SOFTWARE REQUIREMENTS SPECIFICATION

for

XModeler

Release 1.0

Version 1.0 approved

Contents

Revision History

1 Requirements

Req-1.0.0.0.1	Some requirements may be optional. Spec-2.0.2.1.1

1.1 Web Site

Req-1.1.0.0.1	The website is available online.
	A slightly different version is available in XModeler.
OR	The online version is accessed from XModeler, but unavailable when offline.
OR	XModeler tries to access the online version, but keeps a offline version as backup.
	The online version cannot use the XModeler's hijacking mode for references.
Req-1.1.0.0.2	The website will have several parts: The online home page, the in-tool home page, online content, in-tool content, online content which is available offline. In-tool content can access online content. Online content cannot access in-tool content.

1.1.1 Content

Req-1.1.1.0.1	• XModeler
	   – What is XModeler?
	   – Getting started
	      * Install guide
	      * Tutorial
	   – Help
	      * API
	      * Language concepts
	      * Snippets ^{X}
	      * Bluebook ^{X}
	
	• Literature
	   – Articles, Papers
	      * Our Articles, Papers
	      * Conferences
	      * List of others' literature
	   – Lecture Slides and other material for students ^{O}
	
	• External Links
	   – git
	   – shu
	   – ude
	
	• Contact
	   – Who are we
	   – How to contact us
	   – Impressum
	
	^{X}: Not available online
	^{O}: Only available online
	
	Is any material we use copyrighted? Are icons and logos in XModeler our work? If they are from any "'free"' source, are there any conditions on them?
	Do we need any legal advice? Disclaimers?

Req-1.1.1.0.2	The website should include a install guide
Req-1.1.1.0.2a	The install guide should be split into several parts, after each (and before first) some checks to see if step worked.
Req-1.1.1.0.2b	The install guide should contain common problems.
Req-1.1.1.0.3	The website should include a tutorial.
Req-1.1.1.0.4	The website should include a "'API"', explaining the basic data types and classes.
Req-1.1.1.0.5	The website should include an explanation of basic language concepts. (grammar, operators)
Req-1.1.1.0.6	The website should include snippets. (Small examples on a specific topic, like how to read from a file)
Req-1.1.1.0.7	The website should include a list of literature, with links if possible.
Req-1.1.1.0.8	The website should include a link to git, Maybe direct links to the Issues?
Req-1.1.1.0.9	The website should include information about contact. *Whom was it made by? How do I contact them, report a bug?. . .*

1.1.2 Design

Req-1.1.2.0.1	The design should have a primary and a secondary colour. These, together with grayscale colours should be used predominantly. Spec-2.1.1.0.1, Spec-2.1.1.0.2
Req-1.1.2.0.2	The website is split vertically: Title, Links, local Links, Content

1.2 MeMo-Language

Req-1.2.0.0.1	The MultiLevel language MeMo requires. . .
Req-1.2.0.0.2	The MultiLevel language MeMo requires. . .
Req-1.2.0.0.3	The MultiLevel language MeMo requires. . .

1.3 Kernel

1.4 X-Modeler

1.4.1 Diagrams

1.4.1.1 Multi-Level Diagram

1.4.2 Forms

1.4.2.1 FormsClient

Req-1.4.2.1.1	The FormsClient receives messages to add components.
Req-1.4.2.1.2	The FormsClient must be able to display these components: • Label • Textfield • Textarea • Checkbox The component is determined by the type of the value.
Req-1.4.2.1.2a	A boolean is shown as a Checkbox
Req-1.4.2.1.2b	An enum is shown as a drop-down-list
Req-1.4.2.1.2c	A short(?define) Strings or a number is shown as Textfield
Req-1.4.2.1.2d	A long(?define) String is shown as Textarea
Req-1.4.2.1.3	Labels can be grouped with other components to form a key-value-pair.
Req-1.4.2.1.4	A form client has a listener for changes which will be transferred instantly to the underlying model. There are no Save, Cancel or Undo buttons.
Req-1.4.2.1.5	The FormsClient is linked to the underlying model.
Req-1.4.2.1.6	The FormsClient's components are linked to the underlying model's parts by an id.
Comment	Labels should not be empty. Git Issue #34
Comment	Double click should not freeze the form. Git Issue #33

