University of Tartu

Rein Raudjärv

Dynamic Schema-Based Web Forms Generation in Java

Master Thesis (30 EAP)

Supervisor: Marlon Dumas, *PhD*

Tartu, June 7th 2010

Motivation

1. Generating web forms from data schemas with presentation annotations is a useful method for rapid application development.

2. Many Web form generators exist (e.g. IBM XML Forms Generator).

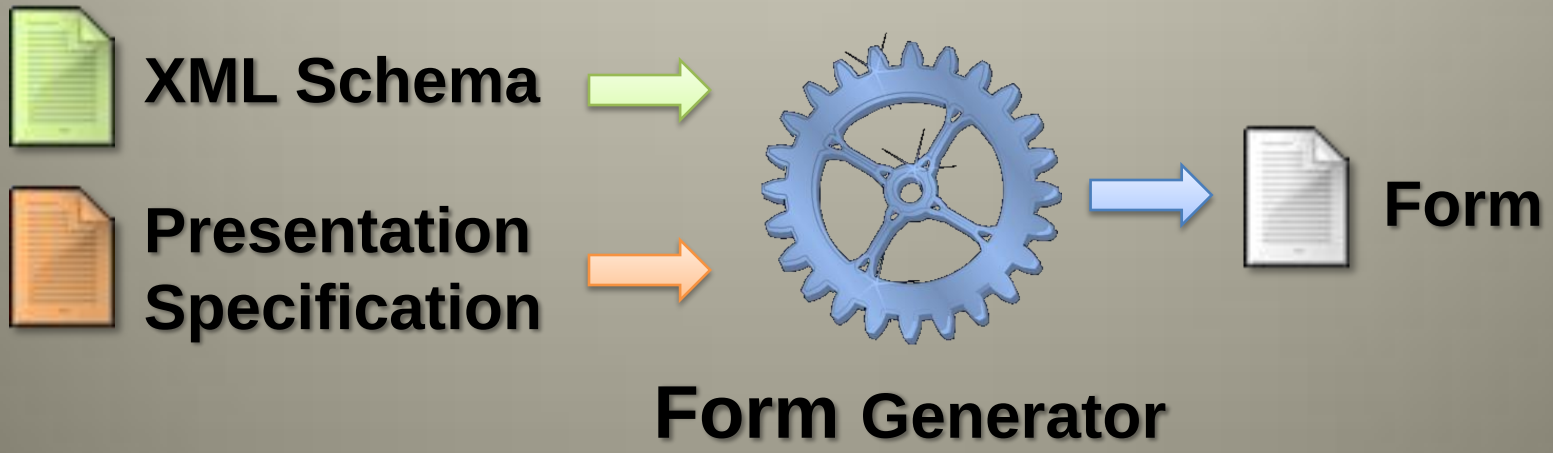
Motivation (2)

3. In existing form generators, the data schema and the presentation specification are tightly connected.

4. Consequence: each schema change requires developer to adjust the form and re-deploy.

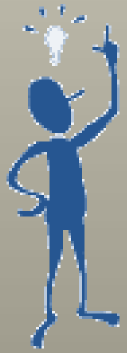
Aim of the Thesis

... was to design and implement
a **web form generator** producing forms out of **XML schemas** and **presentation specifications** which are **allowed to evolve independently**.



Key Ideas

1. Decouple the data schema from the presentation specification.
2. When the schema changes, the form immediately becomes available under the new schema, with a possible degradation of the quality of the presentation.



Form Generation

XML Schema

```
<xs:element
  name="author“ type="xs:string"/>
```

Form Generation

XML Schema

```
<xs:element
  name="author“ type="xs:string"/>
```

Form Generator

1. First Form

Form Generation

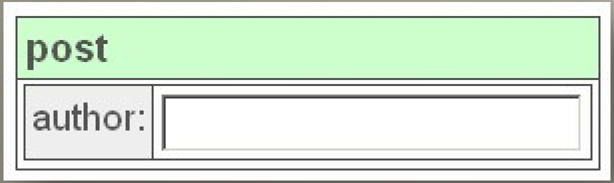
XML Schema

<xs:element
  name="author“ type="xs:string"/>

Presentation specification

/post/author { label: author; control: input; }

Form


post

author:

Form Generator

1. First Form

Form Generation (2)

