Clase 2: Introducción al diseño de software

Rodrigo Arturo Saffie Kattan

Pontificia Universidad Católica de Chile

rasaffie@ing.puc.cl

4 de agosto de 2016

Contenidos

Conceptos Abordados

- ¿Qué es el *software*?
- *Legacy software*
- Procesos de un proyecto

Conceptos Abordados

Procesos de un proyecto:

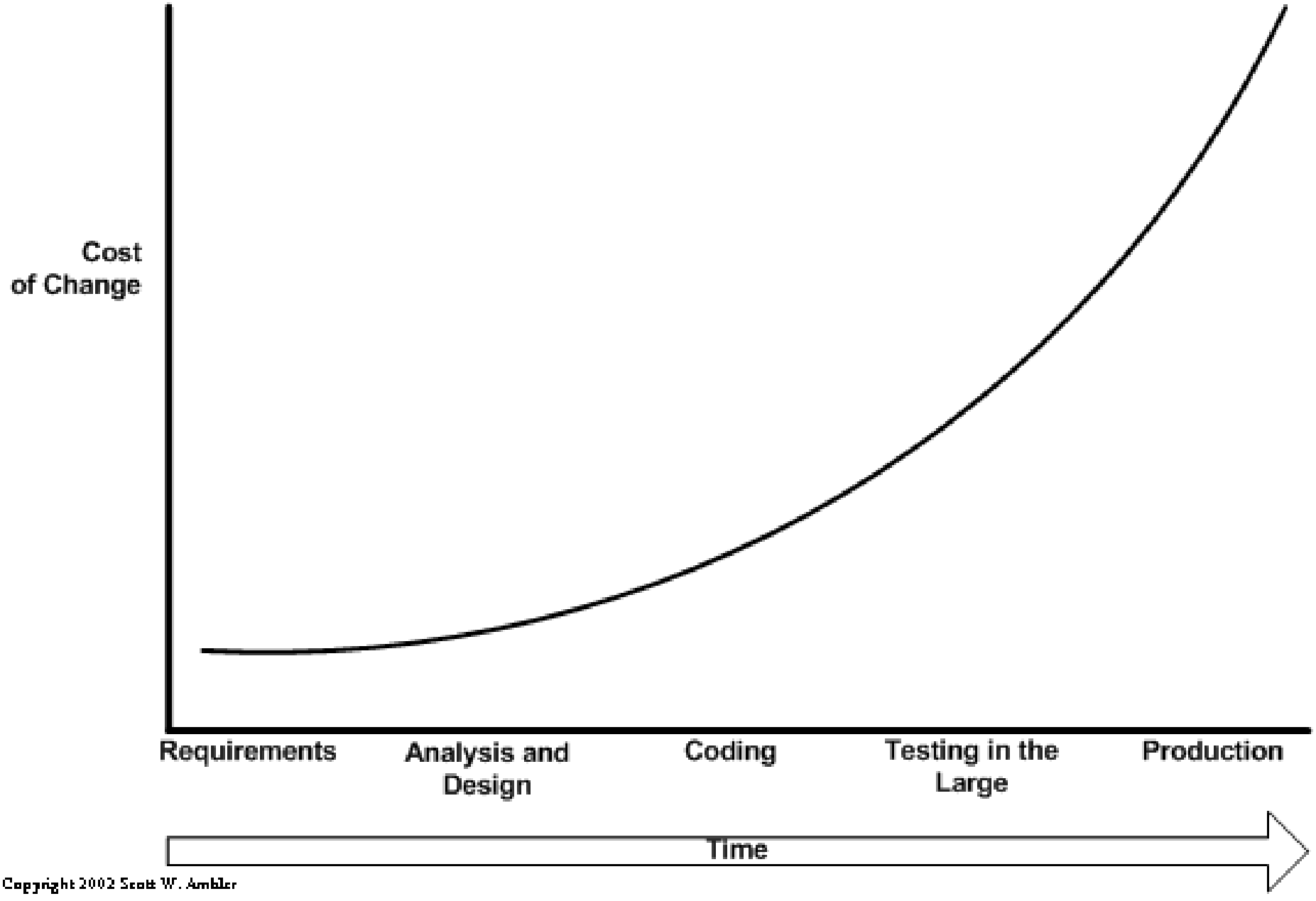
- Comunicación
- Planificación
- Modelación
- Construcción
- Implementación