2 Specifications

Spec-2.0.2.1.1	Some specifications may be optional. Req-1.0.0.0.1

2.1 Web Site

2.1.1 Design

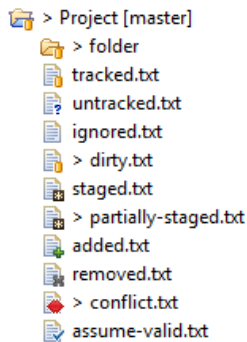| Spec-2.1.1.0.1	The design's primary colour is #1155BB. Req-1.1.2.0.1
Spec-2.1.1.0.2	The design's secondary colour is #22AA44. Req-1.1.2.0.1

2.2 XMF-Language

2.3 Kernel

2.4 X-Modeler

2.4.1 Diagrams

2.4.1.1 Multi-Level Diagram

3 Other

3.1 git

- **dirty (folder)** - At least one file below the folder is dirty; that means that it has changes in the working tree that are neither in the index nor in the repository.

- **tracked** - The resource is known to the Git repository and hence under version control.

- **untracked** - The resource is not known to the Git repository and will not be version controlled until it is explicitly added.

- **ignored** - The resource is ignored by the Git team provider. The preference settings under Team > Ignored Resources, "derived" flag and settings from .gitignore files are taken into account.

- **dirty** - The resource has changes in the working tree that are neither in the index nor in the repository.

- **staged** - The resource has changes which have been added to the index. Note that adding changes to the index is currently possible only in the commit dialog via the context menu of a resource.

- **partially-staged** - The resource has changes which are added to the index and additional changes in the working tree that neither reached the index nor have been committed to the repository. See partial staging from the Git Staging view for how to do that.

- **added** - The resource has not yet reached any commit in the repository but has been freshly added to the Git repository in order to be tracked in future.

- **removed** - The resource is staged for removal from the Git repository.

- **conflict** - A merge conflict exists for the file.

- **assume-valid** - The resource has the "assume unchanged" flag. This means that Git stops checking the working tree files for possible modifications, so you need to manually unset the bit to tell Git when you change the working tree file. Also see Assume unchanged action.

3.1.1 fetch

Fetch from remote repository

3.1.2 pull

Incorporates changes from a remote repository into the current branch. In its default mode, git pull is shorthand for git fetch followed by git merge FETCH_HEAD.

3.1.3 commit

Commit changes to local repository

3.1.4 push

Push Commits to remote repository

3.1.5 conflicts

http://wiki.eclipse.org/EGit/User_Guide#Resolving_a_merge_conflict

3.1.6 branches

branch merge rebase cherry-pick

3.1.7 revert

If a commit which has not yet been pushed has to be undone:

Select folder ⬤Team ➔ Show in History
Note number of commit
then use command line
`git revert <noted number>`

If a commit which has already been pushed has to be undone:
   Is done the same way.

Note: Both reverts don't undo history. The wrong commit and the reversion will be logged.

4 Example

Figure 4.1: Basic Class diagram

Dog extends ($\xrightarrow{\text{extends}}$) `Object`. It does not extend `Class`. It is an instance of ($\dashrightarrow$) `Class`. `Fido` is an instance of ($\dashrightarrow$) `Dog`. Therefore (commonly spoken): Fido is a Dog. Dog is a Class. But: Is `Fido` an `Object`? Is `Dog` an `Object`?
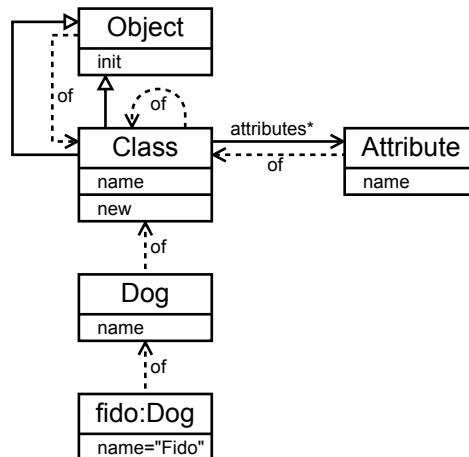
Differentiate between

