Thoughts on the Generic vs. Specific Tradeoff

Stefan Tilkov, innoQ
QCon London 2009

http://www.innoq.com

Phases in a Developer's Life

1. The Enthusiastic Developer

"This stuff is cool -
let's build programs!
For real people!"

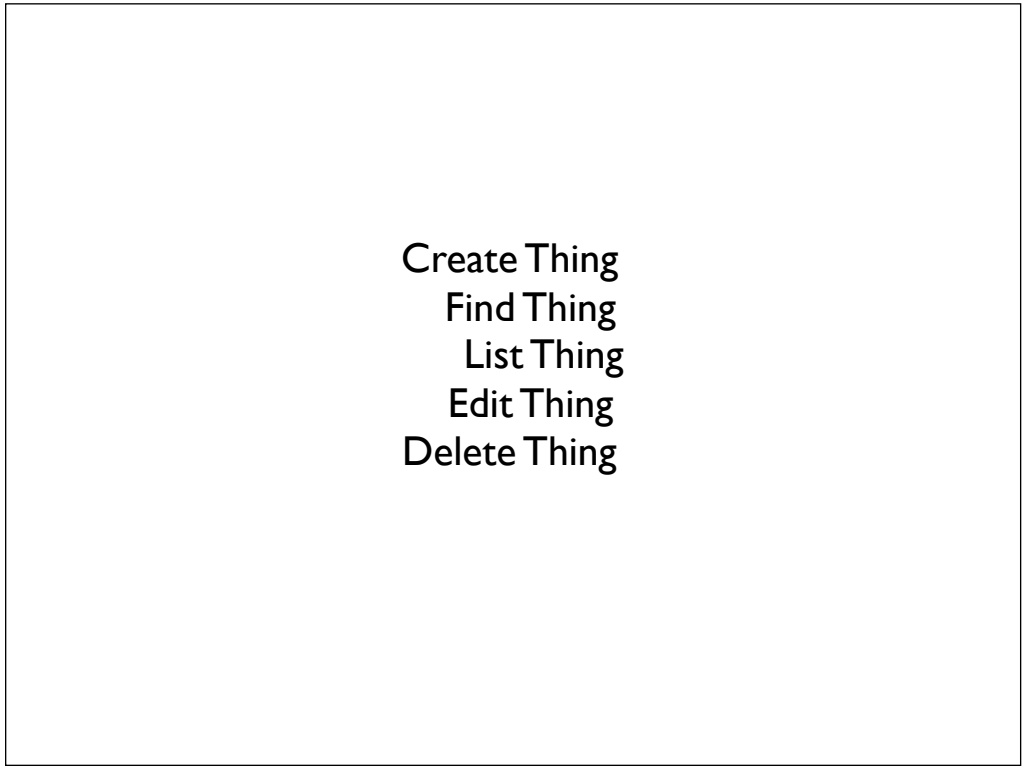
Create Customer	Create Product	Create Order
Find Customer	Find Product	Find Order
List Customers	List Products	List Orders
Edit Customer	Edit Product	Edit Order
Delete Customer	Delete Product	Delete Order

Boring, boring, boring.

2. The Disillusioned Developer

"Oh. Real people
have boring
problems."

