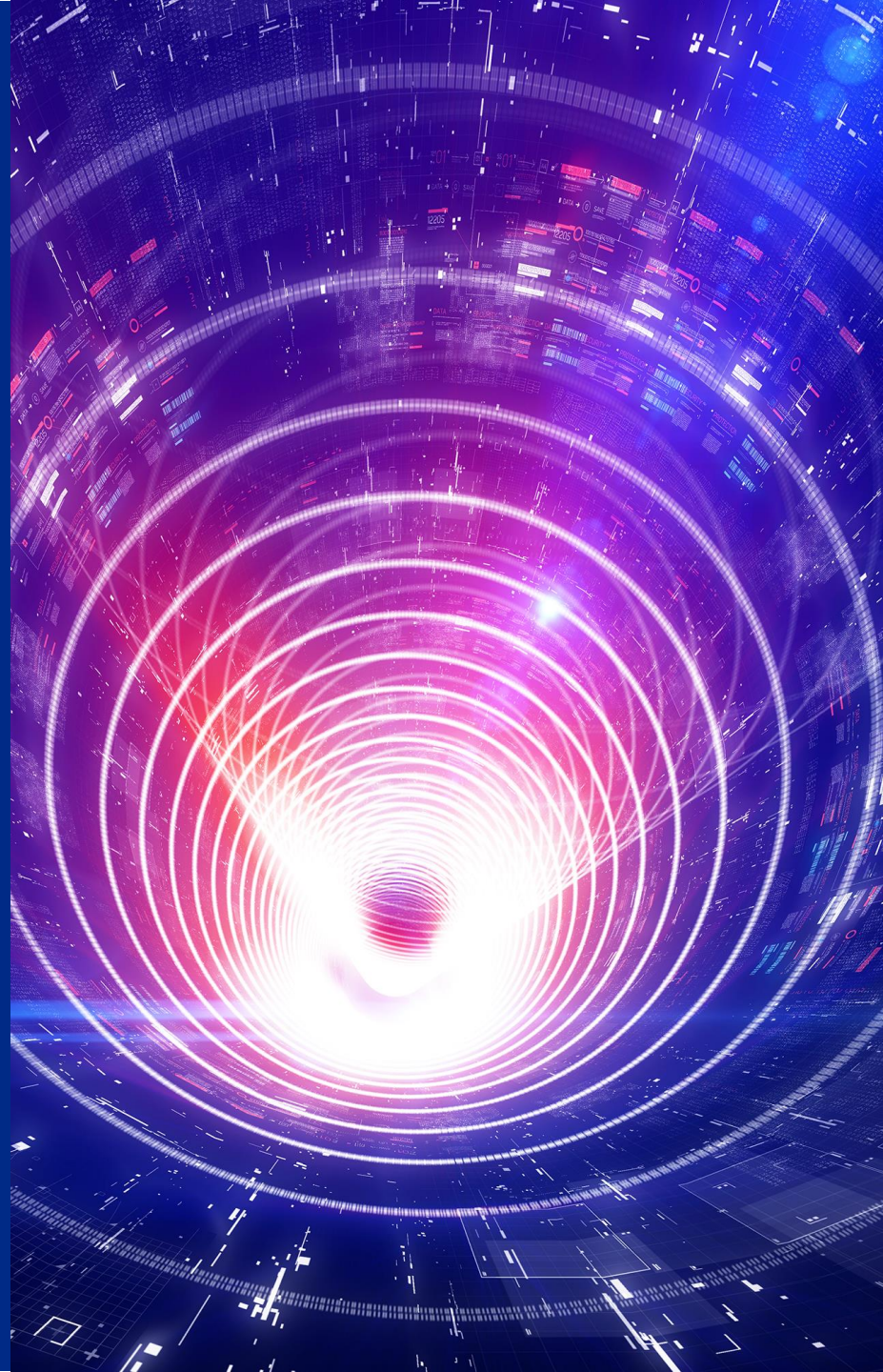
Evaluation and Benchmarking of LLM Agents: A Survey

