Theme: Shanghai library FOLIO project

Time: April 6, 2021 07:00pm (EST) / April 7, 2020 07:00am (GMT+8)

Attendees:
Vincent Bareau (Enterprise Architect, EBSCO)
Gang Zhou (Project manager, Shanghai library)
Sha Jiang (Technical Director, Jiatu)
Lucy Liu (Product Owner, Folio China)

Notes

1. **Project Updates from Shanghai**

   Sha Jiang: It took us three days to transform data from Horizon to Folio. The database went down twice because of the large amount of data. We have to verify and validate the data to make sure it was done correctly. The go live has been delayed.

   Vince: Will you do an official announcement or news release when you go live? A lot of people in the community are interested in knowing when you will go live and are waiting to see and hear that you have successfully gone live because it's a big deal for the project.

   Gang Zhou:

   - No official news release. We will announce it at the conference to be held at Suzhou Library later this month.
   - We have a few servers, such as the production server and the UAT server. When we import all the data to the servers, we found the search was not stable. The response time was one second on the production server and it took more time on the UAT server with the same data size. We thought it might be related to the index in the database. Are there any index rules for the huge size environment in the community? Do you have any experience or advice on this issue? When we validated the data from the product, we found some indexes were changed or lost when we migrated the data or when we deployed the apps.

   Vince: Are you talking about ElasticSearch or just regular inventory search?

   Gang Zhou: PostgreSQL.

   Vince:

   - The database is going to be working very hard when doing index based search on huge data size. So we constantly tune the indexes and the queries to accommodate performance problems that came up.
   - The database will take over on its own. It will calculate an execution plan. Sometimes it may choose to bypass the index for whatever reason. And the index doesn't even matter because it's doing a full table scan.

- I don’t see why it takes significantly longer than the other if the two systems are truly identical. There must be some differences between them. Maybe the data are somehow different, or maybe there is old data left over, or maybe just simply that the database itself does not have the same resources available to it for a CPU or memory, and so it chooses a different execution plan because of that. But I don't really know. It’s going to be really specific to yourself.

Sha Jiang: Should we create index in the module descriptor or put the index definition to the database server?

Vince:

- It doesn't matter as long as the index is created on the database.
- The reason you would want to do it in the module descriptor is because you want to have an automation way or a reproducible way of doing it. For example, when you set up your UAT server after you set up the live server, you can be sure that you set it up exactly the same way.
- Obviously, you can also script your index creation directly against the database if you wish. It allows you to build into a proper consistent process that always executes the same script. And you get the identical result next time you set one up. That’s the only benefit of the scriptor being responsible for the index. The scriptor is going to be managed. So it will be updated when there's an update that needs to be applied accordingly. If there's an update that somehow interferes with the index that you created outside, the module may not be able to handle it. But I don’t think this is necessarily very important or going to be a likely case.
- You have to assume that the module descriptor definition of index is going to be correctly applied and successfully applied by RMB. Obviously, you can always verify to ensure that it's happening properly. Other than that, I don't know that one way or the other makes a lot of difference. You can certainly manage yourself. And as long as you're being consistent how you apply it and put it into the script, it shouldn’t be a problem.
- Keep in mind that the more indexes you have, the more burden you put on the machine to update the index every time there is a write operation to the database. So you have to carefully balanced how many indexes, which ones you put in there, and how you do it.

2. **Spring Way**

Spring Way is one of the big changes that will happen to folio.

- **How did the conversation on this start?**

  We have a lot of experienced Java people in the development teams. They feel that working with RMB is more difficult than working with a more traditional framework for Java. They already have experience with the traditional framework. RMB makes you program in a way that is called reactive or event driven, which is not the way that they are used to working. And it adds extra burden to make that work properly. Some EPAM engineers, and also Mikhail, suggested that we create a new framework using Spring. Taras put a slide deck. Vince, as the architect, worked with them to make sure that they met all the requirements that needed to be there. It needs to meet certain number of conditions and requirements in order to present it as a complete replacement of RMB. This is the background.

- **Why is Spring Way a good replacement for RMB? (See the attached slides)**
    - Spring is more robust and mature compared with other frameworks (slide 3)
    - The defects against RMB rose. (slide 4)
    - The pain points that cause people want to change (slides 5-7)
        - Vert.x & RMB major versions' bump up is difficult
        - Performance issue
        - Developers forced to use “Reactive programming paradigm” only
        - CQL and JSON limitation
        - Lack of functionality available out of the box in other tool-kits or frameworks
        - “Call back hell”
    - Requirements Limitation and Constraints (slide 8)
        - “API First” declarative approach MUST be used
        - Pagination, Filtering, and Sorting options out of the box
        - Full support & compatibility for RMB Management/Administration/Tenant... API required by OKAPI
        - Transparent propagation for “X-okapi” HTTP headers
        - Full support for DB Schema per tenant approach (tenant identifier passed through HTTP headers)
    - Spring Solution (slides 9-13)
    - Proof of Concept: mod-users (slides 14-24)
    - The 1st module that was written for folio using the Spring framework: mod-password-validator (slide 25)

- **Do we need to change the code in the current modules?**

  The new modules (mod-search, data export, etc.) in the community are using Spring Way. If you create some new modules, you should use Spring Way. If you're looking at the old models from the projects such as inventory that were written in RMB, I would not suggest that you choose to rewrite them yourselves. At some point in the future, it is quite likely that all the modules will eventually be re-written with Spring Way, probably as a result of breaking them up into smaller modules. But so far there's no plan to do that.

- **Is Spring Way optional or compulsory for developers or new modules?**

  No framework is compulsory. We only recommend frameworks. The recommended framework will be Spring Way instead of RMB.

- **Are there plan to move Okapi to Spring Way?**

No. Okapi is not written with RMB. It’s a separate context. The scope of this is module development.

- We have a Spring Way Task Force in the community. What will that team do? What kinds of changes can we expect in the near future?

  Some of their initial focuses are to be responsible for

  - any defects that are reported in the framework;
  - anything in the stack that needs to be updated. If there's a new version of any one of the components that are used here, they will ensure that it works and gets updated into the framework;
  - complete those things which are still loose. There are a few things that we need to support, including creating tutorials, guide, and documentation, etc.

- Do we have a timeline for the new framework to totally replace RMB?

  That’s not tasked so far. They are focusing on Spring Way, not the use of Spring Way.

- Some other resources on wiki
  - Spring Force Charter
  - Data export by using Spring Batch (aka Export Manager)
  - Tech Council Meeting Notes 2020-11-04
  - Tech Council Meeting Notes 2021-01-13
  - Tech Council Meeting Notes 2021-02-10
  - Tech Council Meeting Notes 2021-03-24