2024.02.23 Meeting notes

Date
23 Feb 2024
Time: 15:00 (CET)
Meeting link https://tib-eu.webex.com/meet/georgy.litvinov

Attendees
- Georgy Litvinov
- Brian Lowe
- Mark Vanin
- Miloš Popovi
- Ivan Mrsulja

Agenda
- Meeting audio will be recorded, audio will be transcribed to create will be transcribed to create meeting notes
- Meeting schedule: meet every other Friday at 16:00 (GMT +1)
- Current Issues:
  - Synchronization with latest VIVO code base
  - ABAC
    - Current Dynamic API access control is independent from VIVO access control (ABAC).
    - How are we going to align Dynamic API access rules with ABAC policies?
  - When to check access? On endpoint access/procedure invocation/component execution?
  - Checkstyle validation
  - Dependency updates
  - N3 Template substitutions with array of values
    - Draft PR https://github.com/vivo-project/Vitro/pull/426
    - Blocked by the issue related to ConfigurationRDFParser
  - ConfigurationRDFParser
    - Concrete class determination
    - Binding Array parameter type to the java class
    - Array parameter type is a subclass of parameter type
  - Dynamic data isolation
    - Data types
      - Ontology isolation
      - Ontology to class bindings
      - Default set of components and parameters
    - Graph organization
      - Load graphs from directories
        - Current directory types
          - everytime
          - firsttime
          - filegraph
        - Configuration:
          - graph uri from directory
          - graph relation to dynamic api
          - ro/rw mode
          - unions
          - triple store name?
  - Search components aren't ready
    - Removal of deprecated array implementation from N3Template and SolrQuery operations.
    - Lack of tests with procedures using search components
  - Http client component is not ready
    - Cache implementation for expensive queries (spec)
      - Ontology
        - Cache types
          - In memory
          - On disk
          - On cloud
      - Invalidation
        - Time based
        - By cache group names
  - Report generators
    - UI

Meeting notes (transcribed automatically)
Georgy: Hi Brian.

Brian: Hi Georgy.

Georgy: Hi Milos.

Milos: Hello.

Georgy: Let's maybe wait a couple of minutes maybe. Do you know if Ivan will be able to connect today? No?

Milos: I think he will come.

Ivan: Hello.

Georgy: Hi Ivan.

Ivan: Dragan, he will not come to this meeting. He is on vacation, so he is not available this week.

Georgy: Oh, I see. Okay, so let's start. So there was no meeting for quite a while, so let's get to planning first and maybe we'll get to some specific discussion during that.

So first, just to remind you that this meeting will be audio recorded. I will create transcribed meeting notes for two previous meetings too, because my transcription didn't work for some reason and I have to spend some time with it. Also, I plan to host this meeting at this time at 4 p.m. in Berlin, in Belgrade. I think 10 a.m. for Brian, right?

Brian: Sorry, say that again.

Georgy: Is it 10 a.m. for you right now? Yes?

Brian: Yes, it is.

Georgy: And it's comfortable time for all of you, right?

Milos: Yes.

Georgy: Okay, because Dragan asked me to move that because at an hour ago he has some lectures, I suppose, right? So, yeah, let's get with our issues that we have. Let me share my screen.

I think it makes sense to list these issues on each time we meet so we could see an overview each time and hopefully this list will be shorter and hopefully we will finish all of it.

Let's go through the big topic. So we have issues, we need to synchronize our Dynamic API codebase with Vivo main codebase because in Vivo check style validation was applied, dependencies were updated and also a back was merged.

So and as you know in Dynamic API we have our own small configuration properties to define who should be allowed to access endpoints and who shouldn't. Also we have this issue with entry template substitution with array values.

And it's related to configuration rdf parser issue that we will discuss today. Also, we need to take care, I suppose, about the dynamic data isolation, so isolation of procedures, and we'll get into that a little bit later.

Also, we need to take a look at search components because Solr query is not ready, I would say. There is a lack of tests with procedures for that.

Also we planned a while ago to create HTTP client that could be very useful especially for lookups for some external database to get some data or maybe pull some data.

Yeah, also there was an idea when Dynamic API was started and defined to have a cache implementation to be able to cache expensive queries and hopefully we'll find time at some point to do that and that would allow us to resolve some performance issues that we have in Vivo and specifically for Dynamic API.

Okay, so any other topics you would like to discuss today maybe? Or is that okay for your agenda?

