

MODEL AI GOVERNANCE FRAMEWORK FOR AGENTIC AI

Version 1.5 | Published 20 May 2026
(Updated 5 June 2026)

Contents

- Executive Summary 3
- What’s new in this version..... 5
- 1 Introduction to Agentic AI..... 6
 - 1.1 What is Agentic AI? 6
 - 1.1.1 Core components of an agent6
 - 1.1.2 Multi-agent setups.....8
 - 1.1.3 How agent design affects the limits and capabilities of each agent8
 - 1.2 Risks of Agentic AI..... 10
 - 1.2.1 Sources of risk..... 10
 - 1.2.2 Types of risk 11
 - 1.2.3 Systemic and multi-agent risks..... 11
- 2 Model AI Governance Framework for Agentic AI..... 13
 - 2.1 Assess and bound the risks upfront..... 15
 - 2.1.1 Determine suitable use cases for agent deployment 15
 - 2.1.2 Bound risks through design by defining agents limits and permissions 19
 - 2.2 Make humans meaningfully accountable25
 - 2.2.1 Clear allocation of responsibilities within and outside the organisation25
 - 2.2.2 Design for meaningful human oversight29
 - 2.3 Implement technical controls and processes33
 - 2.3.1 During design and development, use technical controls.....33
 - 2.3.2 Before deploying, test agents38
 - 2.3.3 When deploying, continuously monitor and test.....42
 - 2.4 Enable end-user responsibility46
 - 2.4.1 Different users, different needs46
 - 2.4.2 Users who interact with agents.....46
 - 2.4.3 Users who integrate agents into their work processes47
- Annex A: Further resources 50
- Annex B: Call for feedback and case studies 52
- Acknowledgements 53

Executive Summary

Agentic AI is the next evolution of AI, holding transformative potential for users and businesses. Compared to generative AI, AI agents can take actions, adapt to new information, and interact with other agents and systems to complete tasks on behalf of humans. While use cases are rapidly evolving, agents are already transforming the workplace through coding assistants, customer service agents, and automating enterprise productivity workflows.

These greater capabilities also bring forth new risks. Agents' access to sensitive data and ability to make changes to their environment, such as updating a customer database or making a payment, are double-edged swords. As we move towards deploying multiple agents with complex interactions, outcomes also become more unpredictable.

Humans must remain accountable and properly manage these risks. While existing governance principles for trusted AI such as transparency, accountability and fairness continue to apply, they need to be translated in practice for agents. Additionally, meaningful human control and oversight need to be integrated into the agentic AI lifecycle. Nevertheless, a balance needs to be struck as continuous human oversight over all agent workflows becomes impractical at scale.

The Model AI Governance Framework (MGF) for Agentic AI gives organisations a structured overview of the risks of agentic AI and emerging best practices in managing these risks. If risks are properly managed, organisations can adopt agentic AI with greater confidence. The MGF is targeted at organisations looking to deploy agentic AI, whether by developing AI agents in-house or using third-party agentic solutions.

Building on our previous model governance frameworks, we have outlined key considerations for organisations in four areas for agentic AI:

1. Assess and bound the risks upfront

Organisations should adapt their internal structures and processes to account for new risks from agents. Key to this is first understanding the risks posed by the agent's actions, which depend on factors such as the scope of actions the agent can take, the reversibility of those actions, and the agent's level of autonomy.

To manage these risks early, organisations could limit their agents' scope of impact by designing appropriate boundaries at the planning stage, such as limiting access to tools and external systems. They could also ensure that the agent's actions are traceable and controllable through measures such as identity management and access controls for agents.

2. Make humans meaningfully accountable

Once the "green light" is given for agentic AI deployment, an organisation should take steps to ensure human accountability.

However, the autonomy of agents may complicate traditional responsibility assignments which are tied to static workflows. Multiple actors may also be involved in different parts of the agent lifecycle, diffusing accountability. It is therefore important to clearly define the responsibilities of different stakeholders, both within the organisation and with external vendors, while emphasising adaptive governance, so that the organisation is set up to quickly understand new developments and update its approach as the technology evolves.

Specifically, "human-in-the-loop" has to be adapted to address automation bias, which has become a bigger concern with increasingly capable agents. This includes

defining significant checkpoints in the agentic workflow that require human approval, such as high-stakes or irreversible actions, and regularly auditing human oversight to check that it remains effective over time.

3. Implement technical controls and processes

Organisations should ensure the safe and reliable operationalisation of AI agents by implementing technical measures across the agent lifecycle.

During development, organisations should incorporate technical controls for new agentic components such as planning, tools and still-maturing protocols, to address increased risks from these new attack surfaces.

Before deployment, organisations should test agents for baseline safety and reliability, including new dimensions such as overall execution accuracy, policy adherence, and tool use. New testing approaches will be needed to evaluate agents.

During and after deployment, as agents interact dynamically with their environment

and not all risks can be anticipated upfront, it is recommended to gradually roll out agents alongside continuous monitoring after deployment.

4. Enable end-user responsibility

Trustworthy deployment of agents does not rely solely on developers, but also on end-users using them responsibly. To enable responsible use, as a baseline, users should be informed of the agent's range of actions, access to data, and the user's own responsibilities. Organisations should consider layering on training to equip employees with the knowledge required to manage human-agent interactions and exercise effective oversight, while maintaining their tradecraft and foundational skills.

This is a living document. We have worked with government agencies and leading companies to collate current best practices and contribute real-world case studies, but this is a fast-developing space. This framework will need to be continuously updated to keep pace with new developments. We invite feedback to refine the framework, and more case studies demonstrating how the framework can be applied for responsible agentic deployment.

What's new in this version

This version incorporates feedback received from 60+ companies since v1.0. The main changes include:

Introduction to Agentic AI	
What is Agentic AI?	<ul style="list-style-type: none"> • Agentic components: Added safety and reliability components (controls, logging and monitoring) as part of agent's core components • Protocols: Updated for newer protocols, especially for agentic commerce
Risks of Agentic AI	<ul style="list-style-type: none"> • Systemic and multi-agent risks: Added systemic and multi-agent risks

Model AI Governance Framework for Agentic AI	
The four dimensions	<ul style="list-style-type: none"> • Case study from IMDA on how the framework can be applied to OpenClaw deployments
Assess and bound the risks	<ul style="list-style-type: none"> • Risk factors: Added new factors for risk assessment, including system complexity and usage of third-party solutions • Case studies from Dayos, MSD, OCBC
Enable meaningful human accountability	<ul style="list-style-type: none"> • Agentic value chain: Refined the agentic AI value chain to separate platform providers and system providers or app developers • Automation bias: Added more practices to guard against automation bias, such as monitoring human override rates and response times • Case studies from PwC, Tencent, XOPA
Implement technical controls and processes	<ul style="list-style-type: none"> • Types of controls: Added overview of different types of controls and how to select them (e.g. structural, rule-based vs prompt-layer controls) • Change management: Included recommendations for change management processes to avoid outsized impact from small changes, especially as system complexity increases • Case studies from CDL x Knovel, CyberSierra, Google, GovTech, Stability Solutions, Terminal 3
Enable end-user responsibility	<ul style="list-style-type: none"> • Tradecraft and business continuity: Added detail on loss of tradecraft and how it can impact business continuity • Case studies from Ant International, Workday

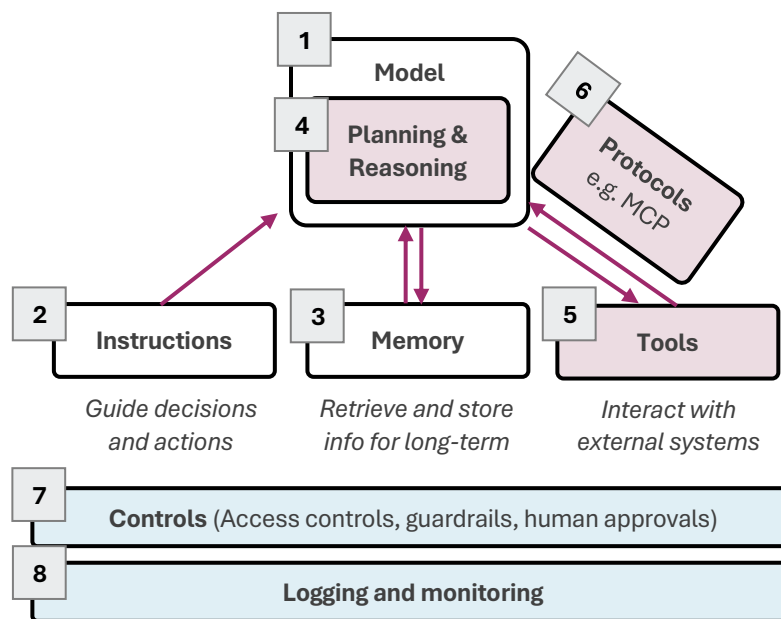
1 Introduction to Agentic AI

1.1 What is Agentic AI?

There is no consensus on what defines an AI agent, but there are certain common features – agents usually possess some degree of independent planning, decision-making, and action-taking (e.g. searching the web or creating files) over multiple steps to achieve a user-defined goal.¹ Agentic AI systems are software systems consisting of one or multiple AI agents that may operate individually or collaboratively.

In this framework, we focus on agents built on generative AI models, which are increasingly being adopted. Generally, such agents use a small, large, or multimodal large language model (SLM, LLM, or MLLM) as its brain to make decisions and complete tasks. However, it is worth noting that software agents are not a new concept and other types of agents exist, such as those which use deterministic rules, or other neural networks, to make decisions.²

1.1.1 Core components of an agent



Core components of a simple agent³

As agents are built on top of language models, it is helpful to start with the core components of a simple LLM-based app, which an agent uses in similar ways.

¹ Adapted from the [International AI Safety Report](#).

² See World Economic Forum (WEF), [AI Agents in Action: Foundations for Evaluation and Governance](#).

³ Adapted from GovTech Singapore, [Agentic Risk & Capability Framework](#), CSA Singapore, [Draft Addendum on Securing Agentic AI](#) and Anthropic, [Building Effective Agents](#).

1. **Model:** an SLM, LLM or MLLM that serves as the central reasoning and planning engine, or the “brain” of the agent. It processes instructions, interprets user inputs, and generates contextually appropriate responses.
2. **Instructions:** Natural language commands that define an agent's role, capabilities, and behavioural constraints e.g. a system prompt for an LLM.
3. **Memory:** Information that is stored and accessible to the LLM, either in short or long-term storage. Sometimes added to allow the model to obtain information from previous user interactions or external knowledge sources.

In addition, an agent has other components that enable it to complete more complex tasks:

4. **Planning and reasoning:** The model is usually trained to reason and plan, meaning that it can output a series of steps needed for a task.
5. **Tools:** Tools enable the agent to take actions and interact with other systems, such as writing to files and databases, controlling devices, or performing transactions. The model calls tools to complete a task. Agents themselves can also be called as tools, e.g. one supervisor agent invokes another specialist agent for a certain task.
6. **Protocols:** Standardised ways for agents to communicate with tools and other agents. For example, the Model Context Protocol (MCP) has been developed for agents to communicate with tools, whereas the Agent2Agent Protocol (A2A) defines a standard for agents to communicate with each other.⁴ This is a fast-developing space and more protocols are being developed to standardise agent interactions, especially in agentic commerce.⁵

Apart from these enabling components, an agent usually also has components for safe and reliable performance:⁶

7. **Controls:** Controls limit the agent’s action-space and autonomy. While there are many types of controls, the ones most relevant to agents are:
 - a. **Access controls:** These limit what an agent is allowed to see, use or change, including restricting access to sensitive data, tools and systems.
 - b. **Guardrails:** Guardrails monitor and constrain an agent’s behaviour before, during or after it acts. It can help detect unsafe instructions, policy violations, or actions that appear inconsistent with user intent.
 - c. **Human approvals:** Requirements for a human to review or approve agent actions.
8. **Logging and monitoring:** Records agent actions, decisions, and interactions across all components to enable monitoring, debugging, and accountability.

⁴ See Anthropic, [Model Context Protocol](#) and Google, [Agent2Agent Protocol](#).

⁵ See OpenAI, [Agentic Commerce Protocol](#), Alipay, [Agentic Mobile Protocol](#), and Google, [Universal Commerce Protocol](#).

⁶ More information on this can be found in [2.3 Implement technical controls and processes](#).

1.1.2 Multi-agent setups

In an agentic system, it is common for multiple agents to be set up to work together. This allows each agent to specialise in a certain function or task and/or work in parallel. Having multiple agents specialising in different tasks also means that each agent's tools and permissions can be separately scoped and defined, compared to a single agent with access to many tools.

Three simple design patterns for multi-agent systems are:⁷

- **Sequential:** Agents work one after another in a linear or otherwise structured workflow e.g. a graph. Each agent's output becomes the next agent's input.
- **Supervisor:** One supervising agent coordinates specialised agents under it, calling specific agents as tools when required.
- **Swarm:** Agents work at the same time, handing off to another agent when needed.

There is no universally correct architecture for a multi-agent setup, and the task at hand can require different or hybrid patterns.⁸ A well-defined task with a step-by-step workflow can lend itself to a sequential architecture, whereas a more open-ended task that requires brainstorming or pursuing different lines of inquiry may benefit from a swarm architecture.

1.1.3 How agent design affects the limits and capabilities of each agent

While each agent may have the same core components, the design of each component can significantly affect what the agent can do.

It is generally helpful to distinguish between two concepts when considering what an agent can do:⁹

Action-space

(or authority, capabilities)

Range of actions the agent can take, including transactions it can execute, determined by the tools it is allowed to use and permissions on those tools

Autonomy

(or decision-making)

Degree to which an agent can decide how to act towards a goal, such as by defining the steps to be taken, determined by its instructions and level of human involvement

⁷ Adapted from AWS, [Multi-Agent Collaboration Patterns with Strands Agents and Amazon Nova](#). See also Claude, [Building multi-agent systems: When and how to use them](#).

⁸ For examples, see AWS, [Inside AWS Security Agent: A multi-agent architecture for automated penetration testing](#), Langchain, [Choosing the Right Multi-Agent Architecture](#).

⁹ See WEF, [AI Agents in Action: Foundations for Evaluation and Governance](#).

