

# Facin Ia Organizational Model

- [1 FACIN\\_Ia - Modelo de Responsabilidade Organizacional](#)
  - [1.1 Visão Geral](#)
  - [1.2 Contexto do FACIN\\_Ia](#)
    - [1.2.1 Visão do Projeto](#)
    - [1.2.2 Domínio de Negócio](#)
  - [1.3 Modelo de Responsabilidade Organizacional Adaptado](#)
    - [1.3.1 Camadas de Responsabilidade](#)
  - [1.4 Papéis e Responsabilidades](#)
    - [1.4.1 CAMADA ESTRATÉGICA](#)
    - [1.4.2 CAMADA TÁTICA](#)
    - [1.4.3 CAMADA OPERACIONAL](#)
    - [1.4.4 CAMADA DE SUPORTE](#)
  - [1.5 Matriz de Conhecimentos por Papel](#)
  - [1.6 Conhecimentos Técnicos Específicos do FACIN\\_Ia](#)
    - [1.6.1 Para Desenvolvimento \(AI/ML Engineers + Backend\)](#)
    - [1.6.2 Para DevOps](#)
    - [1.6.3 Para QA](#)
  - [1.7 Alinhamento com FACIN](#)
    - [1.7.1 Dimensões do FACIN Aplicadas](#)
  - [1.8 Matriz RACI \(Exemplo para Features\)](#)
  - [1.9 Roadmap de Capacitação](#)
    - [1.9.1 Fase 1: Fundamentos \(Mês 1-2\)](#)
    - [1.9.2 Fase 2: Especialização \(Mês 3-4\)](#)
    - [1.9.3 Fase 3: Domínio Técnico \(Mês 5-6\)](#)
    - [1.9.4 Fase 4: Excelência \(Mês 7+\)](#)
  - [1.10 Recursos de Aprendizado](#)
    - [1.10.1 Para AI/ML Engineers](#)
    - [1.10.2 Para DevOps](#)
  - [1.11 Métricas de Sucesso por Papel](#)
  - [1.12 Contatos e Governança](#)
  - [1.13 Anexos](#)
    - [1.13.1 A. Glossário FACIN\\_Ia](#)
    - [1.13.2 B. Checklist de Onboarding](#)
    - [1.13.3 C. Templates](#)
  - [1.14 Referências](#)

## 1 FACIN\_Ia - Modelo de Responsabilidade Organizacional

### 1.1 Visão Geral

Este documento adapta o **Modelo de Responsabilidade Organizacional** para o contexto do projeto **FACIN\_Ia**, alinhando a visão de negócios do FACIN (Framework de Arquitetura Corporativa para Inovação) com as necessidades específicas de um sistema inteligente baseado em agentes de IA.

**Referências:** - [FACIN e o Modelo de Responsabilidade Organizacional](#) - [IEEE - Organizational Responsibility Model](#)

