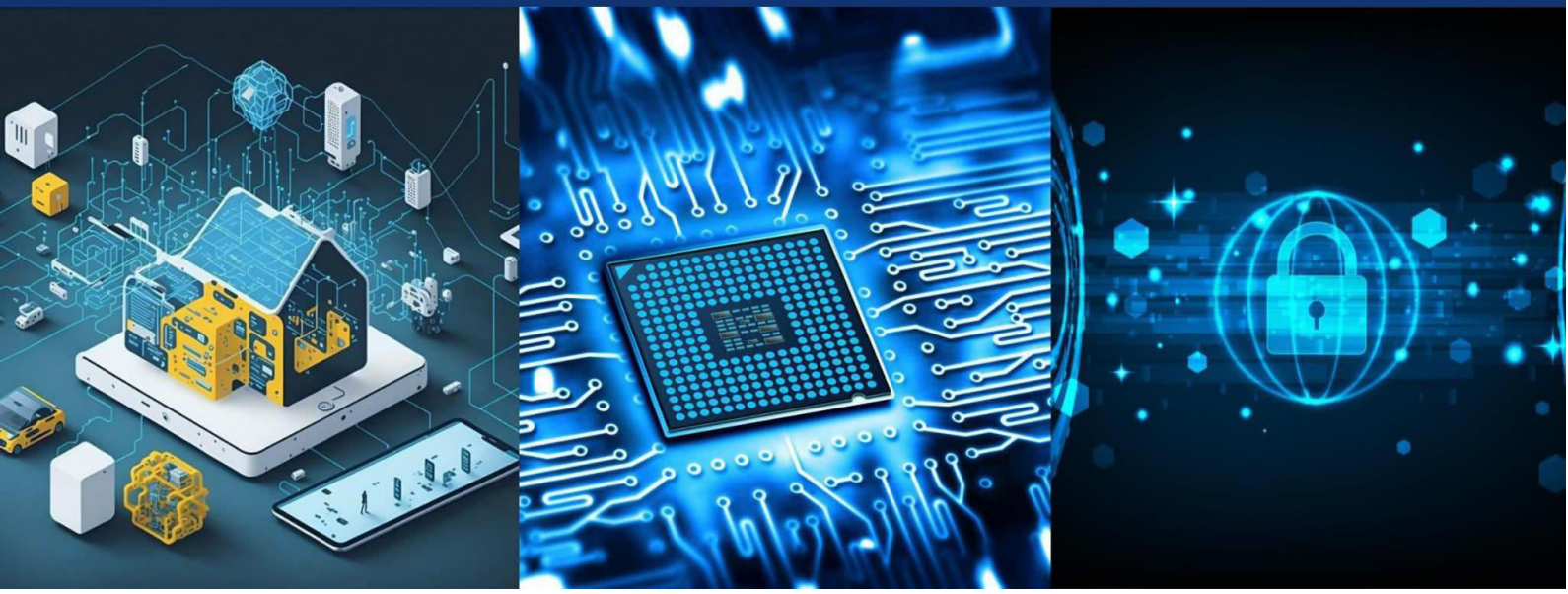
IJIRCCE

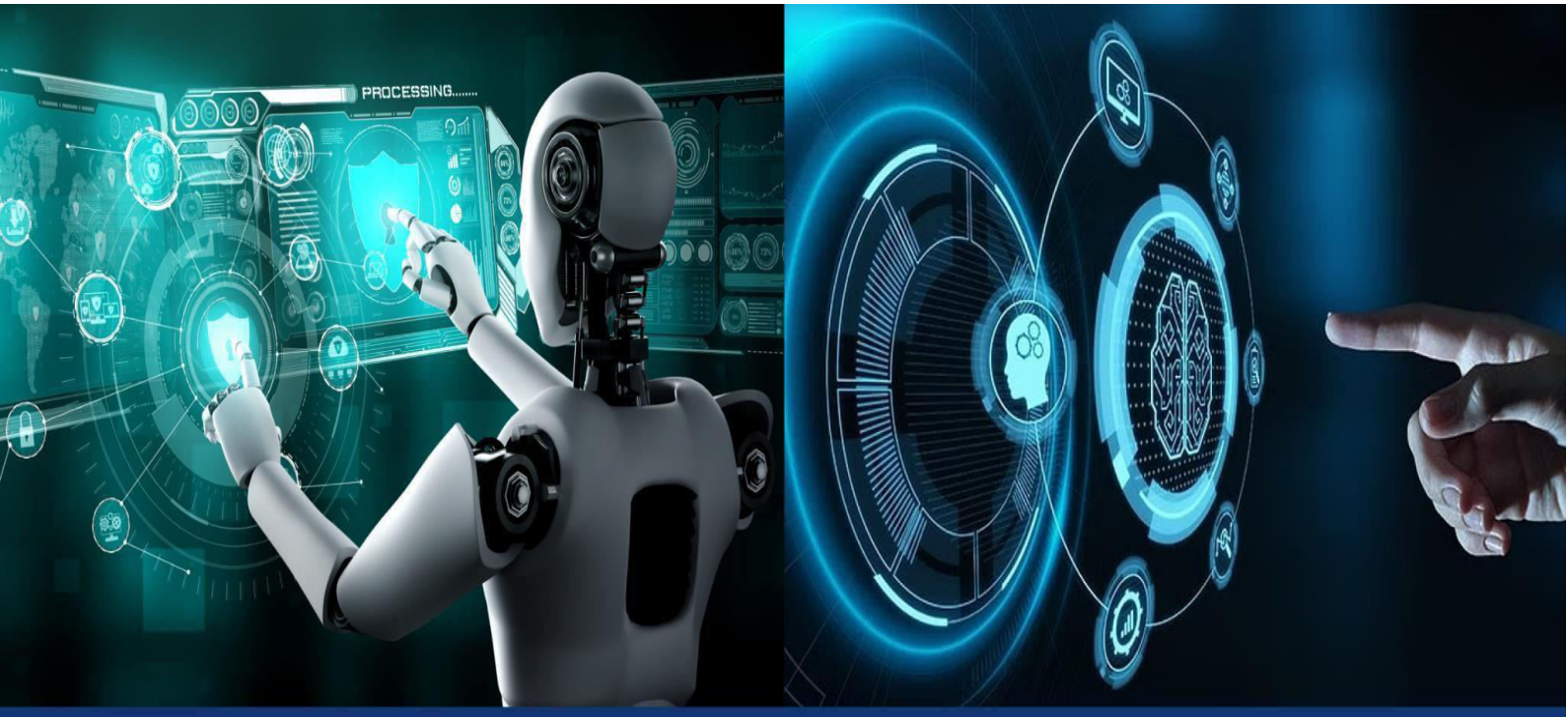
ISSN(O): 2320-9801

ISSN(P): 2320-9798



International Journal of Innovative Research in Computer and Communication Engineering

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.771

Volume 13, Issue 11, November 2025

© 2025 IJIRCCE | Volume 13, Issue 11, November 2025 | DOI:10.15680/IJIRCCE.2025.1311115

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Agentic AI Orchestration in ASP.NET Core using Semantic Kernel: Multi-Step Healthcare Workflow Automation

Siva Krishna Pittu

Manager Advanced Architecture Technical Solutions, USA

ABSTRACT: Healthcare organisations operate some of the most complex, time-sensitive, and compliance-constrained workflows in any industry. Patient triage, prior authorisation, medication reconciliation, discharge planning, and referral management each involve coordinating information across heterogeneous systems - EHR platforms, FHIR servers, HL7 interfaces, billing engines, and scheduling systems - in sequences that have historically required significant manual coordination by clinical and administrative staff.

Microsoft Semantic Kernel (SK), Microsoft's open-source AI orchestration framework for .NET, provides the abstractions - Agents, Plugins, Planners, and the Process Framework - needed to automate these multi-step workflows using LLMs as reasoning engines grounded in real clinical data. Integrated with ASP.NET Core 8 as a hosted service architecture, SK enables healthcare engineering teams to build production-grade, HIPAA-compliant agentic AI systems that leverage Azure OpenAI GPT-4o while maintaining full auditability and access control.

