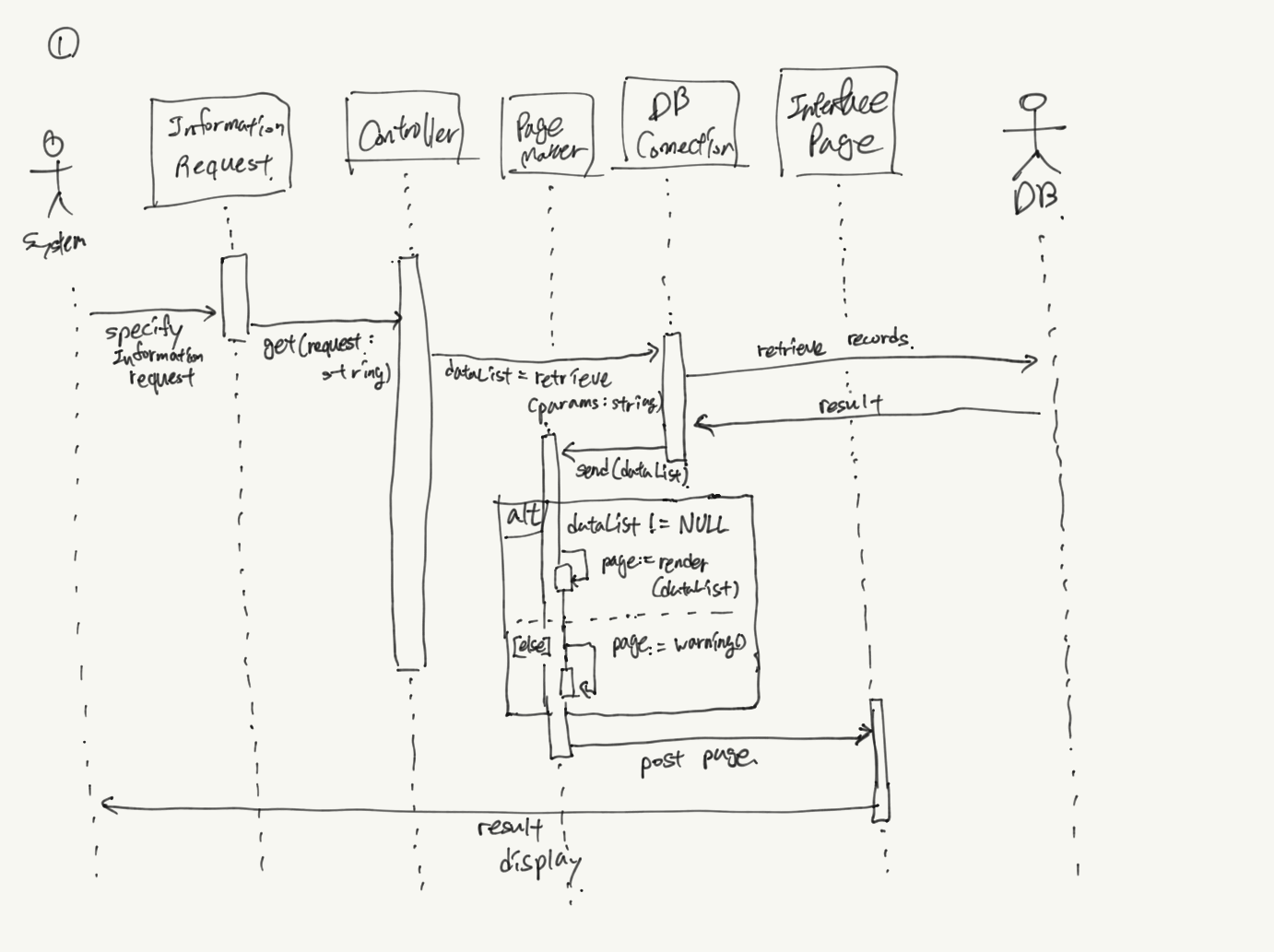
Object sequence diagram : Display 1차 구상

1차 구상시에는 RAD에서 설계하였던 대로 domain model을 사용하여 시간의 흐름에 따라 각 object들이 어떠한 작업을 할지 data와 function의 흐름을 그렸습니다.

