Data access control and measure in the development of web-based health insurance systems

Zhen Jiang
Computer Science Department
Information Assurance Center
West Chester University
E-mail: zjiang@wcupa.edu

Outline

- Introduction
- Problem
- Our approaches
- Conclusion

Introduction

- Policies
- Why?
- Unified Modeling Language (UML) design
- Data attribute
- Data access
  - All the access of a certain data
  - All the access by using member functions of an object (effect area)
- Control of data access
  - Information leakage
  - Fully support of access requirement
  - Redundant access path

Problem

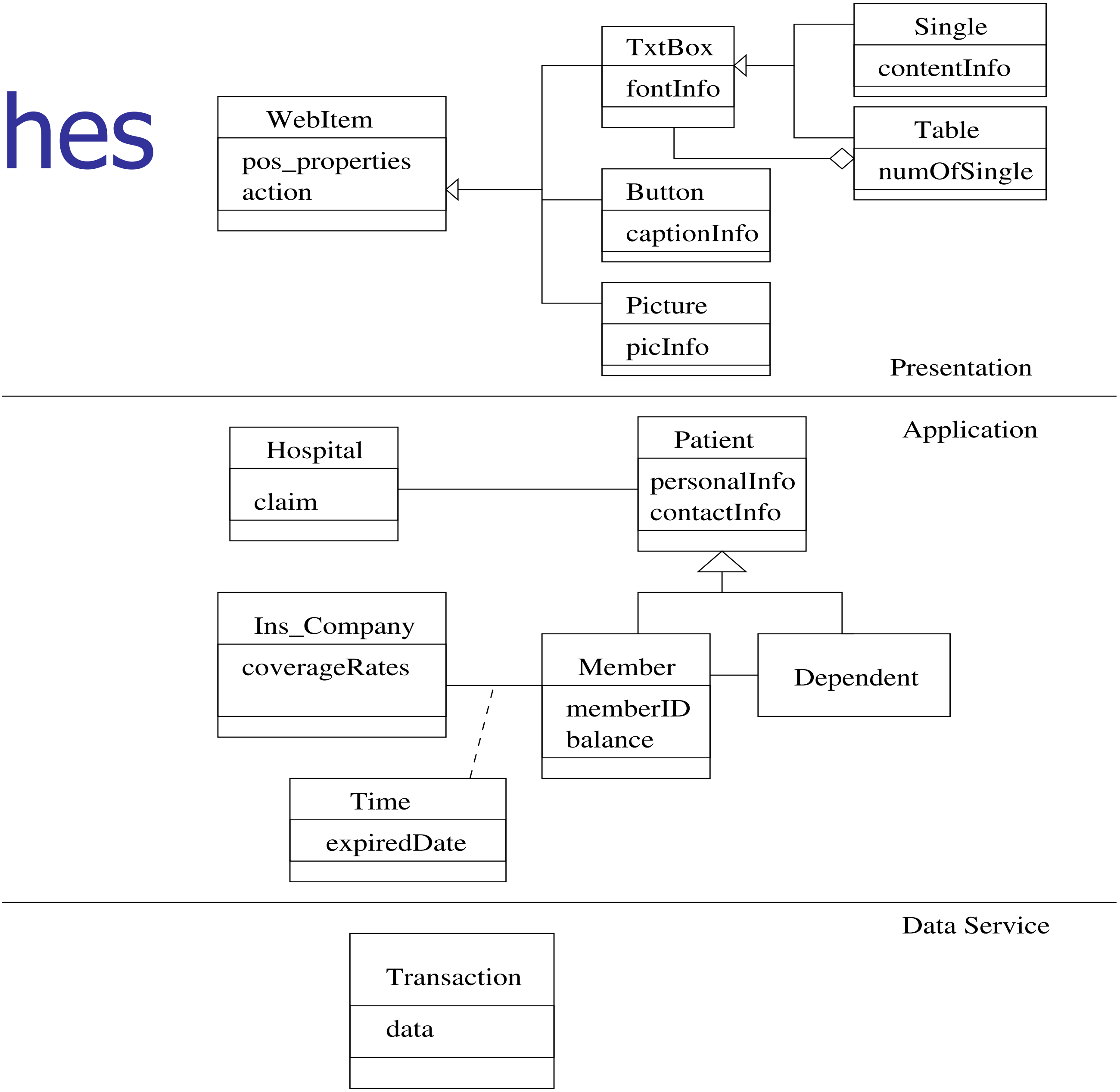
- Control on member functions and relationships between classes
- Control on basic relationships (generalization, aggregation, and association) and their usage in our insurance systems
- Control in design phrase

Our approaches

- Design framework of web-based health insurance systems
  - Quickly catch the complex relations in real world and map to UML diagram
  - Simple system, easy to analyze
- Measure the (possible) accesses of a certain class
  - To check if the requirement of all those accesses can be met (completeness check)
  - To check if the access is feasible (the permission is assigned)
  - To avoid adding redundant access path
  - To provide information for future access conflict analysis

Our approaches

- Design of our health insurance systems



WebItem
pos_properties
action

TxtBox
fontInfo

Single
contentInfo

Table
numOfSingle

Button
captionInfo

Picture
picInfo

Presentation

Application

Hospital
claim

Patient
personalInfo
contactInfo

Ins_Company
coverageRates

Member
memberID
balance

Dependent

Time
expiredDate

Data Service

Transaction
data

Our approaches

- Generalization, association, and aggregation
  - Association is basic relation.
  - Class A is a super class of class B if and only if any object of class B can also play the role as an object of class A.
  - Class A, as the whole class, has a whole-part relationship with class B if and only if any object of class B belonging to an object of class A has a member function involved in the action of a member function of the later one as a part of that.
  - Association class of two classes A and B will be consider a part of relationships between classes A and B.
  - All-to-all aggregation is used for multilayer implementation.

