Modeling Variability
(Jhipster case study)

Mathieu Acher
Maître de Conférences
mathieu.acher@irisa.fr

UNIVERSITÉ DE RENNES 1

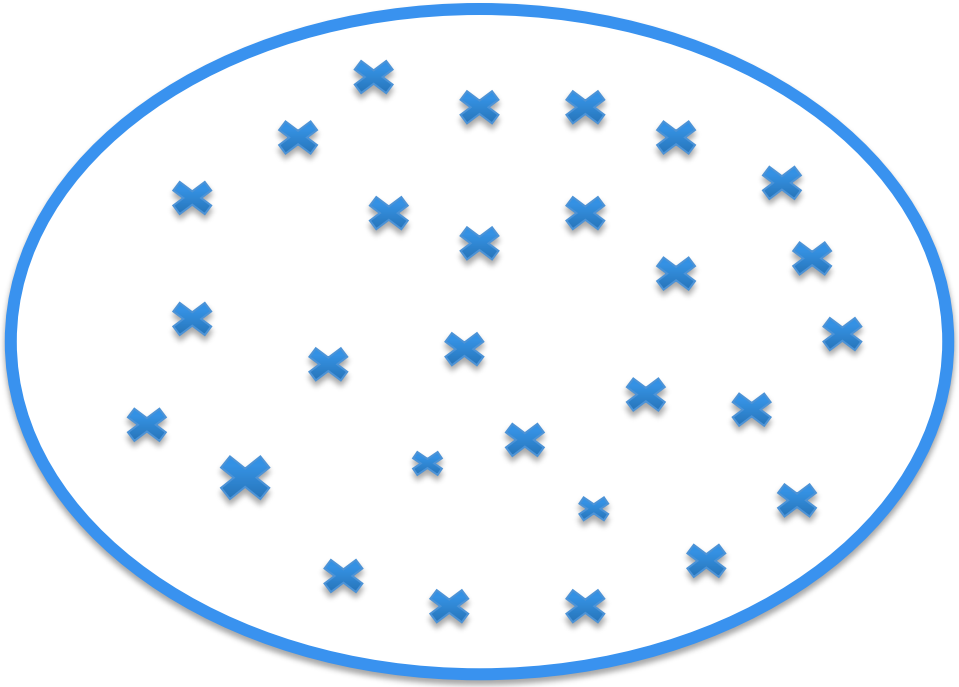
istic Informatique Électronique

Material

http://mathieuacher.com/teaching/MDE/MRI1516/

```
macher-wifi:getting-started macher1$ yo jhipster

I'm all done. Running npm install & bower install for you to install the required dependencies.

 _     _   ___   __  _____  ____  ___      __  _____   __    __   _
| |_| | | | |_) ( (`  | |  | |_  | |_)    ( (`  | |   / /\  / /`  | |_/
|_| | |_| |_|   _)_)  |_|  |_|__ |_| \    _)_)  |_|  /_/--\ \_\_, |_| \
                          ____  ___   ___
                         | |_  / / \ | |_)
                         |_|   \_\_/ |_| \
            _    __    _       __       ___   ____  _      __
           | |  / /\  \ \  /  / /\     | | \ | |_  \ \  / ( (`
          \_|_| /_/--\  \_\/  /_/--\    |_|_/ |_|__  \_\/  _)_)

Welcome to the JHipster Generator v2.17.0

? (1/15) What is the base name of your application? jhipster
? (2/15) What is your default Java package name? com.mycompany.myapp
? (3/15) Do you want to use Java 8? Yes (use Java 8)
? (4/15) Which *type* of authentication would you like to use? (Use arrow keys)
❯ HTTP Session Authentication (stateful, default Spring Security mechanism)
  OAuth2 Authentication (stateless, with an OAuth2 server implementation)
  Token-based authentication (stateless, with a token)
```

```
macher-wifi:getting-started macher1$ yo jhipster


I'm all done. Running npm install & bower install for you to install the required dependencies.



 _     _   ___   __  _____  ____  ___      __  _____   __    __    _
| |_| | | | |_) ( (`  | |  | |_  | |_)    ( (`  | |   / /\  / /`   | |_/
|_| | |_| |_|   _)_)  |_|  |_|__ |_| \    _)_)  |_|  /_/--\ \_\_, |_| \
                          ____  ___   ___
                         | |_  / / \ | |_)
                         |_|   \_\_/ |_| \
           _    __    _       __        ___   ____  _     __
          | |  / /\  \ \  /  / /\      | | \ | |_  \ \  / ( (`
        \_|_| /_/--\  \_\/  /_/--\     |_|_/ |_|__  \_\/  _)_)


Welcome to the JHipster Generator v2.17.0

? (1/15) What is the base name of your application? jhipster
? (2/15) What is your default Java package name? com.mycompany.myapp
? (3/15) Do you want to use Java 8? Yes (use Java 8)
? (4/15) Which *type* of authentication would you like to use? (Use arrow keys)
❯ HTTP Session Authentication (stateful, default Spring Security mechanism)
  OAuth2 Authentication (stateless, with an OAuth2 server implementation)
  Token-based authentication (stateless, with a token)
```



Jhipster variant

```
macher-wifi:getting-started macher1$ yo jhipster

I'm all done. Running npm install & bower install for you to install the required dependencies.


 _     _   ___   __  ____  ____  __      __  ____   __    __    _
| |_| | | | |_) ( (` | |_  | |_)     ( (` | |_   / /\  / /`  | |_/
|_| | |_| |_|   _)_) |_|  |_|__ |_| \     _)_) |_|  /_/--\ \_\_, |_| \
                                 ___   ___   ___
                                | |_  / / \ | |_)
                                |_|   \_\_/ |_| \
          _   __   _     __      ___   ___  _     __
         | | / /\  \ \   / /\     | | \ | |_  \ \  / ( (`
        \_|_| /_/--\  \_\/  /_/--\    |_|_/ |_|__  \_\/  _)_)


Welcome to the JHipster Generator v2.17.0

? (1/15) What is the base name of your application? jhipster
? (2/15) What is your default Java package name? com.mycompany.myapp
? (3/15) Do you want to use Java 8? Yes (use Java 8)
? (4/15) Which *type* of authentication would you like to use? (Use arrow keys)
❯ HTTP Session Authentication (stateful, default Spring Security mechanism)
  OAuth2 Authentication (stateless, with an OAuth2 server implementation)
  Token-based authentication (stateless, with a token)
```

Variability Model

mapping

Branch: master

generator-jhipster / app / templates / src / main / java / package / config / _DatabaseConfiguration.java

jdubois 2 days ago Use Spring Boot's configuration meta-data

9 contributors

184 lines (165 sloc) | 9.69 KB

Raw Blame History

```
1  package <%=packageName%>.config;
2  <% if (databaseType == 'sql') { %>
3  import <%=packageName%>.config.liquibase.AsyncSpringLiquibase;
4  import com.codahale.metrics.MetricRegistry;
5  import com.fasterxml.jackson.datatype.hibernate4.Hibernate4Module;
6  import com.zaxxer.hikari.HikariConfig;
7  import com.zaxxer.hikari.HikariDataSource;
8  import liquibase.integration.spring.SpringLiquibase;<% } %><% if (databaseType == 'mongodb' && authenticationType == 'oauth2') { %
9  import <%=packageName%>.config.oauth2.OAuth2AuthenticationReadConverter;<% } %><% if (databaseType == 'mongodb') { %>
10 import com.mongodb.Mongo;
11 import org.mongeez.Mongeez;<% } %>
12 import org.slf4j.Logger;
13 import org.slf4j.LoggerFactory;<% if (databaseType == 'sql') { %><% if (hibernateCache == 'hazelcast') { %>
14 import org.springframework.cache.CacheManager;<% } %>
15 import org.springframework.beans.factory.annotation.Autowired;
16 import org.springframework.boot.autoconfigure.condition.ConditionalOnExpression;<% } %><% if (databaseType == 'mongodb') { %>
17 import org.springframework.boot.autoconfigure.mongo.MongoAutoConfiguration;
18 import org.springframework.boot.autoconfigure.mongo.MongoProperties;<% } %><% if (databaseType == 'sql') { %>
19 import org.springframework.boot.autoconfigure.jdbc.DataSourceProperties;
20 import org.springframework.boot.autoconfigure.liquibase.LiquibaseProperties;
21 import org.springframework.context.ApplicationContextException;<% } %>
22 import org.springframework.context.annotation.Bean;
23 import org.springframework.context.annotation.Configuration;
24 import org.springframework.context.annotation.Profile;<% if (databaseType == 'mongodb') { %>
25 import org.springframework.context.annotation.Import;<% } %><% if (databaseType == 'sql') { %>
26 import org.springframework.core.env.Environment;<% } %><% if (databaseType == 'mongodb' && authenticationType == 'oauth2') { %>
27 import org.springframework.core.convert.converter.Converter;<% } %><% if (databaseType == 'mongodb') { %>
28 import org.springframework.core.io.ClassPathResource;<% } %><% if (searchEngine == 'elasticsearch') { %>
29 import org.springframework.data.elasticsearch.repository.config.EnableElasticsearchRepositories;<% } %><% if (databaseType == 'mon
30 import org.springframework.data.mongodb.config.AbstractMongoConfiguration;
31 import org.springframework.data.mongodb.config.EnableMongoAuditing;<% } %><% if (databaseType == 'mongodb' && authenticationType =
32 import org.springframework.data.mongodb.core.convert.CustomConversions;<% } %><% if (databaseType == 'mongodb') { %>
33 import org.springframework.data.mongodb.core.mapping.event.ValidatingMongoEventListener;
34 import org.springframework.data.mongodb.repository.config.EnableMongoRepositories;
35 import org.springframework.validation.beanvalidation.LocalValidatorFactoryBean;<% } %><% if (databaseType == 'sql') { %>
```

