Answers

**Why do you choose this job?**

Work sounded interesting and something I could do, would be learning some new things, a challenge but not too far out of my comfort zone. Seems like a well-rounded role. I would not be in a niche, churning out development without any real idea of how it will be used. Would involve development but also interacting with users, designing solutions, deployment. Also I am looking for a job which is mostly working at home. I have appreciated being at home during the last 6 months, being around for my daughter, being able to pick her up and drop her off at school now, going to local parks.

My priorities have changed in the last few years. I don’t want to manage people or be solely responsible for delivery big projects. And I wanted to get back to developing again.

**What can you bring to the job?**

My expertise and my experience. My experience with dealing with stakeholders, developing, testing, releasing, supporting. My experience of dealing with personal data and the effects GDPR had on it.

**GDPR**

Removing data from the data warehouse and from Eloqua that met a certain age limit but depended on engagement. Had to retain opt outs and suppressions. Anonymising personal data in development environments. Creating a process to deal with data access requests, and requests to remove data. Also try to bed privacy into our design. Do not come up solutions that involve holding personal data in file locations, or email or anywhere. If you can come up with a solution that avoids moving data around then that is better

**Approaches to development**

I have a number of precepts that I try to work under.

**Do small releases often.** Ties in with continuous delivery and is a good DevOps principle. Release a piece of functionality in as small a chuck as is useful. Gives value to the user as soon as possible. And also, de-risks the release. In my experience, the bigger the release the more likely it is to fail. Eg with working to optimise client build.

**Test driven development.** Before you start developing, first thing you do is generate test data. That is data that is going to show whether the functionality works or not. Including negative and positive cases. Eg, I’m going to apply rules to change the data in address field. I want to include data that will be changed by these rules, but also data that won’t be. Generating this test data is time-consuming and may take longer than the actual development. Also then create a testing scripts which checks this data is as you’d expect. This data can be used in systest, OAT and also maybe even live.

**Users often don’t know what they really want until they get it.** Expect to keep going back to users to refine requirements. Even as you’re doing technical specification. Do prototypes. Show them examples of actual data. But understand your audience. Expect it not to be always right even when you release it. Another reason to do small releases often.

**Don’t over-engineer.** Up for debate. But try not to make your solution to flexible. Or to be too clever. If the requirement is only ever to have two integrations into a database then don’t spend time creating a solution that could allow more. Don’t fall in to the developer trap of thinking you can give additional functionality “for free”. Because it rarely is for free and often takes longer than you think.

**If something goes wrong, spend the time to fix.** It is easy to feel embarrassed wen something goes wrong and to want to fix it as quickly as possible. But that can lead to more problems. Take your time and go through the normal process.

**Why do you want to work for a charity?**

* I am keen to work for an organisation where the end objective isn’t simply to make money. It is actually to improve people’s live. I would find that more fulfilling and satisfying.
* I’ve been looking at your “working with us” page and you support good work-life balance and employee well-being.
* I think charities are less likely to take commercial risks like More2 did.
* I specifically like the idea of charities that support children. Supporting children and enabling them to reach their full potential not only benefits them and their families but also the whole of society.

**What was the development process?**

* Someone would raise a requirement or bug. This could be massive (integrate with Salesforce) to small change (I want a field relabelled.) They would generally do this by emailing anyone connected to data warehouse (product owners, me, support analysts, BA). Initial discussion to establish if it some development for us to do and I would give it a t-shirt size. Would go into the back log with its size and a priority.
* If it is a bug it would be worked on immediately (depending on severity.)
* Periodically had priority sessions (should follow sprints but often didn’t) to go through back-log and decide what need to be worked on in what order. Any Epics were identified. Requirement would be added a sprint for ANALYSIS.
* During ANALYSIS, a requirement would be understood in more detail. Discussions with users, me, BA, data analyst, anyone else who could be a stakeholder. Requirement broken down into user stores (which generally represented a piece of deliverable functionality.) User story written by BA including any business logic. My also include data analyst work. I would then write technical specification. This had to be very explicit as third-party developers had to develop from it. It would include data examples, mapping documents, test data. I could then give actual hours to the user story.
* Once analysis was complete, it was ready to be assigned to a sprint for DEVELOPMENT. Based on number of hours. I would then brief into Tester and Developers and would break down the user story into actual tasks. Including testing tasks and task for preparing release. Tester would come up with own test cases.
* During DEVELOPMENT, developer would assign task to themselves and work on them. Daily sprints which I acted as the scrum master. Test script would be used for unit testing and done by development. Would then got into PEER REVIEW. After that would go into FEATURE TESTING.
* FEATURE TESTING done by tester and test cases completed. Would then be ready for release.
* I would include in release script and then perform SYSTEM TESTING. Once that is done, unit tests were repeated and also REGRESSION TESTS.
* UAT may also be performed on Systest.
* Release would then be preformed on OAT first and then LIVE.
* Release notification then sent out. Testing in LIVE would also take place after refresh.

