Achieving a unified data model with Cargo and Page Forms

Yaron Koren

EMWCon Spring 2018

March 21, 2018

Houston, TX

A brief history…

2006: Semantic MediaWiki introduced

Data structure:

- Templates
- Properties
- Categories
- Query pages

2007: Page Forms (then called Semantic Forms) introduced

Data structure:

- Templates
- **Forms**
- Properties
- Categories
- Query pages

2007: Semantic Drilldown introduced

Data structure:

- Templates
- Forms
- Properties
- **Filters**
- Categories
- Query pages

2009: Special:RunQuery added to Semantic/Page Forms

Data structure:

- Templates
- Forms
- Properties
- Filters
- Categories
- Query pages
- **Query forms**

2014: Filters in Semantic Drilldown now defined within categories

Data structure:

- Templates
- Forms
- Properties
- Categories
- Query pages
- Query forms

2015: Cargo introduced

Data structure:

- Templates
- Forms
- *Properties* (SMW only)
- *Categories* (more important in SMW than in Cargo)
- Query pages
- Query forms

With either SMW or Cargo, 4-6 page types are still needed.

This is complex, and redundant.

Can we eliminate some (or all) of this redundancy?

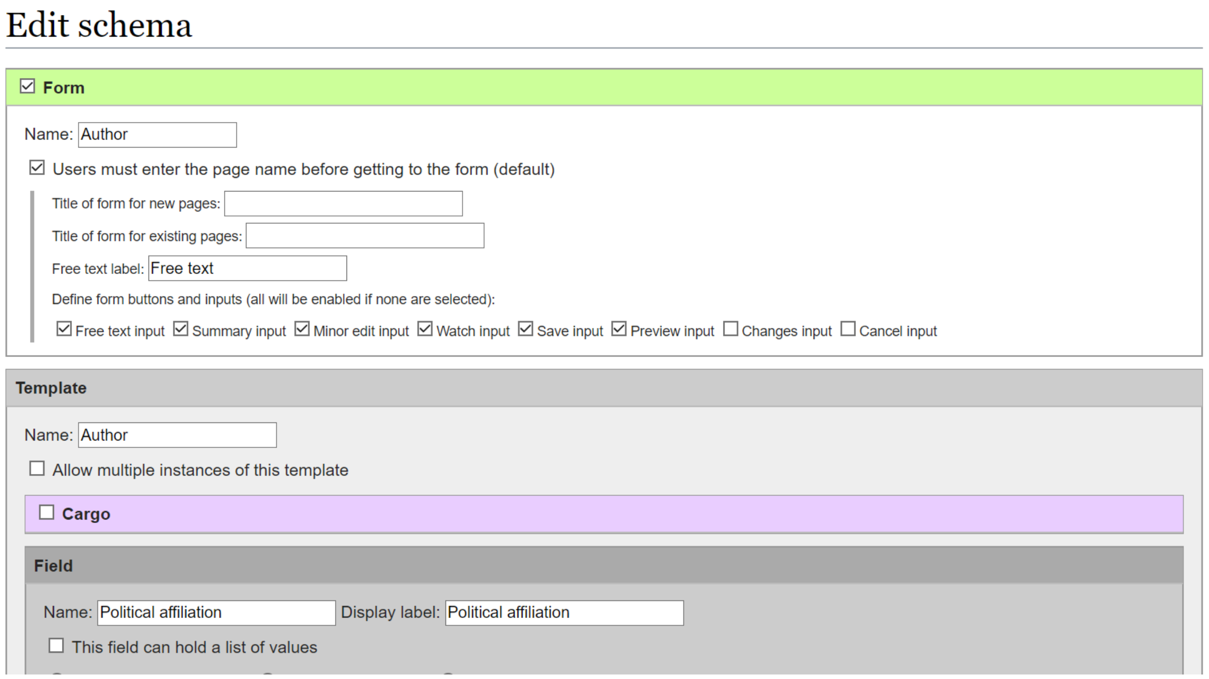
A solution many have implemented:
create a "data model".

Define the structure in one place, then generate (and re-generate) templates, forms etc. from it.

Data model approach 1

Define the structure on the wiki, generate pages from it.

