Platform Development - Variability Realization

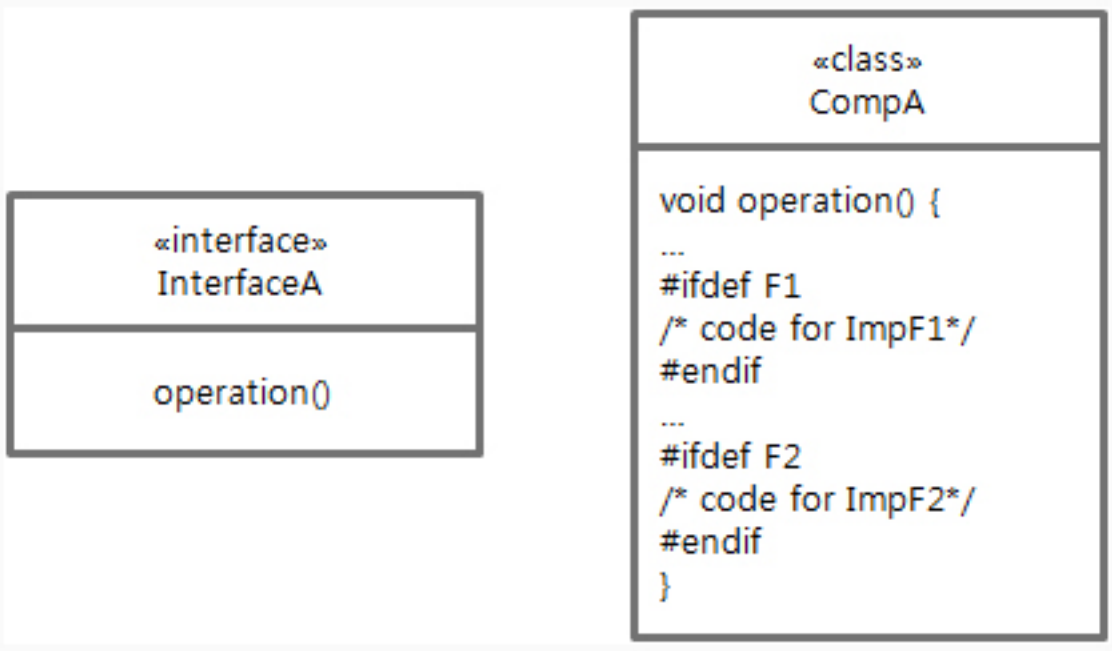
“ SPLE (Software Product Line Engineering)는 다수의 유사 제품 (또는 시스템)들에 대한 공통 플랫폼을 개발하고, 이를 재사용하여 제품을 개발함으로써, 설계 및 생산 (또는 구현)의 품질 및 생산성을 제고하는 기법입니다. ”

이러한 SPLE의 장점을 극대화하기 위해서는 도메인 지식을 바탕으로 제품군에 대한 공통성 (Commonality)과 가변성 (Variability)을 식별하여 플랫폼 상에 반영해야 합니다. 가변성은 소프트웨어 개발의 모든 단계에서 정의되고 추적이 되어야 하는데, 특히, 아키텍처 설계 시 최적의 가변성 구현 방식(Variability Realization)을 선정하는 것이 중요합니다. 다음은 이러한 가변성 구현 방식에 대한 주요한 기법들입니다.

Parameters

소스코드, 컨피규레이션 파일 상에서 파라미터로 가변성을 구현합니다. 컨피규레이션 파일(.ini, .conf)에서 파라미터 값을 정의하거나, 코드 상 #define, Global Variable 형태로 구현하며 파라미터에 따라 실행 시간에 제어 로직(if/else, case)을 달리 수행하는 형태로, 파라미터가 바뀔 때 마다 관련된 제어로직을 수정하고 코드를 재 컴파일 해야 가변성이 반영됩니다.

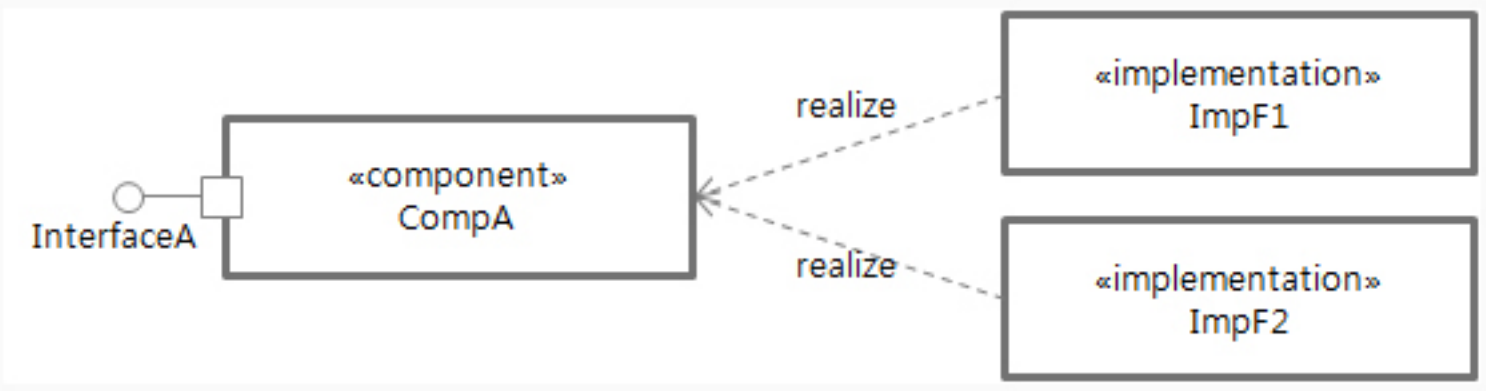
■ Conditional Compilation (Build Time Binding)

