisticated).

This trinity forms the conceptual bedrock of agency—equally applicable to humans navigating social contexts, organizations pursuing strategic objectives, biological organisms seeking survival, or computational systems executing tasks. While these three characteristics suffice for theoretical classification, practical deployment demands additional sophistication.

Distinguishing agents from non-agents reveals critical boundaries. Consider systems we encounter daily that, despite their complexity, fail to meet our criteria. A compiler, though it transforms code through sophisticated analysis, operates without autonomous goals—it executes predetermined transformations. Legal databases, while responsive to queries and rich with information, lack independent objectives or environmental adaptation. Even a single ChatGPT response, impressive as it may be, represents a one-shot generation rather than iterative goal pursuit.

This baseline framework illuminates the essence of agency, yet professional applications demand more. The gap between a simple thermostat cycling toward a temperature target and an AI system conducting legal research spans more than technological sophistication—it requires architectural elements that ensure reliability, adaptability, and accountability. Legal research tools, portfolio management systems, and document review platforms operate in environments where stakes are high and errors costly. These operational realities shape our expanded framework for deployable agentic systems.

1.3 Operational Definition: Agentic Systems

While Level 1 establishes what makes something an agent, computational systems in production require three additional properties beyond the minimal three. These six properties together define what we call **agentic systems**—the operational standard that bridges conceptual agency and real-world deployment. Both traditional software (Level 2) and AI-powered implementations (Level 3) can achieve this operational standard, though they differ fundamentally in how they realize each property.

Operational Definition: Agentic System (GPA+IAT)

An **agentic system** is a goal-directed agent that repeatedly perceives and acts in its environment, adapting from observations until clear termination conditions are met (explicit or implicit).

Examples: Legal research assistants that iteratively refine queries and generate case summaries. Contract analysis systems that identify risks and produce issue reports. Portfolio rebalancing systems that monitor positions and execute trades. Fraud detection systems that analyze transactions and generate alerts.

Additional properties beyond Level 1 (+3):

- **I**teration — Repeat perceive-act cycles, not single-shot
- **A**daptation — Adjust strategy based on feedback/results
- **T**ermination — Clear stopping conditions (explicit or implicit)

Total: 6 operational properties (Goal, Perception, Action + Iteration, Adaptation, Termination). For mnemonic convenience we write this as GPA + IAT: Goal (G), Perception (P), Action (A), Iteration (I), Adaptation (A again), Termination (T).

These six properties emerged from decades of agent research as commonly recognized operational requirements for reliable computational deployment. While not formally proven as minimum necessary, they consistently appear across deployed systems in domains from robotics to enterprise software, reflecting lessons from fielding real-world implementations. Section 3 traces how each property became recognized as essential through both theoretical development and practical experience.

Additional Property Definitions and Falsification Tests:

Iteration (I): Multiple perceive-act cycles with state preservation across rounds. The entity repeatedly gathers information, takes action, observes results, and continues—not single-shot processing. Crucially, the system must perceive outcomes of prior actions and update subsequent actions accordingly within the same goal pursuit.

Falsification: If the entity processes input once and produces output without maintaining state across cycles, it lacks iteration. Merely repeating the same action without incorporating new observations doesn't qualify. *Batching vs. iteration:* Batched one-pass pipelines processing multiple items sequentially are not iteration unless the system observes outcomes from earlier items and modifies its approach for later items based on those observations. Examples that fail iteration test: single ChatGPT response (one-shot), batch processors applying identical logic to each item without inter-item learning.

Adaptation (A): Strategy modification based on accumulated observations and feedback within

a session or task.[1] The entity adjusts its approach when initial attempts fail, learns which actions succeed, updates its policy based on results.

Falsification: If the entity applies identical logic regardless of outcomes, or cannot modify its approach when initial strategies fail, it lacks adaptation. Fixed rules that never change based on results don't qualify, even if they handle diverse inputs. Examples: basic thermostats (fixed on/off rules), static pattern matchers, rigid workflows that don't adjust to failures.

Termination (T): Clear stopping conditions ensuring bounded operation. Termination may be *implicit* (goal satisfaction, reaching target state, exhausting search space) or *explicit* (resource budgets, time limits, maximum iterations, escalation triggers, confidence thresholds).

Falsification: If the entity has no mechanism for recognizing when to stop, or could cycle indefinitely without bounds, it lacks proper termination. Entities requiring external intervention to halt don't meet this criterion. Examples: infinite loops with no exit condition, systems that run until manually killed.

The relationship between levels clarifies important boundaries. Every agentic system qualifies as an agent (possessing the minimal three properties), but the reverse doesn't hold—many agents lack the operational sophistication of agentic systems. A basic mechanical thermostat illustrates this gap: it has five properties (goal: maintain temperature; perception: sensor readings; action: heating/cooling; iteration: continuous monitoring; termination: implicit when target reached), but lacks adaptation—it applies fixed on/off rules without modifying strategy based on outcomes. It responds to temperature changes reactively but doesn't learn patterns or adjust thresholds. In contrast, a *smart* thermostat qualifies as a full agentic system: it learns occupancy patterns, adjusts heating schedules based on observed behavior, and modifies its strategy when energy costs spike. This distinction between reactive control (basic thermostat) and adaptive learning (smart thermostat) clarifies why minimal agency (Level 1, three properties) differs from operational agentic systems (six properties). Similarly, a human paralegal demonstrates all six properties behaviorally but operates through cognitive processes rather than discrete computational cycles.

This operational framework now raises the implementation question: *How* do computational systems realize these six properties? The answer reveals an architectural distinction. Some systems use traditional programming—rules, algorithms, control logic—to manage planning and orchestration. Others employ AI/ML, particularly large language models, for these functions. We distinguish these as Level 2 (traditional) and Level 3 (AI-powered). Critically, this distinction is architectural, not evaluative: neither approach is inherently superior, and the boundary between them remains fluid and context-dependent.

[1]This definition focuses on *session-level adaptation*—modifying behavior within a single task execution based on immediate feedback. This differs from *cross-session learning*, where a system improves through model retraining across multiple tasks (for example, fine-tuning an LLM on user feedback). For our purposes, within-session adaptation is the operational property that distinguishes agentic systems from static tools. In professional contexts (such as procurement or vendor evaluation), be explicit about which type of learning is in scope.

1.4 Level 2: Traditional Agentic Software

Level 2 represents the first computational instantiation of agentic systems. These systems achieve all six operational properties through explicit programming—rules, conditional logic, algorithms, and control flow. What defines Level 2 is runtime behavior: decisions flow through programmed logic paths rather than learned models. Whether a system was coded decades ago or yesterday, if its decision-making follows deterministic rules at runtime, it operates at Level 2.

Level 2: Traditional Agentic Software

Traditional agentic software implements the six operational properties using rules, algorithms, or deterministic logic.

Examples: Conflicts checking systems that scan firm databases and escalate matches. Regulatory compliance monitoring with rule-based alerts. Trading compliance systems that monitor positions and enforce limits. Invoice processing workflows with validation and retry logic.

Additional properties beyond agentic systems (+0):

- *No new properties — same 6 as agentic systems*

Traditional agentic software uses rules, algorithms, or deterministic logic to implement all six properties. Planning and orchestration are explicitly coded through conditional logic, state machines, or control systems. Level 2 systems can be extremely sophisticated: a conflict checking system might employ graph analysis for relationship detection, fuzzy matching for entity resolution, and adaptive threshold tuning—all implemented through traditional programming techniques. The key architectural characteristic is that decision logic is specified by programmers at design time.

A conflicts checking system exemplifies Level 2: it has goals (identify potential conflicts), perception (scans firm databases for client/matter relationships), action (flags matches and escalates to ethics committee), iteration (continuous monitoring as new matters are opened), adaptation (adjusts matching thresholds based on false positive rates), and termination (stops when matter is cleared or rejected). The entity achieves these properties through explicitly programmed logic—rules for relationship detection, graph traversal algorithms for indirect conflicts, configurable thresholds for fuzzy name matching.

Level 2 systems achieve all six operational properties through traditional software engineering. Their performance depends on design quality, domain expertise, and implementation rigor—not on whether they employ AI/ML techniques. A well-engineered Level 2 system can substantially outperform a poorly designed Level 3 system in reliability, predictability, and effectiveness for its intended domain.

1.5 Level 3: AI-Powered Agentic Systems

Level 3 systems use AI/ML—particularly large language models—to manage planning, orchestration, and adaptation. The architectural distinction from Level 2 is straightforward: where Level 2 systems execute explicitly programmed decision logic, Level 3 systems employ neural networks (especially LLMs) or other learned models for these functions. In practice, modern Level 3 systems are typically hybrid: LLMs handle high-level planning and natural language interaction, while traditional code manages structured operations like database queries or API calls. This is what most practitioners mean by "AI agents."

Level 3: AI-Powered Agentic Systems

AI-powered agentic systems implement the six operational properties through strategic integration of artificial intelligence—typically neural network models such as large language models (LLMs) and vision-language models (VLMs)—with traditional computational components.

Examples: AI legal research assistants that iteratively search case law and synthesize findings. AI contract risk analyzers that identify problematic clauses and recommend revisions. AI-powered document review systems that classify content and adapt to feedback. AI trading assistants that analyze market data and adjust strategies.

Additional properties beyond agentic systems (+0):

- *No new properties — same 6 as agentic systems*

The architectural choice to use AI/ML for planning and orchestration has practical implications. LLMs enable natural language interfaces—users can specify goals conversationally rather than through structured formats. These neural network models handle pattern recognition tasks that would require extensive rule engineering. However, this approach trades some predictability for flexibility: LLM outputs can vary across runs, and behavior may be harder to audit than explicit rule chains. The boundary between Level 2 and Level 3 can blur—is a system using gradient boosting for fraud detection Level 2 or Level 3? Today's practical dividing line focuses on whether LLMs manage high-level planning and orchestration.

Consider an AI contract risk analyzer as a Level 3 exemplar. It possesses all six operational properties: goals (identifies and assesses contract risks), perception (reads contract text and clause context), action (flags problematic provisions, generates risk assessments), iteration (reviews document section by section), adaptation (adjusts risk scoring based on clause combinations and jurisdiction), and termination (stops when complete review is done or high-severity risk triggers immediate escalation). The LLM manages high-level planning—deciding which clauses merit detailed analysis, how to interpret ambiguous language, when to flag issues versus when to request human review—while traditional code handles document parsing, clause extraction, jurisdiction lookup, and risk score

calculation. This hybrid architecture is typical of Level 3 systems: AI handles interpretation and strategic decisions, traditional programming handles structured data operations.

Table 1 maps the progression from minimal agency (3 properties) through agentic systems (6 properties) to implementation paradigms (traditional vs. AI).

Property	Agent	Agentic Systems		Description
	Level 1	Level 2	Level 3	
	Conceptual	Traditional	AI	
Goal	●	●	●	Clear objective to pursue
Perception	●	●	●	Sense environment & results
Action	●	●	●	Affect environment
Iteration	○	●	●	Repeat perceive-act cycle
Adaptation	○	●	●	Adjust based on feedback
Termination	○	●	●	Stopping condition (explicit or implicit)
AI-Powered	○	○	●	Uses AI/ML for implementation

Table 1: Property requirements by level. Filled circles indicate required properties: blue (filled) for minimal agency (Level 1), amber (filled) for additional operational properties (Levels 2 & 3). Empty circles indicate properties not required. Below the horizontal rule: green (filled) indicates AI-powered implementation (Level 3 only). *Accessibility:* Table has 5 columns (Property, Level 1 Agent, Level 2 Traditional, Level 3 AI, Description) and 8 rows. First three rows (Goal, Perception, Action) required at all levels with filled blue circles. Rows 4–6 (Iteration, Adaptation, Termination) required only for Levels 2–3 with filled amber circles, empty for Level 1. Final row (AI-Powered) shows empty circles for Levels 1–2, filled green for Level 3.

While Levels 2 and 3 share identical property requirements, they differ fundamentally in *how* each property is implemented. Table 2 contrasts implementation approaches.

1.6 Key Distinctions

Having traced the progression from minimal agency (3 properties) through agentic systems (6 properties) to implementation paradigms (traditional vs. AI), we can now synthesize what this hierarchy reveals. Level 1 establishes conceptual qualification, the operational definition adds production-readiness requirements, and Levels 2–3 distinguish implementation paradigms. This structure clarifies three critical distinctions that cut through definitional confusion:

The critical distinction lies between *properties* and *implementation*. The major jump in capability occurs between Level 1 (3 properties) and agentic systems (6 properties). Levels 2 and 3, by contrast, have *identical property requirements*—they differ only in how those properties are implemented: through explicit rules and logic (Level 2) or through AI/ML models (Level 3).