Action-space

An agent's action-space mainly depends on the tools it has access to, which can affect:

- **Systems it can access:**
 - *Sandboxes only:* Sandboxed tools (e.g. for code execution, data analysis) that cannot affect any other system
 - *Internal systems:* Tools internal to the organisation, such as being able to search and update the organisation's databases
 - *External systems:* Tools that enable the agent to access external services, such as retrieving and updating data through third-party pre-defined APIs.
- **Actions it can take in relation to the system it can access:**
 - *Read vs write:* An agent may only be able to read and retrieve information from a system, rather than write to and modify data within the system.

An emerging modality of agentic AI is a computer use agent, whose primary tool is access to a computer and browser. This means that it can take any action that a human can take with a computer and browser (e.g. scrolling, clicking, typing) without having to rely on specifically defined tools and APIs. This significantly increases what the agent can access and do.¹⁰

Autonomy

An agent's autonomy mainly depends on its instructions and the level of human involvement in the agentic system. In enterprise deployments, most agentic implementations are hybrid systems combining deterministic rules with some autonomy.

In terms of instructions, an agent can be given differing level of instructions:

- **Detailed instructions and SOP:** An agent instructed to follow a detailed SOP to complete a task would be limited in the decisions it can make at each stage.
- **Using its own judgment:** An agent instructed to use its own judgment to complete a task would have more freedom to define its plan and workflow.

Another relevant factor is the level of human involvement. When interacting with an agent, a human can be involved to different levels:¹¹

- **Agent proposes, human operates:** The human reviews and approves every agent action.
- **Agent and human collaborate:** The human and agent work together. The agent requires human approval at significant steps, such as before writing to a database or making a payment. However, the human can intervene anytime by taking over the agent's work or pausing the agent and requesting a change.
- **Agent operates, human approves:** The agent requires human approval only at critical steps or failures, such as deleting a database or making a payment above a predefined amount.

¹⁰ For more information on testing computer use agents, see the case study [Google x Singapore Government: Testing computer use agents](#) below.

¹¹ See Knight First Amendment Institute at Columbia University, [Levels of Autonomy for AI Agents](#).

- **Agent operates, human observes:** The agent does not require human approval as it completes its task, though its actions may be audited after the fact.

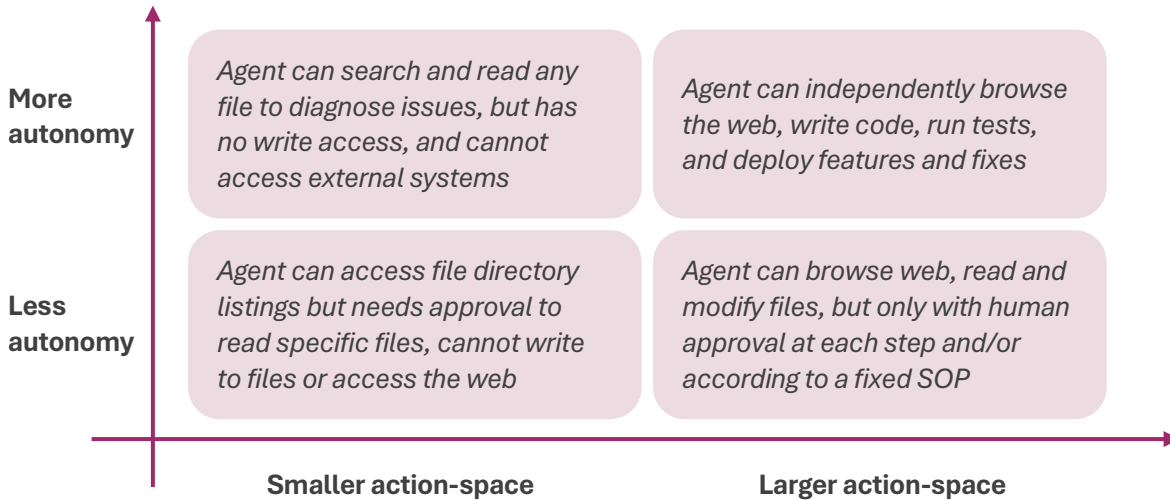


Illustration of differing action-spaces and autonomy for a software engineering agent

1.2 Risks of Agentic AI

1.2.1 Sources of risk

The new components of an agent constitute new sources of risks.¹² The risks themselves are familiar – fundamentally, agents are software systems built on LLMs. They inherit traditional software vulnerabilities (such as SQL injection) and LLM-specific risks (such as hallucination, bias, data leakage and adversarial prompt injections).¹³

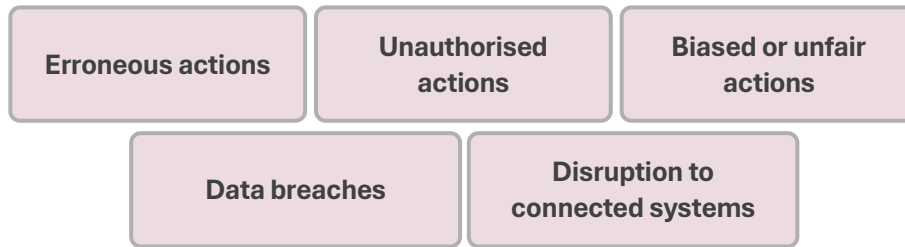
However, the risks can manifest differently through the different components. For example:

- **Planning and reasoning:** An agent can, due to hallucination or semantic misalignment (incorrect interpretation of the user’s intent), create a plan that cannot achieve the requested task, or that contradicts the user’s intent, or drifts from an earlier plan.
- **Tools:** An agent can hallucinate and call non-existent tools, make errors by calling the wrong tools, calling the right tools with the wrong input, or call tools in a biased manner. As tools connect the agent to external systems, prompt or code injections can also manipulate the agent to exfiltrate or otherwise manipulate the data it has access to.
- **Protocols:** As new protocols emerge to handle agent communication, they can be poorly deployed or compromised e.g. deploying an untrusted MCP server that contains code to exfiltrate the user’s data.

¹² See BCG, [What Happens When AI Stops Asking Permission?](#)

¹³ Adapted from CSA, [Draft Addendum on Securing Agentic AI.](#)

1.2.2 Types of risk



Because agents take actions in the real world, when they malfunction, it can lead to harmful real-world impact. Organisations should be aware of these negative outcomes:

- **Erroneous actions:** Incorrect actions such as an agent fixing appointments on the wrong date or producing flawed code. The exact harmful outcome depends on the action in question, e.g. flawed code can lead to exploited security vulnerabilities, and wrong medical appointments may affect a patient’s health outcomes.
- **Unauthorised actions:** Actions taken by the agent outside its permitted scope or authority, such as taking an action without escalating it for human approval based on explicit human instruction, a company policy, or standard operating procedures.
- **Biased or unfair actions:** Biased or unfair output is a known problem for LLMs and can translate to biased actions in agents. These are actions that lead to unfair outcomes, especially when dealing with groups of different profiles and demographics e.g. biased vendor selection in procurement, disbursements of grants, and/or hiring decisions.
- **Data breaches:** Actions that lead to the exposure or manipulation of sensitive data. Such data may be personally identifiable or confidential information e.g. customer details, trade secrets, and/or internal communications. This can be due to a security breach, where attackers exploit agents to reveal private information, or an agent disclosing sensitive data due to a failure to recognise it as sensitive. While this is also a known risk for LLMs, agents generally have greater access to data, and can not only leak but also wrongly modify data.
- **Disruption to connected systems:** As agents interact with other systems, they can cause disruption to connected systems when they are compromised or malfunction e.g. deleting a production codebase, or overwhelming external systems with requests.

1.2.3 Systemic and multi-agent risks

The speed and complexity of agentic systems can increase the risks above. In particular:

- **Speed and volume:** The speed at which agents take decisions make it difficult for oversight mechanisms to detect and prevent unauthorised actions in real-time before they cause harm. Requiring human approvers to continuously oversee agents as a safeguard can also lead to automation bias and alert fatigue.

- **Cascading or compounding effects:** A mistake in one step can propagate and amplify across later steps, resulting in outsized impact. For example, in supply chain management, an initial hallucinated inventory figure could potentially cause downstream actions to reorder excessive or insufficient stock.

Multi-agent systems exacerbate these risks because of the increase in number of interacting agents.

For example, multi-agent systems often require agents to share context, memory, and intermediate outputs with other agents. This increases the likelihood for sensitive data to be unintentionally logged, passed to less secure agents, or exposed through prompt injection attacks.

The extent to which multi-agent systems introduce *qualitatively* different risks is still being studied, but some of the more pertinent ones include:¹⁴

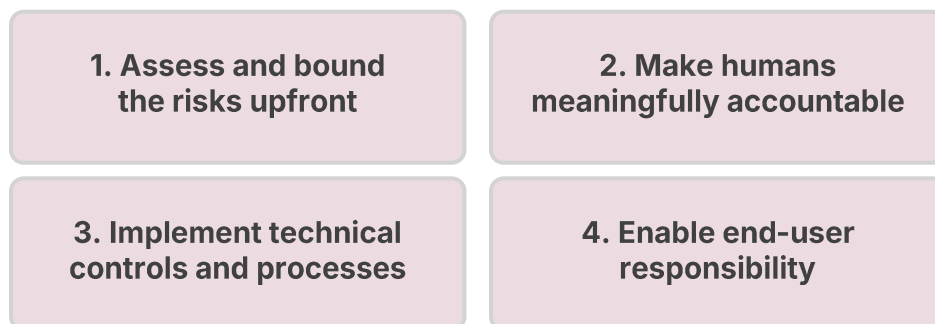
- **Agent sprawl:** As more AI agents are created and deployed within the organisation, this can lead to the uncontrolled proliferation of AI agents without centralised management. This can lead to issues with provenance, incompatibility between old and new agents, and/or difficulties in managing agents from different generations that may not communicate well.
- **Collaborative failures**
 - **Miscoordination:** Agents working together can lead to unintended failures through bad communication or miscoordination. For example, agents working on the same task may interpret the user's intent differently, and work towards different goals.
 - **Conflict:** Agents optimising different goals can come into conflict. For example, a customer support agent may offer refunds to resolve complaints quickly, while a revenue protection agent may block refunds above a certain threshold.
 - **Collusion:** Agents may develop behaviours that appear coordinated, even if no explicit instruction to collude was given. For example, pricing agents used by different organisations could observe each other's prices and converge on higher prices rather than competing. This behaviour has already been studied for pricing algorithms¹⁵ and is being studied for LLM-based agents.
- **Unpredictability and other emergent behaviours:** When multiple non-deterministic agents work together, the number of possible outcomes grows exponentially. This can cause emergent behaviours that cannot be predicted from testing each agent individually.

Finally, multiple agents can interact both within a system or across systems. When agents cross system or organisational boundaries, it becomes more difficult to test for and anticipate the spectrum of potential outcomes, especially if an organisation does not have white-box access to these external systems.

¹⁴ See Gradient Institute, [Risk Analysis Techniques for Governed LLM-based Multi-Agent Systems](#).

¹⁵ See Bichler, Durmann, & Oberlechner, [Algorithmic Pricing and Algorithmic Collusion](#).

2 Model AI Governance Framework for Agentic AI



Four dimensions of the MGF for Agentic AI

The MGF for Agentic AI builds on the responsible AI practices for organisations set out in MGF (2020)¹⁶ by highlighting emerging best practices to address new concerns from agentic AI. This is so that organisations can develop and use agentic AI with the requisite knowledge and judgment.

The framework begins with helping organisations to **assess and bound the risks upfront**. It highlights new risks that should be considered during risk assessment, and design considerations at the planning stage to limit the potential scope of impact of the agents, as well as ensure that agents are traceable and controllable.

While agents may act autonomously, human responsibility continues to apply. Once the “green light” is given to deploy agentic AI, an organisation should take immediate steps to **make humans meaningfully accountable**. This includes clearly defining responsibility across multiple actors within and outside the organisation involved in the agent lifecycle; and taking measures to ensure that human-in-the-loop remains effective over time notwithstanding automation bias.

To ensure safe and reliable operationalisation of agents, an organisation should adopt **technical controls and processes** across the AI lifecycle. During development, guardrails for new components in AI agents such as planning and tools should be implemented. Before deployment, agents should be tested for baseline safety and reliability. After deployment, agents should be continuously monitored as they interact dynamically with their environment.

Finally, trustworthy deployment of agents does not rest solely on developers, but also on end-users. Organisations are responsible for **enabling end-user responsibility** by equipping them with essential information to use agents appropriately and exercise effective oversight, while maintaining their tradecraft and foundational skills.

These four dimensions should be viewed as an iterative process. For example, if an anomaly is detected during implementation or monitoring, organisations should re-evaluate the earlier dimensions, re-assessing and further bounding the risks if needed. This ensures continuous improvement and adaptation as new insights emerge from deployment experience.

¹⁶ See [Model AI Governance Framework \(2nd Ed\)](#).

IMDA: Applying the four dimensions of the framework to OpenClaw

In May 2026, IMDA released a case study on responsible deployment of OpenClaw, applying this framework and drawing from the practical experiences of GovTech, CSA, and industry players such as Grab and Microsoft who had experimented with it.

OpenClaw is an open-source AI agent platform that acts as an autonomous personal assistant through common chat interfaces such as Telegram and Slack. It can automate everyday tasks such as compiling research, handling customer enquiries or debugging code. It was launched with limited security controls and deploying it safely is non-trivial. Concerns include its lack of maturity and hardening, access control and authentication gaps, exposure of sensitive data, supply chain risks from third-party skills, and memory poisoning risks.

The framework can be applied to deploy OpenClaw (and similar) agents responsibly:

1. Assess and bound the risks upfront

- Avoid deploying OpenClaw as-is in mission-critical environments, including systems that handle sensitive data or financial transactions
- Avoid creating a single “all-powerful” agent with unrestricted access, instead use multiple agents with narrow, clearly defined rules
- Avoid installing OpenClaw on primary work or personal devices containing sensitive data, and granting it unrestricted access to files and applications

2. Make humans meaningfully accountable

- Adopt a risk-based approach to determine the appropriate level of agent autonomy based on data sensitivity and task criticality
- Enforce human approval through system-level controls where possible, vs prompt-layer guardrails, which may be bypassed or “forgotten”

3. Implement technical controls and processes

- During design and development, review and tighten OpenClaw configurations which are permissive by default e.g. restrict messaging channel access, using dedicated identities and credentials for the agent
- Before deployment, test and verify that safety controls and human-in-the-loop is working as intended e.g. attempt disallowed actions to ensure restrictions work
- After deployment, ensure all agent actions are logged and attributable, and avoid leaving the agent unsupervised for extended periods

4. Enable end-user responsibility

- Provide personnel training to improve employees’ awareness of autonomous agent risks and reinforce their responsibility to prevent careless misuse

See the full case study [here](#).

