

AI기반 데이터 분석 및 AI Agent 개발 과정

『2과목 :』 AI Agent 개발

2025.09.22-10.02(9일, 62시간)

Prepared by Daekyeong Kim

Ph.D.

1. 과정 소개
 2. Langchain 전 프롬프트 엔지니어링
 3. Langchain 기본
 4. AI 에이전트 개발
 5. AI 에이전트 프로토콜 : MCP와 A2A
- 성취도 평가
 - 미니프로젝트

『2-5』

AI 에이전트 프로토콜 : MCP와 A2A

- AI 에이전트 프로토콜, 클로드 MCP
- AI 에이전트 프로토콜, 구글 A2A



학습목표

- 이 AI 에이전트 프로토콜의 개념과 비즈니스 임팩트를 이해한다
- MCP와 A2A의 기술적 특징과 활용 방안을 구분할 수 있다

눈높이 체크

- AI 에이전트 프로토콜 기반 업무 효율화 방안 도출

1. AI 에이전트 프로토콜 - 클로드 MCP

MCP(Model Context Protocol) 핵심 개념

- MCP(Model Context Protocol)는 2024년 11월 25일 Anthropic이 오픈소스로 공개한 개방형 표준으로, 개발자들이 데이터 소스와 AI 기반 도구 간에 안전하고 양방향적인 연결을 구축할 수 있도록 하는 프로토콜

Introduction

Get started with the Model Context Protocol (MCP)

MCP is an open protocol that standardizes how applications provide context to large language models (LLMs).

Think of MCP like a USB-C port for AI applications



Growing list of pre-built integrations

LLM can directly plug into



Standardized way to build custom integrations

For AI applications



Open protocol free to implement and use

Open protocol free to implement and use



Flexibility to change between apps

And take your context with you



A standard interface
for connecting LLMs
with the world

MCP(Model Context Protocol) 핵심 개념

● 산업 현황과 필요성

◦ 기존 AI 시스템의 한계:

- 정보 사일로와 레거시 시스템에 갇힌 AI
- 각 데이터 소스마다 맞춤형 구현 필요
- $N \times M$ 데이터 통합 문제 (N 개 AI \times M 개 데이터 소스)
- 확장 가능한 연결 시스템 구축 어려움

◦ MCP의 해결 방안:

- 단일 프로토콜로 모든 데이터 소스 연결
- 파편화된 통합을 범용 표준으로 대체
- 프런티어 모델들의 더 나은 응답 생성 지원
- 개발 시간과 비용의 대폭 절감

1. AI 에이전트 프로토콜 - 클로드 MCP

| MCP 기술 아키텍처 심화 분석

- MCP 서버 (Data Provider) 구조
 - MCP 서버는 개발자가 MCP 서버를 통해 데이터를 노출하거나, 이러한 서버에 연결하는 AI 애플리케이션(MCP 클라이언트)을 구축할 수 있는 구조 Introducing the Model Context Protocol \ Anthropic를 제공합니다.
- 주요 구성 요소:

1. AI 에이전트 프로토콜 - 클로드 MCP

MCP 기술 아키텍처 심화 분석

MCP 서버 기본 구조 예시

```
class MCPServer:
```

```
    def __init__(self):
```

```
        self.resources = {} # 데이터 리소스 관리
```

```
        self.tools = {} # 실행 가능한 도구들
```

```
        self.prompts = {} # 준비된 프롬프트 템플릿
```

```
    def add_resource(self, name, config):
```

```
        """구조화된 데이터 추가"""
```

```
        self.resources[name] = {
```

```
            "type": "document_snippet|code_fragment|database_record",
```

```
            "content": config["data"],
```

```
            "permissions": config.get("permissions", ["read"]),
```

```
            "metadata": config.get("metadata", {})
```

```
        }
```

```
    def add_tool(self, name, function):
```

```
        """실행 가능한 함수 추가"""
```

```
        self.tools[name] = {
```

```
            "function": function,
```

```
            "description": function.__doc__,
```

```
            "parameters": self._extract_parameters(function)
```

```
        }
```


1. AI 에이전트 프로토콜 - 클로드 MCP

| MCP 기술 아키텍처 심화 분석

- 사전 구축된 MCP 서버 생태계
 - Anthropic은 Google Drive, Slack, GitHub, Git, Postgres, Puppeteer 등 인기 있는 엔터프라이즈 시스템을 위한 사전 구축된 MCP 서버들을 공유하고 있습니다



