Theme: App Store

Time: June 30, 2022 08:00-9:30pm (GMT+8)

Attendees:

From FOLIO community（10 attendees：2CC，1 PC，4TC；Index Data 3；EBSCO 3；K-Int 2；UChicago 1；LehighU 1）:
Sebastian Hammer (President, Index Data)
Mike Taylor（Software, Index Data）
Peter Murray (PC Member, Open Source Community Advocate, Index Data)
Vince Bareau（Enterprise Architect, EBSCO）
Zak Burke（TC Member, UI dev, EBSCO）
Harry Kaplanian（CC Member, EBSCO）
Ian Ibbotson（CC Member, Director@K-Int）
Marc Johnson（TC Member, Knowledge Integration）
Tod Olson（TC Member, University of Chicago）
Maccabee Levine (TC Member, Senior Library Application Developer, Lehigh University)

From China's community（9 attendees）:
Keven Liu (CC Member, Deputy Director, Shanghai Library)
Zhou Gang (PC Member, Project manager, Shanghai library)
Zhang Chunjing (Secretary of China Community, Shanghai Library)
Jiang Sha (Technical Director, Jiatu)
Xu Chi (Manager, Jiatu)
Cheng Bei (Associate Dean of Shandong University Library)
Xiao Zheng (Associate Dean of Xiamen University Library)
Dong Xiaoju (Associate Dean of Shanghai Jiaotong University Library)
Lucy Liu (Product Owner, Folio China)

Notes:

1. **Getting to know each other**
2. **Review of App Store/Marketplace/Folio Factory discussions in the FOLIO community**
    - Summary of communications on Slack #app-store before the meeting:
      App Store was mentioned in the FOLIO community before, but it was never formally submitted to any decision making bodies as a proposal. A mechanism needs to be in place so that people can find what apps are available in FOLIO or work with FOLIO. However, the App Store might not be the right model.
      Reasons:
        - Modules are heavily inter-dependent. And we don't have a plan to remove or reduce dependency in the community for now.
        - We don't have a definition of an App / module or what artifacts are included in it. So it's hard to say what things are in the App Store and what it means to install them.

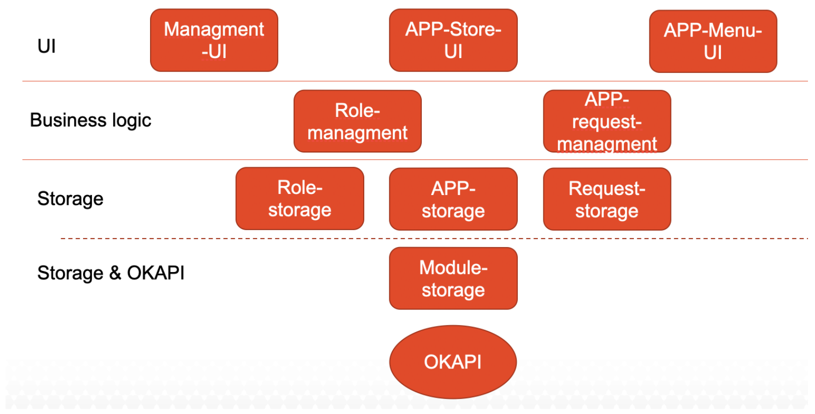
- In meeting discussions:
  - **Tod**: There's an aspiration to share code as convenient as possible. The challenge now is that modules are intertwined. For example, an acquisition action will interact with Inventory and Finances in both ways. We haven't figured out how to address this issue yet.
  - **Jiang Sha**: Modules are inter-depending on FOLIO apis. Could we leave the apis still? We can have different modules implement the same apis and switch from one module to another. Can this reduce inter-dependencies?
  - **Seb**: That was part of the vision when the system first took shape. Agree with Tod. Thinking something like Inventory. There are so many cross dependencies. Thinking of replacing all the inventories, a lot of different apps that you implement. Some of those apis might be specific to particular implementation and particular data model. Replacing Inventory means you have to reimplement so many things to support the same set of apis. Inventory may be unusual. How many apis does it support? Instances, holding, items, and so on. If you want to replace an entire app, you would have to prepare to fill in the dependencies. They may not be well documented as they should be.
  - **Mike**: Every app is supposed to and must define apis in machine readable form. Replacing Inventory is the single most dramatic openhouse surgery we could sense. If we do want to do it, in principle, it wouldn't be an easy thing to do, but would be simple, because it would be well defined what things are in place that an inventory needs to do. To me, an app store is for adding different modules from different places, as some universities have experimented with the idea.
  - **Marc**: Agree with folks that, in theory, we have the apis in place, the things we could use to replace things. And the interface definitions are what we would use to do that. In practice, it is not quite possible because a lot of the things that different parts of the system rely on are not part of the definitions. The machine readable interfaces that we have only tell you the most basic stuff about the apis. They don't tell you a lot of the things that happen like what things can be searched, or what two fields can be altered or used together. Some values might only be validated in those circumstances. All these things have side effects. So we replace Inventory, but there is a background task in Inventory that does stuff. And the interface knows nothing about that. So we lose a lot of functionality. In practice, the challenge is way beyond api definition.
  - **Keven**: We are more interested in using the app store for applications other than the core library applications, i.e., not the ILS modules, but the