Our approaches

- Discovery:
  - It could be much easier for us to draw a UML class diagram if we start from the analysis of relationships of components.
  - The more natural the relationships between UML classes, the easier we read and understand the UML design, the easier and faster the development and maintenance of such a system.

Our approaches

- Data access by member functions
  - Data access defined in the same class
  - Data access from other class
- Rules for assignment of call permission of member functions propagating along all kinds of relations.
  - Rule 1: For any two classes A and B, if A is subclass of B, P(A, B)=true.
  - Rule 2: For any two classes A and B, if A is association related to B, P(A, B)=true.
  - Rule 3: For any two classes A and B, if A is whole class of B, P(A, B)= true.
  - Rule 4: For any association class A of two association related classes B and C, P(A, B)=P(A,C)=P(B,A)=P(C,A)=true.
  - Rule 5: In a multi-layer system, if class A is in the upper layer and class B is in the lower layer, P(A,B)=true.

Our approaches

- Permission assignment collected in set of access permission (SOAP)
  - Initially, SOAP(C)={C} for each class C.
  - Based on rules 1-5, find Re(C)={X| P(C,X) =true}.
  - Repeat SOAP(C) = SOAP(C)$\cup$ {Y|Y $\in$ SOAP(X) $\wedge$ X $\in$ Re(C)} in each round until there is no change of any SOAP.

Our approaches

classes	Re		SOAP		
		Initially	Roud 1	Round 2	Round 3
WebItem(W)	H, P, I, M, D, Ti	W	W, H, P, I, M, D, Ti	W, H, P, I, M, D, Ti, T	
TxtBox(Tx)	W, H, P, I, M, D, Ti	Tx	Tx, W, H, P, I, M, D, Ti	Tx, W, H, P, I, M, D, Ti, T	
Button(B)	W, H, P, I, M, D, Ti	B	B, W, H, P, I, M, D, Ti	B, W, H, P, I, M, D, Ti, T	
Picture (Pi)	W, H, P, I, M, D, Ti	Pi	Pi, W, H, P, I, M, D, Ti	Pi, W, H, P, I, M, D, Ti, T	
Single (S)	Tx, H, P, I, M, D, Ti	S	S, Tx, H, P, I, M, D, Ti	S, Tx, H, P, I, M, D, Ti, T	
Table (Ta)	Tx, H, P, I, M, D, Ti	Ta	Ta, Tx, H, P, I, M, D, Ti	Ta, Tx, H, P, I, M, D, Ti, W, T	
Hospital (H)	P, T	H	H, P, T		
Patient (P)	H, T	P	P, H, T		
Ins_Company (I)	M, Ti, T	I	I, M, Ti, T	I, M, Ti, T, P, D	I, M, Ti, T, P, D, H
Member (M)	P, I, D, Ti, T	M	M, P, I, D, Ti, T	M, P, I, D, Ti, T, H	
Dependent (D)	P, M, T	D	D, P, M, T	D, P, M, T, H, I, Ti	
Time (T)	I, M, T	Ti	Ti, I, M, T	Ti, I, M, T, P, D	Ti, I, M, T, P, D, H
Transaction (T)		T			

Our approaches

- For each class C in our system, we provide a new measure U(C)={ X|C ∈ SOPA(X)} to see all the places in which the member function(s) of class C could be used (directly or indirectly).

Our approaches

Classes	SOAP	U
WebItem (W)	W, H, P, I, M, D, Ti, T	W, Tx, B, Pi, S, Ta
TxtBox (Tx)	Tx, W, H, P, I, M, D, Ti, T	Tx, S, T
Button (B)	B, W, H, P, I, M, D, Ti, T	B
Picture (Pi)	Pi, W, H, P, I, M, D, Ti, T	Pi
Single (S)	S, Tx, H, P, I, M, D, Ti, W, T	S
Table (Ta)	Ta, Tx, H, P, I, M, D, Ti, W, T	Ta
Hospital (H)	H, P, T	W, Tx, B, Pi, S, Ta, H, P, I, M, D, Ti
Patient (P)	P, H, T	W, Tx, B, Pi, S, Ta, H, P, I, M, D, Ti
Ins_Company(I)	I, M, Ti, T, P, D, H	W, Tx, B, Pi, S, Ta, I, M, D, Ti
Member (M)	M, P, I, D, Ti, T, H	W, Tx, B, Pi, S, Ta, I, M, D, Ti
Dependent (D)	D, P, M, T, H, I, Ti	W, Tx, B, Pi, S, Ta, I, M, D, Ti
Time (Ti)	Ti, I, M, T, P, D, H	W, Tx, B, Pi, S, Ta, I, M, D, Ti
Transaction (T)	T	W, Tx, B, Pi, S, Ta, H, P, I, M, D, Ti, T

Our approaches

- No redundant access path
- All access will be ensured in design phrase
- Completeness check process to build assurance of data access of class C.
    - For any existing class X $\in$ U(C), check if class X needs to call any member function f of C although the permission has been assigned.
    - When such X is not found in SOAP(C), only check if C's member functions support the requirement of X. If not, redesign the member functions of C.
    - When such X is also found in SOAP(C), both X and C needs check at the same time and may need redesign if there is conflict or inconsistency among their member functions.

Conclusion

- Control and measure in design phrase
- A control solution from the side of computer science
- Impact analysis