Base Artefacts

Software Generator

(derivation engine)

```
macher-wifi:getting-started macher1$ yo jhipster

I'm all done. Running npm install & bower install for you to install the required dependencies

 _     _   ___   __  _____  ____  ___      __  _____  __    __   _
| |_| | | | |_) ( (`  | |  | |_  | |_)    ( (`  | |  / /\  / /`  | |_
|_| | |_| |_|   _)_)  |_|  |_|__ |_| \    _)_)  |_|  /_/--\ \_\_, |_|
                                 ____  ___  ___
                                | |_  / / \ | |_)
                                |_|   \_\_/ |_| \
   _    __    _      __      ___   ____  _    __
  | |  / /\  \ \  /  / /\    | | \ | |_  \ \  / ( (`
 \_|_| /_/--\  \_\/  /_/--\   |_|_/ |_|__  \_\/  _)_)

Welcome to the JHipster Generator v2.17.0

? (1/15) What is the base name of your application? jhipster
? (2/15) What is your default Java package name? com.mycompany.myapp
? (3/15) Do you want to use Java 8? Yes (use Java 8)
? (4/15) Which *type* of authentication would you like to use? (Use arr
❯ HTTP Session Authentication (stateful, default Spring Security mechan
  OAuth2 Authentication (stateless, with an OAuth2 server implementatio
  Token-based authentication (stateless, with a token)
```

Branch: master

generator-jhipster / app / templates / src / main / java / package / config / _DatabaseConfiguration.java

jdubois 2 days ago Use Spring Boot's configuration meta-data

9 contributors

184 lines (165 sloc) | 9.69 KB

Raw Blame History

```
package <%=packageName%>.config;
<% if (databaseType == 'sql') { %>
import <%=packageName%>.config.liquibase.AsyncSpringLiquibase;
import com.codahale.metrics.MetricRegistry;
import com.fasterxml.jackson.datatype.hibernate4.Hibernate4Module;
import com.zaxxer.hikari.HikariConfig;
import com.zaxxer.hikari.HikariDataSource;
import liquibase.integration.spring.SpringLiquibase;<% } %><% if (databaseType == 'mongodb' && authenticationType == 'oauth2') { %
import <%=packageName%>.config.oauth2.OAuth2AuthenticationReadConverter;<% } %><% if (databaseType == 'mongodb') { %>
import com.mongodb.Mongo;
import org.mongeez.Mongeez;<% } %>
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;<% if (databaseType == 'sql') { %><% if (hibernateCache == 'hazelcast') { %>
import org.springframework.cache.CacheManager;<% } %>
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.autoconfigure.condition.ConditionalOnExpression;<% } %><% if (databaseType == 'mongodb') { %>
import org.springframework.boot.autoconfigure.mongo.MongoAutoConfiguration;
import org.springframework.boot.autoconfigure.mongo.MongoProperties;<% } %><% if (databaseType == 'sql') { %>
import org.springframework.boot.autoconfigure.jdbc.DataSourceProperties;
import org.springframework.boot.autoconfigure.liquibase.LiquibaseProperties;
import org.springframework.context.ApplicationContextException;<% } %>
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.Profile;<% if (databaseType == 'mongodb') { %>
import org.springframework.context.annotation.Import;<% } %><% if (databaseType == 'sql') { %>
import org.springframework.core.env.Environment;<% } %><% if (databaseType == 'mongodb' && authenticationType == 'oauth2') { %>
import org.springframework.core.convert.converter.Converter;<% } %><% if (databaseType == 'mongodb') { %>
import org.springframework.core.io.ClassPathResource;<% } %><% if (searchEngine == 'elasticsearch') { %>
import org.springframework.data.elasticsearch.repository.config.EnableElasticsearchRepositories;<% } %><% if (databaseType == 'mon
import org.springframework.data.mongodb.config.AbstractMongoConfiguration;
import org.springframework.data.mongodb.config.EnableMongoAuditing;<% } %><% if (databaseType == 'mongodb' && authenticationType =
import org.springframework.data.mongodb.core.convert.CustomConversions;<% } %><% if (databaseType == 'mongodb') { %>
import org.springframework.data.mongodb.core.mapping.event.ValidatingMongoEventListener;
import org.springframework.data.mongodb.repository.config.EnableMongoRepositories;
import org.springframework.validation.beanvalidation.LocalValidatorFactoryBean;<% } %><% if (databaseType == 'sql') { %>
```

Jhipster variant

For each release,
how to verify that all variants of Jhipster
are valid?

Unused flexibility


Illegal variant

Research Question

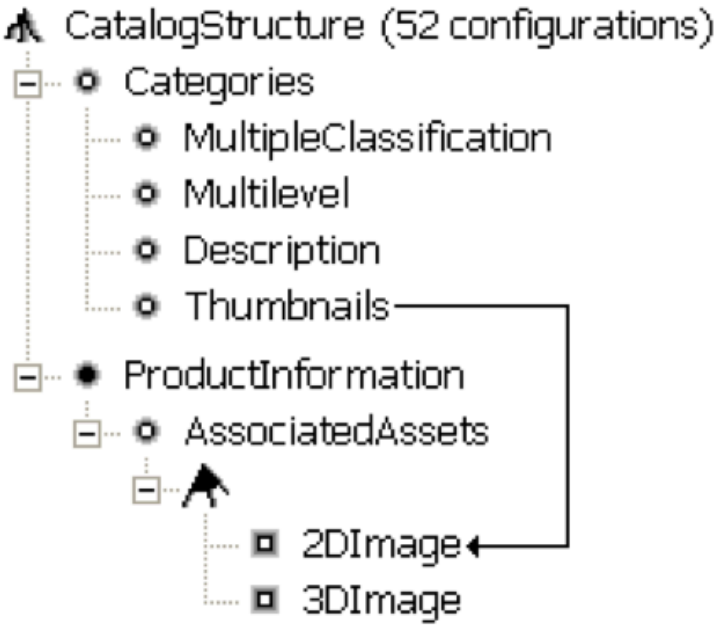
- How to verify that all variants are valid?
  - A very general problem
  - “variants”: code, models, etc.
  - notion of “validity”: very broad
- Many theoretical approaches
- Some tools
- Some empirical results