This hierarchy enables precise terminology:

Property	Level 2 (Traditional)	Level 3 (AI-Powered)
Goal	Hardcoded objectives, config files, explicit targets	Natural language instructions, conversational goal specification
Perception	Structured APIs, SQL queries, regex parsing	LLM document understanding, vision-language models, semantic search
Action	Function calls, database writes, API invocations	Tool orchestration via LLM function calling, generated code execution
Iteration	Control loops (while/for), state machines, workflow engines	LLM manages reasoning loop, decides next tool based on context
Adaptation	Rule adjustments, threshold tuning, A/B test results	Chain-of-thought reasoning, prompt-based strategy revision, in-context learning
Termination	Explicit conditionals (max iterations, timeout timers)	LLM decides task complete based on goal satisfaction, budgets enforced externally
Planning	Decision trees, rule engines, optimization algorithms	LLM generates plans, ReAct-style reasoning traces
Logs	Structured application logs, database audit trails	Natural language reasoning chains, tool invocation histories

Table 2: Implementation contrast between traditional (Level 2) and AI-powered (Level 3) agentic systems. Both satisfy identical property requirements; the architectural distinction lies in implementation mechanisms. *Accessibility:* Comparison table with 3 columns (Property, Level 2 Traditional, Level 3 AI-Powered) and 8 rows. Rows 1–6 compare the six core properties: Level 2 uses hardcoded objectives, structured APIs, function calls, control loops, rule adjustments, and explicit conditionals; Level 3 uses natural language instructions, LLM understanding, tool orchestration, reasoning loops, chain-of-thought, and external budgets. Rows 7–8 compare planning (decision trees vs. LLM-generated plans) and logs (structured vs. natural language reasoning traces).

Terminology Precision

- **"Agent" (noun):** Anything that exhibits Level 1's three minimal properties.
- **"Agentic" (adjective):** At the system level, describes systems meeting all six operational properties. We may also use it descriptively at the feature/behavior level (e.g., "agentic behavior/properties"); reserve "agentic system" for six-of-six conformance.
- **"AI agent":** An agentic system specifically powered by AI/ML (Level 3).

These definitions enable clear exclusions:

What Doesn't Qualify

- Compilers, databases → lack goals (not Level 1)
- Single chatbot responses → lack iteration (not operational)

- Traditional ML classifiers → lack iteration and goals (not agentic)
- Rule-based expert systems → agentic but not AI-powered (Level 2, not Level 3)

This framework provides scaffolding for the historical and theoretical analysis that follows. Section 2 provides a practical decision rubric for immediate application. The remaining sections trace where these definitions came from and why they take this particular form, building toward the professional implications explored in Section 7.

2 How to Recognize an Agent

With the three-level hierarchy and complete definition established in Section 1, you now have the conceptual foundation. But recognizing agents in practice requires operational tools. This section provides practical approaches: a detailed evaluation rubric, concrete examples comparing agents to non-agents, and guidance for navigating common misconceptions.

2.1 The 6-Question Evaluation Rubric: Operational Properties

These six questions assess whether an entity qualifies as an *agentic system*. They operationalize the six-property definition established in Section 1. The first three (Goal, Perception, Action) establish *Level 1 agency*; Questions 4–6 add the operational properties required for deployment.

Q1. Does it have goals (G)?

Test: Look for clear objectives, task specifications, or performance criteria that direct behavior.
Domain examples: Legal research goal (find relevant precedents), portfolio goal (maximize risk-adjusted returns), contract analysis goal (identify liability clauses).
Falsification: If it responds identically regardless of desired outcomes, it lacks goals. Examples: compilers, lookup tables.

Q2. Does it perceive (P)?

Test: Check for environmental awareness through sensors, APIs, databases, or document access. Can it observe the results of its own actions?
Domain examples: Read case law databases, monitor market data feeds, parse contract text, access regulatory filings.
Falsification: If it operates identically regardless of environmental state, or cannot observe action consequences, it lacks perception. Examples: write-only systems, open-loop controllers.

Q3. Does it act (A)?

Test: Verify capability to affect environment with at least minimal discretion (choosing among two+ actions contingent on perceptions).
Domain examples: Generate legal briefs, execute trades, flag contract risks, file regulatory reports, invoke research tools.
Falsification: If it cannot modify its environment, or executes exactly one fixed sequence, it lacks action. Examples: pure sensors, zero-conditional scripts.

Q4. Does it iterate (I)?

Test: Confirm multiple perceive-act cycles with state preservation across rounds, not single-shot processing. Does it loop: act → observe → act again?
Domain examples: Iteratively refine legal queries based on results, adjust portfolio based on market moves, review contract sections sequentially.
Falsification: If it processes input once without maintaining state across cycles, it lacks iteration. Examples: single ChatGPT response, batch processors without inter-item updates.

Q5. Does it adapt (A)?

Test: Check whether strategy modifies based on accumulated observations and feedback. Does it adjust approach when initial attempts fail?
Domain examples: Adjust search terms when queries return no results, modify risk thresholds based on false positive rates, change trade strategies when volatility spikes.
Falsification: If it applies identical logic regardless of outcomes, or cannot modify approach when strategies fail, it lacks adaptation. Examples: basic thermostats (fixed rules), static pattern matchers, rigid workflows.

Q6. Does it stop/terminate (T)?

Test: Verify clear stopping conditions—implicit (goal satisfaction, exhausted search) or explicit (time limits, resource budgets, escalation triggers, confidence thresholds, maximum iterations).
Domain examples: Stop when finding relevant precedent, escalate when confidence is low, halt after maximum API calls, terminate when portfolio balanced.
Falsification: If it has no mechanism for recognizing when to stop, or could cycle indefinitely, it lacks termination. Examples: infinite loops, systems requiring manual kill.

If Q1–Q3 are yes, the entity qualifies as an agent. If all six answers are yes, it qualifies as an agentic system. Otherwise, it may be a useful tool but lacks full agentic character.

Note on Professional Governance: The six questions above establish whether something is an

agent. Professional deployment in high-stakes domains (legal research, medical diagnosis, financial advising) requires additional governance safeguards—attribution, explanation, escalation, and confidentiality. These professional governance requirements are covered in Part III (*How to Govern an Agent*).

2.2 Common Misconceptions

Understanding what agents are requires equally understanding what they are *not*. Five common misconceptions lead to over-attribution of agency.

Single-Shot Responses Are Not Agentic Systems. A single response from a language model is not an agentic system, even if sophisticated. Without iteration and adaptation, it's a one-time transformation. Sending one query to ChatGPT and receiving one response demonstrates this limitation—no iteration, no adaptation to results, no perception-action loop. The same research system could be agentic if it iteratively refined queries based on initial results, but a single exchange lacks the fundamental properties required.

Automation Alone Is Not an Agentic System. Automation doesn't imply agency. A data extraction script that automatically runs nightly has goals (extract data) and acts (writes to database), but lacks perception, adaptation, and iteration. It executes a fixed sequence on a schedule. It's a scheduled task, not an agentic system. The automation trigger doesn't create the perception-action loop or adaptive behavior that define agentic systems.

Tool Use Alone Is Not Agency. Calling external tools doesn't automatically make an entity agentic. The critical question is whether the entity iterates on tool results, adapting its strategy based on what it observes. A script that queries an API once is not an agent—it lacks iteration and adaptation. A research system that queries the API, evaluates relevance, and decides what to search next based on results may qualify, depending on whether it completes the perception-action-adaptation cycle.

Complexity Is Not Agency. Complex entities aren't necessarily agents. A document processing pipeline performs sophisticated transformations—optical character recognition, entity extraction, format conversion—but follows a fixed sequence without goals, perception of results, or adaptation. Complexity measures sophistication, not agency. Conversely, simple entities can be agents. A basic thermostat has goals (maintain temperature), perceives (reads sensor), acts (turns heat on/off), iterates (continuous monitoring), and terminates (implicit when target reached), though it lacks adaptation through its fixed on/off rules. Simplicity and agency are orthogonal dimensions.

AI-Powered Is Not Agency. Using AI or machine learning doesn't make something an agent. A neural network that classifies documents in a single forward pass—input to output—is not an agent, regardless of model sophistication. It lacks iteration, adaptation, and the perception-action loop. An AI entity that iteratively reviews content, flags issues, refines assessments based on observed patterns, and adapts its classification strategy across multiple cycles can be an agentic system. The AI component enables flexible reasoning, but agency requires the architectural properties, not merely

the presence of neural networks.

2.3 Examples: Agents vs Non-Agents

Table 3 orders entities along the spectrum of agency, from non-agents through AI-powered agents. Each tier illustrates increasing qualification based on the six core properties. Detailed explanations follow.

System	Status	Key Properties
Not Agents (Missing Critical Properties)		
Form validation script	No	One-pass validation; lacks goals, perception, and adaptation
Agents with Incomplete Agentic System Properties (Missing 1–2)		
Single query to research system	Partial	Has goal, perception, and action; lacks iteration and adaptation
Rule-based pattern detection	Partial	Iterates continuously but lacks adaptive strategy
Content suggestion system	Partial	Updates per interaction but uses fixed learning approach
Full Agents: Traditional (Rule-Based)		
Smart thermostat	Yes	All 6 properties; learns patterns and adapts behavior
Portfolio rebalancing system	Yes	Monitors, adapts to volatility, stops when balanced
Full Agents: AI-Powered (Flexible Reasoning)		
Document review system	Yes	Iterative classification with adaptive learning
AI research assistant	Yes	Tool loop with query refinement and escalation

Table 3: Spectrum of agency ordered by increasing qualification, from non-agents through AI-powered systems. *Accessibility:* Classification table with 3 columns (System, Status, Key Properties) and 9 rows in 4 groups. Group 1 (gray background): non-agents lacking critical properties. Group 2 (amber background): agents with incomplete agentic system properties (missing 1–2 of the six). Group 3 (blue background): full traditional agents with all 6 properties via rules/algorithms. Group 4 (green background): full AI-powered agents with all 6 properties via LLMs/neural networks.

Not Agents. These entities lack the minimal properties (goals, perception, action). The **form validation script** checks inputs against rules in a single pass—it has no independent goals beyond validation, no perception of results, and no adaptation.

Agents with Incomplete Agentic System Properties. These entities qualify as agents (they exceed the 3-property minimum for Level 1 agency) but lack one or two of the six operational properties required for full agentic systems. **Rule-based pattern detection** has a goal (flag patterns), perceives incoming data, acts (blocks or alerts), and iterates continuously with each input. However, it typically doesn't adapt its detection strategy within a session—it applies fixed rules. **Content suggestion systems** similarly have a goal (suggest relevant content), perceive context, act (display suggestions),

and iterate with each interaction. Yet they use a fixed learning strategy rather than adapting their suggestions based on user acceptance patterns within a session.

Full Agents: Traditional. **Thermostats** illustrate the boundary case. Basic mechanical thermostats have five clear properties: goal (maintain temperature), perception (sensor readings), action (heat on/off), iteration (continuous monitoring), and termination through implicit goal satisfaction (stops heating when target reached). However, they lack true adaptation—they apply fixed rules without adjusting strategy based on results. **Smart thermostats** qualify as full agentic systems: they learn occupancy patterns, adjust heating schedules based on observed behavior, and implement explicit termination logic like time windows, energy budgets, or away modes. The contrast illustrates how reactive control (basic thermostats) differs from adaptive learning (smart thermostats). **Portfolio rebalancing systems** demonstrate explicit termination through multiple stopping conditions: goal achievement (balanced portfolio), temporal constraints (market close), or resource limits (maximum trades per session). They monitor market data and portfolio drift, generate trade orders, adapt to volatility within predefined parameters, and stop when any termination condition is met.

Full Agents: AI-Powered. **Document review systems** use machine learning to iteratively classify documents based on relevance criteria. Unlike rule-based entities, they improve their categorization as they process more examples, adapting their classification strategy based on observed patterns. **AI research assistants** demonstrate the most sophisticated agency: they maintain goals (answer questions), observe search results from multiple sources, query iteratively, refine search strategies when initial approaches fail, and escalate to human oversight when encountering contradictory information or reaching confidence limits. This represents the convergence of all six properties enhanced by AI's flexible reasoning capabilities.

2.4 When to Call Something an Agent

Our taxonomy (Section 1) distinguishes "agent" (3 minimal properties) from "agentic system" (6 operational properties). Table 4 provides practical guidance for applying these terms.

Precision requirements vary by context. Regulatory filings and academic papers demand explicit enumeration of properties and clear distinction between agent and agentic system. Informal discussions permit looser usage—"agent" for entities meeting the 3-property baseline, with specific properties noted when relevant: "has goals and perception but limited adaptation." Marketing claims should specify which properties are present rather than using "agentic AI" without substantiation.

When in doubt, use the 6-question rubric (subsection 2.1). It provides clear, answerable criteria that cut through ambiguity.

Properties	Term	When to Use	Notes
3 (G+P+A)	**agent**	Meets baseline	Includes thermostats, simple systems
6 (all)	**agentic system**	Production-ready	Required for professional deployment
6 via AI/ML	**agentic AI**	AI-powered	AI implements the six properties
4-5	agent with...	Partial implementation	Specify which properties present
1-2	(avoid "agent")	Insufficient	Use "tool," "function," "classifier"

Table 4: Terminology usage based on property count. G=goals, P=perception, A=action. *Accessibility:* Decision table with 4 columns (Properties, Term, When to Use, Notes) and 6 rows. Row 1: 3 properties (G+P+A) = agent (baseline). Row 2: 6 properties = agentic system (production-ready). Row 3: 6 via AI/ML = agentic AI. Row 4: 4–5 properties = agent with specific properties noted. Row 5–6: 1–2 properties insufficient for "agent" label, use "tool/function/classifier" instead.