new demands and new apps. They can usually loosely be coupled and clearly defined with apis.

- **Seb**: Agree with Mike and Marc that we shouldn't get too hung up on the challenges of replacing the core heavy apps that are interdependent with each other. So maybe we should try the new apps, and shouldn't hang up on the legacy apps.
- **Keven**: In China, we have more needs for new modules and new functionality for our platform. So we will not advertise core modules to libraries. They hesitate to change it because they have already had it. We advertise the new functionality and they will accept the new platform.
- **Vince**: Yes. We have several problems with the legacy system. One is the problem of replacing things. Adding new functionality is simpler. There's another form of dependency - the vertical dependency. Which means defining the relationship between modules: what are the backend modules and the frontend modules? We have a pretty good scheme horizontally. But UI modules are tied to backend modules. The top-to-bottom relationship is not very well defined and needs to be dealt with before we build an app store.
- **Tod**: The kind of adding applications around the edges and that kind of extension of functionality is much more approachable. There's a lot of value across the globe for that kind of thing. We have various kinds of small applications that we try to look for ways to trade between libraries or some pieces of functionality that might be contracted for. Some significant percentage of places would like X module to do a particular integration or whatever. To make it easy to bring in, that would be fantastic.
- **Keven**: Let me take some examples. At Shanghai Library, we will have some very special modules, such as remote borrow, LDT (something beyond LDP), resource reservation, navigation, robot services and more. We would like to make them standard after Shanghai Library goes live. We hope such modules can be used by other libraries.
- **Seb**:
  - It would be really nice to be able to extend the platform freely. Everybody has ideas of things they would like to add. And some of them wish to be in the business to add those things.
  - Defining a mechanism to do this is a real necessity for the platform. The experience that we have had of working with pubs to add new functionality is that the core folio community that is responsible for the roadmap of the core functionality is not well set up to process and validate new apps coming from the sideline. So

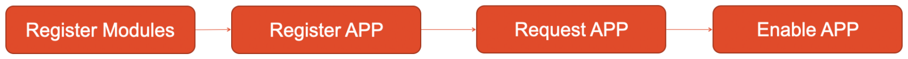
        the process of adding new functionality to the flower releases has felt very difficult, kind of complicated. Like we are almost sort of imposing, when we bring this requirement, we see as one customer sees. But does everybody see it? Should everybody spend resources validating these apps and think about where these apps fit in? I don't think that's anything negative about the core roadmap governance. It just wasn't really decided to handle all possible things that might pass through potential app stores. So coming up with a scheme that would allow us to add functionality without going through a shared roadmap or a shared priorities is pretty important to allow the community to grow as we all hope it would.
        - There are some technical issues in terms of how you package these apps to make this happen. But there are also some organizational/community issues that we have to deal with in terms of quality, quality commitment that we can make to each other, how we approach testing for things that are not part of the flower releases at the joint community testing, and other things like that. It's an essential challenge for folio to deliver the promise of an innovation platform.
    - **Marc**: The trust and the social contract around this is probably more important than the technical implementation.