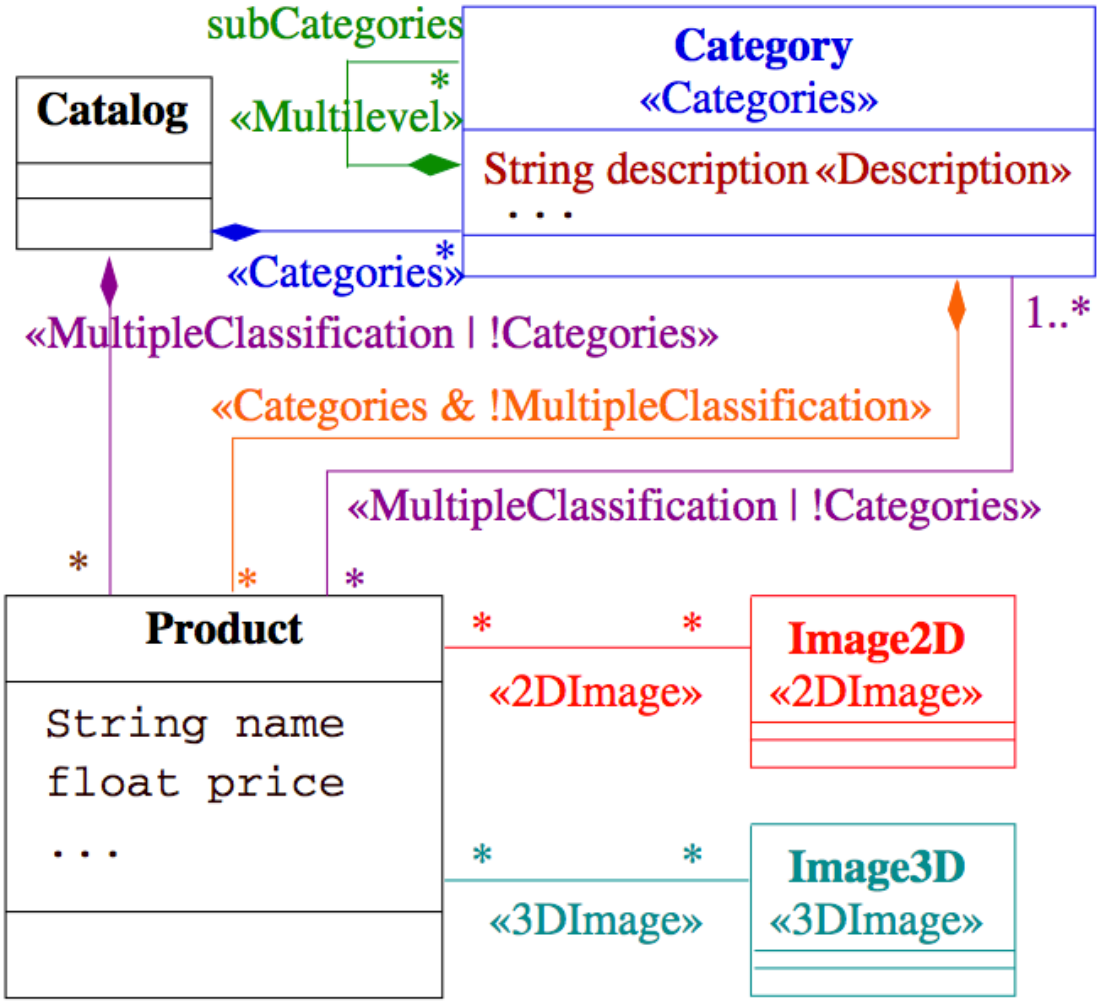
How to verify that all variants are valid?

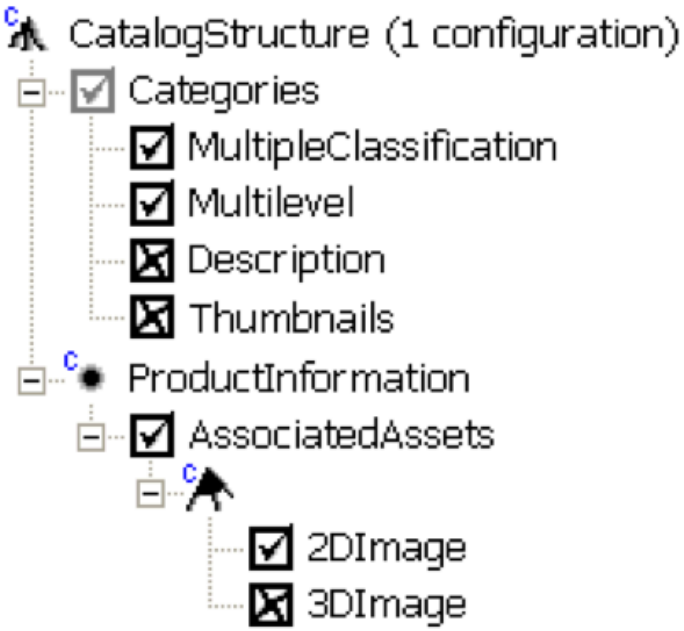
Verifying Feature-Based Model Templates Against Well-Formedness OCL Constraints

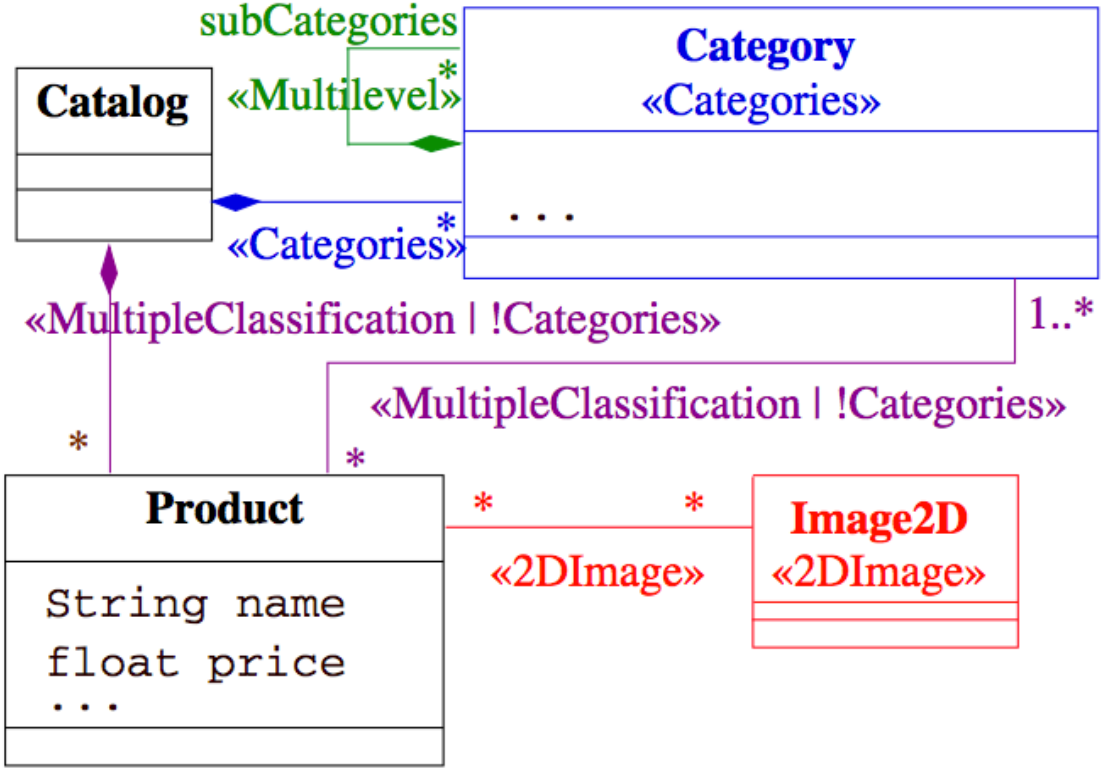
Krzysztof Czarnecki Krzysztof Pietroszek
University of Waterloo, Canada
{kczarnec,kmpietro}@swen.uwaterloo.ca

How to verify that all variants are valid?

