



# AI-Powered Development Teams

Intelligent Agent Workflows for Modern SDLC



Heath Kesler • Savoir Technologies



**<- FEEDBACK**

# Agenda



## Spec-Driven Development

Foundation of AI-powered workflows using structured specifications



## Agent Role Configuration

PO, PE, Architect, Developer, QA agents working in harmony



## Integration Architecture

GitHub, AWS, ServiceNow, Jira, and SpecKit connections



## Implementation Strategy

Actionable roadmap to deploy intelligent automation

# The Challenge: Traditional SDLC Pain Points

## Manual Overhead

- Ticket triage consumes 15-20% of dev time
- Context switching between tools
- Inconsistent code review quality
- Delayed bug resolution cycles
- Manual deployment coordination
- Documentation drift from code

## Coordination Friction

- Spec ambiguity causes rework
- Handoff delays between roles
- Environment promotion bottlenecks
- Release versioning complexity
- Testing coverage gaps
- Knowledge silos across teams



01

# The Vision

AI Agents as Collaborative Team Members

# AI-Powered Team Architecture



**Product Owner Agent**



**Architect Agent**



**Developer Agent**



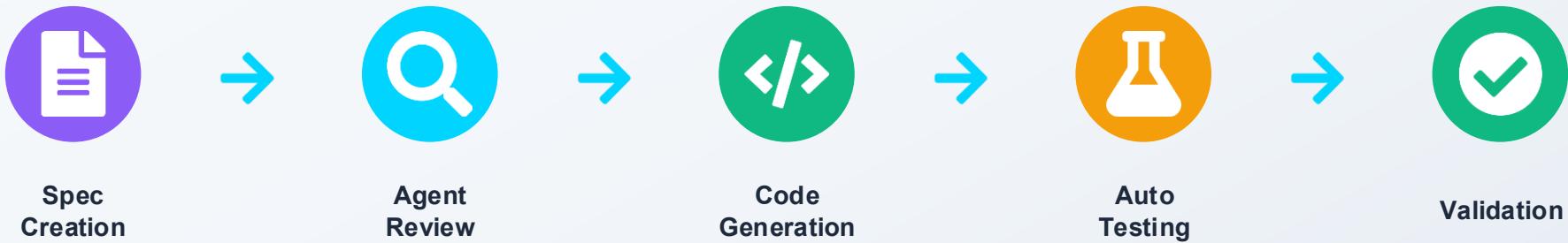
**QA Agent**



**DevOps Agent**

- Each agent operates autonomously within defined boundaries
- Agents communicate through structured spec documents
- MCP servers provide tool access and context
- Human oversight at critical decision points
- Continuous learning from feedback loops
- Seamless handoffs between agent roles
- Unified logging and audit trail

# Spec-Driven Development Workflow



SpecKit provides the structured foundation for agent communication. Specifications define requirements, acceptance criteria, and constraints. Agents parse specs to understand scope, generate implementation plans, and validate outputs against defined criteria. This creates a deterministic, auditable workflow where every change traces back to an approved specification.



02

# Agent Role Configuration

Specialized Agents for Every SDLC Phase



# Product Owner Agent



## Responsibilities

- Requirement gathering and refinement
- Backlog prioritization
- Stakeholder communication
- Acceptance criteria definition
- Sprint planning support

## AI Automations

- Parse incoming requests into structured specs
- Auto-prioritize based on business rules
- Generate user stories from conversations
- Validate completeness of requirements
- Flag spec conflicts and dependencies

## Integration Points





# Product Engineer Agent



## Responsibilities

- Technical feasibility analysis
- Solution design proposals
- Cross-team coordination
- Technical debt assessment
- Integration planning

## AI Automations

- Analyze specs for technical viability
- Generate architecture decision records
- Estimate effort based on codebase analysis
- Identify reusable components
- Suggest technical approaches

## Integration Points





## Responsibilities

- System design and patterns
- Infrastructure planning
- Security architecture
- Scalability assessment
- Technology selection

## AI Automations

- Generate Terraform modules from specs
- Design API contracts automatically
- Validate against security policies
- Create infrastructure diagrams
- Review PRs for architectural compliance

