Domain Applications

Week 11: Code, Finance, and Healthcare Agents

PhD Course in Agentic Artificial Intelligence

12-Week Research-Level Course

Learning Objectives

Bloom's Taxonomy Levels Covered

- **Remember**: Define SWE-bench, code agent, FinAgent, regulatory compliance
- **Understand**: Explain domain-specific requirements for agent deployment
- **Apply**: Implement a code agent using flow engineering (structured pipelines)
- **Analyze**: Compare agent architectures across different domains
- **Evaluate**: Assess regulatory and safety requirements for each domain
- **Create**: Design a domain-specific agent with appropriate safeguards

---

By end of lecture, you will understand how agents adapt to real-world domains.

Domain Maturity Landscape

High Maturity: Software Development

- Clear success criteria (tests pass, code works)
- Sandboxed execution environments
- Active deployment: GitHub Copilot, Cursor, Devin

Medium-High Maturity: Finance

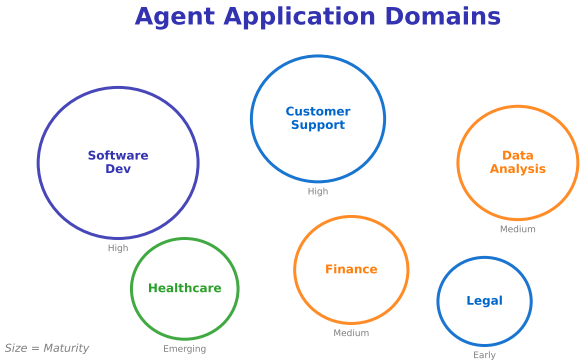
- Well-defined tasks (analysis, research, reporting)
- Heavy regulatory constraints (SEC, FINRA, MiFID II)
- Active deployment: Trading assistants, document analysis, compliance

Finance Sub-Domains

- **Research**: High maturity (summarization, analysis)
- **Trading**: Medium maturity (backtesting safe, live trading risky)
- **Compliance**: Growing (document review, audit trails)

Maturity correlates with ability to verify outputs and contain errors.

Application Domain Maturity



Software development leads in maturity; healthcare is emerging.

Code Agents: The Leading Domain

Why Code is Ideal for Agents

- Clear success criteria: Tests pass or fail
- Safe sandbox: Run code in containers
- Immediate feedback: Execution reveals errors
- Rich context: Codebase provides grounding

Key Capabilities

- Bug fixing and debugging
- Feature implementation from specifications
- Code review and refactoring
- Documentation generation

Current State

- SWE-bench: Best agents solve $\sim$50% of real GitHub issues
- Production systems: Copilot, Cursor, Devin, Claude Code

Code agents now outperform average developers on specific benchmarks.

SWE-bench and AlphaCodium

SWE-bench (Jimenez et al., 2024)

- 2,294 real GitHub issues from 12 Python repositories
- Task: Generate code patch to resolve issue
- Verification: Patch must pass repository tests

AlphaCodium: Flow Engineering (Ridnik et al., 2024)

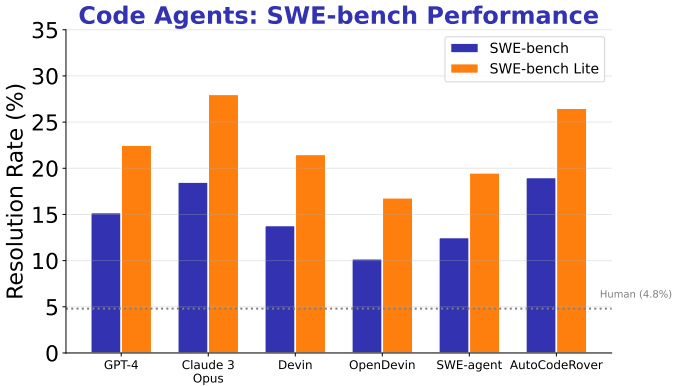
- Structured multi-stage pipeline (not single-shot)
- Stages: Problem reflection, public tests, AI tests, code iteration
- Key insight: Test against multiple cases before submitting

Flow Engineering Principles

- Break complex tasks into simpler stages
- Generate and run tests iteratively
- Use structured output at each stage

Flow engineering = structured pipelines for complex coding tasks.

Code Agents: SWE-bench Performance



Code agents now outperform average human developers on SWE-bench.

Finance Agents

High-Value Applications

- **Research**: Earnings analysis, market research synthesis
- **Trading**: Strategy backtesting, execution assistance
- **Compliance**: Regulatory document analysis, audit trails
- **Operations**: Report generation, data reconciliation

Unique Challenges

- **Regulatory**: SEC, FINRA, MiFID II compliance requirements
- **Explainability**: Must justify recommendations
- **Latency**: Markets move in milliseconds
- **Risk**: Errors have direct financial consequences

Current Deployments

- FinAgent: Multimodal trading agent (research)
- Bloomberg Terminal AI: Document analysis, Q&A

Finance requires compliance (regulatory) awareness at every step.