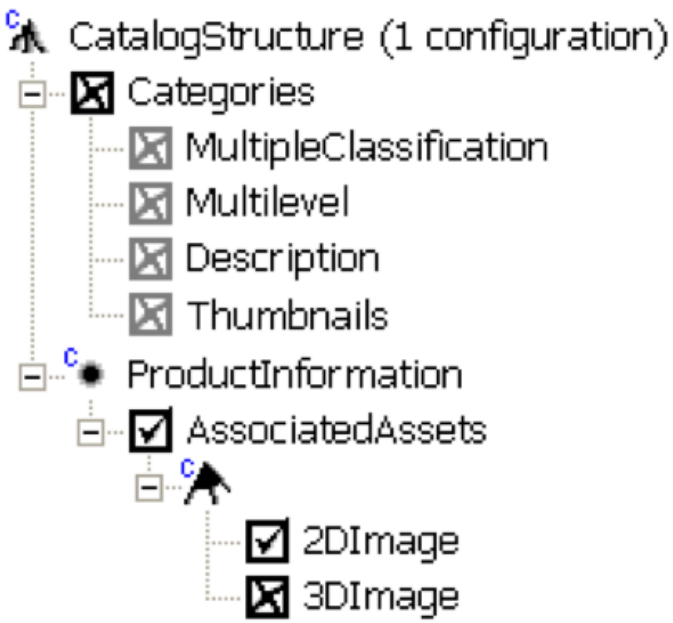


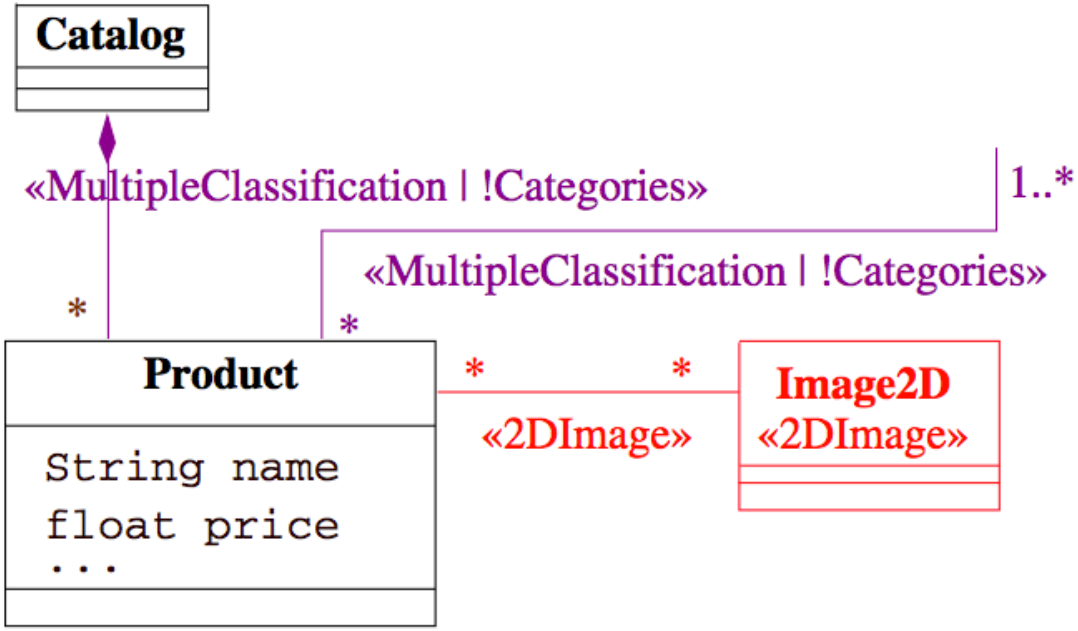



CatalogStructure (1 configuration)
Categories
MultipleClassification
Multilevel
Description
Thumbnails
ProductInformation
AssociatedAssets
2DImage
3DImage

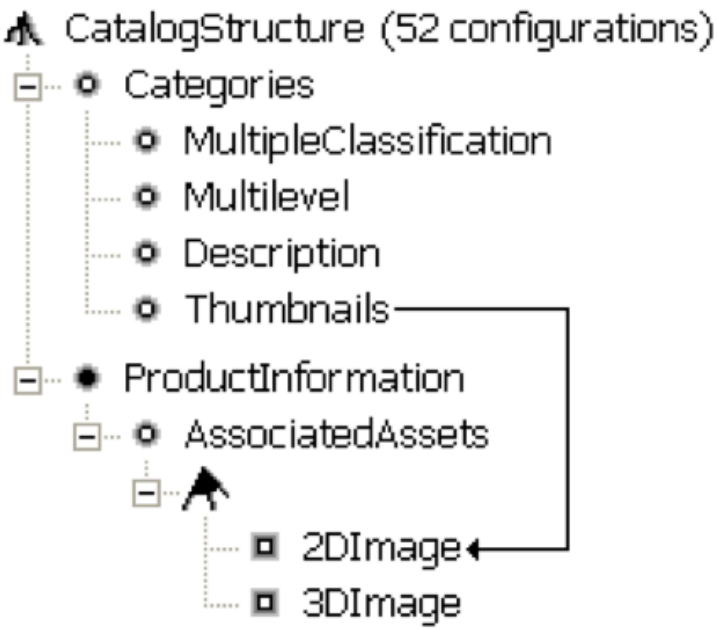

subCategories
Catalog
«Multilevel»
*
Category
«Categories»
. . .
«Categories»
*
«MultipleClassification | !Categories»
1..*
«MultipleClassification | !Categories»
*
*
Product
String name
float price
...
*
*
«2DImage»
Image2D
«2DImage»

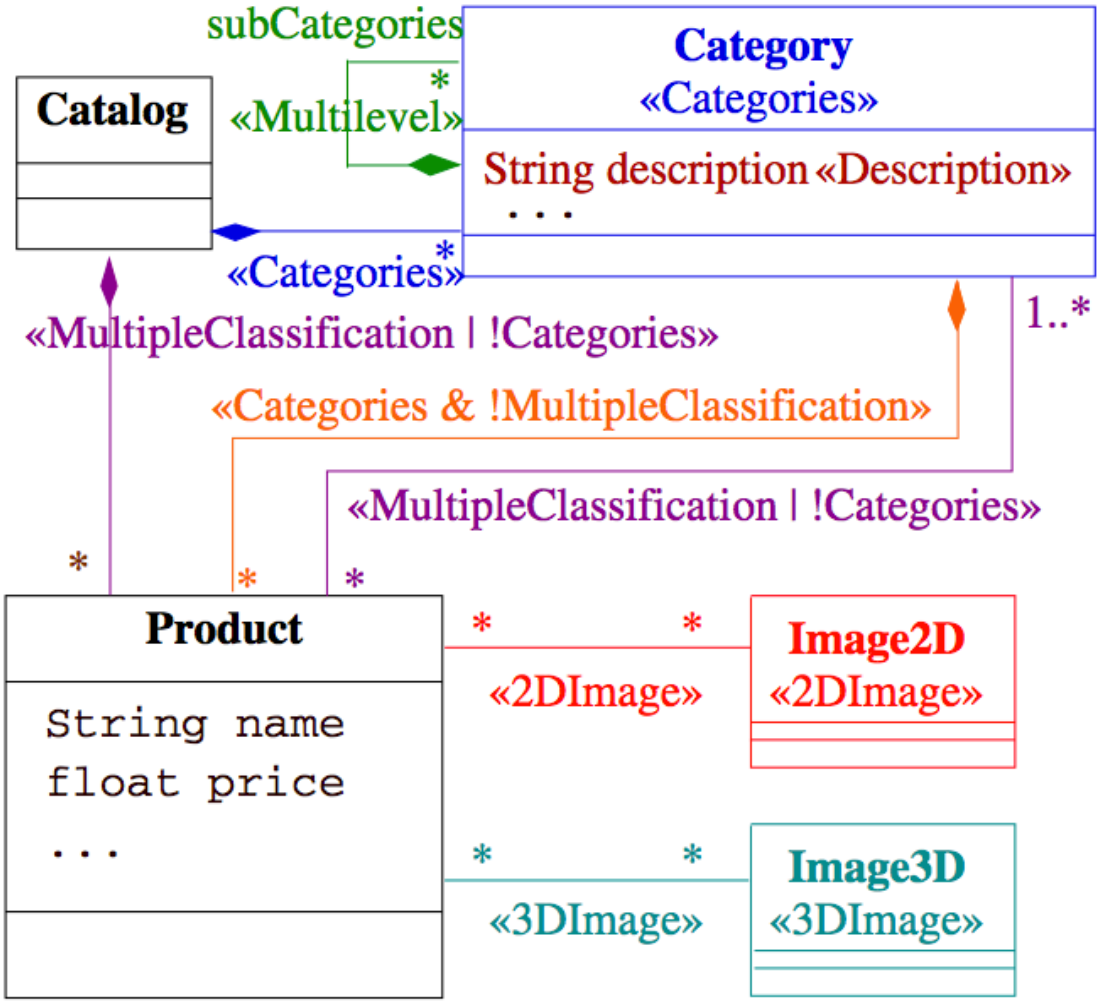
Ooops


CatalogStructure (1 configuration)
Categories
MultipleClassification
Multilevel
Description
Thumbnails
ProductInformation
AssociatedAssets
2DImage
3DImage

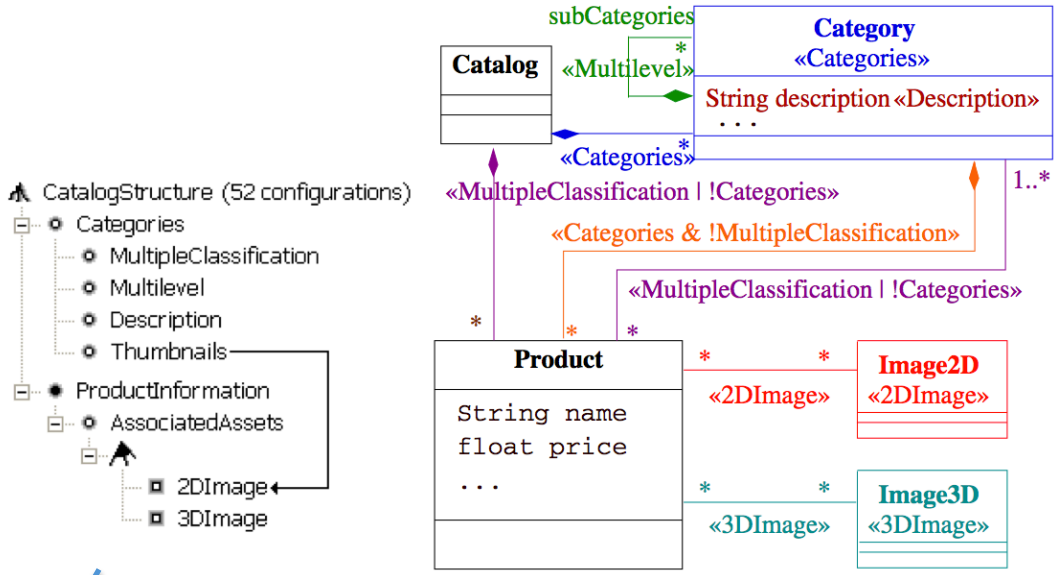

Catalog
«MultipleClassification | !Categories»
1..*
«MultipleClassification | !Categories»
*
*
Product
String name
float price
...
*
*
«2DImage»
Image2D
«2DImage»

Safe composition? No!