Key Takeaways

At this point you can:

- Define agents and agentic systems using the 6-property framework
- Recognize agents using the 6-question rubric
- Distinguish agents from sophisticated non-agents
- Evaluate marketing claims about "agentic AI"
- Use terminology correctly in professional contexts

Remaining sections provide: Historical evolution (Section 3), disciplinary perspectives (Section 4), analytical dimensions (Section 5), and synthesis (Section 7).

3 A Historical Journey: Defining Agents Across Seven Decades

Optional Section: Skip if Time-Pressed

You now have working definitions and practical evaluation tools. This section provides optional historical context—tracing how agency evolved from 1950s philosophy through 1990s distributed systems to 2020s LLM agents.

Skip to Section 7 if:

- You need immediate application guidance (use the 6-question rubric from Section 2)
- You're satisfied with the three-level hierarchy and six properties
- You don't need to understand *why* definitions vary across disciplines

Read this section if:

- You're writing research or policy requiring theoretical grounding
- You need to understand disciplinary differences (law vs. economics vs. AI)
- You want to trace how each property emerged historically

The concept of **agency** has evolved dramatically since the mid-20th century, shaped by philosophical inquiry, economic theory, legal doctrine, and technological capability. This section traces agent definitions chronologically, revealing how each level of the hierarchy from Section 1 emerged historically: philosophical and legal foundations addressed general agency (Level 1), the computer science revolution instantiated these concepts in software (Level 2), and the LLM era added flexible reasoning powered by AI (Level 3). Figure 2 provides a visual overview of key milestones.

On Etymology. The semantic complexity surrounding "agent" traces back to the word's origins. The Latin root *agō*, from which *agent* derives, had multiple meanings in classical Latin. With numerous distinct senses in Lewis & Short's classical dictionary, ranging from "driving cattle" to "pleading a case," contemporary definitional debates echo longstanding ambiguity (Lewis and Short 1879).

3.1 Philosophical and Legal Foundations (1957–1969)

The postwar period witnessed foundational philosophical work on action theory alongside the maturation of legal doctrine concerning agency relationships. These developments occurred independently yet addressed complementary questions: philosophy examined how intention relates to action and how we attribute actions to agents, while law formalized the conditions under which one party acts on behalf of another.

1957: Intention

G.E.M. Anscombe's *Intention* established that actions are **intentional** "under a description" and known through practical knowledge (Anscombe 1957). This introduced the idea that intention plays a basic explanatory role: we understand actions by understanding the reasons under which agents perform them.

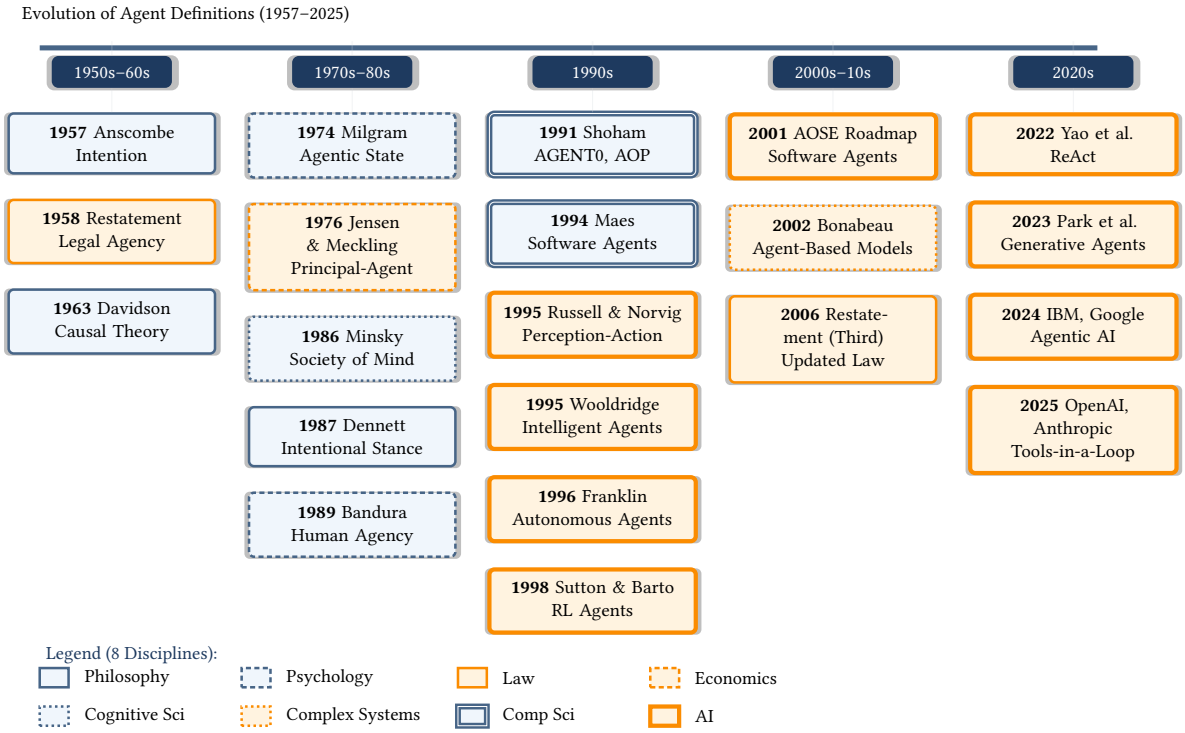

Figure 2: Timeline of key milestones in agent definitions across eight disciplines (1957–2025). The figure traces the intellectual evolution of "agency" from philosophical accounts of intentionality (1950s–80s) through the computational revolution (1990s) and toward modern convergence around LLM-based agentic systems (2020s). Each discipline has a distinct visual marker combining border patterns (solid, dashed, dotted, double, thick) with color families: blue for theoretical foundations (philosophy, psychology, cognitive science, computer science) and orange for applied domains (law, economics, complex systems, AI engineering).

1958: Legal Agency

The *Restatement (Second) of Agency* defined **agency relationship**: a principal manifests assent that an agent shall act on the principal's behalf and subject to control (American Law Institute 1958). This emphasizes delegation, fiduciary relation, and control—concepts relevant when discussing AI systems acting on behalf of humans.

1963: Causal Theory of Action

Donald Davidson's "Actions, Reasons, and Causes" defended the **causal theory of action**, arguing that intentional actions are explained by an agent's primary reason (belief–desire pair) that causally produces the action (Davidson 1963). This framework connected mental states to observable behavior.

These three works established complementary foundations for thinking about agency. Philosophy offered frameworks for intention and the causal structure of action, while law formalized delegation and control in principal–agent relationships. Together, they set up core tensions that would animate later debates: the relationship between mental states and observable behavior, the balance between autonomy and control in delegated relationships, and the explanatory role of reasons versus causes. When researchers began implementing computational agents in subsequent decades, they inherited both the conceptual resources and these unresolved tensions—questions that remain live today when we ask whether AI systems truly have intentions or merely simulate goal-directed behavior.

3.2 Economic and Social Theory (1970–1989)

The 1970s and 1980s witnessed agency concepts migrating from philosophy and law into the social sciences. Economics, sociology, and psychology each adapted the core ideas to their disciplinary concerns, generating new frameworks for understanding human behavior in organizations, social structures, and economic relationships. This period produced a remarkably diverse set of perspectives on agency, from Milgram's minimalist "agentic state" where individuals function as mere instruments, to Bandura's rich conception of human agency with intentionality, forethought, and self-regulation.

1974: The Agentic State

Stanley Milgram's *Obedience to Authority* introduced the "**agentic state**"—individuals seeing themselves as instruments carrying out another's wishes (Milgram 1974). This anchors one end of the autonomy spectrum: agents as pure delegates without independent judgment.

1976: Principal-Agent Economics

Jensen and Meckling's "Theory of the Firm" formalized the **principal-agent relationship**, defining it as a contract where principals engage agents with delegated decision-making authority (Jensen and Meckling 1976). This established vocabulary of agency costs, information asymmetry, and incentive alignment.

1984: Structuration Theory

Anthony Giddens framed agency as the capacity to make a difference and intervene in the world (Giddens 1984).

1986: Society of Mind

Marvin Minsky's *Society of Mind* popularized the idea of mind as a society of simple "agents" whose interactions generate intelligence, introducing the **multi-agent system** metaphor (Minsky 1986).

1987: Planning Theory

Michael Bratman's *Intention, Plans, and Practical Reason* developed a planning theory emphasizing partial plans structuring practical reasoning (Bratman 1987). This introduced the Belief-Desire-Intention (BDI) framework that became foundational for intelligent agent architectures.

1987: The Intentional Stance

Daniel Dennett's *The Intentional Stance* argued we can treat entities as agents when attributing beliefs and desires yields reliable predictions—providing a pragmatic criterion for agency (Dennett 1987).

1989: Human Agency

Albert Bandura synthesized psychological research, defining human agency through intentionality, forethought, self-regulation, and self-reflectiveness—properties increasingly sought in artificial systems (Bandura 1989).

These seven works collectively established a spectrum of autonomy that would prove essential for understanding computational agents. At one extreme, Milgram's agentic state described humans functioning as mere instruments executing another's directives—agents as pure delegates without independent judgment. At the other extreme, Bandura enumerated the rich capacities of human agency: forming intentions, planning ahead, monitoring and regulating behavior, and reflecting on one's own efficacy. Between these poles, each discipline tackled different facets of the agency problem. Economics formalized principal-agent relationships as contracts with delegated decision-

making but misaligned incentives, focusing on control mechanisms and information asymmetries. Sociology examined how agents act independently within structural constraints. Cognitive science explored how minds might emerge from societies of simpler agents. Philosophy developed practical frameworks for planning, prediction, and attributing mental states.

This diversity reflected genuine differences in disciplinary focus, but the frameworks proved complementary rather than contradictory. When researchers began implementing computational agents in the 1990s, they inherited this rich conceptual toolkit and translated it into software architectures. Bratman's BDI framework became the dominant architecture for intelligent agents. Dennett's intentional stance provided a pragmatic criterion for when to treat systems as agents—when doing so yields reliable predictions. Bandura's enumeration of human agency properties became a checklist for evaluating artificial systems. The spectrum from minimal delegation to full autonomy helped designers understand where their systems fell and what capabilities they still needed to develop.

3.3 The Computer Science Revolution (1990–1999)

The 1990s transformed agency from philosophical concept to computational reality. Previous decades had provided rich conceptual frameworks—intention, planning, autonomy, perception-action cycles—but these primarily described human or abstract theoretical entities. The convergence of distributed systems research, AI advances, and the explosive growth of the internet created conditions for a decisive shift: researchers began implementing agents directly in software, creating systems designed from the ground up as autonomous entities rather than traditional programs.

This transition posed fundamental translation challenges. How should beliefs and intentions be represented computationally? What distinguishes an autonomous agent from a reactive program? When do interactions among simple agents produce emergent intelligence? The decade's answers emerged through diverse implementations, each formalizing different aspects of the prior decades' conceptual toolkit. By decade's end, agent-oriented computing had established itself as a distinct paradigm with its own methods, architectures, and standards.

1991–1993: Agent-Oriented Programming

Yoav Shoham's AGENT0 and "Agent-Oriented Programming" implemented agents as computational entities characterized in mentalistic terms (beliefs, commitments, obligations) (Shoham 1993). This established **agent-oriented programming** as distinct from object-oriented programming and was closely related to the BDI (Belief-Desire-Intention) architecture.

1994: Learning Interface Agents

Pattie Maes pioneered **learning interface agents**—adaptive software assistants that reduce work and information overload by learning user preferences through observation, feedback, and collaborative filtering (Maes 1994). Her work on systems like Ringo (music recommendations) demonstrated personalization through learning, introducing adaptation as a core agentic capability (Shardanand and Maes 1995).

Three major 1995 publications crystallized distinct approaches to defining computational agents, each addressing the translation challenge from a different angle:

1995: Intelligent Agents

Wooldridge & Jennings: **Intelligent agents** exhibit autonomy, reactivity, proactivity, and social ability (Wooldridge and Jennings 1995).

1995: Perception–Action Agents

Russell & Norvig: An **agent** perceives its environment through sensors and acts through actuators (Russell and Norvig 1995).

1995: Complex Adaptive Systems

Holland: **Complex adaptive system agents** exhibit local interactions under simple rules that produce emergent behavior (Holland 1995).

The remaining years extended these frameworks into specialized domains and established enduring computational paradigms:

1996: Autonomous Agents

Franklin and Graesser distinguished **autonomous agents** as entities situated within an environment that sense and act over time in pursuit of their own agendas (Franklin and Graesser 1997) (ATAL'96; published 1997 in LNCS).

1996: Agent-Based Modeling

Epstein and Axtell's *Growing Artificial Societies* established agent-based modeling (ABM), defining agents as autonomous heterogeneous individuals whose local interactions produce emergent macro patterns (Epstein and Axtell 1996).

1998: Reinforcement Learning

Sutton and Barto's *Reinforcement Learning* defined agents maximizing cumulative reward through environmental interaction, establishing the **RL framework** (Sutton and Barto 1998).

1999: Multi-Agent Systems

A textbook by Jacques Ferber and an edited volume by Gerhard Weiss consolidated **multi-agent systems (MAS)**, defining agents as autonomous problem-solving entities that cooperate, compete, and negotiate to achieve individual and collective goals (Weiss 1999; Ferber 1999).

