System testing scientific software
Torkild Ulvøy Resheim

- Senior Software Developer & co-owner at Itema.
  - Java developer since 1995.
  - Eclipse developer since 2003.
  - Committer at Eclipse Mylyn and Orbit projects.
- Contractor on the SIMA project at MARINTEK
  - Marine operation simulations workbench

@torkildr
mailto:tur@itema.no
http://www.itema.no
AND THEN HE SAID...

WE HAVE NO SYSTEM TESTS!
The Team

- Three research scientists as product owners.
- Six software engineers developing the Eclipse based simulation workbench.
- Two research scientists developing the physics software.
Typical user story

Implement import of diffracted wave kinematics from WAMIT

- Implement import of diffracted wave from WAMIT.
- Diffracted wave points should be available (by copy paste or import) in Reflex Support W.
- See attached example of WAMIT input and current implementation in SIMA.
- Remove underscore in Diffracted_wave point if.

Some more details:
- The coordinates of the field points should be read from the *.fpt-file.
- The wave elevation is read from *.6p.
- The three velocity components are read from *.6vx, *.6vy and *.6vz.

Wave elevation and the velocities are given as non-dimensional numbers and must be dimensionless. For wave elevation this is trivial multiplication with a factor 1. For velocities specified in an attachment.

Attachments
- Diffracted_wave_ex.tar.bz2
- diffractedwave.task
- diffraktet_bage_til_Riflex_eksempel_Fraydis.pdf
- normalization_fluidvelocities.pdf
How to test a case like this?

- **Unit tests** on the calculations.
- **Integration tests** on the components involved.
  - Hydrodynamics + WAMIT (Wave Interaction Analysis) support.
- **End to end system tests.**
How we usually write system tests

• Done solely by the software developers.
• Time consuming.
• Often requires a lot of rigging.
  • Simulation models are programmed, not designed using the workbench.
  • Sensitive to changes in the data model.
“I believe that the hardest part of software projects, the most common source of project failure, is communication with the customers and users of that software. By providing a clear yet precise language to deal with domains, a DSL can help improve this communication.”

- Martin Fowler, Domain Specific Languages
The system test DSL

• Enable our scientists to model a case and write the tests for it.

• Enable customers to verify that their cases produce the same results after a software upgrade.

• Automatically execute system tests during builds.

Illustration © Allie Brosh of Hyperbole and a Half
What’s in so far

• Created a very simple DSL using Xtext.
  • Asserts
  • Basic simulation control
  • Embed wiki markup.
  • Script command output is HTML.
• Embedded the DSL into the product being tested.
DSL definition and output

- DSL grammar
- DSL generator configuration
- Output generator
  - Does the actual testing
  - Generates the report file
DSL grammar
IDE Tooling

- Content assist provider
Why Xtext?

• Proven Eclipse technology.
• Produces great results, fast.
• Generates editor and framework for content assist, validation and output.
Alternatives to Xtext

- Eclipse Advanced Scripting Environment (EASE)
  - Python, JavaScript
- ANTLR
  - Roll your own everything.
  - Runs independent of Eclipse
Conclusion

• Xtext is powerful and easy to get started with.
  • Sadly no lexer multiplexing
• Our DSL is already useful!
• We will take this further:
  • Generation of JUnit test report
  • Syntax for defining and testing post processor.
  • Output to PDF for inclusion in the product build.
Thank you!

Evaluate the sessions at www.eclipsecon.org