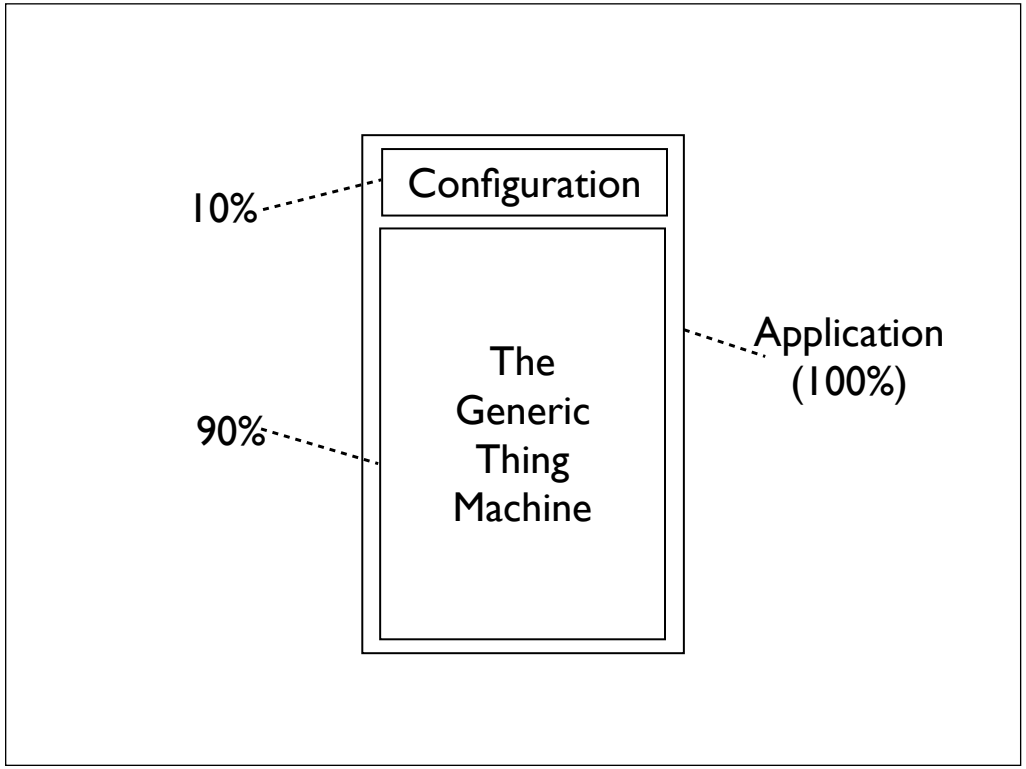

Create Customer
Find Customer
List Customers
Edit Customer
Delete Customer
Create Product
Find Product
List Products
Edit Product
Delete Product
Create Order
Find Order
List Orders
Edit Order
Delete Order