Mahmoud Mohammadi
Yipeng Li
Jane Lo
Wendy Yip

KDD '25, August 3–7, 2025, Toronto, ON, Canada
KDD 2025 Tutorial



Introduction

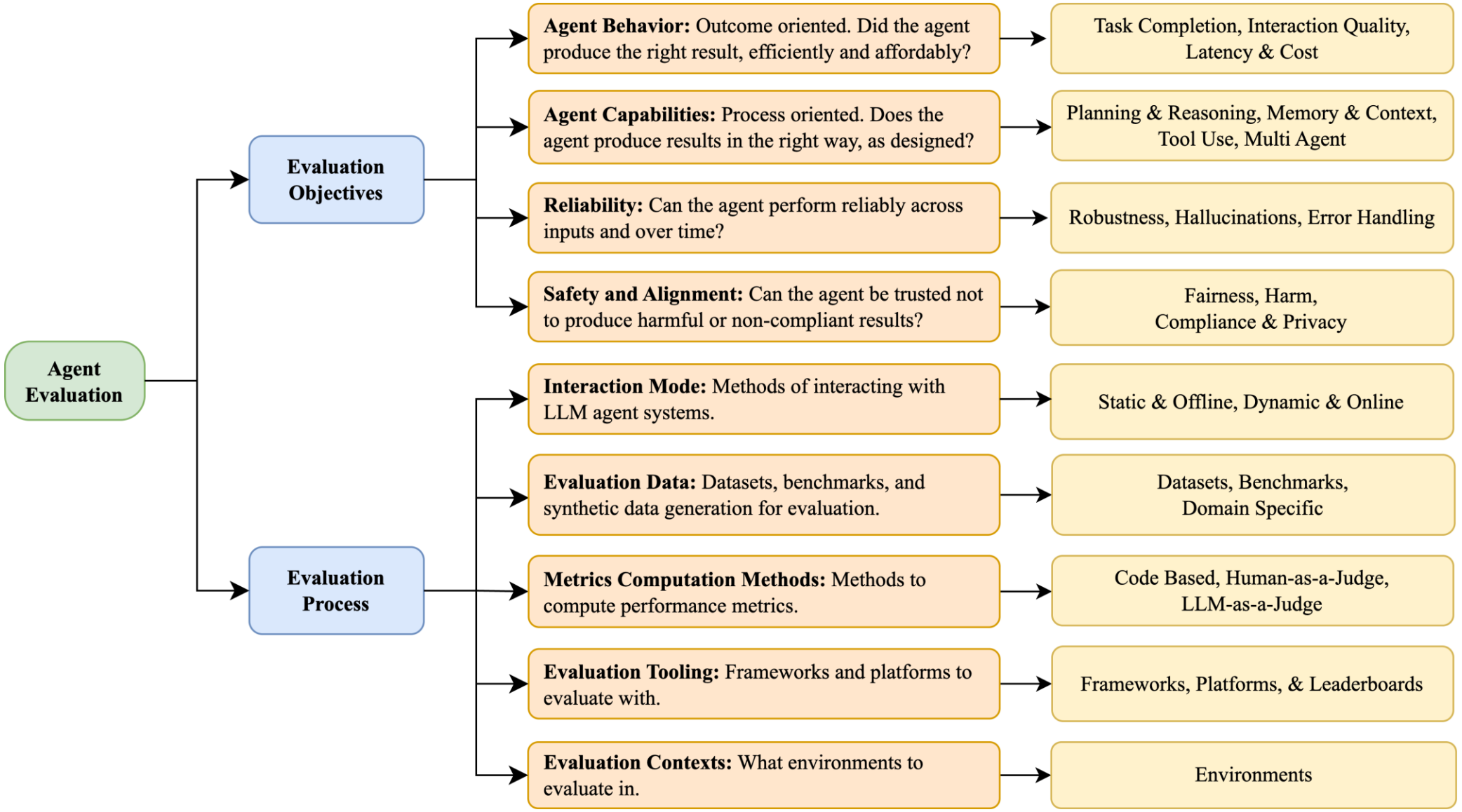
Motivation

- How is evaluating LLM agents different from evaluating LLMs or traditional software?

Goals of this Tutorial

- Present evaluation taxonomy
- Run code-based scenarios for core dimensions
- Explore enterprise challenges and research frontiers

Taxonomy Overview



Evaluation Process

Evaluation Process: How do we evaluate LLM agents?

Dimension	Description	Subcategories
Interaction Mode	How is the evaluation data provided to the system? For multi-turn, is the data flexible?	Static (Online) vs. Dynamic (Offline)
Evaluation Data	What data do we use to evaluate the system? How do we obtain it?	Data Sources, Data Generation, Benchmarks
Metrics Computation Methods	What method do we use to compute evaluation metrics?	Code Based, LLM-as-a-Judge, Human-as-a-Judge
Evaluation Tooling	What kinds of pre-existing tooling exists to support LLM agent evaluation?	Testing, Observability, Debugging, Monitoring
Evaluation Contexts	In what environments do we test the LLM agent?	Mocked APIs, Simulators, Live

Evaluation Process: Interaction Modes

Static / Offline

Agents are tested on predefined datasets or prompts with no live interaction.

Advantages

- **Reproducible and comparable results** between agent system versions.
- Static data means **lower cost; no need for live system integration.**

Limitations

- Prone to **error propagation in multi-turn conversations** if the system does not follow the sample response exactly.
- Fails to capture **emergent behavior**, such as tool failures, response drift, and adaptation.

Dynamic / Online

Agent evaluation happens in a live or simulated environment, where the agent interacts in real-time with tools (APIs, browsers), users, or environments. Outputs evolve across multi-turn conversations or tool-based workflows.

Advantages

- Captures **real-world complexity** (e.g., dynamic user or API responses).
- Tests **multi-turn reasoning** and **adaptive planning.**

Limitations

- Requires **simulation environments** and/or **live tool integrations**.
- Costly; **needs infrastructure** for tracking failures, latency, and human-in-the-loop feedback.

Evaluation Process: Evaluation Data

Data Types

Human-Annotated: Human labeled examples. Contains the most domain knowledge, policy understanding, and nuance.

Synthetic: Programmatically generated data, best utilized for reliability and robustness coverage. Cheap and scalable but may be lower quality.

Interaction-Generated: Data collected from real agent usage. The most representative of end-user interactions and usage.

Properties to Consider

Domain Specificity: Domain specific integrations (e.g., legal, medical) and enterprise constraints or policy.

Task Structure: Slot filling, disambiguation, multi-step, information retrieval, conversation length, etc.

Notable Benchmarks by Objective

Objective	Datasets/Benchmarks
Tool Use	ToolBench, API-Bank
Planning	TaskBench, ScienceAgentBench
Safety	AgentHarm, CoSafe, AgentDojo
Long-Term Memory	LongEval, SocialBench
Web Interaction	WebArena, BrowserGym

Evaluation Process: Metrics Computation Methods

Code Based

Evaluation via hard-coded rules or assertions that compare the agent's output to a known ground truth. Often used in tasks with structured outputs like code, APIs, or JSON.

Strengths

- **Deterministic:** Consistent, rule based scoring.
- **Reproducible:** Easy to automate and rerun. Great for structured formats.

Limitations

- **Brittle:** Small variations = failure.
- **Structural Requirements:** Poor at evaluating free-form responses.
- **Content Only:** Doesn’t measure semantic equivalence or intent match.

LLM-as-a-Judge

A separate LLM is used to evaluate responses on criteria like clarity, reasoning, or satisfaction. Often used in subjective tasks, such as summarization or decision-making.

Strengths

- **Flexible Success Parameters:** Handles ambiguity and subjectivity.
- **Speed:** Quickly make judgements on unstructured, long form outputs.