Safe composition: how does it work?



$$
\begin{aligned}
q_{FM} \quad &= \\
\textit{root:} \quad & cs \wedge \\
\textit{child-parent:} \quad & (\texttt{ct} \Rightarrow \texttt{cs}) \wedge (\texttt{mc} \Rightarrow \texttt{ct}) \wedge (\texttt{ml} \Rightarrow \texttt{ct}) \wedge \\
& (\texttt{ds} \Rightarrow \texttt{ct}) \wedge (\texttt{tn} \Rightarrow \texttt{ct}) \wedge (\texttt{pi} \Rightarrow \texttt{cs}) \wedge \\
& (\texttt{aa} \Rightarrow \texttt{pi}) \wedge (\texttt{i2} \Rightarrow \texttt{aa}) \wedge (\texttt{i3} \Rightarrow \texttt{aa}) \wedge \\
\textit{group:} \quad & (\texttt{aa} \Rightarrow \textit{choice}_{1,2}(\texttt{i2}, \texttt{i3})) \wedge \\
\textit{mandatory:} \quad & (\texttt{cs} \Rightarrow \texttt{pi}) \wedge \\
\textit{additional:} \quad & (\texttt{tn} \Rightarrow \texttt{i2})
\end{aligned}
$$

Another propositional formula

How to verify that all variants are valid?

Model Checking Lots of Systems

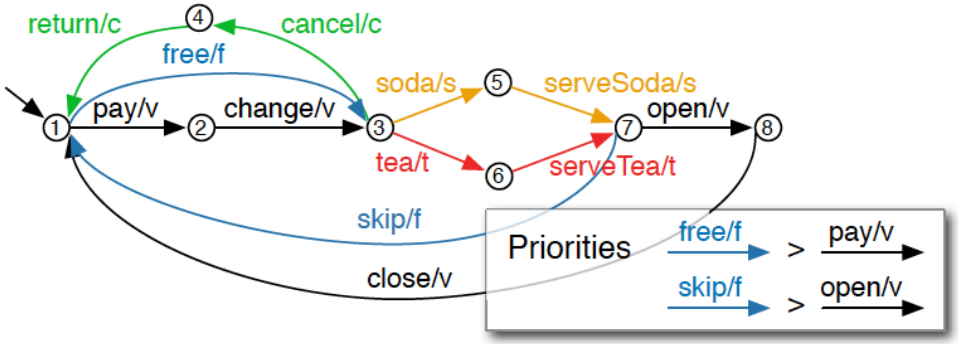
Efficient Verification of Temporal Properties in Software Product Lines

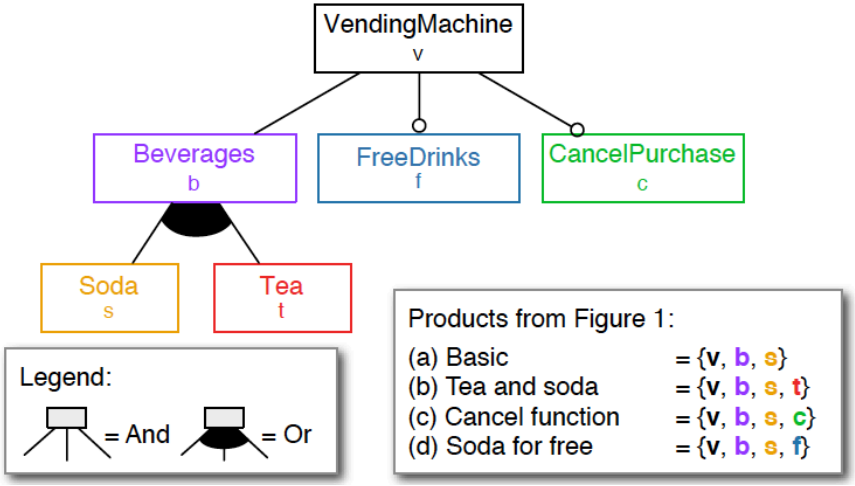
Andreas Classen,*
Patrick Heymans,
Pierre-Yves Schobbens
University of Namur, Belgium
{acs,phe,pys}
@info.fundp.ac.be

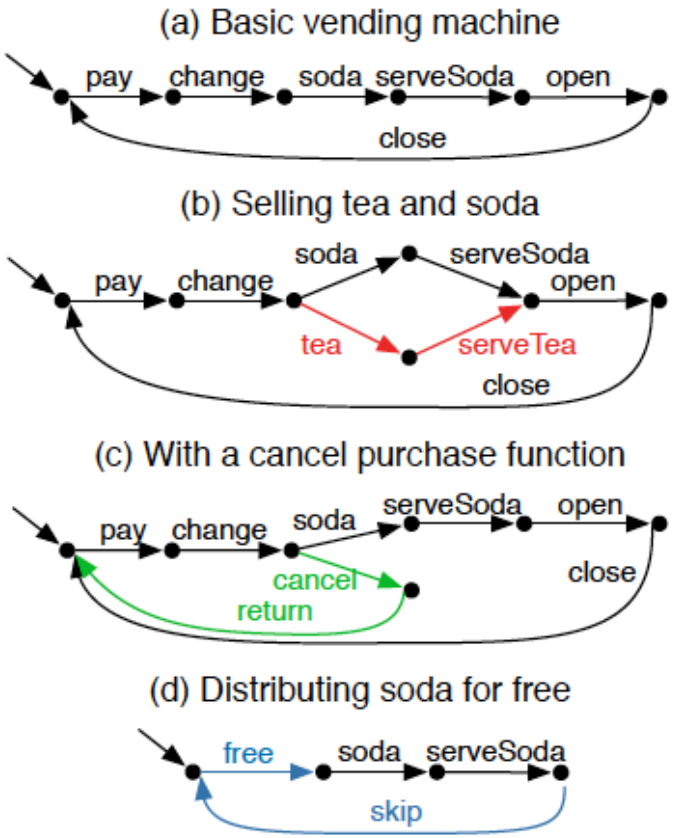
Axel Legay
IRISA/INRIA Rennes, France
axel.legay@irisa.fr

Jean-François Raskin
Université Libre de Bruxelles, Belgium
jraskin@ulb.ac.be

How to verify that all variants are valid?







A Classification and Survey of Analysis Strategies for Software Product Lines

THOMAS THÜM, University of Magdeburg, Germany
SVEN APEL, University of Passau, Germany
CHRISTIAN KÄSTNER, Carnegie Mellon University, USA
INA SCHAEFER, University of Braunschweig, Germany
GUNTER SAAKE, University of Magdeburg, Germany

Research Question

- How to verify that all variants are valid?
- Many theoretical approaches, some tools, some empirical results
- Jhipster acts as a case study

  "an empirical inquiry that investigates a contemporary phenomenon within its real-life context"

Research Question

- How to verify that all variants are valid?
- Jhipster acts as a case study with a quite important configuration space; some bugs related to configurations have been reported
- What are the most cost-effective verification techniques?
  - e.g., ability to find configuration bugs
- Some sub-problems and challenges

```
macher-wifi:getting-started macher1$ yo jhipster

I'm all done. Running npm install & bower install for you to install the required dependencies

 _     _   ___   __  _____  ____  ___      __  _____   __    __    _
| |_| | | | |_) ( (`  | |  | |_  | |_)    ( (`  | |   / /\  / /`  | |_
|_| | |_| |_|   _)_)  |_|  |_|__ |_| \    _)_)  |_|  /_/--\ \_\_, |_|
                                  ____  ___   ___
                                 | |_  / / \ | |_)
                                 |_|   \_\_/ |_| \

 _    __    _      __      ___   ____  _     __
| |  / /\  \ \  /  / /\    | | \ | |_  \ \  / ( (`
\_|_| /_/--\  \_\/  /_/--\  |_|_/ |_|__  \_\/  _)_)

Welcome to the JHipster Generator v2.17.0

? (1/15) What is the base name of your application? jhipster
? (2/15) What is your default Java package name? com.mycompany.myapp
? (3/15) Do you want to use Java 8? Yes (use Java 8)
? (4/15) Which *type* of authentication would you like to use? (Use arr
❯ HTTP Session Authentication (stateful, default Spring Security mechan
  OAuth2 Authentication (stateless, with an OAuth2 server implementatio
  Token-based authentication (stateless, with a token)
```

