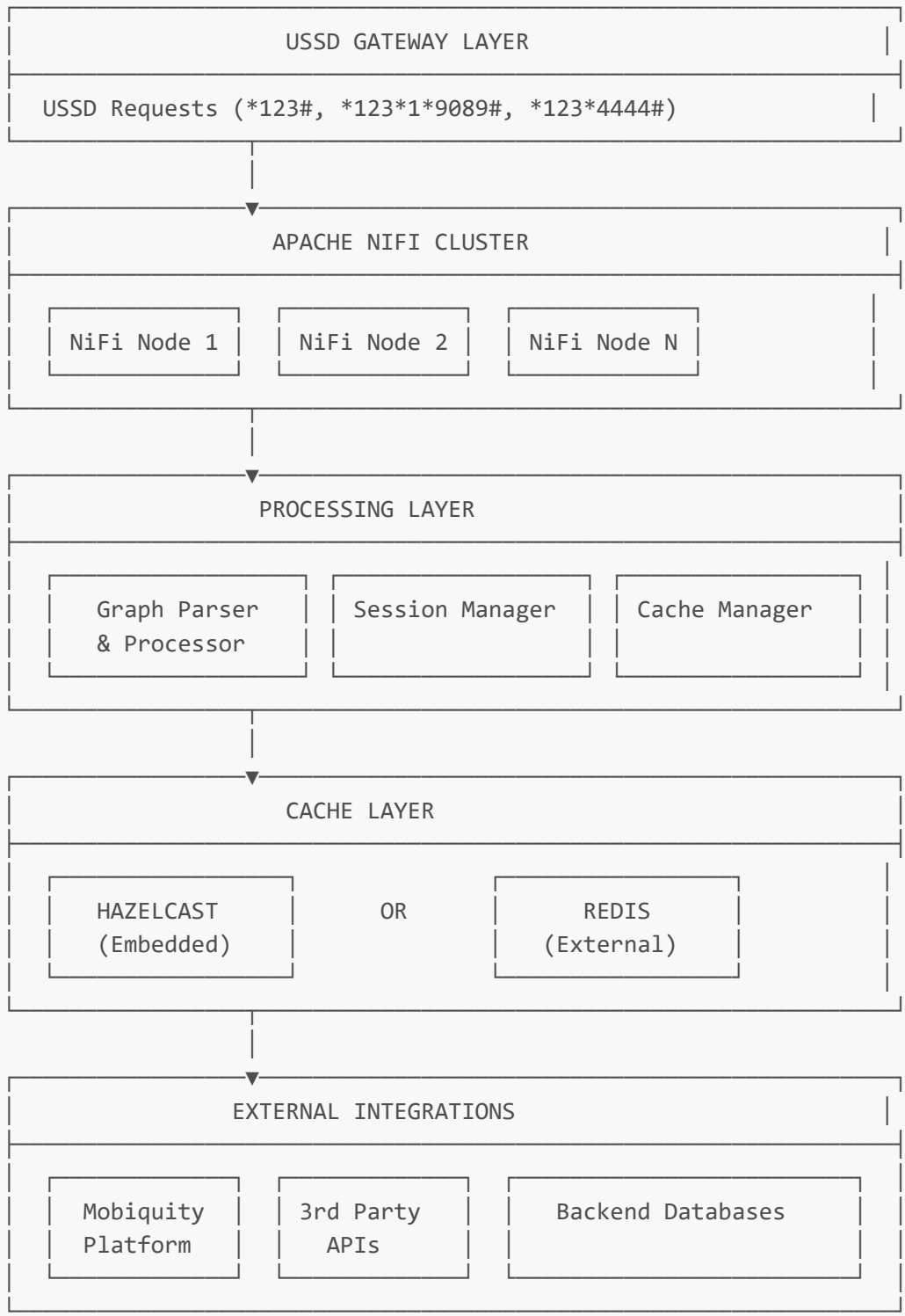
USSD Menu Manager - Architecture Documentation

System Architecture Overview

The USSD Menu Manager is built on Apache NiFi with a microservices-oriented architecture that processes USSD interactions through directed graph navigation. The system employs distributed caching, event-driven processing, and modular Groovy scripts to deliver high-performance USSD services.