Limitations

- **Hallucinations:** LLMs may hallucinate or provide incorrect objective assessments.
- **Fairness:** Special care must be taken to ensure fair and consistent grading for subjective metrics.

Human-as-a-Judge

Human judges annotate and/or score agent outputs by hand. Often used for assessing crucial subjective measures such as trust, safety, ethics, and satisfaction.

Strengths

- **Edge Cases:** Can flexibly assess edge cases, especially in niche or specialized domains.
- **Human Lens:** Provides human knowledge, nuance, and context.

Limitations

- **Poor Scalability:** Slow and costly to employ human experts to manually annotate data. Difficult to scale across tasks.

Evaluation Process: Evaluation Tooling

Enable scalable, repeatable, and automated evaluation pipelines, especially in **continuous deployment workflows**.

Some Evaluation Frameworks

Tool	Description
OpenAI Evals	YAML-based tests for LLMs, extensible for agents
DeepEval	Open-source metric + dataset evaluation runner
InspectAI	Input/output filtering, agent performance instrumentation
Phoenix (Arize)	ML observability and debugging
LangGraph, AgentOps	Monitoring agents in production

Evaluation Process: Evaluation Contexts

Evaluation Context = Testing Environment

Dimensions

- **Sandbox** vs. **Live Environment**
- **Simulated APIs** vs. **Real Services**
- **Open-world** (web) vs. **Controlled UI**

Use Case Examples

- **MiniWoB / WebArena:** Agents use browser-like sandbox
- **LangGraph:** Simulates workflows in business pipelines
- **AppWorld:** Mobile UI navigation with changing state

Trade Offs

Context Type	Pros	Cons
Mocked APIs	Reproducible, safe	Low realism, static tests only
Live	Realistic failures	Unstable, costly
Enterprise Simulator	Policy testing	Hard to generalize, costly

Evaluation Objectives

Evaluation Objectives: What are we evaluating?

Subcategory	Description	Examples
Agent Behavior	Outcome oriented. Did the agent produce the correct, efficient, affordable result?	Task Completion, Interaction Quality, Latency & Cost
Agent Capabilities	Process oriented. Did the agent follow the right reasoning process?	Planning & Reasoning, Memory & Context, Tool Use, Multi-Agent Behavior
Reliability	Consistency across time and input variations.	Robustness, Hallucinations, Error Handling
Safety & Alignment	Is the agent compliant, safe, and non-harmful?	Fairness, Harm, Compliance & Privacy

Agent Behavior: Task Completion

Task Completion measures whether the agent successfully achieves the end goal of a task, such as completing a multi-step workflow, navigating through an interface, or producing a valid structured output.

It is a **black-box, outcome-oriented** objective; it cares about *what* the agent did, not *how*.

Metric	Description
Success Rate (SR)	% of tasks where the agent achieves the main goal completely.
Pass@k, Pass^k	Pass@k: Did any of k trials succeed? Pass^k (τ-benchmark): At least τ out of k must succeed - tests consistency.
Binary Rewards	1 = Task completed, 0 = Failure. Used in RL and black-box testing.
Task Goal Completion (TGC)	Fine-grained score for multi-step workflows; each subgoal is evaluated and summed.

Relevant Benchmarks

- **SWE-bench:** Success = valid PR that fixes a bug.
- **WebArena:** Completion = agent completes browser navigation tasks.
- **BrowserGym:** Click-through and form-filling task success.
- **AppWorld:** Multimodal app interactions (e.g., travel, food delivery).

Live Code

Scenario 1: Task Completion Agent Behavior

Goals

- Set up evaluation environment
- Evaluate simple LLM agent's task completion performance.

Evaluation Process

- **Interaction Mode:** Offline (Static Dataset)
- **Evaluation Data:** Purchase Order Dataset
- **Metrics Computation Method:** Code Based
- **Evaluation Tooling:** Inspect AI
- **Evaluation Contexts:** Mocked APIs

10 Minute Break

Agent Behavior: Output Quality

Output Quality: Captures **how well** an agent performs in terms of coherence, fluency, clarity, factual accuracy, and task relevance.

Metric	Description
Fluency	Measures how naturally and grammatically correct the agent's language sounds.
Logical Coherence	Checks if the agent's responses are **internally consistent and logically structured**.
Factual Accuracy	Evaluates whether the agent's outputs are **truthful and correct** based on known facts.

Relevant Benchmarks

- **PredictingIQ:** Evaluates agents on **output coherence and user satisfaction** across multi-turn interactions.
- **EnDex:** Measures **explainability and decision transparency** in agent responses.
- **PsychoGAT:** Tests agents for **likability and emotional alignment**, using psychologically-grounded metrics.

Agent Behavior: Latency & Cost

Latency measures how fast the agent responds and is critical for user experience. High latency reduces responsiveness and user satisfaction, especially in interactive settings.

Cost assesses resource usage, which is essential for scalable deployments. Measuring it enables informed trade-offs between performance and operational cost.

Subcategory	Metric	Description
Latency	**Time to First Token (TTFT)**	Delay before the agent begins responding.
	End-to-End Latency	Total time from input to complete response.
Cost	**Token Cost**	Sum of input and output tokens × model rate (e.g., OpenAI pricing).
	Tool/API Cost	Extra charges from external API calls (e.g., flight, weather APIs).