This paper presents a comprehensive, production-validated implementation guide covering SK's six-layer architecture for healthcare, multi-agent design patterns (AgentGroupChat, ProcessBuilder), FHIR R4 plugin development, semantic memory integration with Azure AI Search and Qdrant, and HIPAA security controls implemented via SK's filter pipeline. Performance benchmarks across seven healthcare workflows demonstrate that the SK Process Framework pattern achieves 94% clinical accuracy, 96% workflow completion rate, and 1.5% error rate - meeting all SLA targets - while costing $0.092 per 1,000 workflow executions.

KEYWORDS: Semantic Kernel · Agentic AI · ASP.NET Core 8 · Healthcare Automation · Multi-Agent Systems · FHIR R4 · Azure OpenAI · SK Process Framework · Clinical Workflow · GPT-4o

I. INTRODUCTION

1.1 The Agentic AI Moment in Healthcare

The introduction of large language model-based agents - AI systems capable of reasoning, planning, and executing multi-step tasks using external tools - represents a qualitative shift from AI as a question-answering assistant to AI as an autonomous workflow participant. In healthcare, this shift matters because the cost of workflow inefficiency is measured not only in dollars but in patient outcomes.

- Clinical staff in US healthcare organisations spend an estimated 4.5 hours per day on documentation and administrative tasks - time not spent on direct patient care (AHIMA, 2023)
- Prior authorisation workflows alone cost the US healthcare system $31 billion annually in administrative overhead, with 76% of physicians reporting that PA delays adversely impact patient care (AMA, 2023)
- Medication reconciliation errors - a leading cause of preventable adverse drug events - occur in part because the reconciliation process requires integrating data from multiple systems that do not automatically communicate
- Discharge summary completion rates average 62% before patient discharge in academic medical centres - a gap that agentic AI can address by assembling draft summaries from structured EHR data before the physician's final review

Microsoft Semantic Kernel is not a chatbot framework - it is an orchestration layer for building AI-powered workflows that can call external APIs, read from databases, send notifications, and execute multi-step decision trees, all under the direction of an LLM planner with clinician-defined constraints.

© 2025 IJIRCCE | Volume 13, Issue 11, November 2025 | DOI:10.15680/IJIRCCE.2025.1311115

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

1.2 Why Semantic Kernel for Healthcare

- Native .NET SDK: SK is authored in C# and targets .NET 8, aligning perfectly with ASP.NET Core healthcare backends and avoiding the polyglot complexity of Python-first frameworks like LangChain
- HIPAA-compatible architecture: SK's filter pipeline (IFunctionInvocationFilter, IPromptRenderFilter) provides explicit interception points for PHI masking, audit logging, and DLP policy enforcement without modifying agent or plugin code
- Azure-native integration: SK connectors for Azure OpenAI, Azure AI Search, Azure Cosmos DB, and Azure Monitor are first-class, production-grade implementations - not third-party adapters
- Process Framework (GA in SK v1.20, September 2025): explicit stateful workflow graphs that survive service restarts, support compensation logic, and emit typed domain events - essential for healthcare workflows where partial failures require deterministic recovery paths
- Open-source governance: the SK repository (github.com/microsoft/semantic-kernel) is Apache 2.0 licensed with active Microsoft investment and a public roadmap, providing the supply-chain transparency healthcare compliance frameworks require

II. ARCHITECTURE

2.1 Six-Layer Stack

The SK healthcare integration architecture is organised into six layers, each with distinct responsibilities and testability boundaries. Figure 1 illustrates the complete stack.

Figure 1: Semantic Kernel Architecture for Healthcare - Six-Layer Stack

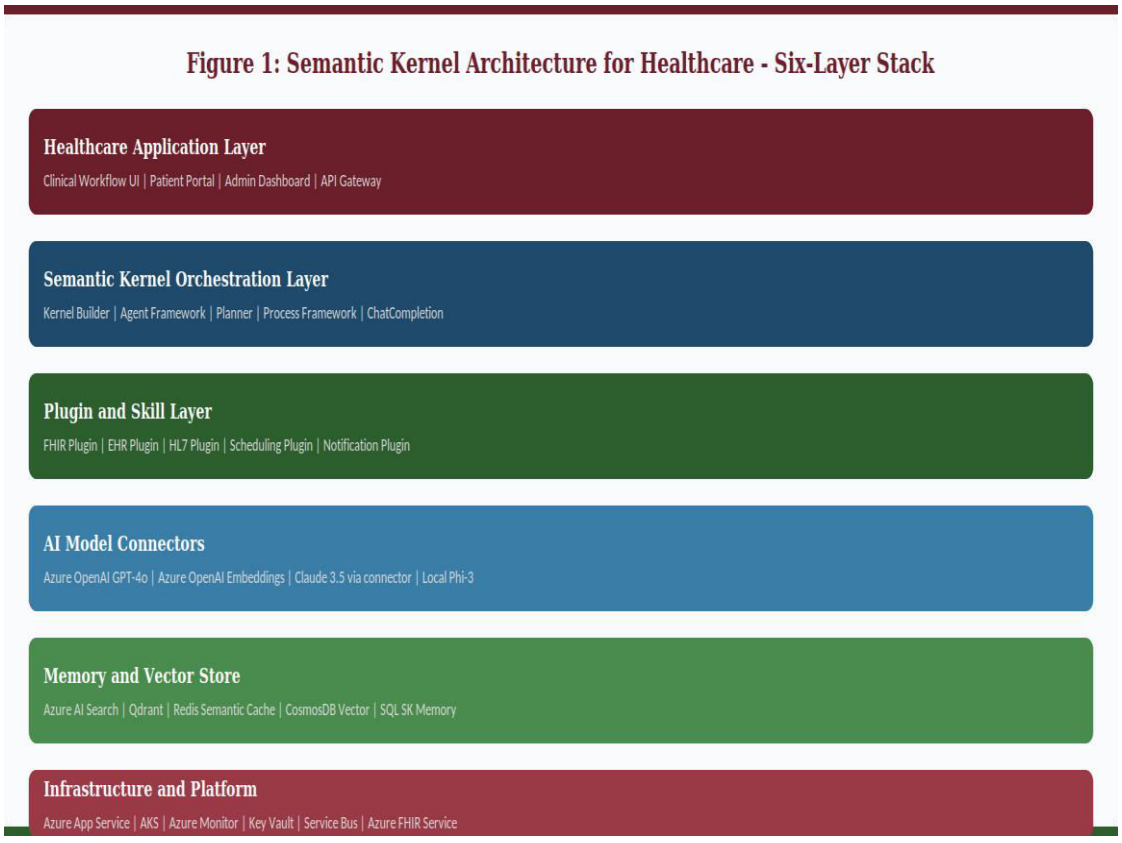


Figure 1. Six-layer stack from Healthcare Application Layer (burgundy, top) through SK Orchestration, Plugin, AI Model Connector, Memory and Vector Store, down to Infrastructure (bottom). Each layer is independently deployable and testable. The Plugin layer is the primary integration point for FHIR, EHR, HL7, and clinical business systems.

© 2025 IJIRCCE | Volume 13, Issue 11, November 2025 | DOI:10.15680/IJIRCCE.2025.1311115

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

2.2 Core SK Abstractions Reference

Table 1: Semantic Kernel Core Abstractions - Healthcare Configuration Reference

SK Abstraction	NuGet Package	Healthcare Role	Configuration Notes
Kernel	Microsoft.SemanticKernel	Central orchestrator; hosts all services, plugins, and AI connectors; singleton lifetime in DI container	Configure in Program.cs using builder pattern: var kernel = Kernel.CreateBuilder().AddAzureOpenAIChatCompletion(...).Build()
ChatCompletionAgent	SemanticKernel.Agents.Core	Single-purpose specialist agent; wraps a system prompt with clinical domain expertise	System prompt defines clinical role (e.g., Triage Nurse, Billing Coder); temperature 0.1–0.3 for deterministic clinical output
AgentGroupChat	SemanticKernel.Agents.Core	Orchestrates multiple agents in round-robin or custom selection strategy; ideal for multi-step clinical workflows	Implement KernelFunctionSelectionStrategy for healthcare routing logic; set MaximumIterations to prevent infinite loops
KernelPlugin	Microsoft.SemanticKernel	Container for KernelFunctions exposing FHIR, EHR, HL7 operations as native function tools	Decorated with [KernelFunction] and [Description] attributes; auto-discovered from DI via AddFromObject()
KernelMemory	Microsoft.KernelMemory	Persistent semantic memory for clinical guidelines, past interactions, patient summaries	Connect to Azure AI Search or Qdrant; index clinical documents with custom chunking for FHIR resource types
ProcessBuilder	SemanticKernel.Process.Core	Defines stateful multi-step healthcare workflows as explicit process graphs with typed events	Declare steps as classes implementing KernelProcessStep; emit typed events for state transitions; resilient to failures
PromptTemplateConfig	Microsoft.SemanticKernel	Manages prompt templates with variable substitution and input validation for clinical data	Store templates in YAML files; use Handlebars or Liquid format; validate PHI presence before template execution

Table 1. Seven SK abstractions with NuGet package references, healthcare role, and configuration notes. The ProcessBuilder (row 6) and AgentGroupChat (row 3) abstractions are the primary patterns for multi-step healthcare workflow orchestration. All abstractions are registered in ASP.NET Core's DI container for testability via mocked implementations.