- Example: Page Schemas extension



Edit schema

☑ Form

Name: Author

☑ Users must enter the page name before getting to the form (default)

Title of form for new pages:

Title of form for existing pages:

Free text label: Free text

Define form buttons and inputs (all will be enabled if none are selected):

☑ Free text input ☑ Summary input ☑ Minor edit input ☑ Watch input ☑ Save input ☑ Preview input ☐ Changes input ☐ Cancel input

Template

Name: Author

☐ Allow multiple instances of this template

☐ Cargo

Field

Name: Political affiliation Display label: Political affiliation

☐ This field can hold a list of values

Data model approach 2

Define the structure outside of the wiki, generate pages from it.

- Example: Mobo
- The most popular approach, but all the solutions so far (including Mobo) remain local & unreleased.

Data model approach 3

Define the structure in a DB, have pages query the structure.

- Example: MITRE's setup
- Templates and forms are just stubs that use Scribunto/Lua to display the current structure.
- Very clever! But few are using it.

One weakness of all these approaches: **they add more complexity**.

More software, more structures, harder to figure out how to change anything.

Using Cargo, it may be possible to truly simplify creation of the data structure, down to just one thing:

Templates.

In Cargo, all of a “class” is defined in a call to #cargo_declare, within a template.

Can we base everything around that #cargo_declare call?

Sample #cargo_declare call

In the page “Template:Person”:

```
{{#cargo_declare:_table=People
|Date_of_birth=Date
|Nationality=Page
|Hobbies=List (,) of String
}}
```

What about SMW?

This could be done in SMW too, in theory...

SMW has the function **#declare** (which no one uses) - basically equivalent to #cargo_declare.

It, too, could be extended to support these settings.

What about SMW? (2)

Will it happen? ¯_(ツ)_/¯

Automating filters

Already done!

Filtering in Special:Drilldown is automatic, based on #cargo_declare.

Automating forms

Can forms be generated automatically, based only on the defined data structure?

For simple, one-template forms, this could already be done.

Logic for automatic forms w/ Cargo

If no form is specified for some page

...and that page calls a template that declares a Cargo table

...then create an on-the-fly form for that table, using the template/table fields.

Form elements already deriveable from the template

- input type
- allowed values (including hierarchy)
- mandatory*
- unique*
- regex*

** New in latest Cargo, PF versions*

In other words...

The following within a #cargo_declare call:

```
|Color=String (allowed
values=Red,Blue,Yellow; mandatory)
```

...will lead to a mandatory dropdown input in the form.

Aspects that cannot yet be set

- more than one template per form
- multiple-instance templates
- "values from ..."
- "show on select"
- "values dependent on"
- ...and more

One example: multiple-instance templates

Specific example: a form called "Person" has one instance of template "Person", many instances of template "Job".

Can we define this relationship within #cargo_declare in the template?

Possible solution

Define “child tables”.

Within #cargo_declare:

```
_childTables=Jobs (embed in field=Jobs)
```

This info can potentially be used not just in forms, but also in filters and queries!

“Child tables” in drill-down

When drilling down on “People” table:

Click on one of the following values to filter the results:

Date of birth: ….

Nationality: …

Jobs:Organization: …

Jobs:Start date: …

“Child table” in queries:

Standard query:

```
{{#cargo_query:table=People,Jobs
|join on=People._pageID=Jobs._pageID
|fields=People._pageName,
People.Date_of_birth, Jobs.Organization
}}
```

Red text can be removed if “child table” relationship has been established.

This is a "holistic" approach that sees **storage**, **data entry**, **querying** and **drill-down** as all related.