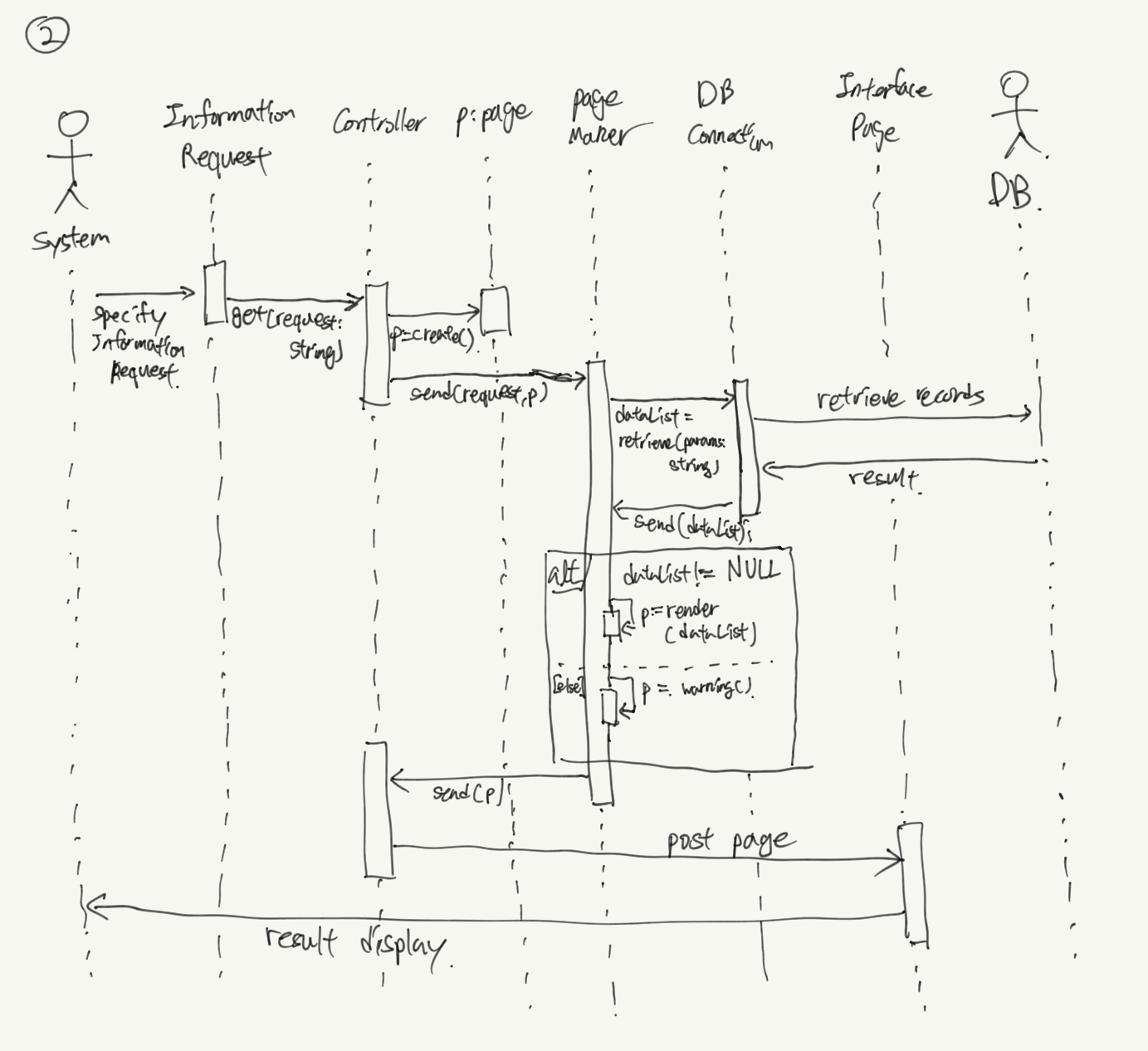
1차 구상에서는 시간의 흐름에 따라 어떻게 진행될지에 초점을 맞춰서 sequence diagram을 작성하다보니 각 object별로 모듈화도 되어있지않고 번잡하게 되어있습니다.

system에서 Information Request로 request를 특정하면 controller로 전달합니다.

Controller에서 DB connection으로 Page 생성시에 사용 될 dataList를 검색해오라 전달하고, DB connection은 DB에서 정보를 찾아와 결과를 받아옵니다.