- the *operation* of() which returns the target of the $\dashrightarrow$ of-Arrow and

- the *operator* instance? which returns true if second argument is $\dashrightarrow$ of() or anything of() $\dashrightarrow$ $\xrightarrow{\text{extends}}$ inherited from

- isTypeOf

- isReallyTypeOf

```
parserImport XOCL;
```

The class `OmniscientClass` extends `Class`. Any instance of it will therefore be a class.

```
context Root
  @Class OmniscientClass isabstract extends Class
  @Attribute instances : Vector = Vector(12) end
  @Attribute numberInstances : Integer = 0 end
  @Attribute pluralName : String (?) end
```
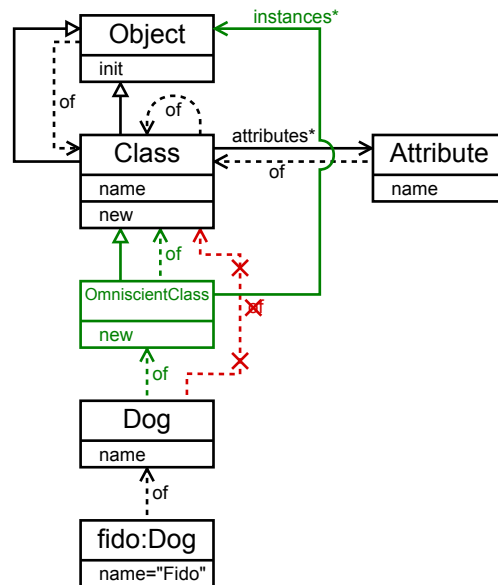
12

Figure 4.2: Extended Class diagram

```
@Operation getPluralName() : String
  name + "s"
end

@Operation new():Element
  if numberInstances >= 3
  then let
     o = super()
     in
```

This line is throwing an exception, when too many instances of the OmniscientClass are created. It uses the function `getPluralName()` to test about overwriting functions. In most cases the plural is formed by appending a `s`. This is done in the default function above. The word "Mouse" however has an irregular plural and therefore needs to overwrite the function, returning "Mice" instead of the default "Mouses".

Instead of overwriting functions, attributes could be used. In this case it proves a worse choice as every slot has to be filled manually, and no useful default can be provided.

```
        self.error("According to popular belief, "
        + (numberInstances+1) + " "
        + o.getPluralName()
    // + pluralName
        + " are deemed unlucky.")
  end
  else
    let
      o = super()
    in let
      c = o.of()
```

```
      in
        c.instances.put(numberInstances,o);
        self.numberInstances := numberInstances +1;
        o
      end end
    end
  end
end
```

Class `Dog` is an instance of `OmniscientClass`. It is a class because it's `of` is `Class` or a any extension thereof.

```
context Root
  @Class Dog metaclass OmniscientClass
  @Attribute name : String = "nameless Dog" end
  @Constructor(name) ! end
  @Slot OmniscientClass::pluralName = "Dogs" end
```

As `Dog` does not extend `Class`, this function is not supposed to be called. However, it can be called by `(Dog())()`, naturally causing an exception to be thrown somewhere in `super()`. It can be called without an exception if `Dog` would extend `Class`.

```
  @Operation new():Element
    "Dog.new".println();
    super()
  end
end

context Root
  @Class Cat metaclass OmniscientClass
  @Attribute name : String = "nameless Cat" end
  @Constructor(name) ! end
  @Slot OmniscientClass::pluralName = "Cats" end
end
```

The class `Mouse`

```
context Root
  @Class Mouse metaclass OmniscientClass
  @Attribute name : String = "nameless Mouse" end
  @Constructor(name) ! end
  @Slot OmniscientClass::pluralName = "Mice" end
  @Operation getPluralName() : String
    "Mice"
  end
end

  Root::dog1 := Dog.new();
  Root::dog1.name := "Fido";
  Root::mouse1 := Mouse("Mickey");
```

Objects have a type and slots. The type is returned by the of() operation and the slots are accessed by name either using the '.' operator or the get(name) operation. Values of slots can be updated using the o.s := v notation or the o.set(s,v) operator. Messages can be sent to an object using the o.m(a1,a2,...) notation or using the o.send(m,Seqa1,a2,...) operator.

