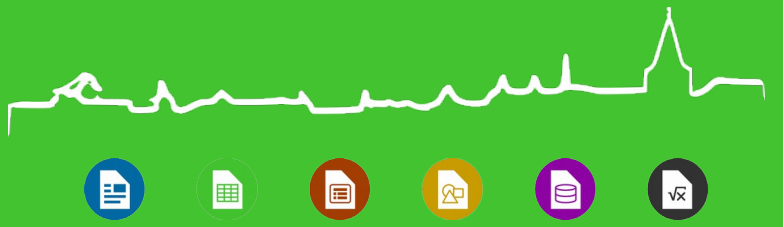
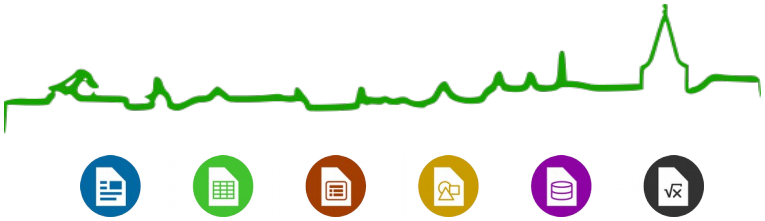
Aarhus 2015 CONFERENCE



State of Automated Testing

- Michael Stahl, Red Hat, Inc.
- 2015-09-23



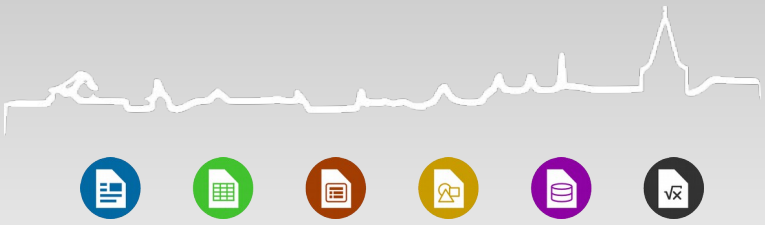
Overview

1. Introduction

2. Regression Tests Developers Run Every Day

3. Even More Tests

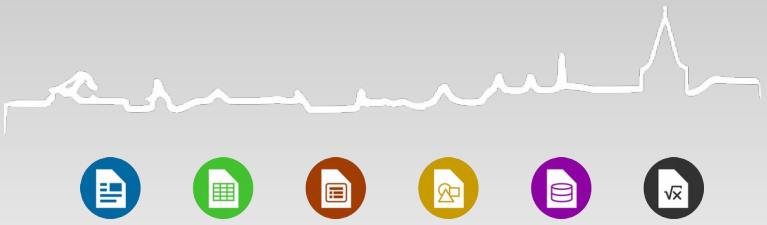
Introduction



- Every release: ~10.000 commits, ~1 million LOC changed
- Regressions?
- Automated testing
- Goal: developers find bugs before they push them to master

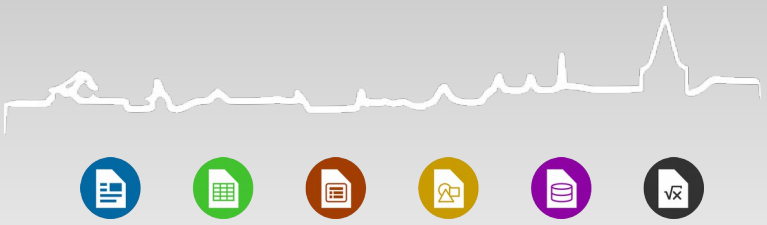
“Your developers—or worst case, test organization—produce tests.”
— Michael Klepikov

Requirements



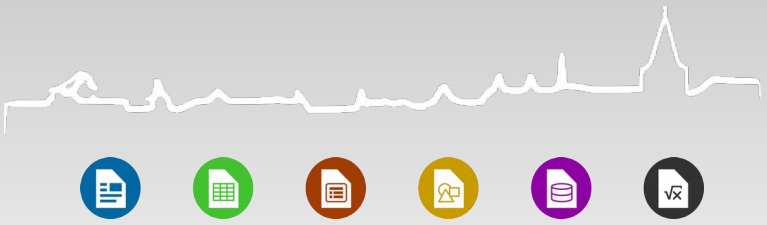
- Goal: developers find bugs before they push them to master
- Requirements
  - Standard unit test libraries
  - Must be run as part of standard build – `make check`
  - Reliable – avoid false positives
  - Fast
  - Good defect localization
  - Debuggable

