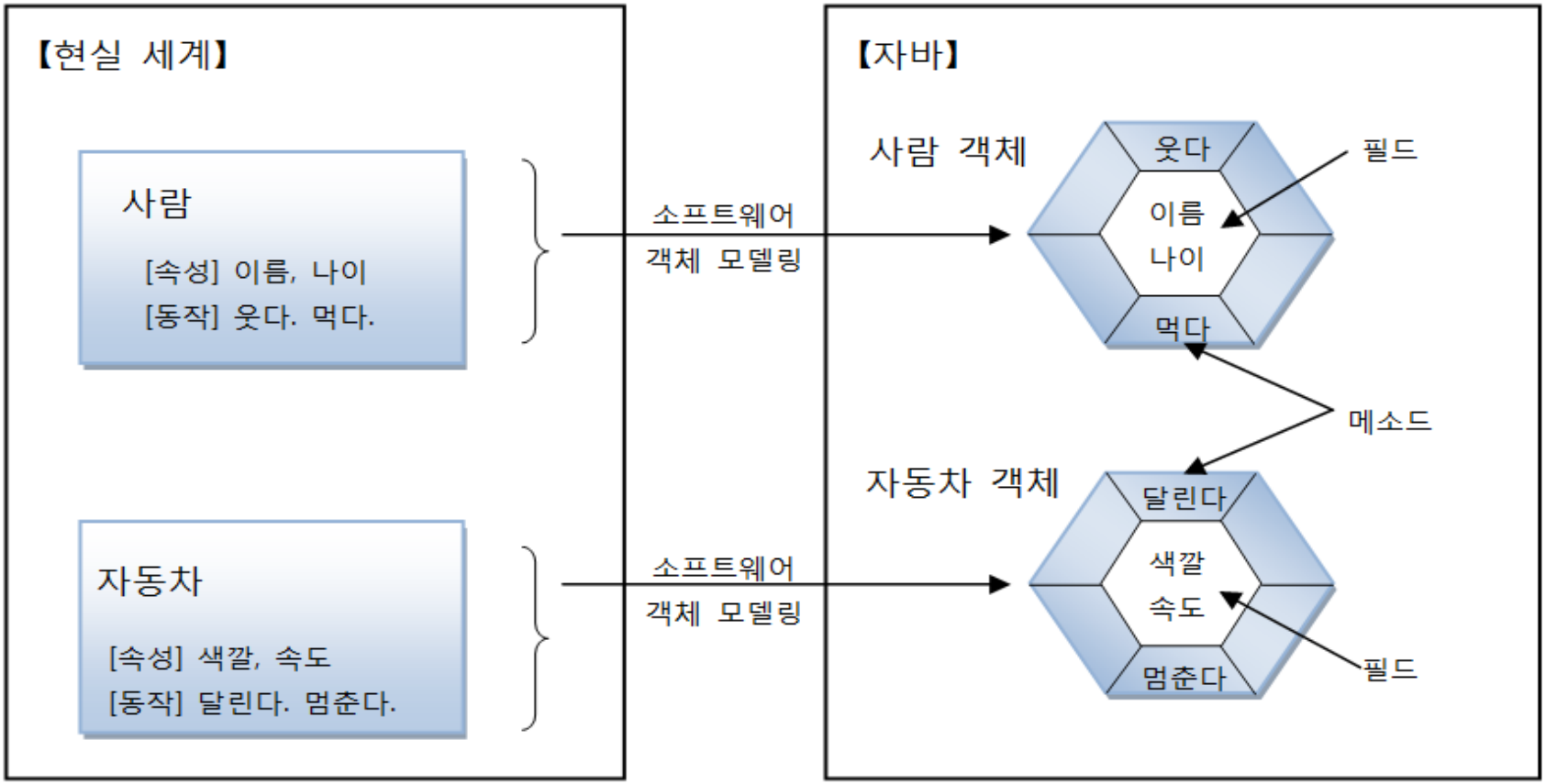
객체 지향 프로그래밍

❖ 객체 지향 프로그래밍

- OOP: Object Oriented Programming
- 부품 객체를 먼저 만들고 이것들을 하나씩 조립해 완성된 프로그램을 만드는 기법

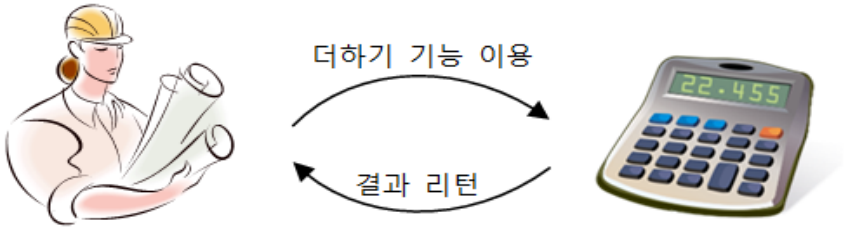
❖ 객체(Object)란?

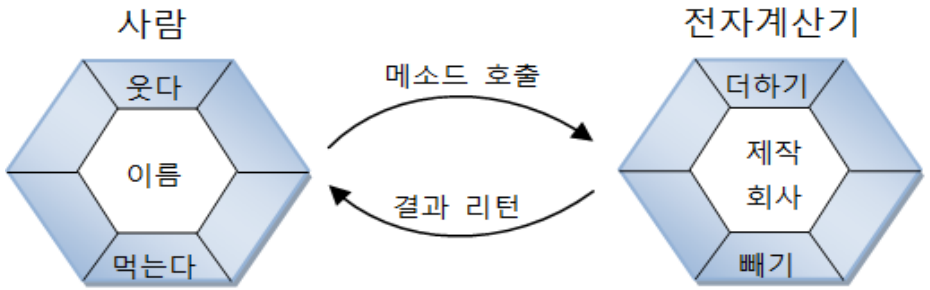
- 물리적으로 존재하는 것 (자동차, 책, 사람)
- 추상적인 것(회사, 날짜) 중에서 자신의 속성과 동작을 가지는 모든 것
- 객체는 필드(속성)과 메소드(동작)로 구성된 자바 객체로 모델링 가능