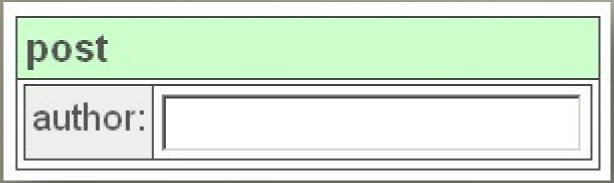
XML Schema

```
<xs:element
  name="author“ type="xs:string"/>
```

Presentation specification

/post/author { label: author; control: **text**; **size: 20;** }

Form


post
author:

Form Generator

2. Customizing Form

Form Generation (2)

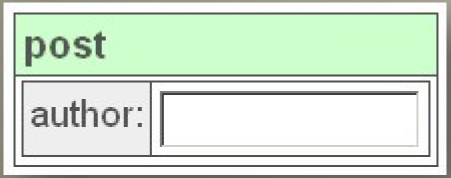
XML Schema

```
<xs:element
  name="author“ type="xs:string"/>
```

Presentation specification

```
/post/author  { label: author;
control: text; size: 20; }
```

Form



Form Generator

2. Customizing Form

Form Generation (3)

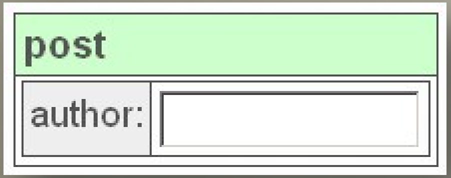
XML Schema

```
<xs:element
  name="author“ type="xs:string"/>
<xs:element
  name="message“ type="xs:string"/>
```

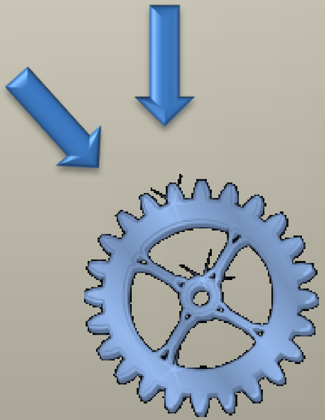
Presentation specification

```
/post/author  { label: author;
control: text; size: 20; }
```

Form



post
author:



Form Generator

3. Updating Schema

Form Generation (3)

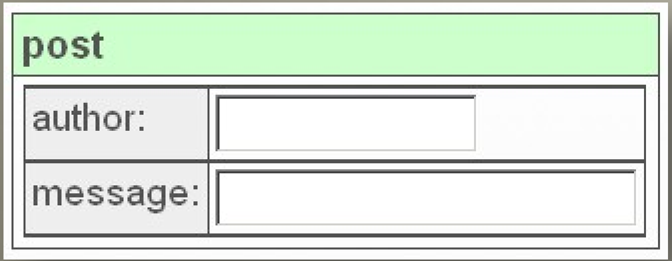
XML Schema

```
<xs:element
  name="author“ type="xs:string"/>
<xs:element
  name="message“ type="xs:string"/>
```

Presentation specification

```
/post/author  { label: author;
control: text; size: 20; }
/post/message  { label: message;
control: input; }
```

Form



post	
author:	
message:	

Form Generator

3. Updating Schema

Form Generation (4)

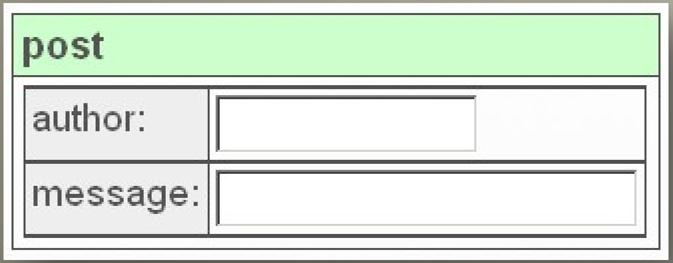
XML Schema

```
<xs:element
  name="author" type="xs:string"/>
<xs:element
  name="message" type="xs:string"/>
```

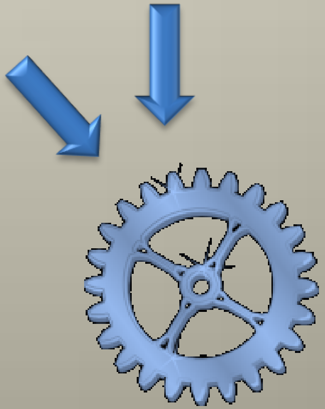
Presentation specification

```
/post/author  { label: author;
control: text; size: 20; }
/post/message  { label: message;
control: textarea; }
```