### 1.2 Contexto do FACIN\_Ia

#### 1.2.1 Visão do Projeto

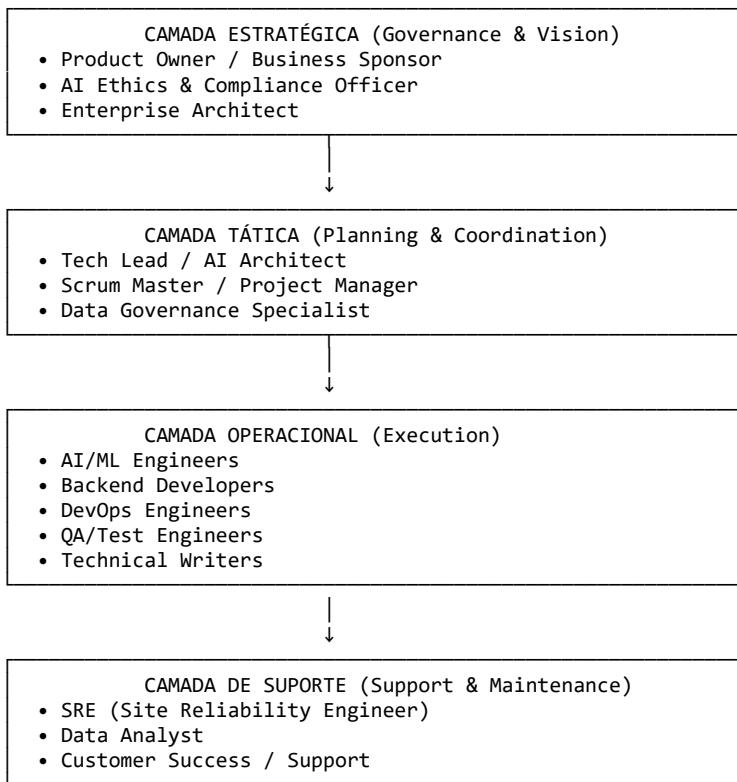
Sistema Multi-Agentes de IA para automação da Folha de Pagamento, utilizando: - **LangGraph**: Orquestração de agentes - **LLMs**: Groq (Llama 3.1) + OpenAI (GPT-3.5) - **AgenticOps**: Monitoramento e observabilidade - **CI/CD**: GitHub Actions com validação obrigatória - **Arquitetura**: Python + Streamlit + SQLite

### 1.2.2 Domínio de Negócio

- Gestão de folha de pagamento
- Consultas inteligentes sobre servidores e remunerações
- Análise de dados com processamento de linguagem natural
- Automação de processos administrativos

## 1.3 Modelo de Responsabilidade Organizacional Adaptado

### 1.3.1 Camadas de Responsabilidade



## 1.4 Papéis e Responsabilidades

### 1.4.1 CAMADA ESTRATÉGICA

#### 1.4.1.1. Product Owner (PO) / Business Sponsor

**Responsabilidades:** - Definir visão e roadmap do produto - Priorizar features e backlog - Alinhar requisitos de negócio com capacidades técnicas - Validar entregas e ROI - Interface com stakeholders e usuários finais

**Conhecimentos Necessários:** -  Gestão de folha de pagamento e RH -  Processos administrativos governamentais -  Legislação trabalhista -  Fundamentos de IA/ML (conceitual) -  Metodologias ágeis (Scrum/Kanban) -  Análise de negócios e KPIs

**Artefatos:** - Product Vision Statement - Product Roadmap - User Stories e Acceptance Criteria - Business Case e ROI Analysis

#### 1.4.1.2 2. AI Ethics & Compliance Officer

**Responsabilidades:** - Garantir uso ético de IA e dados sensíveis - Avaliar vieses e fairness dos modelos - Compliance com LGPD/GDPR - Auditorias de privacidade e segurança - Políticas de governança de dados

**Conhecimentos Necessários:** -  LGPD, GDPR e regulamentações de proteção de dados -  Ética em IA e machine learning -  Auditoria de sistemas de IA -  Gestão de riscos -  Segurança da informação -  Fundamentos técnicos de ML

**Artefatos:** - Privacy Impact Assessment (PIA) - AI Ethics Guidelines - Compliance Checklist - Audit Reports

---

#### 1.4.1.3 3. Enterprise Architect

**Responsabilidades:** - Alinhar FACIN\_IA com arquitetura corporativa - Definir padrões e guidelines técnicos - Avaliar integração com sistemas legados - Estratégia de evolução tecnológica - Governança de arquitetura

**Conhecimentos Necessários:** -  Frameworks de arquitetura (TOGAF, Zachman, FACIN) -  Arquitetura de sistemas distribuídos -  Arquitetura de IA/ML em produção -  Integração de sistemas corporativos -  Cloud computing e infraestrutura -  Padrões de design e boas práticas

**Artefatos:** - Architecture Decision Records (ADR) - Integration Architecture Diagram - Technology Radar - Standards & Guidelines Document