3. **The App Store presentation by Jiang Sha**
    - Goal: Libraries could choose APPs provided by both community and business companies. In high demand in China's market. Not intended to replace the core functions of FOLIO, but to provide new functionality such as remote borrowing, etc.
    - Functions
        - Manage APPs for an online FOLIO system
        - Display APPs for user selection
        - Deal with the APP activation requests
        - Enable & Disable APPs in real time
    - Architecture: Three layers in the architecture.

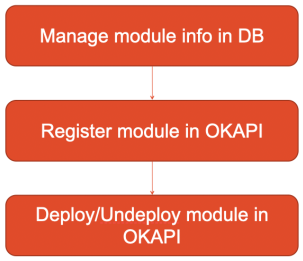


    - Storage has two parts: Role-, APP-, Request-storage purely connected to the database. They store information about apps, roles and requests in the tables of databases. Module-storage collects both database and Okapi. It stores module info in Okapi and maintains this message through Okapi API so that we can manage modules in Okapi.
    - Business logic: Two modules. Role-management manages relations between role and permissions. APP-request-management deals with the workflow of app activation requests.
    - UI: Three modules. Management-UI is for the app store admins to manage app info and user requests. APP-store-UI is for users to search and choose apps and submit requests. APP-Menu-UI is for customization of the Stripes code so that apps can be enabled and disabled in real time.
- Business Logic
    - Definitions
        - Role: Collection of permissions. The permission groups in folio lack the features we need. When we enable an app, we should grant permissions to certain users so that someone can manage the users for that??? For example, in the cataloging app, there should be at least three roles - the app admin, the cataloger, and the receiver. When it's realized, all the three roles should be granted to the admin of this tenant. And this admin can give these roles to the users of the library.
        - Module: OKAPI module + basic Module. Currently in folio, we don't manage module information with modules. We only store modules in Okapi. But when we set up an app, we should have module information. The info from Okapi is not enough. So we set up Module-storage to store information, extend it and add features such as is this module for a tenant? Does this module have dependency on other modules? And more.

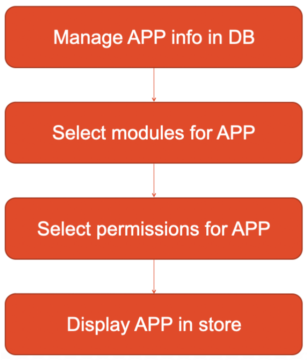
        - APP: Collection of mouldes / roles. Folio hasn't clearly defined an APP yet. We define apps as a collection of modules and roles. So the way we set up an app is to select some modules and some permissions to form the roles. Then an app is created. A user can request an app. The app admin can enable the app for the user.
    - APP store workflow



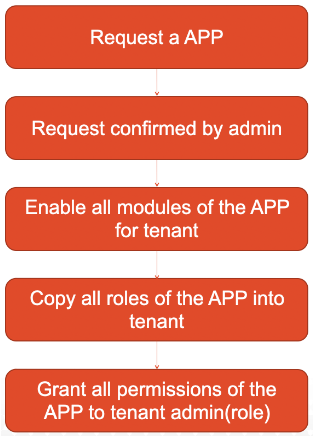
    - Manage modules



    - Manage APPs



    - Manage APP request


Request a APP
Request confirmed by admin
Enable all modules of the APP for tenant
Copy all roles of the APP into tenant
Grant all permissions of the APP to tenant admin(role)