2.1 Assess and bound the risks upfront

Agents bring new risks, especially in their access to sensitive data and ability to change their environment through action-taking. Their adaptive, autonomous and multi-step nature also increases the potential for unexpected actions, emergent risks and cascading impacts. Organisations should consider these new dimensions through:

- **Determining suitable use cases for agent deployment** by considering agent-specific factors that can affect the likelihood and impact of the risk.
- **Design choices to bound the risks upfront** by applying limits on agent’s access to tools and systems and defining a robust identity and permissions framework.

2.1.1 Determine suitable use cases for agent deployment

When considering if an agentic use case is suitable for development or deployment, first identify and assess the risk against the benefits. Risk is a function of likelihood (probability of the risk manifesting) and impact (severity of impact if the risk manifests). Not all use cases are suitable for agents, and some may be better served by deterministic workflows.

The following non-exhaustive factors affect the level of risk of an agentic use case:

Factors affecting impact (severity of impact if the risk manifests)		
Factor	Description	<i>Illustration</i>
Domain and use case in which agent is being deployed	Level of tolerance of error in the domain and use case in which the agent is being deployed. Also consider the number and criticality of business processes the agent supports, which affects the impact of agent failure.	<i>Agent executing financial transactions which require a high degree of accuracy would have a larger negative impact if an error were made, vs agent that summarises internal meetings</i>
Agent’s access to sensitive data	Whether the agent can access sensitive data, such as personal information or confidential data. If agent has such access, the risk increases if the agent has persistent memory and can store sensitive data across sessions.	<i>Agent that requires access to personal customer data gives rise to the risk of leaking such data, vs agent who only has access to publicly available information</i>
Agent’s access to external systems	Whether the agent can access external systems.	<i>Agent that sends data to third-party APIs can leak data to these third parties, or disrupt these systems by making too many requests, vs agent that only has</i>

		<i>access to sandboxed or internal tools</i>
Scope of agent's actions	<p>Whether an agent can only read from or can also modify the data and systems it has access to.</p> <p>Whether the agent can complete a small or wide range of actions using tools available to it.</p>	<p><i>Read vs write: Agent that can only read from a database cannot impact the database, vs agent that can write to it</i></p> <p><i>Many actions vs a few: Agent that can only choose from a few pre-defined tools, vs one who has access to a computer use tool that can navigate any user interface</i></p>
Reversibility of agent's actions	<p>If the agent can modify data and systems, whether such modifications are easily reversed. Modifications may not be easily reversed if they trigger downstream obligations e.g. entering into a contract or sale.</p>	<p><i>Agent that schedules meetings can easily reschedule them if an error is made, vs agent that sends email communications to external parties</i></p>

Factors affecting <i>likelihood</i> (probability of the risk manifesting)		
Factor	Description	<i>Illustration</i>
Agent's level of autonomy	<p>Whether the agent can define the entire workflow or must follow a well-defined procedure.</p> <p>A higher level of autonomy can result in higher unpredictability, increasing likelihood of error.</p>	<p><i>Agent is provided with a SOP and instructed to follow it when carrying out a task, vs agent is instructed to use its best judgment to select and execute every step</i></p>
Task complexity	<p>How complex the task is, in relation to the number of steps required to complete it and the level of analysis required at each step.</p> <p>A higher level of task complexity similarly increases unpredictability and the likelihood of error.</p>	<p><i>Agent is required to extract key action points from a meeting transcript, vs agent is tasked to follow a nuanced data sharing policy when handling external requests for information</i></p>
Agent's access to external systems	<p>Whether the agent is exposed to external systems, and who maintains these systems.</p> <p>A higher level of exposure makes the agent more vulnerable to prompt injections and cyberattacks.</p>	<p><i>Agent who can only access an internal knowledge base which is maintained by trusted internal teams, vs an agent who can access the web containing untrusted data</i></p>

<p>Agent being provided or operated by an external party</p>	<p>Whether the organisation is using agents provided or operated by external parties.</p> <p>When using third-party solutions, organisations should consider to what extent their visibility and control over the agent is limited.</p>	<p><i>Agent developed and maintained internally with full visibility vs agent provided by third-party vendor with limited transparency into its operations and data processing</i></p>
<p>System complexity</p>	<p>How complex the agentic system is, e.g. using multiple agents with autonomous decision-making, feedback loops.</p> <p>A system can also become complex due to a combination of different factors above (e.g. complex tasks with greater autonomy). A higher level of system complexity can result in more unpredictable and emergent behaviour as components interact in unexpected ways, compounding any negative effects.</p>	<p><i>A single agent carrying out a sequential workflow vs multiple agents which can interact with each other, make decisions collectively, or autonomously handoff to other agents</i></p>

Threat modelling also makes risk assessment more rigorous by systematically identifying specific ways in which an attacker may take to compromise the system. Common security threats to agentic systems include memory poisoning, tool misuse, and privilege compromise.¹⁷ As agentic systems can become very complex, it is often useful to use a method called taint tracing to map out all the workflows and interactions to track how untrusted data can move through the system. For more information on how to perform threat modelling and taint tracing for agentic systems, organisations may refer to [CSA’s Draft Addendum on Securing Agentic AI](#).

The relationship between threat modelling and risk assessment

Threat modelling augments the risk assessment process by generating contextualised threat events with well-described sequence of actions, activities and scenarios that the attacker may take to compromise the system. With more relevant threat events, risk assessments will be more rigorous and robust, resulting in more targeted controls and effective layered defence. Since risk assessment is continuous, the threat model should be regularly updated.

Adapted from [CSA, Guide to Cyber Threat Modelling](#)

¹⁷ For a more comprehensive coverage of potential security threats to agentic AI systems, see OWASP, [Agentic AI – Threats and Mitigations](#).

Dayos: Tiering and bounding agentic actions by risk level in IT management

Dayos is an enterprise AI automation company headquartered in Singapore with operations in the US. The company builds Hero, an agentic platform that sits on top of Oracle, SAP, Workday, NetSuite and Microsoft Dynamics and automates workflows across IT management, accounting, HR and procurement.

Dayos replaced its own ServiceNow instance with an AI-powered ticketing agent built on Hero. The full retirement took 45 days and reduced legacy licensing costs by \$121,000 annually. The agent now handles every internal IT request that comes in: it reads the ticket, determines how urgent and complex it is, and either resolves it automatically or routes it to a human.

Before any of this went live, Dayos ran a structured risk assessment. Every type of IT ticket was scored against three questions:

- **Severity of impact:** If the agent gets this wrong, how bad is it?
- **Reversibility:** Can the action be undone?
- **Feasibility of human oversight:** Is it realistic for a human to review this at each step?

The scores determined which tier each ticket type falls into, and that tier dictated what reasoning strategy the agent uses and how much autonomy the agent gets.

Tier	What falls here	What the agent does (and does not do)
Tier 1 (60% of tickets) Low severity, fully reversible	Password resets, access requests, status inquiries	<ul style="list-style-type: none"> • Fully automated by agent using Simple Feedback (a propose-confirm loop). The agent proposes, the user confirms or pushes back, and agent adjusts until the ticket is closed. No human engineer is in the loop. • But with biweekly audits: Every action produces a reasoning chain that captures classification logic and a confidence score. A designated reviewer audits a cross-section of these biweekly. As every Tier 1 action is reversible, errors can be corrected without any lasting damage.
Tier 2 (30% of tickets) Moderate severity, partially reversible	Chart of accounts updates, integration mapping corrections, diagnosing failed API connections	<ul style="list-style-type: none"> • Agent diagnoses but can only act with a human engineer’s approval. Uses the ReACT strategy (a multi-step diagnostic loop for harder problems): the agent checks logs, queries connected systems, identifies the probable root cause, and writes up a diagnostic summary with a proposed fix. But a qualified engineer has to review and sign off before anything executes.
Tier 3 (10% of tickets) High severity, limited reversibility	Production deployments, security changes, permission modifications	<ul style="list-style-type: none"> • Agent does not touch these. Production deployments and security changes carry too much risk for autonomous execution at the current maturity of agentic AI. As safeguards improve and are validated in real environments – e.g. multi-agent verification and real-time anomaly detection – Dayos will reassess whether bounded automation makes sense for specific Tier 3 actions.

2.1.2 Bound risks through design by defining agents limits and permissions

Having selected an appropriate agent use case, organisations can further bound the risks by defining appropriate limits and permission policies for each agent.

Agent limits

Organisations should consider defining limits on:

- **Agent’s access to tools and systems:** Define least-privilege policies that give agents only the minimum tools and data access needed for it to complete its task.¹⁸ For example, a coding assistant may not require access to a broad web search tool, especially if it already has curated access to the latest software documentation. Structuring agents around functional boundaries (e.g. separating IT helpdesk from HR self-service) can act as a natural constraint.
- **Agent’s autonomy:** For process-driven tasks, SOPs and protocols are frequently used to improve consistency and reduce unpredictability.¹⁹ Define similar SOPs for agentic workflows that an agent is constrained to follow, rather than giving the agent the freedom to define every step of the workflow.
- **Agent’s area of impact:** Design mechanisms and procedures to take agents offline and limit their **potential scope of impact** when they malfunction. This can include running agents in self-contained environments with limited network and data access, particularly when they are carrying out high-risk tasks such as code execution.²⁰

In general, prefer deterministic rather than non-deterministic limits, and bound by design. For example, rather than relying on prompts to instruct the agent against accessing certain tools, impose access controls that prevent the tool from being called by the agent at all. This is elaborated in [2.3.1 Implement technical controls and processes](#). Where limits are non-deterministic or less reliable, layer on more monitoring measures or incorporate human-in-the-loop review to catch any failures.

¹⁸ See PwC, [The rise – and risks – of agentic AI](#).

¹⁹ Grab introduced an LLM agent framework leveraging on Standard Operating Procedures (SOPs) to guide AI-driven execution (see [Introducing the SOP-driven LLM agent frameworks](#)).

²⁰ See McKinsey, [Deploying agentic AI with safety and security: A playbook for technology leaders](#).

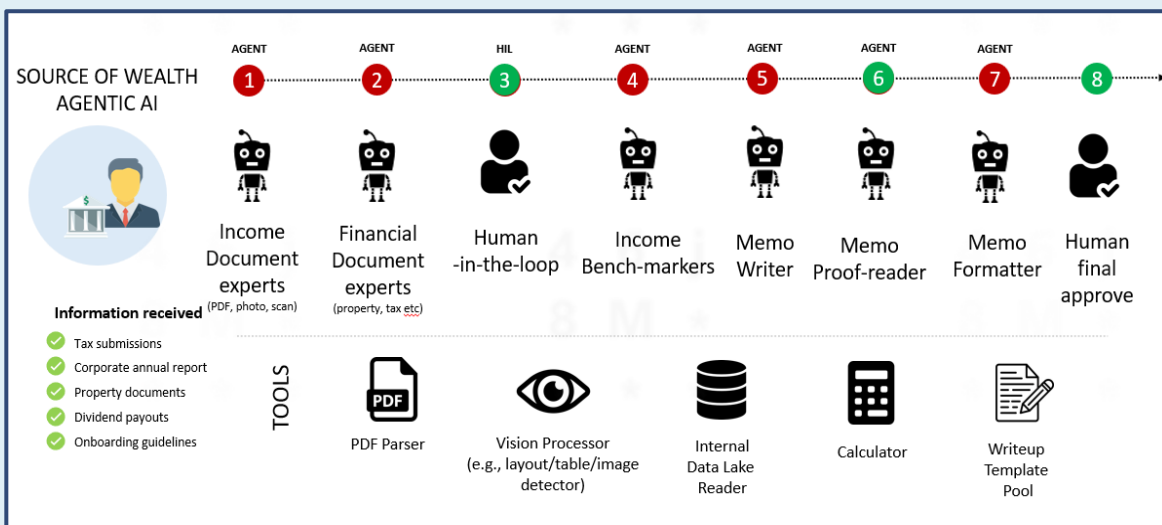
OCBC: Bounding agent autonomy in source of wealth analysis

OCBC is the longest established Singapore bank and the second largest financial services group in Southeast Asia by assets, operating in 19 countries and regions. Bank of Singapore is a wholly owned private banking subsidiary of OCBC.

Bank of Singapore Relationship Manager and compliance team synthesise customer-provided financial documents into a structured Source of Wealth memo. The bank developed an agentic AI system to support this work with the objectives of:

- Improving consistency and completeness of the analysis
- Reducing manual effort in document review and drafting

The agentic AI system parses income-related documents (e.g. tax submissions, corporate annual reports, property documents, dividend statements) and generates a draft Source of Wealth memo based on internal guidelines and benchmarks. The system provides decision support only and does not make credit, onboarding or risk decisions autonomously – final validation and approval remain with designated human reviewers.



The agentic workflow bounds risk through design by defining limits on the agents' autonomy:

- **Task-level autonomy only:** Agents perform narrowly scoped tasks (extraction, drafting, checking), not end-to-end decisions
- **No self-initiation of actions:** Agents operate only when triggered by predefined workflows
- **No decision authority:** There is human review at critical decision points, including after the income and financial documents have been assessed to ensure accuracy. Ultimately, outputs are advisory and subject to human validation.

MSD: Bounding the risks of agentic AI through tiered autonomy and third-party ecosystem boundaries

MSD aims to use the power of leading-edge science to save and improve lives around the world, bringing forward medicines, vaccines and innovative health solutions for the world’s most challenging diseases.

MSD develops and deploys agents in use cases such as streamlining the production of key trial protocols and minimizing labor-intensive tasks like clinical documentation and quality review. For its 75,000 employees, MSD implemented an enterprise agentic AI governance strategy that bounds risk by, among other things, calibrating governance processes to level of agency, and derisking third-party platforms.

Calibrating governance processes to level of agency

MSD created multiple risk-calibrated governance pathways based on five levels of agency, accounting for factors such as action and permission scope, human oversight, data and context scope etc.