CppUnit Tests



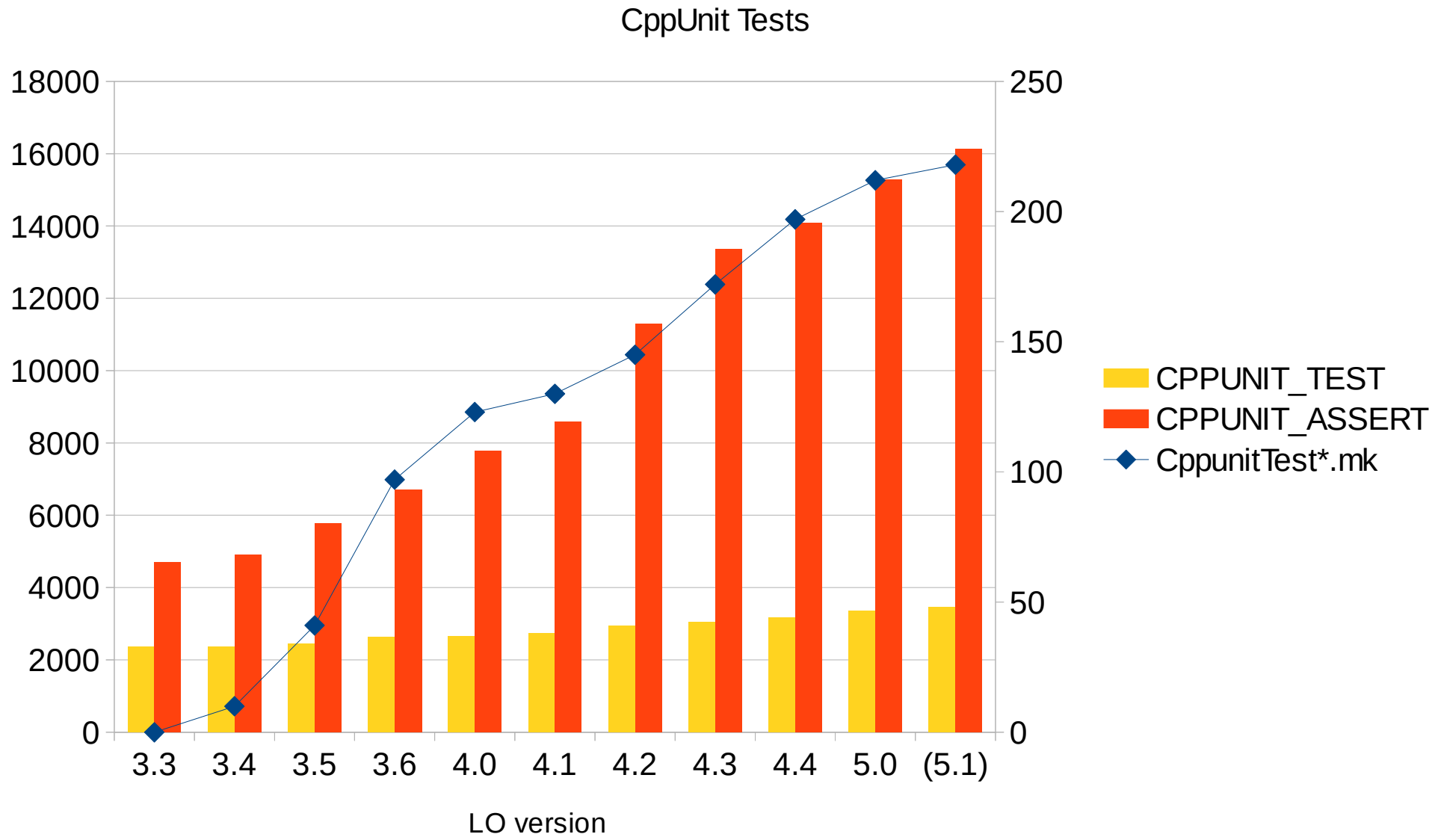
- C++
- Standard CppUnit library (Thanks to Markus Mohrhard)
- In-process (except `smoketest`)
- Kinds:
  - Unit tests
    - C++ classes
    - Components via UNO API
  - Integration tests
    - Filter crash-tests (CVE-tests)
    - Filter tests
    - Etc.
  - System tests: smoketest

Filter Tests

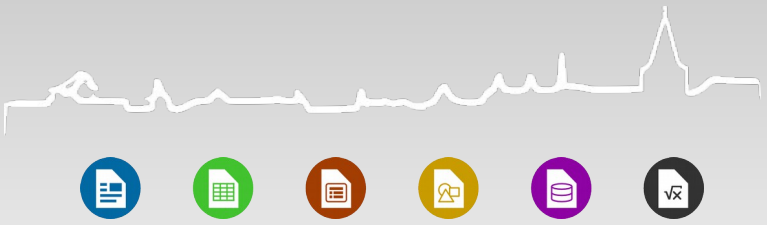


- Filter test:
  - Import file
  - Check some properties were imported correctly
  - Export file, import again
  - Check properties were round-tripped correctly
- `--with-export-validation` validates every exported file
  - ODF Validator
  - OOXML Validator