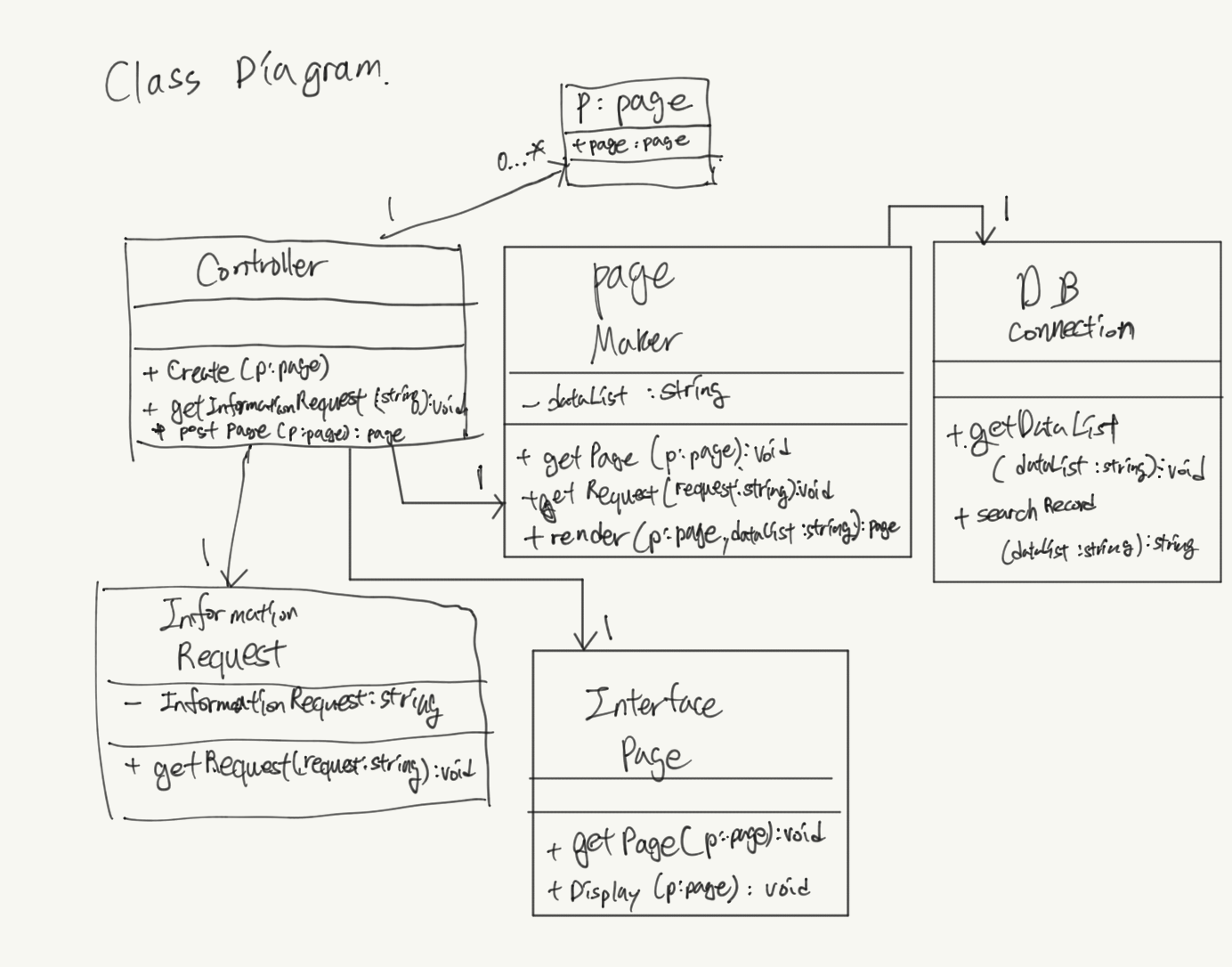
받아온 결과는 PageMaker에서 처리하여 if-then-else로 데이터가 있으면 render, 없으면 error 메시지를 페이지에 표시하여 줍니다.

이 페이지는 interfacePage로 전달이된 후 system에게 display됩니다.

Object sequence diagram : Display 최종 구상

이번 구상 단계에서는 시간의 흐름에 따른 object들의 communication에 초점을 맞추기와는 달리 각 object들의 module화와 이번에 사용할 Design Patterns인 Pub-Sub pattern을 적용하여 작성하였습니다.

controller에서 대부분의 일을 처리하는 방식이 아닌, Controller는 information request를 받아옴과 동시에 page를 create하는 작업만 진행하고, 나머지 작업은 PageMaker에서 진행되도록 작성하여 1차 구상보다 훨씬 간결하면서 각 object들의 module화도 잘 되어있는 것을 볼 수 있습니다. 각 object들의 communication은 1차 구상에 기술한 것과 동일합니다.

Class Diagram

RAD와 Object Sequence Diagram에서 작성한 것들을 토대로 그린 Class Diagram입니다.

각 Object들이 가지고 있어야 할 attirbute와 function을 나타내었고, SOLID원칙을 적용하여 설계하였습니다.

Design Principles에 따라 각 Object들의 모듈화를 통해 Expert Doer를 수행할 수 있도록 하였고, 그렇게함으로써 Cohesion이 높아지게 설계하고, Dependency를 줄여 Coupling을 낮게 만들기 위하여 설계하였습니다.