Metodologías de Desarrollo

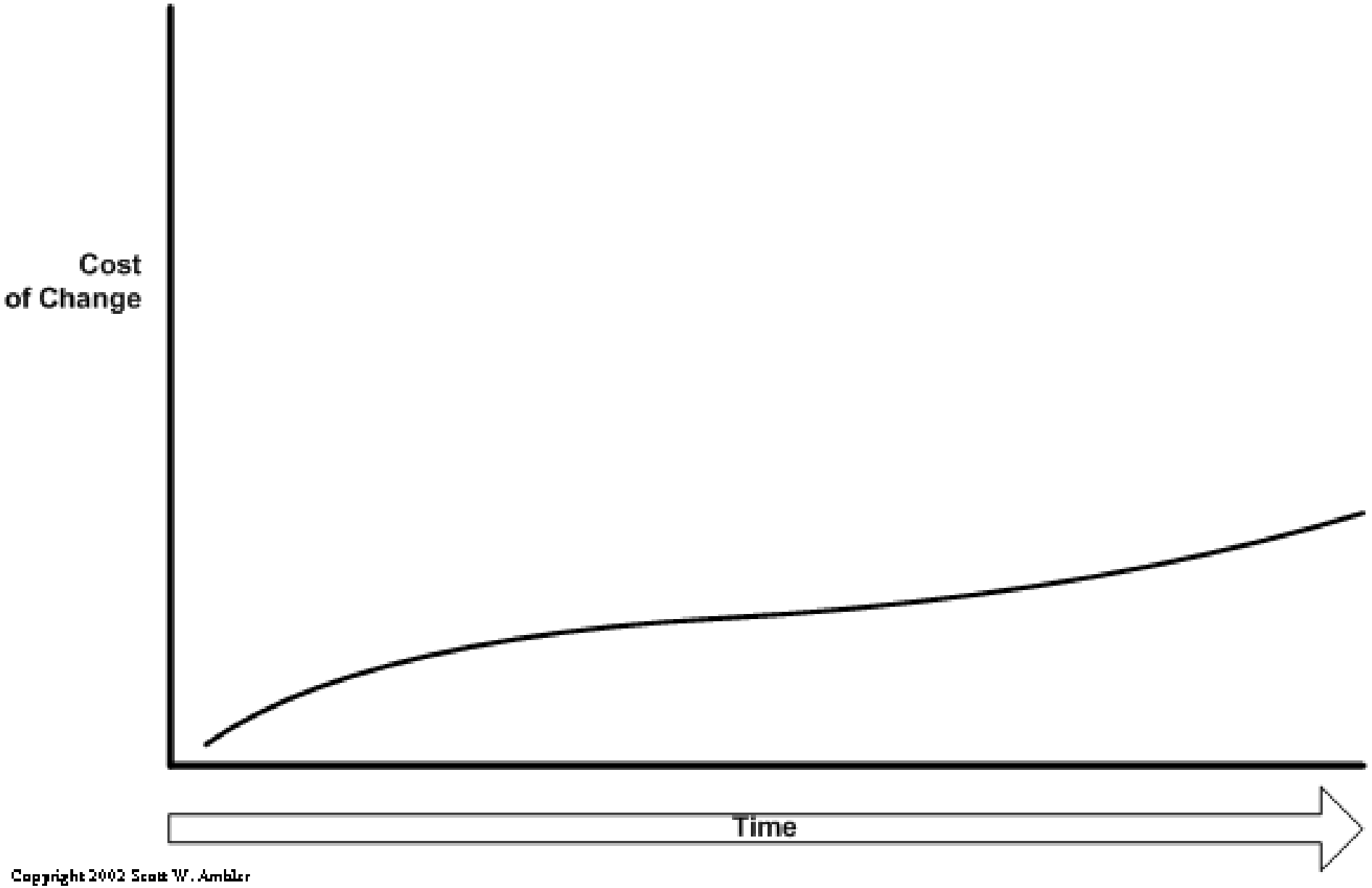
Algunas metodologías de desarrollo:

- Cascada
- Espiral
- Iterativo Incremental
- Agile

Costo Tradicional



Costo Ágil



¿Qué es el diseño de software?