CppUnit Test Growth

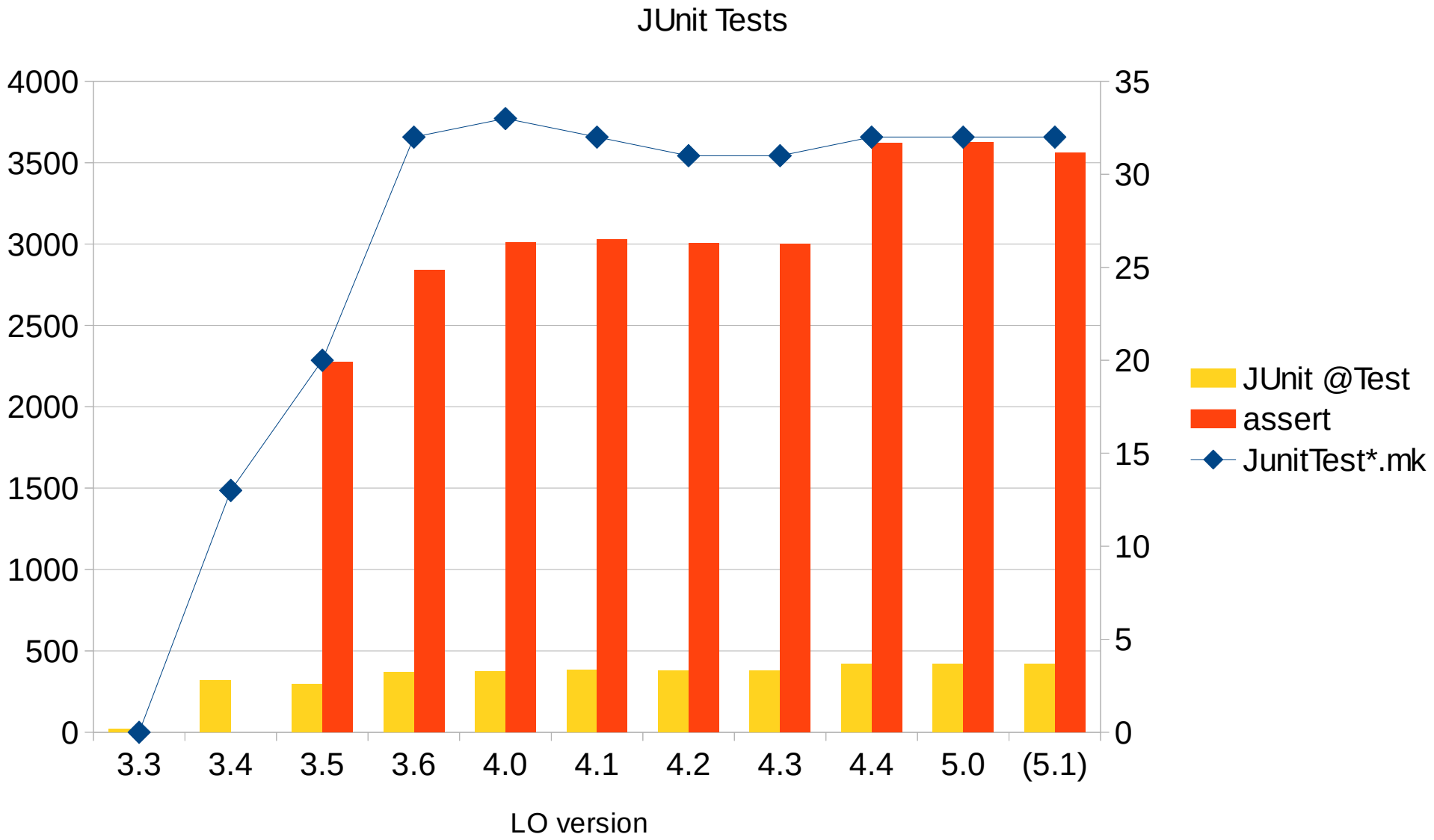


JUnit Tests

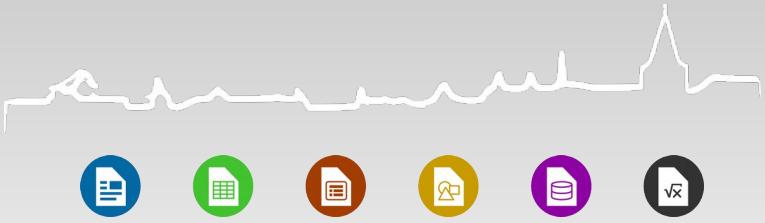


- Java
- Standard JUnit library
- Kinds:
  - Unit tests – in-process
    - URE Java binding
  - “Complex” tests – out-of-process – Remote UNO
    - Unit tests of components
    - Integration tests