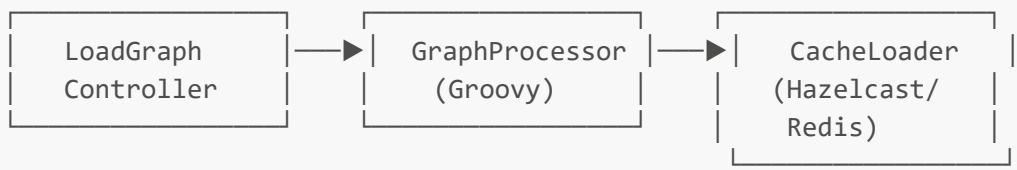


Core Architecture Components

1. NiFi Flow Architecture

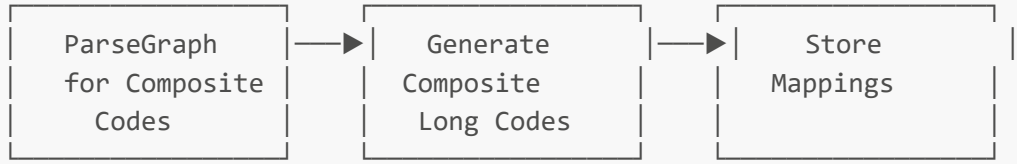
Main Processing Flows

Flow 1: Graph Loading and Node Distribution

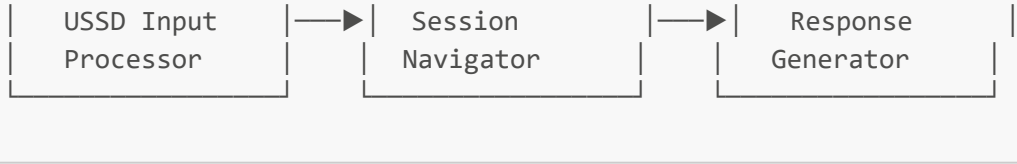


Flow 2: API Template Management

Flow 3: Composite Code Processing

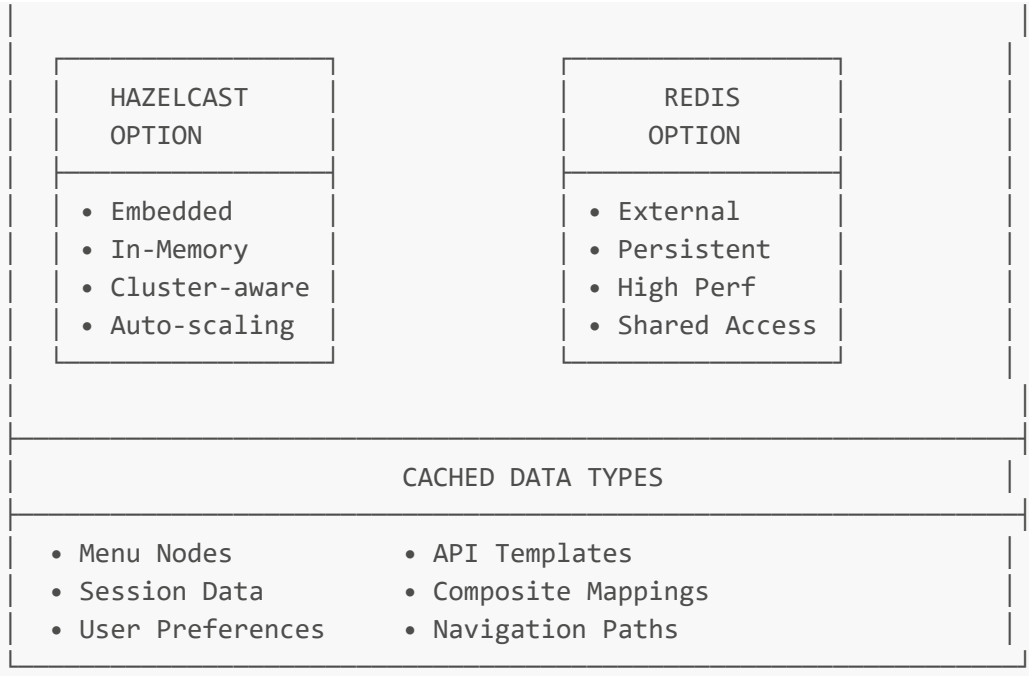


Flow 4: USSD Session Processing



2. Cache Architecture

Dual Cache Strategy



3. Data Models

Graph Node Structure

```
{
  "id": "menu_12345",
  "type": "MENU|INPUT|END|START|ACTION|DYNAMIC-MENU",
  "compositCode": "4444",
  "transitions": {
    "1": "target_node_id",
    "2": "another_node_id"
  },
  "nextNodesMetadata": {
    "1": {
      "nextNodeType": "INPUT",
      "nextNodePrompts": {
        "en": "Enter amount:",
        "es": "Ingrese monto:",
        "fr": "Saisissez le montant:",
        "ar": "أدخل المبلغ:"
      },
      "nextNodeStoreAttribute": "AMOUNT"
    }
  },
  "storeAttribute": "USER_CHOICE"
}
```

Session Data Structure

```
{
  "sessionId": "uuid-session-id",
  "currentNodeId": "menu_12345",
  "userInputs": {
    "AMOUNT": "1000",
    "MSISDN": "1234567890"
  },
  "navigationPath": ["111", "menu_12345"],
  "language": "en",
  "createdAt": "2025-09-30T10:00:00Z",
  "lastActivity": "2025-09-30T10:05:00Z"
}
```

Composite Code Mapping

```
{
  "composite_long_codes": {
    "*111*4444#": "*111*1*${SENDMONEYAMOUNT}*${SENDMONEYMSISDN}#",
    "*111*7634#":
"*111*1*${SENDMONEYAMOUNT}*${SENDMONEYMSISDN}*1*${IMTPIN}*${IMTAMOUNT}*${IMTMSISDN
}#"
  }
}
```

Technical Stack

Core Technologies

```
┌──────────────────────────────────────────────────────────────┐
│                    TECHNOLOGY STACK                          │
├──────────────────────────────────────────────────────────────┤
│  Framework:        Apache NiFi 2.5.0                         │
│  Language:         Groovy (Dynamic Scripts)                  │
│  Cache (Option 1): Hazelcast 2.5.0 (Embedded)                │
│  Cache (Option 2): Redis (External)                          │
│  Data Format:      JSON                                      │
│  Protocol:         USSD over HTTP/REST                       │
└──────────────────────────────────────────────────────────────┘
```