Branch: master

generator-jhipster / app / templates / src / main / java / package / config / _DatabaseConfiguration.java

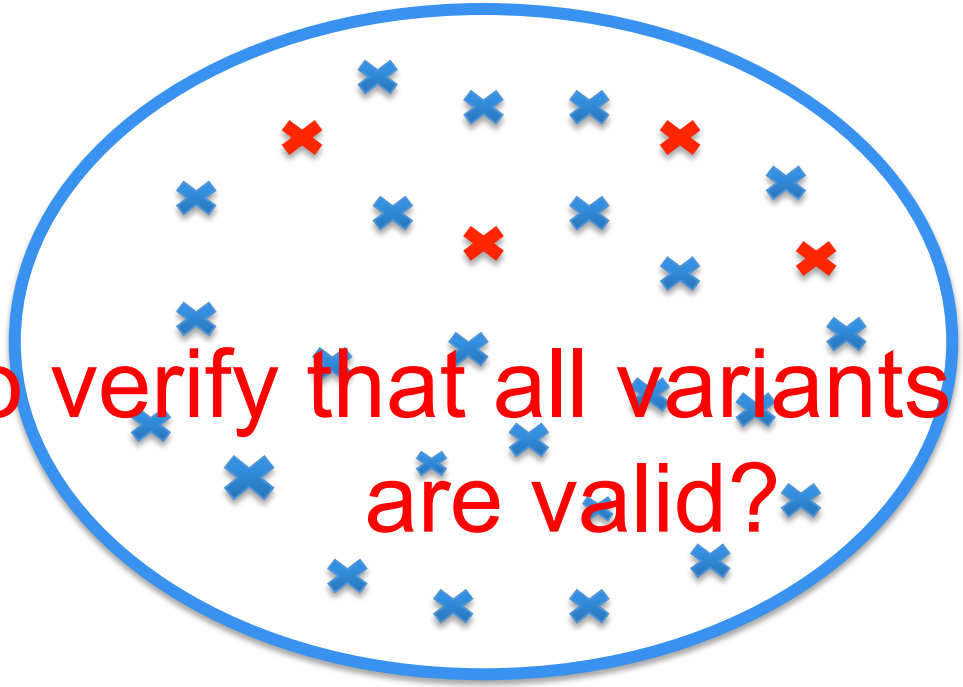
jdubois 2 days ago Use Spring Boot's configuration meta-data

9 contributors

184 lines (165 sloc) | 9.69 KB

Raw Blame History

```
package <%=packageName%>.config;
<% if (databaseType == 'sql') { %>
import <%=packageName%>.config.liquibase.AsyncSpringLiquibase;
import com.codahale.metrics.MetricRegistry;
import com.fasterxml.jackson.datatype.hibernate4.Hibernate4Module;
import com.zaxxer.hikari.HikariConfig;
import com.zaxxer.hikari.HikariDataSource;
import liquibase.integration.spring.SpringLiquibase;<% } %><% if (databaseType == 'mongodb' && authenticationType == 'oauth2') { %
import <%=packageName%>.config.oauth2.OAuth2AuthenticationReadConverter;<% } %><% if (databaseType == 'mongodb') { %>
import com.mongodb.Mongo;
import org.mongeez.Mongeez;<% } %>
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;<% if (databaseType == 'sql') { %><% if (hibernateCache == 'hazelcast') { %>
import org.springframework.cache.CacheManager;<% } %>
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.autoconfigure.condition.ConditionalOnExpression;<% } %><% if (databaseType == 'mongodb') { %>
import org.springframework.boot.autoconfigure.mongo.MongoAutoConfiguration;
import org.springframework.boot.autoconfigure.mongo.MongoProperties;<% } %><% if (databaseType == 'sql') { %>
import org.springframework.boot.autoconfigure.jdbc.DataSourceProperties;
import org.springframework.boot.autoconfigure.liquibase.LiquibaseProperties;
import org.springframework.context.ApplicationContextException;<% } %>
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.Profile;<% if (databaseType == 'mongodb') { %>
import org.springframework.context.annotation.Import;<% } %><% if (databaseType == 'sql') { %>
import org.springframework.core.env.Environment;<% } %><% if (databaseType == 'mongodb' && authenticationType == 'oauth2') { %>
import org.springframework.core.convert.converter.Converter;<% } %><% if (databaseType == 'mongodb') { %>
import org.springframework.core.io.ClassPathResource;<% } %><% if (searchEngine == 'elasticsearch') { %>
import org.springframework.data.elasticsearch.repository.config.EnableElasticsearchRepositories;<% } %><% if (databaseType == 'mon
import org.springframework.data.mongodb.config.AbstractMongoConfiguration;
import org.springframework.data.mongodb.config.EnableMongoAuditing;<% } %><% if (databaseType == 'mongodb' && authenticationType =
import org.springframework.data.mongodb.core.convert.CustomConversions;<% } %><% if (databaseType == 'mongodb') { %>
import org.springframework.data.mongodb.core.mapping.event.ValidatingMongoEventListener;
import org.springframework.data.mongodb.repository.config.EnableMongoRepositories;
import org.springframework.validation.beanvalidation.LocalValidatorFactoryBean;<% } %><% if (databaseType == 'sql') { %>
```



Jhipster variant

How to verify that all variants of Jhipster are valid?

Challenge #1: Modeling variability

- Why?
  - Manual test: very limited!
  - Need an abstraction for automating the sampling of configurations and then testing the corresponding variants
- Manual elaboration of a feature model?
- Reverse engineering?

GitHub, Inc. [US] https://github.com/jhipster/generator-jhipster/blob/master/app/index.js

```
type: 'list',
name: 'authenticationType',
message: '(3/' + questions + ') Which *type* of authentication would you like to use?',
choices: [
    {
        value: 'session',
        name: 'HTTP Session Authentication (stateful, default Spring Security mechanism)'
    },
    {
        value: 'session-social',
        name: 'HTTP Session Authentication with social login enabled (Google, Facebook, Twitter). Warning, this does
    },
    {
        value: 'oauth2',
        name: 'OAuth2 Authentication (stateless, with an OAuth2 server implementation)'
    },
    {
        value: 'xauth',
        name: 'Token-based authentication (stateless, with a token)'
    }
],
default: 0
```

Challenge #2: sampling configurations

- All configurations?
- Cost-effective
- Minimization
- Priority (based on some knowledge)

Configuration Sampling

- Number of configurations dramatically grows with the number of parameters
- Select a subset of combinations

- Key idea: focus on specific interactions among values for parameters
  - Pair-wise interactions
  - T-wise interactions

Configuration Sampling

- $X_1,\ldots, X_n$ n parameters
- $\forall\ i \in [1..n]\ X_i \subset \{V_{i1},\ldots,V_{im}\}$
  - m can be different for every Xi
- A configuration is a set of values for each X_i
- Pairwise covering
  - A set TC of configurations such that all values for every pair of parameters are in one configuration
  - $\forall\ X_j, X_k \mid \forall\ X_{ja}, X_{kb} \mid \exists\ c \in TC \mid TC \subset X_{ja}, X_{kb}$
- T-wise covering
  - Generalization to all tuples of parameters

Illustrative Example

Characteristics	Values			
Background	CountryS	Desert	Urban	Forest
Luminosity	Normal	High	Low	
Speed	0.1	0.2	0.3	
Detractors Level	0.2	0.4	0.6	

There are **108** combinations of factors

12 combinations can satisfy pair-wise

Example

CountryS	normal	0.1	0.2
CountryS	High	0.2	0.4
CountryS	Low	0.3	0.6
Urban	normal	0.3	0.4
Urban	High	0.1	0.2
Urban	Low	0.2	0.6
Desert	normal	0.2	0.6
Desert	High	0.3	0.2
Desert	Low	0.1	0.4
Forest	normal	*	0.4
Forest	High	0.1	0.6
Forest	Low	0.2	0.2