JUnit Test Growth

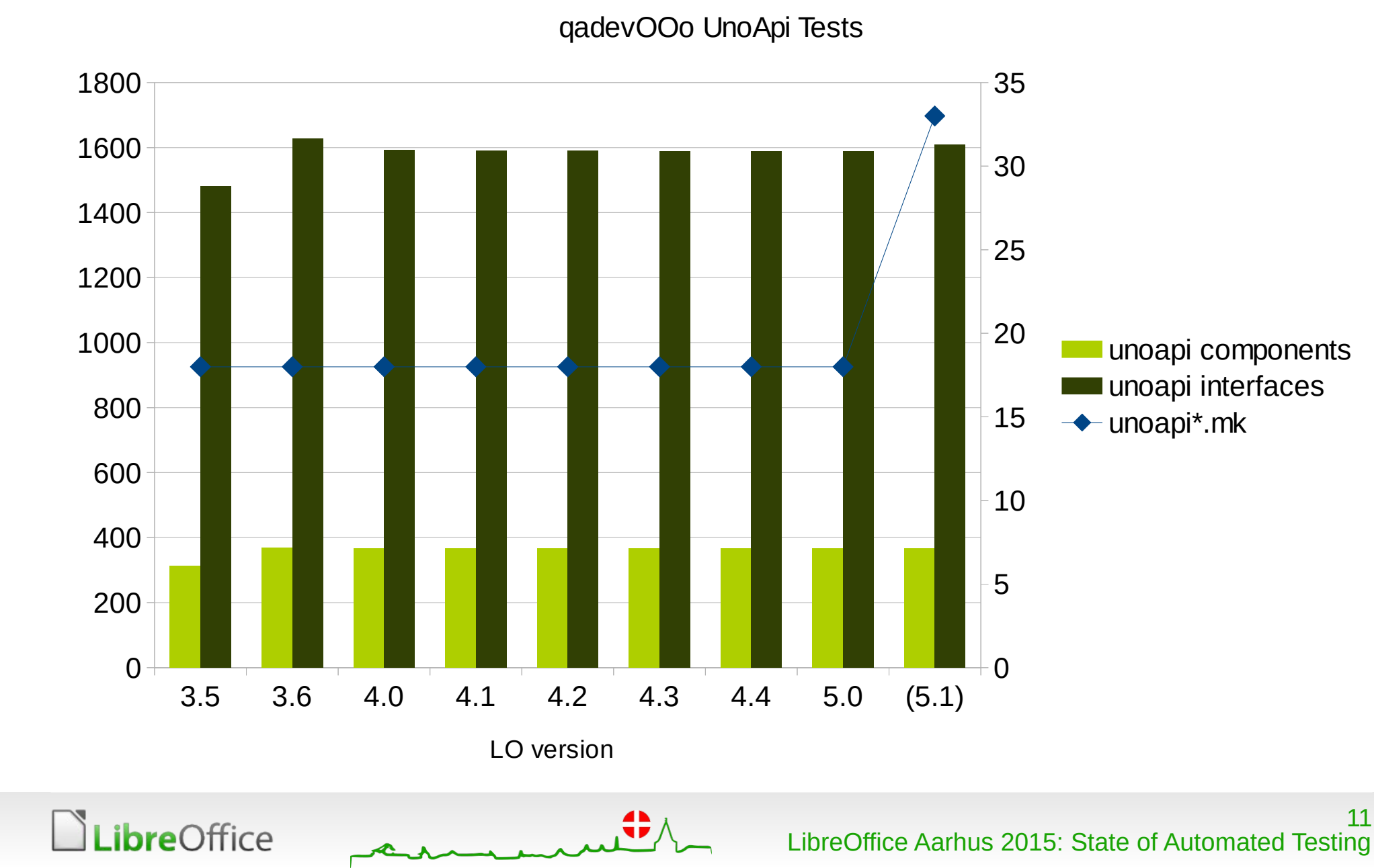


qadevOOo “unoapi” Tests

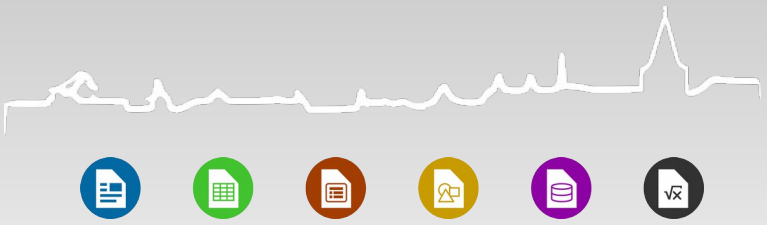


- Java
- Custom test framework
- Out-of-process – Remote UNO
- Obscure test code
- “black-box”, overly generic tests
- Kinds
  - Unit tests – components via UNO API