Form



post	
author:	
message:	



Form Generator

4. Customizing Form

Form Generation (4)

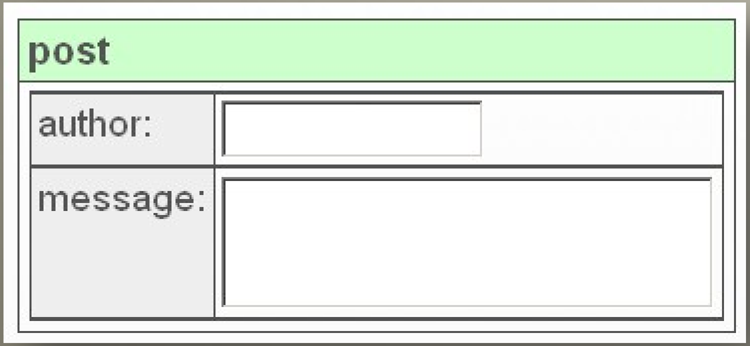
XML Schema

```
<xs:element
  name="author“ type="xs:string"/>
<xs:element
  name="message“ type="xs:string"/>
```

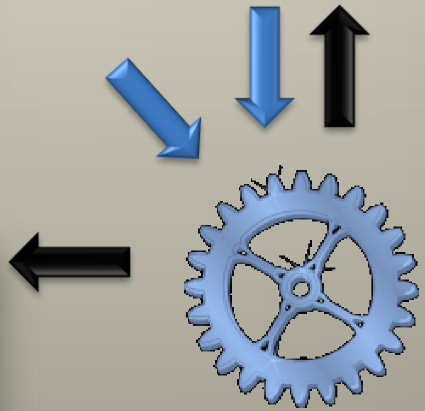
Presentation specification

```
/post/author  { label: author;
control: text; size: 20; }
/post/message  { label: message;
control: textarea; }
```

Form



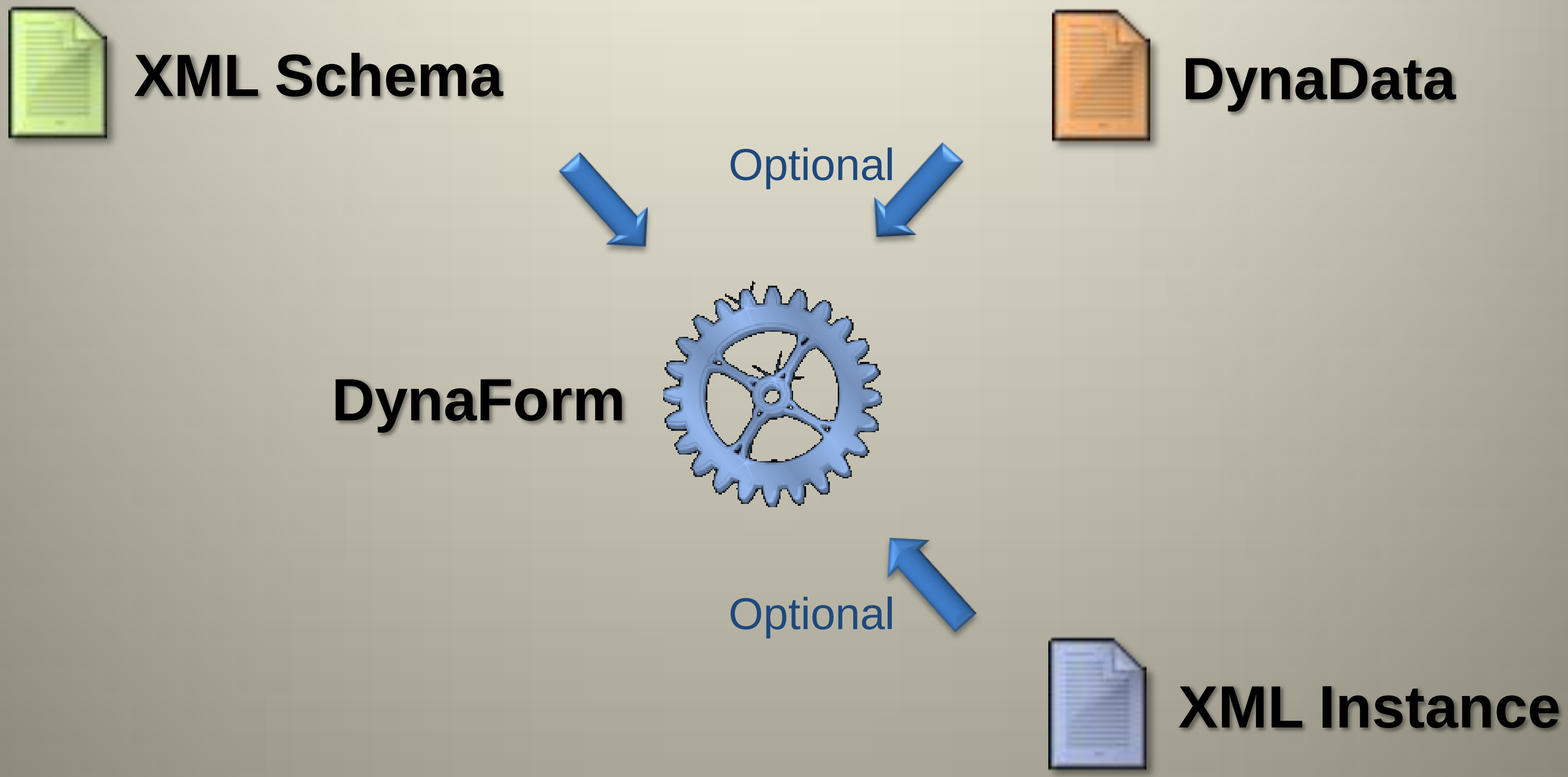
post	
author:	
message:	