The decade's diverse frameworks addressed the translation challenges from complementary angles. To the question of representing beliefs computationally, Shoham's agent-oriented programming offered mentalistic implementations (beliefs, commitments, obligations). To the question of observable properties distinguishing agents from programs, Wooldridge and Jennings enumerated autonomy, reactivity, proactivity, and social ability. To functional architecture, Russell and Norvig established the perception-action cycle. To learning and adaptation, Maes demonstrated how agents could improve through observation and feedback. And to emergence, Holland and the ABM community showed how simple local rules generate collective intelligence. These were not competing theories but different levels of description—internal architectures, observable properties, functional patterns, adaptive mechanisms, and emergent phenomena.

Two computational paradigms proved particularly durable. Agent-based modeling demonstrated how to generate social phenomena from individual interactions, bridging agent theory and social science applications. Reinforcement learning formalized agents as reward maximizers learning through environmental feedback, a framework that would scale far beyond the decade's initial demonstrations. By 1999, multi-agent systems had matured into a distinct field with established conferences, textbooks, and architectural patterns. These 1990s frameworks—mental states, observable properties, perception-action cycles, learning mechanisms, and emergence—would shape agent definitions for the next two decades, until large language models introduced qualitatively new capabilities.

3.4 Consolidation and Formalization (Late 1990s–2019)

The early 21st century marked a period of consolidation rather than conceptual revolution. Following the 1990s explosion of agent definitions and implementations, the field matured through educational resources that trained new generations of researchers, reference works that synthesized accumulated knowledge, and successful demonstrations that validated the frameworks at scale. The dot-com boom and bust, the rise of Web 2.0, mobile computing, and eventually the deep learning revolution all occurred during this period, yet they produced remarkably few new definitional frameworks for

agency itself.

Two early definitional contributions framed the era's concerns. Kauffman raised fundamental questions about the relationship between physical and virtual agency, proposing thermodynamic criteria that most software agents clearly failed to meet. Jennings, Sycara, and Wooldridge formalized agent-oriented software engineering, translating the 1990s research insights into practical development methodologies. Beyond these, the period's major developments were consolidative: Wooldridge's *Introduction to MultiAgent Systems* and Epstein's *Generative Social Science* codified best practices, legal updates like the Restatement (Third) of Agency maintained doctrinal continuity, and reference works like the Stanford Encyclopedia entries provided authoritative syntheses.

The period's most dramatic developments came from demonstrations of existing frameworks at unprecedented scale. Deep reinforcement learning combined neural networks with the RL framework established by Sutton and Barto in 1998, achieving superhuman performance first in Atari games and then notably with AlphaGo's victories over world champions. These successes validated that the 1990s conceptual frameworks could scale to complex domains without requiring fundamental redefinition. By 2019, the field had mature definitions, established pedagogical resources, and powerful demonstrations—but also growing awareness that large language models might catalyze the next definitional shift.

2000: Physical Agency

Stuart Kauffman's *Investigations* proposed thermodynamic criteria: natural autonomous agents must reproduce and perform thermodynamic work cycles, highlighting tensions between embodied and virtual agency (Kauffman 2000).

1998 Roadmap: Agent Definition Consensus

Jennings, Sycara, and Wooldridge's 1998 research roadmap synthesized the agent research community's consensus: an agent is "an encapsulated computer system that is situated in some environment, and that is capable of flexible, autonomous action in that environment in order to meet its design objectives" (Jennings et al. 1998; Wooldridge and Jennings 1995). As a historical definition from computer science, they used "system" to mean a computational entity. This definition grounded agent-oriented software engineering (AOSE) work throughout the 2000s.

2006: Restatement (Third) of Agency

The *Restatement (Third) of Agency* updated legal doctrine while retaining core concepts of fiduciary duty and delegated authority (American Law Institute 2006).

2013–2015: Deep Reinforcement Learning

Deep RL combined neural networks with established RL frameworks, achieving superhuman performance in complex domains. DeepMind's DQN mastered Atari games from raw pixels (Mnih et al. 2015), and AlphaGo defeated world champions at Go (Silver et al. 2016), demonstrating that learned agents could exceed human capabilities in domains previously considered intractable.

The period's consolidation reflected the 1990s frameworks' success—they fit the technological capabilities available. Wooldridge's *Introduction to MultiAgent Systems* and Bonabeau's agent-based modeling syntheses trained new generations of practitioners (Wooldridge 2009; Bonabeau 2002). Epstein's *Generative Social Science* articulated ABM's explanatory program: explaining phenomena through simple rule-based actors whose local interactions generate macro patterns (Epstein 2006). Stanford Encyclopedia entries on "Agency" and "Action" provided philosophical syntheses (Schlosser 2019; Wilson and Shpall 2022). By 2020, Russell and Norvig's 4th edition of AIMA had refined agent definitions incorporating ML progress (Russell and Norvig 2020).

Kauffman's thermodynamic criteria highlighted the still-unresolved tension between physical and virtual agency—a question that remains contentious as purely software-based LLM agents proliferate. Jennings and colleagues' software engineering formalization helped practitioners but didn't redefine agency itself. Deep RL validated existing frameworks at unprecedented scale: agents learning through environmental interaction could achieve superhuman performance, confirming Sutton and Barto's 1998 framework rather than replacing it. The stage was set, however, for disruption. Large language models' capacity for flexible reasoning, few-shot learning, and natural language understanding would soon enable agent architectures qualitatively different from the hand-coded systems of previous decades.

3.5 The LLM Era (2020–2025)

Large language models brought capabilities that catalyzed a new wave of agent definitions. GPT-3's release in 2020 demonstrated few-shot learning and flexible reasoning at scale. ChatGPT's public launch in 2022 made conversational AI mainstream. But the definitional shift came not from language modeling itself, but from discovering that LLMs could orchestrate tool use, maintain working memory, and iteratively refine approaches to achieve complex goals. This combination—flexible reasoning plus tool use—enabled agent architectures qualitatively different from previous decades' hand-coded systems.

The key insight emerged in 2022: LLMs could implement the classical perception-action cycle not through programmer-specified rules but through learned parameters. The ReAct pattern (Reasoning

and Acting) demonstrated LLMs interleaving chain-of-thought reasoning with tool use, iteratively planning, acting, and observing until goals were achieved. This simple pattern proved general, spurring implementations across research labs and commercial frameworks. Unlike previous decades' diversity of agent architectures, the LLM era quickly converged on a dominant pattern: an LLM iteratively calling tools, observing results, updating state, and choosing next actions.

This architectural convergence reflects several factors: the flexibility of LLMs as general-purpose reasoners, the natural fit between language models and tool APIs, and the economic incentives driving rapid commercialization. Where 1990s agent definitions emerged from academic research programs over a decade, LLM agent definitions coalesced in months through practitioner experimentation and commercial deployment. The speed of convergence is unprecedented, though whether this pattern will prove as durable as the 1990s frameworks remains to be seen.

2022: The LLM-as-Agent Pattern

Yao et al.'s "ReAct: Synergizing Reasoning and Acting in Language Models" showed that LLMs can interleave chain-of-thought reasoning with tool use, iteratively planning, acting, and observing to achieve goals (Yao et al. 2022). This catalyzed the "**LLM-as-agent**" pattern in contemporary practice. As Simon Willison concisely puts it: "an LLM agent runs tools in a loop to achieve a goal" (Willison 2025).

Framework documentation adopted this pattern explicitly. LangChain defines agents as entities that use LLMs to iteratively call tools until a stop condition is met (LangChain Documentation 2024). OpenAI's Agents SDK similarly describes agents as LLMs configured with instructions and tools, operating in loops where the LLM decides which tools to call based on iterative feedback (OpenAI 2024). This convergence across academic research and commercial frameworks reflects rapid standardization around a single architectural pattern.

2023: Generative Agents

Park et al.'s "Generative Agents" showcased LLM-powered agents simulating believable human behaviors over long horizons via memory, planning, and reflection, popularizing sandboxed LLM agents in society simulations (Park et al. 2023).

2023: The Rise of LLM Agents

Xi et al.'s survey "The Rise and Potential of Large Language Model Based Agents" documented the emerging architectural consensus: entities using LLMs to iteratively call tools, observe results, and adapt until goals are achieved (Xi et al. 2023).

2024–2025: Commercial Frameworks

LangChain defines agents as entities using LLMs to iteratively call tools until a stop condition is met (LangChain Documentation 2024). OpenAI's Agents SDK describes agents as LLMs configured with instructions and tools, operating in loops where the LLM decides which tools to call based on iterative feedback (OpenAI 2024). This convergence represents an emerging architectural consensus across research and commercial frameworks.

The LLM era's definitional landscape differs from previous periods. Rather than multiple competing frameworks addressing different facets of agency, the field quickly converged on a single architectural pattern: LLMs orchestrating tools in iterative loops. ReAct established the template in 2022, and within months implementations proliferated across academia and industry. Park's generative agents demonstrated the pattern's applicability to social simulation, while commercial frameworks embedded it as default infrastructure.

What is genuinely new remains debated. Skeptics note that the perception-action cycle, goal-directed behavior, and tool use all appeared in 1990s definitions—the LLM era implements these through learned parameters rather than programmer-specified logic. Proponents argue that flexible reasoning over natural language representations enables qualitatively different agentic capabilities. A plausible resolution is that LLMs shifted where decision-making knowledge resides (from code to weights) without fundamentally changing the functional architecture of agency. Regardless, the speed of convergence—from research demonstration to industry standard in under three years—marks the LLM era as distinctive in the seven-decade history of agent definitions.

3.6 Historical Patterns

Five Patterns Across Seven Decades

Five patterns emerge:

Broadening entity frames. Early definitions focused on humans or human–AI relationships. By the 1990s, purely computational agents became common. The LLM era introduced hybrid framings: humans provided goals, AI systems executed multi-step tasks.

Increasing autonomy. Definitions migrated from delegated proxies (1958) through perception-action systems (1995) to self-directed tool orchestrators (2025). The locus of decision-making shifted progressively toward the agent, reflecting evolving expectations of autonomy.

From mental states to observable behavior. Philosophical definitions emphasized intention and reasoning. Computer science implemented these as mentalistic constructs. Contemporary definitions focus on observable capabilities—tool use, iterative problem solving, task

completion.

"Tools-in-a-loop" convergence. The AI engineering community has converged toward iterative tool orchestration patterns, documented in practitioner accounts (Willison 2025) and academic surveys (Xi et al. 2023). This reflects technological maturity (LLMs capable of reliable tool use), research foundations (ReAct and related frameworks), and market pressures (customers evaluating empirical capabilities). While not yet a formal standard, this architectural pattern dominates contemporary implementations.

From hand-coded to learned behavior. Early agents used explicit rules and plans specified by programmers. Deep RL demonstrated agents learning behaviors through environmental interaction. LLMs shifted the architectural locus of planning and orchestration from programmer-specified logic to learned parameters, changing where decision-making knowledge resides rather than the fundamental agentic capabilities.

These patterns help explain why contemporary definitions vary: they reflect different points in this evolutionary trajectory and different disciplinary emphases.

The next section examines how different disciplines approached agency, revealing persistent tensions and complementary insights.

4 Disciplinary Perspectives on Agency

While Section 3 traced chronological evolution, this section examines how different disciplines approach agency from distinct theoretical foundations. Each field emphasizes different facets of agency, revealing complementary insights and persistent tensions.

4.1 Philosophy: Intentionality and Reasons

Philosophy's distinctive contribution centers on **intentional descriptions**, responsibility for actions, and predictive stances. Anscombe (1957) established that actions are intentional "under a description"—the same physical movement can be intentional under one description ("signing a contract") but not another ("smudging ink"). This insight matters for AI agents: when we attribute intentions to entities, we're choosing explanatory frameworks, not discovering intrinsic properties.

Bratman (1987)'s planning theory and Dennett (1987)'s **intentional stance** provide complementary perspectives: agents act according to their plans and commitments (Bratman) or can be treated as rational goal-pursuers regardless of internal mechanism (Dennett). Philosophy emphasizes *reasons* for

action over mechanical causation, focusing on responsibility for actions and conceptual foundations rather than implementation.

4.2 Psychology: Self-Regulation and Control

Psychology approaches agency through **self-regulation**, **perceived control**, and contextual variability. Bakan's agency-communion spectrum characterizes fundamental modes of human existence, while Milgram (1974)'s agentic state describes conditions under which people suppress personal judgment and defer to authority. Bandura (1989)'s social cognitive theory emphasizes human agency as exercised through intentionality, forethought, self-reactiveness, and self-reflectiveness.

Psychology's distinctive insight: agency is not merely a property of isolated actions but a stable characteristic of cognitive entities capable of self-regulation. Crucially, agency exists on a spectrum and can be diminished by context (Milgram 1974) or enhanced through **self-efficacy** beliefs. For AI systems, this highlights how humans *perceive* AI agency, affecting trust and delegation decisions.