NiFi Processors Used

```
┌──────────────────────────────────────────────────────────────┐
│                    NIFI PROCESSORS                           │
├──────────────────────────────────────────────────────────────┤
│  • ExecuteScript          → Groovy script execution          │
```

```
│  • GetFile                → Graph JSON input                    │
│  • PutFile                → Output generation                   │
│  • RouteOnContent         → Flow routing                        │
│  • AttributeToJSON        → Data transformation                 │
│  • UpdateAttribute        → Session management                  │
│  • MonitorActivity        → Health monitoring                   │
└─────────────────────────────────────────────────────────────────┘
```

Cache Services Configuration

```
┌─────────────────────────────────────────────────────────────────┐
│                      CACHE SERVICES                             │
├─────────────────────────────────────────────────────────────────┤
│  HAZELCAST CONFIGURATION:                                       │
│  • EmbeddedHazelcastCacheManager                                │
│  • HazelcastMapCacheClient                                      │
│  • Cluster-aware distributed caching                            │
│                                                                 │
│  REDIS CONFIGURATION:                                           │
│  • RedisConnectionManager (Custom Pool)                         │
│  • Jedis 6.2.0 Client Library                                   │
│  • Connection pooling optimization                              │
└─────────────────────────────────────────────────────────────────┘
```

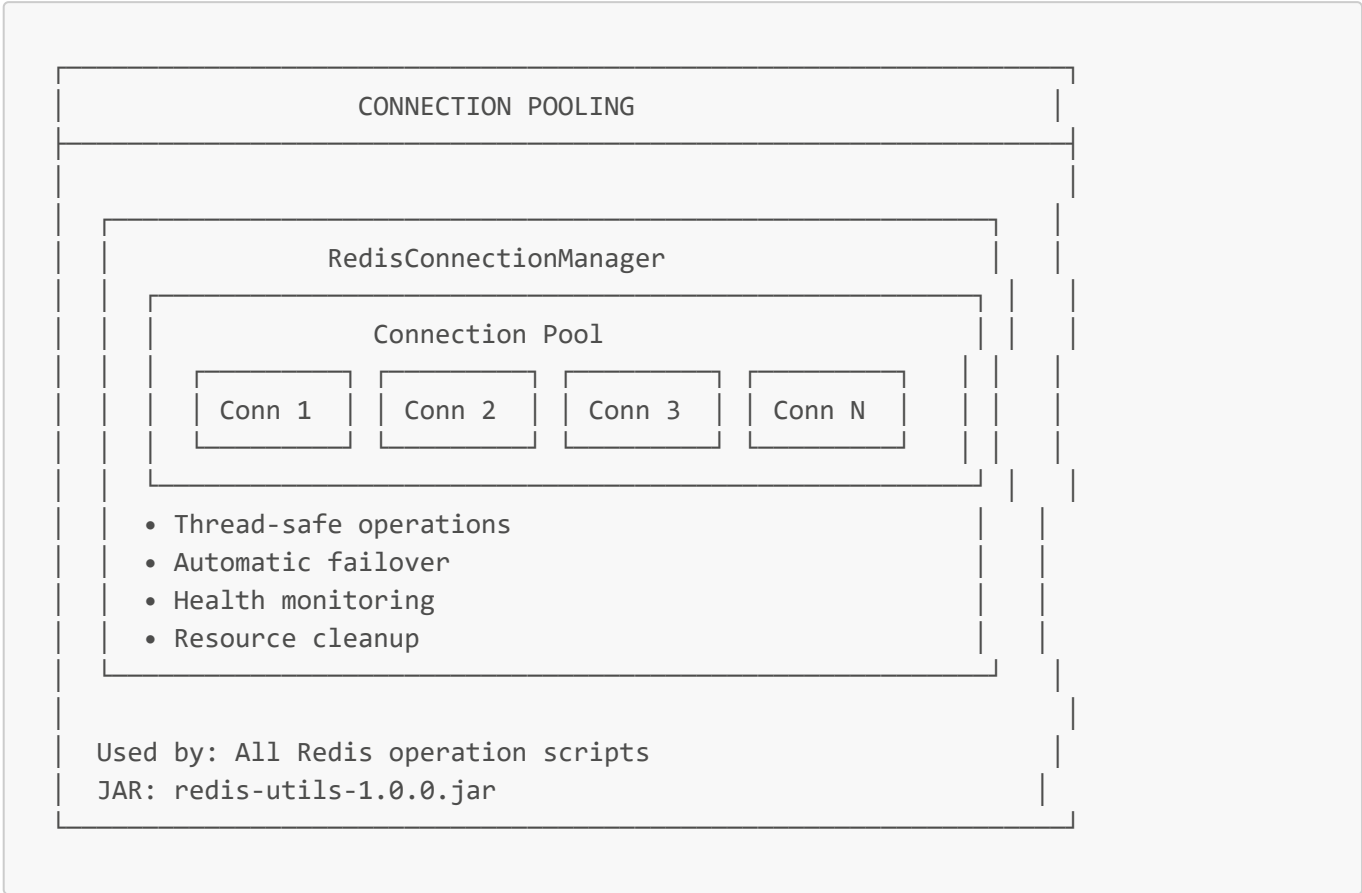
Groovy Script Architecture

Script Modules

```
┌─────────────────────────────────────────────────────────────────┐
│                      GROOVY SCRIPTS                             │
├─────────────────────────────────────────────────────────────────┤
│  Core Processing:                                               │
│  • GraphProcessor.groovy     → Graph navigation logic           │
│  • redisJsonProcessor.groovy → Data processing engine           │
│                                                                 │
│  Cache Operations:                                              │
│  • getRedisJSON.groovy       → Cache retrieval                  │
│  • putRedisJSON.groovy       → Cache storage                    │
│  • deleteRedisJson.groovy    → Cache cleanup                    │
│  • mergeRedisJson.groovy     → Data merging                     │
│                                                                 │
│  Navigation & Flow:                                             │
│  • actionTransition.groovy   → State transitions                │
│  • createDynamicMenu.groovy  → Dynamic content                  │
│  • nodeAttribute.groovy      → Node processing                  │
│                                                                 │
│  Utilities:                                                     │
│  • redisConnectionManager.groovy → Connection management        │
```