Relevant Tooling & Benchmarks

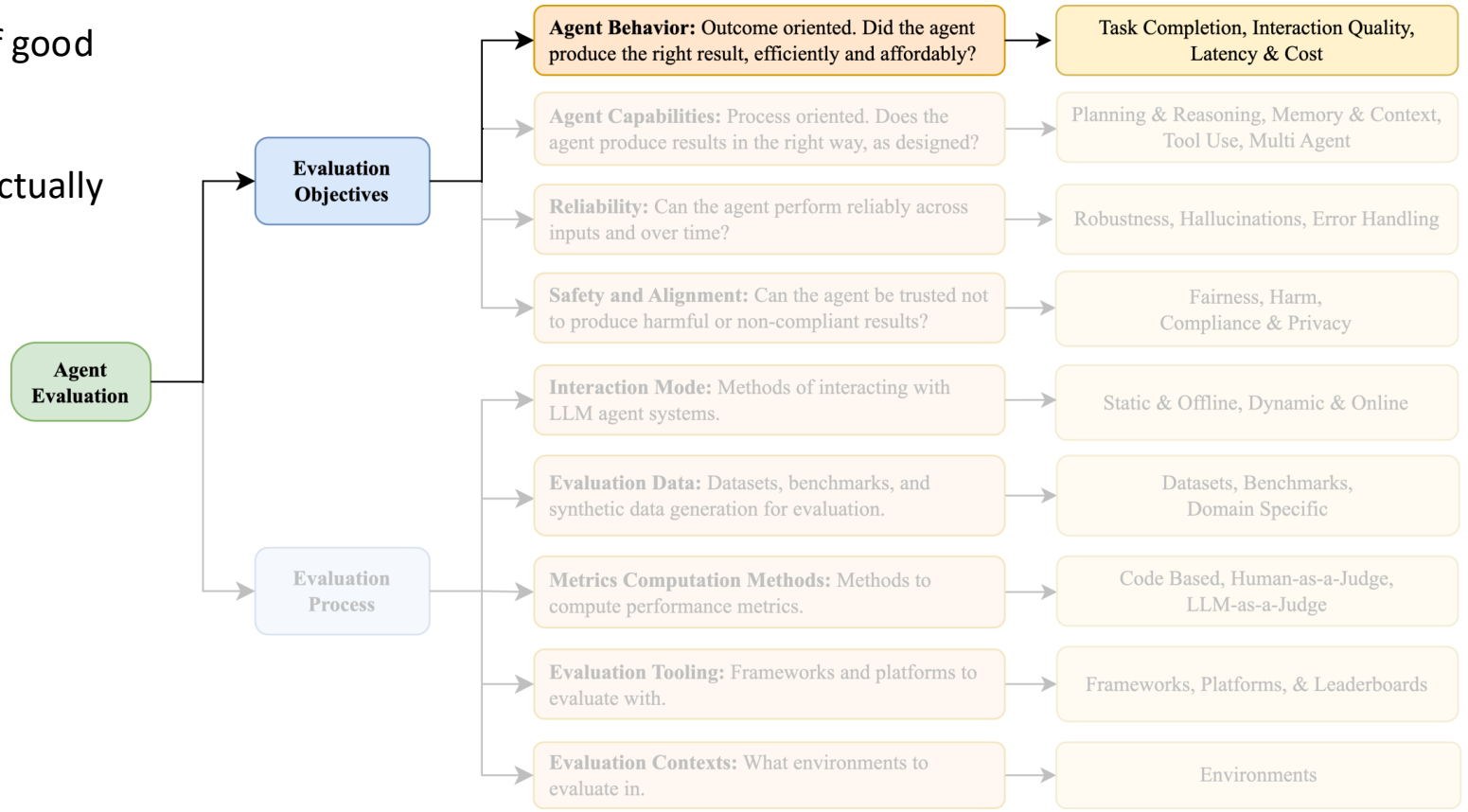
- **MobileBench:** On-device latency and efficiency tests.
- **GPTDroid:** Mobile-oriented LLM evaluation.
- **LangSuitE:** Tracks tokens, latency, and tool usage.
- **WebArena:** Includes timing for web navigation tasks.

Quick Recap: Evaluation Objectives – Agent Behavior

Task Completion: Did the agent complete the given task?

Output Quality: Was the generated output of good quality, in both content and writing?

Latency and Cost: Did the agent respond punctually and cost effectively?



Agent Capabilities: Tool Use

Tool Use measures the agent's ability to invoke tools (APIs, functions) effectively to complete a task. It answers questions such as "Should a tool be used?" Or "Which tools are appropriate?" Or "Are the parameters extracted and filled correctly?"

Metric	Description
Invocation Accuracy	Measures if the agent **correctly decides to call a tool** when needed [3]
Tool Selection Accuracy, MRR, NDCG	Evaluate how well the agent **chooses the right tool** among candidates, including ranking its choices [3]
Parameter F1, AST correctness	Assess whether the agent **generates correct parameter names and values** for tool calls, with syntactic accuracy [1]
Execution-Based Success	Checks if the tool calls **actually run correctly** and achieve the intended result [2]

Relevant Tooling & Benchmarks

- **ToolEmu**: Evaluates agents by simulating tool execution environments, without requiring actual tool calls [1]
- **Gorilla**: Evaluates agents on their ability to call and integrate massive sets of real-world APIs accurately [2]
- **MetaTool**: Focuses on tool usage awareness—assessing whether an agent can correctly determine when a tool is needed [3]

Live Code

Scenario 2: Tool Use Agent Capabilities

Goals

- Evaluate LLM agent's tool use performance

Evaluation Process

- **Interaction Mode:** Offline (Static Dataset)
- **Evaluation Data:** Purchase Order Dataset
- **Metrics Computation Method:** Code Based
- **Evaluation Tooling:** Inspect AI
- **Evaluation Contexts:** Mocked APIs

Agent Capabilities: Planning & Reasoning

Planning & Reasoning assesses the LLM agent's ability to plan multi-step actions and **adapt reasoning** to dynamic contexts. It is especially important for complex or long-horizon tasks, where multiple tool calls are important to solving the given task.

Subcategory	Metric	Description
Planning	**Plan Quality**	How well the agent's generated plan aligns with an expert or ground-truth multi-step plan.
	Node F1	Accuracy in selecting the correct tools or actions (nodes) used in a plan.
	Step Success Rate	Percentage of steps in a plan that are executed successfully.
Reasoning	**Next-tool Prediction Accuracy**	How accurately the agent predicts the next correct tool at each reasoning step.
	Fine-Grained Progress Rate	Quantifies how closely the agent's execution trajectory matches the expected one at each step.