4. **Q & A**
   - **Mike**: What purposes does the "role" have? How is the role different from the high level permission sets, given that some roles have existed in folio and have been defined in that way? What roles in the app store go beyond that?
     Jiang Sha: We have permission sets to collect permission. But when we initialize an app, we should have an indicator. We should know who is the admin of that tenant so that they can get all the permissions of this app. The key reason is that when managing permissions, we should have someone to manage all the roles. We have the rules that you can only grant other people the permission you have. So we need someone to be the super admin. We can't tell whether someone is the super admin or other library staff. So we designed "role" to contain those features. We need more attributes, for example, the super admin indicator.
   - **Seb**: The proposal is well thought through. As to the roles, one concern that comes to my mind is that libraries are structured very differently. And the way libraries organize themselves can be different. It might be useful to think of ways to avoid tying libraries down to specific structures or organizational structures. Especially when we move away from the core apps and into the new apps that we want to see added, you can imagine confusions of the roles arising if every app passes opinions of the role associated with this type of app. I can certainly see the value of every app creating permission sets. I'm worried about the idea of creating too much complexity on the admin side. Maybe we can extend the permission sets to include the attributes associated with "roles"?
     Jiang Sha: Every app should have its own roles. At some libraries, there might be hundreds of roles.
     Seb: App could introduce permission sets or roles. Maybe that can be very high level attributes. Also associate with those permission sets will make it very easy to navigate them. Maybe every app can introduce a super user and a regular user, and several roles. So when an app is introduced, it can automatically assign certain privileges to a super user to be able to draw out regular usage, or you only give privileges to individuals.
   - **Vince**: The idea of having roles in folio is a long standing debate. I do think there are possibilities with the "role" model. By which I mean, the way the permissions work now, they are very much bottom up. They are defined by the applications, and the applications inject new permissions whenever they want. They can certainly be grouped in terms of permission sets, which help solve the issues that there are too many permissions to manage. Roles can be more than just permission sets. One way, which we don't do in folio, but it can be part of the model where we manage at higher level destruction. Also, roles can play a part in the structure. Imagine we have the app store here. And it's possible that one of the requirements for an app to join the app store is to "we expect the app to have the

following roles", the roles are defined by the folio set level of things. And then the application conforms to that. So it's a way of allowing patrons to insert themselves into an existing structure. There are differences across libraries. But we can have a rigid set of roles. The roles need to be definable and flexible. So we need a separate layer where we can do that sort of management. But I am not sure why the role is put in this proposal for an app store at such a high level. It's possible to implement the app store without the role. My question is: who is the intended user in this case? We all know the model of iPhone or other smart phones, where you can add apps to the phone. And it's the choice of the individual user how to do it. In folio, you don't want one librarian to install some apps and another librarian to install other apps. This should be done at a higher level. The app management needs to be done at the tenant level. Is that what is in your mind in this proposal?
Jiang Sha: Our proposed app store is not for mobile apps. Our app is a collection of folio modules and roles. In an app, we need to grant all the roles to the super admin of the library/tenant, who can give roles to others.
Vince: The super admin will install an app from the app store into folio. That app is not potentially available to all the librarians. But only individual librarians based on their role, will be granted access to that new app. The idea is that the decision to install the application is made by the super admin, the administrator of the library. It's potentially available to everybody. But only some people will be granted access based on their role. That's my understanding.
Jiang Sha: Correct.