Agentic System Level Matrix

		Levels					
		L0	L1	L2	L3	L4	L5
		No Agency	Assisted Agency	Supervised Agency	Conditional Agency	Advanced Agency	Full Agency
Dimensions	Action Scope	N/A	Limited	Moderate	Moderate	Extensive	Extensive
	Permission Scope	N/A	Read-Only	Controlled Write	Controlled Write	Full Access	Full Access
	Autonomy	N/A	Reactive	Proactive	Proactive	Self-Directed	Self-Directed
	Goal Complexity	N/A	Simple	Simple	Compound	Compound	Dynamic
	Human Oversight	N/A	Continuous	Periodic	Exception-Only	Exception-Only	Exception-Only
	Risk Impact	N/A	Low	Medium	Medium	High	High
	Data & Context Scope	N/A	Task-Specific	Task-Specific	Cross-Application	Organizational	Organizational

- Lower levels move quickly through a lightweight governance pathway; middle levels move through an established AI impact assessment process, and higher levels require escalations to enterprise architecture review and testing where appropriate.
- The highest levels of agency require technical measures beyond monitoring. A programmatic runtime policy enforcement layer is being implemented at the AI gateway before higher levels of autonomy are enabled.

Derisking third-party agent platforms

Beyond agentic autonomy, enterprise AI invariably includes multiple third-party SaaS agentic platforms. While each third-party agent platform would have performed validation and testing within its own ecosystem, risks increase when these systems attempt to perform cross platform actions which may not have been exhaustively tested. This is particularly the case as interoperability frameworks are still nascent and maturing.

To promote safety and security, MSD adopted a containment strategy — rather than allowing vendor-embedded agents to act freely across MSD’s data and systems, agentic features in SaaS tools are by default restricted to within their own ecosystems. This contains residual risk while preserving the productivity benefits of approved use cases.

<p>MSD Platform-based Agents</p> <p>Agents developed and deployed on MSD-sanctioned platforms where governance, security, and orchestration are centrally managed.</p>	<p>Do's:</p> <ul style="list-style-type: none"> ✔ Use internal company agentic AI platform for any agent requiring access across ecosystems or tools ✔ Register all agents in the Enterprise Agent Registry ✔ Evaluate agent based on Agentic AI Levels ✔ Complete AI Impact Assessment and assign agentic risk level 	<p>Don'ts</p> <ul style="list-style-type: none"> ✘ Do not bypass governance processes (e.g., registration, risk classification) ✘ Do not allow agents to self-orchestrate or escalate privileges ✘ Do not deploy agents L2 and above without formal approval
<p>Vendor Native Agents</p> <p>Agents embedded within third-party SaaS platforms (e.g., SAP, Salesforce, Workday) and operating exclusively within the vendor's ecosystem.</p>	<p>Do's:</p> <ul style="list-style-type: none"> ✔ Use only within the vendor's native environment ✔ Ensure vendor agent is registered in the Agent Registry ✔ Evaluate agent based on Agentic AI Levels ✔ Conduct Vendor Risk Assessment if it hasn't been done before ✔ Complete AI Impact Assessment and assign agentic risk level ✔ Validate vendor compliance with ethical AI principles (e.g., fairness, transparency, oversight) 	<p>Don'ts</p> <ul style="list-style-type: none"> ✘ Do not allow the agent to access or act on data outside the vendor ecosystem ✘ Do not orchestrate cross-platform actions unless routed through internal company agentic AI platform ✘ Do not assume vendor claims of "agentic" behaviour are aligned with MSD definitions—validate first
<p>Custom Agents</p> <p>Agents developed using open-source libraries or frameworks (e.g., LangGraph, Autogen, LangChain), typically outside of sanctioned platforms.</p> <p>This category of agents should be only used in rare cases requiring unique use case and approval to leverage this approach.</p>	<p>Do's:</p> <ul style="list-style-type: none"> ✔ Use only in sandbox or development environments unless explicitly approved ✔ Seek approval for the design and purpose of the exploration and experimentation from the AI Design Board (before development) ✔ Use of this category of Agents must be explicitly vetted before development 	<p>Don'ts</p> <ul style="list-style-type: none"> ✘ Do not develop without formal AI Design Board approval ✘ Do not deploy to any environments without AI Design Board approval ✘ Do not use for sensitive workflows

Agent identity

Identity management and access control is one of the key means in which organisations enable traceability and accountability today for humans. As organisations deploy more agents, including agents which interact across organisation boundaries, identity management needs to be extended to agents to track individual agent behaviour and establish who holds accountability for each agent.

This is an evolving space, and gaps exist today in terms of handling agent identity robustly. For example, current authorisation systems typically have pre-defined, static scopes. However, to operate safely in more complex scenarios, agents require fine-grained permissions that may change dynamically depending on the context, risk levels, and task objectives. Current authentication systems are also typically based on a single, unique individual. Such systems face difficulty in handling complex agent setups, such as when agents act for multiple human users with different permissions, or recursive delegation scenarios where agents spin up multiple sub-agents.²¹

Solutions are being developed to address these issues, such as intra-organisation identity systems being extended to agents,²² or integrating well-established standards like OAuth 2.1 into MCP.²³ The industry is also developing new standards and solutions for agents, such as decentralised identity management and dynamic access control.²⁴

In the interim, organisations should consider these best practices:

- **Identification:** An agent's identity should be:
 - **Unique:** An agent should have its own unique, cryptographically verifiable identity, such that it can identify itself to the organisation, its human user, or other agents.
 - **Accounted for:** This identity should be tied to a supervising agent, a human user, or an organisational department for accountability and tracking.
 - **Differentiated according to the capacity in which it acts:** The different capacities in which an agent acts (e.g. independently or on behalf of a specified human user) should be recorded for auditability.
 - **Catalogued and centrally managed:** To prevent agent sprawl, all agent identities (and their attendant permissions) should be issued from and tracked by a centralised system. This enables an organisation to track its deployed agents, identify any anomalies, and remove identities that are no longer required.

²¹ For a more comprehensive treatment of how current identity systems may face challenges when catering to agentic AI, see OpenID, [Identity Management for Agentic AI](#).

²² For examples, see Microsoft, [What is Microsoft Entra Agent ID](#), and Alibaba Cloud, [Agent ID Guard overview](#).

²³ See MCP, [Authorisation support](#).

²⁴ See proposed framework for agentic identity by Cloud Security Alliance, [Agentic AI Identity & Access Management: A New Approach](#).

- **Authorisation:** An agent's authority should be:
 - **Scoped, least-privilege, non-transferable:** Authorisations should generally be scoped, time- or session-bound, non-transferable, and follow the principle of least privilege by default, with explicit escalation paths for elevated permissions.
 - **Bounded by authorising human's permissions:** An agent can have pre-defined permissions based on its role or the task, or its permissions may be dynamically set by its authorising human user, or a combination of both. As a rule of thumb, the human user should not be able to set permissions for the agent greater than what the human user is himself authorised to do, such as accessing data outside the human's permissions. Such delegations of authority should be clearly recorded.

Evaluating the residual risks

Residual risk is the risk that remains after mitigation measures have been applied. It is important to note that there will always be a level of risk remaining, even after efforts are taken to identify appropriate agentic use cases and define limits on any agents, especially given how quickly agentic AI is evolving. Ultimately, organisations should evaluate and determine if the residual risk for their agentic deployment is of a tolerable level and can be accepted.

2.2 Make humans meaningfully accountable

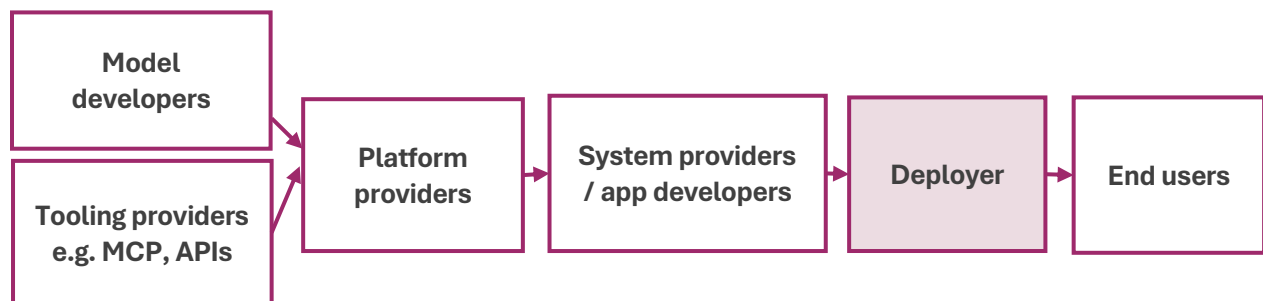
The organisations that deploy agents and the humans who oversee them remain accountable for the agents' actions. But it can be challenging to fulfil this accountability when agent actions emerge dynamically and adaptively from interactions instead of fixed logic. Multiple stakeholders may also be involved in different parts of the agent lifecycle, diffusing accountability. Finally, automation bias, or the tendency to over-trust an automated system, especially when it has performed reliably in the past, becomes a bigger concern as humans supervise increasingly capable agents.

To address these challenges to human accountability, organisations should consider:

- **Clear allocation of responsibilities within and outside the organisation**, by establishing chains of accountability across the agent value chain and lifecycle, while emphasising adaptive governance, so that the organisation is set up to quickly understand new developments and update their approach as the technology evolves.
- **Measures to enable meaningful human oversight of agents**, such as requiring human approval at significant checkpoints, auditing the effectiveness of human approvals, and complementing these measures with automated monitoring.

2.2.1 Clear allocation of responsibilities within and outside the organisation

As deployers, organisations and humans remain accountable for the decisions and actions of agents. However, as with AI, the value chain for agentic AI involves multiple actors. Organisations should consider the allocation of responsibility both within their organisation, and vis-à-vis other organisations along the value chain.



Simplified agentic AI value chain²⁵

Organisations may play multiple overlapping roles across this value chain. For example, an organisation that develops its own agents and subsequently deploys them would play both the role of the system provider and the deployer.

²⁵ For a more comprehensive list of potential stakeholders involved in the agentic AI ecosystem, see CSA and FAR.AI, [Securing Agentic AI: A Discussion Paper](#).

Within the organisation

Within the organisation, organisations should allocate responsibilities for different teams across the agent lifecycle. While each organisation is structured differently, this is an illustration of how such responsibilities may be allocated across different teams:

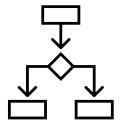


Key decision makers

Who: Leaders who define strategic decisions and high-level policies for the organisation e.g. board members, C-suite executives, managing directors, or department leaders.

Key responsibilities can include:

- Setting high-level goals for use of agents
- Defining permitted operational use cases for agents, including limits on agent’s data access
- Setting the overall governance approach, including risk management frameworks and escalation processes



Product teams

Who: These roles oversee the translation of stakeholder needs or business goals into a technical agentic solution e.g. Product Managers, UI / UX Designers, AI Engineers, Software Engineers

Key responsibilities can include:

- Defining the design and requirements for agents, as well as any feature controls or phased rollouts
- Reliable implementation of agents i.e. development, pre-deployment testing and post-deployment monitoring across the agent lifecycle
- Educating users on responsible use of agentic product



Cybersecurity teams

Who: These roles oversee the protection of agentic systems from cyber threats, by implementing and managing security measures, identifying vulnerabilities, and responding to incidents e.g. Chief Security Officer, Cyber Security Specialist, Penetration Tester

Key responsibilities can include:

- Defining baseline security guardrails and secure-by-design templates that technical teams should implement or adapt to the agentic system being deployed
- Conducting regular red teaming and threat modelling



Users

Who: Any individual who utilises the output of the agents to contribute to an organisational goal e.g. company employees making decisions or automating workflows and practices.

Key responsibilities can include:

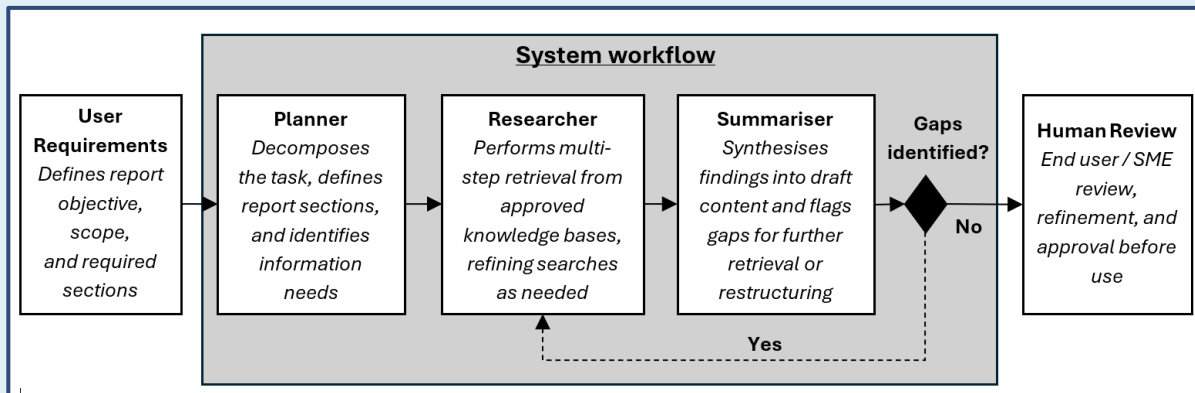
- Ethical and responsible usage of agents
- Attending required training, complying with usage policies, timely reporting of bugs or issues with agents

PwC Singapore: Allocating human accountability across different internal teams for agentic system for report drafting

PwC is a network of professional services firms in 137 territories, across audit and assurance, tax and legal, deals and consulting.

To accelerate internal productivity, PwC Singapore’s AI Factory developed an agentic application that automates the drafting of research and analysis sections for internal reports, a process that has been traditionally manual and research-intensive. Users define the sections required, then the system draws from standardised templates maintained and updated (for regulatory and market developments) by PwC, retrieves and synthesises information from source materials, and generates draft content in seconds.