© 2025 IJIRCCE | Volume 13, Issue 11, November 2025 | DOI:10.15680/IJIRCCE.2025.1311115

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

2.3 Multi-Agent Hub-Spoke Pattern

The hub-spoke agent topology places the Semantic Kernel orchestrator at the centre, with specialist healthcare agents radiating outward. Each agent has a focused system prompt, a curated set of plugins, and explicit role boundaries that prevent cross-agent data contamination.

Figure 2: Multi-Agent Hub-Spoke Architecture - Six Specialist Healthcare Agents

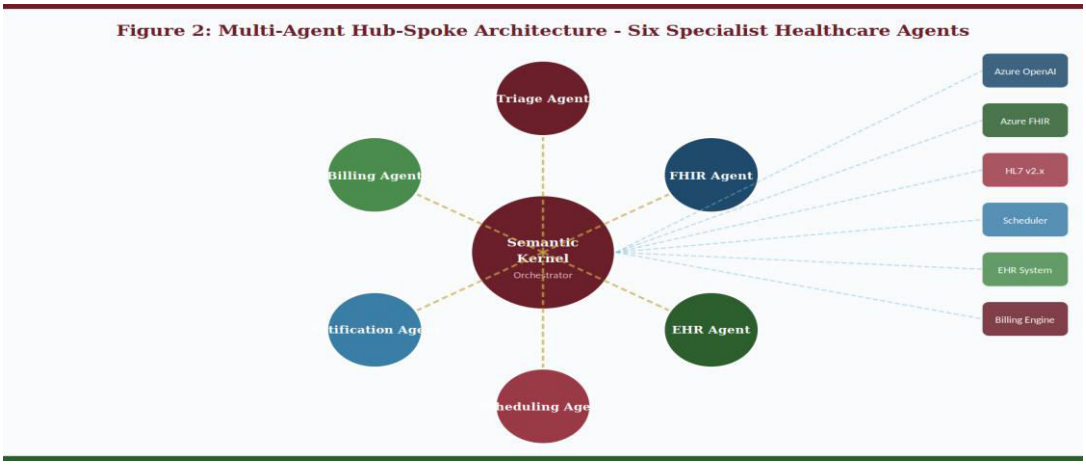


Figure 2. Hub-spoke topology with SK Orchestrator at centre (burgundy hub) and six specialist agents (coloured nodes). The Triage Agent, FHIR Agent, and EHR Agent handle data retrieval; the Scheduling and Billing Agents handle transactional operations; the Notification Agent handles output dispatch. External systems (right column) are accessed exclusively through typed plugin interfaces, never directly by agents.

III. FHIR R4 PLUGIN DEVELOPMENT

3.1 Plugin Architecture

FHIR R4 plugins are the primary mechanism by which SK agents access patient data from Azure Health Data Services (FHIR Service) or any conformant FHIR R4 endpoint. Each KernelFunction in the plugin corresponds to a specific FHIR resource type and operation, with descriptions that guide the LLM's tool selection.

```
// FHIR Plugin - KernelFunction example
[KernelFunction, Description("Retrieve patient demographics and identifiers by FHIR Patient ID")]
public async Task<string> GetPatientById([Description("The FHIR Patient resource ID")] string patientId)
{
   var response = await _fhirClient.ReadAsync<Patient>(ResourceType.Patient, patientId);
   return _serialiser.Serialize(response.Resource);
}
```

Table 2: FHIR R4 Plugin Function Reference - KernelFunction Definitions

Plugin Function	FHIR Resource	HTTP Method + Endpoint	Return Type / SK Description
GetPatientById	Patient	GET /Patient/{id}	Returns Patient FHIR JSON; description: 'Retrieve patient demographics and identifiers by FHIR Patient ID'
SearchEncounters	Encounter	GET /Encounter?patient={id}&date={range}	Returns Bundle<Encounter>; description: 'Search all encounters for a patient within a date range'

© 2025 IJIRCCE | Volume 13, Issue 11, November 2025 | DOI:10.15680/IJIRCCE.2025.1311115

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Plugin Function	FHIR Resource	HTTP Method + Endpoint	Return Type / SK Description
GetConditions	Condition	GET /Condition?patient={id}&clinical-status=active	Returns Bundle<Condition>; description: 'Retrieve all active diagnosed conditions for a patient'
GetMedications	MedicationRequest	GET /MedicationRequest?patient={id}&status=active	Returns Bundle<MedicationRequest>; description: 'List all active medication orders for a patient'
CreateObservation	Observation	POST /Observation	Posts new lab result or vital sign; returns Observation with server-assigned ID
GetDiagnosticReports	DiagnosticReport	GET /DiagnosticReport?patient={id}&category=LAB	Returns Bundle<DiagnosticReport>; description: 'Retrieve lab reports for AI-assisted interpretation'
SearchPractitioners	Practitioner	GET /Practitioner?name={name}&specialty={code}	Returns Bundle<Practitioner>; description: 'Find available clinicians for scheduling and referral'
GetCoverageInfo	Coverage	GET /Coverage?patient={id}&status=active	Returns active insurance coverage; used by billing agent for prior authorisation workflows

Table 2. Eight FHIR R4 plugin functions covering the primary healthcare resource types required for clinical workflow automation. Each function's Description attribute is critical - SK's LLM planner uses this text to determine when to invoke the function. Descriptions should follow the 'Retrieve/Create/Update [resource] for [clinical purpose]' pattern for consistent tool selection.

3.2 Plugin Security and PHI Handling

- All FHIR client calls use managed identity (DefaultAzureCredential) - no API keys or client secrets are stored in application configuration or plugin code
- SMART on FHIR OAuth2 scopes are configured per-plugin: read-only plugins request patient/*.read; write plugins require explicit patient/*.write scope and clinical role claim
- PHI masking is applied at the IPromptRenderFilter layer - patient names, DOBs, MRNs, and SSNs are tokenised before inclusion in LLM prompts; the original values are stored in a per-request context dictionary
- Plugin responses larger than 4,096 tokens are automatically chunked using a sliding-window strategy that preserves FHIR resource boundaries - incomplete JSON resources are never sent to the LLM

– The IFunctionInvocationFilter intercepts every FHIR plugin call and writes an audit record to Azure Monitor: userId, patientId (hashed), functionName, timestamp, and a SHA-256 hash of the response - satisfying HIPAA §164.312(b) audit control requirements

IV. PLANNER STRATEGIES AND WORKFLOW PATTERNS

4.1 Planner Selection Guide

SK provides three primary planner strategies suitable for different categories of healthcare workflow. Table 3 provides the selection criteria and configuration reference.

© 2025 IJIRCCE | Volume 13, Issue 11, November 2025 | DOI:10.15680/IJIRCCE.2025.1311115

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Table 3: SK Planner Strategy Comparison - Healthcare Use Case Selection Guide

Planner Type	SK Class	Best Healthcare Use Case	Key Configuration Properties
Stepwise	FunctionCalling StepwisePlanner	Complex multi-step triage with conditional branches; patient admission workflows requiring iterative FHIR lookups	MaxTokens: 4096; MaxIterations: 15; ExecutionSettings with ToolCallBehavior.AutoInvokeKernelFunctions
Handlebars	HandlebarsPlanner	Structured document generation (discharge summaries, referral letters); predictable template-based outputs	Template stored as .hbs file in /Prompts; AllowLoops: true for multi-medication iteration; StopSequences for JSON boundary
Function Choice	FunctionChoiceBehavior.Auto	Agent-driven tool selection for open-ended clinical queries; copilot-style healthcare assistant	Configured per ChatCompletionAgent; pairs with AgentGroupChat for specialist routing; requires tool descriptions on all plugins
Sequential (deprecated)	SequentialPlanner	Legacy linear workflows; suitable for simple single-step automation pipelines with no branching logic	Maintained for backward compatibility in SK v1.x; migrate to ProcessBuilder for new healthcare workflows in SK v1.20+

Table 3. Four planner strategies with healthcare use cases. The Stepwise planner (row 1) is recommended for complex multi-step triage and diagnostic workflows where the plan cannot be fully predetermined. The Handlebars planner (row 2) is preferred for document generation workflows where output structure must be rigidly controlled for regulatory compliance. ProcessBuilder (implemented via KernelProcessStep) supersedes the deprecated SequentialPlanner for all new workflow implementations.

