A Case Study in Qlik Core's Developer Experience

Background and Purpose

This paper aims to

Software platforms and software ecosystems

__ Text about openness/closeness of a software platform grows dependence, but risks given advantages to competitors __ __ Qlik Core builds on other platforms (Javacript, Docker, etc). Is it it's own platform? __

What is Developer Experience?

Developer Experience, or DX, is similiar to the more well known User Experience (UX), but with the user being a software developer. DX is defined by Sam Jarman as

"*/...the experience developers have when they use your product, be it client libraries, SDKs, frameworks, open source code, tools, API, technology or service.*" *Sam Jarman*.

How do we define 'Good DX'?

There are many potential factors for defining what constitues 'Good' DX. *EveryDeveloper* has developed a *DX Index* from 1-10, where they consider four factors:

- Are the libraries available in popular languages?
- How prominent, in-depth are the starting guides?
- Are the solutions self-serving, without need of demos or 'call us'?
- Is the pricing clearly stated?

Sam Jarman has other factors for he uses to evaulate if something gives a good DX. He for example puts emphesis on communication between the product provider and the developer. The dialog between the product provider and the community needs to be authentic, open and honest in order the give a good developer experience, according to Jarman. He also states that ...

Graziotin, et. al. researched what makes a developer happy and unhappy, and found several indicators. They found both internal- and extrernal factors, where external are the most interesting for this project. However, the internal unhappiness factor of 'work withdrawel' is worth noticing. Being stuck on a task without any progress for too long leads to unhappiness.

Execution

Initial Survey

After gathering potential factors, through litteratur and brainstorming, on what would lead to good DX, it was concluded that a smaller, initial survey would be relevant to conduct to test the waters on the potential factors that had been found. It also intended to find more potential factors that had not been thought of. The small survey got 38 responses, mostly from Qlik employees. The survey confirmed some initial assumtions but also raised some questions. It also highlighted that some questions may need to be added, some rephrased and some removed for the main survey. The survey questions can be seen in Appendix X.

Survey structure

The survey consisted of three parts. The first part was a small screener to gather some information about the person taking the survey. The second part was about users usage of new software and the third and last part was about the DX factors, and how important they were to the user. After the survey was done, the data was gathered and evaluated using Qlik Sense. Although the dataset was too small to make any direct affirmations, it gave some indications. In the part about the DX factors, the user were to rank each factor depending on how often they considered the factor when finding new software. To able to rank the answers, each answer was given a value: 'Always Consider' (+5), 'Often Consider' (+3), 'Sometimes Consider' (0), 'Rarely Consider' (-3), 'Never Consider' (-5). After summing them up, I was able to give each question a score from -10 to 10, where a 10.0 score would mean that everyone answered 'Always consider' and -10.0 would mean that everyone answered 'Never consider', and a 0/10 would mean that the result had been even outed from both sides. The result of the survey can be seen in Appendix Y.

Indications of survey results

The results were evaluated through initially three different filters, which were compared to the overall group, to see if there were any trends amongst certain groups. The groups that were filtered by were: 'People with more than 5 years experience in the industry', 'People who are developers within the industry' and 'People in companies with more than 200 employees'. The last filter with the larger companies was scrapped since the majority of all answers were from Qlik, and the controll group was therefor to small to compare with.

Software factors

Overall, all but 3 factors scored a positive score, meaning that they are all factors are more often considered than not. The most important factor was 'The API has code examples', followed by the importance of an active community around the software, and thirdly that the API explainations were thorough. The least two important factors were 'The documentation has consistent language' and 'The release- and change notes are thorough'. This is an interesting find, since litterature highlights these to factors as being very important. Many companies also spend a lot of resources to keep documentation up-to-date and release notes thorough. Here we see that most people, more often than not, do *not* consider these factors.

For the more experienced users, the result differ some to the control group. They take less consideration to factors surrounding APIs, documentation and time to get started, and care more about pricing, cross-platform compatibility and the thoroughness of release notes.

With the group consisting of mainly developers, we see almost the inversion of some trends in the more-experience-group. The developer group cares about documentation and time-to-get-started, and do not care about release notes and cross-platform compatiblity.

Creator behind the software factors

Overall, the creator behind the software seem to be not non-important, but not very important, with all factors scoring close to 0/10. The most commonly considered factor was that the creator seemed professional, but that only ranked 3.0/10. The least consided factor was 'I have heard of the creator of the software before', which ranked -2.47/10.

With the more experienced users, they considered all factors even more rarely than the controll group, apart form transparency, which rose slightly. Still, the scores are close to 0, and the creator once again seem to be a somewhat important factor.

With the group consisting of only developers, it's much the same as before, with the one exception that 'I have heard of the creator of the software before' had an increase by 10%. It is however still the least considered factor.

Conclusions from survey

The two factors 'The documentation has consistent language' and 'The release- and change notes are thorough' scored surprisingly low. This goes against the litterature, and will be investigated more. A potential difference could be that question only states what would make the user *try* a new software. It may not be an initial stopper, but could cause issues once the user has decided to use the software.

The non-developer group cares about other things than the developers. This groups includes consultants, directors, managers and architects. These are positions in which they have power to make major decissions in the company. The group therefor have other priorities than the lone developer, that only considers his own workflow.

Good Sources

B2D practices and stratergies. Some key notes are that in B2D is that the value of the product must shown before the purchase. Let developers use the product! Not demos or trials.

Developer Personas. Defines different personas of developers. Who are the developers that will be using our product?