---

### 1.4.2 CAMADA TÁTICA

#### 1.4.2.1 4. Tech Lead / AI Architect

**Responsabilidades:** - Design técnico da solução de IA - Seleção de modelos LLM e frameworks - Definição de arquitetura de agentes (LangGraph) - Code reviews e qualidade técnica - Mentoria técnica do time - POCs e experimentação

**Conhecimentos Necessários:** -  Python avançado (tipagem, async, decorators) -  LangChain + LangGraph (orchestration, agents, tools) -  LLMs: OpenAI, Groq, Anthropic, fine-tuning -  Prompt Engineering (system prompts, few-shot, CoT) -  RAG (Retrieval-Augmented Generation) -  Vector databases (Pinecone, Weaviate, ChromaDB) -  AgenticOps e observability -  Arquitetura de software (SOLID, Design Patterns) -  MLOps e deployment de modelos -  Performance optimization (latency, cost)

**Artefatos:** - Technical Design Document (TDD) - Agent Architecture Diagrams - LLM Selection Matrix - Performance Benchmarks - Code Review Guidelines

---

#### 1.4.2.2 5. Scrum Master / Project Manager

**Responsabilidades:** - Facilitar cerimônias ágeis - Remover impedimentos do time - Gestão de riscos e cronograma - Comunicação com stakeholders - Métricas de performance do time

**Conhecimentos Necessários:** -  Metodologias ágeis (Scrum, Kanban) -  Ferramentas de gestão (Jira, Azure DevOps) -  Gestão de riscos -  Comunicação e facilitação -  Métricas ágeis (velocity, burn-down) -  Fundamentos de desenvolvimento de software

**Artefatos:** - Sprint Planning & Retrospectives - Risk Register - Project Timeline - Team Metrics Dashboard

---

#### 1.4.2.3 6. Data Governance Specialist

**Responsabilidades:** - Governança de dados sensíveis (CPF, salários) - Qualidade e linhagem de dados - Catálogo de dados e metadados - Políticas de acesso e retenção - Compliance com regulamentações

**Conhecimentos Necessários:** -  Data governance frameworks (DAMA-DMBOK) -  Qualidade de dados (data profiling, cleansing) -  SQL e modelagem de dados -  LGPD/GDPR aplicado a dados -  Data lineage e catalogação -  Master Data Management (MDM)

**Artefatos:** - Data Governance Policy - Data Quality Reports - Data Catalog - Access Control Matrix

### 1.4.3 ⚡ CAMADA OPERACIONAL

#### 1.4.3.1 7. AI/ML Engineer

**Responsabilidades:** - Implementação de agentes LangGraph - Desenvolvimento de ferramentas (@tool) - Prompt engineering e otimização - Integração com LLMs (Groq, OpenAI) - Fine-tuning de modelos (se necessário) - Experimentação e A/B testing

**Conhecimentos Necessários:** -  Python (expert level) -  LangChain: ChatModels, Prompts, Tools, Chains -  LangGraph: StateGraph, nodes, edges, checkpoints -  LLM APIs: OpenAI, Groq, Anthropic -  Prompt Engineering: system prompts, templates, few-shot -  Streamlit (interface web) -  SQLite/PostgreSQL (queries, schemas) -  Git/GitHub (branching, PRs) -  Testing: pytest, mocking LLM calls -  AgenticOps: observability, logging

**Tecnologias do Projeto:**

```
langchain==0.3.24
langchain-groq==0.3.2
langchain-openai==0.3.14
langgraph==0.3.34
streamlit==1.44.1
```

**Artefatos:** - Agent implementations (groq\_agent\_node, openai\_agent\_node) - Tool definitions (@tool decorators) - Prompt templates - Unit tests (pytest) - Performance experiments

#### 1.4.3.2 8. Backend Developer

**Responsabilidades:** - Implementação de APIs REST/GraphQL (se necessário) - Integração com banco de dados - Lógica de negócio (validações, cálculos) - Otimização de queries SQL - Refatoração e manutenção de código

