Advanced Patterns

> **Description:** Cutting-edge capabilities including streaming, MCP protocol, and agent-to-agent communication

Purpose: Explore cutting-edge ADK capabilities for real-time interaction, standardized protocols, and distributed agent systems.

Source of Truth:
google/adk-python/src/google/adk/agents/live_request_queue.py (https://github.com/google/adk-python/tree/main/src/google/adk/agents/live_request_queue.py)
(ADK 1.15) + MCP/A2A implementations

Table of Contents

Streaming & Real-Time Interaction

SSE (Server-Sent Events)

```
# Text streaming to users
async def stream_response(query):
    runner = Runner()
    async for event in runner.run_async(streaming=SSE):
        if event.type == 'content':
            yield f"data: {event.content}\n\n"
        elif event.type == 'done':
            yield "data: [DONE]\n\n"
```

BIDI (Bidirectional Streaming)

```
# Voice/video conversations
queue = LiveRequestQueue()
runner = Runner()

async def live_conversation():
    async for event in runner.run_live(queue):
        if event.type == 'audio_response':
            play_audio(event.audio_data)

        # Send user input
        queue.send_realtime(audio_blob)
```

Models: `gemini-2.0-flash-live-*`, `gemini-live-2.5-*`

🔌 MCP (Model Context Protocol)

Universal Tool Standard

```
# Standardized tool interface
mcp_tools = MCPToolset(
    connection_params=StdioConnectionParams(
        command='npx',
        args=['-y', '@modelcontextprotocol/server-filesystem', '/data']
    )
)

# Works with any MCP-compatible server
# - Filesystem operations
# - Database queries
# - Git operations
# - Slack/Teams integration
```

MCP Benefits

- **Interoperability**: One protocol, many tools
- **Security**: Built-in authentication
- **Discovery**: Auto-detect capabilities
- **Community**: 100+ MCP servers available

🤝 A2A (Agent-to-Agent Communication)

Microservices Architecture

```
# Remote agent integration
youtube_agent = RemoteA2aAgent(
    name='youtube_expert',
    base_url='https://youtube-agent.company.com'
)

# Local agent uses remote expertise
orchestrator = Agent(
    name="content_strategist",
    tools=[AgentTool(youtube_agent)],
    instruction="Create strategy using YouTube analytics"
)
```

A2A vs Local Multi-Agent

- **Distribution**: Agents on different services
- **Scaling**: Independent deployment/scaling
- **Teams**: Cross-team collaboration
- **Specialization**: Domain-specific experts

🚀 Next-Level Capabilities

Multimodal Integration

- **Images**: Vision analysis and generation
- **Audio**: Speech recognition and synthesis
- **Video**: Real-time video processing
- **Documents**: PDF/text extraction and analysis

Code Execution

```
# Built-in Python interpreter
code_agent = Agent(
    name="programmer",
    model="gemini-2.0-flash",  # Code execution enabled
    instruction="Write and test Python code"
)
```

Custom Planners

```
# Advanced reasoning strategies
reasoning_planner = CustomPlanner(
    strategy="tree_of_thought",
    max_depth=5
)

agent = Agent(
    name="deep_reasoner",
    planner=reasoning_planner
)
```

🎯 Key Takeaways

1. **Streaming**: Real-time text (SSE) and voice/video (BIDI)
2. **MCP**: Universal tool protocol for interoperability
3. **A2A**: Distributed agent communication
4. **Multimodal**: Images, audio, video, documents
5. **Code Execution**: Built-in Python interpreter
6. **Custom Planners**: Advanced reasoning strategies

🔗 **Next**: Master Decision Frameworks (decision-frameworks.md) for choosing the right patterns.

Generated on 2025-10-21 09:03:13 from advanced-patterns.md

Source: Google ADK Training Hub