- Un concepto reciente (70 años)
- Se define principalmente entre la **Modelación** y la **Construcción**
- Traduce los requisitos a especificaciones técnicas

¿Qué es el diseño de software?

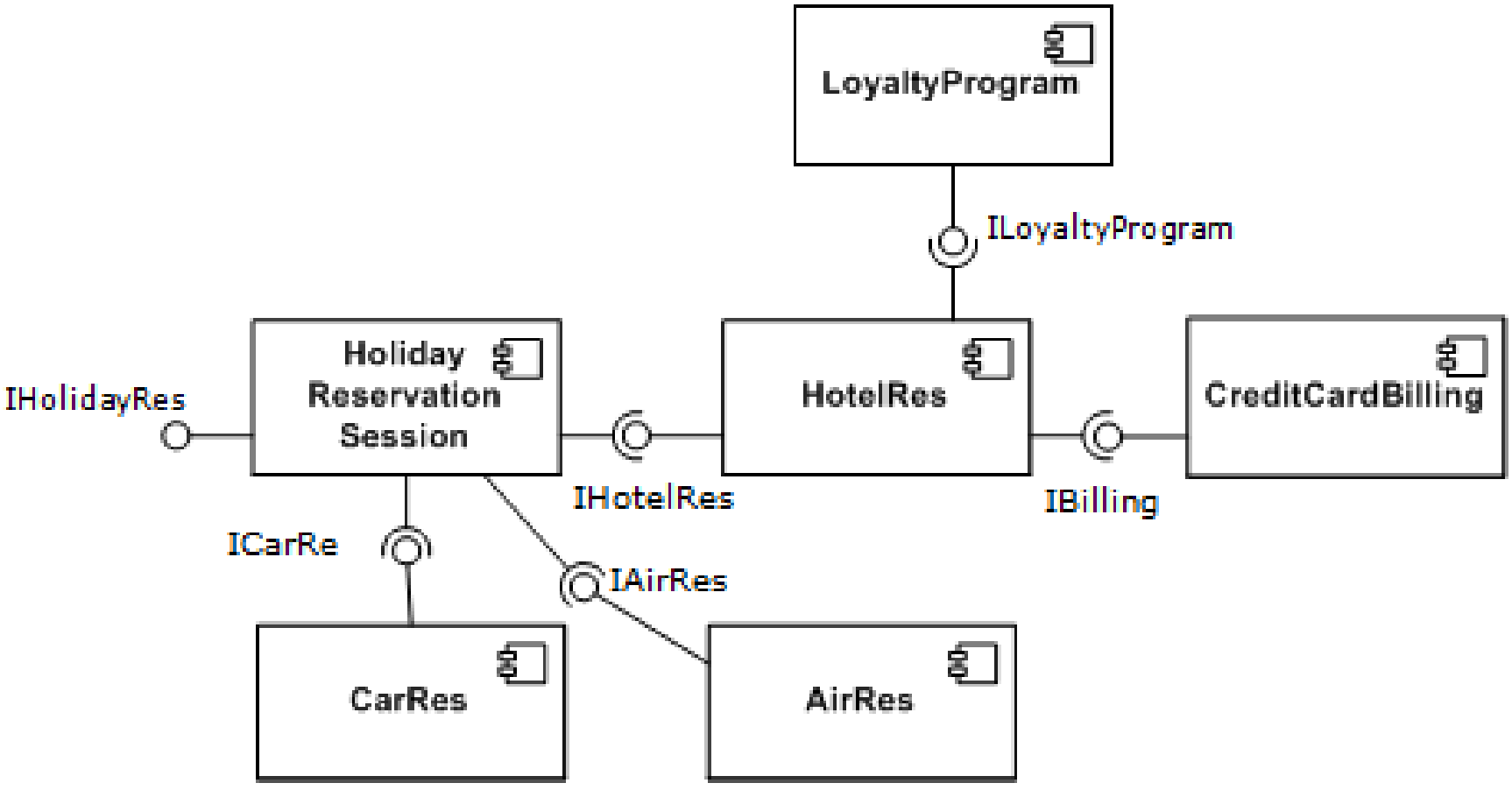
El diseño se divide en 4 áreas:

- Diseño de componentes
- Diseño de arquitectura
- Diseño de clases/datos
- Diseño de interfaces

¿Qué es el diseño de software?