qadevOOo Test Non-Growth

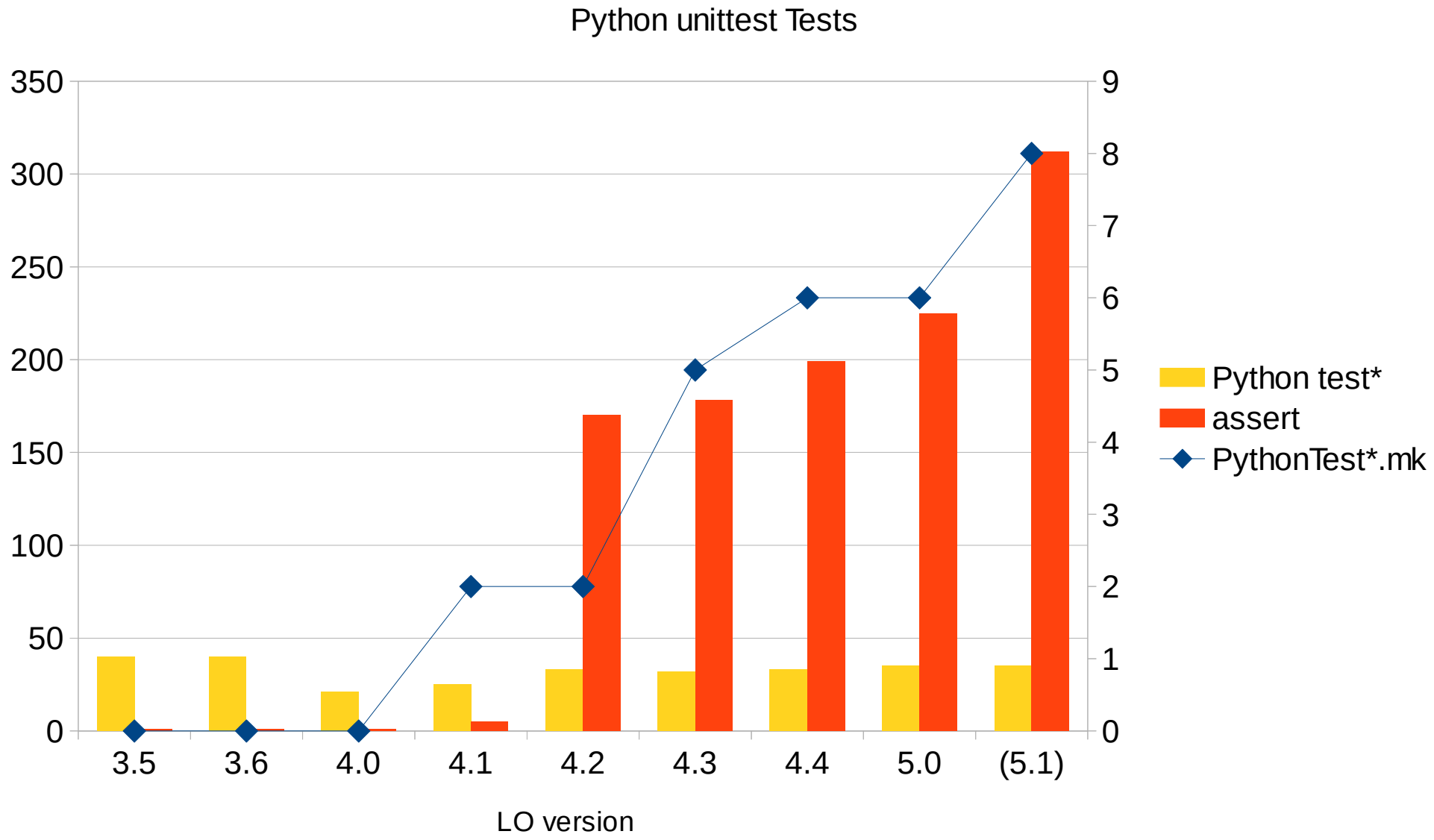


Python Tests

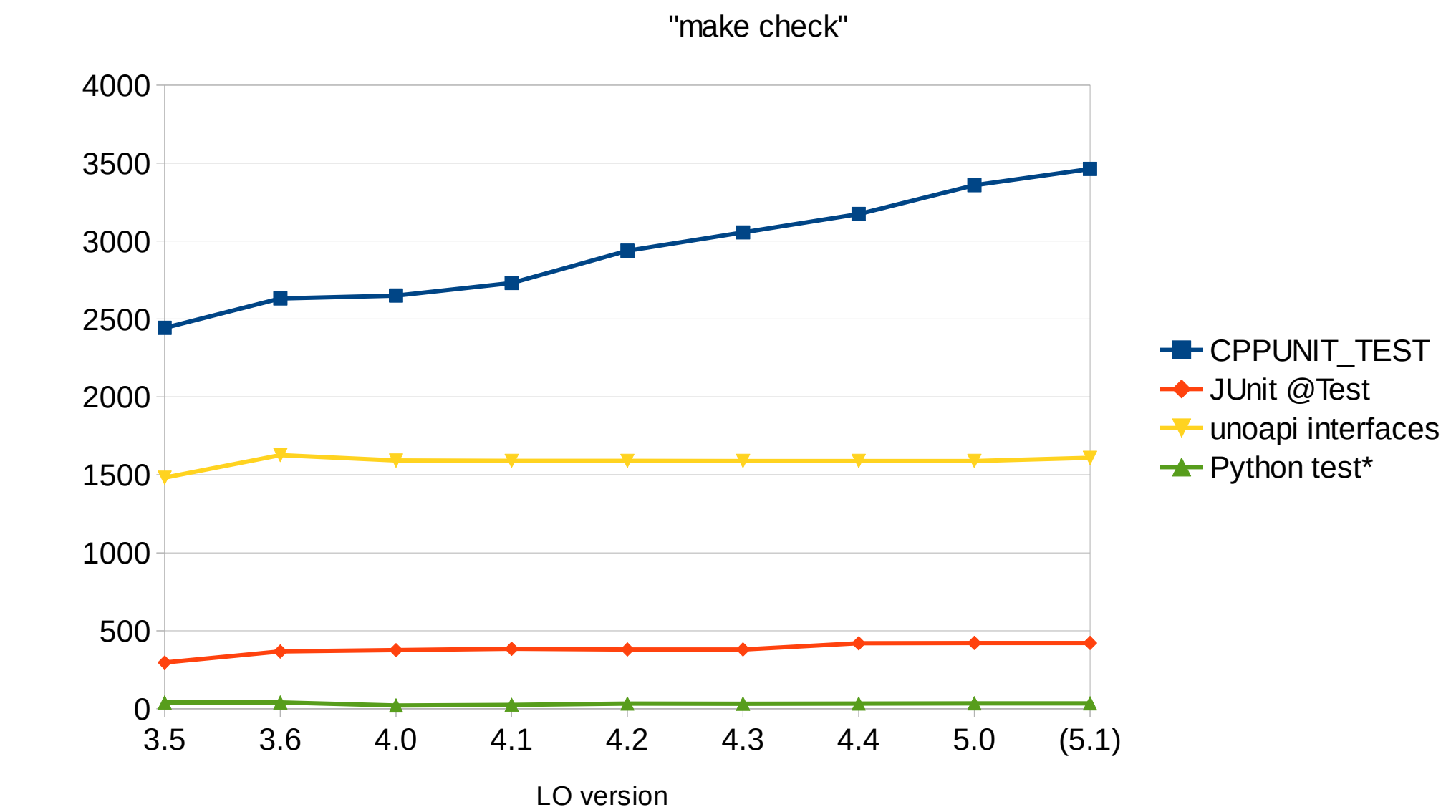


- Python
- Standard unittest library
- In-process
- Thanks to David Ostrovsky
- Kinds:
  - Unit tests – PyUNO binding
  - Unit tests – components via UNO API