- **Vince**: Do you envision your app store to be something that is in the context of a single installation? Or, is the app store something that is made available to a large number of folio installations? You have multiple universities. Each of them will have their own folio installation. Will they share a common app store? Or, is the app store a separate feature on each of their installations?
  Jiang Sha: For now, I think every folio installation should have its own app store.
  Vince: Do you see the need for a super store that provides apps to the individual stores?
  Jiang Sha: That's very challenging to manage the remote folio system.
- **Keven**: I think there's another possibility to implement the app store. Maybe we can have two levels. We can develop the core level, which is maintained by the Chinese folio community. The private companies may want their apps to join the app store. And there might be compatibility problems. We, as the community, can issue the specifications to describe the functional requirements, frontend, backend, etc.
  Seb: We've had some internal conversations about the app store ideas at Index Data. Our thoughts had been about how to package and distribute apps. So the

different operators in folio, whether you are libraries, consortium, service providers, might easily download apps and install them, making them available without being part of the flower release. We haven't really thought about this in the context of your work. But it feels to me if we think about a super app store, in my mind, that might be potentially sth not going to happen by itself in the repository or some place, maybe on Github, where an operator of folio can easily download a set of files that correspond to an app and make them part of the local folio operation. And have some models for trusts. Once they are part of that, any tenant admin, a library super admin, can activate that particular app for that library. Maybe that could be a way to realize the two level app store idea.
Vince: That touches a couple of things that you mentioned earlier - trust and testing. One important aspect of any app store, in particular third party apps provided, there needs to be a measure of validation or certification. There needs to be a process that can help establish that the apps won't break the system. We need to ensure a happy secure environment that doesn't introduce bad practices or security vulnerabilities. We need to ensure it has the level of support it needs. So there is a mechanism that really needs to be established. And for the more established trust, meaning that if an app has been approved and has been reviewed by some group or community based selection committee, it has now been blessed and is valid to be installed in a particular version of folio. It's more than a Github or repository. We need a structure around the super store or app store. One of the ideas I also heard from China was there were multiple systems of folio which could communicate with each other. So it's possible that this super app store could be a dedicated specialized version of folio, whose purpose is to provide apps to other folio systems. And it inherits the same role models or whatever you put in place for the systems.
Keven: That's what we thought to do. We plan to have some mechanism to test the modules from the private companies and to certificate. We do that for the libraries and for them to choose partners. It's the service from our community to the libraries and the help to the companies that can help them find the market.
Mike: The danger here is we could weaned out with certification purchase with heavy weight that we have for the moment for flower releases …I wonder whether the more distributed goal is have every packet signed by the organization that created the app or it's up to the individual customer to figure out who they trust, who the organization is, and what the organization's procedures are.
Keven: We have a group of companies to help us implement folio. We have competing companies. We just want to give help to our colleagues, our library partners. They need us.
Marc:

    - One level is not sufficient for the discrepency. We need at least three levels of app stores.
    - Some folks are talking about a peer-to-peer trust based model. That doesn’t require an app store to exist. A hosting provider can talk to the app provider directly. An app store is an intermediary of trust. We need to first decide what trust model we need before we dive into the complicated technical issues.

  Keven: It’s up to the libraries or the user. If a library wants to partner with a vendor, it’s ok. They can choose to use all the local modules if they can pay the cost. We just serve the common needs.

  Tod: There’s the business of getting an app available to a tenant. That’s one problem. Once an app is available to a tenant, how do we make it easy for the library, for the tenant administrator, to make sure the correct people, the users have access to the ability to see and interact with the apps. It makes sense to me to think of the two levels separately. With this proposal, there is a potentially rich and challenging issue around getting the roles right. Providing a better environment for managing how apps are exposed to individual library users and maybe some additional benefits could come out of that. And maybe the issue of how those apps become available into a tenant and could be treated as a separable issue.
- **Marc**: This meeting specifically asked for technical feedback, so I'm going to give you my two biggest pieces of the technical feedback on this:
    - No.1, do we have to do with the dynamic loading of the dynamic enabling of a module and the deployment of it? First, any generalized app store are not relying on using Okapi’s internal deployment mechanisms for deploying the module because each hosting provider, each organization, each operating library that runs a folio implementation now can choose it because currently we don't constrain it to use a variety of different mechanisms to manage deployment and all of those kinds of things. That means that a general app store, the finishing, cannot rely on necessarily using docker images, or it cannot certainly rely on those docker images being spun up by Okapi because that might be totally inappropriate for that hosting provider's infrastructure. So there are significant challenges to address with it. Once we've figured out what the files are, what the bundle of stuff is that defines an app, and it's been sent in from that you've pulled it from the app store, actually manifesting that in the system, there's some significant technical challenges to do to actually achieve that in a way that would work that isn't couple to a specific hosting providers, like infrastructure, because if you do that, then it's not really a general app store, it's just something a hosting provider did.