Example

CountryS	normal	0.1	0.2
CountryS	High	0.2	0.4
CountryS	Low	0.3	0.6
Urban	normal	0.3	0.4
Urban	High	0.1	0.2
Urban	Low	0.2	0.6
Desert	normal	0.2	0.6
Desert	High	0.3	0.2
Desert	Low	0.1	0.4
Forest	normal	*	0.4
Forest	High	0.1	0.6
Forest	Low	0.2	0.2

Example

CountryS	normal	0.1	0.2
CountryS	High	0.2	0.4
CountryS	Low	0.3	0.6
Urban	normal	0.3	0.4
Urban	High	0.1	0.2
Urban	Low	0.2	0.6
Desert	normal	0.2	0.6
Desert	High	0.3	0.2
Desert	Low	0.1	0.4
Forest	normal	*	0.4
Forest	High	0.1	0.6
Forest	Low	0.2	0.2

Example

CountryS	normal	0.1	0.2
CountryS	High	0.2	0.4
CountryS	Low	0.3	0.6
Urban	normal	0.3	0.4
Urban	High	0.1	0.2
Urban	Low	0.2	0.6
Desert	normal	0.2	0.6
Desert	High	0.3	0.2
Desert	Low	0.1	0.4
Forest	normal	*	0.4
Forest	High	0.1	0.6
Forest	Low	0.2	0.2

Example

CountryS	normal	0.1	0.2
CountryS	High	0.2	0.4
CountryS	Low	0.3	0.6
Urban	normal	0.3	0.4
Urban	High	0.1	0.2
Urban	Low	0.2	0.6
Desert	normal	0.2	0.6
Desert	High	0.3	0.2
Desert	Low	0.1	0.4
Forest	normal	*	0.4
Forest	High	0.1	0.6
Forest	Low	0.2	0.2

Example

CountryS	normal	0.1	0.2
CountryS	High	0.2	0.4
CountryS	Low	0.3	0.6
Urban	normal	0.3	0.4
Urban	High	0.1	0.2
Urban	Low	0.2	0.6
Desert	normal	0.2	0.6
Desert	High	0.3	0.2
Desert	Low	0.1	0.4
Forest	normal	*	0.4
Forest	High	0.1	0.6
Forest	Low	0.2	0.2

Example

CountryS	normal	0.1	0.2
CountryS	High	0.2	0.4
CountryS	Low	0.3	0.6
Urban	normal	0.3	0.4
Urban	High	0.1	0.2
Urban	Low	0.2	0.6
Desert	normal	0.2	0.6
Desert	High	0.3	0.2
Desert	Low	0.1	0.4
Forest	normal	*	0.4
Forest	High	0.1	0.6
Forest	Low	0.2	0.2

Challenge #3: “validity”

- It compiles
- It can be executed/deployed
- It passes the test suites
- It passes the benchmarks
- …

```
macher-wifi:getting-started macher1$ yo jhipster

I'm all done. Running npm install & bower install for you to install the required dependencies

 _     _   ___   __  _____  ____  ___      __  _____  __    __   _
| |_| | | | |_) ( (`  | |  | |_  | |_)    ( (`  | |  / /\  / /`  | |_
|_| | |_| |_|   _)_)  |_|  |_|__ |_| \    _)_)  |_|  /_/--\ \_\_, |_|
                                 ____  ___   ___
                                | |_  / / \ | |_)
                                |_|   \_\_/ |_| \
 _    __    _        __        ___   ____  _    __
| |  / /\  \ \  /  / /\      | | \ | |_  \ \  / ( (`
\_|_| /_/--\  \_\/  /_/--\    |_|_/ |_|__  \_\/  _)_)

Welcome to the JHipster Generator v2.17.0

? (1/15) What is the base name of your application? jhipster
? (2/15) What is your default Java package name? com.mycompany.myapp
? (3/15) Do you want to use Java 8? Yes (use Java 8)
? (4/15) Which *type* of authentication would you like to use? (Use arr
❯ HTTP Session Authentication (stateful, default Spring Security mechan
  OAuth2 Authentication (stateless, with an OAuth2 server implementatio
  Token-based authentication (stateless, with a token)
```

Branch: master

generator-jhipster / app / templates / src / main / java / package / config / _DatabaseConfiguration.java

jdubois 2 days ago Use Spring Boot's configuration meta-data

9 contributors

184 lines (165 sloc) | 9.69 KB

Raw Blame History

```
package <%=packageName%>.config;
<% if (databaseType == 'sql') { %>
import <%=packageName%>.config.liquibase.AsyncSpringLiquibase;
import com.codahale.metrics.MetricRegistry;
import com.fasterxml.jackson.datatype.hibernate4.Hibernate4Module;
import com.zaxxer.hikari.HikariConfig;
import com.zaxxer.hikari.HikariDataSource;
import liquibase.integration.spring.SpringLiquibase;<% } %><% if (databaseType == 'mongodb' && authenticationType == 'oauth2') { %
import <%=packageName%>.config.oauth2.OAuth2AuthenticationReadConverter;<% } %><% if (databaseType == 'mongodb') { %>
import com.mongodb.Mongo;
import org.mongeez.Mongeez;<% } %>
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;<% if (databaseType == 'sql') { %><% if (hibernateCache == 'hazelcast') { %>
import org.springframework.cache.CacheManager;<% } %>
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.autoconfigure.condition.ConditionalOnExpression;<% } %><% if (databaseType == 'mongodb') { %>
import org.springframework.boot.autoconfigure.mongo.MongoAutoConfiguration;
import org.springframework.boot.autoconfigure.mongo.MongoProperties;<% } %><% if (databaseType == 'sql') { %>
import org.springframework.boot.autoconfigure.jdbc.DataSourceProperties;
import org.springframework.boot.autoconfigure.liquibase.LiquibaseProperties;
import org.springframework.context.ApplicationContextException;<% } %>
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.Profile;<% if (databaseType == 'mongodb') { %>
import org.springframework.context.annotation.Import;<% } %><% if (databaseType == 'sql') { %>
import org.springframework.core.env.Environment;<% } %><% if (databaseType == 'mongodb' && authenticationType == 'oauth2') { %>
import org.springframework.core.convert.converter.Converter;<% } %><% if (databaseType == 'mongodb') { %>
import org.springframework.core.io.ClassPathResource;<% } %><% if (searchEngine == 'elasticsearch') { %>
import org.springframework.data.elasticsearch.repository.config.EnableElasticsearchRepositories;<% } %><% if (databaseType == 'mon
import org.springframework.data.mongodb.config.AbstractMongoConfiguration;
import org.springframework.data.mongodb.config.EnableMongoAuditing;<% } %><% if (databaseType == 'mongodb' && authenticationType =
import org.springframework.data.mongodb.core.convert.CustomConversions;<% } %><% if (databaseType == 'mongodb') { %>
import org.springframework.data.mongodb.core.mapping.event.ValidatingMongoEventListener;
import org.springframework.data.mongodb.repository.config.EnableMongoRepositories;
import org.springframework.validation.beanvalidation.LocalValidatorFactoryBean;<% } %><% if (databaseType == 'sql') { %>
```

Jhipster variant

For each release,
how to verify that all variants of Jhipster
are valid?

```
macher-wifi:getting-started macher1$ yo jhipster

I'm all done. Running npm install & bower install for you to install the required dependencies

 _     _   ___   __  _____  ____  ___      __  _____  __    __   _
| |_| | | | |_) ( (`  | |  | |_  | |_)    ( (`  | |  / /\  / /`  | |_
|_| | |_| |_|   _)_)  |_|  |_|__ |_| \    _)_)  |_|  /_/--\ \_\_, |_|
                                 ____  ___  ___
                                | |_  / / \ | |_)
                                |_|   \_\_/ |_| \
 _    __    _      __      ___   ____  _     __
| |  / /\  \ \  /  / /\    | | \ | |_  \ \  / ( (`
\_|_| /_/--\  \_\/  /_/--\  |_|_/ |_|__  \_\/  _)_)

Welcome to the JHipster Generator v2.17.0

? (1/15) What is the base name of your application? jhipster
? (2/15) What is your default Java package name? com.mycompany.myapp
? (3/15) Do you want to use Java 8? Yes (use Java 8)
? (4/15) Which *type* of authentication would you like to use? (Use arr
❯ HTTP Session Authentication (stateful, default Spring Security mechan
  OAuth2 Authentication (stateless, with an OAuth2 server implementatio
  Token-based authentication (stateless, with a token)
```

Branch: master

generator-jhipster / app / templates / src / main / java / package / config / _DatabaseConfiguration.java

