OpenClaw Integration Services
Multi-Channel AI Gateway Architecture

I implement self-hosted OpenClaw gateways to create persistent AI agents that interface with your existing team communication stack. OpenClaw operates as a local-first bridge between large language models (Claude, GPT, DeepSeek) and messaging surfaces including Signal, Telegram, Discord, WhatsApp, Slack, Matrix, Microsoft Teams, and iMessage.

Technical Capabilities:

Bidirectional Message Orchestration: Deploy agents that receive natural language commands via chat and execute tasks across distributed systems—calendar management, email dispatch, flight check-ins, file system operations
Lobster Communication Protocol (LCP): Implement peer-to-peer agent-to-agent messaging for autonomous workflow coordination between multiple AI instances
Local Execution Model: All agent logic runs on your infrastructure; LLM inference occurs via API while sensitive operations remain air-gapped
Plugin Architecture: Extend agent capabilities through TypeScript/YAML workflow definitions for project-specific automation
Use Cases for Project Management:

Cross-platform notification aggregation (consolidate GitHub, Linear, Jira alerts into a single Signal thread)
Async standup collection via scheduled WhatsApp prompts with structured response parsing
Calendar conflict resolution through natural language scheduling commands
Hermes Agent Deployment
Self-Improving Autonomous Agent Infrastructure

I configure Hermes Agent (Nous Research) as a persistent, memory-aware AI co-worker that maintains longitudinal context across your project lifecycle. Unlike stateless chatbots, Hermes employs FTS5 full-text search with LLM-powered summarization for cross-session project recall.

Technical Capabilities:

Persistent Memory System: Automatic CLAUDE.md/MEMORY.md generation per project; agents retain context across months of interaction
26-Event Hook Architecture: Trigger custom workflows on agent state changes (tool execution, message receipt, skill acquisition)
Multi-Modal Gateway Support: Native integrations with Telegram, Discord, Slack, Signal, SMS, Matrix, Mattermost, Email, and webhook endpoints
MCP (Model Context Protocol) Integration: Connect Hermes to external tool ecosystems (Google Meet, Notion, Linear) through structured tool calling with trajectory logging
RL-Ready Infrastructure: Export agent trajectories for reinforcement learning fine-tuning via Atropos (optional advanced deployment)

Use Cases for Team Communication:

Automatic Documentation: Agents monitor Discord project channels and generate MEMORY.md archives with decision trails
Skill Auto-Acquisition: Agents learn new capabilities from conversation context—teach it once via chat, it persists the skill
Scheduled Workflow Execution: Cron-like job scheduling for recurring project check-ins ("@channel weekly status summary every Friday 4PM")
Implementation Approach

Both frameworks support hybrid cloud/local deployments. I typically architect solutions where:

OpenClaw handles high-velocity, transactional messaging (alerts, quick commands, multi-platform broadcasting)
Hermes manages long-term project memory and skill development (documentation, institutional knowledge retention, complex multi-step workflows)
Security Model: Both frameworks support API key rotation, message encryption at rest, and agent isolation via containerization. OpenClaw emphasizes "your data never leaves your hardware"; Hermes offers cloud-managed scheduling on Anthropic infrastructure with local-memory options.