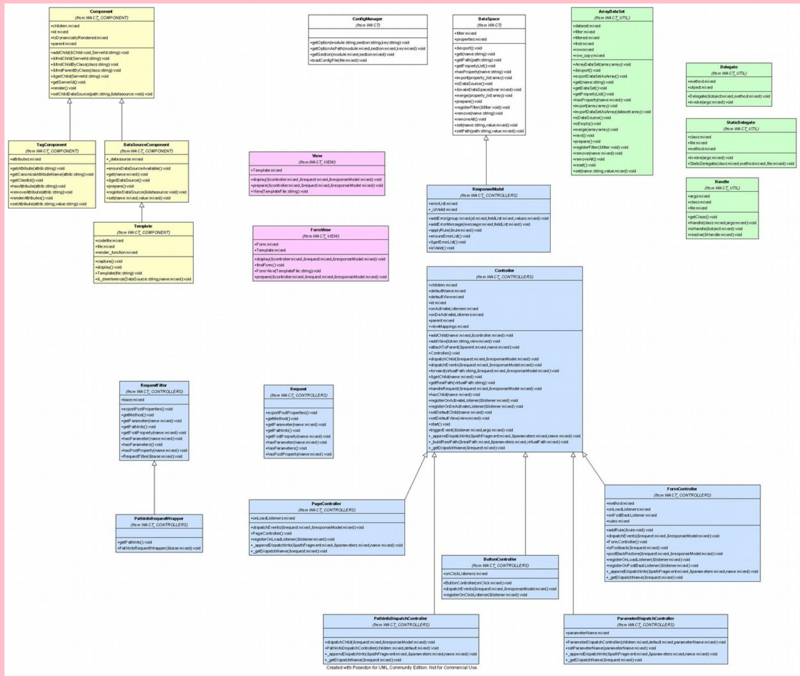
Some thoughts on UML

The old "holy grail"

Define all your classes in a UML file, from which are generated templates, forms, etc.

UML

- “Unified Modeling Language”
- Enables graphical display of data structure



This can’t work…

…because **UML is not good enough**.

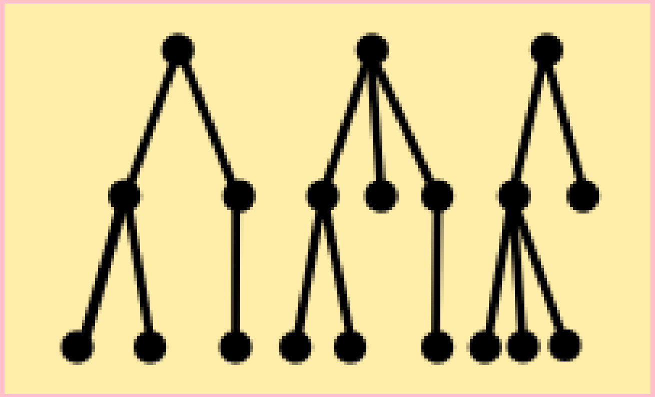
It’s basically just a simple wrapper around SQL.

Whatever can’t be done with a single DB table, can’t be structured in a single UML class. (I think.)

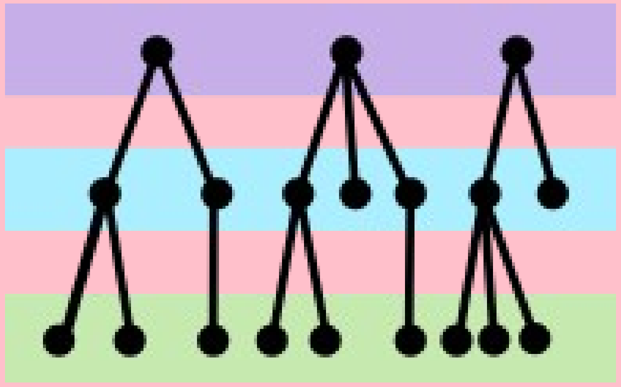
One example: hierarchies

Two main kinds of hierarchies:

Single-field: “Genre” can have a value of "Drama" > "Historical drama" > "Biblical drama" etc.



Multi-field: "Continent", "Country" & "City" fields represent different levels in the hierarchy



UML can’t represent hierarchies!

Yes, you can define all the tables needed to *represent* a hierarchy field.

But you can’t just say, “this field can hold any of the following hierarchy of values”.

Not good enough!

Should UML represent hierarchies?

I think so.

You just need to allow a UML class to represent more than one DB table.

Hierarchies in UML: an example

In UML:

“In the Movies class, the Genre field is a hierarchy with the following tree of allowed values.”

In the DB:

- “Movies” table gets a “**GenreID**” column
- **Additional table, “Genre”**, with fields “ID”, “Name” & “ParentID”

Hierarchy support in Page Forms / Cargo / SMW

- Single-field hierarchies - supported in Page Forms and Cargo (and SMW, sort of, via categories)
- Multi-field hierarchies - sort of supported in Page Forms (with "show on select" and "values dependent on"), not supported yet in Cargo or SMW

Simplifying templates

Finally, template creation may be simplifyable too.

Maybe #cargo_declare is (more or less) all that's needed within a *template* as well?

What’s next

A planned Google Summer of Code project for this summer, “Improve the Cargo’s Special:Drilldown page”, may add some of this functionality.

Exit question

Has anyone tried to make a comprehensive data modelling language like this before??

If so, I don’t know about it.

Questions/comments/concerns