Diseño de componentes:

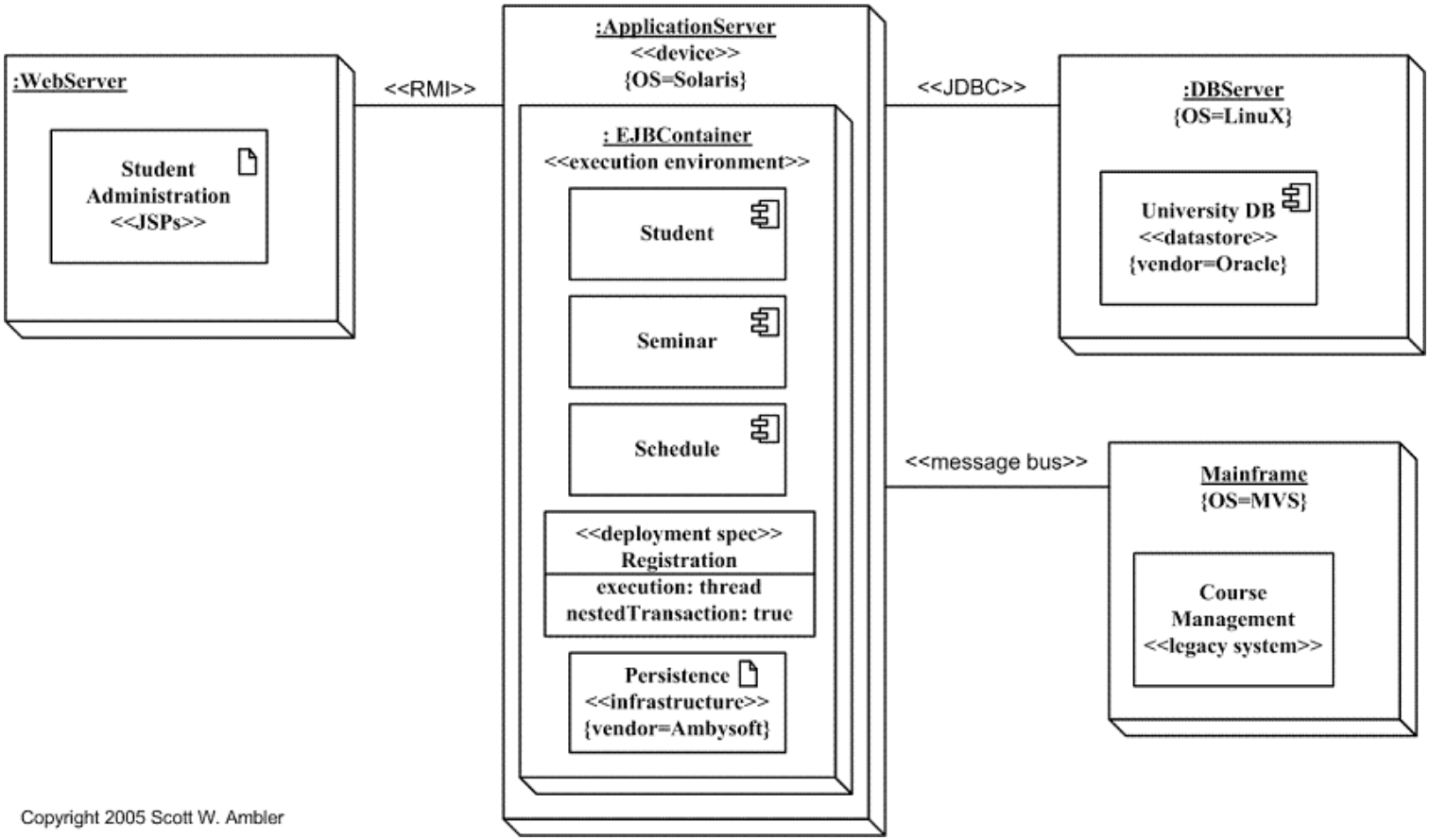
Descripción de los componentes del sistema



¿Qué es el diseño de software?