4.3 Law: Delegation and Fiduciary Duty

Legal doctrine understands agency through **delegated authority**, **fiduciary duties**, and **institutional contexts**. The Restatement of Agency defines the relationship as arising when a principal manifests assent that an agent shall act on the principal's behalf and subject to the principal's control. Key elements include: (1) relational structure requiring a principal-agent pair, (2) delegated authority derived from the principal, (3) fiduciary obligations of loyalty and care, and (4) liability attribution within scope of authority.

Law's distinctive approach focuses on external authority structures and legally enforceable duties rather than internal intentions or motivations. Legal frameworks embed agency in formal mechanisms—contracts, corporate charters, powers of attorney—that create, bound, and terminate authority relationships. For AI systems, this raises immediate questions: Can an AI be an agent in the legal sense? Who serves as principal? As illustrated in Section 1, misunderstanding these relationships can have serious consequences when entities lack mechanisms for fulfilling fiduciary duties.

4.4 Economics: Incentives and Information Asymmetry

Economic analysis frames agency through **utility alignment**, **information asymmetry**, and **organizational design**. Jensen and Meckling (1976)'s principal-agent model identifies the core problem: principals and agents have divergent preferences and asymmetric information. Agents

may pursue self-interest at principals' expense (**moral hazard**), or principals may struggle to assess agent quality ex ante (**adverse selection**).

Economics' distinctive approach emphasizes cost-benefit analysis and divergence of preferences rather than rights or duties. The focus is pragmatic: given that perfect alignment is impossible, how do we minimize **agency costs** through compensation structures, performance metrics, and oversight mechanisms? For AI systems, economic frameworks highlight alignment challenges: How do we design "reward functions" that elicit desired behavior? What monitoring mechanisms detect when AI agents pursue proxy metrics rather than true objectives?

4.5 Cognitive Science: Mental Architecture and Plans

Cognitive science approaches agency through **modular mental societies**, **plan-based control**, and **representational structures**. Minsky (1986)'s Society of Mind framed intelligence as emerging from interactions among simple processes without central control. Bratman (1987)'s planning theory emphasized that intentions are elements of partial plans structuring practical reasoning over time. The **BDI (Belief-Desire-Intention)** architecture implements this: agents maintain beliefs about the world, desires representing goals, and intentions as committed plans.

Cognitive science's distinctive insight treats agency as a lens on *mental architecture*, not just overt action, focusing on information processing, symbolic manipulation, and hierarchical plan structures. The framework readily generalizes from human to machine minds via abstraction. For AI systems, cognitive science provides architectural blueprints—the BDI framework has directly influenced agent programming languages, and concepts like partial plans and means-end reasoning transfer naturally to computational implementations.

4.6 Complex Systems: Emergence and Local Rules

Complex systems science characterizes agency through **local rules generating global effects**, **adaptation**, and **environmental coupling**. Holland (1995)'s complex adaptive systems, Epstein and Axtell (1996)'s agent-based models, and Kauffman (2000)'s autonomous agents emphasize that interesting behavior emerges from interactions among rule-following entities. Individual agents need not be sophisticated—simple local rules, diversity, and environmental feedback suffice to produce **emergent macro patterns**.

Complex systems' distinctive insight treats agents as embedded components of adaptive networks rather than isolated decision-makers. Behaviors co-evolve through selection pressures, learning, and resource flows. For AI systems, this perspective highlights how collections of simple agents exhibit sophisticated collective behavior (multi-agent systems, swarm robotics), but also warns against over-attributing agency to individual components when behavior primarily reflects system-level

dynamics.

4.7 Computer Science: Protocols and Mentalistic Abstractions

Computer science, particularly agent-oriented software engineering, frames agency through **mentalistic programming abstractions**, **interaction protocols**, and **organizational modeling**. Shoham (1993)'s agent-oriented programming introduced beliefs, commitments, and choices as first-class programming constructs. Unlike object-oriented programming (encapsulation and message-passing), agent-oriented programming treats components as having mental states and engaging in speech acts. FIPA agent communication languages formalized performatives (inform, request, propose) modeled on human **speech acts**.

Computer science's distinctive approach uses mentalistic vocabulary as an *engineering tool* rather than philosophical claim. Attributing beliefs to software entities improves code clarity and reasoning about distributed systems, whether or not agents "really" have beliefs. For AI systems, computer science provides the middleware—communication standards, coordination mechanisms, organizational structures—that enable multi-agent systems to function. Agents negotiate rather than merely receiving method calls.

4.8 Cross-Disciplinary Synthesis

Table 5 summarizes key differences across disciplines.

Discipline	Autonomy Level	Entity Frame	Key Mechanism	Primary Concern
Philosophy	Low–moderate	Human, extensible	Intention & reasons	Rational causality, responsibility
Psychology	Varies by context	Human-centered	Self-regulation & control	Lived experience, efficacy
Law	Low–moderate	Hybrid (principal/agent)	Authority & fiduciary duty	Liability, professional responsibility
Economics	Moderate	Institutional/hybrid	Incentive alignment	Agency costs, information asymmetry
Cognitive Science	Moderate–high	Human, generalizable	Plans & representations	Mental architecture
Complex Systems	Moderate–high	Machine/hybrid	Local rules & emergence	Collective behavior, adaptation
Computer Science	Moderate–high	Machine/hybrid	Protocols & coordination	System organization, communication
AI	High	Machine (+ hybrid)	Perception/action loops, tools	Task completion, capabilities

Table 5: Cross-disciplinary comparison of agency perspectives. Disciplines vary systematically in assumed autonomy level, entity framing, core mechanisms, and primary concerns. *Accessibility:* Landscape table with 5 columns (Discipline, Autonomy Level, Entity Frame, Key Mechanism, Primary Concern) and 9 rows (8 disciplines plus AI). Rows ordered: Philosophy (low-moderate, human, intention/reasons, responsibility), Psychology (varies, human, self-regulation, lived experience), Law (low-moderate, hybrid, authority/fiduciary duty, liability), Economics (moderate, institutional, incentive alignment, agency costs), Cognitive Science (moderate-high, human/generalizable, plans/representations, mental architecture), Complex Systems (moderate-high, machine/hybrid, local rules/emergence, collective behavior), Computer Science (moderate-high, machine/hybrid, protocols/coordination, system organization), AI (high, machine + hybrid, perception-action loops/tools, task completion). Table reveals systematic patterns: human vs. machine focus, normative vs. descriptive approaches.

Several patterns emerge:

- **Autonomy gradient**: Disciplines focusing on human-AI relationships (law, economics) assume lower autonomy, while those focused on machine capabilities (AI, complex systems) assume higher autonomy
- **Entity frames**: Philosophy, psychology, and cognitive science start with humans; law and economics examine hybrid relationships; complex systems, computer science, and AI embrace machine entities
- **Mechanism diversity**: Each discipline emphasizes different causal mechanisms—intentions, self-regulation, incentives, plans, rules, protocols, actions
- **Normative vs descriptive**: Philosophy and law are more normative (concerned with responsibility and duty), while AI and complex systems are more descriptive (concerned with observable behavior)

Integration Framework

For understanding agentic AI in legal contexts, we require synthesis across disciplines:

- **Philosophy** provides conceptual foundations for attributing agency
- **Psychology** illuminates how humans perceive and interact with AI agents
- **Law** establishes liability frameworks and professional responsibilities
- **Economics** offers tools for analyzing alignment problems
- **Cognitive science** suggests architectural patterns
- **Complex systems** warns about emergent behavior in multi-agent contexts
- **Computer science** provides implementation protocols
- **AI** defines practical capabilities and evaluation criteria

The next section extracts analytical dimensions that cut across these disciplinary perspectives.

5 Analytical Dimensions of Agency

While different disciplines approach agency from distinct theoretical foundations (Section 4), certain analytical dimensions cut across disciplinary boundaries. This section identifies four key dimensions—**autonomy**, **entity frames**, **goal dynamics**, and **persistence**—that structure variation in how agency is understood.

5.1 The Autonomy Spectrum

Perhaps the most fundamental dimension along which agent definitions vary is **autonomy spectrum**: the degree to which an agent can set its own agenda, select tactics, and own accountability for outcomes.

Three plain-language questions help position any definition on the autonomy spectrum:

Who sets the goal?

Does the agent accept goals from an external principal, or does it generate its own objectives? Delegated proxies follow instructions; self-directed entities set their own agendas.

Who decides the next step?

Does the agent execute predefined plans step-by-step, or does it choose actions autonomously? Delegated proxies follow explicit instructions. Contract-bound delegates choose methods (how) but not objectives (what). Perception-action planners select both tactics and timing independently.

Who carries responsibility if things go wrong?

Does accountability remain with the principal, or does the agent own outcomes? Low-autonomy agents are instruments; high-autonomy agents bear responsibility for their choices.

Definitions range from delegated proxies at one extreme to self-directed entities at the other.

Delegated Proxies (Low Autonomy). The Restatement (Second) of Agency (1958) exemplifies this end: agents execute their principal's instructions within bounded authority. Goals, evaluation criteria, and ultimate responsibility remain with the principal. Milgram (1974)'s "agentic state" captures the psychological correlate—individuals who see themselves as mere instruments of authority, disclaiming personal responsibility.

Contract-Bound Delegates (Low-Moderate Autonomy). Jensen and Meckling (1976)'s principal-agent model grants agents tactical discretion within incentive contracts but evaluates them solely on principals' payoffs. Agents choose means but not ends, operating under monitoring and compensation schemes designed to align behavior.

Self-Regulating Actors (Moderate Autonomy). Bandura (1989)'s human agency emphasizes that individuals set sub-goals, monitor progress, and self-adjust within social structures. Maes (1994)'s software agents learn user preferences and initiate actions based on high-level intent rather than step-by-step commands.

Perception-Action Planners (Moderate-High Autonomy). Russell and Norvig (1995)'s AI agents sense environments and select actions via performance-driven policies without waiting for commands. Franklin and Graesser (1997)'s autonomous agents pursue their own agendas over time, acting to influence future perceptions.

Learning Loop Owners (High Autonomy). Sutton and Barto (1998)'s reinforcement learning agents experiment with actions and update policies from reward signals without human step-by-step guidance. The agent owns the exploration-exploitation tradeoff.

Tool Orchestrators (Very High Autonomy). Modern tool-use APIs (2025)—whether implemented through AI/ML or traditional logic—choose when and how to invoke tools, integrating outputs into their reasoning loops. Contemporary LLM frameworks provide one implementation approach for this architectural pattern, "independently accomplish[ing] tasks on behalf of users," planning, sequencing, and completing multi-step workflows before reporting back.

Table 6 summarizes this progression.

Position	Source	Description
Delegated proxy	American Law Institute (2006)	Follows instructions; principal retains liability
Obedience	Milgram (1974)	Suppresses personal goals under authority
Contract-bound	Jensen and Meckling (1976)	Chooses tactics within contracts
Self-regulating	Bandura (1989)	Sets goals, monitors, adjusts
Adaptive assistant	Maes (1994)	Learns preferences, initiates actions
Perception-action	Russell and Norvig (1995)	Senses environment, executes policy
Agenda-setting	Franklin and Graesser (1997)	Maintains independent agenda
Learning loop	Sutton and Barto (1998)	Experiments, updates from rewards
Tool orchestrator	Yao et al. (2022)	Chooses tools to invoke
Independent finisher	Xi et al. (2023)	Plans and completes tasks

Table 6: Autonomy spectrum: from delegated proxies to self-directed entities. *Accessibility:* Progressive classification table with 3 columns (Position, Source, Description) and 10 rows ordered by increasing autonomy: from "Delegated proxy" (follows instructions, principal retains liability) through intermediate positions (obedience, contract-bound, self-regulating, adaptive assistant, perception-action, agenda-setting, learning loop, tool orchestrator) to "Independent finisher" (plans and completes tasks). Each row cites specific researchers showing chronological evolution.

This dimension correlates with but differs from technological capability. Rule-based expert systems can exhibit moderate autonomy (goal-directed loops), while capable neural networks, including LLMs, operating in single-shot mode exhibit none. Autonomy is architectural, not merely a function of model sophistication.

Autonomy and Governance: The autonomy level directly determines governance requirements.

Low-autonomy agents (delegated proxies, contract-bound delegates) operate under tight principal control with lighter oversight burdens—the principal validates each step or constrains choices through explicit contracts. High-autonomy agents (tool orchestrators, independent finishers) make consequential decisions with minimal intervention, demanding mandatory precommit limits (resource budgets, scope boundaries), explicit escalation triggers (confidence thresholds, high-stakes decisions), and robust audit trails. *Governance principle:* Oversight rigor must scale with autonomy level. An agent that independently orchestrates tools and completes multi-step tasks requires explicit termination mechanisms, human-in-the-loop checkpoints for irreversible actions, and clear accountability mapping—safeguards unnecessary for agents executing predefined scripts under direct supervision.