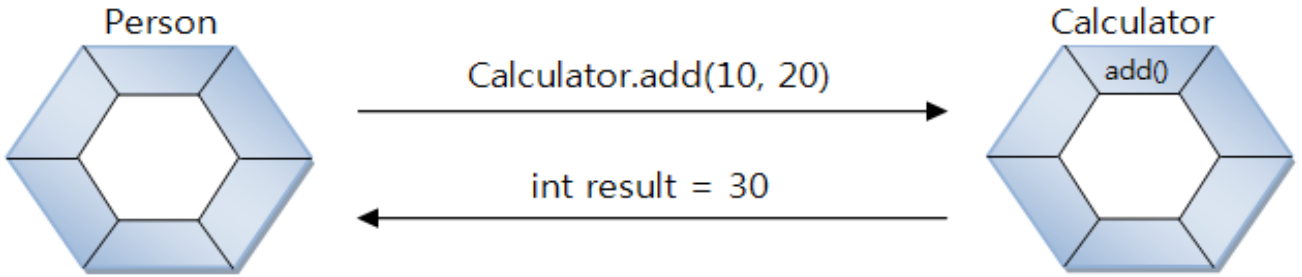


❖ 객체의 상호 작용

- 객체들은 서로 간에 기능(동작)을 이용하고 데이터를 주고 받음





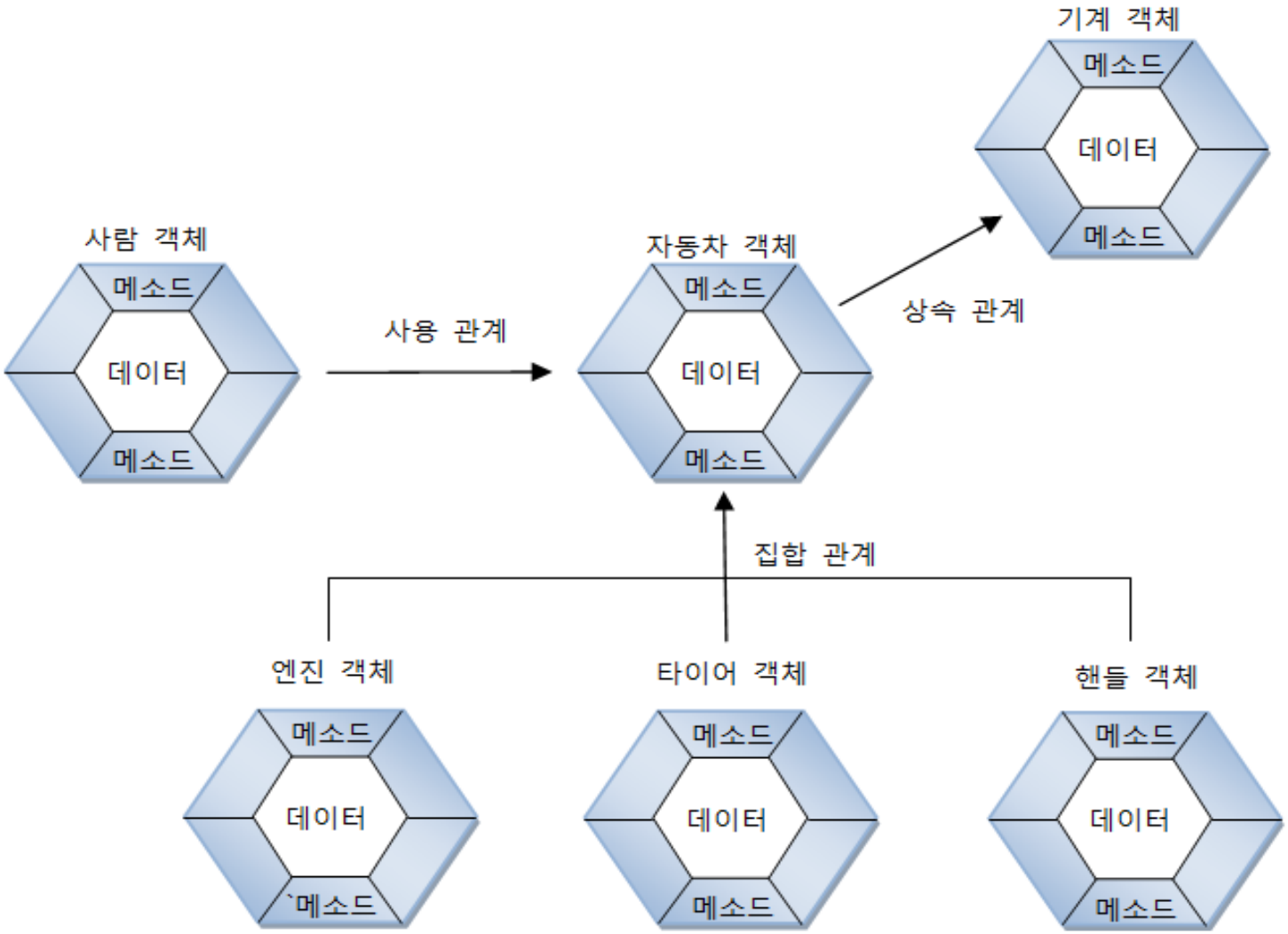


❖ 객체간의 관계

- 객체 지향 프로그램에서는 객체는 다른 객체와 관계를 맺음
- 관계의 종류
  - 집합 관계: 완성품과 부품의 관계
  - 사용 관계: 객체가 다른 객체를 사용하는 관계
  - 상속 관계: 종류 객체와 구체적인 사물 객체 관계

