Software
Architecture

데이터가 모여 있는 문제.

Data-Centered (DC) Software Architectures

Eunmi Choi
Kookmin University

전까지는 데이터가 흐르는 문제.

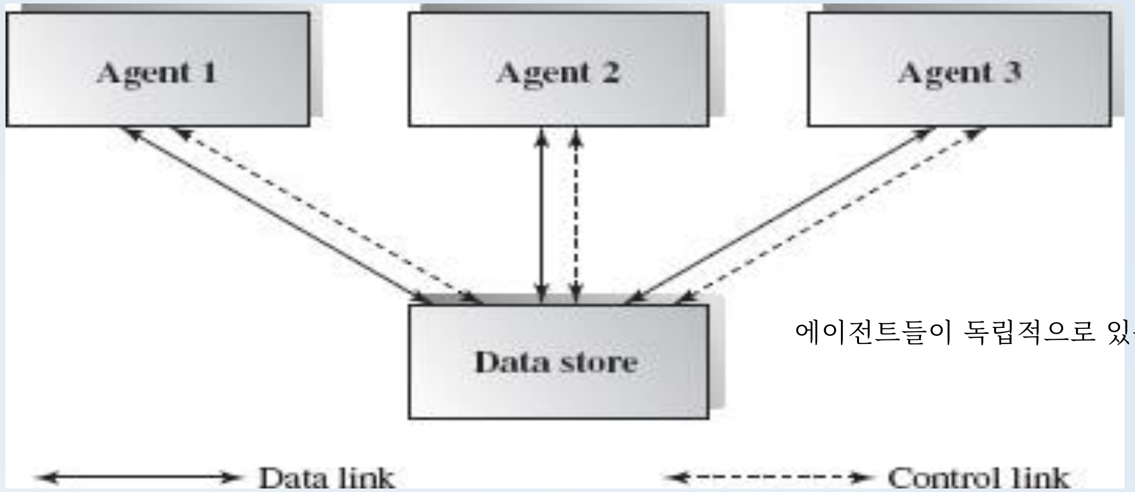
Data-Centered (DC) Software Architectures

데이터의 스토리지가 있어서 거기에 데이터를 가지고 있고 가지고 있는 데이터 주위에 프로그램이나 컴포넌트 들이 있어서 데이터를 공유하는 구조. (데이터가 모여있는 문제)
ex) DB store : 저장, 읽기, 수정, 삭제 등등.
데이터가 모여있고 프로그램이 접근해 원하는 데이터를 처리함. 이럴 경우 커넥션은 SQL 을 통해 함.

- DC is characterized by a centralized data store that is shared by all surrounding software components.
- Structure
  - decomposed into two major partitions:
    - data store
    - independent software component or agents
  - The connections between the data module and the software components are implemented either by explicit method invocation or by implicit method invocation.

Data-Centered Software Architectures

- Two categories of data-centered architecture
  - *Repository vs. Blackboard:*
    - differentiated by the flow control strategy.



에이전트들이 독립적으로 있음. 서로는 서로를 몰라도 됨.

Repository

데이터가 있는 곳이 수동적 = 누군가 프로그램이 데이터를 넣어주고 그 데이터는 persistence한 것.
데이터가 수동적으로 repository에 있고 누군가 필요한 사람이 있으면 데이터를 가져감.
데이터를 접근하는 쪽에서 active하게 read와 write를 하고, repository가 데이터를 passive하게 갖고 있음.

데이터에 대한흐름도는 클라이언트에게 있음.

- Structure
  - The data store in the repository architecture is passive, and clients of the data store are active; that is, clients (software components or agents) control the logic flow.
  - Clients may access the repository interactively or by a batch transaction request.
- Application Examples
  - database management systems
  - library information systems
  - interface repository (IR) in CORBA
  - UDDI registry for web services
  - compilers
  - Computer Aided Software Engineering (CASE) environments like Rational Rose
    - It supports a graphic editor to draw UML diagrams, generates various programming code, and provides reverse engineering functionality to generate graphic diagrams from code.
  - Interactive Development Environments (IDE)
  - software development kits
  - complex information management systems

이런게 있음.

Blackboard

칠판. 데이터를 가지고있는 곳이 능동적으로일을 처리함. 데이터를갖고 있는 쪽이 active하게일 처리를 하고 클라이언트 쪽이 passive하게 일을 처리.
데이터의내용에 따라 로직이 달라짐. non deterministic problem을 푸는데 좋다.

deterministic(결정적인, 모든경우에동일한 절차를 가짐. 프로그램을돌리면 항상 그결과가 나옴. 절차가 정해져 있음.) < - >
nondeterministic (어떤일이 일어날지 정해지지 않음.)

- Structure
  - The data store is active, and its clients are passive; thus, the flow of logic is determined by the current data status in the data store.
  - The clients of a blackboard are called knowledge sources, listeners, or subscribers.

필요할 때마다자기 자신이 그 문제를풀고
데이터를 변경해 나감

  - A new data change may trigger events so that the knowledge sources take actions to respond to these events.
  - These actions may result in new data, which may in turn change the logic flow; this could happen continuously until a goal is reached.
- Application Examples
  - knowledge-based AI systems
  - voice and image recognition systems
  - security systems
  - business resource management systems,

이런걸 풀때 좋음.
해답이없는걸 풀 때 좋은구조임 .