    - And the same can be said about the front end. It's nice to say that we might have a way of automatically granting the roles and that permission sets all that stuff like that, yes, that's not that hard. And what might be harder, is the UI module. Something has to take that thing and rebuild the bundle for that tenant to redefine it to include that module in it, because without that, granting permissions to somebody to say "you have access to this new wonderful module" won't do anything about it, or will do anything at worst, it will fail because actually the app itself isn't in the UI that user is using. And every user will likely have to rebuild, have to reload the UI in order to get access to that. But as far as I'm aware, we don't have the ability to hard reload every user currently running inside folio/inside the browser with new UI modules. We currently distribute/host a static bundle that has to be rebuilt.
    - So specific technical feedback: the actual mechanisms for deploying or putting these things into the infrastructure are hard to do right now. And if you want to build this, those are the two major technical hurdles I would be trying to point out.

Sha Jiang: The deployment is not included in our proposal, we should prepare the deployment before the activation of apps and by the UI. So we said we should have a combination of Stripes Core, so that we can activate apps and the bundle.
Marc: If you take deployment out of the scope of this, there's a challenge there, which is that effectively somebody can choose something in the app store, and then it's basically up to the hosting provider to figure out how to actually make that happen for real. It's very valuable for a hosting provider to build. I would imagine I'm sort of surprised that hosting providers haven't already started to build it. But that's not a general app store. That's an app store that's dependent upon an individual hosting provider. So as long as we understand that distinction, I think it's not the important thing.

Chat history:

00:46:58 Mike Taylor: Tiewei Liu, you have my sympathy!
00:50:46 Ian Ibbotson: harry :)
00:50:53 Ian Ibbotson: then pick someone
00:56:20 Ian Ibbotson: So sorry Maccabee!
00:57:01 Tod Olson: I should have added that at UChicago we have a long history of programming around the edges of the library system, to extend functionality or integrate with other systems.
01:09:50 Tod Olson: Vince, we are hearing a lot of echo in your sound.

01:11:38	Marc Johnson:	The belief that itâ€™s easy to extend is based upon the assumption that those apps arenâ€™t themselves inter-dependent and that the core doesnâ€™t end up caring about them
01:16:37	Ian Ibbotson:	Sounds a lot like 2 live mics
01:16:39	Peter Murray:	I've never seen Zoom do that before...
01:29:09	Peter Murray:	Question: is the missing role functionality (functionality that isn't in the existing "permission set" mechanism) directly related to the app store proposal, or is the missing functionality a broader problem in FOLIO?
01:31:16	Sebastian Hammer:	I wonder if roles could be implemented as an attribute (flag) attached to permission sets, so we refine the existing permission model rather than create a new one. So some permission sets correspond to roles and others do not.
01:31:19	Tod Olson:	Peter: speaking personally, I think roles address a broader issue, and would have benefits beyond this proposal.
01:31:48	Peter Murray:	Thank you, Seb and Tod.
01:32:24	Marc Johnson:	What would the purpose of the flag be?