Python Test Growth



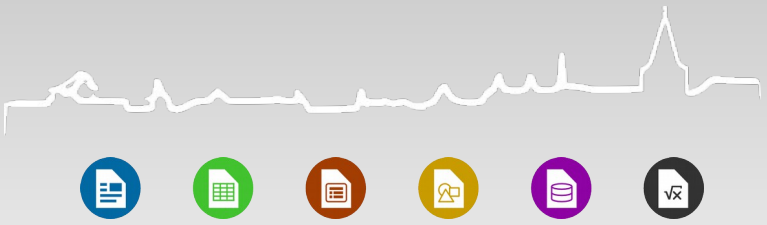
Tests Run by make check



Requirements Checklist

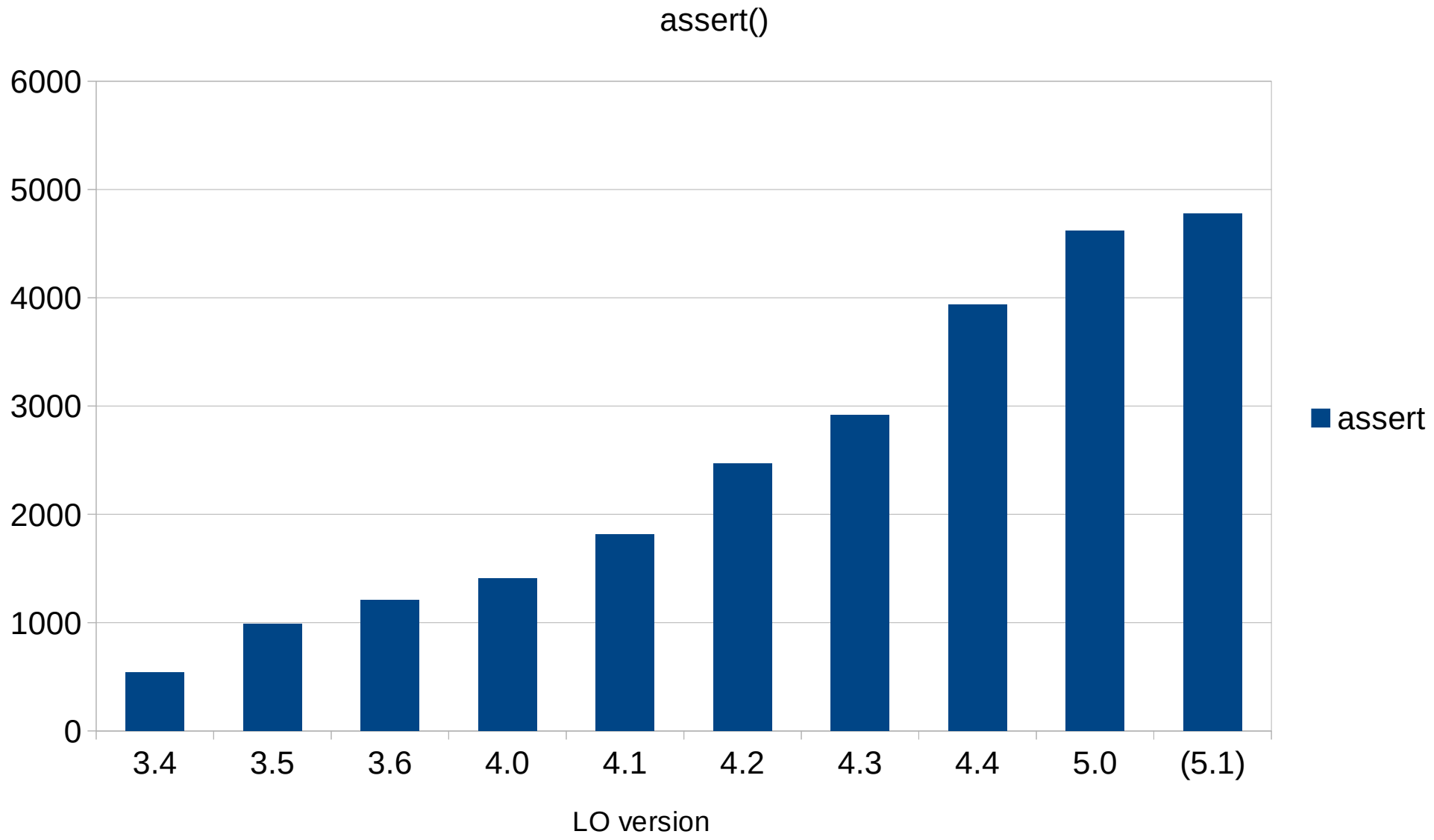
	Std. libs	make check	Reliable	Fast	Defect localization	Debug -able
AutoTest / testtool	✘	✘	✘	✘	✘	?
CppUnit „unit-test“	✔	✔	✔	✔	✔	✔
CppUnit „filter-test”	✔	✔	(-)	✔	✘	✔
JUnit „complex“	✔	✔	(-)	✔	✔	✘
qadevOOo „unoapi“	✘	✔	(-) / ✘	(-) / ✔	✘	✘
Python unittest	✔	✔	✔ / ?	✔	✔	(-)

Better Testing With Assertions

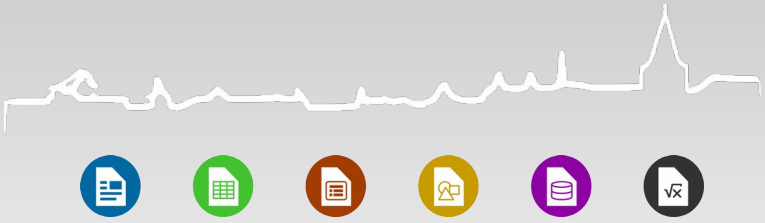


- Use `assert()` liberally in the product code to detect invalid states
- Assertion failure → abort and test failure
- But not necessarily good defect localization