Ivan: Just one question. Are we going through the agenda again slowly or can I ask any question that comes to mind? Because I have read the caching.

Georgy: Yeah, if you would like to discuss something let's discuss it and then we go through other items in the agenda as you like. So you have some questions.

Ivan: I just wondered if you'd like to implement caching by using Redis or some other Redis alternative or we are making in-house solution.

Georgy: So there is a JSR 107 specification in Java that allows you to define the logic how you're going to work with cache in general and there are a lot of implementations like caffeine for example something else.

So you can use different types of caching: in memory caching, on disk caching, caching on the cloud. So when it's something like redis memcache d2 is an option.

So, a lot of opportunities here, but it's not easy because once we implement the specification, of course, you would need to provide some kind of implementation.

What implementation that would be maybe depends on what we decide in Vivo, or maybe we should provide some options how to do that. But I think if we get to that, we need to be able to specify configurations as flexible as possible.

So, I think optimal solution would be to store some configuration in triple store in specific graph and define it like that. But the problem is, so when we store something not in Java, not in memory of the same Java servlet, we need to serialize and deserialize data.

And hopefully that's not a big overhead for sparkle query results, but maybe it is.

We would need to be able to decide at what point it makes sense to cache some request and at what point it's not, because for example if you use something like some cloud-based solution and your latency time is greater than 40 milliseconds but your sparkle query takes 10 milliseconds so it doesn't matter.

So it's better to do the sparkle query and it might be not only the sparkle queries, because sometimes you might want to split your task into multiple sparkle queries, right?

But you don't want to just do the same thing and take everything from the cache, you would like to return the data to the user that was accumulated, right?

So accumulated data for some requests for maybe some procedure response. And invalidation is always...

Ivan: So what L1 and L2 caching if I understand correctly?
Georgy: Sorry?

Ivan: So we would need both L1 and L2 caching. L2 caching for the requests that go to queries that go to front end to back end. And L1 cache inside one, let's say, transaction, but inside one procedure.

Georgy: Yeah, maybe. Yeah, most likely. But we'll see. We need to define that and most likely define an ontology for that. How are we going to configure that? How are we going to store that?

Ivan: Well, I'm not sure what he was talking about. Should we granulate it to more specific policies, some kind of it? Not just to allow component, but to allow some specific query to execute.

Georgy: I think we need to take care of that problem. Yeah, so we go further, okay? So, about synchronization with the latest Vivo codebase. As I said, we updated, we have the ABEC now in Vivo.

And, yeah, we need to decide. So, now we have these very nice options to define what user role should be able to access and point, but that basically duplicates the authorization system that we have in Vivo.

And if you have an idea, is it okay to leave it that way, to have a duplicated authorization?

Ivan: There are some pros and cons for that, because in case there is something wrong with the authorization, this will still work, because it's defined in the definition of the procedures and endpoints. Procedures, as I remember.

And you're also able to move that to share the procedures between instances and it comes with authorization configuration for each of the procedures, but at the same time, I don't know if that would be okay for Vivo to have that kind of duplicated authorization.

Brian: What do you think about it?

Brian: My natural inclination would be not to have duplicated things. It sounds odd, but maybe there would be a justification for it. I don't know enough about it yet, I don't think.

Georgy: Yeah, my inclination too. That's why I brought this topic today. And if we agree about that, so I thought that maybe we need to have some procedures, not procedures, sorry, policies in our access control system and maybe these policies could be universal.

So we don't have to save the URI of endpoint each time. We could just create some some labels or marks and then reuse it in a dynamic API.

Ivan: Would that be okay?

So how to decouple them? So how to make them less coupled, I mean, procedures and the policies that we store in our instances. So what do you think?

Georgy: Could we create that kind of universal policy that would have, for example, a list of roles and maybe some labels that you can define in dynamic API procedure?

Ivan: And then by this label defined in dynamic API procedure, you would gain access or you would deny access as a user?

Georgy: Kind of. So we have these policies in Attribute-based Access Control. And in these policies, what policy do, they check some attributes of the request. So in the request you can provide something.

What are you asking to access to and who you are and what are some other environment variables. And, I suggest to, in this policy, define a list or predefined list of that kind of environment condition.

So, that kind of condition could be the groups of users, for example, right? That could be something else. So groups of users is the first thing I think about.

And we could just provide the same data into this, as an environment variable, into the policy. So then technically the attribute-based access control will be used and will be in charge of how this request is checked.

Georgy: So the current condition why it seems not very clear, because now we just don't go through the standard pipeline of checking access control. We're doing that ourselves.

