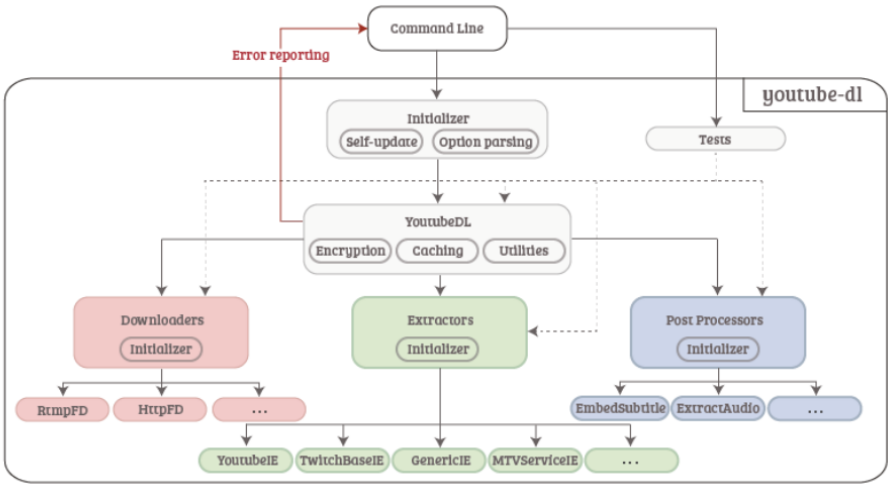
1. The system uses a pipe and filter architecture for its architectural pattern. The model uses a runtime organization of the system where function transformations process their inputs and produce outputs. Data moves from one to another and it transforms as it moves through the sequence. The primary components that need to be developed are easily identified by the architectural model so managers can easily create a distribution plan for the development of the system.



In this diagram, it shows how the software applictaion's program flows. The software has 4 main components: Youtube-D(the core of the application and is responsible for the overall process), Extractors (responsible for gathering the information about the video), Downloaders (allows the transfer of a remote video to the local filesystem) and Post Processors (responsible for any post-download operations that should be applied on the video. The input is received in the command line. Then, the software receives this information. There can be two options for the input, either it is placed by the customers or going to be tested by the developers. If the input is made by the developers, then the primary purpose of running the application is to test the code. If the input is made by the customer, then it would go through the initializer. Then, it goes through the Youtube-DL system for encryption, coaching and utilities. Then, the program would proceed to either Downloaders, Extractors and Post Processors. This structure is designed so that the functionalities of youtube-dl can be extended easily.
The architecture model of Youtube-Dl is designed in a way to encourage discussions about the system design. It uses a high-level architectural view of a system to communicate easily with stakeholders by giving them an abstract view of the system. Thereof, the stakeholders can discuss the system with full understanding and less confusion. The software system has a folder structure wherein there are separate folders for the different kinds of implementations. Inside the repository the folder of youtube-dl exists and then inside it are subfolders where the code for the systems downloader, extractor and post processor are contained.

2. The site does not account for dependability of the software since it is only maintained by the public-domain license and does not have a foundation nor company attached to it that is involved in backing or supporting the maintenance or development of the application. Thus, it would be hard for the system to build trust with its customers if it does not have consistent engineers to work and ensure that the program would operate correctly at all times.
   The site takes into account reliability. The software has a built-in testing system that tests each of the IEs of the application and the developers are required to write a test in order to make sure that the code that they are uploading works. The developers are also required to use the Flake8 tool to check adherence to the PEP8 style conventions, logical errors and code complexity. Then, when developers contribute to the software, they have to pull a request first and it has to be reviewed by the maintainers before being uploaded. Hence, the system is reliable since the engineers ensured that the system will correctly deliver its services as expected by the user.
   The site does not take into account the system resilience since it did not plan or build a tool that would help the system to maintain the continuity of its services in the presence of disruptive events such as cyber attacks or equipment failure.

3. The foundation and the Python Programming language enabled the Youtube-DLL to run on all operating systems. This enabled the application to use software reuse since there would be no need to make different versions of the application to be developed for different operating environments and peripheral devices. There is a possibility to integrate or reuse Youtube-DL in another application like a GUI wrapper since the code of Youtube-DL is free to be used and distributed. However, if some changes were to be made in the code of Youtube-DL that will affect the operation of GUI wrapper especially if the changes were made in the portion of the code that was reused. So far, there was only some change in parameters and added functionality so updating Youtube-DL would not cause the integrated software to not operate.
   To assess the quality of the software that is being developed, the developers uses a built-in testing system that tests each of the IEs of the application and the developers are required to write a test in order to make sure that the code that they are uploading works. Moreover, the developers are also required to use the Flake8 tool to check adherence to the PEP8 style conventions, logical errors and code complexity. To check that the code passes the software standard, when developers contribute to the software, they have to pull a request first and it has to be inspected by the maintainers before being uploaded. The maintainers oversee and review all of the contributions. This ensures that the software maintains its quality and produces code that operates correctly which would make the software reliable. Whenever there would be issues, the maintainers are

responsible for fixing bug reports and receiving feature requests that makes sure that the software is continuously improving.

