An Archetype-based Testing Framework

Rong Chen\textsuperscript{a,b}, Sebastian Garde\textsuperscript{c}, Thomas Beale\textsuperscript{c},

Mikael Nyström\textsuperscript{a}, Daniel Karlsson\textsuperscript{a}, Gunnar Klein\textsuperscript{d}, Hans Åhlfeldt\textsuperscript{a}

\textsuperscript{a} Department of Biomedical Engineering, Linköping University, Sweden
\textsuperscript{b} Cambio Healthcare Systems, Sweden
\textsuperscript{c} Ocean Informatics UK, London
\textsuperscript{d} Karolinska Institutet, Sweden
Where do we stand?

• Electronic Health Records (EHR)

• openEHR, EN/ISO 13606
  – 15 years EU & Australian R&D on EHR

• Archetypes & Semantic Interoperability

• Interoperable & adaptive EHR systems

*But how to validate that?*
The openEHR artefact ecosystem

Data capture

Data validation

Reference Model

Archetypes

Code Skeletons

Templates

Queries

UI Forms

Data Sets

XML schemas

Messages

Display HTML

Terminology Mappings

Terminology

Terminology subsets
Validation Objectives

- Not to find syntactic errors – the parsing tools find those
- To validate archetypes against the clinical requirements semantic validation

- For a single archetype
  - Use ‘fixtures’ to test whether the archetype behaves the way it was intended

- For a repository of archetypes
  - Search for refactoring and other design improvement opportunities, e.g. repetition, overlap, ‘bad design’
The Classic Way in Software

• Unit testing
  – it works, sort of…

• But
  – implementation & platform specific
    Implementation and platform independent
  – testcases maintained in source code
    Portable test case specifications
  – tedious manual effort
    Preferrably auto-generated tests
The technical Requirements

• Testing should be platform-independent and implementation-independent

• Tests are archetype-based and preferably auto-generated
  – Test fixtures as input and expected output
  – Kernel operations & Archetypes

• **Scope:** [ kernel ] + archetypes
Any other platform

.Net

Kernel Implementation A

Java

Kernel Implementation 1

Kernel Implementation 2

Kernel Implementation 3

Test runner

derived from

Test Cases

derived from

Test Fixtures

derived from

Archetypes

Platform-independent

Design Diagram
Test Fixtures

- RM instances or primitives
- Valid or invalid according to archetypes
- dADL format
- Auto-Generation strategies
  - based on constraints in the archetypes
  - e.g. validation ranges, node existence

May 26, 2008  MIE 2008  in Göteborg, Sweden
Test Case Specifications

1. Kernel operation for the test case
2. Common test fixtures
3. List of tests
   • unique name
   • input values
   • expected output

<queryTestSpecification>
  <instance>blood_pressure_002.dadl</instance>
  <tests>
    <test name="Blood Pressure - Systolic Value Query - native language">
      <path>/data/events['baseline reading']/data/items['systolic']/value/magnitude</path>
      <expected>120.0</expected>
    </test>
    <test name="Blood Pressure - Diastolic Value Query - native language">
      <path>/data/events['baseline reading']/data/items['diastolic']/value/magnitude</path>
      <expected>80.0</expected>
    </test>
  </tests>
</queryTestSpecification>
<queryTestSpecification>
  <instance>blood_pressure_002.dadl</instance>
  <tests>
    <test name="Blood Pressure - Systolic Value Query - native language">
      <path>/data/events["baseline reading"]
        /data/items["systolic"]/value/magnitude</path>
      <expected>120.0</expected>
    </test>
    <test name="Blood Pressure - Diastolic Value Query - native language">
      <path>/data/events["baseline reading"]
        /data/items["diastolic"]/value/magnitude</path>
      <expected>80.0</expected>
    </test>
  </tests>
</queryTestSpecification>
Proof of Concept

- Extend JUnit testing framework
  - For organizing testcases, fixtures and results
  - Test result assertions
  - Test runners for test execution

- openEHR Java components
  - the kernel, the parsers

- Prototype: archetype-testing
  - ArchetypeTestCase
  - ArchetypeQueryTestCase
  - XML Schema for query test specification
Archetype Testing in Action

JUnit testing framework showing successful test results for various archetype queries.
A Failed Query Test

Expected “Standing” but was “Sitting”
Summary

Testing framework for validation of archetype formalism implementations

Future

• EHR Client API + more operations
• Auto-generation of test fixtures / test cases
• Integration with archetype tools / repository

EHR Semantic Interoperability!!
Thanks!

Acknowledgements

The openEHR Foundation
Linköping University
Karolinska Institute
Ocean Informatics
Cambio Healthcare Systems

Contact Info

Rong Chen,
E-mail: rong.chen@imt.liu.se

Development

http://www.openehr.org/svn/ref_impl_java/SANDBOX/archetype-testing