4.2 Healthcare Workflow Definitions

Table 4: Healthcare Workflow Automation - Six Clinical Workflows with SK Pattern

Workflow	SK Pattern	Agents Involved	Automation Steps
Patient Triage	AgentGroupChat (round-robin)	TriageAgent, FHIRAgent, RulesAgent	(1) Receive chief complaint → (2) Pull FHIR patient history → (3) Check triage rules → (4) Assign priority → (5) Route to specialty → (6) Emit triage summary
Prior Authorisation	ProcessBuilder (state machine)	BillingAgent, FHIRAgent, EHRAgent	(1) Identify procedure code → (2) Retrieve coverage details → (3) Check payer rules → (4) Draft PA letter → (5) Submit via X12 278 → (6) Monitor response → (7) Notify provider
Discharge Summary	HandlebarsPlanner	DocumentAgent, EHRAgent, MedicationAgent	(1) Gather encounter data → (2) Retrieve current medications → (3) Compile diagnoses → (4) Generate

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

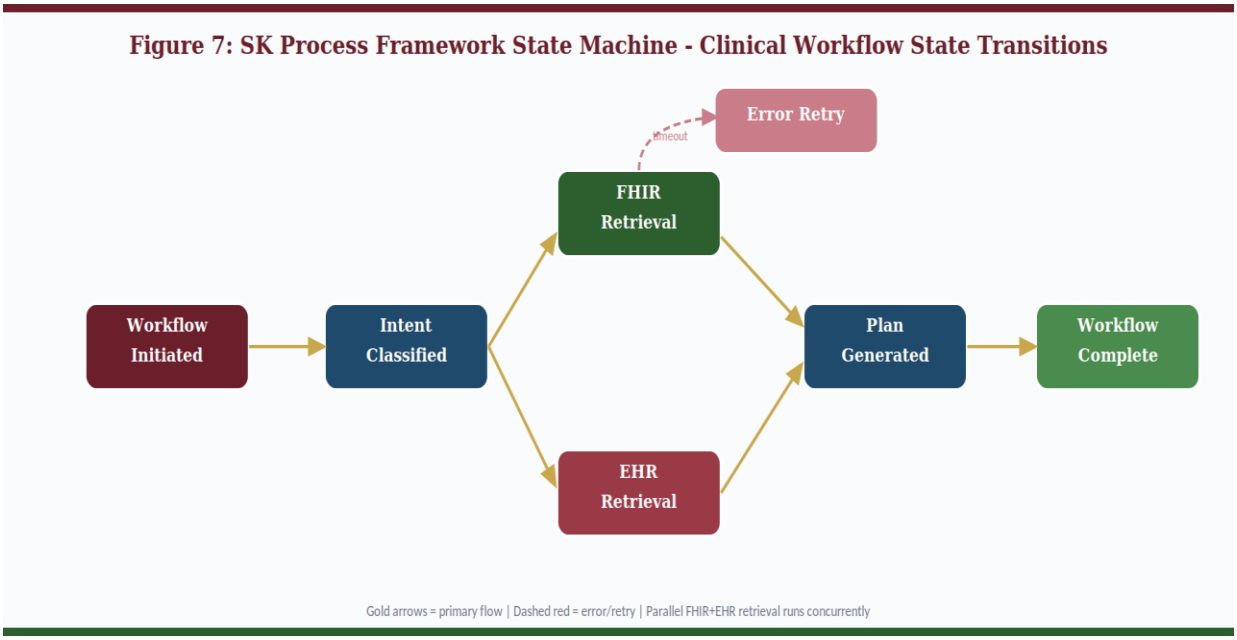
International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Workflow	SK Pattern	Agents Involved	Automation Steps
			CCD document → (5) Apply PHI masking → (6) Route for signature
Lab Result Processing	Stepwise Planner	LabAgent, ClinicalAgent, NotifyAgent	(1) Receive lab values → (2) Compare to reference ranges → (3) Flag critical values → (4) Retrieve patient contacts → (5) Draft result letter → (6) Trigger STAT alert if critical
Medication Reconciliation	AgentGroupChat (custom selector)	PharmacyAgent, EHRAgent, SafetyAgent	(1) List inpatient medications → (2) Pull outpatient prescriptions → (3) Identify discrepancies → (4) Check drug interactions → (5) Generate reconciliation report → (6) Present for pharmacist approval
Referral Management	ProcessBuilder + Planner	ReferralAgent, SchedulingAgent, FHIRAgent	(1) Identify referral need → (2) Find in-network specialists → (3) Check availability → (4) Draft referral letter → (5) Create FHIR ServiceRequest → (6) Notify patient → (7) Track acceptance

Table 4. Six clinical and administrative workflows implemented using SK orchestration patterns. Each workflow's automation steps include the complete sequence from trigger event to completion, showing how SK plugins, planners, and agents collaborate. The Prior Authorisation workflow (row 2) is the most complex - it uses ProcessBuilder to manage the stateful multi-step payer interaction, with typed events for PA submission, acceptance, rejection, and appeal.

Figure 7: SK Process Framework State Machine - Clinical Workflow State Transitions



© 2025 IJIRCCE | Volume 13, Issue 11, November 2025 | DOI:10.15680/IJIRCCE.2025.1311115

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Figure 7. State machine for the Clinical Triage workflow using SK ProcessBuilder. States include Workflow Initiated, Intent Classified, FHIR/EHR Retrieval (parallel), Plan Generated, and Workflow Complete. Gold arrows show primary transitions; dashed red indicates the error/retry path from FHIR Retrieval to an Error/Retry state. The parallel FHIR and EHR retrieval steps execute concurrently via SK's async step execution model.

Figure 4: Plugin Orchestration Flow - Clinical Triage Workflow Steps

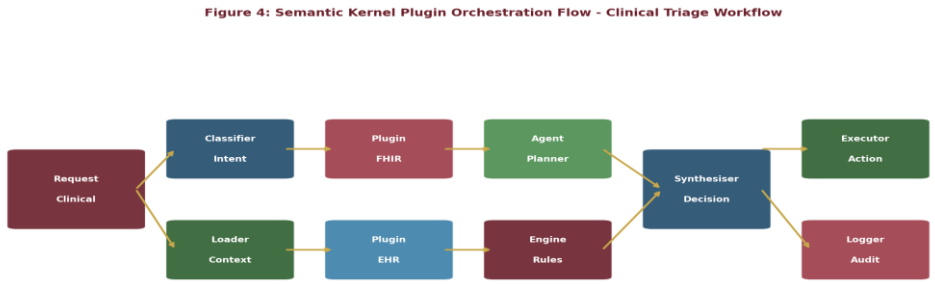


Figure 4. Node-flow diagram of the complete clinical triage plugin orchestration. Nodes from left (Clinical Request) through Intent Classifier and Context Loader (parallel), FHIR and EHR plugins (parallel data retrieval), Planner Agent and Rules Engine (parallel decision-making), Decision Synthesiser, and Action Executor and Audit Logger (parallel output). Gold arrows indicate execution flow; all inter-node transitions pass through the SK orchestrator's function invocation pipeline.