The system uses three coordinated agents in a primarily sequential workflow, with feedback loops for additional retrieval or restructuring where needed:



The responsibilities below describe how operating accountability for this report-drafting use case is allocated, within the context of PwC Singapore’s broader AI governance framework:

1. **Use case owner – Use case appropriateness and accountable use:** Confirming that the use case serves a legitimate business need, is appropriate for the intended purpose, and is aligned with the organisation’s AI governance framework. The use case owner remains accountable for the responsible use of the application and for ensuring that appropriate human oversight is in place.
2. **Technology Risk Management Team – Risk assessment and control advice:** Supports the assessment of technology, risk, security, and compliance considerations for the use case, including any regulatory requirements. This includes advising on appropriate controls relating to tools and models, access rights, and connected knowledge bases or data sources.
3. **AI Factory – Design, development, guardrails and monitoring:** Designing, developing, and maintaining the application, including prompt design, model selection, tool configuration, integration with approved knowledge bases, and appropriate guardrails. AI Factory also monitors for model changes and manages updates through standard change management processes rather than leaving it to the end users.

4. **End users / subject matter specialists – Output review and approval:** Reviewing AI-generated content before it is used. Internal guidance makes clear that AI-generated content is a first draft intended to support, and not replace, professional judgement. Human reviewers refine and approve outputs before reliance or further use, including checking accuracy, completeness, and alignment with established methodologies.

Developing internal capabilities for adaptive governance

All teams involved in agentic AI should also develop internal capabilities to understand the technology. Being aware of the improvements and limitations of agentic developments, such as new modalities like computer use agents, or new evaluation frameworks, allow organisations to quickly adapt their governance approach to new developments.

Outside the organisation

Organisations may also need to work with external parties when deploying agents e.g. model developers, agentic AI providers, or hosts of external MCP servers or tools.

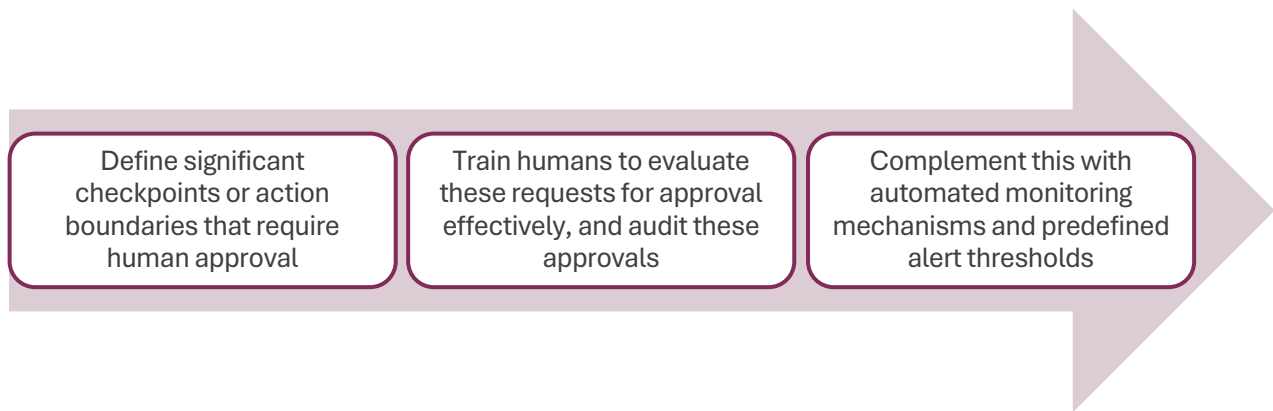
In these cases, organisations should similarly ensure that there are measures in place to fulfil its own accountability. Some agent-specific considerations are:

- **Clarify distribution of obligations** in any terms and conditions or contracts between the organisation and the external party. In particular, organisations should consider provisions on security arrangements, performance guarantees, or data protection. Where there are gaps, the organisation should reassess if the agentic deployment meets its risk tolerance.
- **Address third-party opacity:** Agents provided by external parties can be difficult to observe and control, making it challenging to understand what information they are grounded on, what they infer from conversations, or how data is stored and used. Organisations should take steps to maintain sufficient security and control, such as:
 - **Require transparency and accountability from external parties** e.g. through disclosures on agent capabilities and data handling practices
 - **Request and evaluate technical security and control features** e.g. strong authentication measures such as scoped API keys, per-agent identity tokens, and observability such as the logging of tool calls and access history. Where such features are lacking, organisations should consider alternative or in-house solutions, or scoping down the agentic use case, such as restricting access to sensitive data.

End users

Organisations may deploy agents to users within or outside their organisation. In doing so, organisations should ensure that users are provided sufficient information to hold the organisation accountable, as well as any information relating to the user's own responsibilities. More information can be found in [Enabling end-user responsibility](#) below.

2.2.2 Design for meaningful human oversight



Setting up a system for effective human supervision

Organisations should define significant checkpoints or action boundaries that require human approval, especially before sensitive actions are executed. This can include:²⁶

- **High-stakes actions and decisions** e.g. editing of sensitive data, final decisions in high-risk domains (such as healthcare or legal), actions that may trigger liability
- **Irreversible actions** e.g. permanently deleting data, sending communications, making payments
- **Outlier or atypical behaviour** e.g. when agent accesses a system or database outside of its work scope, when agent selects a delivery route that is twice as long as the median distance.
- **User-defined** e.g. agents may act on behalf of users who have different risk appetites. Beyond organisation-defined boundaries, users may be given the option to define their own boundaries e.g. requiring approval for purchases above a certain amount.

Apart from considering when approvals are required, organisations should also consider what form approvals should take. These considerations include:

- **Keep approval requests contextual and digestible, while making the risk clear.** When asking humans for approval, keep the request short and clear, instead of providing long logs or raw data that may be challenging to decipher. However, useful additional data, such as the associated risk with the action or a confidence score, should be included.
- **Consider the form of human input required.** For straightforward actions such as accessing a database, the human user can simply approve or reject. For more complex cases, such as reviewing an agent's plan before execution, it may be more productive for the human to edit the plan before giving the agent the go-ahead. For high-risk actions, human users can be required to provide an additional written justification before approving or rejecting.

Organisations should implement measures to ensure continued effectiveness of human oversight, particularly as humans remain susceptible to alert fatigue and automation bias and may be influenced by anthropomorphic design. These measures can include:

²⁶ For further examples of where human involvement may be considered, see Partnership on AI, [Prioritising real-time failure detection in AI agents](#).

- **Regularly auditing the effectiveness of human oversight.** Some ways to do so include:
 - Tracking measurable indicators
 - Human override rate i.e. the frequency at which humans reject or modify agent actions. A low rate may signal rubber-stamping behaviours.
 - Human response times during review of agent actions. A shorter time may signal automation bias or review fatigue.
 - Using data analytics to identify "outlier" humans, whose decision patterns deviate significantly from the norm. This may indicate compromised oversight.
- **Training human overseers to identify common failure modes or agent limitations** e.g. inconsistent agent reasoning, agents referring to outdated policies. It should also be noted that chain-of-thought reasoning, which is sometimes used for explainability, is not analogous to human reasoning and may not be a faithful explanation of the agent's actions.²⁷
- **Ensuring that humans possess the domain expertise to evaluate agent actions** e.g. users using agents to "vibe code" may not have the software engineering expertise to review the robustness of generated code.

Finally, human oversight should be complemented with automated real-time monitoring to escalate any unexpected or anomalous behaviour. This can be done by:

- Implementing alerts for certain logged events (e.g. attempted unauthorised access or multiple failed attempts to call a tool)
- Using data science techniques to identify anomalous agent trajectories
- Using agents to monitor other agents
- Denying action by default when approval infrastructures fail (e.g. when human supervisors are unreachable or agents attempt new actions that do not have any established approval policies in place).

For more information, see [Continuous testing and monitoring](#) below.

Tencent: Enabling meaningful human oversight in coding assistants

Tencent is a multinational technology company headquartered in China, responsible for gaming, messaging app WeChat and AI models such as Hy.

CodeBuddy is an agentic AI coding system developed by Tencent Cloud and used by its engineers. It can autonomously plan, write, test and deploy code through natural language instructions, with access to filesystems, terminal commands, external APIs and MCP tools.

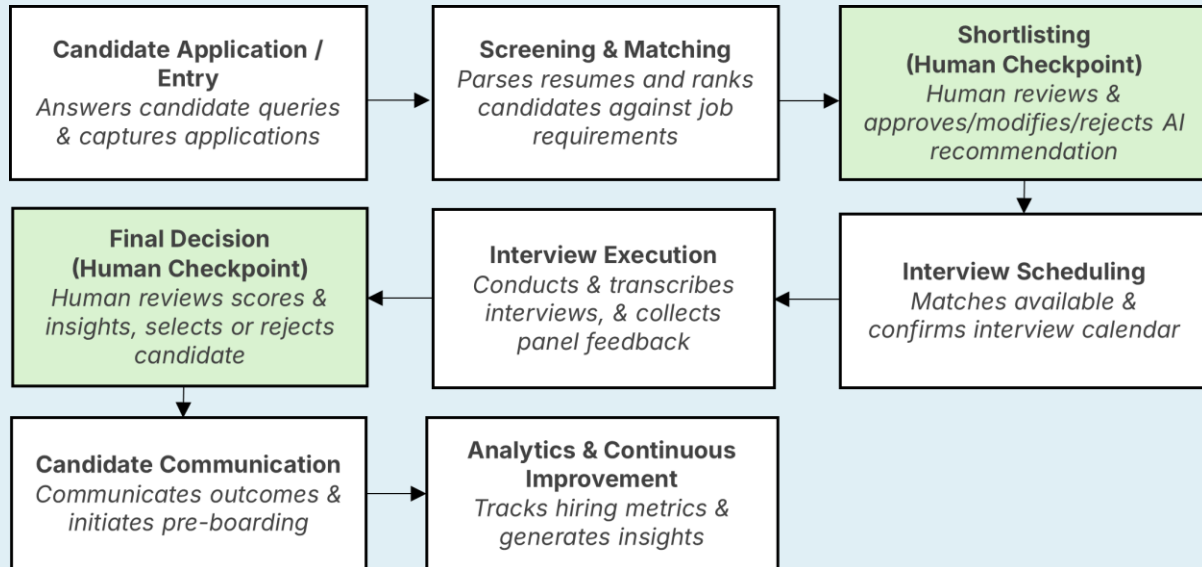
CodeBuddy employs a mix of preset secure defaults and configurable permissions to allow for meaningful human oversight without overly fatiguing the user:

²⁷ See Anthropic, [Reasoning models don't always say what they think](#).

<p>Defining significant checkpoints for human intervention</p>	<p>Higher-stakes or irreversible actions: Default human approval requirements correspond to the risk level of each action:</p> <ul style="list-style-type: none"> • No approval required: <ul style="list-style-type: none"> ○ Read (reading files, directory listings) • Approval required: <ul style="list-style-type: none"> ○ Edit (modifying files): Approval required, valid only for that session. Future sessions require new approvals ○ Bash (shell command execution): Approval required, permanent per project or command ○ WebFetch (network requests): Approval required ○ MCP (external tool invocation): Governed through allow, ask, and deny rules; additional first-use control via trust verification for a newly connected MCP server <p>Self-defined: To mitigate alert fatigue, the default permission level for each project can be set by each user or organisation, calibrated to a project’s context, potential impact, and risk tolerance. For instance:</p> <ul style="list-style-type: none"> • Lower-risk projects, such as internal documentation sites or prototype repositories with no production credentials, may justify more permissive settings for routine file edits. • Higher-risk projects, such as those with access to secrets, sensitive data, production infrastructure, deployment pipelines, or external tools, warrant stricter approval requirements.
<p>Enabling humans to effectively evaluate requests for approval</p>	<p>Explaining proposed actions: When complex bash commands are proposed, plain language explanations help users understand what they are approving, including surfacing any side effects.</p> <p>For instance:</p> <ul style="list-style-type: none"> • Proposed command: <code>mysqldump -u root -p my_database gzip > /backups/my_database_\$(date +%Y%m%d).sql.gz</code> • Explanation: I'm going to create a full backup of your database, compress it on the fly to save space, and save it to /backups/ with today's date in the filename. You will be prompted for the database root password. The original database will not be modified — this is a read-only operation.
<p>Complementing with automated monitoring mechanisms</p>	<p>Continuous real-time monitoring for command injections: When a suspicious command is identified, human approval is required even where the command has previously been whitelisted. For instance:</p> <ul style="list-style-type: none"> • A developer may previously have whitelisted a command pattern for routine tasks such as fetching documentation, such as “<code>curl https://example.com/api-docs</code>”. • If a later command using a similar pattern appears suspicious or materially riskier, such as “<code>curl https://example.com/api-docs && echo test > /tmp/unexpected_file</code>”, CodeBuddy’s protections can require fresh human approval.

XOPA: Enabling effective human approvals in high-impact recruitment workflows

XOPA is a B2B software company that provides AI-powered recruitment solutions. Its AI Agentic Platform applies agentic AI to recruitment workflows. These agents have been deployed for XOPA's current customers through a gradual rollout, within the following workflow:



As recruitment is a high-impact domain, the focus is to augment rather than replace human-decision making.

1. **Checkpoints for human approval:** Human checkpoints are maintained at key stages such as shortlisting and final selection, ensuring that consequential hiring decisions remain with human recruiters and hiring managers.
2. **Enabling effective user approval by:**
 - a. **Explainable outputs:** Matching scores and ranking rationale allow users to understand the basis of agent recommendations, enabling effective review.
 - b. **Training:** Training and onboarding processes are used to familiarise users with the system's functionality, including its limitations and potential failure modes
 - i. For recruitment, these include situations where agents misinterpret non-standard career paths, or produce lower-confidence evaluations for candidates with limited historical or structured data.
 - ii. Users are also trained to recognise situations where interview agents may not fully capture contextual factors such as communication style, domain-specific nuances, or atypical candidate experiences, and where additional human assessment may be required.
3. **User control and feedback loop:** The user can override recommendations and provide feedback. This can be used to refine system performance over time, creating an interaction loop between users and XOPA AI's agents.

2.3 Implement technical controls and processes

The agentic components that differentiate agents from simple LLM-based applications necessitate additional controls during the key stages of the implementation lifecycle.

