Workflows in Perl

Chris Winters

Optiron Corporation

June 17, 2004

What are workflows?

You see them every day:

- Purchase order requests
- Bug tracking system
- Customer relationship management

Lots of glue

- Workflows glue different processes together

Lots of glue

- Workflows glue different processes together
- Hey, Perl is a glue language

Lots of glue

- Workflows glue different processes together
- Hey, Perl is a glue language
- Natural fit!

Where's Perl?

No CPAN module for workflow systems

- Lots of in-house work done? Maybe
- Fat lot of good that does us...

Workflow CPAN module

Embeddable, standalone workflow engine

- Comes with working example
- ...and tests!
- ...and decent docs!

Simple state machine

State == "What's it doing now?"

Transition == "What can it do next?"

Workflow module lingo

- **Actions** move application from state to state
- **Validators** check data moving into the workflow
- **Conditions** ensure that you can execute an action
- ...all of them are just normal classes

How you create the workflow

- Configuration!

How you create the workflow

- Configuration!
- ...sorry, it's in XML (tried INI, honest)

A set of XML files

One each for:

- workflow
- actions
- conditions
- validators

Naming the workflow

The workflow itself is just a few properties...

```
<workflow>
 <type>Issue</type>
 <description>Track issues</description>
 <persister>MyPersister</persister>
```

```
# Create a new one:
my $wf = Workflow::Factory
            ->create_workflow( 'Issue' );

# Get an existing one:
my $wf = Workflow::Factory
            ->fetch_workflow( 'Issue', 43 );
```

Workflow = states

...and a bunch of states

```
<workflow>
  <state name="INITIAL" ... />
  <state name="ISSUE_CREATED" ... />
  <state name="ISSUE_IN_PROGRESS" ... />
  <state name="ISSUE_NEEDS_APPROVED" ... />
  <state name="ISSUE_CLOSED" ... />
```

Defining a state

A state contains actions to execute:

```
<state name="INITIAL">
  <description>Initial issue state</description>
  <action name="create_issue"
          resulting_state="ISSUE_CREATED"/>
</state>
```

Names link everything

Our action name links to another declaration:

```
<action name="create_issue"
        class="App::Action::CreateIssue">
  <description>Create a new issue</description>
  <field name="due_date"
         label="Due Date"
         description="Format: yyyy-mm-dd hh:mm)"/>
    <validator name="DateValidator">
      <arg>$due_date</arg>
    </validator>
  </field>
```

Another name for the validator

And from the action's validator reference:

```
<validator
  name="DateValidator"
  class="Workflow::Validator::MatchesDateFormat">
    <description>Ensure proper dates</description>
    <param name="date_format" value="%Y-%m-%d %H:%M" />
</validator>
```

Workflow is reflective

Want to know what actions you can run? Ask the workflow!

```
my $wf = Workflow::Factory
            ->fetch_workflow( 'Issue', 42 );
my @actions = $wf->get_current_actions;
print "You can do the following actions: ",
      join( ', ',  $wf->get_current_actions );
```

More reflection

Want to know required data for an action? Ask the workflow!

```
my @action_fields = $wf->get_action_fields( $action_name );
print "Data for action '$action_name':\n";
foreach my $field ( @action_fields ) {
    my @values = $field->get_possible_values;
    print join( "\n",
      "Field name: " . $field->name,
      "Type:       " . $field->type,
      "Required?   " . $field->is_required,
      "Values:     " . join( ', ', @values ),
      "Describe:   " . $field->description
   );
}
```

Just do it!

- Is this module perfect?
- Hell no!
- But it's better than none at all...

Get up! Like a state machine...