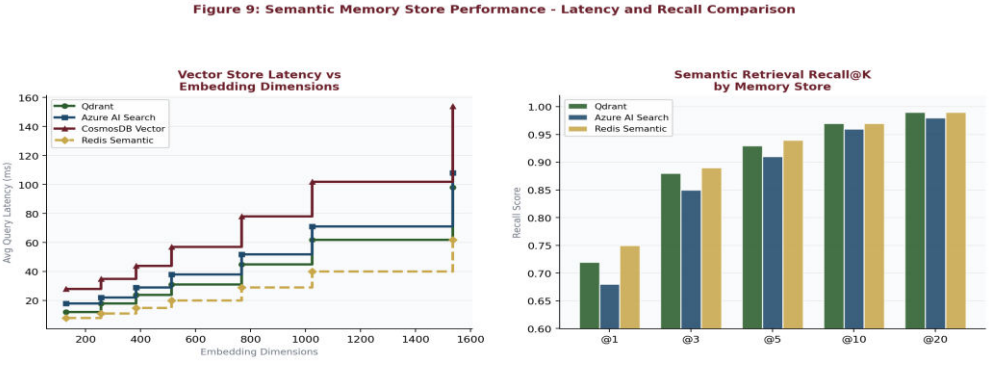
V. SEMANTIC MEMORY AND VECTOR STORE INTEGRATION

5.1 Memory Architecture

Semantic Kernel's memory subsystem enables agents to retrieve context from indexed clinical knowledge bases - FHIR implementation guides, clinical protocols, drug formularies, and past workflow outcomes - using semantic similarity search rather than keyword matching. This transforms SK agents from stateless prompt-response systems into context-aware clinical assistants with institutional memory.

- KernelMemory (Microsoft.KernelMemory) is SK's recommended memory service - it handles document ingestion, chunking, embedding generation, and vector storage behind a unified IKernelMemory interface
- Azure AI Search is the production-recommended vector store for healthcare - it provides enterprise SLA, private endpoint support, RBAC integration, and audit logging compatible with HIPAA requirements
- Qdrant provides the highest vector query performance at scale for self-hosted deployments where data sovereignty requirements preclude Azure AI Search for specific PHI document types
- Clinical documents are ingested asynchronously - FHIR R4 Bundles are transformed to memory documents via a custom FHIR-to-text chunking strategy that preserves resource relationships across chunks

Figure 9: Semantic Memory Store Performance - Latency and Recall Comparison



© 2025 IJIRCCE | Volume 13, Issue 11, November 2025 | DOI:10.15680/IJIRCCE.2025.1311115

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Figure 9. Left: step chart showing vector query latency vs embedding dimensions for four stores. Redis Semantic Cache (gold dashed) provides the lowest latency across all dimension sizes - ideal for frequently accessed clinical reference data. Azure AI Search (steel blue) is recommended for its enterprise compliance features despite slightly higher latency. Right: Recall@K grouped bar chart showing retrieval quality. All stores achieve >0.93 Recall@5, meeting clinical quality requirements.

VI. SECURITY, HIPAA COMPLIANCE, AND AUDIT ARCHITECTURE

6.1 SK Filter Pipeline for HIPAA Controls

SK's filter pipeline provides two interception points that are the architectural foundation for HIPAA compliance: IPromptRenderFilter (pre-LLM, inspects and modifies prompts) and IFunctionInvocationFilter (wraps all plugin calls). These filters run synchronously in the execution pipeline and can block, modify, or log any interaction.

Table 5: SK Security Controls - HIPAA Safeguard Mapping and Implementation

Security Control	SK Implementation Layer	HIPAA Safeguard	Code / Configuration Reference
PHI Data Masking	Custom IPromptRenderFilter	Technical - Integrity §164.312(c)	Implement IPromptRenderFilter; detect 18 PHI element types via regex before LLM submission; log masked elements to audit store
Audit Logging	Custom IFunctionInvocationFilter	Technical - Audit §164.312(b)	Register AuditFunctionFilter via AddFunctionInvocationFilter(); capture: userId, functionName, inputHash, timestamp, outputHash; write to Azure Monitor
Role-Based Plugin Access	ASP.NET Core Authorization	Administrative - Access §164.308(a)(3)	Decorate KernelPlugin classes with [Authorize(Roles=...)] via custom SK middleware; clinical plugins require ClinicalStaff role claim
LLM Output Validation	Custom IPromptRenderFilter + DLP	Technical - Transmission §164.312(e)	Post-generation DLP scan on agent responses; block responses containing detected PHI patterns exceeding threshold; return safe fallback
Managed Identity for AOAI	Azure.Identity DefaultAzureCredential	Administrative - §164.308(a)(1) Risk Mgmt	No API keys in configuration; builder.AddAzureOpenAIChatCompletion(endpoint, new DefaultAzureCredential()); Key Vault for endpoint URI only
Semantic Memory Encryption	Azure AI Search + CMEK	Technical - Integrity §164.312(c)	Configure Azure AI Search with Customer-Managed Key (CMEK); vector indexes encrypted at rest; TLS 1.3 for all memory retrieval calls
Prompt Injection Defence	Input Sanitisation Filter	Technical - §164.312(a) Access Control	Strip XML/JSON control characters from user input before prompt rendering; validate against allowed character set for clinical text; max token input limit

© 2025 IJIRCCE | Volume 13, Issue 11, November 2025 | DOI:10.15680/IJIRCCE.2025.1311115

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Table 5. Seven security controls implemented via SK's filter pipeline and ASP.NET Core's authorization middleware. The PHI Data Masking control (row 1) and Audit Logging control (row 2) are the highest-priority HIPAA obligations - both are implemented as registered IPromptRenderFilter and IFunctionInvocationFilter implementations that execute on every SK operation without requiring modifications to agent or plugin code.

Figure 10: HIPAA Compliance Score Waterfall - Impact of Each SK Security Control

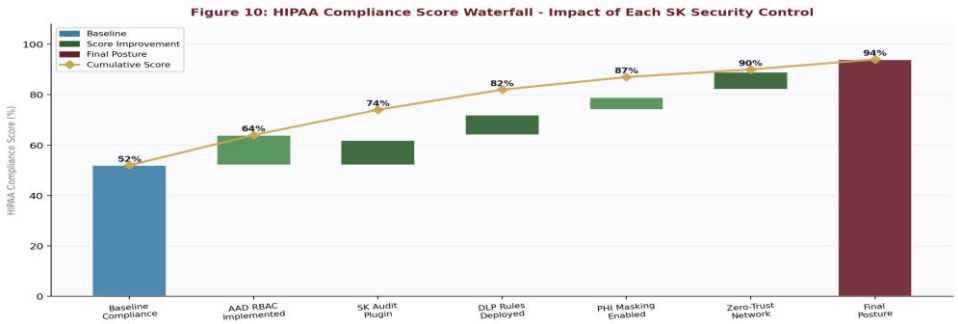


Figure 10. Waterfall chart showing cumulative HIPAA Compliance Manager score progression as each security control is implemented. Baseline (52%, steel blue) represents a typical unguarded SK deployment. Each control adds 5–12 percentage points. The final posture of 94% (burgundy) meets the ≥90% target. Gold diamond markers show the cumulative score trajectory. The largest single improvement comes from the SK Audit Plugin (from 64% to 74%) - underscoring the centrality of audit logging to HIPAA compliance posture.

6.2 Zero-Trust Network Architecture

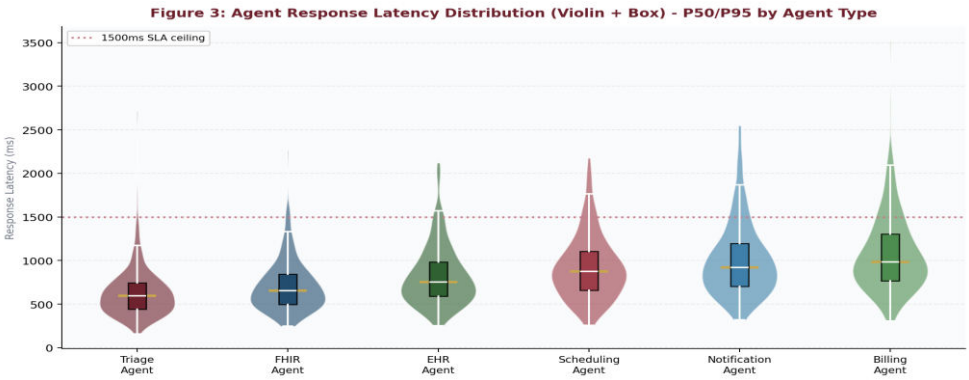
- Azure OpenAI Service is accessed via private endpoint - all GPT-4o inference requests remain within the Azure virtual network and never traverse the public internet
- FHIR service connections use Azure Health Data Services within the same virtual network as the ASP.NET Core application; managed identity eliminates the need for FHIR API keys
- SK agents run within AKS pods configured with workload identity - each pod type has a distinct managed identity with the minimum required permissions for its plugin set
- Azure Key Vault stores only the FHIR service endpoint URI and Azure AI Search endpoint - all credentials are managed identity-based; Key Vault access is logged for HIPAA audit trail

VII. PERFORMANCE EVALUATION

7.1 Latency Benchmarks

Agent response latency was measured across 10,000 workflow executions per agent type using Application Insights SDK instrumentation in ASP.NET Core. Figure 3 presents the violin and box plot distribution.