Diseño de arquitectura:

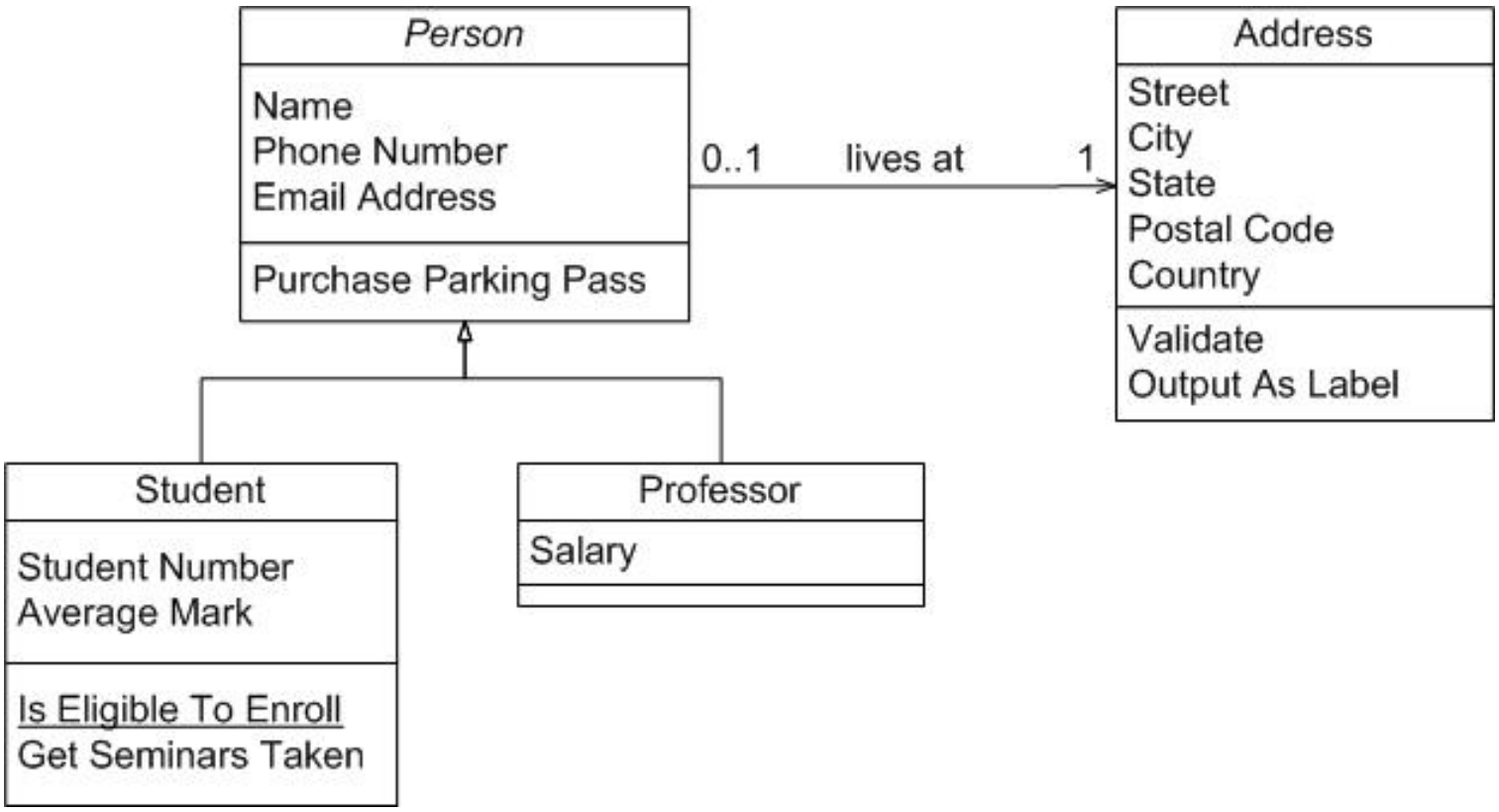
Relaciones entre los componentes del sistema



¿Qué es el diseño de software?