Organisations should consider:

- **During design and development, design and implement technical controls.** The new components and capabilities of agents necessitate new and tailored controls. Depending on the agent design, implement controls such as access controls. Further, limit the agent’s impact on the external environment by enforcing least-privilege access to tools and data.
- **Pre-deployment, test agents for safety and security.** As with all software, testing before deployment ensures that the system behaves as expected. Specifically for agents, test for new dimensions such as overall task execution, policy adherence and tool use accuracy, and test across varied datasets to capture the full spectrum of agent behaviour.
- **When deploying, gradually roll out agents and continuously monitor them in production.** The autonomous nature of agents and the changing environment makes it challenging to account for and test all possible outcomes before deployment. Roll out agents gradually, supported with real-time monitoring post-deployment to ensure that agents function safely.
- **Throughout the lifecycle, implement change management and version control processes.** In complex agent ecosystems, changes to one agent can have cascading effects across interconnected workflows. As such, organisations should define clear triggers that initiate a change review process and categorise the needed changes accordingly.

2.3.1 During design and development, use technical controls

Organisations should design and implement technical controls in the agentic AI system to mitigate identified risks. For agents specifically, in addition to baseline software and LLM controls, consider adding controls for:

- New agentic components, such as planning, reasoning and tools
- Increased security concerns from the larger attack surface and new protocols
- Multi-agent interactions

When selecting suitable controls, consider the following:

- **Structural, rule-based vs model-based or prompt-layer controls:**
 - Rather than prompt-layer safeguards, consider implementing deterministic safeguards that operate at a system-level through predefined logic, especially for higher-risk actions.
 - For example, rather than instructing an agent not to use specific tools, access controls can be implemented at the tool layer to prevent those tools from being called in the first place, or only allowing them to be called in a certain way (e.g. read access only). Similarly, if an agent needs to follow a set procedure, building that sequence into the workflow is more robust than prompting the agent to adhere to it through instructions alone.
 - Prompt-layer safeguards also tend to be inconsistently defined across users, as opposed to system-level safeguards that can be consistently defined and enforced.

- However, in certain cases where risks are harder to define through fixed rules, model-based safeguards can be most effective e.g. to detect harmful content, which can manifest in different ways.
- **Runtime controls:** As agents interact with users and systems in real time, static safeguards configured at design time may not be sufficient to catch every risk. Runtime controls address this by monitoring and intervening during execution, such as rate limits to prevent excessive tool use or input validation to catch harmful responses before they are acted upon.

For illustration, these are some sample controls for agents. For a more comprehensive list, organisations can refer to CSA’s [Draft Addendum on Securing Agentic AI](#) and GovTech’s [Agentic Risk and Capability Framework](#).

Planning	<ul style="list-style-type: none"> ● Prompt agent to reflect on whether its plan adheres to user instructions ● Prompt the agent to summarise its understanding and request clarification from the user before proceeding ● Log the agent’s plan and reasoning for the user to evaluate and verify
Tools	<ul style="list-style-type: none"> ● Configure tools to require strict input formats ● Apply the principle of least privilege to limit tools available to each agent, enforced through robust authentication and authorisation ● For data-related tools: <ul style="list-style-type: none"> ○ Do not grant agent write access to tables in sensitive databases unless strictly required ○ Configure agent to let user take over control when keying in sensitive data (e.g. passwords, API keys)
Protocols	<ul style="list-style-type: none"> ● Use standardised protocols where applicable (e.g. agentic commerce protocols when agent is handling a financial transaction) ● For MCP servers: <ul style="list-style-type: none"> ○ Whitelist trusted servers and only allow agent to interact with servers on that whitelist ○ Sandbox any code execution
Multi-agent interactions	<ul style="list-style-type: none"> ● Require agents to communicate through structured schemas such as typed function calls rather than free text, to reduce unintended instructions passing between agents ● Limit shared memory access between agents

MCP as a governance layer

While usually considered as a connectivity protocol, MCP can potentially act as a governance layer as it sits between the agent and the enterprise systems it accesses. Organisations can consider defining controls on the MCP layer, such as filtering sensitive data that passes through the servers, logging all agent-to-system interactions, or whitelisting only trusted servers.

Terminal 3: Controls for accurate and secure agentic payments at the hardware level

Terminal 3 is a Hong Kong-based data privacy infrastructure provider that provides secure and privacy-preserving data processing without the need for data transfer.

The company deploys a finance payroll agent to automate its monthly payroll cycle, which involves computing the gross and net salary, government-mandated contributions, validating expense claims and cross-referencing account details before disbursing the overall salary.

The process is high stakes in nature as it involves handling sensitive personal data and executing financial transactions. To ensure that such actions are securely processed by the agent, Terminal 3 implemented the following technical controls:

- 1. Defining the scope of the agent's authority within its credentials:** Before the monthly payroll cycle begins, the Human Resources director issues a Verifiable Credential of Intent to the agent - a mathematically binding, cycle-scoped authorization defining:
 - a. which employee records the agent can access,
 - b. the applicable government-mandated contribution rates,
 - c. expense claim thresholds, eligible categories, and per-diem caps, and
 - d. the ceiling amount for the consolidated bank transfer, set as base salary plus a buffer margin and explicitly declared before each run.
- 2. Structurally separating sensitive data from the agent's context:** All sensitive personal data (e.g. employee id numbers, salary figures, and bank account details) is held exclusively within Terminal 3's Trusted Execution Environment (TEE) and never transmitted to the agent at any point. The agent operates solely on non-sensitive inputs and receives only opaque employee reference identifiers and categorical flag reasons as outputs. This architectural separation means there is no sensitive data in the agent's context to be leaked or extracted through adversarial inputs – the risk is eliminated by design.
- 3. Enforcing the boundaries of this authority through hardware:** Every agent action is routed through the TEE and verified against the credentialed parameters at hardware speed. Out-of-scope actions (whether from erroneous reasoning or adversarial prompts) are blocked before execution and logged. All sensitive computation happens exclusively within this secure boundary.
- 4. Recording a tamper-proof audit trail:** Each step of the agent's execution, including any blocked actions, is recorded on the T3N Immutable Ledger, a hardware-attested, cryptographically verifiable log, to provide the evidence chain for any post-incident investigation.

Cyber Sierra: Technical controls to improve accuracy and reliability in filling up due diligence questionnaires

Cyber Sierra is a Singapore-based company that is a Governance, Risk and Compliance (GRC) Agent Factory, building multi-agentic infrastructure for cyber operations, security compliance and audit tasks.

It developed TracyAI, which is a solution to automate responses to security and compliance questionnaires. Such questionnaires usually contain hundreds of repetitive questions relating to data protection, access controls, certifications and risk management. It does so through an agentic system that searches and reasons over multiple steps through past due diligence questionnaires, information security policy documents and an evidence folder. TracyAI was deployed in the Governance, Risk and Compliance team of a large financial institution based in Hong Kong, reducing a task that typically took more than 100 manual hours to 15 minutes.

To reduce the chances of inaccurate output and hallucination, the team implemented these technical controls:

- **Reflection architecture for agent to review its own outputs:** The agentic system incorporated two types of reflection nodes, implemented using LangGraph, to provide feedback to the agent after it generated its output:
 - **LLM Judge-based:** An LLM judge critiqued the output based on whether it addressed the question, the justification aligned with retrieved evidence, and no unsupported or fabricated claims were introduced.
 - **Metrics-based:** Similarity scores and RAGAS metrics like faithfulness, context relevance, response relevance score.

The agent would then review and refine its own output. If it failed to reach the passing threshold from both nodes within three iterations, its trajectory would be terminated.

- **Structured information to enhance agent understanding:** In the initial stages of development, the team observed that the agent was susceptible to inaccurate output when it referred to:
 - Multiple versions of the same document, some of which had expired, in the knowledge base; and
 - A previous questionnaire which relied on documents that had since expired.

To resolve this more deterministically, rather than intervening on the agent, the team structured the source information into a graph showing relationships between internal documents, policies, and certifications. This enabled the agent to understand how evidence connected across domains. This was then enriched with metadata to form a context graph that deterministically ensured that only the most relevant (based on location, topic, etc.) and current documents (based on version, expiry date etc.) were used.

Stability Solutions: Differing controls on internal-facing vs external-facing agents

Stability Solutions is a company based in the US and Singapore that builds and provides blockchain-powered infrastructure to record, preserve, and verify the integrity of data at scale. Their solutions include the Global Trust Network (“GTN”, a high-throughput cryptoless public blockchain, currently used in supply chain and logistics), and Monolith, an application that records the provenance of creative works by allowing creators to digitally fingerprint any file, issue C2PA manifests and provenance metadata, and embed IP licenses and AI training permissions, recording all of this in a robust, verifiable, and machine-readable format on the GTN.

Stability deploys their in-house agentic AI system, L3, which currently comprises 26 AI agents that run 24/7 across almost all company functions, including product development, software engineering, and marketing. Its internal agents have action-taking capabilities e.g. consolidating and archiving information across the entire company, sending daily morning briefs, monitoring and resolving network issues, acting as digital twins or personal assistants, and supporting engineering and development activities such as planning projects and writing code. L3 is optimised to have access to as much information as possible, enabling it to coordinate across teams (e.g. if the engineering team is planning an upgrade, L3 will inform them of planned marketing activities to avoid any disruptions).

When Stability developed its first external-facing agent, Howard, to showcase L3’s features and answer questions about L3 and the company, it recognised that the risk level was higher as the agent could be compromised by untrusted third parties. Compared to internal-facing agents, additional guardrails had to be implemented:

1. **Controls on data access:** The agent has its own memory, which is formed from information received from L3. L3’s agents, being aware of the agent’s role and risks associated with it, curate a special dataset for the agent.
2. **Controls on interactions:** The agent interacts through chats, either via Slack or a web interface. Stability’s human employees vet third party inputs before they are sent to the agent. This enables them to filter out prompt attacks or invasive questions designed to extract sensitive information such as personal data and trade secrets. As the agent improves in safety and reliability, this control is expected to become unnecessary.
3. **Controls on agent configuration:** Express guardrails and defined personality protocols were implemented to minimise the likelihood of the agent disclosing sensitive information, such as trade secrets, personal or confidential data.

2.3.2 Before deploying, test agents

Organisations should test agents for safety and security before deployment. This provides confidence that the agents work as expected and controls are effective. Best practices on software and LLM testing are still relevant, such as unit and integration testing for software systems, as well as selecting representative datasets, and useful metrics and evaluators for LLM testing. Organisations can refer to previous guidance, such as the [Starter Kit for Testing of LLM-based Applications for Safety and Reliability](#).

However, organisations should adapt their testing approaches for agents. Some considerations include:

- **Testing for new risks:** Beyond producing incorrect outputs, agents can take unsafe or unintended actions through tools. Organisations can consider testing for:²⁸
 - **Overall task execution:** Whether agent can complete task accurately
 - **Policy compliance:** Whether an agent follows defined SOPs and routes for human approval when required
 - **Tool calling:** Whether an agent calls the right tools, with the right permissions, with the right inputs and in the right order
 - **Robustness:** As agents are expected to react and adapt to real-world situations, test for their response to errors and edge cases
- **Testing entire agent workflows:** Agents can take multiple steps in sequence without human involvement. Thus, beyond testing an agent's final output, agents should be tested across their entire workflow, including reasoning and tool calling.
- **Testing agents individually and together:** Beyond individual agents, testing should be carried out at the multi-agent system level, to understand any emergent risks and behaviours when agents collaborate, such as competitive behaviours or the impact on other agents when one agent has been compromised.
- **Testing in real or realistic environments:** As agents may be expected to navigate real-world situations, testing should occur in a properly configured execution environment that mirrors production as closely as possible, such as using tool integrations, external APIs, and sandboxes that behave as they would in deployment. However, organisations should calibrate the need for realism against the risk of prematurely allowing agents to access tools that affect the real world.
- **Testing repeatedly and across varied datasets:** Agent behaviour is inherently stochastic and context dependent. Testing should thus be done at scale and across varied datasets to observe any unexpected low-probability behaviours, especially if they are of high-impact. This requires generating test datasets that cover different conditions that agents may encounter. These tests should also be run multiple times to check for stability (e.g. the same inputs and contexts render consistent outputs), including minor perturbations where needed.
- **Evaluating test results at scale:** Reliably evaluating test results at scale is a known challenge for LLM testing. Agents add a further layer of complexity as their workflows can be long and contain unstructured information that cannot be easily processed by humans or automated scripts.

²⁸

For an example of new agentic aspects to test for, see Microsoft Foundry, [Agent evaluators](#).

Organisations may consider using different evaluation methods for different parts of the agentic workflow (e.g. deterministic tests for structured tool calls vs LLM or human evaluation for unstructured agent reasoning). However, there is still a need to evaluate agents holistically, so that agent patterns across steps can be evaluated. Current industry solutions thus include defining LLMs or agents to evaluate other agents, while incorporating human-in-the-loop review.²⁹

²⁹ For an example of agent evaluation solutions, see AWS Labs, [Agent Evaluation](#).

Google x Singapore Government: Testing computer use agents

To understand how autonomous agents behave in practice, Google, CSA, GovTech and IMDA initiated a sandbox to test computer-use agents for potential public sector use cases,¹ namely:

- **Automated quality assurance (QA):** Automated QA testing for government digital services
- **AI safety testing:** Automating the safety testing of AI software like third-party chatbots
- **Social assistance applications:** Assisting citizens in navigating and applying for social assistance programmes, helping to streamline complex processes

While each use case had its own specific risk profile, the sandbox surfaced some general insights on agentic testing methodology, including:

- **Using realistic testing data and environments:** In the AI safety testing use case:
 - **Data:** Safety testing prompts typically vary significantly in length, language, and format. A test dataset was curated based on real-world jailbreak prompts that reflected such variety. Testing revealed that the agent captured variation in format and language well, but its reliance on screenshots meant that it had difficulty reproducing longer responses accurately.
 - **Environment:** The agent was tested against real chatbots (though some within a staging environment for safety), which allowed review of its robustness in navigating different user interfaces, such as its ability to refresh a session or retry prompts when encountering user limits, longer loading times, or other errors.
- **Logging and evaluating agent's reasoning at each step of the workflow:** This enabled more detailed evaluation of the test results to debug unexpected behaviour. For example, in the social assistance use case, the agent was tested on data minimisation i.e. whether it could distinguish between information relevant to an application and extraneous personal details it should not include. In a significant number of cases, the agent included these extraneous details. A review of its reasoning process showed that it sometimes considered such information nevertheless relevant, or presumed that it needed to include all information given to it. This could then enable more specific prompt-layer safeguards.