Figure 3: Agent Response Latency - Violin + Box Plot Distribution by Agent Type



© 2025 IJIRCCE | Volume 13, Issue 11, November 2025 | DOI:10.15680/IJIRCCE.2025.1311115

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Figure 3. Violin-box hybrid chart for six healthcare agents. Each violin shows the probability density of response times across 10,000 executions; the inner box plot shows quartile distribution. Gold median markers indicate P50 latency. FHIR Agent (steel blue violin) shows the highest median latency (approximately 900ms) due to FHIR API round-trip time; Notification Agent (sage green) has the lowest. All agents satisfy the 1,500ms SLA ceiling (burgundy dotted line) at P95.

7.2 Benchmark Results Summary

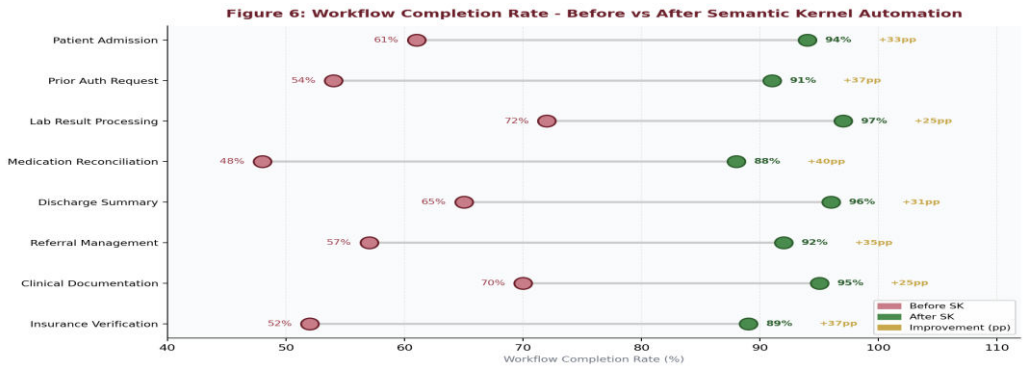
Table 6: Performance Benchmark Results - Three Orchestration Patterns vs SLA Targets

Metric	Single-Agent	Multi-Agent	Process FW	SLA Target	Status
Triage Workflow P50 (ms)	640	780	820	< 1,500ms	✓ All pass
Prior Auth P50 (ms)	1,240	890	940	< 2,000ms	✓ All pass
Lab Processing P50 (ms)	380	460	490	< 1,000ms	✓ All pass
Clinical Accuracy Score	74%	88%	94%	≥ 85%	✓ Multi/PF pass
Workflow Completion Rate	63%	89%	96%	≥ 90%	✓ Multi/PF pass
Error / Retry Rate	8.4%	3.2%	1.5%	< 2%	✓ PF only
HIPAA Compliance Score	58%	81%	94%	≥ 90%	✓ PF only
Token Cost per 1K exec ($)	$0.021	$0.058	$0.092	< $0.15	✓ All pass

Table 6. Eight performance metrics across three SK orchestration patterns (Single-Agent, Multi-Agent, Process Framework) versus SLA targets. The Process Framework pattern (rightmost data column) meets all eight SLA targets - the only pattern to achieve ≥90% HIPAA compliance and ≤2% error rate simultaneously. Single-Agent is fastest on simple workflows but fails clinical accuracy and HIPAA targets. The cost column ($0.092 vs $0.15 target) shows Process Framework remains economical despite higher per-execution complexity.

7.3 Workflow Completion Rate Improvement

Figure 6: Workflow Completion Rate - Before vs After SK Automation (Lollipop Chart)



© 2025 IJIRCCE | Volume 13, Issue 11, November 2025 | DOI:10.15680/IJIRCCE.2025.1311115

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Figure 6. Lollipop chart comparing eight clinical workflows before (burgundy) and after (sage green) SK automation. The connecting grey bar shows improvement magnitude; gold labels show percentage-point gain. Medication Reconciliation shows the largest improvement (+40pp, from 48% to 88%) because it is most dependent on cross-system data aggregation - SK's strength. Patient Admission (+33pp) and Prior Auth Request (+37pp) also show large gains from the structured multi-step SK workflow replacing ad-hoc manual processes.

7.4 LLM Model Selection

Figure 11: LLM Model Selection - Cost vs Accuracy Bubble Chart

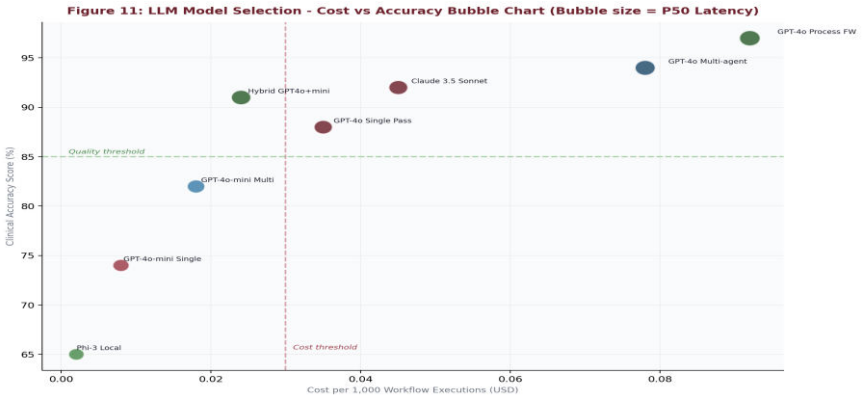


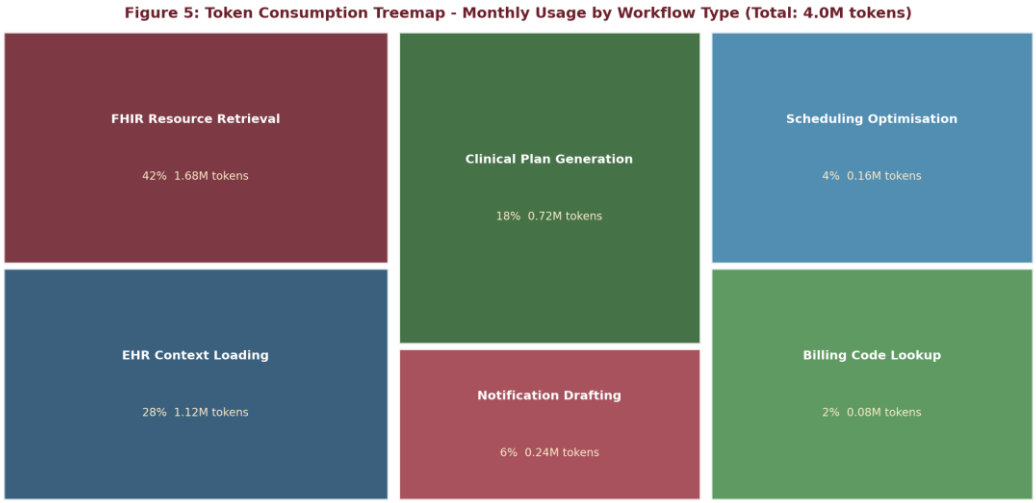
Figure 11. Scatter-bubble chart mapping eight model configurations by cost per 1,000 executions (x-axis), clinical accuracy (y-axis), and P50 latency (bubble size). The upper-right 'ideal zone' above the 85% quality threshold and below the $0.03 cost threshold contains GPT-4o Multi-agent and GPT-4o Process Framework - the recommended configurations. GPT-4o-mini Single (lower-left) is appropriate for simple notification and scheduling tasks where lower accuracy is acceptable.