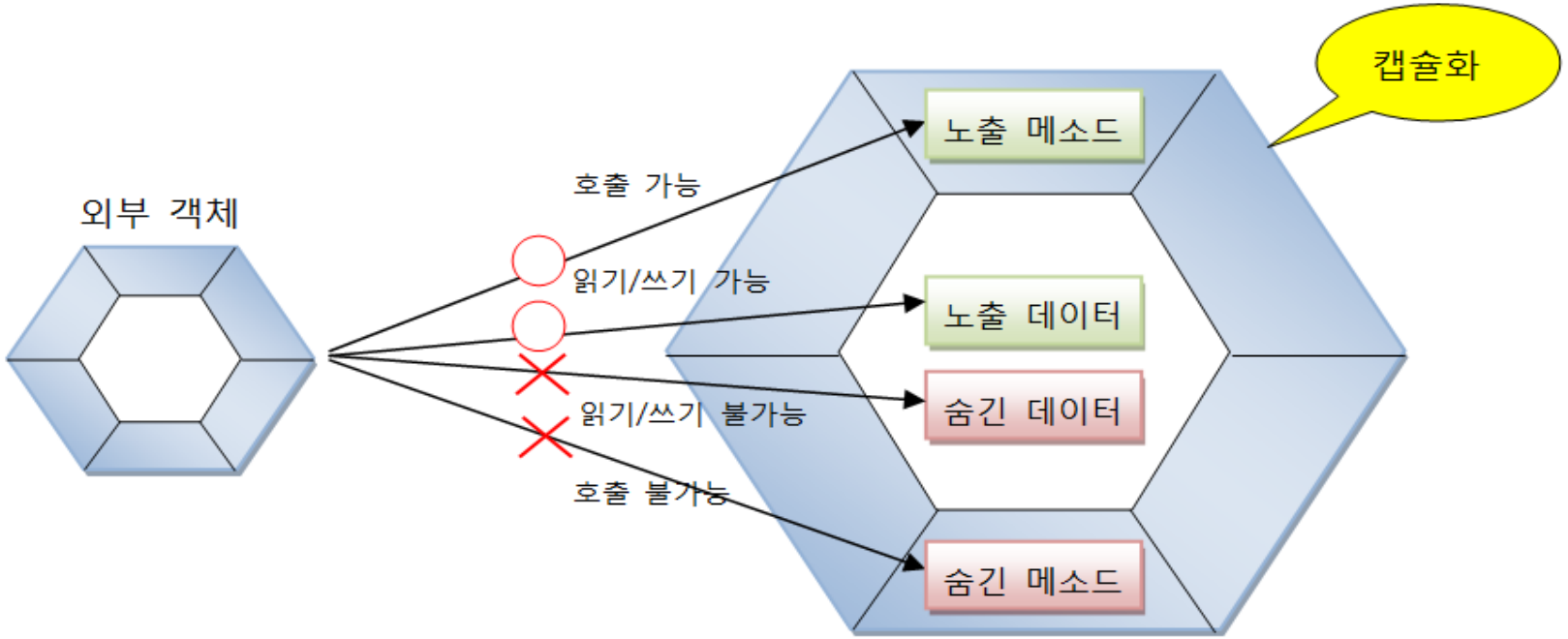
객체 지향 프로그래밍

❖ 객체간의 관계



❖ 객체 지향 프로그래밍의 특징

- ㅇ 캡슐화
  - 객체의 필드, 메소드를 하나로 묶고, 실제 구현 내용을 감추는 것
  - 외부 객체는 객체 내부 구조를 알지 못하며 객체가 노출해 제공하는 필드와 메소드만 이용 가능
  - 필드와 메소드를 캡슐화하여 보호하는 이유는 외부의 잘못된 사용으로 인해 객체가 손상되지 않도록
  - 자바 언어는 캡슐화된 멤버를 노출시킬 것인지 숨길 것인지 결정하기 위해 접근 제한자(Access Modifier) 사용



❖ 객체 지향 프로그래밍의 특징

- 상속
  - 상위(부모) 객체의 필드와 메소드를 하위(자식) 객체에게 물려주는 행위
  - 하위 객체는 상위 객체를 확장해서 추가적인 필드와 메소드를 가질 수 있음
  - 상속 대상: 필드와 메소드

❖ 객체 지향 프로그래밍의 특징

- ㅇ 상속
  - ▪ 상속의 효과
    - 상위 객체를 재사용해서 하위 객체를 빨리 개발 가능
    - 반복된 코드의 중복을 줄임
    - 유지 보수의 편리성 제공
    - 객체의 다형성 구현