

Microsoft Certified: Multi-Agent AI Solutions Expert

Dominios, pesos oficiales y habilidades clave del temario de AI-500, para repasar antes del examen.

Nivel: Advanced (Expert) Nota de corte: 700/1000 Fuente: guía de estudio oficial de Microsoft Learn

1. Architect multi-agent solutions

15–20%

Design logical architecture for multi-agent solutions

- Antes de crear agentes, elige el nivel mínimo de complejidad: una llamada directa al modelo si basta con un prompt, un agente único con herramientas como opción habitual y varios agentes solo ante límites de seguridad o cumplimiento, varios equipos, crecimiento previsto o cuando las pruebas demuestran que un agente no basta. Los bucles de control necesitan criterios de aceptación, límite de iteraciones y una alternativa como escalar a una persona. La intervención humana se define como aprobación o comentario, opcional u obligatoria, y conviene guardar el estado en esos puntos para reanudar sin repetir el trabajo.
- Cada agente debe tener una identidad propia con roles RBAC mínimos y herramientas permitidas de forma explícita (allowed, tools, approbaciones para operaciones de alto riesgo). En Foundry, la identidad del agente sirve para el acceso desatendido o para dar niveles distintos por agente; la identidad administrada del proyecto da el mismo acceso a todos, y el paso de identidad OAuth conserva los permisos de cada usuario. A2A conecta herramientas, recursos y prompts, y el transporte es sído en local o Streamable HTTP en remoto. A2A conecta MCP y servidores MCP en un único punto de conexión con autenticación y versiones centralizadas, y Foundry CG ofrece bases de conocimiento compartidas que respetan los permisos del usuario.
- Persistencia por niveles separa el estado de sesión (Agent Session) y proveedores de historial, como Cosmos DB, para reanudar en otra instancia; el estado compartido del equipo (estado del flujo de trabajo con ámbitos de control y cumplimiento) comparte las interacciones de los asistentes. Los agentes hospedados ejecutan tu contenedor compartido y puntos de control duraderos y la memoria semántica a largo plazo (almacenes tu memoria de espacios aislados por sesión con escalamo, identidad y punto de conexión administrados, y su costo depende de la CPU y memoria por sesión. Durable Extension en Azure Functions aporta orquestación duradera sin server, y el autonoexecute, tool con argumentos y resultados y propagación del contexto W3C hasta los servidores MCP. Agent Framework exporta trazas a Application Insights; los datos confidenciales solo se capturan en desarrollo y los puntos de control permiten reproducir ejecuciones completas de flujos de trabajo.

2. Develop multi-agent solutions in Azure

30–35%

Design and implement advanced prompt engineering strategies in multi-agent solutions

- La ingeniería de solicitudes avanzada combina ejemplos de pocos disparos como pares usuario-asistente, inyección dinámica de contexto con proveedores de contexto antes de cada ejecución, y directrices defensivas: salidas alternativas, repetición de instrucciones y Prompt Shields para ataques ocultos en contenido de herramientas. El agente como herramienta mantiene el control en el agente principal. Los AIContextProvider inyectan instrucciones, pero de vida corta, versiones limitadas de agente, evaluación antes de publicar y supervisión.
- Mensajes en Microsoft Foundry cambian después del estado con respuestas de ejemplo. DPO con pares preferidos/rechazados y RFT con un evaluador. Separar siempre los datos de entrenamiento, validación y prueba y referenciarlos, pero no garantizarlos, el comportamiento dentro del ámbito y contrapunto de reserva y

- **Design** al punto de control agent tareas reservadas, context management, and permission de datos actualizados; reentrena cuando cambia los datos de la tarea o se refra el modelo base.