2. 실습

| 실습 23: MCP 서버 만들기

Node.js® 다운로드

FastMCP

`simple_mcp_server.py`

`simple_mcp_client.py`

Node.js 설치

- Node.js는 크롬 V8 엔진 위에서 동작하는 자바스크립트 실행 환경
- 설치 확인 방법:
 - 먼저 내 컴퓨터에 Node.js가 설치되어 있는지 확인해 봅시다.
 - MacOS: 터미널 열고 → `node --version`
 - Windows: Win + R → cmd 입력 후 실행창에서 `node --version`
- 버전 번호가 정상적으로 출력되면 이미 설치가 잘 되어 있는 겁니다
- 만약 `command not found` 또는 `'node' is not recognized` 같은 메시지가 뜬다면, Node.js를 새로 설치해야 합니다.
- 운영체제에 맞는 설치 파일을 받으세요.
- 추천 버전: 최신 LTS(Long-Term Support) 버전

『2-5』 AI 에이전트 프로토콜 : MCP와 A2A

- AI 에이전트 프로토콜, 클로드 MCP
- AI 에이전트 프로토콜, 구글 A2A



학습목표

- AI 에이전트의 핵심 개념과 역할을 이해합니다.
- Google의 A2A(Agent-to-Agent) 프로토콜의 주요 특징과 작동 방식을 학습합니다.
- A2A 프로토콜이 가져올 미래 기술 변화와 비즈니스 기회를 탐색합니다.

눈높이 체크

- Google의 A2A(Agent-to-Agent)

AI 에이전트란 무엇인가?

- A2A의 정의와 비전

- 2025년 4월 9일 Google이 발표한 Agent2Agent(A2A)는 AI 에이전트 간 통신, 안전한 정보 교환, 다양한 엔터프라이즈 플랫폼이나 애플리케이션 위에서의 활동 조율을 가능하게 하는 새로운 오픈 프로토콜
Announcing the Agent2Agent Protocol (A2A) - Google Developers Blog입니다.

AI 에이전트란 무엇인가?

- 정의: 특정 목표를 달성하기 위해 환경을 감지(Perceive)하고, 판단(Think)하며, 자율적으로 행동(Act)하는 소프트웨어 개체입니다.
- 핵심 3요소:
 - 센서(Sensor): 주변 환경으로부터 데이터를 수집합니다. (예: 사용자 입력, API 데이터, 센서 값)
 - 처리(Processor/Brain): 수집된 데이터를 바탕으로 최적의 행동을 결정합니다. (예: LLM, 머신러닝 모델)
 - 액추에이터(Actuator): 결정된 행동을 실행하여 환경에 영향을 줍니다. (예: API 호출, 메시지 전송, 기기 제어)
- 예시:
 - 단순한 에이전트: 스팸 메일 필터, 챗봇
 - 복잡한 에이전트: 자율 주행 자동차, 스마트 홈 비서, 자동화된 금융 거래 시스템

왜 지금 AI 에이전트가 중요한가?

- 복잡한 문제 해결: 인간의 개입 없이 여러 단계의 복잡한 작업을 자율적으로 처리할 수 있습니다.
- 초개인화 서비스: 사용자의 맥락과 의도를 깊이 이해하여 맞춤형 서비스를 제공합니다.
- 생산성 혁신: 반복적이고 시간이 많이 소요되는 업무를 자동화하여 인간이 더 창의적인 일에 집중할 수 있도록 돕니다.

A2A(Agent-to-Agent) 프로토콜 심화 개념

- 산업 파트너십과 생태계

- A2A는 Atlassian, Box, Cohere, Intuit, Langchain, MongoDB, PayPal, Salesforce, SAP, ServiceNow, UKG, Workday 등 50개 이상의 기술 파트너와 Accenture, BCG, Capgemini, Cognizant, Deloitte, HCLTech, Infosys, KPMG, McKinsey, PwC, TCS, Wipro 등 주요 서비스 제공업체의 지원과 기여를 받고 있습니다

A2A 핵심 설계 원칙

- 5가지 핵심 원칙:

- A2A 프로토콜 설계 시 5가지 핵심 원칙을 준수했습니다: 1) 에이전트 역량 수용, 2) 기존 표준 기반 구축, 3) 기본 보안, 4) 장기 실행 작업 지원, 5) 협업 가능

- 에이전트 능력 중심 설계

- 기존 도구 중심 vs A2A 에이전트 중심:
- 기존: 에이전트를 단순한 "도구"로 제한
- A2A: 메모리, 도구, 컨텍스트를 공유하지 않아도 자연스럽게 구조화되지 않은 방식으로 협업 가능

A2A 핵심 설계 원칙

- **표준 기술 스택 활용**

- A2A는 HTTP, SSE(Server-Sent Events), JSON-RPC 등 기존의 인기 있는 표준 위에 구축되어 기업이 이미 매일 사용하는 기존 IT 스택과 통합하기 쉽습니다. JSON-RPC 2.0을 HTTP(S) 위에서 핵심 통신 방법으로 사용합니다 Google DevelopersTowards Data Science.