A role is only a collection of permissions (to the system at least)
01:32:33	Mike Taylor:	I believe there are already high-level permissions sets in the standard FOLIO configuration that implement roles, e.g. cataloguer
01:33:13	Vince Bareau:	A role can be more than just a set of permissions.
01:33:18	Ian Ibbotson:	+1 ^^
01:34:44	Marc Johnson:	What is the difference between a role and a set of permissions?
01:39:13	Mike Taylor:	Apps include modules, and modules can (and do) define permission sets.
01:39:34	Marc Johnson:	Modules can also be part of more than one app ðŸ˜ƒ
01:39:57	Marc Johnson:	Based upon the idea that apps are the set of all the things they need
01:40:11	Mike Taylor:	Each module, by convention, defined a permission `module.NAME.enabled`, which is what you give to a user who you want to have access to the app.
01:40:25	Marc Johnson:	Only the UI modules do that
01:40:26	Mike Taylor:	I think everything that this proposal wants roles to be able to do, permissions can and do already cover.
01:40:44	Mike Taylor:	@marc Yes -- but only UI modules are the face of apps.
01:40:56	Marc Johnson:	The gap seems to be associating semantic meaning to a role
01:41:11	Mike Taylor:	So ... associate a semantic meaning to the role?
01:42:50	Marc Johnson:	Yeah, I get the sense that folks think of roles having semantic meaning yet permission sets do not
01:43:29	Marc Johnson:	I think Vinceâ€™s point might be more important.

That a role can be app context independent and mean different things in each app

01:44:00 Ian Ibbotson: Right RBAC is very well understood as a general mechanism in computing at the moment - FOLIO is slightly off the beaten path here
01:45:49 Marc Johnson: It is kinda inverting the model.

Roles are defined and then an app decides what access that permits.

As opposed to an app defining granular chunks of access and folks combining them together
01:48:18 Mike Taylor: Thanks, Marc. That approach is of course perfectly well implementable in terms of FOLIO's fine and flexible permission mechanism.
01:50:06 Marc Johnson: Implementing the inversion of pre-defined roles having different meaning in apps using permission sets could be challenging

It would likely require a shared module that all of the apps that want to use a role would depend upon
01:55:26 Mike Taylor: Marc, yes, that sounds right. Why is that challenging?
01:57:30 Marc Johnson: In part itâ€™s challenging because it increases inter-module dependencies

A fundamental notion of FOLIO is that modules depend upon interfaces.

Permissions are not part of interfaces. Thus these would be concrete implementation dependencies on implementations not interfaces.
02:00:36 Mike Taylor: Actually, whether permisions are part of interfaces is an open question, since there is at present no separate machine-readable representation of an interface. See discussion at https://github.com/folio-org/okapi/blob/master/doc/proposal-to-separate-interfaces.md#difficulties
02:01:05 Sebastian Hammer: Good observation Marc
02:01:34 Mike Taylor: Nothing stops us from stating that (for example) mod-roles does nothing but define some high-level permissions, and that those permissions *are* its interface.
02:04:55 Maccabee Levine: Two logistics questions since the meeting is almost over. Will the proposal be shared to the Slack channel? And what are the next steps for discussion?
02:07:23 Marc Johnson: To me, interfaces today in FOLIO are manifested as RAML or OpenAPI docs alone.

I understand folks donâ€™t agree with this model.
02:09:23 Ian Ibbotson: Apologies all - have to leave for next meeting. Will keep track on slack - thanks for an interesting chat.
02:10:21 Mike Taylor: Marc's point about Okapi's depoyment facilities is right on target and very important.

02:10:37	Peter Murray:	Thank you for the discussion, everyone. The recording of the meeting will be in a directory at https://recordings.openlibraryfoundation.org/folio/ â€" I need to disconnect to get ready for the product council meeting.
02:11:03	Harry:	I need to leave. Thank you for this presentation and discussion.
02:11:20	Keven Liu:	thank you all
02:12:57	Sebastian Hammer:	Thank you for tackling this incredibly important but also challenging area straight on. If we can make this work it will be a huge benefit for the community
02:13:31	Keven Liu:	We will come back to you when we got problems
02:14:11	Maccabee Levine:	Thank you for all of this helpful work!
02:14:23	Tod Olson:	Thank you everyone!
02:14:55	xu chi:	thank you everyone