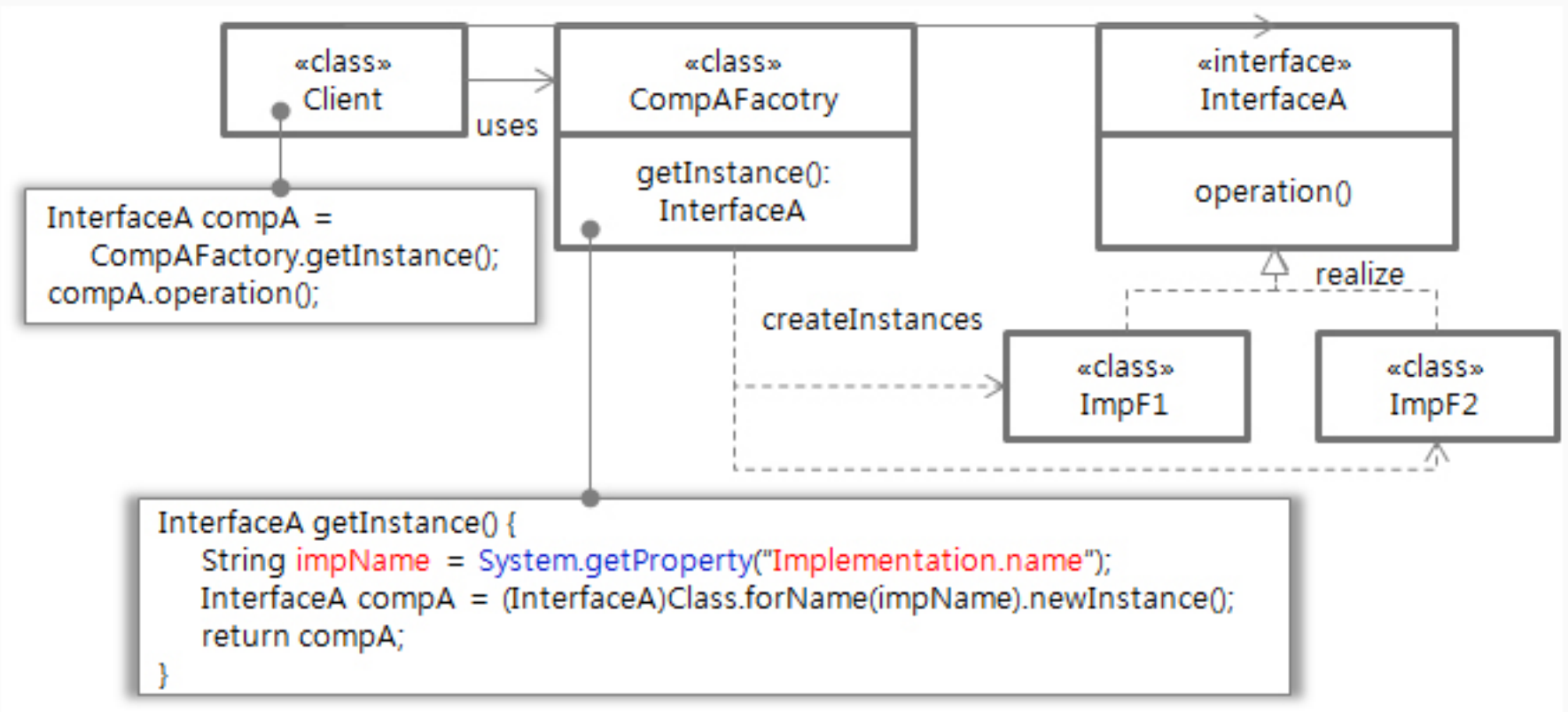


Design patterns

모듈 설계 시 잘 알려진 디자인 패턴 (Observer, Template-Method, Strategy 등)을 적용하여 가변성을 구현합니다.추상화 클래스를 정의하여 이를 상속하는 서브 클래스로 가변성을 구현하는 형태로, 파라미터 방식 제어 로직을 구조화한 형태로 볼 수 있습니다. 추상화 클래스로 정의되는 기본 (공용) 코드의 사용으로 파라미터 방식 대비 가변성을 쉽게 확장할 수 있습니다.

■ Component Factory Pattern (Dynamic Binding)





Frameworks

기본 기능을 제공하는 프레임워크 코드 위에 플러그인 형태로 가변성을 구현합니다. 디자인 패턴(Template-Method)을 적용 하여 구현될 수 있으며 이때 추상화 부분은 프레임워크에서, 가변성 구현부는 플러그인에서 분리되어 개발되는 형태이므로 프레임워크 코드에 대해서는 재사용성을 제공합니다. 플러그인의 사용 여부가 일반적으로 로딩 타임에 정의되므로 가변성 역시 로딩 타임에 반영됩니다.

Components & Services

프레임워크 방식과 유사하나 별도 동작하는 컴포넌트(COM) 형태로 가변성을 구현합니다. 플러그인은 프레임워크 코드에 종속적으로 개발되는데 비해 컴포넌트는 상호 운용 인터페이스만 정의되면 독립적으로 개발되는 차이점이 있으며, 독립 실행되는 형태로 런타임에 가변성이 반영됩니다.