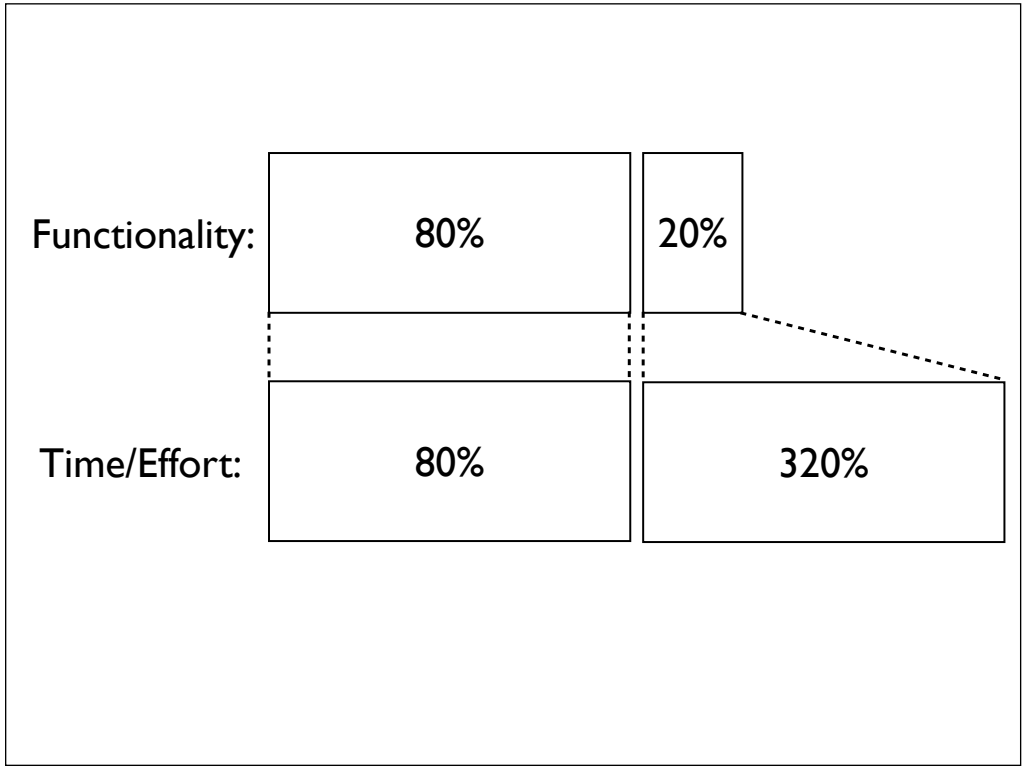

Create Thing
Find Thing
List Thing
Edit Thing
Delete Thing

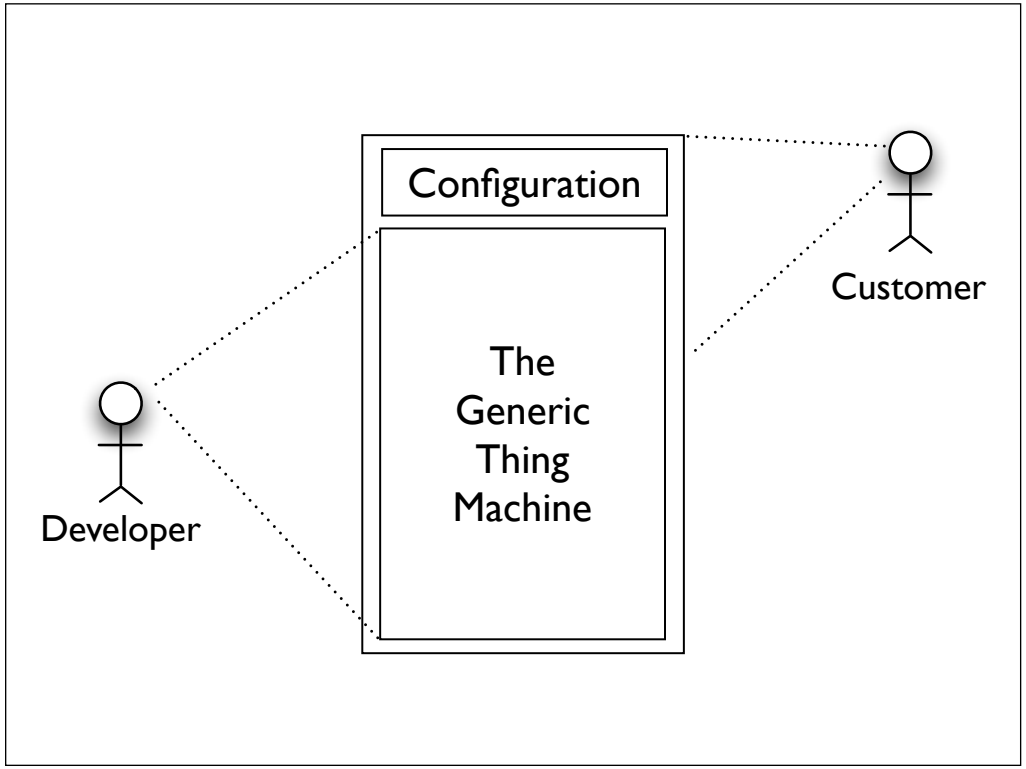
3. The Enthusiastic Architect

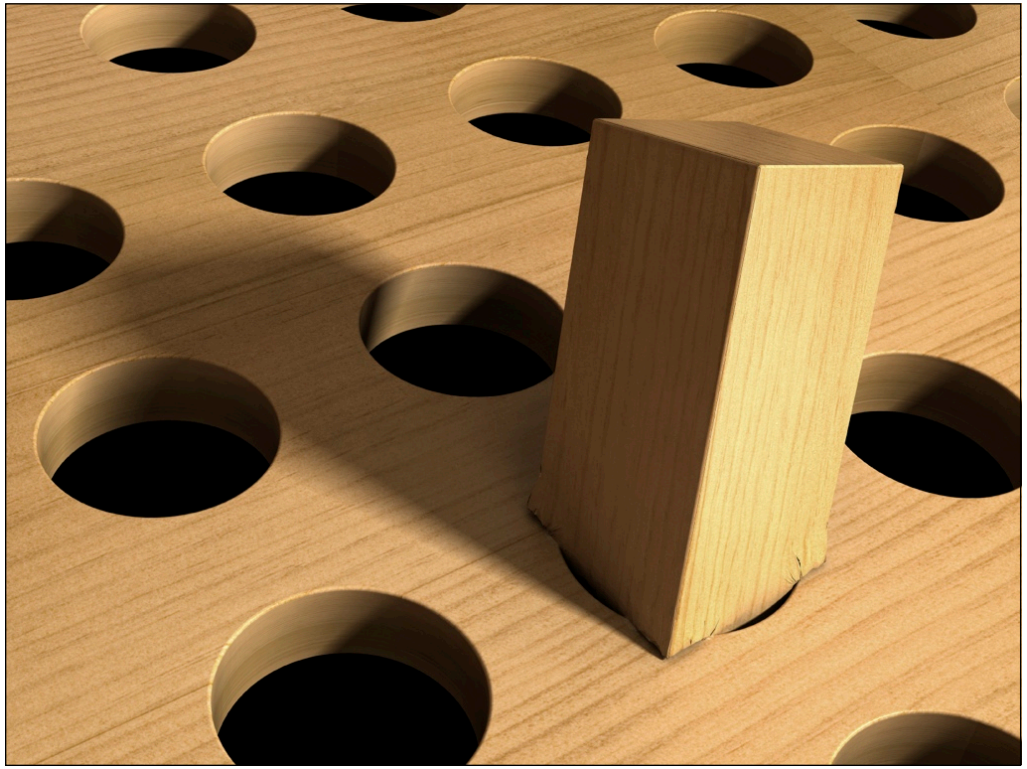
Create Thing
Find Thing
List Thing
Edit Thing
Delete Thing

"Generic solutions! Cool!"


Configuration
10%
The
Generic
Thing
Machine
90%
Application
(100%)


Functionality:
80%
20%
Time/Effort:
80%
320%


Configuration
The
Generic
Thing
Machine
Customer
Developer



4. The Disillusioned Architect

"Some programmers, when faced with a problem, turn to a generic solution … now they have two problems."

YAGNI KISS

Working software

5. The "Wise" Architect

Question: *

Answer: It depends.

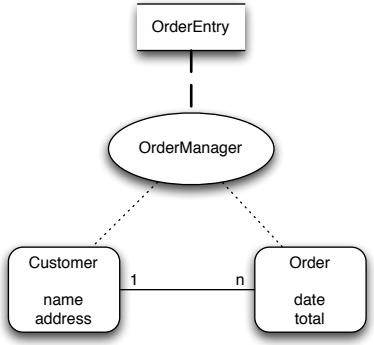
Examples

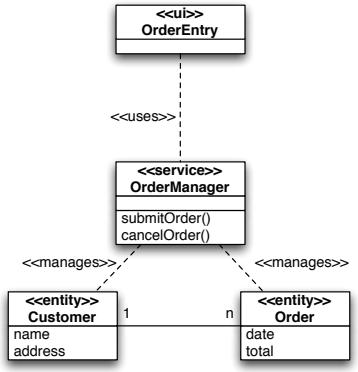
XML vs. HTML

```xml
<customer xmlns='http://example.com/schemas/crm'>
    <id>4711</id>
    <name>Schulze Systems AG</name>
    <city>Ratingen</city>
    <country>Germany</country>
</customer>
```

```html
<html>
  <head>
    <title>Customer Info</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  </head>
  <body>
    <div class="customer">
        <span class="id">4711</span>
        <span class="name">Schulze Systems AG</span>
        <span class="city">Ratingen</span>
        <span class="country">Germany</span>
    </div>
  </body>
</html>
```

DSM vs. UML


OrderEntry
OrderManager
Customer
name
address
1
n
Order
date
total


<<ui>>
OrderEntry
<<uses>>
<<service>>
OrderManager
submitOrder()
cancelOrder()
<<manages>>
<<manages>>
<<entity>>
Customer
name
address
1
n
<<entity>>
Order
date
total