So my idea is to take the same information, provide it to the access control system, and define access control system, so it would know by these conditions should it allow or not allow. Is it better?

Ivan: I know about one solution. I've heard about it in the past. It's the XML standard for Java for these attribute-based access controls. It's called, I don't know how to pronounce, but I'll write it in the chat.

Something like this. Let me just find it on the web. It's like an XML notation to write a policy configuration and then it can be parsed by Java.

Georgy: No, no, we already have policies and we already have definition of how we define the policy and provide the data.

So I think we won't be able to reuse some XML standard. We just need to have, maybe you remember, you're a reviewer of that pull request, we just need to define an entry file for the access policy and that would be it.

Ivan: So you want to save the whole policy to a label and then add that label as a name of the policy in the definition of the procedure, if I understood correctly.

Georgy: Yeah, we have these policies in Attribute-based Access Control. And in these policies, what policy do, they check some attributes of the request. So in the request you can provide something.

What are you asking to access to and who you are and what are some other environment variables. And, I suggest to, in this policy, define a list or predefined list of that kind of environment condition.

So, that kind of condition could be the groups of users, for example, right? That could be something else. So groups of users is the first thing I think about.

And we could just provide the same data into this, as an environment variable, into the policy. So then technically the attribute-based access control will be used and will be in charge of how this request is checked.

So my idea is to take the same information, provide it to the access control system, and define access control system, so it would know by these conditions should it allow or not allow. Is it better?

Ivan: I know about one solution. I've heard about it in the past. It's the XML standard for Java for these attribute-based access controls. It's called, I don't know how to pronounce, but I'll write it in the chat.

Something like this. Let me just find it on the web. It's like an XML notation to write a policy configuration and then it can be parsed by Java.

Georgy: No, no, we already have policies and we already have definition of how we define the policy and provide the data.

Ivan: I know about one solution. I've heard about it in the past. It's the XML standard for Java for these attribute-based access controls. It's called, I don't know how to pronounce, but I'll write it in the chat.

Something like this. Let me just find it on the web. It's like an XML notation to write a policy configuration and then it can be parsed by Java.

Georgy: No, no, we already have policies and we already have definition of how we define the policy and provide the data.

Ivan: I know about one solution. I've heard about it in the past. It's the XML standard for Java for these attribute-based access controls. It's called, I don't know how to pronounce, but I'll write it in the chat.

Something like this. Let me just find it on the web. It's like an XML notation to write a policy configuration and then it can be parsed by Java.

Georgy: No, no, we already have policies and we already have definition of how we define the policy and provide the data.

Ivan: I know about one solution. I've heard about it in the past. It's the XML standard for Java for these attribute-based access controls. It's called, I don't know how to pronounce, but I'll write it in the chat.

Something like this. Let me just find it on the web. It's like an XML notation to write a policy configuration and then it can be parsed by Java.

Georgy: No, no, we already have policies and we already have definition of how we define the policy and provide the data.

Ivan: I know about one solution. I've heard about it in the past. It's the XML standard for Java for these attribute-based access controls. It's called, I don't know how to pronounce, but I'll write it in the chat.

Something like this. Let me just find it on the web. It's like an XML notation to write a policy configuration and then it can be parsed by Java.
Brian: I guess that sounds possible, yeah, so maybe it needs to be checked every time, I don't know.

nasty that he shouldn't be allowed to. Is that possible?

But then, go through the data, maybe creating some temporary graphs, you find out, some procedure at some point, find out that this user does something

So you make a request and you see that, okay, this user is generally should be allowed to do that.

The middle case is when we have an endpoint that does something and this endpoint consists of multiple procedure executions. And on each procedure,

Ivan: Well, that was to me a bit more reasonable because we could have some headache if you have some access to some components but not to one

and then can break the whole pipeline. Reasonable? Which one? Which of that?

To have just the endpoint, not every single component.

Georgy: Just the end point.

Ivan: Actually, here we have experience with AWS policies. Do you think that we can create something like that, where we have something, do we allow or
disallow this policy, this action, which action and which resource, where we can specify from the URI?

Can we maybe look at AVS and create something similar to it, or we need something more specific to our problem?

Georgy: I think we need specific because in this case we deal with triple stores and here a lot of different factors make sense, so the graph

name, are you going to work with the data itself?

So sometimes if it's a self-editor, for example, the most complicated case, when we work with self-editors, self-editors should be able to access his own

profile in a graph, but not other profiles, and decide which data should be accessible by the user.

