

## Re: UML: extends .vs.generalize

**Source:** <http://coding.derkeiler.com/Archive/General/comp.object/2003-10/0070.html>

---

**From:** Wayne Dernoncourt ([wayned\\_at\\_panix.com](mailto:wayned_at_panix.com))

**Date:** 10/17/03

Date: Fri, 17 Oct 2003 11:14:36 +0000 (UTC)

Steve <[steve\\_a\\_haigh@hotmail.com](mailto:steve_a_haigh@hotmail.com)> wrote:

> *On Wed, 15 Oct 2003 11:40:34 +0000 (UTC), Wayne Dernoncourt*

> *<[wayned@panix.com](mailto:wayned@panix.com)> wrote:*

> <snip>

> > *The example used to illustrate the terms involved a*  
> > *college student application. The base class was*  
> > *"enroll student", the extension was "enroll foreign*  
> > *student" since the foreign student may have to have*  
> > *additional paperwork & steps. The "generalization"*  
> > *was "family member" (I believe a family member of*  
> > *an alumni or an active student, but I could be*  
> > *mis-interpreting and/or mis-remembering stuff).*

> > *I could see the student not needing all of the steps*  
> > *(maybe, maybe not) of a non-family member.*

> >

> > *Am I missing something here?*

> *Sort of... The term Extends applies to Use Cases and the term*  
> *Generalize applies to class heirarchies. Extending a Use Case is a*  
> *form of specialization, and is therefore the opposite of*  
> *generalization, but you and your friend seem to be talking about 2*  
> *different things. You are correct in saying they are the opposite of*  
> *each other, but from your examples I think you are perhaps applying*  
> *them in the wrong way.*

> *I think the example you give is for a Use Case not a class – "Enroll*  
> *student" sounds to me like a use case. Enrolling a foreign student makes*  
> *use of all the normal enrolling but also needs something extra (visa*  
> *application maybe), hence the Use Case for Enrol Foreign Student is an*  
> *extension of the basic Use Case.*

> *Family Member sounds like a class, and as such could be part of an*  
> *inheritance heirarchy. In a real system I can't see how modeling*  
> *Family Member as a generalisation of student is helpful, but if you*  
> *really wanted to you could say a student is-a family member, and in*  
> *turn a family member is-a person.*

(this is from "The Elements of UML Style" by Scott Ambler, similiar in style to "Elements of Style" by Strunk & White)

I managed to leave out a word, "Enroll" from the example from the book -- oops. It's supposed to be "Enroll Family Member". The book explains (pg. 22, #43) "...you can see that the \_Enroll\_International\_Student (italicized) use case extends the \_Enroll\_Student\_ use case..." I understand this, the international has more paperwork, etc.

But the next paragraph says "An <<extend>> association is a generalization relationship where the extending use case continues the behavior of the base use case by conceptually inserting additional action sequences into the base use case..."

I'm missing something here. If the use case first started out with "Enroll student" and did that stuff and then while doing the use case analysis on the foreign student noticed that some/most/all of the steps needed for the rest students also applied to the foreign students with some other paperwork tossed in, the "Enroll student" use case could be extended to "Enroll domestic student", "Enroll faculty student", "Enroll full time student", "Enroll international student", "Enroll part time student", etc. (similiar in concept to superclass and subclass with inheritance). The "Enroll student" use case may be used rarely (most students may be part time, etc.).

Could I be getting confused over the authors choice of words? I would thing "specialization" would fit better since actions are being added to a general item to make it more specific. Of course, I can full of crap!

> *A better example might be faculty member. Let's say a student must be  
> be a faculty member, so to is a teacher and an administrator. They may  
> all have some things in common, such as they all have a mail box in  
> the faculty, or they all have an ID card – you could extract these  
> attributes and put them in a Faculty Member class which is a  
> generalization of student, teacher and administrator. This is only a  
> toy example, in the real world I would model this differently, but  
> that would only complicate things here.*

--  
Take care | This clown speaks for himself, his job doesn't  
Wayne D. | pay for this, etc. (directly anyway)