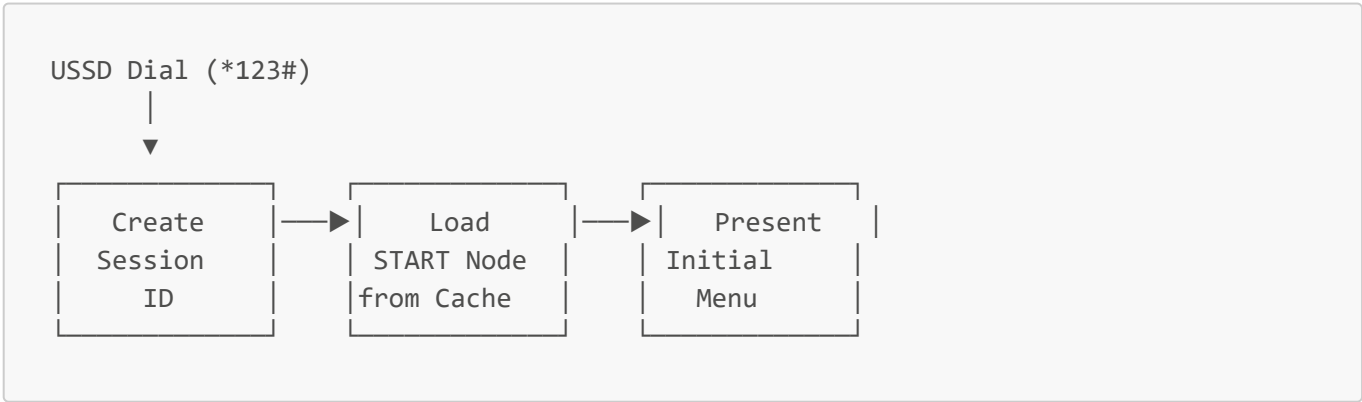
```
│  • RedisConnectionManager.groovy → Pool management              │
└─────────────────────────────────────────────────────────────────┘
```