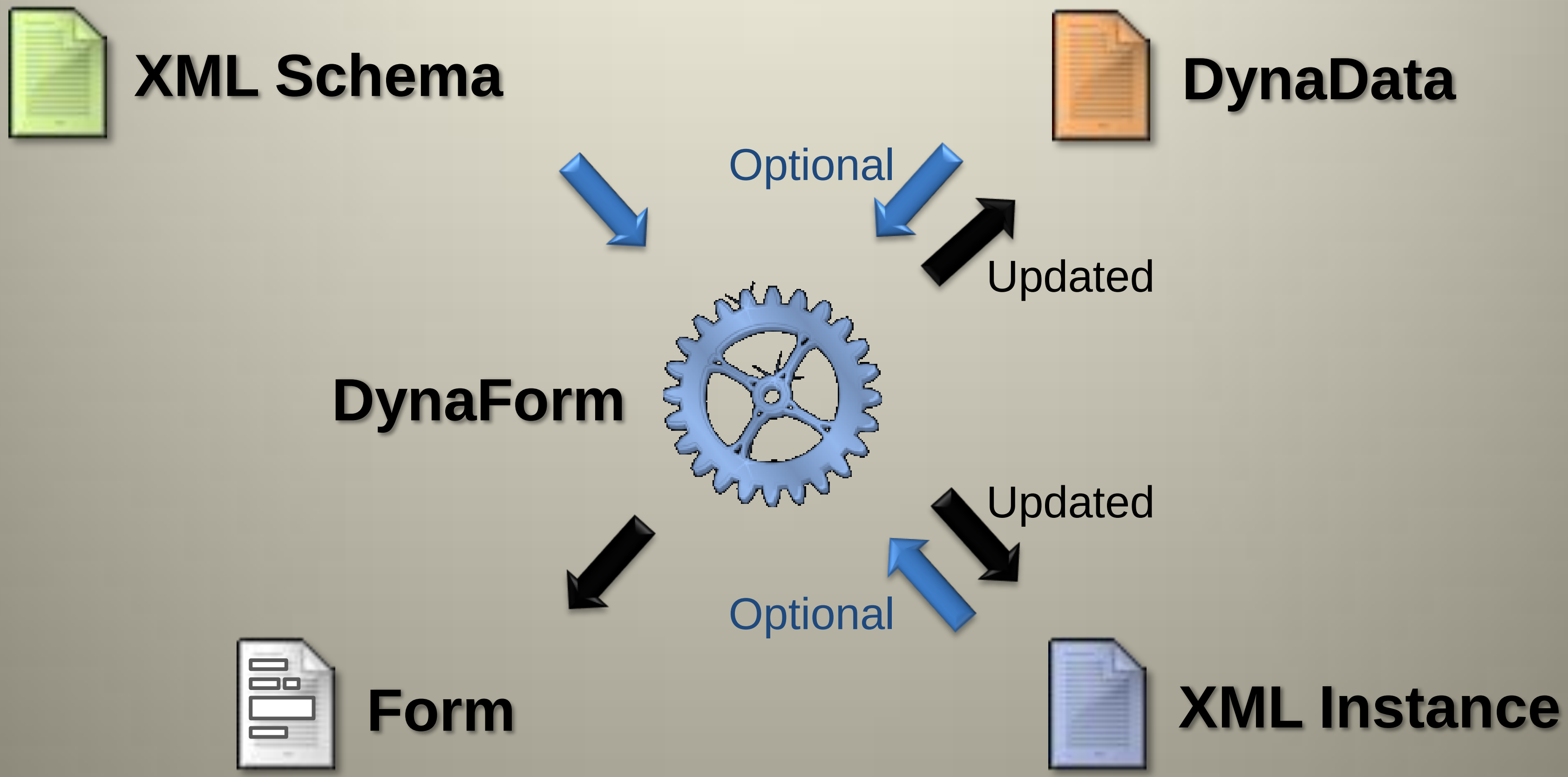
Form Generator

4. Customizing Form

Inputs and Outputs of DynaForm



Inputs and Outputs of DynaForm



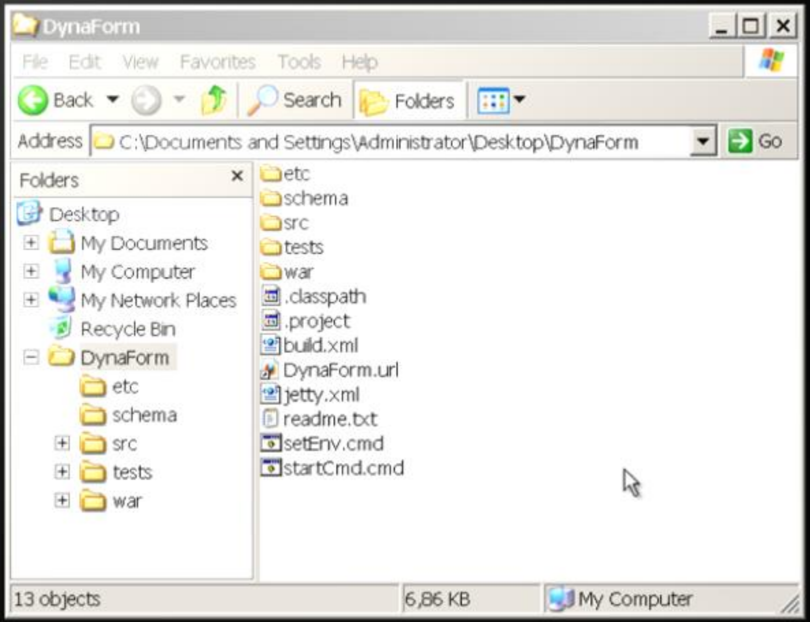
Dynadata Example

```
@CUSTOMIZED
//quantity { label: “Qty."; control: Text; size: 3; }
//price { label: "Price"; control: Text; size: 5; }
```

```
@BROKEN
//title { label: "Title"; control: Text; size: 30; }
//note { label: "Note"; control: TextArea; }
```

```
@GENERATED
//zip { label: "zip"; control: Input; }
//productName { label: "productName";control: Input;}
```

Demo



C:\Documents and Settings\Administrator\Desktop\DynaForm

etc
schema
src
tests
war
.classpath
.project
build.xml
DynaForm.url
jetty.xml
readme.txt
setEnv.cmd
startCmd.cmd

13 objects | 6,86 KB | My Computer

Conclusions

1. The form generator was designed and implemented.

2. **Possible future work:**
   1. Support other web frameworks.
   2. Improve layout techniques.
   3. Support more XSD elements.

Outline of the Thesis

1. Introduction
2. Requirements
3. State of the Art
4. High-Level Architecture
5. Walkthrough

Appendix 1. Implementation Details

Appendix 2. CD with the Source Code

Thank You! Questions?

http://code.google.com/p/xsd-web-forms/