Applying TDD for Creating DSM solutions: demo

Juha-Pekka Tolvanen
30 October, 2016
DSM Workshop, Amsterdam
Content

- Motivation
- Idea on applying TDD for DSM
- Demonstration
- Discussion
Motivation

- Better ensure that DSM solution (= language and generator) works as expected
  - When created
  - When modified while the domain evolves

- Our special focus is on generators as their development takes more time than languages

- Various approaches have been presented to test languages and generators
  - We present one considered cost-effective
Idea

Apply TDD for DSM creation and maintenance as follows:

1. Create a library of small test models along with related output (in a format preferred)

2. Build the DSM solution in small steps starting from the simplest test model of the library and then extend to others

   \[
   \text{Step} = \text{language element(s)} + \text{its test model} + \text{its generator} + \text{expected output}
   \]

3. Test DSM solution after each step of adding/removing/changing metamodel or generator

4. Run all previous tests for regression testing

5. Automate the process
Domain for demo: Apps for IoT device

- **Sensors**
  - altitude, movement (in 3 directions), humidity, location, luminance, pressure, speed, temperature, time, geofence, battery status...

- **Actions**
  - sending SMS, send to cloud, push notification, saving logs
Sample: Sauna App
Sample 2: Boat monitor
Some test models to start with

- 4 out of 37 in total
Test model and expected code

- Model with 3 elements and 1 connection => 50+ LOC
- Automated test checking that generated = expected
- Test individual tests or all tests from the library
Test library at the end
Demo
Experiences on applying TDD for DSM

- Makes DSL development "systematic"/phased
  - Supports planning and estimating on the DSM development effort
- Provides regression testing giving conformance that nothing breaks when language & generator changed
  - Is not the only testing approach as also existing models (and generated code) can be applied as test data
- Can be extended to cover results generated with the platform/components/etc.
  - i.e. integrate resulting code with the rest and test together
- Considered applicable in practice (by MetaEdit+ users)
Remarks

- This approach is been applied in industrial projects when the customer has known:
  1. what needs to be generated early on
  2. enough language concepts to create test models

- Extensions applied include running the tests from other tools, including the framework code (that is not generated from models) with the tests

- Is not applied to cover all parts of the DSM solution (e.g. translators from model entry to output entry, naming rules with regexp)
Thank you!

Questions?

Contacts: Juha-Pekka Tolvanen
jpt@metacase.com
www.metacase.com