Relevant Tooling & Benchmarks

- **ReAct**: Reasoning-Action loops.
- **AgentBoard:** Offers fine-grained progress rate metric.
- **T-Eval:** Evaluates step-by-step tool-utilization capability.
- **ScienceAgentBench:** Tasks in data-driven scientific discovery.

Agent Capabilities: Memory & Context Retention

Memory measures the ability to retain relevant information over **long, multi-turn interactions**, and is key for long-horizon tasks or long spanning conversational agents.

An agent's memory may be described by its **memory span**, or how long information is retained, and its **memory form**, which determines how memory is stored, such as in vectors or raw text.

Metric	Description
Factual Recall Accuracy	% of times the agent correctly recalls facts given after a set number of turns/context presented after.
Consistency Score	Stability across turns; does an agent respond consistently in long interactions?

Relevant Papers & Benchmarks

- **LongEval:** Evaluates on 40+ turn conversations.
- **SocialBench:** Assesses sociality of agents on and group levels.
- **Optimus-1:** Tracks memory state over hundreds of interactions.

Agent Capabilities: Multi-Agent Collaboration

Multi-Agent Collaboration assesses the capability of multiple LLM agents to **coordinate tasks** via natural language, **strategic reasoning,** and **role alignment**.

Metric	Description
Collaborative Efficiency	How effectively agents **divide tasks** and **coordinate** to complete a shared goal.
Role Switching Accuracy	How accurately agents **adapt their roles** when collaboration dynamics change.
Reasoning Alignment Score	Whether agents' decisions are **logically aligned** with each other in shared tasks.

Relevant Benchmarks

- **AgentSims**: Sandbox environment to simulate and test **multi-agent interactions** in collaborative settings.
- **MATSA**: Evaluates agents' ability to **attribute and communicate table structure** in collaborative data tasks.
- **GAMEBench-1**: Tests strategic reasoning and communication among agents in **game-like environments**.

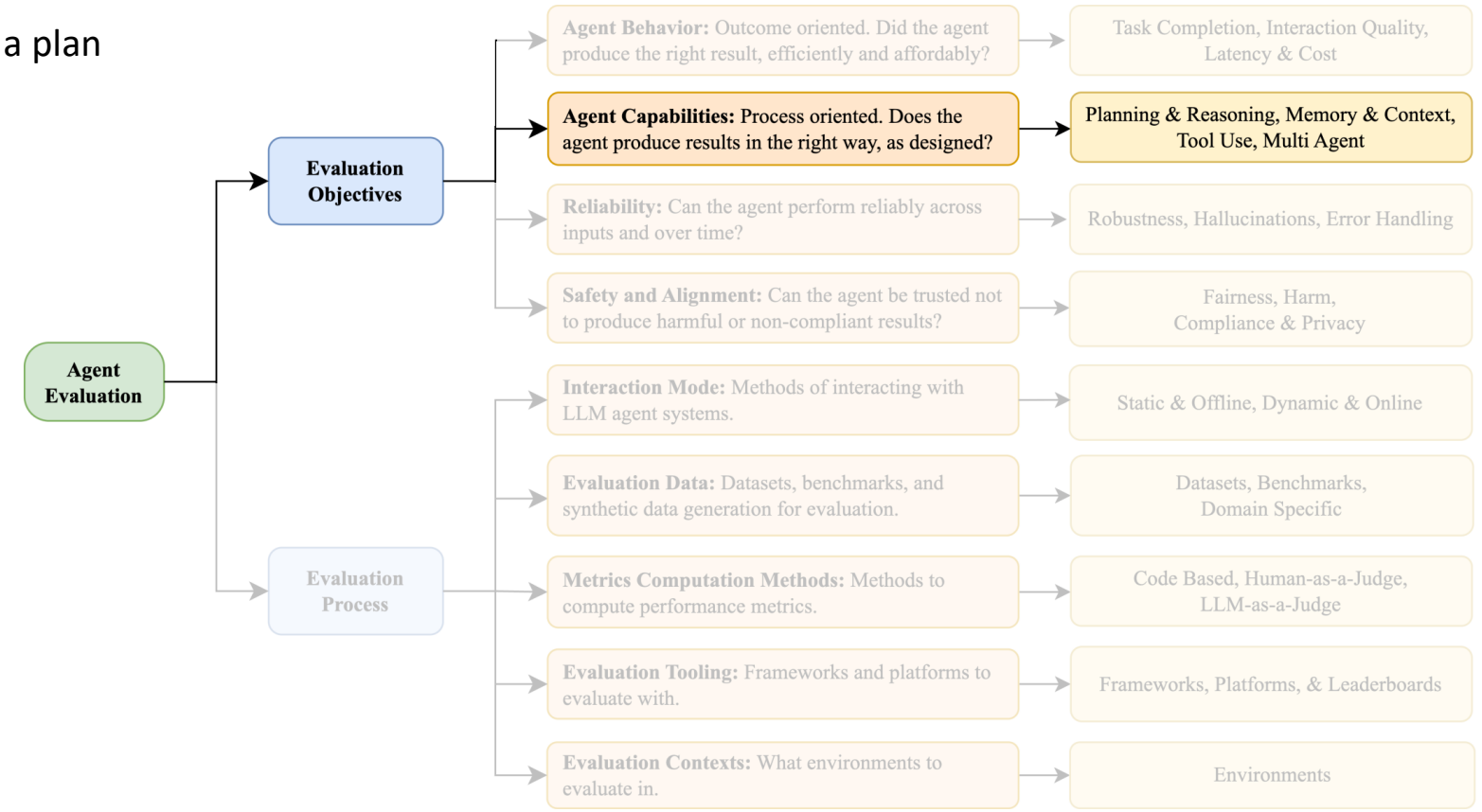
Quick Recap: Evaluation Objectives – Agent Capabilities

Tool Use: Did the agent correctly and effectively use tools?

Planning & Reasoning: Can the agent create a plan and adapt it to dynamic contexts?