Diseño de clases/datos:

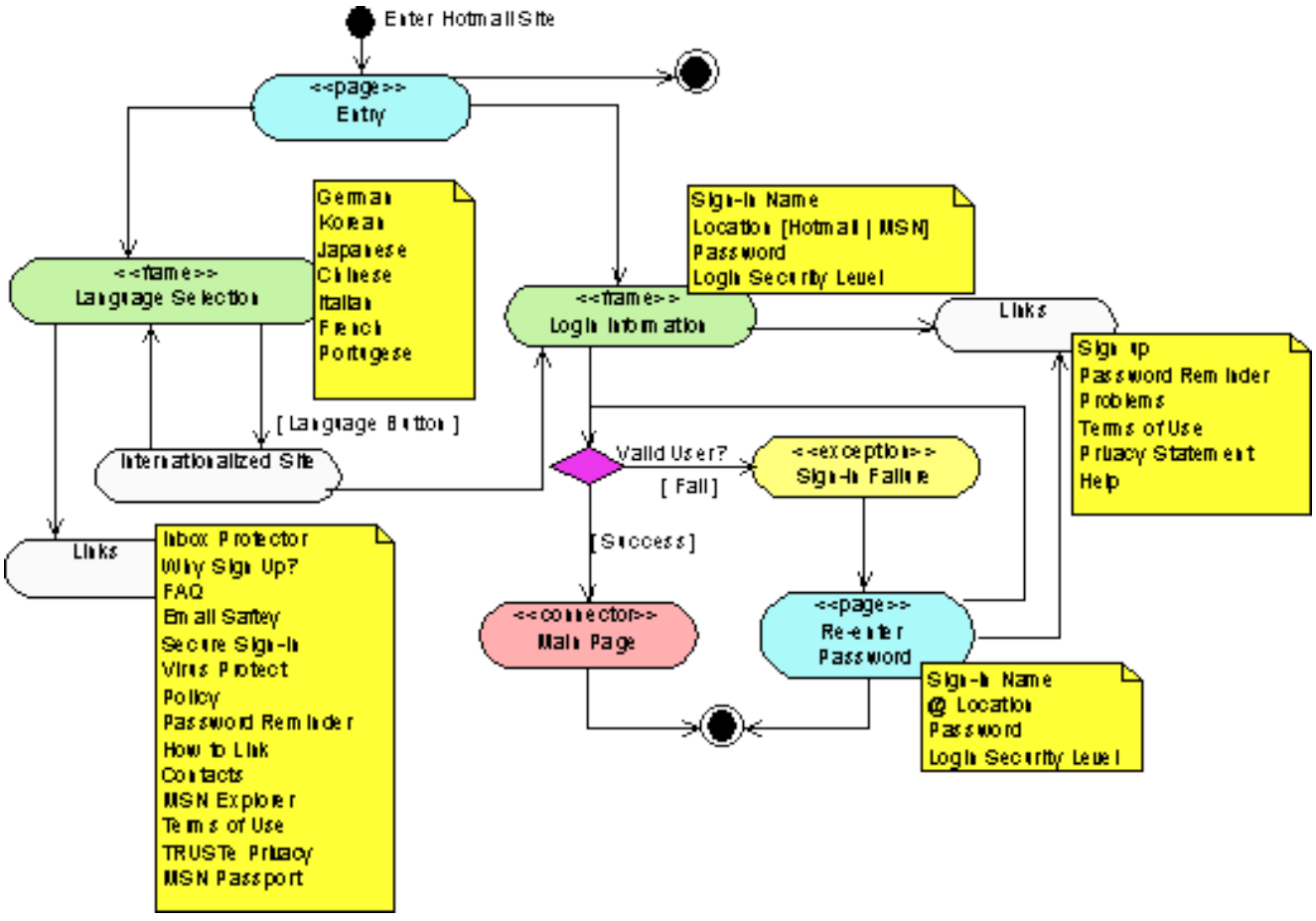
Esquema de clases y sus relaciones



¿Qué es el diseño de software?