- La memoria de Foundry Agent Service es a largo plazo: extrae, consolida y recupera perfiles de usuario, resúmenes de chat y procedimientos. El agente aísla los recuerdos por usuario ({{user.id}} con el encabezado x-memory-user-id), un ID y la eliminación por Azure AI Search de los datos de estado para la nueva fragmentación; tamaño fijo con solapamiento por secciones o semántica y fragmentar también cuando un único vector representa mal el contenido. Usa el mismo Foundry IO con Azure AI Search en bases de conocimiento reutilizables: cada base agrupa orígenes indexados y remotos (SharePoint, web, servidores MCP) y la recuperación agnética divide la pregunta en subconsultas RAG y el clasificador semántico @search.relevance.score) mejoran la precisión.

Build and integrate tool ecosystems

- **Builder de herramientas MCP:** Los agentes de la consumen como herramienta MCP (knowledge_base_retrieve) con la que se llama a funciones. El modelo propone la llamada y la aplicación la ejecuta y devuelve el resultado con su tool_call_id. tool_choice permite dejar decidir al modelo (auto), impedir llamadas (none) o forzar una función concreta. En Agent Framework, la exposición progresiva (ctx.add_tools) y el middleware de funciones controlan que herramientas se ven y cuándo se ejecutan.
- **MCP en Functions:** Los errores de protocolo (errores JSON-RPC, por ejemplo una herramienta desconocida) de los SDK en Functions o Container Apps, exponer flujos de Logic Apps Standard o convertir una API REST de API Management en herramientas MCP gobernadas con directivas.
- **Trata como no confiables las descripciones y resultados de herramientas remotas y validalos antes de pasarlos al modelo:** si la herramienta declara outputSchema, el cliente debe validar structuredContent. Activa la detección de degradación elegante sin ocultar errores.

Implement multi-Agent orchestration

- Con evaluadores como `Tool Call Success` o `Tool Output Utilization`.
- Una estrategia de agregación; el chat de grupo usa un administrador central que elige quién habla; `handoff`
- Las aprobaciones pueden limitarse a herramientas sensibles (`approval_mode` o `ApprovalRequiredAgentFunction` en `Agent Framework`, requiere `approval` en el MCP de Foundry) o pausar tras la respuesta de un agente o del plan de Magentic.
- No permitiendo conservar el control de `prompts` de Azure OpenAI; reutiliza prefijos idénticos (mensajes y definiciones de herramientas), no cambia la respuesta y se mide con `cached_tokens`; coloca el contenido estable al principio. La
- `cache` se maneja de `API Management` devolviendo respuestas por similitud; `adjust_score_threshold` (más bajo es más estricto), separa usuarios con `var-by-vi` y añade limitación de velocidad tras la búsqueda.
- En `A2A`, la `Agent Card` en `well-known-agent-card.json` describe habilidades, punto de conexión y `securitySchemes`; las credenciales se obtienen fuera de banda, viajan en encabezados y el transporte debe ir cifrado.

En MCP, el servidor valida que el token se emitió para él y no transmite otros tokens. En Foundry, guarda las credenciales en conexiones del proyecto y limita las herramientas con `allowed_tools`.

3. Evaluate, optimize, and monitor multi-agent solutions

20–25%

Design and implement evaluation and validation strategies for multi-agent solutions

- En Foundry, la evaluación humana usa plantillas con preguntas de pulgar arriba/abajo, control deslizante, opción múltiple o texto libre; solo una plantilla puede estar activa por agente. Los revisores valoran cada respuesta desde la aplicación web de vista previa, y los resultados se guardan en Application Insights según su retención, por lo que conviene descargarlos en CSV para conservarlos.
- Foundry ofrece evaluadores de proceso para herramientas (Tool Call Accuracy, Tool Selection, Tool Input Accuracy, Tool Output Utilization, Tool Call Success), evaluadores RAG para el conocimiento (Retrieval, Document Retrieval, Groundedness, Relevance, Response Completeness) y Task Adherence para las instrucciones. Para medir la memoria a lo largo de un diálogo se usa `evaluation_level=conversation` con datos en formato messages.

