

Table of Contents

Preface.....	xiii
1. Introduction to Agents.....	1
Defining AI Agents.....	1
The Pretraining Revolution	2
Types of Agents	3
Model Selection	5
From Synchronous to Asynchronous Operations	6
Practical Applications and Use Cases	7
Workflows and Agents	8
Principles for Building Effective Agentic Systems	11
Organizing for Success in Building Agentic Systems	12
Agentic Frameworks	13
LangGraph	13
AutoGen	14
CrewAI	14
OpenAI Agents Software Development Kit (SDK)	14
Conclusion	15
2. Designing Agent Systems.....	17
Our First Agent System	17
Core Components of Agent Systems	20
Model Selection	21
Tools	24
Designing Capabilities for Specific Tasks	24
Tool Integration and Modularity.....	25
Memory	25
Short-Term Memory	26

Long-Term Memory	26
Memory Management and Retrieval	26
Orchestration	27
Design Trade-Offs	27
Performance: Speed/Accuracy Trade-Offs	27
Scalability: Engineering Scalability for Agent Systems	28
Reliability: Ensuring Robust and Consistent Agent Behavior	29
Costs: Balancing Performance and Expense	30
Architecture Design Patterns	32
Single-Agent Architectures	32
Multiagent Architectures: Collaboration, Parallelism, and Coordination	32
Best Practices	34
Iterative Design	34
Evaluation Strategy	35
Real-World Testing	37
Conclusion	39
3. User Experience Design for Agentic Systems.....	41
Interaction Modalities	42
Text-Based	43
Graphical Interfaces	46
Speech and Voice Interfaces	50
Video-Based Interfaces	53
Combining Modalities for Seamless Experiences	54
The Autonomy Slider	55
Synchronous Versus Asynchronous Agent Experiences	58
Design Principles for Synchronous Experiences	58
Design Principles for Asynchronous Experiences	59
Finding the Balance Between Proactive and Intrusive Agent Behavior	59
Context Retention and Continuity	60
Maintaining State Across Interactions	61
Personalization and Adaptability	62
Communicating Agent Capabilities	63
Communicating Confidence and Uncertainty	64
Asking for Guidance and Input from Users	65
Failing Gracefully	65
Trust in Interaction Design	66
Conclusion	68
4. Tool Use.....	71
LangChain Fundamentals	72
Local Tools	73

API-Based Tools	75
Plug-In Tools	78
Model Context Protocol	81
Stateful Tools	84
Automated Tool Development	85
Foundation Models as Tool Makers	85
Real-Time Code Generation	86
Tool Use Configuration	87
Conclusion	88
5. Orchestration	89
Agent Types	90
Reflex Agents	90
ReAct Agents	90
Planner-Executor Agents	91
Query-Decomposition Agents	91
Reflection Agents	91
Deep Research Agents	92
Tool Selection	93
Standard Tool Selection	94
Semantic Tool Selection	97
Hierarchical Tool Selection	101
Tool Execution	105
Tool Topologies	105
Single Tool Execution	106
Parallel Tool Execution	107
Chains	107
Graphs	109
Context Engineering	112
Conclusion	113
6. Knowledge and Memory	115
Foundational Approaches to Memory	116
Managing Context Windows	116
Traditional Full-Text Search	117
Semantic Memory and Vector Stores	119
Introduction to Semantic Search	119
Implementing Semantic Memory with Vector Stores	119
Retrieval-Augmented Generation	121
Semantic Experience Memory	122
GraphRAG	123
Using Knowledge Graphs	123

75	Building Knowledge Graphs	API-based Tools	124
78	Promise and Peril of Dynamic Knowledge Graphs	Plug-In Tools	130
81	Note-Taking	Model Context Protocol	133
84	Conclusion	Statistical Tools	134
85	Performance Speed/Accuracy Trade-Offs	Automated Tool Development	135
7.	Learning in Agentic Systems.....		135
86	Nonparametric Learning	Real-Time Code Generation	135
87	Nonparametric Exemplar Learning	Tool Use Configuration	135
88	Reflexion	Conclusion	137
	Experiential Learning		141
89	Parametric Learning: Fine-Tuning	Orchestration and Workflow Engines	146
90	Fine-Tuning Large Foundation Models	Agent Types	146
90	The Promise of Small Models	Reflex Agents	151
90	Supervised Fine-Tuning	React Agents	153
91	Direct Preference Optimization	Planner-Executor Agents	158
91	Reinforcement Learning with Verifiable Rewards	Query-Decomposition Agents	161
91	Conclusion	Reflection Agents	162
92	User Experience Design for Agentic Systems.....	Deep Research Agents	163
8.	From One Agent to Many.....		163
94	How Many Agents Do I Need?	Standard Tool Selection	163
97	Single-Agent Scenarios	Semantic Tool Selection	163
101	Multiagent Scenarios	Hierarchical Tool Selection	170
102	Swarms	Tool Execution	177
102	Principles for Adding Agents	Tool Topologies	178
105	Multiagent Coordination	Single Tool Execution	180
107	Democratic Coordination	Parallel Tool Execution	180
107	Manager Coordination	Chains	181
109	Hierarchical Coordination	Graphs	182
112	Actor-Critic Approaches	Context-Specific Ingesting and Intrusive Agent Behavior	182
113	Automated Design of Agent Systems	Conclusion	184
	Communication Techniques		189
115	Local Versus Distributed Communication.....	Knowledge and Memory	189
116	Agent-to-Agent Protocol	Foundational Approaches to Memory	189
116	Message Brokers and Event Buses	Managing Context Windows	192
117	Actor Frameworks: Ray, Orleans, and Akka	Traditional Full-Text Search	195
119	Orchestration and Workflow Engines	Semantic Memory and Vector Stores	199
119	Managing State and Persistence	Introduction to Semantic Search	201
119	Conclusion	Implementing Semantic Memory with Vector Stores	202
121		Retrieval-Augmented Generation	
9.	Validation and Measurement.....		205
123	Measuring Agentic Systems	GraphRAG	205
123	Measurement Is the Keystone	Using Knowledge Graphs	206