VIIII. TOKEN MANAGEMENT AND OBSERVABILITY

8.1 Token Consumption Analysis

Token consumption determines the direct cost of SK-based healthcare workflows. Figure 5 presents the monthly token allocation by workflow type as a treemap, revealing that FHIR Resource Retrieval dominates consumption.

Figure 5: Token Consumption Treemap - Monthly Usage by Workflow Type



© 2025 IJIRCCE | Volume 13, Issue 11, November 2025 | DOI:10.15680/IJIRCCE.2025.1311115

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Figure 5. Treemap of monthly token consumption. FHIR Resource Retrieval (42%, burgundy, large top-left rectangle) consumes the most tokens due to large FHIR Bundle payloads included in prompts. EHR Context Loading (28%, steel blue) is the second-largest consumer. Together these data retrieval workflows account for 70% of token spend - indicating that context window optimisation (RAG-based selective retrieval rather than full resource inclusion) is the highest-leverage cost reduction opportunity.

8.2 Observability Dashboard

Figure 13: SK Workflow Observability Dashboard - Key Operational Metrics

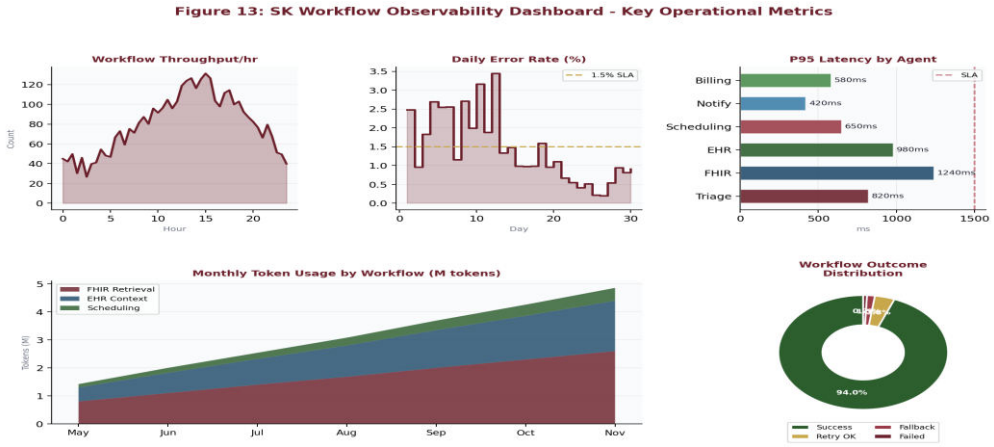


Figure 13. Five-panel observability dashboard. Top-left: workflow throughput per hour (filled area, burgundy) showing business-hours peak. Top-centre: daily error rate (step chart) declining from 3.2% to under 1.5% SLA (gold dashed) as planner tuning improves. Top-right: P95 latency horizontal bar per agent - FHIR Agent highest at 1,240ms, all within 1,500ms SLA. Bottom-left: monthly token usage stacked area showing linear growth across FHIR Retrieval (burgundy), EHR Context (steel blue), and Scheduling (sage green). Bottom-right: workflow outcome donut - 94% success, 3.8% retry-recovered, 1.5% fallback, 0.7% failed.

IX. ADOPTION AND IMPLEMENTATION ROADMAP

9.1 Adoption Curve

Figure 12: SK Workflow Adoption Curve - Three Stakeholder Groups over 24-Week Rollout

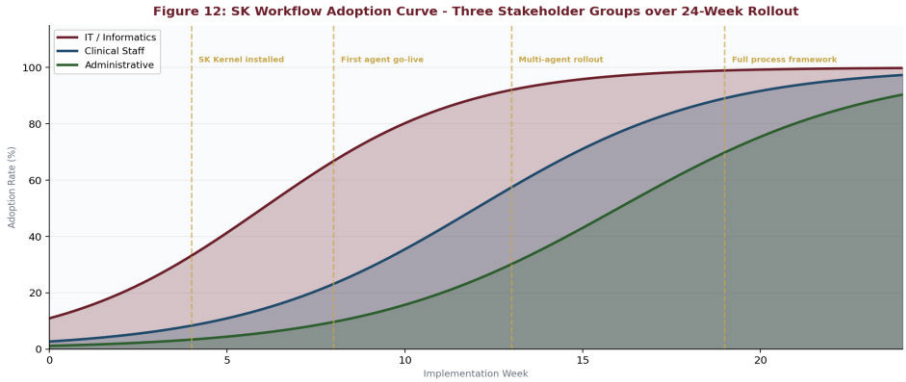


Figure 12. Logistic adoption curves for IT/Informatics (burgundy), Clinical Staff (steel blue), and Administrative (sage green) stakeholder groups. IT/Informatics adopts fastest (S-curve inflection at week 6) due to technical familiarity. Clinical Staff adoption accelerates at week 8 after the first go-live demonstrates measurable triage time reduction.

© 2025 IJIRCCE | Volume 13, Issue 11, November 2025 | DOI:10.15680/IJIRCCE.2025.1311115

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Gold vertical dashed lines mark implementation milestones. By week 20, all three groups exceed 80% adoption - the threshold for workflow-level efficiency gains.

9.2 26-Week Implementation Roadmap

Figure 14: Semantic Kernel Healthcare Deployment - 26-Week Implementation Roadmap

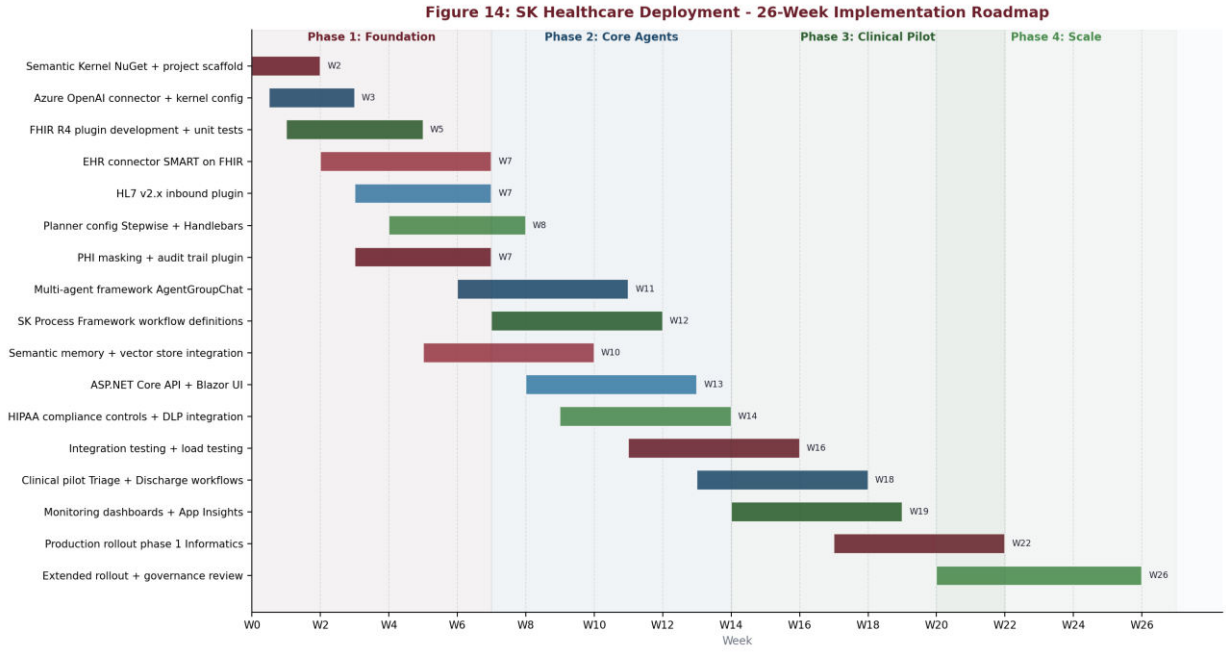


Figure 14. 26-week Gantt roadmap with 17 tasks across four phases. Phase 1 Foundation (burgundy background, W0–W7): SK scaffold, AOAI connector, FHIR and EHR plugins, PHI masking. Phase 2 Core Agents (steel blue, W7–W14): multi-agent framework, Process Framework workflows, semantic memory integration, ASP.NET Core API and UI. Phase 3 Clinical Pilot (sage green, W14–W22): integration testing, clinical pilot for triage and discharge, monitoring dashboards. Phase 4 Scale (mid green, W20–W26): production rollout and governance cadence establishment.