**What do you know about NDCS**

Leading charity for deaf children

Founded in 1944 as Society of St John of Beverley. Changed to Deaf Children’s Society and then National Deaf Children’s society

Visions is a world without barriers for every deaf child.

Your ambition is that, by 2030, no deaf child will be left behind – at school, in their community or at home. And you are undergoing various activities to meet that ambition. Your are measuring your success in these to ensure you are keeping on track. So you seem like you might be quite busy.

You provide a number of services for families, deaf children/young people, professionals and you run events and support groups. And supply a lot of information. And canvass governments/schools. Eg recently writing to schools to ensure they support deaf children when they go back to school.

**When have you made a mistake?**

EG updating everyone’s jobs in development. First thing, admit that you did something wrong. Then commit the time to fixing it. And I made sure my code to fix it was correct. Got peer review

**What has been a biggest challenge?**

Delivering on GDPR. Given a wholey unrealistic deadline 25th May that there was no chance of meeting. I tried to meet it but as it got closer, I decided it was too risky to go live. We had not done enough iterations of testing. Senior stakeholders wanted me to go live anyway and I had to convince them it was too risky. More risky than being in a perceived non-compliant state passed 25th May. In the end we took a different approach that de-risked it, delivered in chunks which completed in November.

At a lower level, wrote some development to load a new file used More2’s inhouse import tool. But it didn’t work on the dev server and I spent about 3 days getting it to work. Had to work with the only person in the team who knew anything about it. The task was supposed to take 1 days and ended up taking 5 days. That sort of thing happened all the time at More2 which is why it was such a frustrating place to work.

**What has been your biggest achievement to date?**

Taking on a project management role when I first started at More2. Wasn’t supposed to me my job but did it, improved the process (made it KANBAN from sprint-based) and ensured all changes were delivered on time and too specification.

Getting off-shore development team up and running.

**When have you changed you mind about an approach?**

GDPR, was going to every market at once but decided that was too risky so changed it to market by market.

**When have you had to understand a system with very little documentation?**

More2 client databases all had a complex architecture though they were trying to standardise it. Had to reverse engineer a number of elements to it. And know when to ask questions and what the questions should be and know when to work it out for myself.

**What complex SSIS package have you developed?**

Created a package which loaded data from various files into a standard set of tables in the etl schema. The package looped through a control table which held a list of sources, where the files where, how old they could be. Another table held which source provided which files. The package looped through each source. First it looked for the files. If the files couldn’t all be found, or if the files were too old then an error would be logged and it would move on to the next source. If it passed that check then the files were loaded into the tables in parallel. Any rows that couldn’t be loaded were put into an error file in the same source location. This was date stamped. Once every table had been loaded then a success would be recorded in the log with the number of rows that errored and the number of rows that errored. If a source was interrupted for some reason then an error would be logged.

The package was flexible enough to not expect all the columns. So a column could be added for a particular source and it wouldn’t have to be added to the other source.

The package used script task to do the validation and the loading from the files. Written in C#. I generally try to avoid doing too much transformation in the package itself. My experience is it doesn’t perform well. And can be buggy (years ago tried to do a merge join in a package and it just wouldn’t join the two record sets correctly.) My preference is to get the data into the database in to a staging environment and transform the data from there.

**When have you had to supply ad hoc queries**

More2 used SSRS to deliver their reporting

**When have you written technical documentation?**

Did that all the time. The most complex was writing technical documentation for changes to consolidation rules. Broke it down into functional areas eg

* Implementing rules to identify changes that an email belonged to an individual
* Implement contact group
* Identify master contact in consolidation group
* Storing the previous derived contactID so didn’t have to

Document rules for each function. Give examples of how should deal with data. So had excel spreadsheets full of how records should be dealt with. Test data. Then had to brief each user story into developers and testers.

**How did you ensure database was performant?**

Huge transformation of the data took place between the data warehouse and the presentation layer. Also, quite complex queries were hitting the database from the front-end tools.

My approach to transforming data was to break it down. Not try to do too much in one go. Do a small amount of transformation. Put it into a table. Do a bit more. Put it into another table. Delete the original table. This was the quickest way of doing the transformation, allowed it to be picked up at a certain point if it failed and also meant some of these tables could be available for end-users (with a caveat.) We also recorded how long each piece of the transformation took. We would monitor the process for where it was starting to take a longer time. Index were used when necessary but these had to be activity managed. Transformation would be done in parallel where it could. SSIS package controlled this.

There was a downside to this approach and that it generated a lot of I/O. This wasn’t a problem until the ETL process started to overlap business hours and users were also generating there own I/O. This could have been fixed by moving to solid-state drives but that never happened. There is a solution to this and that is to do the transformation on a different server to what is being used by users and just push the data over there at the last minute.