Finance agents span research, trading, compliance, and operations.

FinAgent: Multimodal Finance Agent

Architecture (Li et al., 2024)

- Multimodal: Text (news, filings), numeric (prices, fundamentals), charts
- Dual memory: Short-term (recent trades), long-term (market patterns)
- Tool use: Market data APIs, technical indicators, portfolio analytics

Key Components

- **Market Perception**: Process multi-modal market signals
- **Agent Memory**: Store and retrieve trading experience
- **Decision Module**: ReAct-style reasoning for trade decisions

Results

- Outperforms baselines on paper trading benchmarks
- Caveat: Simulated environment, not live trading

Multimodal perception is critical for financial markets.

Finance Agent Architectures

Research Agents

- Earnings call analysis and summarization
- SEC filing extraction (10-K, 10-Q, 8-K)
- News sentiment aggregation across sources

Trading Agents

- Strategy backtesting with historical data
- Signal generation from technical/fundamental indicators
- Portfolio rebalancing recommendations

Compliance Agents

- Regulatory document parsing (MiFID II, Dodd-Frank)
- Trade surveillance and anomaly detection
- Audit trail generation and reporting

Different finance tasks require different agent architectures.

Regulatory Compliance in Finance Agents

Key Regulations

- **SEC/FINRA** (US): Suitability rules, best execution, record-keeping
- **MiFID II** (EU): Transparency, investor protection, reporting
- **Basel III**: Capital requirements, risk management

Agent Compliance Patterns

- **Audit logging**: Every decision must be traceable
- **Explainability**: Justify recommendations to regulators
- **Human oversight**: Compliance officer approval for actions
- **Data governance**: Handle PII and market data appropriately

Risk: Unexplainable AI decisions = regulatory violations

Compliance-by-design is mandatory for production finance agents.

Risk Management in Trading Agents

Risk Categories

- **Market risk**: Position limits, VaR constraints, stop-losses
- **Execution risk**: Slippage, failed orders, latency
- **Model risk**: Strategy drift, overfitting, regime change

Agent Safeguards

- Hard position limits (cannot be overridden by agent)
- Kill switches for automated trading
- Human approval above threshold sizes
- Real-time P&L monitoring with alerts

Key Principle: Agents recommend, humans execute high-risk trades

---

Risk controls must be enforced at infrastructure level, not by the agent.

Finance Agent Case Studies

Bloomberg Terminal AI

- Document Q&A over financial filings
- Earnings call summarization
- Human-in-loop for all outputs

Quantitative Research Assistants

- Alpha factor discovery from alternative data
- Automated literature review for trading ideas
- Strategy prototyping (not live execution)

Compliance Automation

- KYC document verification
- Transaction monitoring for AML
- Regulatory report generation

Current focus: Research and compliance; trading execution remains human-controlled.

Cross-Domain Design Patterns

Verification Strategy by Domain

- **Code**: Run tests, syntax checking, type checking
- **Finance Research**: Cross-reference sources, fact-check numbers
- **Finance Trading**: Backtesting, risk limits, compliance rules

Human-in-the-Loop Intensity

- **Code**: Low (automated tests catch most errors)
- **Finance Research**: Medium (analyst review of summaries)
- **Finance Trading**: High (human execution for significant trades)

Common Success Factors

- Domain-specific tools and knowledge bases
- Clear escalation paths for uncertainty
- Audit trails for accountability

Adapt verification intensity to domain risk level.

Required Readings

This Week

- Jimenez et al. (2024). "SWE-bench: Can Language Models Resolve Real-World GitHub Issues?" arXiv:2310.06770
- Ridnik et al. (2024). "AlphaCodium: Code Generation with Flow Engineering." arXiv:2401.08500
- Li et al. (2024). "FinAgent: A Multimodal Foundation Agent for Financial Trading." arXiv:2402.18485

Supplementary

- Yang et al. (2024). "SWE-agent: Agent-Computer Interfaces Enable Software Engineering." arXiv:2405.15793
- Lopez-Lira & Tang (2023). "Can ChatGPT Forecast Stock Price Movements?" arXiv:2304.07619

Focus on SWE-bench for code agents and FinAgent for finance agents.

Summary and Key Takeaways

Domain Insights

- **Code**: Most mature; clear success criteria, safe sandboxing
- **Finance Research**: High value; summarization and analysis
- **Finance Trading**: High risk; requires strict safeguards
- **Finance Compliance**: Growing rapidly; audit and documentation

Design Principles

- Match verification intensity to domain risk
- Build domain-specific tools and knowledge
- Design clear human escalation paths

Next Week

- Research Frontiers and Final Projects

Domain expertise + agent capabilities = real-world impact.