External vs. Internal DSL

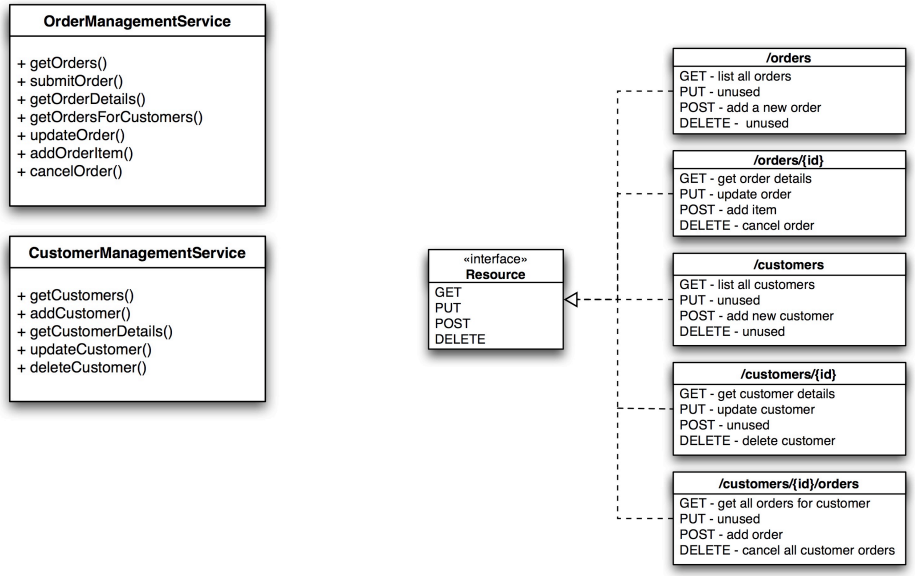
```
cancel:
 transitions from submitted to cancelled,

accept:
 transitions from received to accepted,
             from checking to checked
```

```
event :cancel do
   transitions :from => :submitted, :to => :cancelled
end

event :accept do
  transitions :from => :received, :to => :accepted
  transitions :from => :checking, :to => :checked
end
```

SOAP/WSDL vs. REST/HTTP



HTTP Verbs vs. POST Tunneling

```
PUT /xyz HTTP/1.1
<data>...</data>
```

```
POST /xyz HTTP/1.1
<update><data>...</data></update>
```

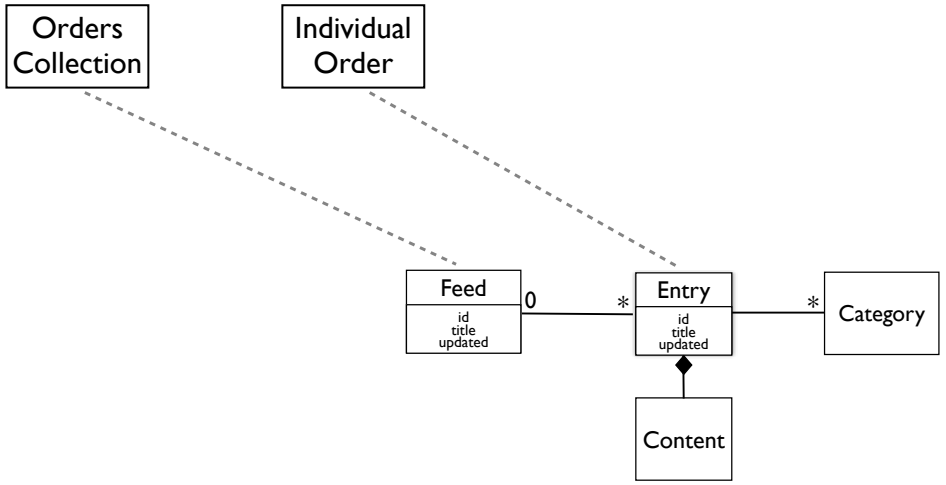
```
DELETE /xyz HTTP/1.1
```

```
POST /xyz HTTP/1.1
<delete>...</delete>
```