**Conhecimentos Necessários:** -  Python (OOP, type hints, async) -  FastAPI ou Flask (se APIs necessárias) -  SQLite/PostgreSQL: queries complexas, JOINs, índices -  Pandas: manipulação de dados CSV/Excel -  ORMs: SQLAlchemy (opcional) -  Padrões de design: Repository, Service Layer -  Testing: pytest, integration tests -  Git/GitHub

**Artefatos:** - Database schemas (tb\_servidores, tb\_folha\_pagamento) - SQL queries e migrations - Business logic layer - Integration tests

#### 1.4.3.3 9. DevOps Engineer

**Responsabilidades:** - CI/CD pipelines (GitHub Actions) - Containerização (Docker) - Infraestrutura como código (Terraform) - Deployment e releases - Monitoramento e alertas - Gestão de secrets e configurações

**Conhecimentos Necessários:** -  GitHub Actions: workflows, jobs, secrets -  Docker: Dockerfile, docker-compose, multi-stage builds -  CI/CD: pipelines, gates, automação -  Cloud platforms: AWS, GCP, Azure, Heroku -  Kubernetes (para escalabilidade futura) -  Terraform (infraestrutura como código) -  Monitoring: Prometheus, Grafana, AgenticOps -  Secrets management: GitHub Secrets, Vault

**Artefatos do Projeto:** - .github/workflows/validate.yml - Dockerfile - docker-compose.yml - Deployment scripts - Monitoring dashboards

#### 1.4.3.4 10. QA / Test Engineer

**Responsabilidades:** - Testes de qualidade (funcionais, não-funcionais) - Automação de testes - Testes de LLMs (prompt validation) - Testes de regressão - Relatórios de bugs e qualidade

**Conhecimentos Necessários:** -  **pytest**: fixtures, parametrize, mocking -  **Testing LLMs**: determinism, prompt testing -  **Load testing**: Locust, JMeter -  **Test automation**: Selenium (se UI complexa) -  **Python** (intermediário) -  **SQL** (validação de dados) -  **CI/CD integration**

**Artefatos do Projeto:** - tests/ directory (test\_app.py, test\_database.py, etc.) - Test coverage reports (70%+ required) - Test plans e test cases

---

#### 1.4.3.5 11. Technical Writer

**Responsabilidades:** - Documentação técnica (APIs, arquitetura) - Guias de usuário - Documentação de onboarding - Manutenção de wikis e READMEs - Vídeos tutoriais (opcional)

**Conhecimentos Necessários:** -  **Markdown** (documentação) -  **Diagramas**: Mermaid, PlantUML, Draw.io -  **Sphinx ou MkDocs** (documentação estruturada) -  **Git/GitHub** (commits de documentação) -  Fundamentos técnicos de IA/ML -  **UX Writing**

**Artefatos do Projeto:** - docs/README.md - docs/API.md - docs/ARCHITECTURE.md - docs/QUICKSTART.md - docs/CONTRIBUTING.md

---

#### 1.4.4 ⚙ CAMADA DE SUPORTE

##### 1.4.4.1 12. SRE (Site Reliability Engineer)

**Responsabilidades:** - Disponibilidade e confiabilidade (SLA) - Incident management e postmortems - Monitoramento e alertas - Performance tuning - Disaster recovery e backups

**Conhecimentos Necessários:** -  **Observability**: metrics, logs, traces -  **AgenticOps**: LLM monitoring, cost tracking -  **Alerting**: PagerDuty, Opsgenie -  **Performance**: profiling, optimization -  **Reliability engineering**: SLOs, error budgets -  **Python** (debugging produção) -  **SQL** (troubleshooting queries)

**Artefatos:** - SLA/SLO definitions - Runbooks e playbooks - Incident reports - Performance dashboards

---

##### 1.4.4.2 13. Data Analyst

**Responsabilidades:** - Análise de uso do sistema - Métricas de negócio (consultas, tempo resposta) - Análise de dados de folha de pagamento - Dashboards e visualizações - Insights para melhorias