Objects implement a MOP via their meta-class that determines how slot access and update is handled. The MOP is defined on an objects type via the getInstanceSlot, hasInstanceSlot and setInstanceSlot operations. Define a new meta-class (sub-class of Class) that implements a new MOP and therefore handles object storage completely differently to the VM.

Slot storage is usually allocated when a class is instantiated and **is always consistent with the attributes defined by the class**. You can modify this using addStructuralFeature and removeStructuralFeature, but you **must be aware that certain key execution features of XMF rely on the slots of an object conforming to the attributes of its class**. NB adding new slots to an object that do not clash with these attributes is safe and can be used to squirrel away information in particular objects.

4.0.8 Multilevel Draft Essen

Extend Association, Attribute, Compiled Operation by isIntrinsic, instLevel, isCore and Constraint and Doc by id.

new class/interface MetaAdapter overriding new(decreasing level by one) with additional function newAtLevel(decreasing level by more than one)

???

Is a level $<= 1$ (class, object) automatically forbidden to extend MetaAdapter/Class? Is a level 0 (object) automatically forbidden to be of type MetaAdapter/Class? Is an element which does not extend MetaAdapter/Class but is of type MetaAdapter/Class automatically a level 1? Is an element which is not of type MetaAdapter/Class automatically a level 0?

Why start numbering levels at object level? This means: a classic object is always level 0, a classic class always level 1. Therefore it is always necessary to know before the number of the top layer. Or do the upper layers number automatically? But then the instLevel for attributes does not give an absolute level-distance. Maybe allow relative intLevels? Or use another concept than integer numbering.

!!!

4.0.9 next Try

An element e is level 0, if and only if $e.of()$ is level 1.
$Level_0(e) \Leftrightarrow Level_1(e.of())$
$Level_1(e) \Rightarrow \neg e$ inheritsFrom $Class$.
$Level_1(e) \Leftrightarrow Level_2(e.of()) \wedge \neg e$ inheritsFrom $Class$.
$Level_n(Class) \forall n \geq 2$

4.0.10 next Try II

4.0.10.1 numbering up

Elements can be numbered by their distance to an object, if that object is given.
Object o's Level in respect to o is 0. Object o's of()'s Level in respect to o is 1. Class' Level in respect to o is the number of *of()*s one has to traverse,. including but not counting *inherits*.

```
int levelUp(Element a, Element b) {
    if(a == b) return 0;
    if(b.of() != b) return 1 + levelUp(a, b.of());
    if(b.inherits() != b) return levelUp(a, b.inherits());
    throw Exception;
}
```

4.0.10.2 numbering down

Elements can be numbered by their distance to *Class*
Object o's Level in respect to *Class* is the number of *of()*s one has to traverse from o to Class, including but not counting *inherits*. Be this number negative for easier disambiguation.

```
int levelDown(Element e) {
    if(e == Class) return 0;
    if(e.of() != e) return -1 - levelDown(e.of());
    if(e.inherits() != e) return levelDown(e.inherits());
    throw Exception;
} <==> {
    return - levelUp(Class, e);
}
```

4.0.10.3 allowing multi-level jumps

If we have a of-relation with a specified weight $>= 1$ we can amend the formulas above nad replace 1 with the weight.

```
int levelUp(Element a, Element b) {
    if(a == b) return 0;
    if(b.of() != b) return b.ofWeight() + levelUp(a, b.of());
    if(b.inherits() != b) return levelUp(a, b.inherits());
    throw Exception;
}
```