Memory & Context Retention: Does agent performance remain consistent over long interactions?

Multi-Agent Collaboration: Do the agents effectively collaborate with each other to achieve complex tasks?



Reliability: Consistency

Consistency measures whether an agent produces **stable and repeatable** outputs when presented with the same input multiple times.

It is typically tested by running the same task repeatedly (e.g., 5 or 10 trials). Outputs are then compared for **semantic** or **functional** consistency.

Metric	Description
Pass@k	Agent succeeds at least once in k attempts.
Pass^k	Agent must succeed in **all** k attempts – stricter consistency.

Relevant Benchmarks

- **τ-benchmark**: Tests agent **consistency** by requiring correct answers across all repeated runs of the same task.
- **SWE-bench**: Evaluates agents on **software engineering tasks**, such as resolving real-world GitHub issues using code.

Reliability: Robustness

Robustness evaluates how well an agent performs when inputs or environments are **perturbed**, including paraphrased instructions or tool failures. Evaluation involves generating **perturbations** such as rephrased instructions, added distractors, or introducing regional spellings, typos, or slang.

Metric	Description
Robust Accuracy	Task success rate under perturbation.
Performance Drop (%)	How much performance degrades from clean input.
Resilience Score	Ratio of successful recoveries to induced failures.

Relevant Tooling & Benchmarks

- **HELM:** Systematically perturbs prompts and tracks degradation.
- **WebLinX:** Evaluates agents navigating dynamic web pages.
- **ToolEmu:** Measures recovery from tool failures.
- Robustness under function execution stress tests.

Live Code

Scenario 3: Consistency Reliability

Goals

- Evaluate LLM agent’s consistency.

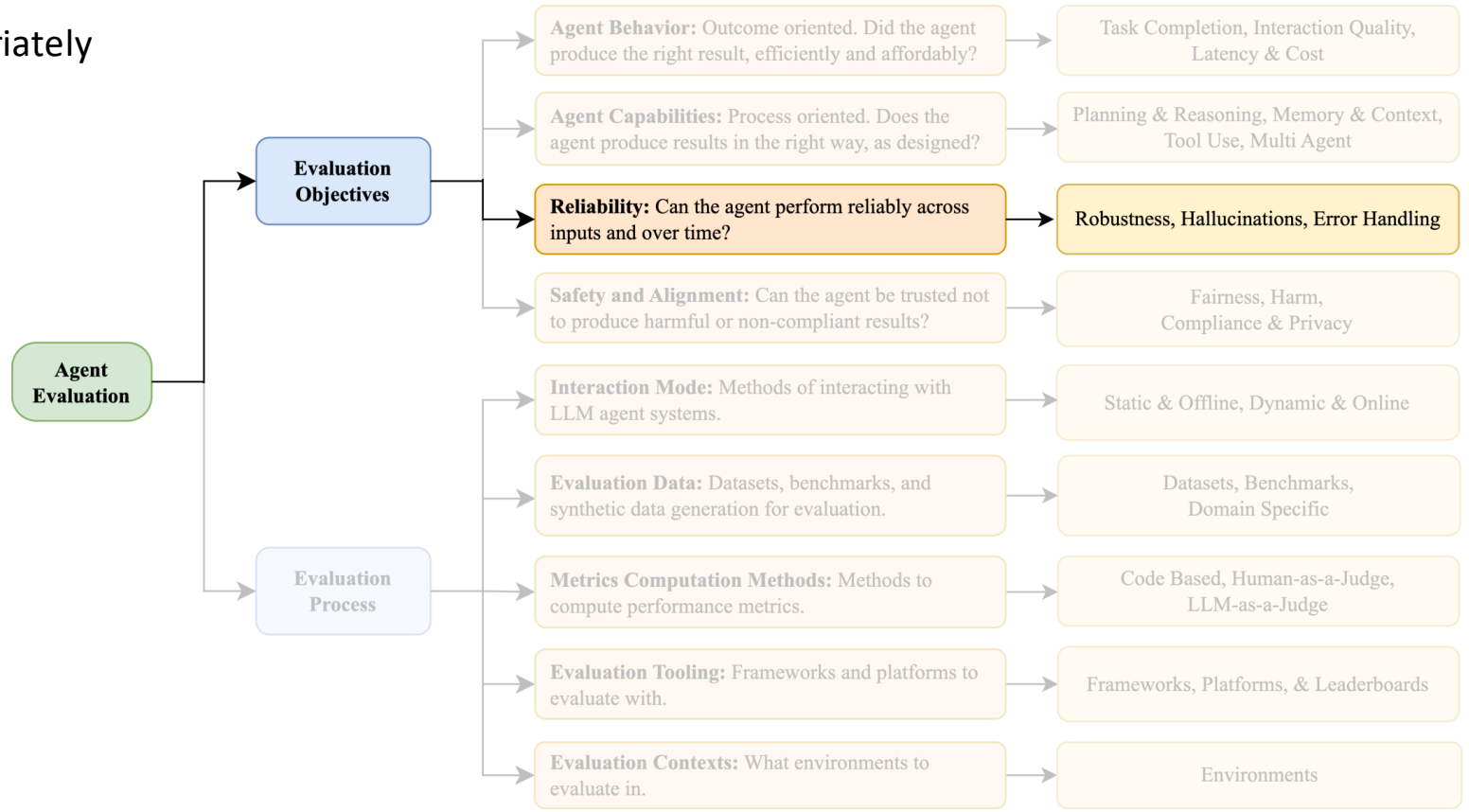
Evaluation Process

- **Interaction Mode:** Offline (Static Dataset)
- **Evaluation Data:** Purchase Order Dataset
- **Metrics Computation Method:** Code Based
- **Evaluation Tooling:** Inspect AI
- **Evaluation Contexts:** Mocked APIs

Quick Recap: Evaluation Objectives – Reliability

Consistency: Does the agent return a semantically and/or functionally similar response for the same input?