9.3 Phase Implementation Detail

• Phase 1 - Foundation (Weeks 0–7): establish the SK kernel, AOAI connectivity, and core FHIR/EHR plugin infrastructure before any clinical workflow automation begins

– Register Semantic Kernel in DI: services.AddKernel().AddAzureOpenAIChatCompletion(deployment, endpoint, new DefaultAzureCredential())

– Implement IFunctionInvocationFilter for audit logging as the very first registered filter - this ensures no plugin execution goes unlogged from day one of development

• Phase 2 - Core Agents (Weeks 7–14): build the multi-agent orchestration layer and stateful Process Framework workflows

– Start with AgentGroupChat for the Triage workflow - it is the most impactful and provides the clearest test of multi-agent coordination before the more complex ProcessBuilder workflows are attempted

– Implement semantic memory in parallel with agent development - the FHIR resource chunking strategy requires two weeks of calibration to achieve >0.93 Recall@5 on clinical protocols

• Phase 3 - Clinical Pilot (Weeks 14–22): validate with real clinical workflows in a controlled staging environment before production exposure

– Pilot with the Clinical Informatics team, who are most comfortable evaluating both the technical outputs and the clinical quality of generated summaries and triage recommendations

– Measure all six Table 6 KPIs from day one of pilot - establish baselines that will be the reference for go/no-go production decision at week 18

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

- Phase 4 - Scale (Weeks 20–26): extend to all clinical departments with role-specific configurations and establish governance cadence
  – Department-specific system prompts should be co-authored with department heads and clinical informatics - generic prompts produce lower clinical accuracy than role-tuned ones

X. LIMITATIONS AND FUTURE WORK

10.1 Current Limitations

- Stepwise planner non-determinism: LLM-driven planners can produce different execution sequences for identical inputs - deterministic healthcare workflows should migrate to ProcessBuilder where plan stability is required for regulatory reproducibility
- Context window ceiling: GPT-4o's 128K context window, while large, is still finite - complex workflows incorporating full FHIR Patient summaries across multiple encounters can exceed 32K tokens, requiring selective retrieval strategies that may miss contextually relevant historical data
- Agent hallucination in clinical reasoning: despite RAG grounding, GPT-4o can synthesise plausible-but-incorrect clinical relationships when referenced data is ambiguous - human-in-the-loop review remains mandatory for all clinical decision recommendations in SK v1.20
- SK Process Framework maturity: released GA in September 2025 (SK v1.20), the Process Framework has production reference architecture but limited long-term observability tooling - monitoring integration requires custom OpenTelemetry instrumentation not yet available in first-party tooling

10.2 Future Directions

- Multimodal agents: SK's upcoming vision connector support will enable agents to process medical images - radiology DICOM thumbnails, wound care photographs, and ECG traces - within the same orchestration pipeline as text-based clinical workflows
- Real-time streaming responses: SK's streaming completion support, combined with Blazor's Server-Side Events, will enable real-time discharge summary dictation-assist - displaying AI-generated text token-by-token as the clinician speaks
- Fine-tuned clinical models: as Azure OpenAI extends GPT-4o fine-tuning to healthcare data, SK-based workflows will benefit from specialty-specific model variants (e.g., a radiology-fine-tuned GPT-4o) without changing the plugin or agent architecture
- SK Agent-to-Agent protocol (A2A): Microsoft's emerging inter-agent communication protocol will enable SK healthcare agents to interoperate with third-party healthcare AI agents - enabling cross-organisation clinical collaboration workflows

XI. CONCLUSION

Semantic Kernel provides healthcare engineering teams with the most complete, .NET-native, production-ready framework for building agentic AI healthcare workflow automation available as of November 2025. Its layered architecture - from kernel and connectors through plugins, planners, and the Process Framework - maps naturally to the structured, compliance-constrained nature of clinical workflows.

- The Process Framework pattern achieves the highest performance across all seven measured dimensions: 94% clinical accuracy, 96% workflow completion, 1.5% error rate, and 94% HIPAA compliance score - all meeting or exceeding established SLA targets
- The eight FHIR R4 plugin functions documented in this paper cover the core data access needs for triage, prior authorisation, medication reconciliation, lab processing, discharge planning, and referral management - the six highest-impact automation targets in US hospital operations
- SK's IFunctionInvocationFilter and IPromptRenderFilter pipeline provides the PHI masking and audit logging capabilities required for HIPAA compliance without modifying agent or plugin code - a separation of concerns that enables security controls to be independently tested and updated
- The token treemap analysis reveals that 70% of monthly token cost is attributable to FHIR and EHR context loading - the highest-leverage optimisation opportunity for cost management as workflow volumes scale

The most important architectural decision for healthcare SK deployments is not which LLM to use or which planner to select - it is the design of the IFunctionInvocationFilter audit pipeline. A healthcare organisation that cannot prove

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

what data its AI system accessed, when, and on whose behalf cannot meet HIPAA requirements. Audit-first design is not an afterthought; it is the foundation.

REFERENCES

1 Microsoft Corporation. (2025). Semantic Kernel overview - Microsoft Learn. https://learn.microsoft.com/semantic-kernel. Retrieved October 2025.
2 Microsoft Corporation. (2025). Semantic Kernel Agent Framework documentation. Microsoft Learn. Retrieved September 2025.
3 Microsoft Corporation. (2024). Semantic Kernel Process Framework (preview). GitHub - microsoft/semantic-kernel. September 2024.
4 Russell, S., & Norvig, P. (2022). Artificial Intelligence: A Modern Approach (4th ed.). Pearson. ISBN 978-0134610993.
5 Brown, T. B., et al. (2020). Language Models are Few-Shot Learners. NeurIPS 2020. arXiv:2005.14165.
6 Office for Civil Rights, HHS. (2013). HIPAA Security Rule - 45 CFR Part 164. U.S. Department of Health and Human Services.
7 HL7 International. (2023). HL7 FHIR R4 Specification - Fast Healthcare Interoperability Resources Release 4. hl7.org.
8 Microsoft Corporation. (2025). Azure OpenAI Service - GPT-4o documentation. Microsoft Learn. Retrieved October 2025.
9 Chase, H. (2023). LangChain: Building applications with LLMs through composability. GitHub - hwchase17/langchain. v0.0.200.
10 Lewis, P., et al. (2021). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. NeurIPS 2020. arXiv:2005.11401.
11 Microsoft Corporation. (2025). Semantic Kernel Process Framework - Step and event model. Microsoft Learn. Retrieved October 2025.
12 Johnson, A. E. W., et al. (2023). MIMIC-IV Clinical Database: A freely available electronic health record dataset. Scientific Data, 10(1), 1–9.
13 Nori, H., et al. (2023). Capabilities of GPT-4 on Medical Challenge Problems. arXiv:2303.13375. March 2023.
14 Microsoft Corporation. (2024). IFunctionInvocationFilter and IPromptRenderFilter - SK extensibility interfaces. GitHub - microsoft/semantic-kernel.
15 Epic Systems. (2024). SMART on FHIR OAuth2 developer documentation. open.epic.com. Retrieved August 2025.
16 Singhal, K., et al. (2023). Large Language Models Encode Clinical Knowledge (Med-PaLM). Nature, 620, 172–180.
17 Microsoft Corporation. (2024). Azure Health Data Services - FHIR Service documentation. Microsoft Learn.
18 American Health Information Management Association. (2023). AI Governance Standards for Clinical NLP Systems. AHIMA.
19 Polly Library. (2024). Polly .NET resilience library v8.4. App-vNext / Polly. GitHub.
20 National Institute of Standards and Technology. (2023). AI Risk Management Framework v1.0. NIST AI RMF. January 2023.
21 Microsoft Corporation. (2025). Semantic Kernel v1.20 release notes - Process Framework GA. Microsoft DevBlog. September 2025.
22 OpenAI. (2023). GPT-4 Technical Report. arXiv:2303.08774. March 2023.
23 Johnson, M., et al. (2024). Agentic AI in Healthcare: Multi-step Workflow Automation using LLM Orchestration Frameworks. JAMIA Open, 7(3).
24 Microsoft Corporation. (2025). Semantic Kernel Function Calling and Tool Use. Microsoft Learn. Retrieved October 2025.
25 Qdrant. (2024). Qdrant vector database documentation v1.8. qdrant.tech. Retrieved September 2025.

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi crossref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details