Optimize prompt and model performance for multi-agent solutions

- La orquestación simultánea reduce la duración cuando los agentes trabajan de forma independiente sobre la misma entrada, pero no conviene si necesitan contexto acumulado o si la cuota lo impide. En Azure OpenAI, el límite de solicitudes se evalúa en ventanas cortas, la estimación de tokens incluye `max_tokens` y las solicitudes fallidas cuentan; usa reintentos con retroceso exponencial y variación aleatoria.
- En Agent Framework, la compactación del historial en memoria combina estrategias de la más suave a la más agresiva: contraer resultados de herramientas, resumir con un LLM, ventana deslizante y truncamiento. La ventana deslizante olvida datos iniciales y un prompt de resumen genérico los diluye; la memoria de Foundry da continuidad entre sesiones y la búsqueda híbrida corrige la recuperación solo vectorial.
- La mejora continua en Foundry combina jueces LLM (evaluadores basados en prompt y de rúbrica), datos sintéticos (Simple QnA o Simulation seed) y conjuntos de datos creados a partir de trazas de producción. El optimizador de agentes cierra el ciclo: evalúa una línea base, genera candidatos de instrucciones, herramientas y modelo, los puntúa con el mismo conjunto de datos y promueve al mejor.

Design and implement observability and monitoring for multi-agent solutions

- El panel de supervisión de agentes de Foundry lee de Application Insights el uso de tokens, la latencia, la tasa de éxito y las evaluaciones, con alertas configurables. La evaluación continua muestrea el tráfico en vivo y la programada usa conjuntos de datos fijos para detectar deriva. Las trazas OpenTelemetry (`invoke_agent`, `plan`, `execute_tool`) correlacionan errores entre agentes y herramientas.
- Define SLO medibles con SLI y calcula el SLO compuesto de cada flujo como producto de sus dependencias críticas; un SLA es un contrato con alcance limitado, no un objetivo que copiar. Combina Service Health y Resource Health para la salud de la plataforma con pruebas de disponibilidad estándar de Application Insights, con reintentos y varias ubicaciones, para tus endpoints.
- Acota los bucles autónomos con `max_iterations` y limita la invocación automática de herramientas con `max_iterations`, `max_function_calls` y `max_duration_seconds`; al llegar al límite, el cliente pide una respuesta final. Ajusta `max_tokens` al mínimo necesario, porque cuenta para el límite de TPM, y reduce el contexto con compactación, caché de prompts o caché semántica.
- Analiza el gasto de Foundry en Cost Management por recurso y medidor, filtra por la etiqueta `project` automática para repercutir costes entre proyectos y usa presupuestos con grupos de acciones, ya que Azure OpenAI no tiene límites de gasto estrictos. La cuota se asigna en TPM por región, modelo y tipo de implementación, y `llm-emit-token-metric` mide tokens por consumidor.
- El seguimiento de Foundry está desactivado hasta conectar Application Insights; después registra en el servidor entradas, salidas, herramientas, tokens y latencias con las convenciones GenAI de OpenTelemetry (`invoke_agent`, `execute_tool`, `plan`). Agent Framework propaga traceparent a servidores MCP abiertos por el cliente. Limita el acceso a las trazas y usa alertas de búsqueda de registros.

