The Agent Communication Protocol (ACP) is an open standard designed to facilitate communication and interoperability between AI agents, regardless of their underlying frameworks, programming languages, or runtime environments. It aims to standardize agent-to-agent, agent-to-application, and human-to-agent communication.

Key features of ACP:

- **REST-based:** ACP leverages a RESTful interface for communication, making it compatible with existing web technologies.
- **Modality-agnostic:** It supports various communication modalities, including text, images, audio, and binary data.
- **Framework-neutral:** Agents built with different AI frameworks can interoperate seamlessly through ACP's standardized interface.
- **Streaming-capable:** ACP supports streaming interactions for real-time data exchange.
- **State-aware or stateless:** It can accommodate both stateful and stateless communication patterns.
- **Designed for async-first communication:** ACP prioritizes asynchronous communication for efficient agent collaboration.

How ACP works:

ACP utilizes a client-server architecture where agents are hosted behind an ACP server, which acts as a standardized interface. This server handles the translation of ACP messages to and from the formats understood by the individual agents. ACP clients can then send requests to these ACP-enabled agents, enabling communication and collaboration across diverse agent ecosystems.

Example of ACP code (Python):

Python

```python
import asyncio

from collections.abc import AsyncGenerator

from acp_sdk.models import Message
from acp_sdk.server import Context, RunYield, RunYieldResume, Server


server = Server()


@server.agent()
async def echo(input: list[Message], context: Context) -> AsyncGenerator[RunYield,
RunYieldResume]:
   """Echoes everything"""
   for message in input:
       await asyncio.sleep(0.5)
       yield {"thought": "I should echo everything"}
       await asyncio.sleep(0.5)
       yield message


if __name__ == "__main__":
   server.run()
```

This example demonstrates a simple "echo" agent using the ACP SDK. The `@server.agent()` decorator registers the `echo` function as an ACP agent, and the `server.run()` method starts the ACP server, making the agent accessible via the defined protocol.