**Considerations**:

* Maintenance is different in OSS, as it’s performed even after initial stable release (it’s all development)
* Many projects are never truly finished and simply evolve
* Does language dependence matter?
* Does paradigm matter?
* Testing Solution: utilize tools that show consistency between them (this will show better results)
  + http://delivery.acm.org.libproxy.auc.ca/10.1145/1400000/1390648/p131-lincke.pdf?ip=199.212.55.169&id=1390648&acc=ACTIVE%20SERVICE&key=FD0067F557510FFB%2E2E114FAB5F912086%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&CFID=939957675&CFTOKEN=22411919&\_\_acm\_\_=1495551165\_e674614c6ce5f4fd29325e64f174bb93
* Project Solution: Improve off the tests performed in Object-Oriented article

Note: consider some metrics for normal software, and some which analyze open source software

Things to consider:

* Well-designed
* Well coded
* Well tested
* Error free

Open Source Factors

* Different levels of programmers (levels of involvement)
* Frequent beta releases
* Platform-independence
* Many compile-time and run-time configurations

Models: aggregate the attributes of quality in order to give an overall view of the quality

* ISO 9126 Model (not necessarily direct measurements)
  + Functionality
  + Reliability
  + Usability
  + Maintainability
  + Portability
  + Efficiency

Things to Look Over

* <http://dl.acm.org.libproxy.auc.ca/citation.cfm?id=2641525&CFID=931840923&CFTOKEN=96999064>
* <http://www.sciencedirect.com/science/article/pii/S1571066109000632>
* <http://www.sciencedirect.com/science/article/pii/S2212667813000178>
* <http://dl.acm.org.libproxy.auc.ca/citation.cfm?id=2791466&CFID=931840923&CFTOKEN=96999064>
* <http://dl.acm.org.libproxy.auc.ca/citation.cfm?id=1370131&CFID=931840923&CFTOKEN=96999064>
* <http://dl.acm.org.libproxy.auc.ca/citation.cfm?id=1083268&CFID=931840923&CFTOKEN=96999064>
* <http://dl.acm.org.libproxy.auc.ca/citation.cfm?id=1852864&CFID=931840923&CFTOKEN=96999064>
* <http://dl.acm.org.libproxy.auc.ca/citation.cfm?id=2833013&CFID=931840923&CFTOKEN=96999064>
* <http://dl.acm.org.libproxy.auc.ca/citation.cfm?id=2501539&CFID=931840923&CFTOKEN=96999064>
* <http://dl.acm.org.libproxy.auc.ca/citation.cfm?id=3011786&CFID=931840923&CFTOKEN=96999064>
* <http://flosshub.org/sites/flosshub.org/files/HalloranScherlis.pdf>

<http://blog.smartbear.com/news/measuring-success-in-an-open-source-project/>

<https://jose-manuel.me/thesis/references/GB_Expert_Letter_Open_Source_Maturity_Model_1.5.3.pdf>

<https://www.dmst.aueb.gr/dds/pubs/conf/2008-OSS-qmodel/html/SGSS08.htm>

Book:

<https://books.google.ca/books?hl=en&lr=&id=lx_OBQAAQBAJ&oi=fnd&pg=PP1&dq=software+metrics+articles&ots=_TpVVlROZB&sig=hlFBGEIGsCtG45h_54oO2KuJ1Iw#v=onepage&q=software%20metrics%20articles&f=false>

Static Errors for code quality

Coupling for code quality

Runtime Length for common tasks comparison

Documentation per code comparison

Bug resolving rate

Get a couple open source programs in same language and compare based on review