Diseño de interfaces:

Comunicación con sistemas externos (y humanos)



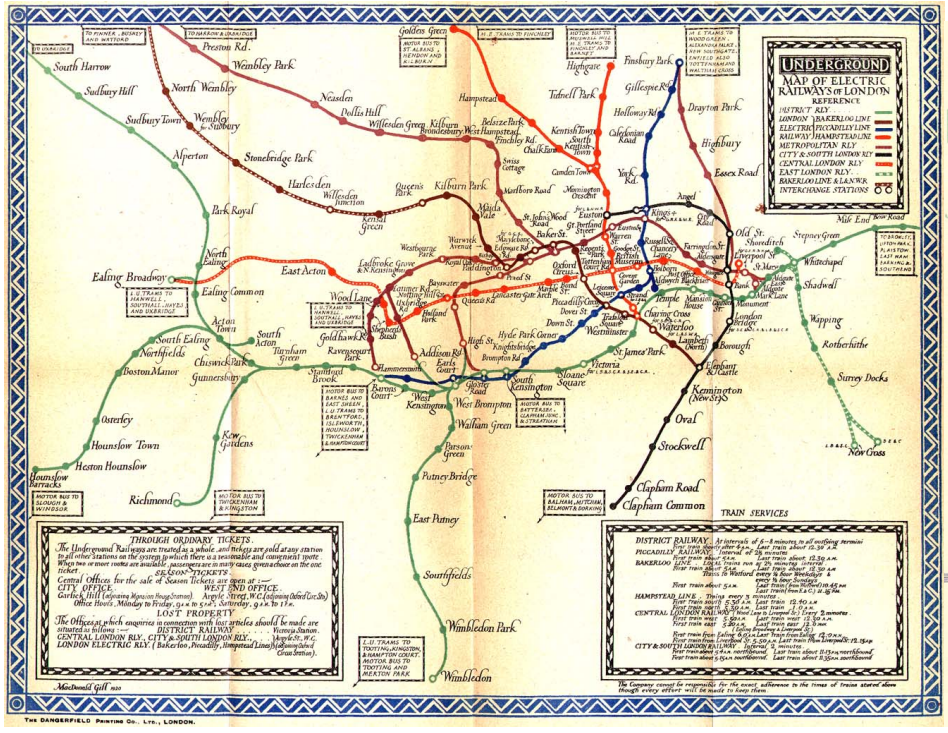
Principios del diseño detallado

- Abstracción
- Ocultamiento
- Cohesión
- Acoplamiento

Principios del diseño detallado

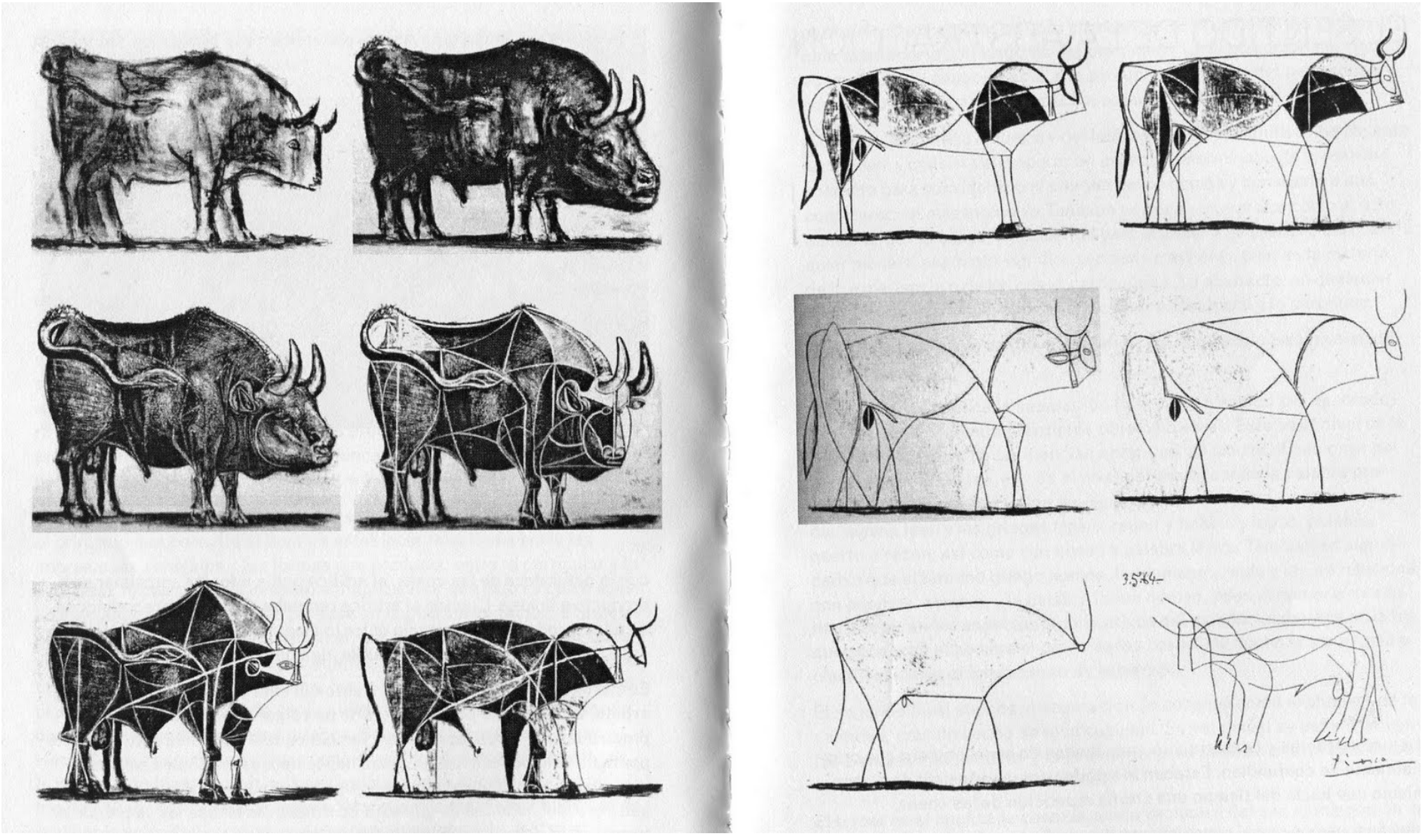
Abstracción:

Rescatar información relevante dado un contexto



Principios del diseño detallado

Abstracción:



Principios del diseño detallado

Ocultamiento:

No exponer información o lógica innecesaria

Ejemplos

- Servicios Web
- Librerías
- Modulos con modificadores de acceso

Principios del diseño detallado

Ocultamiento:

Beneficios

- Mantenibilidad
- Reusabilidad
- Extensibilidad

Principios del diseño detallado

Cohesión:

Medida de cuán relacionados están los datos, responsabilidades y métodos de una clase

Beneficios

- Reduce complejidad
- Aumenta mantenibilidad

Principios del diseño detallado

Acoplamiento:

Medida de cuán conectados están dos subsistemas o clases

```
class Warehouse
  def sale_price(item)
    (item.price - item.rebate) * @vat
  end
end
```

Bajo Acomplamiento, Alta Cohesión

Encuesta

Responder la siguiente encuesta antes del lunes (correo *UC*):

https://goo.gl/forms/sWLNr9mYIXv7dZeF3

Referencias

Jeff Kramer

Is Abstraction The Key To Computing?

http://www.ics.uci.edu/~andre/informatics223s2007/kramer.pdf

Agile Modeling

http://www.agilemodeling.com/essays/costOfChange.htm

Agile Modeling

http://agilemodeling.com/style/deploymentDiagram.htm

Agile Modeling

http://agilemodeling.com/artifacts/classDiagram.htm

Wikipedia

https://en.wikipedia.org/wiki/Component-based_software_engineering

Benjamin Lieberman

http://www.ibm.com/developerworks/rational/library/4697.html

Fin