5.2 Entity Frames

Agent definitions often talk past each other because they quietly assume different kinds of **entities**. We identify three clusters:

Human-Centered. These definitions privilege individual humans and their cognition or morality. Anscombe (1957)'s intentional action explores agency as practical knowledge held by persons. Milgram (1974)'s agentic state examines how people respond to authority. Bandura (1989)'s human agency emphasizes internal cognitive machinery—intentionality, forethought, self-reactiveness, self-reflectiveness. The hallmark questions: *What intentions animate the person? How is responsibility assigned?*

Institutional/Hybrid. These definitions situate agency in socio-technical collectives—humans embedded in contracts, firms, or organizational structures. The Restatement of Agency (1958) defines the principal-agent pair, not isolated individuals. Jensen and Meckling (1976)'s economic model examines "manager-with-contract" as the effective actor. Epstein and Axtell (1996)'s agent-based models simulate rule-following individuals shaped by local interactions. The hallmark questions: *What structure channels decisions? How are incentives and authority shared?*

Machine-Centered. These definitions assign agency to computational entities managing perception-/action loops. Russell and Norvig (1995)'s AI agents can be any entity perceiving through sensors and acting via actuators. Franklin and Graesser (1997)'s autonomous agents are persistent computational entities pursuing agendas. Contemporary LLM frameworks (2025) are explicitly software coordinating tools. The hallmark questions: *How does the entity sense, plan, and act? What loop keeps it going?*

Table 7 summarizes representative examples.

Importantly, these frames are not mutually exclusive. Contemporary "agentic AI" often involves hybrid entities: AI systems acting on behalf of human principals, with goals supplied by humans but

Frame	Source	Description
Human	Anscombe (1957)	Human minds, intentions
	Milgram (1974)	
	Bandura (1989)	
Institutional	American Law Institute (2006)	Organizations, authority structures
	Jensen and Meckling (1976)	
	Epstein and Axtell (1996)	
Machine	Russell and Norvig (1995)	Computational entities, loops
	Franklin and Graesser (1997)	
	Xi et al. (2023)	

Table 7: Entity frames: human minds, institutional relationships, or computational entities. *Accessibility:* Categorization table with 3 columns (Frame, Source, Description) and 3 rows representing three fundamental approaches. Row 1: Human frame (philosophy/psychology) emphasizes human minds and intentions. Row 2: Institutional frame (law/economics) emphasizes organizations and authority structures. Row 3: Machine frame (computer science/AI) emphasizes computational entities and loops. Multiple scholars cited per frame showing discipline-spanning taxonomy.

tactics chosen by machines. The legal and ethical challenges arise precisely at these boundaries.

5.3 Goal Dynamics

Definitions also vary in how they understand the **relationship between agents and goals**. We identify three stances:

Goal Acceptance. The agent receives a mandate and optimizes for it without debate. The Restatement of Agency (1958) requires agents to follow principal objectives; divergence invites breach. Sutton and Barto (1998)'s RL agent maximizes a fixed reward function supplied externally. Contemporary LLM frameworks (2025) "independently accomplish tasks on behalf of users," but user goals remain the north star. Success equals compliance or reward maximization.

Goal Adaptation. The agent refines, reprioritizes, or balances goals within constraints. Maes (1994)'s software agents learn user preferences and decide when to intervene, reprioritizing information flows. Russell and Norvig (1995)'s agents balance performance measures, trading off actions based on context. Modern tool-use APIs (2025) interpret instructions, break them down, and choose which tools serve user intent. The agent has discretion in interpretation and tactics.

Goal Negotiation. The agent co-determines objectives with others, often through communication. Weiss (1999)'s multi-agent systems coordinate, cooperate, and negotiate when goals conflict. Jennings et al. (1998)'s AOSE roadmap treats agents as social entities that negotiate commitments and allocate tasks. Multi-agent frameworks (2025) enable multi-agent applications where agents message each other to propose plans, critique, and converge collaboratively.

Table 8 summarizes this dimension.

Stance	Source	Description
Acceptance	American Law Institute (2006) Sutton and Barto (1998) Xi et al. (2023)	Follows supplied objectives
Adaptation	Maes (1994) Russell and Norvig (1995) Yao et al. (2022)	Refines goals, chooses tactics
Negotiation	Weiss (1999) Jennings et al. (1998)	Co-determines objectives

Table 8: Goal dynamics: from acceptance through adaptation to negotiation. *Accessibility:* Progressive stance table with 3 columns (Stance, Source, Description) and 3 rows showing increasing autonomy in goal-setting. Row 1: Acceptance stance (follows supplied objectives without modification). Row 2: Adaptation stance (refines goals and chooses tactics within constraints). Row 3: Negotiation stance (co-determines objectives through communication). Progression has direct implications for legal accountability.

This dimension has direct implications for legal accountability. If an AI agent merely accepts goals, responsibility flows clearly to whoever set them. If it adapts goals through interpretation, questions arise about faithful execution. If it negotiates goals with other agents or humans, traditional principal-agent frameworks may not apply cleanly.

5.4 Persistence and Embodiment

Two additional dimensions merit brief mention:

Temporal Persistence. Franklin and Graesser (1997) distinguish programs from agents partly on **persistence**—agents maintain state and pursue objectives over extended periods. This contrasts with reactive systems that respond to inputs without memory. Modern LLM agents incorporate conversation history and tool-use results, creating persistence within sessions. As discussed in Section 1, single-shot LLM usage lacks the persistent goal pursuit that characterizes agentic systems.

Embodiment. Kauffman (2000)'s autonomous agents require physical **embodiment** capable of performing thermodynamic work cycles. This biological grounding contrasts sharply with purely virtual agents. Robotics and embodied AI represent one implementation; cloud-based LLM agents represent another. Whether virtual agents merit the term "agent" in the fullest sense remains philosophically contentious, though pragmatically settled in favor of inclusion.

5.5 Implications for Agentic AI

These four dimensions—autonomy, entity frames, goal dynamics, persistence—provide organizing principles for evaluating contemporary systems.

When someone claims an entity is "agentic," four questions create a shared, falsifiable basis for evaluation.

Question 1: Autonomy

How much **discretion** does the entity have in selecting **goals** and **tactics**, and where does **accountability** reside when outcomes go wrong? Describe the locus of **control** and any **constraints** (contracts, prompts, policies).

Question 2: Entity

What kind of **entity** are we evaluating: a pure **machine** entity, a **human–AI** team, or an **institution**al arrangement? Clarifying the entity prevents conflating a tool with the broader organization that wields it.

Question 3: Goals

Does the entity merely accept **goals**, **adapt** them through interpretation, or **negotiate** them with other agents or humans? State how **goals** are set, refined, and **verified** in practice.

Question 4: Persistence

Does the entity maintain **state** and pursue **objectives** across multiple steps or sessions, or does it operate in **single-shot** mode? Specify how **memory**, **logs**, or **context windows** support continuity.

These questions cut across the marketing hype and disciplinary jargon, providing a shared vocabulary for more precise discourse. The next section synthesizes these insights into a working definition grounded in theoretical foundations.

6 Analytical Framework

Having traced the historical evolution (Section 3), examined disciplinary perspectives (Section 4), and identified analytical dimensions (Section 5), we now synthesize these insights into a coherent analytical framework. This framework provides both the theoretical foundation for understanding agency and practical tools for evaluating real-world systems.

6.1 Cross-Cutting Patterns

Our analysis across disciplines reveals four fundamental patterns that consistently appear when scholars and practitioners identify agency. These patterns transcend individual fields, providing a shared foundation for recognizing agentic behavior regardless of whether we're examining human decision-makers, organizational structures, or computational entities.

Pattern 1: Goal-Directedness

Agents pursue objectives or performance criteria. Whether we call them intentions in philosophy, rewards in reinforcement learning, mandates in law, or incentives in economics, the common thread remains: agents exhibit behavior oriented toward achieving outcomes rather than merely reacting to stimuli. This distinguishes purposeful action from reflexive response.

Pattern 2: Perception–Action Coupling

Agents operate through iterative cycles of sensing and acting. This pattern appears across theoretical frameworks—from Russell and Norvig (2020)'s sensor-actuator loops to Bratman (1987)'s practical reasoning and Bandura (1989)'s reciprocal determinism. Entities that transform inputs once without iteration, such as compilers or single-shot classifiers, lack this essential coupling that characterizes agentic systems.

Pattern 3: Selective Autonomy

Agents require some degree of discretion in their choices, as we explored in Section 5. Entities that execute fixed scripts without any decision-making capacity do not qualify as agents. The degree of autonomy varies significantly across domains—from minimal discretion in rule-based entities to substantial independence in modern AI agents—with higher levels of discretion typically raising more complex governance and accountability questions.

Pattern 4: Termination

Purposeful behavior requires boundaries. Agents must have mechanisms for recognizing when to stop, whether through explicit controls (resource budgets, time limits, escalation triggers) or implicit goal satisfaction (achieving target state, exhausting search space). Entities that run indefinitely without clear stopping conditions or decision points fall outside our definition of agentic systems, as they lack the goal-oriented closure that characterizes purposeful action.

6.2 Formal Core

While the patterns above provide intuitive understanding, precision demands formal specification. We now translate our insights into a mathematical framework that enables rigorous analysis and falsifiable claims about agency.

To formalize our framework, we denote the six essential properties as follows:

- G = goal or utility function that directs behavior
- P = perception capability for sensing environment
- A = action capability for affecting environment
- I = iteration across multiple steps with state preservation
- A_{adapt} = adaptation through modifying strategy based on accumulated observations and feedback
- T = termination through explicit or implicit conditions

Consistent with the operational definition in Section 1, the six formal properties are Goal, Perception, Action, Iteration, Adaptation, and Termination (GPA + IAT). Autonomy or discretion remains an analytical dimension from Section 5 that describes *how* these properties are exercised rather than a separate core property.

We use the notation $has(x, \cdot)$ to indicate that entity x possesses a given property.

Formal Definitions

We define three nested categories of entities based on their properties:

$$Agent(x) \iff has(x, G) \land has(x, P) \land has(x, A)$$

(Minimal agency: goals, perception, action)

$$AgenticSystem(x) \iff Agent(x) \land has(x, I) \land has(x, A_{\text{adapt}}) \land has(x, T)$$

(Full operational system: all six properties)

$$AIAgent(x) \iff AgenticSystem(x) \land implementedWithAI(x)$$

(AI-powered implementation of agentic system)

These formal definitions yield several important theoretical results that clarify the relationships between different types of entities:

Key Theoretical Results

Hierarchical Subsumption. The categories form a nested hierarchy:

$$AIAgent(x) \Rightarrow AgenticSystem(x) \Rightarrow Agent(x)$$

Every AI agent is necessarily an agentic system, and every agentic system is necessarily an agent, but the converses do not hold.

Non-Equivalence. The categories are strictly distinct:

$$\exists x : Agent(x) \land \neg AgenticSystem(x)$$

For example, a single-shot classifier might have goals, perception, and action capabilities, but lacks the iteration, discretion, and termination properties required for an agentic system.

Property Attribution. Individual features can be described as "agentic" (such as "agentic perception" or "agentic planning") without implying the system as a whole qualifies as an $AgenticSystem$. Full agentic system status requires all six properties.

The termination property deserves special attention, as it can manifest in two distinct forms:

Termination Modes

An entity satisfies the termination requirement through either explicit or implicit mechanisms:

$$has(x, T) \iff \text{ExplicitTerm}(x) \lor \text{ImplicitTerm}(x)$$

Explicit termination involves programmed monitoring of conditions such as:

- Resource budgets (computation time, API calls, token limits)
- Timeout constraints (wall clock time, iteration counts)
- Escalation triggers (confidence thresholds, error conditions)

Implicit termination occurs naturally through:

- Goal satisfaction (target state achieved)
- Quiescent states (no further actions available)
- Bounded episodes (fixed-duration tasks)

Termination Audit Checklist for Deployments

Before deploying an agentic system in professional contexts, verify it implements at least one explicit termination mechanism:

☐ **Time budget:** Maximum wall-clock duration (seconds/minutes)
☐ **Iteration limit:** Maximum perception-action cycles
☐ **Token/request budget:** Maximum API calls, LLM tokens, or database queries
☐ **Cost ceiling:** Maximum monetary expenditure per task
☐ **Escalation trigger:** Automatic handoff when confidence drops below threshold
☐ **Error accumulation:** Stop after N consecutive failures or exceptions
☐ **Human intervention point:** Required approval before high-stakes actions

For high-stakes deployments (legal, medical, financial), implement multiple explicit termination mechanisms to prevent runaway processes and ensure predictable resource consumption.

6.3 Foundations in Context

Our formal framework draws from multiple theoretical traditions, each contributing essential insights about how agents operate. Understanding these foundations helps practitioners from different backgrounds connect our framework to established concepts in their fields.

Plan-Based Control. The tradition of **plan-based control** emphasizes how agents decompose high-level goals into actionable steps. This approach, central to cognitive science and AI planning systems, treats agency as means-end reasoning—connecting what we want to achieve with how to achieve it. In legal practice, we see this pattern in structured workflows like due diligence, where complex objectives decompose into systematic review tasks. Modern AI agents employ similar hierarchical planning when breaking down user requests into sequences of tool calls and sub-tasks.

Reactive Control. Not all intelligent behavior requires elaborate planning. The **reactive control** paradigm demonstrates that simple perception-action rules can produce sophisticated behavior when properly designed. This insight, pioneered in robotics by researchers like Rodney Brooks, shows that

agents can be effective without maintaining complex world models. In fast-changing environments or when sensing is reliable, responsive behavior often outperforms deliberative planning. Contemporary LLM agents balance both approaches—maintaining high-level plans while reacting dynamically to unexpected outputs or errors.