**Conhecimentos Necessários:** -  **SQL** (queries analíticas) -  **Pandas, NumPy** (análise de dados) -  **Visualização**: Matplotlib, Plotly, Streamlit -  **BI tools**: Power BI, Tableau (opcional) -  **Excel avançado** -  **Estatística descritiva**

**Artefatos:** - Usage analytics dashboards - Business intelligence reports - Data quality assessments

---

##### 1.4.4.3 14. Customer Success / Support

**Responsabilidades:** - Suporte a usuários finais - Treinamento e onboarding - Coleta de feedback - Documentação de FAQs - Escalação de issues

**Conhecimentos Necessários:** -  Domínio de folha de pagamento -  Uso do sistema FACIN\_Ia -  Comunicação e empatia -  Ferramentas de ticketing (Zendesk, Freshdesk) -  Fundamentos de IA (explicar limitações)

**Artefatos:** - FAQ documentation - Training materials - User feedback reports

---

## 1.5 Matriz de Conhecimentos por Papel

Papel	Python	LangChain/Graph	LLMs	DevOps	SQL	Negócio	Gestão
Product Owner	● Básico	● Conceitual	● Conceitual	-	● Básico	● Expert	● Expert
AI Ethics Officer	● Básico	● Conceitual	● Intermediário	-	-	● Intermediário	● Básico
Enterprise Architect	● Intermediário	● Básico					
Tech Lead / AI Architect	● Expert	● Expert	● Expert	● Intermediário	● Intermediário	● Básico	● Básico
Scrum Master	● Básico	● Conceitual	● Conceitual	-	-	● Intermediário	● Expert
Data Governance	● Básico	-	-	-	● Expert	● Intermediário	● Intermediário
AI/ML Engineer	● Expert	● Expert	● Expert	● Básico	● Intermediário	● Básico	-
Backend Developer	● Expert	● Básico	● Básico	● Básico	● Expert	● Básico	-
DevOps Engineer	● Intermediário	● Básico	● Básico	● Expert	● Básico	-	-
QA Engineer	● Intermediário	● Intermediário	● Intermediário	● Básico	● Intermediário	● Básico	-
Technical Writer	● Básico	● Intermediário	● Intermediário	-	● Básico	● Intermediário	-
SRE	● Intermediário	● Básico	● Básico	● Expert	-	● Intermediário	-
Data Analyst	● Intermediário	-	-	-	● Expert	● Intermediário	-
Support	-	-	● Conceitual	-	● Básico	● Expert	● Básico

Legenda: - ● **Expert:** Conhecimento profundo, capaz de ensinar outros - ● **Intermediário:** Trabalho autônomo, resolve problemas complexos - ● **Básico/Conceitual:** Entendimento geral, requer suporte - -: Não aplicável

## 1.6 Conhecimentos Técnicos Específicos do FACIN\_IA

### 1.6.1 Para Desenvolvimento (AI/ML Engineers + Backend)

#### 1.6.1.1 Stack Principal

```
# LLM Orchestration
langchain==0.3.24
langchain-groq==0.3.2
langchain-openai==0.3.14
langgraph==0.3.34
```

```
# Web Framework
streamlit==1.44.1
```

```
# Database
sqlite3 (built-in)
pandas==2.2.3
SQLAlchemy==2.0.40
```

```
# Utilities
```

```
python-dotenv==1.1.0
pydantic==2.11.3
```

### 1.6.1.2 Conceitos Chave

1. **LangGraph State Management**
  - AgentState(TypedDict) com operator.add
  - StateGraph com nós e arestas condicionais
  - Mensagens: HumanMessage, AIMessage, ToolMessage
2. **Tool Creation**
  - Decorator @tool para funções
  - Docstrings como descrição para LLM
  - Type hints obrigatórios
3. **Agent Routing**
  - Lógica de roteamento condicional
  - Alternância automática entre agentes
  - Menções explícitas (@groq, @openai)
4. **Prompt Engineering**
  - System prompts específicos por agente
  - ChatPromptTemplate com MessagesPlaceholder
  - Few-shot examples (se necessário)