4. The team who made the software does not earn from the application since Youtube-dl is released in the public domain and it is free to be copied, modified, sold, distributed etc. Thus, the software does not generate cash. Most of the developers that have contributed in the making of the application are not working in the application to gain income. They are just granted as collaborators and access to the repository. There are 4 main Maintainers for the software and the rest are just developers who wanted to collaborate and help build the software. It wasn't stated in the document if the 4 maintainers are given money for maintaining the system. The application used to use the Travis CI System system for continuous integration but they have abandoned it for years since the system builds up technical debt. I believe that the website is being run for free and the people who have helped develop the application are people who just wanted to support the application so there is no effort cost. The site uses testing tools such as Flake8 and Tox test which are free.

5. The software system is mainly run by 4 people and some developers who wanted to contribute to the creation of the software. Therefore, Youtube-DL is only a very small team so it does not need a very formal management structure. When making changes in the code, the team uses Github. One of the members, Sergey, is often the one involved in answering questions from the community. The website of Ricardo Garcia is what the customers use to ask questions and communicate with the developers. They use this feature request tool to look at risk for a potential problem in the software and be able to solve it right away. This serves as the Risk Monitoring tool for the software. The software being developed is a consumer product so formal records of project management are unnecessary. The teamwork is handled well in this project. The 4 main members are all actively involved in the development and have collaborator access to the project in Github. They oversee and review all new contributions and assure that the quality of the project remains par. The members of the software development team are not chosen. Everyone is free to collaborate and help improve the software, however, their code just needs to be scrutinized first.
   The software contains testing features created for the developers and supporters of the project and these tests are made for error reporting, logging and simulation. These tests are made independent of each other. The testing system used is Tox test tool for Python which is used to test each of the IE in the application. The developers also use the Flake8 tool to check for adherence to the PEP8 style conventions, logical errors and code complexity.Once developers finish their code, the maintainers review them. In order to do that, the maintainers perform software inspection by checking if the code operates perfectly and if the developers attempt to follow the instructions and adhere to the

guidelines and standards of the External entities and Interface. The developers had also created a test program flow wherein tests can be invoked from the command-line and they can be performed on the software itself, its Extractors, the Downloaders and the post processors. The developers had found that being able to run only in specific tests allows them to quickly test a specific part.

Risk Assessment of Youtube-DL

Risk Identification:

- Staff Turnover
  - A potential risk in this project is losing one of the developers
- Management Change

- If there will be a change in the management of the software
  - Software Tool Underperformance
    - This would be a risk if one of the tools that are used in the software does not perform as it should be.
  - Product Competition
    - This is a risk especially if there is a new software that works exactly as Youtube-dl that is introduced in the market

Risk Analysis:

- Staff Turnover
  - A potential risk in this project is losing one of the developers, especially one of the 4 min Maintainers. It would be a huge loss for the software application since these 4 people are the one who developed and know how the code operates and functions. It would take a huge cost in time to train a new staff to do the job. Also, one of the maintainers Sergey M is responsible for answering from the community oor taking care of bug reports or code that misses functionality. Hence, losing a staff would take a huge toll on the software application especially to its maintenance.

- Management Change

- If there will be a change in the management of the software, this is a huge risk to the organizational environment where the software is being developed. This might slow the production of the software since some of the staff might have trouble performing well in their newly assigned roles.
  - Software Tool Underperformance
    - This would be a risk if one of the tools that are used in the software does not perform as it should be. For example, the tos test tool and Flake 8 are used in testing the software. If these tools fail to identify errors in the code, the developers would upload it in Github and the users will

experience errors and bugs when they use the application. This might risk potential security and safety problems and dissatisfaction among customers.

- Product Competition
  - This is a risk especially if there is a new software that works exactly as Youtube-dl that is introduced in the market. For example, youtube-dl is dealing with competitors like DVDVideoSoft that has almost the same functionality as the application.

Risk Planning:

- Staff Turnover & Management Change
  - In order to avoid complications when a staff turnover occurs, the team should have more overlap of work and each one should understand each other's jobs
- Software Tool Underperformance
  - Find a replacement for tools that are potentially not working as anticipated. It would cost more money and time if the tool is still used even if it does not operate as how it should be.
- Product Competition
  - The product should be continuously maintained and updated in order to ensure that the system operates well and maintains high quality. The developers should consider the user's feature request in order to maintain satisfaction of the customers and build trust and have a loyal relationship with them.

Risk Assessment:

The developers should regularly monitor all the stages of the project. They should determine the arising risk and check the consequences that it imposes to the software and the management.

- Staff Turnover and A Management Change
  - A potential indication of a staff turnover is if one of the developers has been reporting personal and professional problems that are affecting his work performance. This should be acknowledge right away.
- Software Tool Underperformance
  - If most of the developers refuses or have complaints in using the tool, then the software tool must have a problem or does not meet the software standards and should be changed.
- Product Competition
  - This is noticed if there is a decreased in the usage of the software. Then, the user must be using an alternative application.