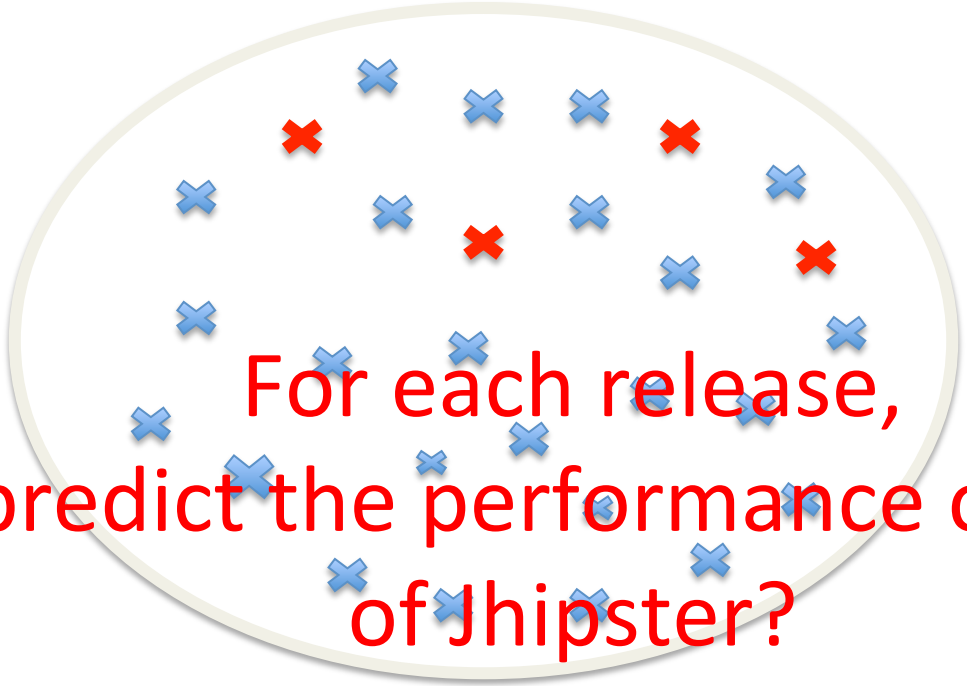
jdubois 2 days ago Use Spring Boot's configuration meta-data

9 contributors

184 lines (165 sloc) | 9.69 KB

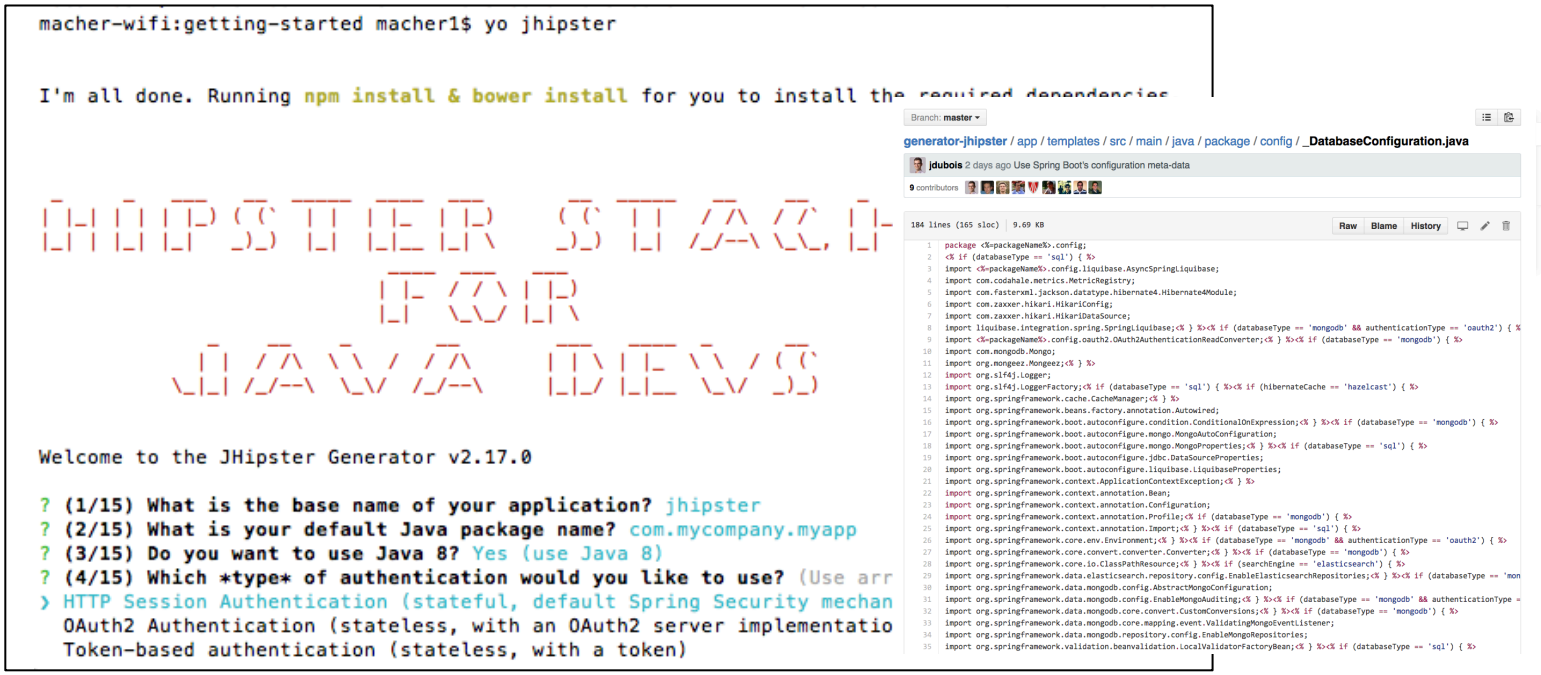
Raw Blame History

```
package <%=packageName%>.config;
<% if (databaseType == 'sql') { %>
import <%=packageName%>.config.liquibase.AsyncSpringLiquibase;
import com.codahale.metrics.MetricRegistry;
import com.fasterxml.jackson.datatype.hibernate4.Hibernate4Module;
import com.zaxxer.hikari.HikariConfig;
import com.zaxxer.hikari.HikariDataSource;
import liquibase.integration.spring.SpringLiquibase;<% } %><% if (databaseType == 'mongodb' && authenticationType == 'oauth2') { %
import <%=packageName%>.config.oauth2.OAuth2AuthenticationReadConverter;<% } %><% if (databaseType == 'mongodb') { %>
import com.mongodb.Mongo;
import org.mongeez.Mongeez;<% } %>
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;<% if (databaseType == 'sql') { %><% if (hibernateCache == 'hazelcast') { %>
import org.springframework.cache.CacheManager;<% } %>
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.autoconfigure.condition.ConditionalOnExpression;<% } %><% if (databaseType == 'mongodb') { %>
import org.springframework.boot.autoconfigure.mongo.MongoAutoConfiguration;
import org.springframework.boot.autoconfigure.mongo.MongoProperties;<% } %><% if (databaseType == 'sql') { %>
import org.springframework.boot.autoconfigure.jdbc.DataSourceProperties;
import org.springframework.boot.autoconfigure.liquibase.LiquibaseProperties;
import org.springframework.context.ApplicationContextException;<% } %>
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.Profile;<% if (databaseType == 'mongodb') { %>
import org.springframework.context.annotation.Import;<% } %><% if (databaseType == 'sql') { %>
import org.springframework.core.env.Environment;<% } %><% if (databaseType == 'mongodb' && authenticationType == 'oauth2') { %>
import org.springframework.core.convert.converter.Converter;<% } %><% if (databaseType == 'mongodb') { %>
import org.springframework.core.io.ClassPathResource;<% } %><% if (searchEngine == 'elasticsearch') { %>
import org.springframework.data.elasticsearch.repository.config.EnableElasticsearchRepositories;<% } %><% if (databaseType == 'mon
import org.springframework.data.mongodb.config.AbstractMongoConfiguration;
import org.springframework.data.mongodb.config.EnableMongoAuditing;<% } %><% if (databaseType == 'mongodb' && authenticationType =
import org.springframework.data.mongodb.core.convert.CustomConversions;<% } %><% if (databaseType == 'mongodb') { %>
import org.springframework.data.mongodb.core.mapping.event.ValidatingMongoEventListener;
import org.springframework.data.mongodb.repository.config.EnableMongoRepositories;
import org.springframework.validation.beanvalidation.LocalValidatorFactoryBean;<% } %><% if (databaseType == 'sql') { %>
```



Jhipster variant

For each release,
how to predict the performance of all variants of Jhipster?



Variability modeling for

- improving quality-assurance of configurable systems
- re-engineering configurators
- predicting performance
- prioritizing the development of features