### 1.6.1.3 Segurança

- Validação SQL (apenas SELECT permitido)
- Sanitização de inputs
- Tratamento de dados sensíveis (CPF mascarado)
- LGPD compliance

## 1.6.2 Para DevOps

### 1.6.2.1 CI/CD Pipeline

```
# .github/workflows/validate.yml
- Spec validation (obrigatória)
- Code quality (black, isort, flake8, mypy)
- Tests (pytest, 70%+ coverage)
- Build (Docker image)
- Deploy (staging/production)
```

### 1.6.2.2 Containerização

```
# Dockerfile
FROM python:3.12-slim
# Multi-stage builds para otimização
# Healthchecks para reliability
```

### 1.6.2.3 Monitoramento

- AgenticOps: LLM calls, latency, costs
- Logs estruturados: JSON format
- Metrics: Prometheus format
- Alerts: Threshold-based

## 1.6.3 Para QA

### 1.6.3.1 Estratégia de Testes

1. **Unit Tests (70%+ coverage)**

```
# tests/test_agents.py
# Mocking LLM responses
# Testing tool execution
```

## 2. Integration Tests

```
# tests/test_database.py
# Real SQLite database
# CSV data Loading
```

## 3. LLM Testing

- Prompt regression tests
- Output format validation
- Edge cases (empty results, errors)

## 4. Performance Tests

- Latency benchmarks
- Token usage optimization
- Concurrent users

# 1.7 Alinhamento com FACIN

## 1.7.1 Dimensões do FACIN Aplicadas

### 1.7.1.1 Estratégia de Negócio

- **Visão:** Automatizar folha de pagamento com IA conversacional
- **Objetivos:** Reduzir tempo de consultas, melhorar eficiência
- **KPIs:** Tempo médio de resposta, taxa de sucesso, satisfação do usuário

### 1.7.1.2 Arquitetura Corporativa

- **Camada de Apresentação:** Streamlit (web UI)
- **Camada de Lógica:** LangGraph agents
- **Camada de Dados:** SQLite (dev), PostgreSQL (prod)
- **Integrações:** APIs de LLM (OpenAI, Groq)

### 1.7.1.3 Governança

- **Políticas:** LGPD compliance, AI ethics
- **Padrões:** Code quality (black, mypy), testing (70%+)
- **Processos:** CI/CD obrigatório, code reviews, spec validation

### 1.7.1.4 Tecnologia

- **Frameworks:** LangChain, LangGraph, Streamlit
- **Infraestrutura:** Docker, GitHub Actions, Cloud (AWS/GCP/Heroku)
- **Observability:** AgenticOps, structured logging

# 1.8 Matriz RACI (Exemplo para Features)

Atividade	PO	Tech Lead	AI Engineer	Backend	DevOps	QA
Definir requisitos	R/A	C	C	C	I	C
Design de agentes	C	R/A	C	I	I	I
Implementar agentes	I	R	A	C	I	I
Implementar DB logic	I	R	C	A	I	C
Setup CI/CD	I	C	I	I	R/A	C

Atividade	PO	Tech Lead	AI Engineer	Backend	DevOps	QA
Testes	I	R	C	C	I	A
Deploy produção	A	R	I	I	A	C
Monitoramento	I	C	I	I	R/A	C

**Legenda RACI:** - **R** (Responsible): Executa a tarefa - **A** (Accountable): Responsável final, aprova - **C** (Consulted): Consultado, fornece input - **I** (Informed): Informado sobre progresso

---

## 1.9 Roadmap de Capacitação

### 1.9.1 Fase 1: Fundamentos (Mês 1-2)

- Python intermediário/avançado
- LangChain basics
- Prompt engineering fundamentals
- Git/GitHub workflows