Connection Management Architecture

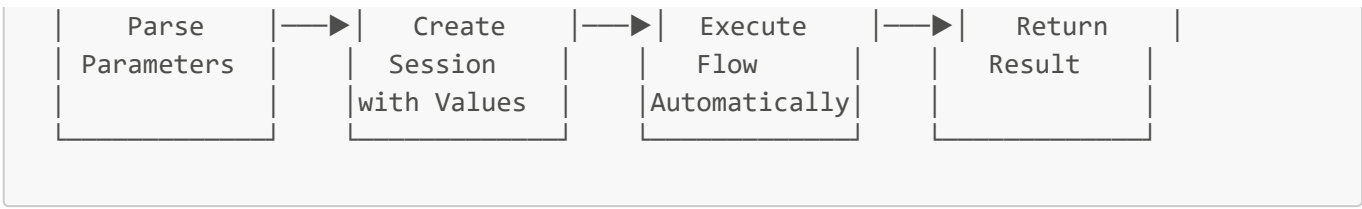


Data Flow Patterns

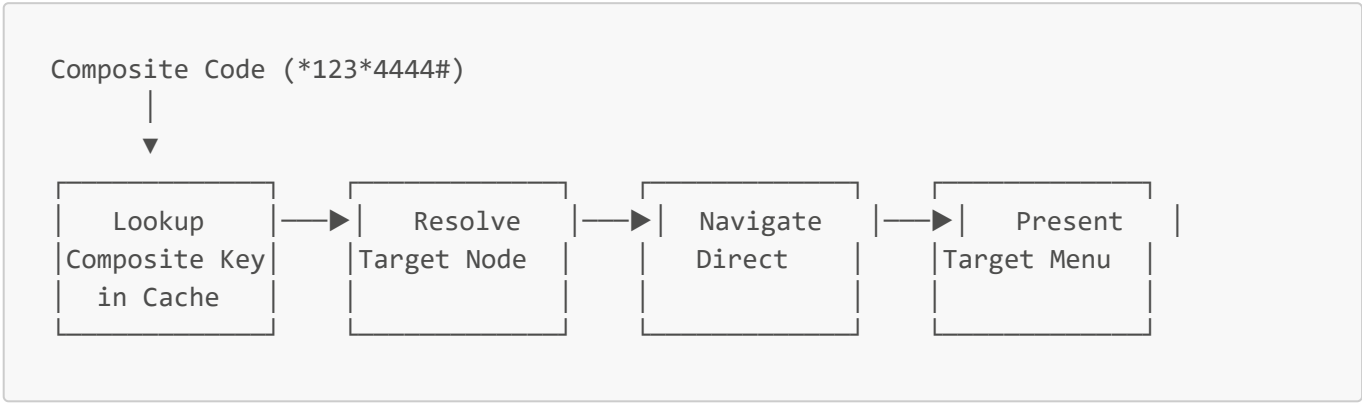
1. Session Initialization Flow



2. Long Code Processing Flow

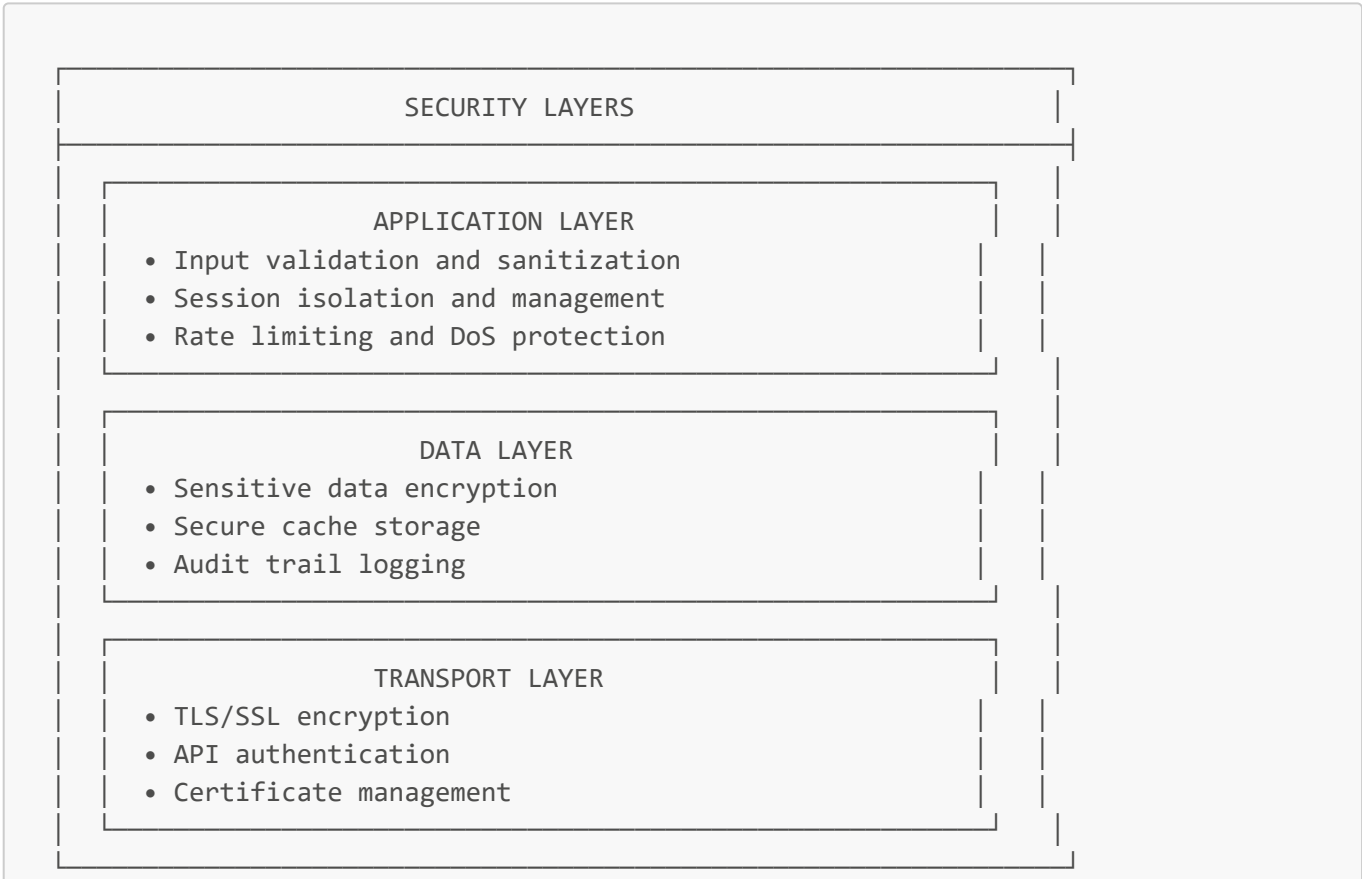


3. Composite Code Resolution Flow



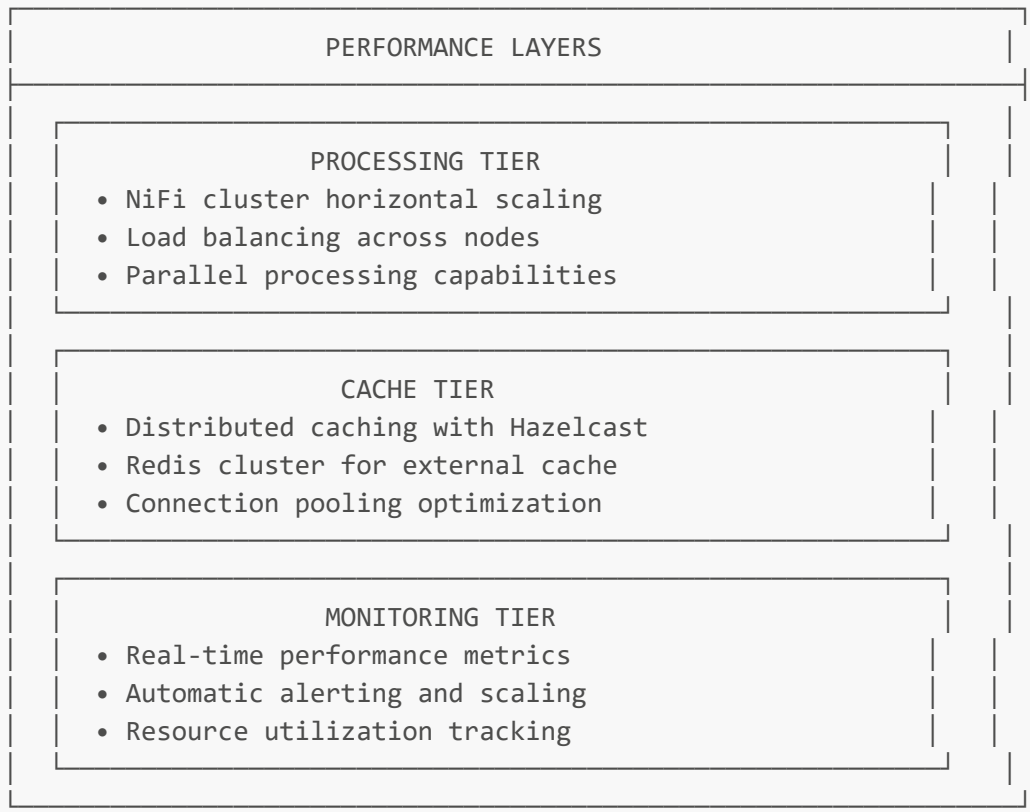
Security Architecture

Data Security Layers



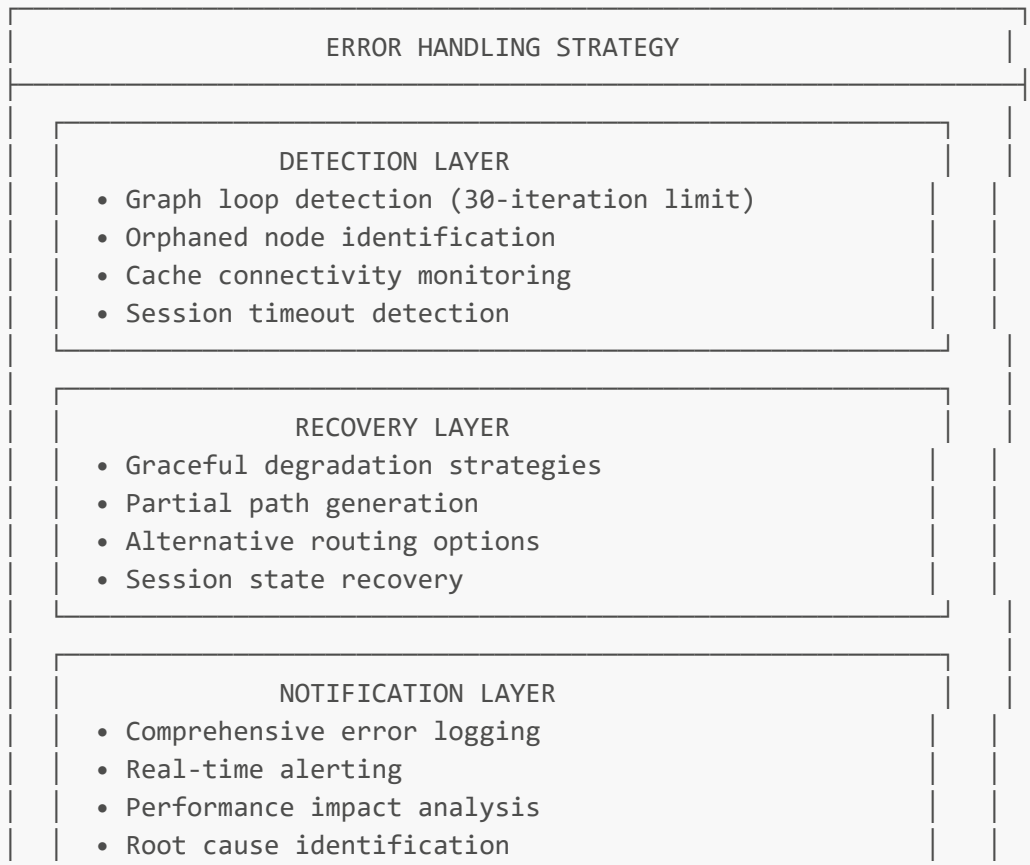
