Noun Identification Method

Software Requirements and Design
T-216-GHOH

Skúli Arnlaugsson | 14. október 2019

How can we find classes?

- There are many methods used to identify what classes should be in a system:
  - use known design patterns (there is a lot of them!)
  - use the "noun identification” method

Noun identification

- **Step 1**: Use whatever text we have prepared in requirement gathering phase, e.g.:
  - general description of the system
  - list of requirements
  - use cases
- **Step 2**: Identify the nouns in the text
  - Only the nouns (*ísl. nafnorð*)

Noun identification

- **Step 3**: Categorize the nouns into (usually) 3 categories:
  a) not relevant, can be thrown away
  b) relevant, are immediately recognizable as classes
  c) fuzzy, might be but we're not quite sure

Noun identification

- **Step 4**: Refine the list:
  - remove **redundant** words, i.e. if two words have same or similar meanings
  - remove **vague** words, where the meaning is not exactly clear
  - remove **meta-words**, words like "system" and "data" (these would be terrible classes!)
  - remove words which are **outside the scope** of the project
  - remove words which will obviously **become attributes** (or even operations), but won't become classes

Noun identification

- Noun identification is usually just the first step of a longer iterative phase
- The list of classes (or the class diagram) usually evolves over time
- Classes will be removed, others (that were even rejected during noun identification) will be introduced, some will be split up into more than one class etc.

An example

Let’s imagine that we have done the following so far:

- Analysis and UI design
  - User analysis
  - Identified stakeholders
  - Gathered information
  - Requirement analysis – *what* the system should do
  - UI prototyping from the above (might have involved more, e.g. user stories)
  - We have documented the above and have data for test cases and have even done some testing already (we see that later in the course)

The system boundaries have been set during the analysis phase (it can evolve though with iterations)

An example, continued

Now we continue and start to **design the programming part**, that is how to realize all of the before mentioned analysis in programming

Designing the programming part

- Decide the architecture of the system (not covered in this course)
- Make class diagrams
- Revise the state diagrams
- Make sequence diagrams of important parts of the system
- Make object diagram of selected parts if needed

So where do we start?

- Best to start to identify the classes
- Let’s do that together

General description – Útlánakerfi bóka

Á bókasafninu eru bækur, DVD og tímarit. Fleiri en eitt eintak af hverju getur verið til. DVD og sumar bækur eru einungis lánaðar skammtímaláni, þ.e. Í eina viku. Allar aðrar bækur geta bókasafnsmeðlimir fengið lánaðar í allt að 3 vikur. Bókasafnsmeðlimir geta verið með allt að 6 bækur lánaðar í einu, en starfsmenn 12 stykki. Hafa má 2 DVD diska og 2 tímarit í láni í einu.

Kerfið verður að halda utan um útlánin, þ.e. hvenær bækur og tímarit eru lánuð út og þeim skilað.

English on next slide

General description – Book lending system

The library has books, DVD’s and magazines. More than one unit of each may exist. DVD’s and some books are only let out short term, i.e. for one week. All other books can be borrowed by library members for up to 3 weeks. Library members can borrow up to 6 books at a time, but employees 12 books. Two DVD’s and 2 magazines can be borrowed at a time.

The system must control the lending of books, i.e. when books and magazines are let out and when they are returned.

Icelandic on previous slide

Name:	User borrows a book
Number:	21
Priority:	High
Precondition:	None
Description (base flow):	A user gives the clerk the book he wants to borrow, the clerk scans in the book barcode, or types in the ISBN of the book. The user then shows his library card, the clerk scans that in as well, and the user can then take the book to his home.
Alternative flow:	• This is a new user, and he doesn't have a library card. The clerk will have to register the user (see use case 22: Register new user). • The users library card has expired. Clerk offers user to renew his subscription (see use case 23: Renew user subscription). • The user forgot his library card. The clerk types in his SSN instead of scanning his card (see use case 28: Clerk looks up a user) • The user has borrowed a maximum amount of books, and must return some of them before he can borrow other books.
Postcondition:	This particular book is now in "borrowed" state, and cannot be borrowed by another user until it has been returned.
Source (requirements):	3, 11, 17
Actors	User, Clerk
Author	Jón Jónsson

Possible steps in our example

Identify the nouns and refine the list

- Find texts from analysis phase to work with

Make a class diagram

- Entity classes first from the nouns, then identify UI, Business and Repository layer classes

Look at e.g. use cases, find those that seem to involve a lot of logic and steps

- These might be the ones that all parties need to agree upon

Make a sequence diagram of those to clear up possible logic/design decisions and get confidence that the design is right, before entering the programming phase