4.0.10.4 Finding a unified numbering scheme

Assume the following relation:

fido ⋅⋅of⋅⋅▸ Dog ⋅⋅of⋅⋅▸ Quadruped ⋅⋅of⋅⋅▸ Animal ⋅⋅of⋅⋅▸ MetaClass ⋅⋅of⋅⋅▸ Class

squidward ⋅⋅of⋅⋅▸ Octopus ⋅⋅of⋅⋅▸ Animal ⋅⋅of⋅⋅▸ MetaClass ⋅⋅of⋅⋅▸ Class

where (Quadruped and Animal) $\xrightarrow{\text{extends}}$ MetaClass $\xrightarrow{\text{extends}}$ Class

x	levelDown(x)	levelUp(x, fido)	levelUp(x, Octopus)	levelUp(x, Animal)
fido	-5	0	N/A	(-3)
Dog	-4	1	N/A	(-2)
Quadruped	-3	2	N/A	(-1)
squidward	-4	N/A	(-1)	(-2)
Octopus	-3	N/A	0	(-1)
Animal	-2	3	1	0
MetaClass	-1	4	2	1
Class	0	5	3	2

The levelUp function can be extended downwards into negative results.

If we want a Slot name for every Animal, we would add an Attribute name to Animal to become a Slot on Level 0. But we cannot say which level Animal is on.

We cannot instantiate an animal object directly to level 0, as Animal inherits Class

5 Literature

- A Unifying Approach to Connections for Multi-Level Modeling [2]
  - Melanie: Multi-level Modeling and Ontology Engineering Environment [3]
  - Referenced 2
  - Referenced 3

5.1 Integrating Language Engineering [1]...

6 Weekly Reports

6.15 2015

6.15.37 7 Sep 2015 - 11 Sep 2015

- Made browser tabs closeable

- Init Requirements and Specifications in LaTeX

- Added Panic button to GUI

- Image loading: XML file now derived from IMG file name and path instead of storing that information in the IMG file.

- Division of BigIntegers fixed, Multiplication of negative Integers fixed

6.15.38 14 Sep 2015 - 18 Sep 2015

- Tested and documented some git features.

- Produced conflicts in git: Managed to resolve them finally. It was challenging. Further testing required.

- Tried to install safari. Waiting for Help desk...

- Made the browser work again. Several fixes: Removed layouter, changed locking mechanism, set to native browser

6.15.39 21 Sep 2015 - 25 Sep 2015

Used XMF to do exercises with meta-classes

- to understand the concept of class, object, instantiation and inheritance

- to override the default meta-object-protocol, be redefining new/get/set etc.

- to understand the default behaviour of XMF

6.15.40 28 Sep 2015 - 2 Oct 2015

- Continued experiments with meta-classes

- Gathered ideas for website; some experiments with html.

6.15.41 ToDo

- improve webpage

- Specify FormClient

- git commands in Eclipse

- (nextWeek) MeMo language requirements

- pw protected folder: `https://wincent.com/wiki/git_repository_access_control`

- Snippets

- look at/play with Kernel (Object/Class/Attribute)

- Use Notepad++ with syntax highlighting for user defined languages: `http://weblogs.asp.net/jongalloway/creating-a-user-defined-language-in-notepad`

- HTMLViewer.xmf: Are ".o", ".xip", ".xto", ".xtd", ".xtml" still in use? Welcome.xmf: requestURL ... requirements

Bibliography

[1] T. Clark and U. Frank. Integrating language engineering, conceptual modelling and programming: A multilevel development and runtime environment. 0000.

[2] C. Atkinson, R. Gerbig and T. Kühne. A unifying approach to connections for multi-level modeling. *Proceedings of 18th ACM/IEEE International Conference MODELS*, 2015.

[3] C. Atkinson and R. Gerbig. Melanie - multi-level modeling and ontology engineering environment. 2012.

[4] H. J. Farnsworth and L. Simpson. Mathematics in the year 3000. *Proceedings of 318th IEEE International Conference*, 55(4), 3015, pp. 309–319.