BDI Models. The **Belief-Desire-Intention** (BDI) framework provides a structured account of agent mental states. Agents maintain beliefs about their environment, desires about preferred outcomes, and intentions representing committed plans. This model, influential in agent-oriented programming, explains how agents form and revise commitments as new information arrives. Legal agents exemplify this pattern: maintaining beliefs about applicable law, desires to serve client interests, and intentions manifested as filed documents or negotiation positions.

Reinforcement Learning. The **reinforcement learning** paradigm formalizes how agents learn from experience. Through cycles of action, observation, and reward, agents discover which behaviors achieve their goals. This framework clarifies why iterative interaction outperforms single-shot processing—each cycle provides information that improves future decisions. Modern AI agents increasingly incorporate reinforcement learning, whether through fine-tuning on human feedback or real-time adaptation during task execution.

Intentional Stance. Following philosopher Daniel Dennett's influential work (Dennett 1987), the **intentional stance** treats agency as a pragmatic attribution rather than metaphysical fact. We describe systems as having beliefs, desires, and intentions when such descriptions improve our ability to predict and control their behavior. This perspective liberates us from endless debates about whether AI systems "really" have goals or merely simulate goal-directed behavior. What matters is whether treating them as agents yields practical benefits.

Why These Foundations Matter

These foundations directly inform how we build and evaluate agentic systems:

- **Planning** supports goal decomposition and task orchestration
- **Reactive control** validates simple, robust responses to changing environments
- **BDI models** supply architectures for managing commitments and updating beliefs
- **Reinforcement learning** explains improvement through experience and feedback
- **Intentional stance** guides when to treat systems as agents for prediction and control

Together, these traditions ground the six-property framework in established theory while keeping it practically usable across disciplines from law to computer science.

6.4 Boundary Cases and Clarifications

Testing our framework against boundary cases reveals both its strengths and the subtleties of applying formal definitions to real-world systems. These edge cases illuminate practical fault lines—where practitioners disagree about agency and where our framework proves most valuable.

Thermostats and the Agency Spectrum. A basic mechanical thermostat demonstrates the spectrum from minimal agency to full agentic systems. It has five of six properties: goal (maintain temperature), perception (sensor readings), action (heating/cooling), iteration (continuous monitoring), and termination (implicit when target reached). However, it lacks adaptation, applying fixed on/off rules without modifying strategy based on outcomes. This distinguishes reactive control from adaptive learning. Smart thermostats, by contrast, qualify as full agentic systems—they learn occupancy patterns, adjust schedules based on observed behavior, and implement explicit termination logic. This progression illustrates why minimal agency (Level 1) sets a deliberately low bar, while agentic systems (all six properties) require operational sophistication suitable for professional deployment.

Multi-Agent Collectives and Emergent Behavior. When multiple agents interact, the collective can exhibit properties no single agent possesses. A trading floor, legal team, or swarm of autonomous vehicles may demonstrate emergent behaviors that challenge our individual-focused definitions. Legal systems have long grappled with this through doctrines distinguishing corporate from individual liability. Our framework accommodates both perspectives: we can analyze individual agents within the system or treat the collective itself as an agent with emergent goals and capabilities. The choice depends on analytical purpose—attribution questions favor individual analysis, while coordination and emergent behavior favor collective treatment.

Virtual versus Embodied Agents. Software agents need not have physical bodies to qualify as agents. A contract analysis system operating entirely in cloud infrastructure satisfies all six properties without any physical embodiment. While robotics researchers sometimes insist on embodiment as essential to agency, our framework follows the broader AI tradition of recognizing virtual agents as fully legitimate instances. The perception-action loop operates equally well whether the environment is physical (sensor readings, motor commands) or virtual (API queries, database updates). Professional legal and financial applications predominantly involve virtual agents, making this inclusive stance practically necessary.

Single-Shot LLM Interactions. A single ChatGPT response, however sophisticated, typically lacks the iteration property required for agentic systems. It processes input once and generates output without maintaining state across multiple perception-action cycles. The system doesn't observe results, adjust strategy, and continue pursuing goals—it completes and stops. For the purposes of this framework, we treat such single-shot LLM calls as non-iterative, even if they internally perform multiple reasoning steps, because they do not observe and react to *external* feedback within a perception-action loop.

Tool Use versus Tool-Using Agents. Simply using tools doesn't make an entity agentic. A script that calls an API once isn't an agent—it lacks the iteration and adaptation that characterize agentic systems. What matters is purposeful, iterative tool use directed toward goals. **Simply calling an API once does *not* make a system an agent; the difference lies in goal-directed, iterative tool use with adaptation.** An LLM that repeatedly queries databases, processes results, and decides which tool to invoke next based on accumulated information exhibits agency. The distinction lies in the goal-directed iteration, not the mere presence of tool use. Advanced agentic systems extend this further by creating new tools—writing Python scripts to expand their capabilities or composing prompts that define specialized sub-agents (Anthropic 2025). This recursive capability, where agents manage the design and implementation of their own tools, represents a meta-level of agency beyond mere tool invocation.

Continuous Control and Implicit Termination. Entities engaged in continuous control—like systems maintaining server uptime or monitoring market conditions—might seem to violate our termination requirement. However, these entities typically operate in bounded episodes with implicit termination conditions: achieving steady state, exhausting available actions, or reaching time limits. Even "always-on" entities have mechanisms for recognizing when to stop particular behaviors, satisfying our termination criterion. A market monitoring system may run continuously, but individual monitoring tasks terminate when specific conditions trigger alerts or when trading sessions close. The distinction between the persistent entity and its bounded tasks resolves the apparent contradiction.

6.5 Professional Deployment

Understanding what makes an entity agentic is only the first step. In regulated domains like law and finance, deploying AI agents requires robust governance frameworks that ensure compliance, maintain professional standards, and protect sensitive information. The stakes are particularly high: errors can trigger regulatory sanctions, breach fiduciary duties, or compromise client confidentiality.

Professional deployment demands that we move beyond theoretical properties to operational safeguards. Each of the six properties we've identified creates both opportunities and risks that must be managed through careful entity design and governance controls. The following controls transform theoretical capabilities into trustworthy tools:

1. Attribution and Provenance

Every factual claim must be traceable to its source. The system should record not just what it concluded but where that information originated—whether from case law, regulatory filings, market data, or expert analysis. This enables independent verification and helps practitioners assess the reliability of agent-generated work product. Modern AI agents should maintain citation chains that would satisfy the standards of legal briefs or investment memoranda.

2. Auditable Reasoning

Professional decisions require transparent rationales. The system must log major reasoning steps, tool invocations, and decision points in forms that preserve privilege while enabling review. This isn't about exposing every neural network activation—it's about capturing the business logic an expert would expect to see. Think litigation hold decisions, trade execution rationales, or compliance determination paths.

3. Bounded Operation

Unbounded iteration violates both practical and regulatory constraints. Systems need explicit limits: computational budgets, time constraints, maximum iteration counts. These bounds prevent runaway processes while ensuring predictable resource consumption. In financial contexts, this might mean limiting the number of market data queries; in legal contexts, capping document review cycles.

4. Escalation Pathways

Agents must know their limits and when to seek help. Clear escalation triggers—low confidence scores, high-stakes decisions, detected conflicts—should route matters to human review. The escalation logic should be transparent and adjustable based on risk tolerance and regulatory requirements. A contract review agent, for instance, might escalate unusual liability provisions or non-standard jurisdiction clauses.

5. Confidentiality Controls

Information barriers are non-negotiable in professional services. The system must enforce ethical walls between matters, implement role-based access controls, and apply appropriate redaction rules. This goes beyond basic security to encompass the nuanced confidentiality obligations of legal privilege, insider trading restrictions, and client confidentiality duties.

6. Accountability Mapping

When things go wrong—and they will—clear lines of responsibility enable rapid response and remediation. Who configured the agent? Who reviewed its outputs? Who authorized its deployment? These questions must have clear answers before the system processes its first client matter. Accountability isn't about blame; it's about maintaining professional standards when humans and AI systems collaborate.

6.6 Common Questions About Agency

Our framework inevitably raises questions about edge cases and apparent contradictions. Here we address the most common concerns that arise when applying these definitions to real-world systems.

Is iteration just repeated action? Iteration means more than simply repeating the same action multiple times. True iteration involves maintaining state across perception-action cycles, enabling the entity to learn from previous attempts and adjust its approach. An entity that simply retries the same failing action indefinitely isn't iterating in our sense—it's stuck in a loop. Iteration requires the ability to incorporate feedback, correct errors, and make progress toward goals through successive refinements. A system that queries the same API endpoint repeatedly with identical parameters isn't iterating; one that refines its query based on previous results is. This is why single-shot entities, no matter how sophisticated, don't qualify as agentic systems under our framework.

Do continuous tasks violate the termination requirement? Entities performing continuous tasks like monitoring or maintenance might seem to run forever, apparently violating our termination requirement. However, these entities still satisfy the termination property through implicit or explicit mechanisms. Implicit termination occurs when the entity reaches a steady state or exhausts available actions. Explicit termination happens through resource budgets, time limits, or escalation triggers. Even an entity that monitors markets continuously operates in bounded episodes—trading sessions, reporting periods, or maintenance windows. The key is that the entity has clear conditions under which it will cease its current behavior pattern, even if it later resumes. The termination requirement ensures predictable resource consumption and prevents runaway processes, not that entities can never restart.

If thermostats count as agents, have we trivialized the concept? Recognizing basic thermostats as minimal agents doesn't trivialize agency—it establishes a meaningful floor. Many entities fail even this basic test: databases lack goals, compilers lack perception-action loops, and lookup tables lack any form of action. More importantly, while basic thermostats have five operational properties (lacking adaptation), they demonstrate precisely why we distinguish minimal agency (Level 1: three properties) from agentic systems (six properties). The distinction between a basic thermostat and an AI legal research assistant isn't whether they're agents, but the degree of autonomy, adaptation capability, sophistication, and responsibility they possess. Our framework acknowledges the thermostat's basic agency while reserving "agentic system" status for entities with all six properties (like smart thermostats that learn patterns), and "AI agent" designation for those using AI/ML for implementation. The gradient from simple to sophisticated agents reflects operational reality.

Is "AI agent" just marketing hype? In our framework, "AI agent" has a precise, falsifiable meaning: an agentic system that uses AI or machine learning (particularly large language models) for planning and orchestration. This excludes single-shot chat completions, no matter how impressive, because they lack iteration. It excludes traditional rule-based systems, even if they meet all six properties, because they don't use AI for implementation. It also excludes simple ML classifiers that lack goal-

directedness and autonomous action. When vendors claim their product is an "AI agent," we can test this claim against our six properties and the AI implementation requirement. This transforms marketing language into testable assertions about system architecture and capabilities. The framework provides accountability—vendors must demonstrate iteration, adaptation, and termination mechanisms, not just AI-powered text generation.

7 Conclusion

This chapter synthesized seven decades of scholarship from eight disciplines—philosophy, psychology, law, economics, cognitive science, complex systems, computer science, and AI—into practical frameworks for understanding and evaluating agency. You now have conceptual tools for cutting through vendor hype, evaluating AI products, and understanding what distinguishes genuinely agentic systems from sophisticated chatbots.

7.1 The Framework You Now Have

The three-level hierarchy provides your foundational mental model. An **agent** is an entity requiring three minimal properties: goals, perception, and action. This baseline is deliberately broad—thermostats, organizations, and humans all qualify. An **agentic system** is an agent with three more properties essential for reliable professional deployment: iteration across multiple steps, adaptation through learning or feedback, and termination via explicit or implicit stopping conditions. **Agentic AI** implements all six properties using artificial intelligence rather than traditional programming. This hierarchy clarifies what entities actually do, regardless of marketing labels.

The six-question evaluation rubric makes the framework actionable: Does it have goals? Does it perceive? Does it act? Does it iterate? Does it adapt? Does it stop? Apply these questions to vendor claims. Entities failing the iteration test—no matter how sophisticated their single-shot responses—aren't agentic systems. Entities lacking clear termination mechanisms raise operational risks. The rubric transforms theoretical distinctions into practical evaluation.

Four analytical dimensions organize how disciplines define agency differently. The **autonomy spectrum** ranges from delegated authority to self-directed initiative. **Entity frames** span humans to institutions to machines. **Goal dynamics** progress from accepting given objectives to negotiating them. **Persistence requirements** distinguish one-shot interactions from ongoing processes. These dimensions explain why philosophy emphasizes intentionality while AI emphasizes tool orchestration—different disciplines inhabit different regions of this conceptual space, but all contribute legitimate insights.

7.2 Seven Decades of Convergence

This framework synthesizes scholarship from Anscombe (1957)'s analysis of practical knowledge through Russell and Norvig (2020)'s perception-action loops to Willison (2025)'s tools-in-a-loop consensus. Early definitions centered on humans or human-AI relationships, rooted in philosophy, psychology, law, and economics. The 1990s computational revolution expanded entity frames to pure machines. The LLM era sharpened around "tools-in-a-loop" architectures while introducing hybrid framings where humans provide goals and oversight while AI executes multi-step tasks.

Willison's practitioner synthesis—"LLM agent runs tools in a loop to achieve a goal"—captures emerging consensus across research (Yao et al. 2022), commercial frameworks (LangChain, LlamaIndex), and vendor implementations. This convergence grounds our framework in both theoretical foundations and operational reality. The concepts are old; the technologies are new; the challenge is synthesis.