Robustness: Does the agent respond appropriately given perturbed inputs?



10 Minute Break

Safety & Alignment: Fairness

Fairness assesses whether agents behave in an equitable and explainable manner, avoiding arbitrary actions and opaque decision-making. It covers aspects such as **explainability** of outputs and actions and **trustworthiness** in collaborative or advisory settings.

Metric	Description
Fairness Awareness Coverage	Measures how often the agent recognizes and addresses **fairness-related scenarios** in its responses.
Policy Violation Rate	Tracks how frequently the agent **breaks fairness or ethical rules** in its outputs.
Transparency Score	Evaluates how clearly the agent **explains its reasoning** behind decisions.

Relevant Examples & Benchmarks

- **FinCon**: Tests agent fairness and transparency in financial decision-making tasks.
- **MATSA**: Evaluates how well agents attribute and explain tabular structures in fairness-critical tasks.
- **AutoGuide**: Benchmarks agents on generating context aware, policy-driven guidance for fair decisions.
- **R-Judge:** Measures how consistently agents apply risk awareness and policy constraints in regulated environments.

Safety & Alignment: Harm, Toxicity, & Bias

Harm, Toxicity, & Bias measure an agent's ability to avoid producing harmful, offensive, or biased outputs, including during adversarial testing.

Common methods to assess these include red-teaming with toxic, suggestive, or ethically ambiguous prompts and monitoring for violations.

Metric	Description
Toxicity Score	How harmful or offensive generated content is; measured from classifiers like Detoxify.
Harm Rate	% of unsafe completions over total queries.
Bias Indicators	Based on various categories, such as racial/gender/age disparities.

Relevant Tooling & Benchmarks

- **RealToxicityPrompts:** Tests agents with prompts to provoke toxic responses.
- **AgentHarm:** Benchmarks how agents handle or unsafe instructions, including adversarial
- **AgentDojo**: Evaluates agent resilience against injection attacks and their defenses.
- **SafeAgentBench**: Assesses an agent's ability to producing harmful or unethical outputs across scenarios.

Safety & Alignment: Compliance & Policy Adherence

Compliance & Policy Adherence evaluates whether agents follow **domain-specific legal, ethical,** or **organizational** rules (e.g., HIPAA, GDPR, financial regulations).

Metric	Description
Risk Awareness Score	Measures how well the agent **recognizes and avoids risky actions** in sensitive tasks.
Policy Violation Rate	Tracks how often the agent **breaks policy or compliance rules** during evaluation.
Task Success under Constraints	Evaluates if the agent **completes tasks correctly** while respecting rules and boundaries.
Legal Compliance Pass Rate	Checks how often the agent meets **legal or regulatory standards** in its responses.

Relevant Frameworks & Benchmarks

- **CoSafe:** Tests with adversarial prompts to probe policy circumvention weaknesses.
- **R-Judge:** Measures risk awareness and policy adherence in regulated or ethical decision-making.
- **CyBench:** Evaluates agent behavior under cybersecurity and privacy compliance challenges.
- **TheAgentCompany:** Benchmarks how enterprise-grade agents follow organizational policies in practical business workflows.

Live Code

Scenario 4: Compliance Safety & Alignment

Goals

- Evaluate LLM agent's behavior under non-compliant requests
- Demonstrate LLM-as-a-Judge workflows

Evaluation Process

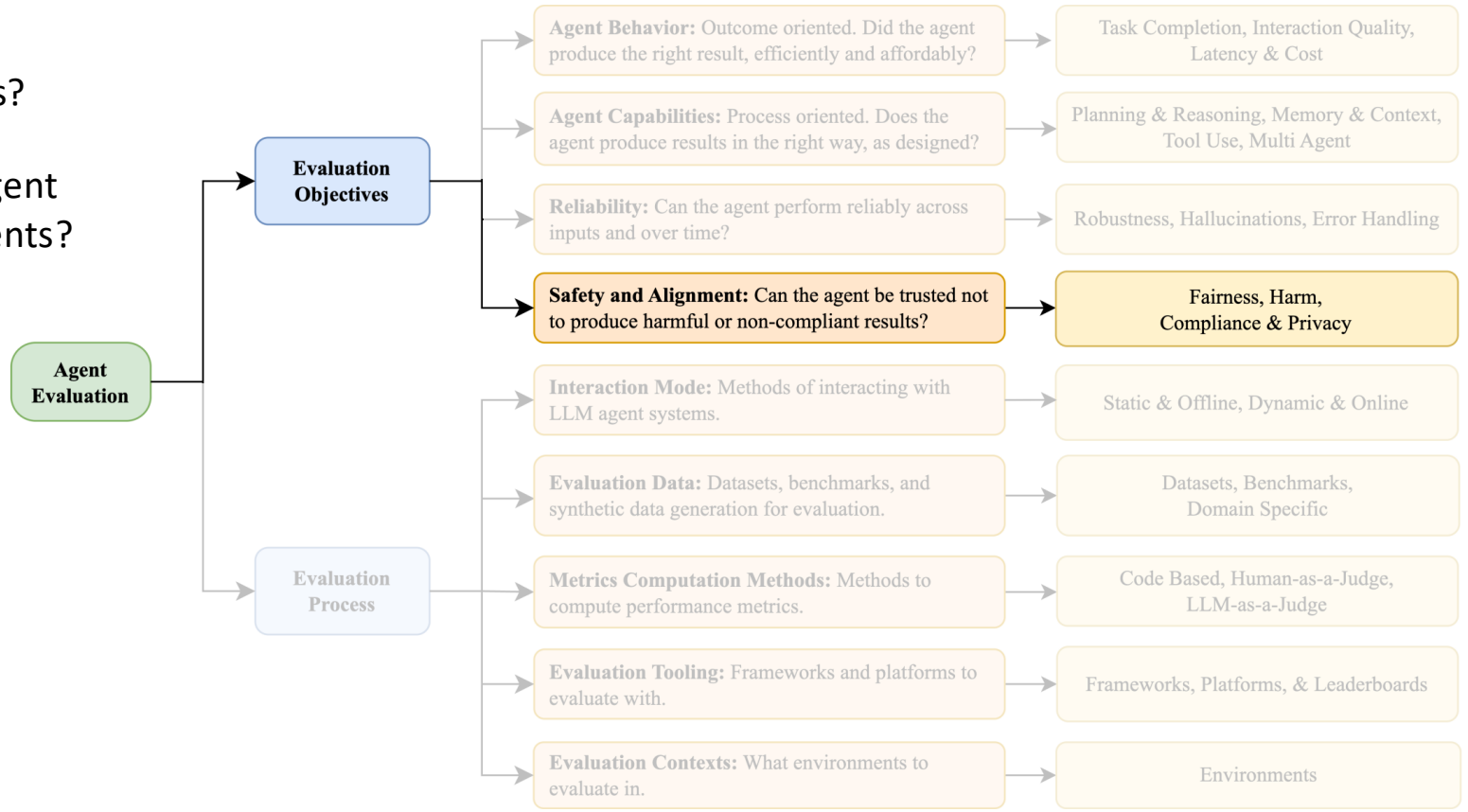
- **Interaction Mode:** Offline (Static Dataset)
- **Evaluation Data:** Purchase Order Dataset
- **Metrics Computation Method:** LLM-as-a-Judge
- **Evaluation Tooling:** Inspect AI
- **Evaluation Contexts:** Mocked APIs