Performance Architecture

Scalability Design



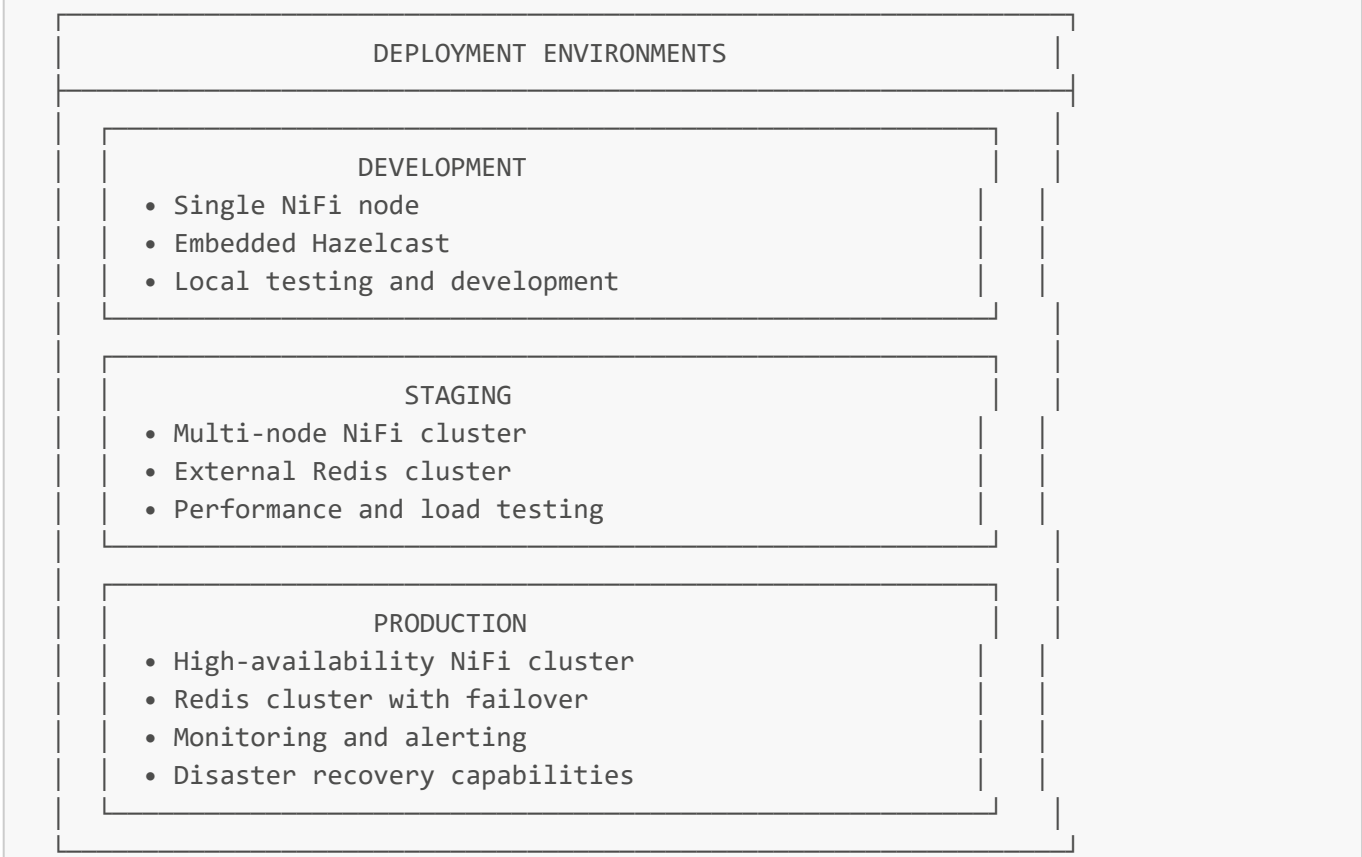
Error Handling Architecture

Fault Tolerance Design



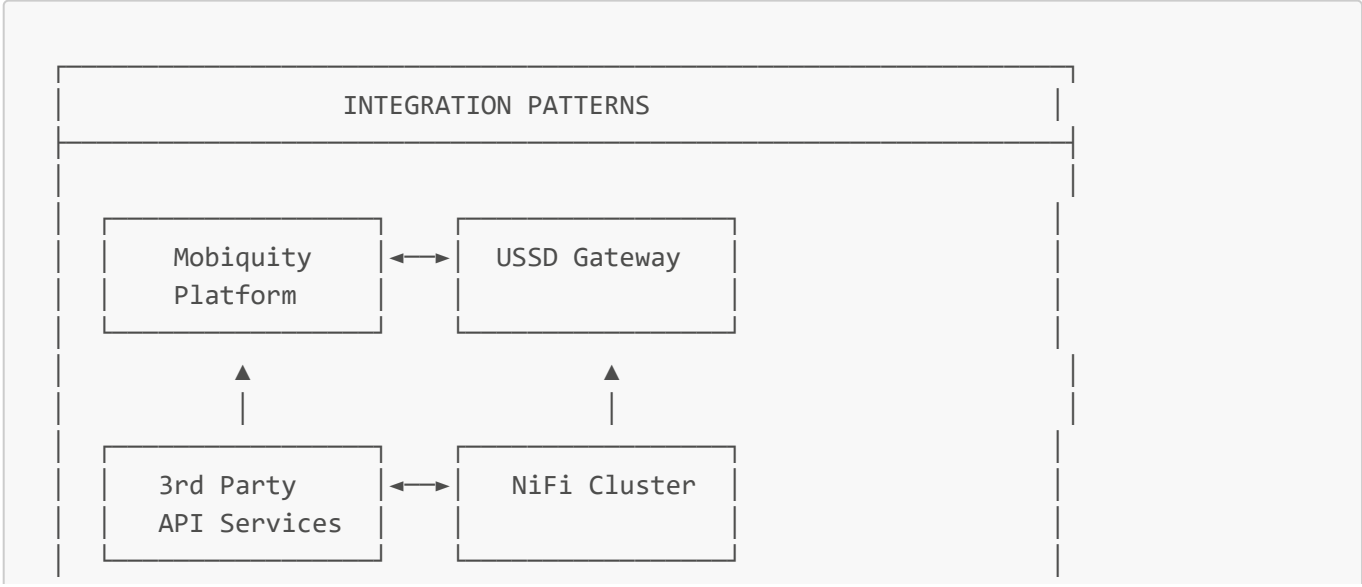
Deployment Architecture

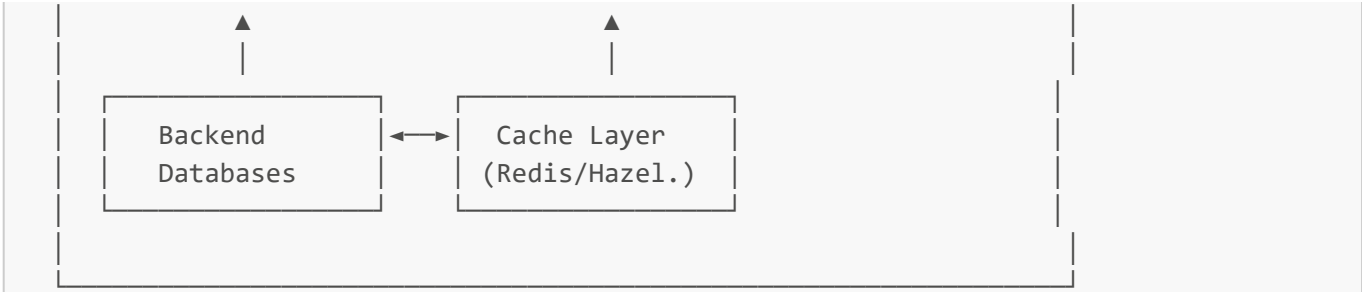
Environment Configuration



Integration Architecture

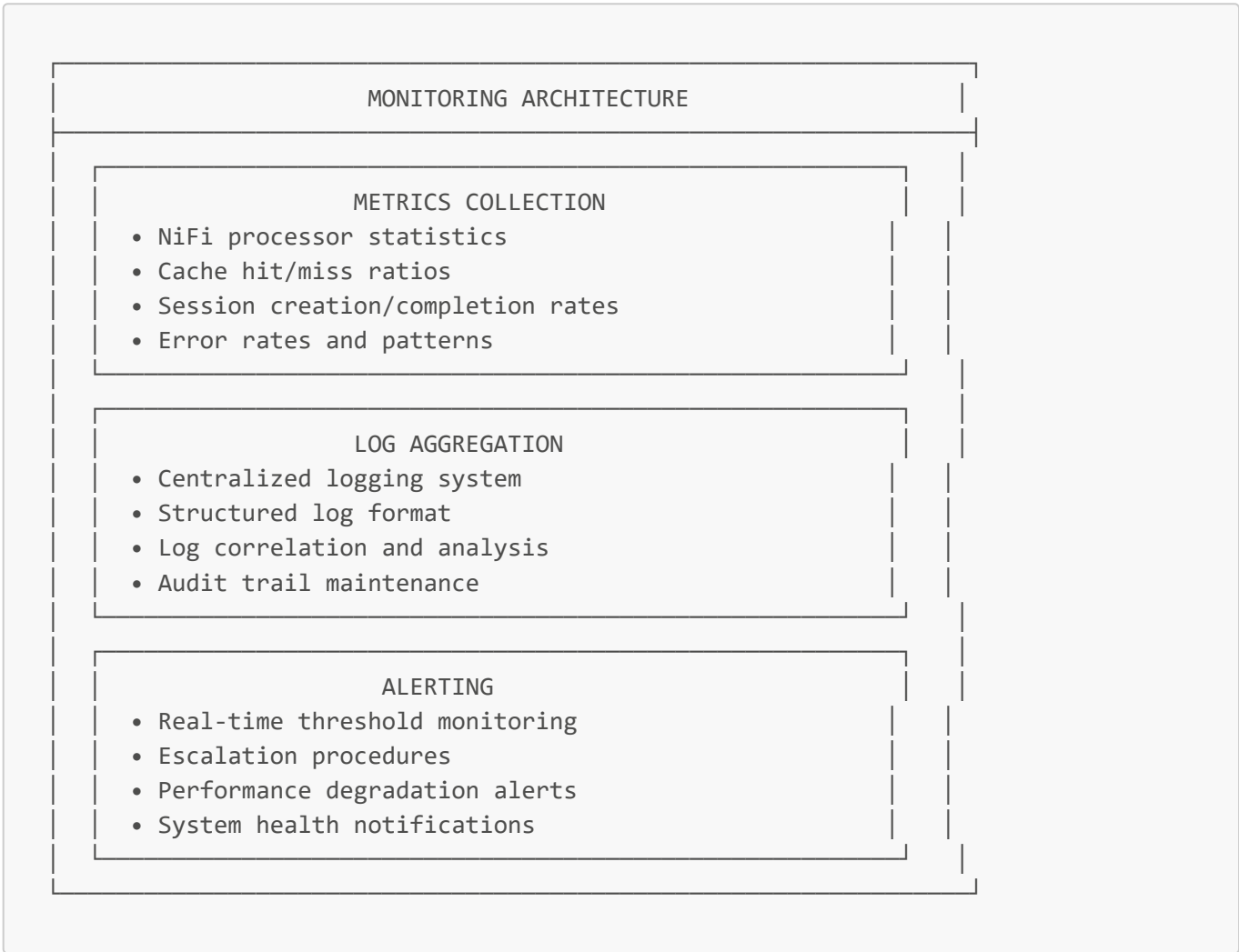
External System Connections





Monitoring and Observability

Monitoring Stack



Future Architecture Considerations

Scalability Roadmap

- **Microservices Migration**: Decompose monolithic flows into independent services
- **API Gateway Integration**: Centralized API management and routing
- **Event Streaming**: Apache Kafka integration for real-time event processing
- **Container Orchestration**: Kubernetes deployment for enhanced scalability

Technology Evolution

- **Cloud-Native Deployment**: Migration to cloud-native NiFi distributions
- **Machine Learning Integration**: Predictive analytics for user behavior
- **Edge Computing**: Distributed processing closer to USSD gateways
- **Blockchain Integration**: Immutable audit trails for financial transactions

This architecture provides a robust, scalable foundation for high-volume USSD processing while maintaining flexibility for future enhancements and integrations.