## Integration Points





# Developer Agent



## Responsibilities

- Feature implementation
- Code review participation
- Bug fixing and debugging
- Unit test creation
- Documentation updates

## AI Automations

- Generate code from specifications
- Auto-fix issues from test failures
- Respond to PR review comments
- Refactor based on static analysis
- Create inline documentation

## Integration Points





# Quality Assurance Agent



## Responsibilities

- Test strategy development
- Test case creation
- Regression testing
- Performance validation
- Bug verification

## AI Automations

- Generate test cases from specs
- Execute automated test suites
- Analyze test coverage gaps
- Create bug reports from failures
- Validate fixes against test criteria

## Integration Points





# DevOps / Release Agent



## Responsibilities

- CI/CD pipeline management
- Environment provisioning
- Release coordination
- Monitoring and alerting
- Incident response

## AI Automations

- Orchestrate deployments across environments
- Manage release versioning
- Rollback on failure detection
- Scale infrastructure automatically
- Generate release notes

## Integration Points



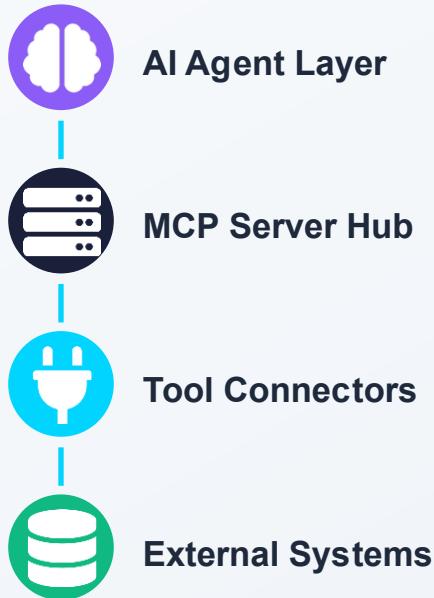


03

# Integration Architecture

Connecting Tools Through MCP Servers

# MCP Server Architecture



- MCP servers expose tool capabilities to AI agents
- Standardized protocol for tool invocation
- Context injection for relevant data access
- Authentication and authorization handling
- Rate limiting and resource management
- Audit logging for all tool interactions
- Extensible connector framework



# GitHub Integration



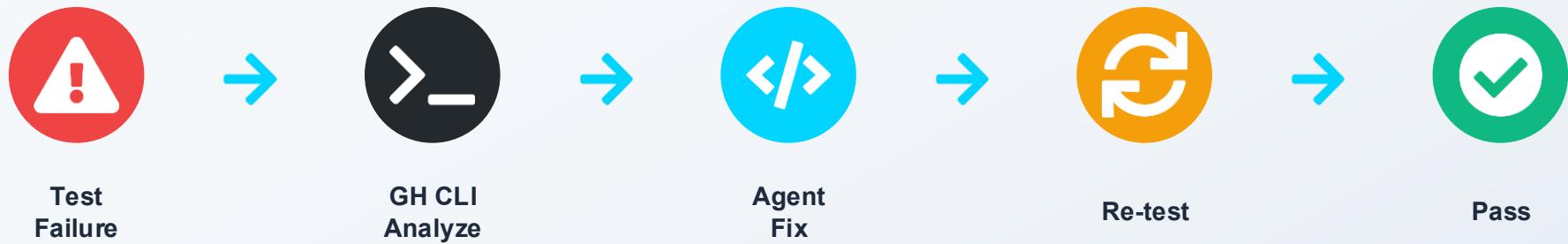
## Key Features

- Repository management via API
- Branch protection enforcement
- PR automation and review
- Issue tracking integration
- Actions workflow triggers
- Release tag management

## Automated Workflow

1. Agent creates feature branch
2. Code pushed with linked spec
3. PR auto-generated with description
4. Review comments processed
5. Merge on approval
6. Release tag created

# GitHub CLI: Automated Issue Resolution



The gh CLI enables agents to interact with GitHub conversations, analyze test failures, and automatically propose fixes. When CI detects a failure, the agent retrieves the error context, analyzes the root cause against the codebase, generates a fix, and submits it for review. This closed-loop system dramatically reduces mean time to resolution while maintaining code quality standards.