assert() Growth

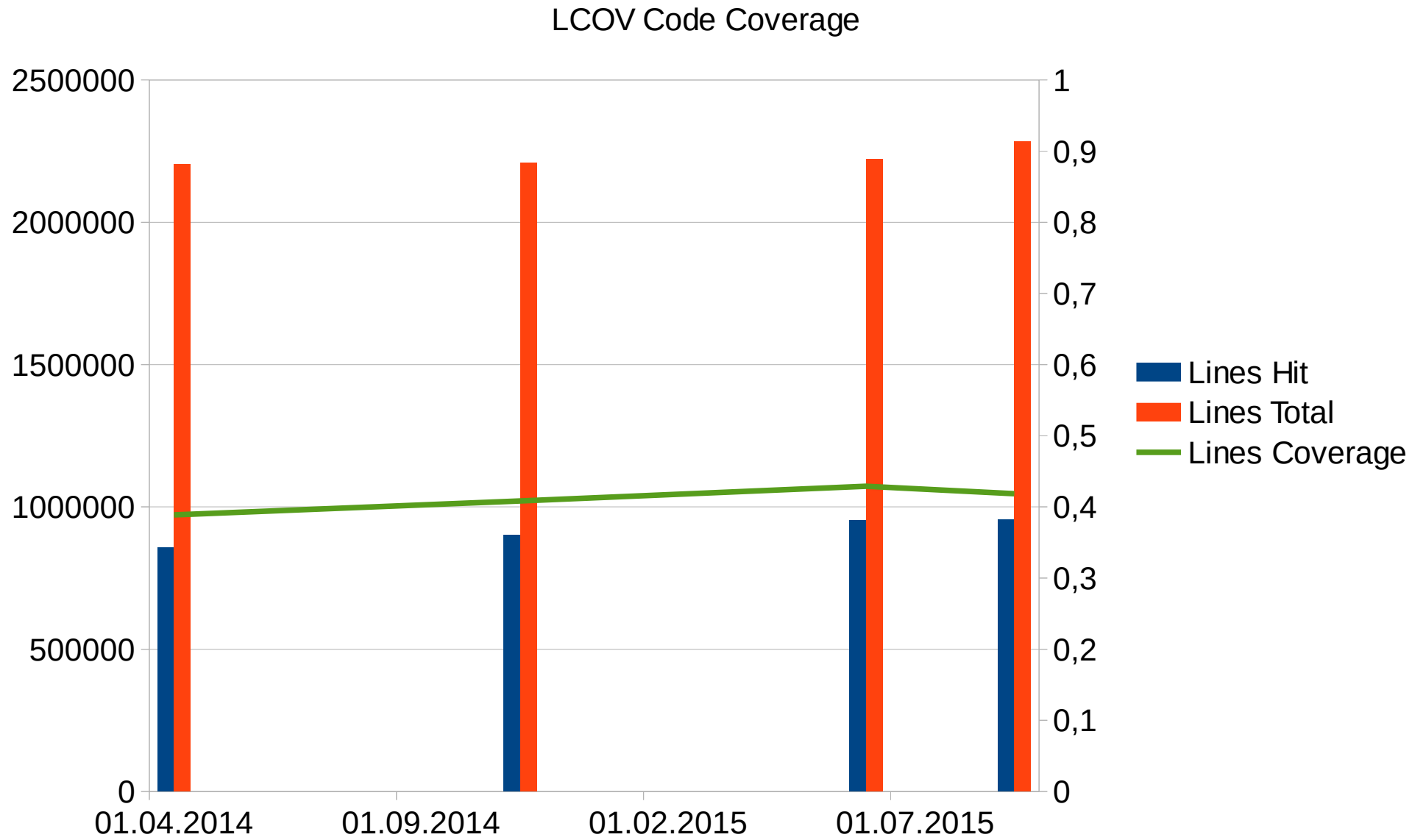


Code Coverage

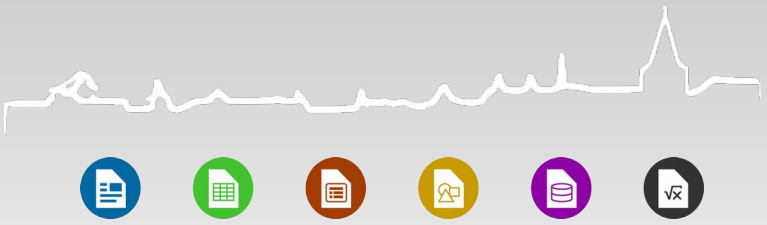


- GCC `-fprofile-arcs -ftest-coverage`
- LCOV
- http://lcov.libreoffice.org/
- Updated daily
- Thanks to Maarten Hoes

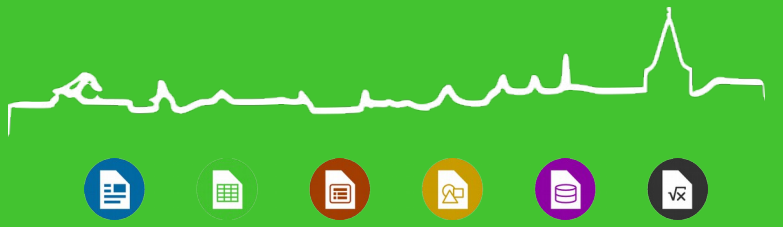
LCOV Code Coverage



More Tests



- “Crash Testing”
  - Import and export 80k documents
  - See talk by Caolán McNamara
- Performance test
  - Callgrind profiling
  - `make perfcheck`
  - http://perf.libreoffice.org/
  - Additional out-of-tree tests in test-files.git repo
  - Thanks to Matúš Kukan, Laurent Godard, Norbert Thiebaud



Now … go forth and **write more tests**!

- Thanks for listening.