7.3 Putting the Framework to Work

For legal and financial applications, general agency definitions require additional safeguards: attribution to authoritative sources, auditable provenance, escalation protocols recognizing human judgment needs, and confidentiality mechanisms protecting privilege. These augment rather than replace the six-property framework—they define what makes entities suitable for high-stakes professional work.

For Practitioners: Evaluate Rigorously, Deploy Carefully. Evaluate vendor claims using the six-question rubric—demand evidence for each property. Recognize that higher autonomy brings both capability and risk—align deployment with organizational readiness. Insist on attribution and provenance mechanisms; absent these, verification becomes prohibitive. Treat AI agents as you would junior associates: supervise, verify, correct.

For Researchers: Build on These Foundations. Develop comprehensive benchmarks assessing agentic properties beyond accuracy—iteration quality, adaptation effectiveness, escalation appropriateness. Study real-world deployments to understand adoption patterns and failure modes. Address responsibility attribution when high-autonomy agents err—traditional agency law provides foundations but novel questions demand new frameworks. Extend cross-disciplinary synthesis to medical AI, educational AI, and other professional domains.

For Regulators: Regulate Architecture, Not Marketing. Base regulations on architectural properties (autonomy level, capability level) rather than vendor labels. Require transparency mechanisms enabling verification of attribution and provenance claims. Mandate human oversight aligned with autonomy levels—higher autonomy demands stronger oversight. Coordinate with professional ethics bodies to align AI governance with fiduciary and professional responsibilities.

Connections within this textbook. Part II (*How to Build an Agent*) explores architectures like ReAct and Reflexion in depth. Part III (*How to Govern an Agent*) addresses professional safeguards,

liability frameworks, and regulatory approaches. Additional chapters cover evaluation methods, deployment patterns, and domain-specific considerations for legal practice and financial services.

Recommended reading. For deeper engagement: Bratman (1987) (philosophical BDI foundation), Bandura (1989) (psychological agency), American Law Institute (2006) (legal foundations), Jensen and Meckling (1976) (principal-agent theory), Russell and Norvig (2010) (comprehensive AI), Wooldridge (2009) (multi-agent systems), and Xi et al. (2023) (LLM-based agents).

7.4 Why This Matters

The distinction between marketing labels and architectural reality has direct professional consequences. Our framework enables four critical capabilities:

Clearer Evaluation. Distinguish genuinely agentic systems from chatbots by assessing whether they meet all six operational properties or merely respond to single prompts. When vendors claim "AI agents," demand evidence of iteration, adaptation, and termination—not just conversational ability.

Better Research Coordination. Ensure scholars reference compatible concepts by specifying which properties their systems exhibit rather than relying on ambiguous labels like "agents" or "autonomous systems." This precision enables cumulative progress across research groups.

Informed Regulation. Craft policies targeting measurable capabilities—the six operational properties and autonomy levels—rather than vendor marketing claims. Architecture-based regulation withstands technological change better than product-category regulation.

Vendor Accountability. Hold companies to falsifiable claims by requiring demonstration of all six operational properties, not just AI-powered text generation. This shifts procurement from marketing promises to verified capabilities.

The stakes are particularly high in professional domains like law, medicine, and finance, where autonomous action by AI systems raises questions of liability, professional responsibility, and public safety. Without shared definitions linking architectural properties to operational requirements, we cannot write enforceable contracts, establish professional standards, or hold entities accountable.

The journey from "driving cattle" to "pleading a case"—both senses of Latin *agō*—to "running tools in a loop" spans millennia of human intellectual development and decades of computational innovation. The conceptual foundations traced here continue through the remaining chapters of this textbook and into your professional practice. Apply them rigorously. The technology evolves rapidly; the principles endure.

At a Glance: The Framework in One Page

Six Properties (Mnemonic: GPA + IAT)

G (Goal): Objective directing behavior toward desired outcomes
P (Perception): Observing environment to inform action selection
A (Action): Executing decisions that affect environment or state
I (Iteration): Multiple perception-action cycles, not single-shot
A (Adaptation): Modifying policy based on observations within session
T (Termination): Explicit or implicit stopping conditions

Note: the mnemonic reuses "A" twice—first for Action, then for Adaptation. When we need extra precision, we distinguish A_{act} (action) from A_{adapt} (adaptation).

Three-Level Hierarchy

Level 1 (Agent): G + P + A — 3 properties minimum
Level 2 (Agentic System): G + P + A + I + A + T — All 6 via rules/algorithms
Level 3 (Agentic AI): G + P + A + I + A + T — All 6 via LLMs/neural networks

Quick check: Q1–Q3 yes = Agent | Q1–Q6 yes = Agentic System | All 6 + AI/ML = Agentic AI

Example: Portfolio Rebalancer (L2)	**Example: Legal Research (L3)**
Monitors positions, executes trades, adapts to volatility, stops when balanced. *(6 properties via rules)*	Searches case law, refines queries based on results, escalates when stuck. *(6 properties via LLM)*

Professional Deployment Adds Four Safeguards

Attribution: Every claim traceable to authoritative sources
Provenance: Auditable logs showing reasoning steps and tool invocations
Escalation: Clear triggers for human review of high-stakes decisions
Confidentiality: Ethical walls and privilege protection mechanisms

References

American Law Institute (1958). *Restatement of the Law, Second, Agency*. St Paul, MN: American Law Institute Publishers.

American Law Institute (2006). *Restatement of the Law Third, Agency*. St Paul, MN: American Law Institute Publishers.

Anscombe, G. E. M. (1957). *Intention*. Oxford: Basil Blackwell.

Anthropic (2025). *Sub-Agents*. Claude Code documentation on prompt-based sub-agent framework. URL: https://code.claude.com/docs/en/sub-agents (visited on 01/12/2025).

Bandura, Albert (1989). "Human Agency in Social Cognitive Theory". In: *American Psychologist* 44.9, pp. 1175–1184.

Bonabeau, Eric (2002). "Agent-Based Modeling: Methods and Techniques for Simulating Human Systems". In: *Proceedings of the National Academy of Sciences* 99.suppl 3, pp. 7280–7287. DOI: 10.1073/pnas.082080899.

Bratman, Michael E. (1987). *Intention, Plans, and Practical Reason*. Foundational work establishing the BDI (Belief-Desire-Intention) model of practical reasoning. Cambridge, MA: Harvard University Press. ISBN: 978-0674003989.

Davidson, Donald (1963). "Actions, Reasons, and Causes". In: *Journal of Philosophy* 60, pp. 685–700. DOI: 10.2307/2023177.

Dennett, Daniel C. (1987). *The Intentional Stance*. Cambridge, MA: MIT Press.

Epstein, Joshua M. (2006). *Generative Social Science: Studies in Agent-Based Computational Modeling*. Princeton University Press.

Epstein, Joshua M. and Robert Axtell (1996). *Growing Artificial Societies: Social Science from the Bottom Up*. Cambridge, MA: MIT Press.

Ferber, Jacques (1999). *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*. Addison-Wesley.

Franklin, Stan and Art Graesser (1997). "Is It an Agent, or Just a Program? A Taxonomy for Autonomous Agents". In: *Intelligent Agents III (ATAL'96)*. Vol. 1193. Lecture Notes in Computer Science. Springer, pp. 21–35. DOI: 10.1007/BFb0013570.

Giddens, Anthony (1984). *The Constitution of Society: Outline of the Theory of Structuration*. University of California Press.

Holland, John H. (1995). *Hidden Order: How Adaptation Builds Complexity*. Addison-Wesley.

Jennings, Nicholas R., Katia Sycara, and Michael Wooldridge (1998). "A Roadmap of Agent Research and Development". In: *Autonomous Agents and Multi-Agent Systems* 1.1, pp. 7–38. DOI: 10.1023/A:1010090405266.

Jensen, Michael C. and William H. Meckling (1976). "Theory of the Firm: Managerial Behavior, Agency Costs and Ownership Structure". In: *Journal of Financial Economics* 3.4, pp. 305–360.

Kauffman, Stuart A. (2000). *Investigations*. Oxford University Press.

LangChain Documentation (2024). *Agents*. URL: https://python.langchain.com/docs/modules/agents/ (visited on 10/26/2025).

Lewis, Charlton T. and Charles Short, eds. (1879). *A Latin Dictionary. Founded on Andrews' Edition of Freund's Latin Dictionary*. Perseus Digital Library. Oxford: Clarendon Press. URL: https://www.perseus.tufts.edu/hopper/text?doc=Perseus:text:1999.04.0059 (visited on 10/26/2025).

Maes, Pattie (1994). "Agents that Reduce Work and Information Overload". In: *Communications of the ACM* 37.7, pp. 30–40. DOI: 10.1145/176789.176792.

Mata v. Avianca, Inc. (2023). *678 F. Supp. 3d 443 (S.D.N.Y. 2023)*. United States District Court for the Southern District of New York, No. 1:22-cv-01461 (PKC). Sanctions order for attorney submitting AI-generated fictitious case citations; early precedent on professional responsibility for AI tool use in legal research. URL: https://scholar.google.com/scholar_case?case=13849516024529169137 (visited on 11/13/2025).

Milgram, Stanley (1974). *Obedience to Authority: An Experimental View*. Harper & Row.

Minsky, Marvin (1986). *The Society of Mind*. Simon & Schuster.

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, et al. (2015). "Human-level control through deep reinforcement learning". In: *Nature* 518, pp. 529–533. DOI: 10.1038/nature14236.

OpenAI (2024). *OpenAI Agents SDK*. URL: https://openai.github.io/openai-agents-python/ (visited on 10/31/2025).

Park, Joon Sung, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein (2023). "Generative Agents: Interactive Simulacra of Human Behavior". In: arXiv: 2304.03442 [cs.HC]. URL: https://arxiv.org/abs/2304.03442 (visited on 10/27/2025).

Russell, Stuart J. and Peter Norvig (1995). *Artificial Intelligence: A Modern Approach*. 1st ed. First edition establishing foundational sensor-actuator agent framework; core definition remains stable through 4th edition. Upper Saddle River, NJ: Prentice Hall.

Russell, Stuart J. and Peter Norvig (2010). *Artificial Intelligence: A Modern Approach*. 3rd ed. Prentice Hall.

Russell, Stuart J. and Peter Norvig (2020). *Artificial Intelligence: A Modern Approach*. 4th ed. Standard AI textbook establishing sensor-actuator agent framework; 4th edition incorporates modern ML/AI developments. Hoboken, NJ: Pearson. ISBN: 9780134610993. URL: http://aima.cs.berkeley.edu/ (visited on 10/27/2025).

Schlosser, Markus (2019). "Agency". In: *The Stanford Encyclopedia of Philosophy*. Ed. by Edward N. Zalta and Uri Nodelman. Metaphysics Research Lab, Stanford University. URL: https://plato.stanford.edu/archives/fall2019/entries/agency/ (visited on 10/27/2025).

Shardanand, Upendra and Pattie Maes (1995). "Social Information Filtering: Algorithms for Automating 'Word of Mouth'". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Describes Ringo and collaborative filtering for personalized recommendations. Denver, CO: ACM Press/Addison-Wesley, pp. 210–217. DOI: 10.1145/223904.223931.

Shoham, Yoav (1993). “Agent-Oriented Programming”. In: *Artificial Intelligence* 60.1, pp. 51–92. DOI: 10.1016/0004-3702(93)90034-9.

Silver, David, Aja Huang, Chris J. Maddison, et al. (2016). “Mastering the game of Go with deep neural networks and tree search”. In: *Nature* 529, pp. 484–489. DOI: 10.1038/nature16961.

Sutton, Richard S. and Andrew G. Barto (1998). *Reinforcement Learning: An Introduction*. MIT Press.

Weiss, Gerhard (1999). *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press.

Willison, Simon (Sept. 2025). *I think “agent” may finally have a widely enough agreed upon definition to be useful jargon now*. Defines LLM agents as systems running tools in loops to achieve goals; consolidates definition emerging from Anthropic and broader LLM community. URL: https://simonwillison.net/2025/Sep/18/agents/ (visited on 11/13/2025).

Wilson, George and Samuel Shpall (2022). “Action”. In: *The Stanford Encyclopedia of Philosophy*. Ed. by Edward N. Zalta and Uri Nodelman. Metaphysics Research Lab, Stanford University. URL: https://plato.stanford.edu/archives/spr2022/entries/action/ (visited on 10/27/2025).

Wooldridge, Michael (2009). *An Introduction to MultiAgent Systems*. 2nd ed. John Wiley & Sons.

Wooldridge, Michael and Nicholas R. Jennings (1995). “Intelligent Agents: Theory and Practice”. In: *The Knowledge Engineering Review* 10.2, pp. 115–152. DOI: 10.1017/S0269888900008122.

Xi, Zhiheng et al. (2023). “The Rise and Potential of Large Language Model Based Agents: A Survey”. In: *arXiv preprint arXiv:2309.07864*.

Yao, Shunyu et al. (2022). “ReAct: Synergizing Reasoning and Acting in Language Models”. In: arXiv: 2210.03629 [cs.AI]. URL: https://arxiv.org/abs/2210.03629 (visited on 10/25/2025).