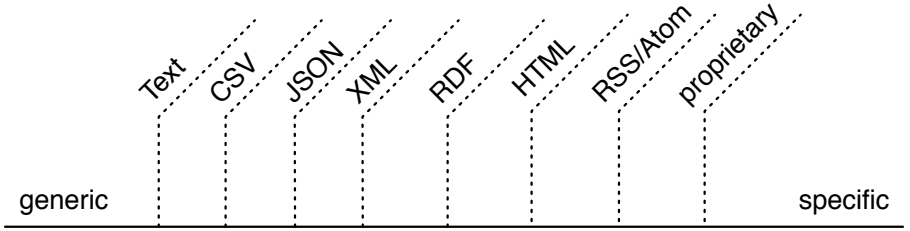
```
PATCH /xyz HTTP/1.1
<diff>...</diff>
```

```
POST /xyz HTTP/1.1
<diff>...</diff>
```

Custom Protocol vs. AtomPub



Formats Continuum



RDBMS Tables vs. Metatables

Customer

id	name	address	status

Order

id	date	amount	total	cust_id

Class

id	name
1	Customer
2	Order

Attribute

id	name	type	class_id
1	cust_no	int	1
2			

The List Goes on and on …

Smalltalk Image vs. Filebased IDEs

Custom-built Web App vs. CMS

Custom Protocols vs. Standards

Maven vs. Ant (vs. scripts)

Considerations

Problem/Solution
Congruence

Diversity

Ramp-up Cost

Development
Performance

Runtime Performance

Knowledge

Skill

Folklore

Ecosystem

XML

1. View it in tree rendering
2. Check for wellformedness
3. Run XSLT on it
4. Query with XPath
5. Process with XQuery
6. Validate against schema
7. Encrypt/Decrypt parts
8. Sign and verify signature
9. Archive it
10. Process w/ SAX/DOM

HTTP & URIs

1. Embed links in representations
2. Drive application flow
3. Expose Multiple Representations
4. Use curl/wget
5. Control access
6. Get indexed by Google (public or appliance)
7. Bookmark or email Links
8. Redirect
9. Use 404, 412, 409
10. Use Caches

SOAP/WSDL/WS-*

1. ESBs
2. Platforms
3. Tooling
4. Intermediaries
5. Standard software
6. Mainstream choice
7. People
8. Politics
9. Hype
10. Job security

RDBMS

1. Standard Query Language
2. Optimized access
3. Parallel processing
4. Scalability & Performance
5. Metadata management
6. Report generators & BI Tools
7. Hot backup
8. Portability
9. Program-independent storage
10. Caching

Files

1. Search
2. Backup
3. Debug
4. Diff
5. Edit
6. Version control
7. Import/Export
8. Convert
9. Generate
10. Process

UML

Concepts

Diagram types

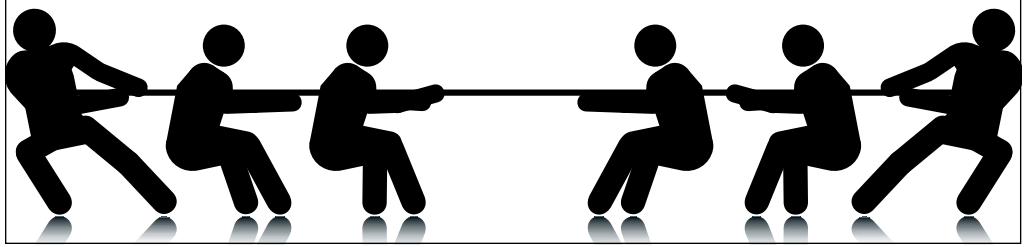
CASE Tools

DSM

Eclipse EMF

MS SW Factories

Generic	Specific
Useful ecosystem	Niche needs
“Obvious” match	“Unique” problem
Existing skills	Performance
Static environment	Soft environment



Q&A

Stefan Tilkov
stefan.tilkov@innoq.com
http://www.innoq.com/blog/st/
Twitter: stilkov