4. Secure, govern, and deploy multi-agent solutions

20–25%

Design and implement security for multi-agent solutions

- Usa Microsoft Entra ID y RBAC de mínimo privilegio: Foundry Agent Consumer para quien solo llama a agentes y Foundry User para quien los desarrolla; las claves de API ignoran RBAC. Los agentes publicados tienen identidad propia que necesita sus roles. Aísla la red con endpoints privados para la entrada y una subred delegada para la salida de los agentes.
- En producción, autentica con Microsoft Entra ID (ámbito <https://ai.azure.com/.default>) en lugar de claves de API, que no identifican al usuario. Para actuar en nombre del usuario usa on-behalf-of, que solo lleva ámbitos delegados; para herramientas MCP o A2A elige clave, identidad de agente, identidad administrada del proyecto o paso de identidad OAuth si cada usuario debe usar sus permisos.
- Accede a Key Vault con identidades administradas y roles RBAC de mínimo privilegio, como Secrets User para leer secretos o Crypto User para usar claves. Las claves rotan con una directiva y se referencian con URI sin versión; los secretos, con Event Grid y Azure Functions; los certificados se renuevan solos. Las claves administradas por el cliente exigen eliminación temporal y protección de purga.
- El AI Red Teaming Agent usa PyRIT para simular ataques, puntúa cada par de ataque y respuesta y mide la tasa de éxito de los ataques (ASR). Desplaza las pruebas a la izquierda: elección del modelo, desarrollo, antes de implementar y ejecuciones programadas después. Los riesgos de agentes, como filtración de datos o acciones prohibidas, solo se prueban en la nube, mejor en un entorno púrpura.

Design and implement guardrails for multi-agent solutions

- En Microsoft Foundry, una barrera de protección es una colección de controles que definen el riesgo, el punto de intervención (entrada del usuario, llamada a herramienta, respuesta de la herramienta y salida) y la acción. Los puntos de llamada y respuesta de herramientas solo se aplican a agentes, y la barrera asignada al agente reemplaza por completo la del modelo; sin asignación, el agente hereda la de su implementación.
- Las restricciones propias del negocio se cubren con listas de bloqueo personalizadas (coincidencia exacta o regex, aplicadas a solicitudes o finalizaciones), el filtro de datos personales por categoría, Task Adherence para detectar llamadas a herramientas que no encajan con la intención del usuario y, en código, el middleware de Agent Framework, que puede inspeccionar argumentos de herramientas y terminar la ejecución.
- El AI Red Teaming Agent, basado en PyRIT, lanza consultas adversarias sintéticas y estrategias de ataque contra modelos o agentes y resume el resultado con la tasa de éxito de ataques (ASR). Los riesgos agénticos (acciones prohibidas, fuga de datos sensibles, adherencia a la tarea) solo se prueban en la nube con datos sintéticos y herramientas simuladas. Las evaluaciones de Foundry también pueden generar conversaciones simuladas.

Deploy multi-agent solutions to Azure

- DTAP promueve cada versión por entornos de desarrollo, pruebas, aceptación y producción. Blue/green mantiene dos entornos idénticos capaces de soportar toda la carga y cambia el tráfico al nuevo; canary entrega la versión primero a un grupo pequeño y amplía por fases con tiempos de observación de horas o días. Ambos son formas de exposición progresiva recomendadas por el Well-Architected Framework.
- Ante un fallo en el despliegue se detiene la implantación y se elige entre revertir a la última configuración buena, avanzar con una corrección o desplegar infraestructura nueva. En Azure Container Apps se revierte moviendo el peso de tráfico entre revisiones etiquetadas. En Azure Pipelines, las aprobaciones y comprobaciones de los entornos, como Query Azure Monitor Alerts, controlan cuándo avanza cada etapa.
- La lógica determinista del orquestador se valida con pruebas unitarias aisladas del modelo; las integraciones con herramientas y agentes, con dependencias simuladas y fallos inyectados. Las evaluaciones automáticas puntúan las respuestas con otro modelo y evaluadores de agente como Tool Call Accuracy o Task Completion, y en CI se comparan versiones de agente con pruebas de significación estadística para detectar regresiones.
- La infraestructura se despliega solo como código declarativo (Bicep, ARM o Terraform) mediante canalizaciones de CI/CD, con un mismo conjunto de archivos parametrizado por entorno. La operación what-if muestra los cambios previstos sin aplicarlos. La CI se ejecuta al crear una solicitud de incorporación de cambios, y las canalizaciones se autentican en Azure con Microsoft Entra ID mediante OpenID Connect.