- **엔터프라이즈급 보안**

- A2A는 런치 시 OpenAPI의 인증 스킴과 동등한 엔터프라이즈급 인증 및 인가를 지원하도록 설계되었습니다 Announcing the Agent2Agent Protocol (A2A) - Google Developers Blog.

| Agent Card (에이전트 카드)

- 에이전트들은 JSON 형식의 "Agent Card"를 사용하여 자신의 능력을 광고할 수 있어, 클라이언트 에이전트가 작업을 수행할 수 있는 최고의 에이전트를 식별하고 A2A를 활용해 원격 에이전트와 통신할 수 있습니다

1) Agent Card (에이전트 카드)

```
{
  "agent_card": {
    "name": "Financial Analysis Agent",
    "version": "2.1.0",
    "description": "전문적인 재무 분석 및 예측 서비스",
    "capabilities": {
      "skills": [
        "financial_modeling",
        "risk_assessment",
        "market_analysis",
        "budget_planning"
      ],
      "input_modalities": ["text", "spreadsheet", "json"],
      "output_modalities": ["text", "charts", "reports", "forms"]
    },
    "endpoints": {
      "base_url": "https://finance-agent.company.com",
      "health_check": "/health",
      "capabilities": "/capabilities"
    },
    "authentication": {
      "scheme": "bearer_token",
      "required": true
    },
    "sla": {
      "response_time": "< 30s",
      "availability": "99.9%"
    }
  }
}
```

Task Management (작업 관리)

- 클라이언트와 원격 에이전트 간의 통신은 작업 완료 지향적이며, 에이전트들이 최종 사용자 요청을 이행하기 위해 작업합니다. 프로토콜에 의해 정의된 "작업" 객체는 생명주기를 가지며, 즉시 완료되거나 장기 실행 작업의 경우 각 에이전트가 작업 완료의 최신 상태에 대해 서로 동기화를 유지할 수 있습니다 [Announcing the Agent2Agent Protocol \(A2A\)](#) - Google Developers Blog.

Task Management (작업 관리)

- 작업 생명주기:

- 작업 상태 전이:

- submitted → accepted → in_progress → completed/failed/cancelled

- 각 상태별 특징:

- submitted: 작업 요청 제출됨
- accepted: 에이전트가 작업 수락
- in_progress: 작업 진행 중 (진행률 업데이트 가능)
- completed: 작업 완료 (결과물 포함)
- failed: 작업 실패 (에러 정보 포함)
- cancelled: 작업 취소됨

| Message Structure (메시지 구조)

- 에이전트들은 컨텍스트, 응답, 아티팩트, 사용자 지시사항을 전달하기 위해 서로 메시지를 보낼 수 있습니다. 각 메시지는 생성된 이미지와 같은 완전히 형성된 콘텐츠 조각인 "parts"가 포함됩니다

Message Structure (메시지 구조)

```
{
  "message": {
    "id": "msg_2024_001234",
    "task_id": "task_2024_567890",
    "sender": "marketing_agent",
    "receiver": "analytics_agent",
    "timestamp": "2024-12-31T10:30:00Z",
    "parts": [
      {
        "type": "text",
        "content": "지난 분기 캠페인 성과를 분석해주세요"
      },
      {
        "type": "data",
        "content": {
          "format": "json",
          "data": {...}
        }
      }
    ]
  }
}
```

3. 소통의 표준, 에이전트 프로토콜

| 프로토콜의 필요성

- "똑똑한 AI 비서가 여러 명 있다면, 그들끼리 어떻게 대화하고 협력할 수 있을까?" 라는 질문에서 시작합니다.
- 문제점: 제조사나 개발사마다 AI 에이전트의 '언어'와 '행동 방식'이 다르면 서로 협력할 수 없습니다. 마치 다른 언어를 쓰는 사람들처럼 말이죠.
- 해결책: 에이전트들이 서로의 존재를 인지하고, 능력을 파악하며, 작업을 요청하고 결과를 받을 수 있는 **표준화된 소통 규약(Protocol)**이 필요합니다.
- 인터넷과의 비유:
 - HTTP: 웹 브라우저와 서버가 정보를 주고받는 표준 규칙
 - SMTP: 이메일 서버들이 메일을 주고받는 표준 규칙
 - A2A 프로토콜: AI 에이전트들이 서로 소통하고 협력하는 표준 규칙

3. 소통의 표준, 에이전트 프로토콜

| Google의 A2A (Agent-to-Agent) 프로토콜

- **개념**: Google이 제안하는 개방형 AI 에이전트 간 통신 프로토콜입니다.
- **비전**: 특정 플랫폼이나 회사에 종속되지 않고, 전 세계의 AI 에이전트들이 서로 원활하게 협력하는 '에이전트 생태계'를 구축하는 것을 목표로 합니다.
- **주요 특징**:
 - 상호운용성(Interoperability): 서로 다른 개발사가 만든 에이전트도 A2A 프로토콜을 따르면 원활하게 통신할 수 있습니다.
 - 작업 위임(Task Delegation): 자신의 능력을 벗어나는 작업은 해당 작업을 가장 잘 처리할 수 있는 다른 에이전트에게 위임할 수 있습니다.
 - 에이전트 탐색(Discovery): 특정 작업을 수행할 수 있는 다른 에이전트를 찾고 발견하는 메커니즘을 제공합니다.