Integrating Evaluation into the Development Lifecycle	207
Creating and Scaling Evaluation Sets	207
Component Evaluation	209
Evaluating Tools	209
Evaluating Planning	210
Evaluating Memory	212
Evaluating Learning	213
Holistic Evaluation	214
Performance in End-to-End Scenarios	214
Consistency	216
Coherence	217
Hallucination	218
Handling Unexpected Inputs	219
Preparing for Deployment	220
Conclusion	221
10. Monitoring in Production	223
Monitoring Is How You Learn	224
Monitoring Stacks	226
Grafana with OpenTelemetry, Loki, and Tempo	227
ELK Stack (Elasticsearch, Logstash/Fluentd, Kibana)	227
Arize Phoenix	228
SigNoz	229
Langfuse	229
Choosing the Right Stack	230
OTel Instrumentation	230
Visualization and Alerting	232
Monitoring Patterns	235
Shadow Mode	235
Canary Deployments	235
Regression Trace Collection	236
Self-Healing Agents	236
User Feedback as an Observability Signal	236
Distribution Shifts	237
Metric Ownership and Cross-Functional Governance	239
Conclusion	241
11. Improvement Loops	243
Feedback Pipelines	245
Automated Issue Detection and Root Cause Analysis	250
Human-in-the-Loop Review	251
Prompt and Tool Refinement	254

Aggregating and Prioritizing Improvements	259
Experimentation	260
Shadow Deployments	261
A/B Testing	262
Bayesian Bandits	263
Continuous Learning	265
In-Context Learning	265
Offline Retraining	267
Conclusion	268
12. Protecting Agentic Systems.....	271
The Unique Risks of Agentic Systems	272
Emerging Threat Vectors	273
Securing Foundation Models	275
Defensive Techniques	276
Red Teaming	278
Threat Modeling with MAESTRO.....	281
Protecting Data in Agentic Systems	283
Data Privacy and Encryption	283
Data Provenance and Integrity	285
Handling Sensitive Data	286
Securing Agents	288
Safeguards	288
Protections from External Threats	290
Protections from Internal Failures	292
Conclusion	296
13. Human-Agent Collaboration.....	297
Roles and Autonomy	297
The Changing Role of Humans in Agent Systems	298
Aligning Stakeholders and Driving Adoption	299
Scaling Collaboration	300
Agent Scope and Organizational Roles	302
Shared Memory and Context Boundaries	303
Trust, Governance, and Compliance	305
The Lifecycle of Trust	305
Accountability Frameworks	306
Escalation Design and Oversight.....	309
Privacy and Regulatory Compliance	310
Conclusion: The Future of Human-Agent Teams	312

When I first started connecting language models, tools, orchestration, and memory together into what we now call an agent, I was surprised by how capable this design pattern was, and just how much confusion there was about this topic. During my time building agents and sharing my findings on incident investigation, threat hunting, vulnerability detection, and more, I found that this latest design pattern enabled us to solve whole new classes of problems, but also came with many practical hurdles in making them reliable for real-world applications. Engineers, scientists, product managers, and leadership all wanted to know more. "How do I get my agent to work?" "I can get my agent to work some of the time, but how do I get it to work most or all of the time?" "How do I choose a model for my use case?" "How do I design good tools for my agent?" "What kind of memory do I need?" "Should I use LLM?" "Should I build a single-agent or multiagent system?" "What architecture should I use?" "Do I need to fine-tune?" "How do I enable agents to learn from experience and improve over time?"

While there are many blog posts and research papers that focus on specific aspects of the topic of designing agent systems, I realized there were a lack of accessible, holistic, trustworthy guides for this. I couldn't find the book that I wanted to share with my colleagues, so I set out to write it.

Through in-depth discussions, I've helped teams navigate the complexities of AI agents, considering their unique goals, constraints, and environments. AI agent systems are intricate, blending autonomy, decision making, and interaction in ways that traditional software doesn't. They're data-driven, adaptive, and involve multiple components like perception, reasoning, action, and learning, all while interfacing with users, tools, and other agents. Complicating matters, the foundation models that power these agents are probabilistic and stochastic by nature, making evaluation and testing more challenging.

This book takes a comprehensive approach to building applications with AI agents. It covers the entire lifecycle, from conceptualization to deployment and maintenance.