# SpecKit: Specification-Driven Development



## Key Features

- Structured spec templates
- Version-controlled specs
- Dependency mapping
- Validation rules engine
- Change tracking
- Agent-readable format

## Automated Workflow

1. PO creates spec from template
2. Architect validates feasibility
3. Developer parses requirements
4. QA extracts test criteria
5. All changes tracked to spec



# AWS Infrastructure (Terraform + Docker)



## Key Features

- Infrastructure as Code (Terraform)
- Container orchestration (ECS/EKS)
- Auto-scaling policies
- Security group management
- Cost optimization rules
- Multi-region deployment

## Automated Workflow

1. Spec triggers infra requirements
2. Terraform plan generated
3. Review and approval gate
4. Infrastructure provisioned
5. Docker images deployed
6. Health checks validated

# Environment Promotion Pipeline



Automated promotion through environments with quality gates at each stage. DEV for rapid iteration with automatic deploys on commit. QA for integration testing with spec validation. PERF for load testing and performance benchmarks. PROD deployment requires explicit approval with release versioning. Rollback capabilities at each stage ensure stability.



# ServiceNow: Ticket Processing & Release Control



## Key Features

- Automated ticket creation
- Change request workflows
- Release version tracking
- Approval chain automation
- Audit compliance logging
- SLA monitoring

## Automated Workflow

1. Code changes linked to tickets
2. Release candidate created
3. Version number assigned
4. Approval workflow triggered
5. PROD deploy on approval
6. Ticket auto-closed on success

# Release Versioning Strategy



**Feature Branch**



**Version Tag**



**QA Validation**



**Release Lock**



**PROD Deploy**

- Semantic versioning enforced automatically
- QA-passed versions tagged for release
- Only validated versions eligible for PROD
- Release notes auto-generated from commits
- Rollback targets previous validated version
- ServiceNow tracks deployment history
- Prevents untested code reaching production



# Jira: Automated Bug Ticket Processing



## Key Features

- Auto-triage incoming bugs
- Duplicate detection
- Priority assignment
- Developer auto-assignment
- Fix verification workflow
- SLA tracking

## Automated Workflow

1. Bug reported via any channel
2. Agent analyzes reproduction steps
3. Codebase searched for root cause
4. Fix PR linked to ticket
5. QA validates fix
6. Ticket closed on deploy



# Jira: New Logic Initiatives



## Key Features

- Initiative intake automation
- Epic breakdown assistance
- Story generation from specs
- Capacity planning integration
- Progress tracking
- Dependency visualization

## Automated Workflow

1. Initiative submitted
2. Agent breaks into epics
3. Stories generated from specs
4. Estimates calculated
5. Sprint assignment suggested
6. Progress auto-updated

# Documentation: Confluence vs. Agent Query

## Confluence Wiki

- Auto-generated from code comments
- Architecture decision records
- Runbook automation
- API documentation sync
- Searchable knowledge base
- Version history tracking

## Agent-Based Queries

- Natural language questions
- Context-aware responses
- Real-time codebase analysis
- No documentation drift
- Instant onboarding support
- Reduced maintenance overhead



04

# Implementation Roadmap

Practical Steps to Deploy AI Automation

# Implementation Phases



## Phase 1: Assessment

Audit current workflows, identify automation candidates, baseline metrics



## Phase 2: Foundation

Deploy MCP servers, configure integrations, establish spec templates



## Phase 3: Agent Rollout

Deploy agents incrementally, starting with low-risk automations



## Phase 4: Optimization

Measure impact, refine prompts, expand automation scope

# Expected Outcomes

**40%**

Reduction in ticket triage time

**60%**

Faster bug resolution

**3x**

Deployment frequency increase

**25%**

Developer productivity gain

**90%**

Spec-to-code traceability

**50%**

Reduction in context switching

# Key Takeaways



## Specs are the Contract

Structured specifications enable deterministic agent behavior and auditability



## Integration is Critical

MCP servers unify tool access; seamless connections drive automation value



## Agents Augment, Not Replace

AI handles routine tasks while humans focus on creative problem-solving



## Iterate and Improve

Start small, measure outcomes, expand based on proven success



# Thank You

## Questions & Discussion



Heath Kesler • Savoir Technologies



Implement intelligent automation that reduces overhead  
while improving productivity and delivery speed.

<- FEEDBACK