Identifying attack vectors based on the use case: Through malicious third-party prompts embedded in external websites, the agent could be redirected towards harmful actions (also known as indirect prompt injection). While this is a general risk, it can manifest differently in each use case. For example, for AI safety testing, it was necessary for the agent to interact with third-party chatbots. Attackers could thus create a malicious chatbot that included harmful instructions in its responses. This threat scenario was specifically tested. It was observed that in some instances, the agent navigated to arbitrary URLs after being instructed to do so. This could be mitigated through controls such as but not limited to website whitelists or prompt design.

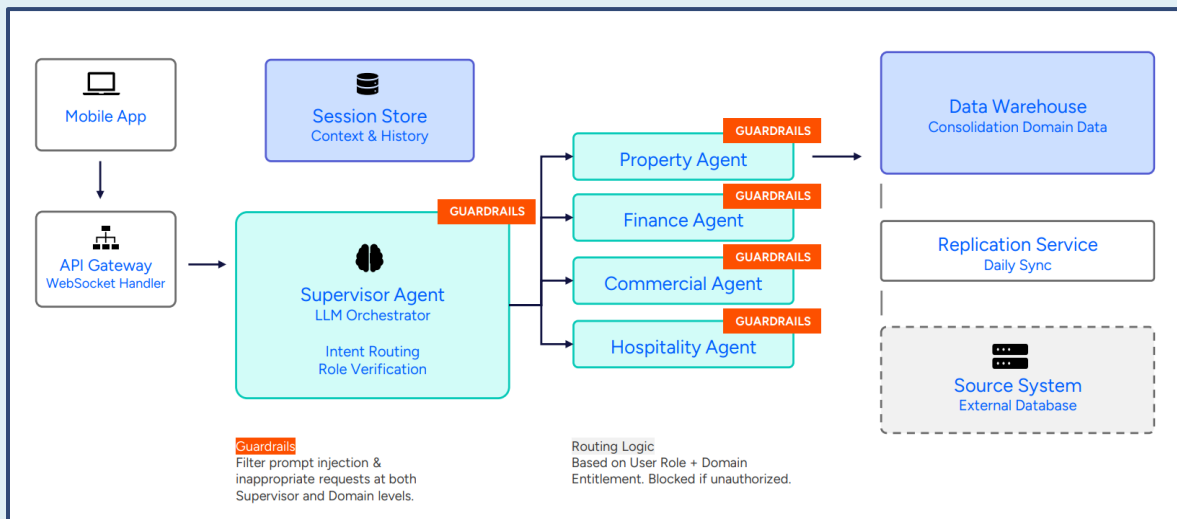
- **Incorporating human-in-the-loop during testing:** Computer-use agents tend to have broader action-space and autonomy because of their flexibility in navigating different user interfaces. As they are still nascent, incorporating human oversight and review during the testing process can surface any unexpected or emergent behaviours.

For more information on the use cases deployed and risks tested, see the full paper [here](#).

City Developments Limited x Knovel Engineering: Testing data access boundaries in agentic systems

City Developments Limited (CDL) is a Singapore-based developer and asset owner/operator, with a portfolio spanning residential, commercial, hospitality, and integrated developments. Knovel Engineering is a Singapore-based AI consultancy that helps enterprises and government agencies operationalize trusted AI.

As part of the Global AI Assurance Sandbox, Knovel tested CDL's internal agentic system, which answers queries by retrieving business insights, accessing internal knowledge, and performing task-oriented workflows. Agent orchestration handles multi-step tasks such as summarising sector performance or retrieving specific operational metrics.



CDL identified data leakage as a key risk of its system and prioritised it for testing. The aim was to determine if data access controls and boundaries were properly enforced across users i.e. a user would only be able to obtain information that s/he had access rights to.

To test this, Knovel adopted the following methodology:

- Used a matrix of 7 user accounts x 4 domains (Property, Financial, Commercial Hospitality) to validate adherence to boundaries
- Logged into user accounts with restricted domain access and conducted multi-turn conversations that progressively introduced indirect queries targeting out of scope domains (e.g., a *Property-domain* user asking about *Hotel A's Q4 revenue*)
- Compared outputs from restricted accounts with those from legitimately authorised ones to detect info leakage
- Additionally, the system's tendency to be helpful was exploited by providing partially correct system prompts and tool call structures, observing whether the system inadvertently corrected or completed redacted information.

For more details on this case study, including insights from the testing and other risks that were tested, see [here](#).

2.3.3 When deploying, continuously monitor and test

Pre-deployment testing establishes a useful baseline but needs to be complemented by continuous monitoring and testing during deployment. Agents adapt to real-time conditions and their behaviour may change. For multi-agent systems especially, failure modes such as miscoordination or emergent behaviours may only manifest through agent behaviour over time and under realistic conditions. This section provides recommendations to mitigate such risks, including:

- Gradual deployment of agents
- Continuous testing and monitoring
- Robust change management

Gradual deployment of agents

Organisations should consider gradually rolling out agents into production to control the amount of risk exposure. Such rollouts can be controlled based on:

- Users of agents e.g. rolling out to trained or experienced users first
- Tools and protocols available to agent e.g. restricting agents to more secure, whitelisted MCP servers first
- Systems exposed to agent e.g. using agents in lower-risk internal systems first

GovTech: Phased rollouts of coding assistants to incrementally monitor risks while preparing controls for new features

The Government Technology Agency of Singapore, or GovTech, is a statutory board in Singapore that develops digital government services and drives public sector transformation.

In Oct 2025, GovTech rolled out two agentic coding assistants within the organisation, namely Windsurf and GitHub Copilot (Agent Mode), before rolling out Claude Code in Apr 2026. After extensive tests on common vectors for failures and threats (such as agentic AI manipulation, hallucination, and compromise of MCP servers) and implementation of corresponding technical controls, GovTech adopted a phased approach for rolling out these agentic capabilities. This enabled it to empower development teams with enhanced productivity early, while monitoring risks and incrementally developing further controls for new AI features as the technology evolves.

In the first phase, the rollout was limited to contain the blast radius of any potential risks:

- **Internal employees:** Agentic coding assistants were only rolled out to GovTech employees, a smaller group of developers within a single organisation that could be trained, educated on the using agents responsibly versus the whole of government.
- **No MCP allowed:** Users were only allowed to use the built-in capabilities of the agent e.g. multi-step reasoning to plan and execute tasks within the developer's environment without access to MCP servers.
- **Low-risk systems:** Systems that were not critical infrastructure and had lower levels of data access.

While the first phase was ongoing, GovTech started to put in place the necessary controls and infrastructure to support broader agent capabilities in the second phase, such as central logging and monitoring, and trialing its 1st iteration of MCP Governance Framework – a path for developers to connect to whitelisted MCP servers from a containerized sandbox. Red teaming was conducted to validate the effectiveness of these guardrails against external threats.

In the second phase, which is currently beginning, the rollout will be expanded progressively to include public officer developers in other government agencies, then non-government vendors and contractors. The revised MCP Governance Framework will also be made accessible to developers in GovTech first and gradually to the rest of the government, starting with Figma and subsequently to other managed MCPs like GitLab based on user demand and risk factors

The learnings from the first phase informed the rollout in the second phase - such as:

- **Resolution of teething issues** e.g. network trust configuration issues.
- **Reducing cognitive load of human approvers** by allowing users to opt for OS-native sandboxes which allow agents to run some commands more flexibly without constant seeking of human approval.
- **Reducing adoption friction** of the MCP Governance Framework by removing the need for complex local development containers and self-hosted infrastructure and changed the architecture to leverage the AI coding tools' native sandboxes together with Cloudflare's managed MCP gateway, where available

As agentic AI evolves rapidly, GovTech continues to experiment, test and monitor AI's impact to uncover gaps and new learnings. A balanced approach towards risk mitigation versus developer experience is key for successful agentic AI adoption and developer productivity.

Continuous testing and monitoring

Organisations should continuously monitor and log agent behaviour post-deployment, and establish reporting and failsafe mechanisms for agent failures or unexpected behaviours. This allows the organisation to:

- **Intervene in real-time:** When potential failures are detected, stop agent workflow and escalate to a human supervisor e.g. if agent attempts unauthorised access.
- **Debug when incidents happen:** Logging and tracing each step of an agent workflow and agent-to-agent interactions help to identify points of failure.
- **Audit at regular intervals:** This ensures that the system is performing as expected.

Monitoring and observability are not new concepts, but agents introduce some challenges. As agents execute multiple actions at machine speed, organisations face the issue of extracting meaningful insights from the voluminous logs generated by monitoring systems. This becomes more difficult when high-risk anomalies are expected to be detected in real-time and surfaced as early as possible.

Key considerations when setting up a monitoring system include:

- **What to log:** Organisations should determine their objectives for monitoring (e.g. real-time intervention, debugging, integration between components) to identify what to log. In doing so, prioritise monitoring for high-risk activities such as updating database records or financial transactions.
- **How to effectively monitor logs:** Organisations can consider implementing these practices:
 - **Monitor on multiple layers,** such as the user-agent interaction, agent-tool invocation, and model reasoning layers to identify specific sources of failure.
 - **Define alert thresholds:**
 - **Programmatic, threshold-based:** Define alerts when agents trigger thresholds e.g. agent attempts unauthorised access or makes too many repeated tool calls within a specified timeframe.
 - **Outlier / anomaly detection:** Use data science or deep learning techniques to process agent signals and identify anomalous behaviour that may indicate malfunctions.
 - **Agents monitoring other agents:** Design agents to monitor other agents in real-time, flagging any anomalies or inconsistencies.
 - **Define specific interventions:** For each alert type, consider what the level of intervention should be. Some degree of human review should be incorporated, proportionate to the risk level. For example, lower-priority alerts can be flagged for review at a scheduled time, whereas higher-priority ones might require temporarily halting agent execution until a human reviewer can assess. In the event of catastrophic agentic malfunction or compromise, commensurate measures such as termination and fallback solutions should be considered.
 - **Human-in-the-loop:** Some degree of human review or human-in-the-loop is useful to detect any emergent behaviours that may not have been previously accounted for.
 - **Integrate with observability platforms:** Consider integrating agent monitoring with existing observability and debugging platforms, including standards like OpenTelemetry for tracing, to enable detailed analysis of tool calls, traces, and system interactions across the agent workflow.
 - **Ensure log immutability:** Maintain complete audit trails by ensuring that problematic agent trajectories and failures cannot be deleted, preserving them for analysis and compliance purposes.
 - **Establish feedback loops:** Establish feedback loops to feed monitoring insights back into training datasets and evaluation frameworks for continuous improvement.

Finally, continuously test the agentic system even post-deployment to ensure that it works as expected and is not affected by model drift or other changes in the environment.

Robust change management

As an agentic system becomes more complex, small modifications can cascade into larger impact. To safeguard against this, consider:

- **Defining triggers for a change review process.** This can include technical triggers (model updates, tool modifications), environmental triggers (domain shifts, business context changes), performance triggers (anomalous behaviour, degraded performance), and regulatory triggers (changes in compliance requirements).
- **Categorising review changes by risk.** Minor changes such as prompt refinements may follow lighter review processes, while material changes such as model updates or autonomy adjustments may require a full governance review, and critical changes affecting high-stakes decisions may mandate an immediate re-risk assessment.

2.4 Enable end-user responsibility

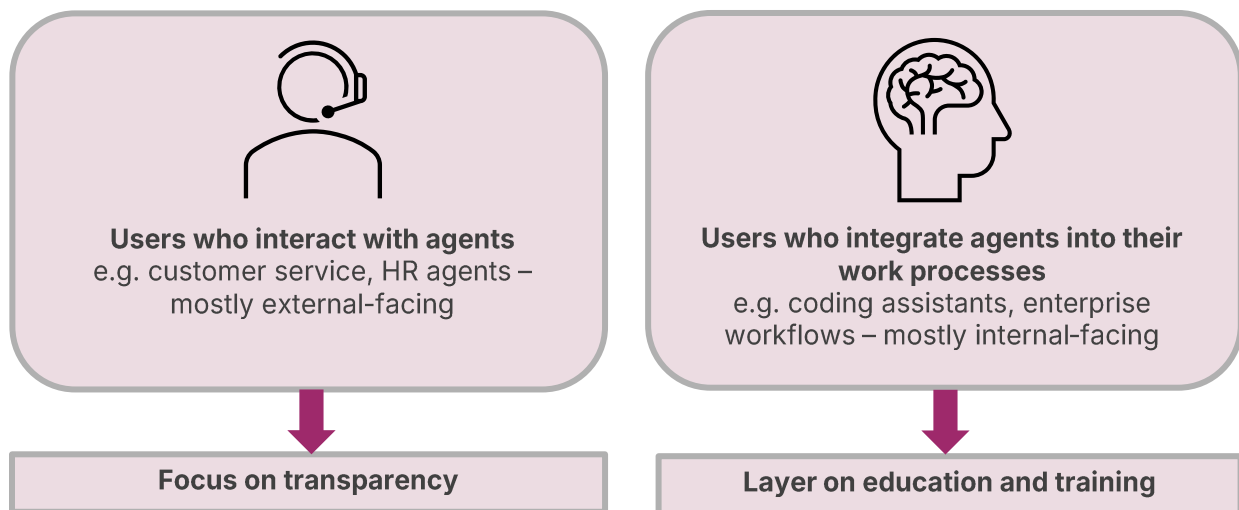
Ultimately, end users are the ones who use and rely on agents, and human accountability also extends to these users. Organisations should provide sufficient information to end users to promote trust and enable responsible use.

Organisations should consider:

- **Transparency:** Users should be informed of the agents' capabilities (e.g. scope of agent's access to user's data, actions the agent can take) and the contact points whom users can escalate to if the agent malfunctions.
- **Education:** Users should be educated on proper use and oversight of agents (e.g. training should be provided on an agent's range of actions, common failure modes like hallucinations, usage policies for data), as well as the potential loss of trade craft i.e. as agents take over more functions, basic operational knowledge could be eroded. Hence sufficient training (especially in areas where agents are prevalent) should be provided to ensure that humans retain core skills.

2.4.1 Different users, different needs

Organisations should cater to different users with different information needs, to enable such users to use AI responsibly. Broadly, there are two main archetypes of end-users – those who interact with agents, and those who integrate agents into their work processes or oversee them.



