

Domain Modeling Basics – Recap of key Points

- Reference: Chapter 10
- Software systems & their domains are different!
- Objects, etc. are *not* in the real world
 - But, the real world *motivates* classes, objects, etc.
 - *Analyzing* the world helps *design* the software later
- Thus, chapters 10-13 discuss the world
 - Philosophy, religion, etc. are other ways to understand the world!

Domain Modeling Recap II

Domain models show:

- Domain entities (“conceptual classes”) – chap. 10
- Relationships among conceptual classes – chap. 11
- “Attributes of conceptual classes” (p. 128) – chap. 12

Conceptual Classes

- A conceptual class is a “real-world...thing”
- Usually a noun
- Written with a rectangle in UML class diagrams
- See Figure 10.1
 - – notice the conceptual classes

Associations in Domain Models (Chapt. 11)

- Again, see Figure 10.1 for examples
- Note that the association need not read left-to-right (any direction is possible)
 - Convention: left-to-right or top-to-bottom
 - UML might be updated to have this convention
 - Note the optional little arrow for “captured-on”
- Many associations **can** exist
 - What extra associations could be added to 10.1?
 - Why not add them, would you guess?

Finding the Right Associations

- Associations are good if:
 - they help design software
 - (philosophy, theology, etc. have other goodness criteria)
- Try these methods to find associations:
 - Go through a list of **common association types**
 - Ask **which** of the potential associations need to have **information about them stored**
 - (Even if only for a tiny millisecond)

Common Association Types

- See Table 11.1
 - Examples: given the conceptual classes
 - Wing, airplane; flightLeg, flightRoute; passenger, flight; flightSchedule, flightDescription; maintenanceJob, maintenanceLog; reservation, flightManifest; pilot, airline; maintenance, airline; pilot, airplane; reservationAgent, passenger; passenger, ticket; reservation, cancellation; city, city
 - On the board, let’s add associations from the table now...
 - What are the numerical labels on the associations?

The Most Important of Those

- Physical part
- Logical part
- Physically contained in
- Logically contained in
- Recorded in (not in table but see p. 157)
- Let's highlight those now...

Getting Rid of Some of Them

- Too many associations confuse things
- Prefer those for which information needs to be stored for 1 ms or longer
- Avoid redundant or derivable ones
 - See (Figure 11.8+top of table p. 165)
- Let's check for these
 - (One needs to assume requirements)

Multiplicity

- See Figure 11.4
- See Figure 10.1
- What would the association look like for:
 - Week contains days
 - Class is held on odd days
 - (What is the problem here? Solution?)
 - Airplane transports passengers
 - Store stocks Jack's Famous Frozen Cheeze Pizzas 3-Packs

There can be Multiple Associations

- Sure, why not
- See Figure 11.7
- How about 2 associations for wing/plane?

Attributes

- **Conceptual classes** "are" rectangles
- They are connected by **associations**
 - (arcs)
- **Attributes** are the third component of domain models

More on Attributes

- They are values associated with an entity
- They are in the same rectangle, under the class name
- See e.g. Figure 10.1 or many others

What Should be an Attribute?

- The requirements should imply that its value needs to be known
- Otherwise, don't make it an attribute
- What are some attributes to add/keep/not add?
 - (See figures)
 - What is an attribute not needed for the POS system? Registration system? UML assistant system?

Attributes and Programming

- The value of an attribute should correspond to a fairly simple type
- Number, boolean, string, zip code...
- Note that they may or may not be
 - simple data types in Java/whatever
 - that's not a domain issue
- Let's find or update some for each case

Attribute Vs. Association?

- Simple data – try for an attribute
- Complex data – associate with a conceptual class
- Note that classes are more shareable
- Should the following be class+attribute, or two classes?
 - A flight has a flight number
 - A flight has a destination
 - A flight has an airplane
 - A flight has a number of passengers
 - A flight has a meal (or maybe it doesn't)

Judgement Calls

- Should
 - time, address, phone number...
 - be attributes or conceptual classes?

Judgement Calls II

- Answer: it depends
 - Yes – if complicated things need to be done to it
 - No – if it is essentially atomic for the purposes of this system
- When might time, address, & phone number be yes's? no's?
- How about zip code as another example

Just Being Simple is not Enough

- See Figure 12.5
 - Why is it this way?
- (This might make a good think-pair-share question)