4. A2A 프로토콜 심층 분석

A2A 프로토콜의 핵심 구성 요소

- 에이전트 식별(Identity): 각 에이전트는 고유한 주소나 ID를 통해 자신을 식별합니다.
- 기능 명세(Capability Schema): 각 에이전트는 자신이 수행할 수 있는 작업(예: 항공권 예약, 날씨 정보 제공)의 종류와 필요한 정보(파라미터)를 표준화된 형식으로 정의하고 공개합니다.
- 의사소통 언어(Communication Language): 작업 요청, 정보 전달, 결과 보고 등을 위한 메시지 형식을 표준화합니다. (예: JSON, gRPC 기반)
- 실행 흐름(Execution Flow): 사용자가 에이전트 A에게 작업을 요청했을 때, 에이전트 A가 에이전트 B에게 하위 작업을 위임하고 결과를 받아 다시 사용자에게 전달하는 일련의 과정을 정의합니다.



4. A2A 프로토콜 심층 분석

A2A 작동 시나리오 예시: "제주도 여행 계획"

- 사용자: (나의 개인 비서 에이전트에게) "다음 주 금요일에 출발하는 2박 3일 제주도 여행 계획 좀 짜줘."
- 개인 비서 에이전트 (A):
 - 사용자의 요청을 분석합니다. (항공, 숙박, 렌터카 예약 필요)
 - A2A 네트워크에서 각 작업을 가장 잘 수행할 전문 에이전트를 탐색합니다.
- 작업 위임:
 - A -> 항공권 예약 에이전트 (B): "다음 주 금요일 출발, 일요일 도착 제주행 1인 항공권 최저가 검색 및 예약 요청"
 - A -> 숙소 추천 에이전트 (C): "제주공항 근처, 2박, 1인, 평점 4.5 이상 호텔 검색 및 추천 요청"
 - A -> 렌터카 예약 에이전트 (D): "제주공항에서 3일간 소형차 렌트 최저가 검색 및 예약 요청"
- 결과 취합:
 - B, C, D 에이전트는 각각 작업을 수행하고 결과를 표준화된 형식으로 A에게 회신합니다.

5. A2A가 바꿀 미래와 우리의 준비

| 비즈니스 및 산업에 미치는 영향

- 비스의 결합: 단일 앱이나 서비스가 아닌, 여러 에이전트가 협력하여 사용자에게 끊김 없는(Seamless) 통합 경험을 제공합니다.
- 새로운 비즈니스 모델: 특정 기능에 고도로 특화된 '전문 에이전트'를 개발하고 판매하는 시장이 형성될 수 있습니다.
(예: 법률 자문 에이전트, 의료 진단 보조 에이전트)
- 하이퍼-오토메이션(Hyper-automation): 기업의 복잡한 워크플로우(예: 공급망 관리, 고객 지원)가 다수의 전문 에이전트 협력을 통해 완전히 자동화될 수 있습니다.

5. A2A가 바꿀 미래와 우리의 준비

| 우리가 준비해야 할 것

- 기술적 관점: 표준 프로토콜에 대한 이해와 이를 기반으로 에이전트를 개발할 수 있는 역량이 필요합니다.
- 기획적 관점: 단일 기능이 아닌, 다른 에이전트와 어떻게 '협력'하여 더 큰 가치를 만들 수 있을지 고민하는 생태계적 사고가 중요합니다.
- 보안 및 신뢰: 에이전트 간의 데이터 교환 시 발생할 수 있는 보안 문제, 개인정보 보호, 그리고 위임된 작업의 신뢰성을 확보하는 방안에 대한 고민이 필요합니다.

THANK YOU.

앞으로의 엔지니어는 단순한 '코더'나 '기계 조작자'가 아니라 뇌-기계 인터페이스를 통해 지식과 능력을 즉각 확장하는 존재(뉴로-인터페이스: Neuro Interface)가 될 수 있습니다.

- 🎯 목표 달성을 위한 여정이 시작됩니다.
- 🌟 궁금한 점이 있으시면 언제든지 문의해주세요!
- 🚀 함께 코더와 프롬프트 전문가로 성장해 나갑시다!

