Ao's Notes on Medical Imaging Software

Ao Dong

May 13, 2020

Table 1: Revision History

Date	Author(s)	Change
Jan/30/2020	Ao Dong	Initial draft
Feb/02/2020	Ao Dong	Update
Feb/08/2020	Ao Dong	Changed table to a spreadsheet link
Mar/01/2020	Ao Dong	Updated quality measurements

1 Software List

The link to the software list (Google Sheet): LINK
There are 50 packages in the list.

1.1 Sources

The above list is built based on 7 sources - 3 academical papers, 3 blog articles, and 1 online forum discussion. All of the sources are recorded on the second sheet in the list.

1.2 Orders

The software packages are put in the list roughly by the order of how many times they are mentioned in the above sources.

1.3 Selection

A few software packages mentioned in the sources are excluded, since they are in the following cases:

- Commercialized. Some software are not free any more. For example, PacsOne Premium, JVS-DiComPlus, RadiAnt, Synedra, TurtleSeg, AlgoM, Athena.

- No recent updates. A few software without recent release are excluded, such as PacsOne Basic (2005), K-PACS (developed for Windows XP or Windows VISTA), JVS-DiCom (2012), MITO-DICOM Viewer (2015), Aeskulap (2007), Fusionviewer (2008), Kradview (2013), Mayam (2013), Eviewbox (2015), EzDICOM (2013), Agnosco (2011), ONIS Free Version (2016 and not open-source), DICOMscope (2001).

1.4 Categories

The software packages are roughly divided into 3 major categories - Tool Kit, PACS, and Viewer.

- Tool Kits are usually used by developers to create new software, and there are only a few of them in the list.
- PACS stands for Picture Archiving and Communications System, which can be regarded as a type of server. There are also only several of them in the list, since most PACS solutions are commercial nowadays.
- Most software in the list are in the Viewer category, and many of these ones also provide other functions such as analysis, conversion, segmentation, etc.

2 Quality Measurements

2.1 Interoperability

Definition 1. The degree to which two or more systems, products or components can exchange information and use the information that has been exchanged.

There are not many measuring methods in papers, and most of them are very complicated.

- Does the software use any API to transfer information?
- Does the software generate output files with uncommon formats or extensions that can only be used by itself?
- Can the output be used by other software as input?
- Can it take output from other software as input?
- Can the software work with customized plug-ins?

Measuring aspects from [Smith et al., 2018]:

- Does the software interoperate with external systems? (yes*, no)
- Is there a workflow that uses other softwares? (yes*, no)
- If there are external interactions, is the API clearly defined? (yes*, no, n/a)

2.2 Visibility/Transparency

Definition 2. The extent to which all of the steps of a software development process and the current status of it are conveyed clearly.

The ones with a * mark might be hard to know.

- Does the software use any version and issue tracking system, such as Github, Gitlab, JIRA, Rally, VersionOne, etc.?
- The percentage of (open) issues assigned to specific developers.
- Does the software have documents recording the development process and status?
- Does the software have clear release log with essential information, such as release date, bug fixed and new features?
- *Are there weekly or monthly status reports for the development process?
- *Are there any project management tools used?
- *Are there any project indicators used, such as burndown charts, cumulative flow diagrams, and other metrics?

Measuring aspects from [Smith et al., 2018]:

- Is the development process defined? If yes, what process is used. (yes*, no, n/a)
- Ease of external examination relative to other products considered? (1 ..10)

2.3 Productivity

Definition 3. Productivity is the amount of output per unit of input used, which can be measured by the summation of all output (such as the number of lines of new code, the number of pages of new documents and the number of new test cases) produced per person-day.

It can be hard for a third party to know how many days the developers worked on the project.

- The lines of new code produced per person-day.
- The pages of new documents produced per person-day.
- The number of new test cases finished per person-day.
- The number of issues closed per person-day.

2.4 Completeness

Definition 4. A specification is complete to the extent that all of its parts are present and each part is fully developed.

Are there the following documents?

- SRS
- VnV plan & reports
- MG & MIS
- Readme
- User manual (including tutorials installation instructions)
- Release log

Inspired by [Boehm, 1984]:

- Is there anything left to be determined?
- Is there any references in the specification to functions, inputs, or outputs (including databases) not defined in the specification?
- Is there any specification items missing?
- Is there any functions missing?
- For multi-products software package, is there any products missing?

2.5 Consistency

Definition 5. A specification is consistent to the extent that its provisions do not conflict with each other or with governing specifications and objectives.

Are the following items of documents consistent?

- Language
- File format (pdf, html)
- Location (GitHub wiki, offline)
- Wording format
- Syntax
- Abbreviations and Acronyms

Inspired by [Boehm, 1984]:

- Is there any specification items conflicting with each other in one document?
- Is there any conflicts between different documents?

References

B. W. Boehm. Verifying and validating software requirements and design specifications. *IEEE Software*, 1(1):75–88, Jan 1984. doi: 10.1109/MS.1984.233702.

W. Spencer Smith, Zheng Zeng, and Jacques Carette. Seismology software: State of the practice. *Journal of Seismology*, 22(3):755–788, May 2018.