### 1.9.2 Fase 2: Especialização (Mês 3-4)

- LangGraph advanced
- LLM APIs (OpenAI, Groq)
- AgenticOps e observability
- CI/CD com GitHub Actions

### 1.9.3 Fase 3: Domínio Técnico (Mês 5-6)

- Fine-tuning de modelos
- Performance optimization
- Security e compliance
- Deployment em produção

### 1.9.4 Fase 4: Excelência (Mês 7+)

- RAG avançado
  - Multi-modal agents
  - Escalabilidade e MLOps
  - Innovation e research
- 

## 1.10 Recursos de Aprendizado

### 1.10.1 Para AI/ML Engineers

**Cursos:** - LangChain Academy (oficial) - DeepLearning.AI - LangChain courses - Fast.ai - Practical Deep Learning

**Livros:** - “Building LLM Apps” - Valentina Alto - “Designing Data-Intensive Applications” - Martin Kleppmann

**Documentação:** - [LangChain Docs](#) - [LangGraph Docs](#) - [OpenAI Cookbook](#)

### 1.10.2 Para DevOps

**Cursos:** - GitHub Actions Fundamentals - Docker & Kubernetes - MLOps Specialization (Coursera)

**Certificações:** - AWS Certified DevOps Engineer - Certified Kubernetes Administrator (CKA)

---

## 1.11 Métricas de Sucesso por Papel

Papel	Métricas-Chave
<b>Product Owner</b>	Feature adoption rate, User satisfaction, Business value delivered
<b>Tech Lead</b>	Code quality scores, Architecture decisions documented, Team velocity
<b>AI/ML Engineer</b>	Model performance, Latency (p95), Cost per query, Test coverage
<b>DevOps</b>	Deployment frequency, MTTR, Pipeline success rate, Uptime SLA
<b>QA</b>	Bug detection rate, Test coverage, Automation rate

## 1.12 Contatos e Governança

### Modelo de Escalação:

Nível 1: Support → Documentação/FAQs  
 Nível 2: AI/ML Engineer → Troubleshooting técnico  
 Nível 3: Tech Lead → Decisões arquiteturais  
 Nível 4: Product Owner + Enterprise Architect → Estratégia

**Cadência de Reuniões:** - Daily Standup (15 min) - Sprint Planning (2h, bi-weekly) - Sprint Review (1h, bi-weekly) - Sprint Retrospective (1h, bi-weekly) - Tech Sync (1h, weekly) - Architecture Review Board (2h, monthly)

## 1.13 Anexos

### 1.13.1 A. Glossário FACIN\_IA

- **Agent:** Entidade autônoma que processa mensagens e toma decisões
- **LLM:** Large Language Model (Groq, OpenAI)
- **Tool:** Função Python decorada com `@tool`, executável por agentes
- **StateGraph:** Estrutura LangGraph para orquestração de agentes
- **AgenticOps:** Plataforma de observabilidade para sistemas de agentes

### 1.13.2 B. Checklist de Onboarding

- Acesso ao repositório GitHub
- Setup de ambiente local (Python, venv, dependencies)
- Acesso às APIs (OpenAI, Groq)
- Treinamento em LangChain/LangGraph
- Leitura da documentação (docs/)
- Primeiro PR (contribuição guiada)

### 1.13.3 C. Templates

- [Architecture Decision Record \(ADR\)](#)
- [Test Case Template](#)
- [Incident Report](#)

**Versão:** 1.0.0

**Data:** 28/02/2026

**Autor:** FACIN\_IA Team

**Revisão:** Enterprise Architecture Board

## 1.14 Referências

1. FACIN - Framework de Arquitetura Corporativa para Inovação
2. IEEE - Organizational Responsibility Model (2014)
3. TOGAF 9.2 - The Open Group Architecture Framework
4. RACI Matrix - Project Management Institute
5. DAMA-DMBOK - Data Management Body of Knowledge

- 6. MLOps Principles - Google Cloud
- 7. LangChain Documentation
- 8. AgenticOps Best Practices