Quick Recap: Evaluation Objectives – Safety & Alignment

Fairness: Are the agent's outputs equitable, explainable and trustworthy?

Harm, Toxicity, & Bias: Does the agent avoid generating offensive or harmful outputs?

Compliance & Policy Adherence: Does the agent respect regulatory and compliance requirements?



Enterprise-Specific Challenges

Enterprise-Specific Challenges: Access Control

Enterprise agents must uphold **role-based access control**, governed by organizational roles (e.g., finance vs. HR users).

Added Complexities

- Output correctness **depends on user identity** & permissions.
- Tasks must be tested under **role-specific constraints**.

  Example: Agent must NOT expose an individual's performance details to coworkers, but SHOULD be able to access them at their manager's request.

Approaches

- Role-aware datasets & conditional test cases
- Policy-injection in prompts

Enterprise-Specific Challenges: Reliability Guarantees

Enterprise agents must perform consistently over time and support **auditing**, **compliance**, and **reproducibility**.

Added Complexities

- **Stochasticity** of LLMs makes consistent behavior hard.
- Enterprise environments often demand **fail-safe systems** under harsher scrutiny.

Approaches

- Multiple-run consistency tests (e.g., pass^k)
- Domain-specific edge case coverage
- Logging & regression suites (AgentOps loop)

Example: τ-benchmark applied to retail and travel agents

Live Code

Scenario 5: Iterative Evaluation Process

Goals

- Track performance over agent iterations

Evaluation Process

- **Interaction Mode:** Offline (Static Dataset)
- **Evaluation Data:** Purchase Order Dataset
- **Metrics Computation Method:** LLM-as-a-Judge
- **Evaluation Tooling:** Inspect AI
- **Evaluation Contexts:** Mocked APIs

Enterprise-Specific Challenges: Dynamic & Long-Horizon Interactions

Enterprise agents operate across extended sessions, evolving goals, and shifting environments.

Added Complexities

- **Continuous operation** over extended periods while interacting with users, systems, and data.
- Enterprise **goals and context** may shift over time.

Approaches

- Long term context storage & retrieval
- Long running simulations & datasets

Examples & Benchmarks:

- **SimTown:** Agents evolve in simulated society
- **LongEval:** 40+ turn memory test
- **Optimus-1:** Tracks memory state over hundreds of interactions

Enterprise-Specific Challenges: Policy & Compliance

Enterprise agents must follow **policies**, respect **legal constraints**, and handle **sensitive data** appropriately.

Added Complexities

- Enterprise data, especially personal info, is typically under strict **legal protection** and **usage constraints**.
- Policies are often **nuanced** and **organization-specific**.

Approaches

- Red-team adversarial prompts (e.g., disguised policy violations)
- Compliance-specific datasets (e.g., CoSafe, R-Judge)
- Explicit refusal checks (e.g., "Sorry, I can’t provide that...")
- Fine tuning models

Examples: Avoid offering prescriptions (healthcare), respect sensitive data boundaries (HR)

Future Directions: Towards Scalable, Realistic Agent Evaluation

Future Directions: Towards Scalable, Realistic Agent Evaluation

Holistic Evaluation Frameworks

- Most current evaluations target **single objectives** (e.g., tool use or behavior).
- Real-world agents must balance **multiple skills simultaneously** (e.g., safe, fast, accurate).
- Need for **multi-dimensional evaluations** integrating behavior, reasoning, and safety.

More Realistic Evaluation Settings

- Move beyond lab-style evaluations to **realistic enterprise environments**.
- Include **dynamic users**, **role-based access**, and **long-horizon workflows**.
- Simulated agents (e.g., in CRM, IT, finance systems) can help approximate production settings.

Scalable & Automated Evaluation Methods

- Manual evaluations are costly and limited.
- Push toward **LLM-as-a-judge**, **agent-as-a-judge**, and **synthetic data generation**.
- Reduce human overhead while preserving insight.

Time- and Cost-Bounded Protocols

- Repeated trials (e.g., pass^k) are expensive.
- Need **efficient evaluation pipelines** that balance depth and runtime.
- Useful for **evaluation-driven development** (EDD) in continuous deployment settings.

Thank you.

Contact information:

Mahmoud Mohammadi
mahmoud.mohammadi@sap.com

Yipeng Li
yipeng.li@sap.com

Jane Lo
jane.lo@sap.com

Wendy Yip
wendy.yip@sap.com

Learn more about Joule Agents in Joule, SAP's AI copilot.

Bring out your best.