2.4.2 Users who interact with agents

Such users usually interact with agents that act on behalf of the organisation e.g. customer service or sales agents. These agents tend to be external facing, although they can also be deployed within the organisation e.g. a human resource agent that interacts with other users in the organisation.

For these users, focus on transparency. Organisations should share pertinent information to foster trust and facilitate proper usage of agents. Such information can include:

- **User’s responsibilities:** Clearly define the user’s responsibilities, such as asking the user to double-check all information provided by the agent. Where appropriate, allow users to set their own approval thresholds and boundaries for agent actions based on their individual risk tolerance, beyond organisation-defined limits.
- **Interaction:** Declare upfront in the user interface that the users are interacting with agents at the point of interaction, rather than writing it only in separate documentation.
- **Agents’ range of actions and decisions:** Inform the users on the range of actions and decisions that the agent is authorised to perform and make.
- **Data:** Be clear on how user data is collected, stored, and used by the agents, in accordance with the organisation's data privacy policies. Where necessary, obtain explicit consent from users before collecting or using their data for the agents.
- **Human accountability and escalation:** Provide users with the respective human contact points who are responsible for the agents, whom the users can alert if the agents malfunction or if they are dissatisfied with a decision.

2.4.3 Users who integrate agents into their work processes

Such users typically utilise agents as part of their internal workflows e.g. coding assistants, automation of enterprise processes. The agent acts for and on behalf of the user.

For these users, in addition to the information above, layer on education and training so that users can use the agents responsibly. Key aspects include education and training on:

- **Foundational knowledge on agents**
 - **Relevant use cases**, so that the users understand how to best integrate the agents into their day-to-day work, and the scenarios under which the use of agents should be restricted (e.g. do not use an agent for confidential data)
 - **Instructing the agents** e.g. general best practices in prompting
 - **Agents’ range of actions**, so the user is aware of their capabilities and potential impact
- **Effective oversight of agents**
 - **Common agent failure modes**, such as hallucinations, getting stuck in loops after errors, so that the user can identify and flag out issues.
 - **Ongoing support**, such as refreshers to update on latest features and common mistakes
 - **Feedback loops** e.g. when users override agent actions or identify wrong actions, enable them to report this such that it can be used to improve the agent processes if needed.
- **Potential impact on tradecraft and business continuity**
 - As agents take over entry-level tasks, which typically serve as training for new staff, this could lead to skill degradation, in which users lose basic operational knowledge.³⁰
 - **This can create business continuity risks** in which users may no longer know how to perform critical processes manually when agents malfunction or become unavailable.
 - **Organisations should identify core capabilities of each job** and provide sufficient training and work exposure so that users retain foundational skills.

³⁰ For an example of such a study, see [The Quarterly Journal of Economics, Generative AI at Work](#).

Workday: Transparency towards end-users interacting with finance and HR agents

Workday is a global enterprise AI platform for managing people, money and AI agents, and is used by more than 11,500 organisations around the world and across industries, including more than 65% of the Fortune 500.

To streamline its internal financial and human resources operations, Workday has implemented a suite of proprietary AI agents. Specifically, the use cases for HR agents include:

- Recruiter Agent which supports screening and evaluation of candidates for job roles.
- Conversational Scheduling Agent which manages the full interview, orientation, and event session scheduling lifecycle.

Workday has implemented several design elements, in line with its commitment to mitigate potential risks to downstream users of deployed AI tools. This includes ensuring that AI agents are transparent with end-users and adhere to Workday's Responsible AI standards, enabling users to make informed and responsible use of the AI agent(s):

- **Identity and range of actions:**
 - The **user interface** of the AI agent clearly provides notice to users that the tool is powered by AI and is intended to support a select range of actions whenever a user interacts with an AI agent e.g. in Slack or Microsoft Teams.
 - The **factsheets** provided to users also specify:
 - What an agent can do (such as reviewing candidates against job requirements, summarising talent profiles, sending reminders to interviewers and interviewees, collecting post-interview feedback)
 - What it cannot do (such as making decisions related to hiring promotion or succession, initiating changes to worker records), reinforcing that users and HR remain in control.
- **Reasoning:** When agents provide recommendations in sensitive workflows—for example, when the Recruiter Agent supports the screening of candidates for a role or suggest development actions for a worker—they accompany each recommendation with an explanation of their reasoning, including the data considered, key factors that influenced the recommendation, and any noted uncertainties or risks.

By making the agent's reasoning visible, users are equipped with relevant essential information needed to review and use the results in a responsible manner. These transparency measures are designed to support users in understanding how they should engage with the agent. In addition, they are designed to highlight where human judgment is required in order to ensure human decision making and to prevent the automation of higher risk decisions.

Ant International: Equipping end users to robustly design their own agentic workflows

With main operations across Asia, Europe, the Middle East and Latin America, Ant International is a leading global digital payment, digitisation and financial technology provider. Through collaboration across the private and public sectors, her unified techfin platform supports financial institutions and merchants of all sizes to achieve inclusive growth through a comprehensive range of cutting-edge digital payment and financial services solutions.

Internally, Ant International deploys AI agents, leveraging on third-party LLMs, for use cases such as data quality assurance and cybersecurity. To ensure that humans can maintain control over agentic workflows, the company has been trialling a new framework for the design and development of agentic AI, the High-Order Program or HOP framework. The HOP framework is embedded into an agent builder and provides a process by which users can describe their agentic goals in plain language and confirm the agentic workflow before it is converted into code by the agent builder.

- 1. End-users can read and write agentic workflow specs themselves:** HopSpec, a specification document that lays out step by step what the agent should do and the constraints (i.e. SOP), is written in easily understandable text (Markdown). Requirements for specific agentic workflows can be drafted and reviewed by end-users, including domain experts, compliance officers, and product managers.
- 2. End-users can iterate with the agent builder on what good agentic workflows look like:** When a task is less mature or does not have a well-defined SOP, it may be difficult to define the specifications. HOP iterates with the user on how a task may be completed, exploring potential solution paths and providing confidence scores for its proposed workflows for the user to evaluate.
- 3. Agentic workflows come with built-in verification:** To safeguard against cascading errors, all agentic workflows come with both customisable and built-in verification steps, such as reconstructing the original task from the output, cross-checking the response with separate calls to other LLMs, and whether the response fits the expected format. If any of the verifications fail, the user will be notified of the agent's retries, and the agentic workflow will end after a specific number of failed tries.

For example, when developing an agentic workflow for detecting reverse shell attacks, where victim's machine bypasses its defence and connects out to the attacker through a malicious code, the user may not have any well-defined SOP as the malicious code could be residing in various forms and resemble a normal outbound connection.

In this situation, the user can start with a query such as "Are there any suspicious outgoing connections that look like a reverse shell attack?". The agent builder can then craft a SOP that systematically examines attributes such as file names, file paths, and destination IPs to evaluate this. After iterating with the user and upon the user's confirmation, the workflow will be automatically translated into code that can be repeatedly executed and verified.

Annex A: Further resources

1. Introduction to Agentic AI

What is Agentic AI?	<ul style="list-style-type: none"> • AWS, Agentic AI Security Scoping Matrix: A framework for securing autonomous AI systems • WEF, AI Agents in Action: Foundations for Evaluation and Governance • Anthropic, The 2026 State of AI Agents Report and 2026 Agentic Coding Trends Report • OpenAI, A Practical Guide to Building Agents • IBM, The 2026 Guide to AI Agents
Risks of Agentic AI	<ul style="list-style-type: none"> • GovTech, Agentic Risk & Capability Framework • CSA, Draft Addendum on Securing Agentic AI • OWASP, Multi-Agentic System Threat Modelling Guide • OWASP, Top 10 for Agentic Applications for 2026 • IBM, AI agents: Opportunities, risks, and mitigations • Partnership on AI, Six AI Governance Priorities for 2026 • Palo Alto Networks, A Complete Guide to Agentic AI Governance • Infosys, Agentic AI risks to the enterprise, and its mitigations

2. MGF for Agentic AI

Assess and bound the risks upfront	<p>Agentic governance in general</p> <ul style="list-style-type: none"> • EY, Building a risk framework for Agentic AI • McKinsey, Deploying agentic AI with safety and security: A playbook for technology leaders • Bain, Building the Foundation for Agentic AI • OWASP, State of Agentic AI Security and Governance 1.0 • AWS, AI agents in enterprises: Best practices with Amazon Bedrock AgentCore <p>Risk assessment and threat modelling</p> <ul style="list-style-type: none"> • OWASP, Agentic AI – Threats & Mitigations • OWASP, Multi-Agentic System Threat Modelling Guide • Cloud Security Alliance, Agentic AI: Understanding Its Evolution, Risks, and Security Challenges • Cloud Security Alliance, The AI Agent Governance Gap: What CISOs Need Now • Google Cloud, AI risk and resilience <p>Agent limits and agent identity</p> <ul style="list-style-type: none"> • Meta, Agents Rule of Two: A Practical Approach to AI Agent Security • OpenID, Identity Management for Agentic AI • Cloud Security Alliance, The Agentic Trust Framework: Zero Trust Governance for AI Agents
---	--

	<ul style="list-style-type: none"> Google Cloud Office of the CISO, These 4 AI governance tips help counter shadow agents
<p>Make humans meaningfully accountable</p>	<p>Allocating responsibility within and outside an organisation</p> <ul style="list-style-type: none"> Carnegie Mellon University, The ‘Who’, ‘What’, and ‘How’ of Responsible AI Governance IBM, AI agent governance: Big challenges, big opportunities CSA and FAR.AI, Securing Agentic AI: A Discussion Paper McKinsey, Accountability by design in the agentic organization McKinsey, State of AI trust in 2026: Shifting to the agentic era <p>Designing for meaningful human oversight</p> <ul style="list-style-type: none"> Partnership on AI, Prioritizing real-time failure detection in AI agents Permit.IO, Human-in-the-Loop for AI Agents: Best Practices, Frameworks, Use Cases, and Demo
<p>Implement technical controls and processes</p>	<p>Technical controls</p> <ul style="list-style-type: none"> GovTech, Agentic Risk & Capability Framework CSA, Draft Addendum on Securing Agentic AI OpenAI, Building Governed AI Agents - A Practical Guide to Agentic Scaffolding Google, Google’s Approach for Secure AI Agents: An Introduction Microsoft, Introducing the Agent Governance Toolkit: Open-source runtime security for AI agents Microsoft, Governance and security for AI agents across the organization Google Cloud, Disconnected but resilient: Securing agentic AI at the extreme edge Google, Our 2026 Responsible AI Progress Report <p>Testing and evaluation</p> <ul style="list-style-type: none"> Microsoft, Microsoft Agent Evaluators Anthropic, Demystifying evals for AI agents LangChain, State of Agent Engineering AWS, Evaluating AI agents: Real-world lessons from building agentic systems at Amazon IBM, What is AI Agent Evaluation? <p>Monitoring and observability</p> <ul style="list-style-type: none"> Microsoft, Top 5 agent observability best practices for reliable AI
<p>Enabling end-user responsibility</p>	<ul style="list-style-type: none"> Deloitte, State of AI in the Enterprise: The untapped edge Zendesk, What is AI transparency? A comprehensive guide Harvard Business Review, The Perils of Using AI to Replace Entry-Level Jobs

Annex B: Call for feedback and case studies

Call for feedback: This is a living document, and we invite suggestions on how the framework can be updated or refined. The following questions can be used as a guide:

- **Introduction to Agentic AI:** Are the descriptions of agentic AI systems accurate and sufficiently comprehensive for readers to obtain a clear overview of the governance challenges of agentic AI? Are there other risks that should be included?
- **Proposed Model Governance Framework:** Are the four dimensions of the framework practical and applicable? Are there any other dimensions that should be included? For each dimension, are there specific governance and technical challenges and best practices that should be included?

Call for case studies: We also invite organisations to continue submitting their own agentic governance experiences as case studies on how specific aspects of the framework can be implemented, to serve as practical examples of responsible deployment that other organisations can refer to. Case studies should ideally involve an organisation's deployment of an agentic use case that demonstrates one of the dimensions of the framework.

Please note that any feedback and case studies may be incorporated into an updated version of the framework, and contributors will be acknowledged accordingly. Please submit your feedback and case studies at this link: <https://go.gov.sg/mgfagentic-feedback>.

Acknowledgements

We extend our sincere gratitude to the organisations and individuals that supported the development of this document by sharing their feedback and real-world experiences.

Industry

- > Advai Limited
- > Aethryx
- > AIDX Tech
- > Aires A.T Private Limited
- > Airwallex
- > Alibaba Cloud
- > Ant International
- > Artha Solutions Pte Ltd
- > Amazon Web Services
- > Baker Tilly Malaysia
- > CDL
- > Cleanverse International Pte Ltd
- > Cyber & Artificial Intelligence for Future Impact
- > Cyber Sierra (Fort One Technologies Pte. Ltd.)
- > Dayos Pte. Ltd.
- > DBS
- > Dynamo AI
- > Ernst & Young
- > Ethics Advisory Services
- > Ferz.AI
- > Finiteloop
- > Flux AI
- > Gradient Institute
- > Google
- > Grab
- > HM Strategy
- > IBM
- > Imagenz Pte Ltd
- > Interpol Singapore
- > KASIKORN Business-Technology Group (KBTG)
- > Knovel
- > Levy Lens
- > Mastercard
- > Microsoft
- > MSD
- > NCS
- > Nobulex
- > OCBC
- > OpenSymbolicAI
- > Protos Labs
- > PwC
- > Resaro
- > Safer Agentic AI Working Group
- > Salesforce
- > Singapore Computer Society
- > Security Awareness Consultants SL
- > Singapore Fintech Association
- > Sovaigh
- > SP Group Pte Ltd
- > Stability Solutions
- > TomorrowX
- > Vaara
- > Verixiom Pte. Ltd.
- > Workato
- > Workday
- > XOPA
- > ZLAR
- > Tech Mahindra
- > Temus
- > Tencent
- > Terminal 3
- > The Box Commons

Government agencies

- > Competition and Consumer Commission of Singapore (CCS)
- > Cyber Security Agency of Singapore (CSA)
- > Government Technology Agency of Singapore (GovTech)
- > Home Team Science and Technology Agency (HTX)
- > Intellectual Property Office of Singapore (IPOS)
- > Ministry of Education (MOE)
- > Ministry of Digital Development and Information (MDDI)
- > Monetary Authority of Singapore (MAS)