So now in the policies, there are sparkle queries that ask, okay, this user tries to access these triples. Should we, are these triples connected to the profile

or is just someone else's triples?

So that's complicated. And the same, what's not done in Vivo, the problem is when we use the same self-editors and the self-editors use endpoints with

auto-completion and they see some labels of some types of objects.

Of course, I think search endpoints is used to get that. Search endpoints returns all of these objects, no matter if these objects should be accessible by this

user or shouldn't.

And yeah, that's complicated to make and especially in the search indexes. So that's why our search indexes, yeah.

Ivan: Can we maybe, what I was talking about earlier, like access control for me should be at the end point.

Because if we check every time we query the database, I think we can have major performance issues in the future when it's a lot of operations going one

after the other, and everyone is being checked.

Also, can we maybe update the query every time it comes to the backend with the additional some checks?

Well, if this person is in any relation to these triplets and what not, can we mutate the query a little bit so it can return the data we actually want?

Georgy: I'm not quite sure I got it. What do you mean? If you send the query, then you get a response, are you allowed or not to do that, right?

Ivan: Like, when you send the query, you'll say like, I want to read all the data, so it doesn't know if you have access to the data or not. But you can add

some maybe filters that will show you if you are maybe related to the data.

I'm not quite sure if that can be achieved in this scope.

Georgy: What kind of filters?

Ivan: Well, sometimes if you have triplets and you want to see if you're connected to them, you can maybe add a triplet. Do you have a relation to that?

Maybe?

Georgy: These relations are defined in policies. I don't think that users should be able to send that kind of sparkle queries, especially because sparkle

queries are very expensive to execute.

So if it's a huge graph with millions of triple, that could take a lot of time.

Ivan: Yeah, but not user to send them. User sends his sparkle query but you actually mutate that query with all that additional filters. I don't know if that's

possible.

Georgy: I think that's possible but that could be very complicated because you need to be able to parse sparkle queries and yeah, a lot of factors of attacks

I think are possible, unfortunately. Brian, what do you think? Now question for you.

The same question. What do you think? What should be the granularity of access control? Should access control be checked only once at the endpoint

entry or it should be checked on each execution of procedure?

So in case we have endpoint with multiple procedure that calls multiple procedures and maybe goes... So each time the procedure would be required to

check the access.

Brian: I don't know. It seems like it would be nice to know before, if something recursively calls other things, it would be nice to know before you start it

whether anything that it's going to do is unauthorized rather than sort of doing that.

On the fly? I don't know. Is that the question?

Georgy: Yes, yes, yes. But I thought, for example, if you have an endpoint that traverses some graphs and some data, and at first you don't know what's

going to be result of that.

So you make a request and you see that, okay, this user is generally should be allowed to do that.

But then, go through the data, maybe creating some temporary graphs, you find out, some procedure at some point, find out that this user does something

nasty that he shouldn't be allowed to. Is that possible?

Brian: I guess that sounds possible, yeah, so maybe it needs to be checked every time, I don't know.
Georgy: Okay, so something for you all to think about and we can continue discussing this next time. I myself am inclined to procedure checks. I don't know.

Because I think per component that would be too expensive to make multiple sparkle queries. Of course, in most cases, our attribute-based access control doesn't require you to make sparkle queries, but still there could be. So okay, let's go further.

So next point is we need to align dynamic API codebase to check style validation and update dependencies. I think I'll do that after we get release 1.15 when it's done. I think it would make sense to create additional branch and do everything there.

And that's basically all. I don't know if anything else was involved in that. So let's go to the next point.

So, about the problem with entry template substitution with array of values, I was trying today to replace the problem and I found that the problem appears because of, I think, my mistake with configuration in a dev parser and class inheritance that I created in Java classes and at the same time ontology of Dynamic API. So, specifically we have in this configuration a dev parser. One second, I'll open all of them.

So, we have here a check determine concrete class and what it does by the URI of the class it finds the Java implementations and Java classes. Basically, Java interfaces, Java abstract classes and just Java classes.

But there could be defined multiple implementations but only one not abstract and not interface.

So if the implementation implements a few interfaces there will be interfaces but if some class for example inherits from another class so this will not work and that's the case for me because there is a class called parameter type and I also create a subclass of it called array parameter type which allows to define type of objects that this array contain. Easy thing would be just to merge that into one class and make a mess.

But I think, maybe I would be able to resolve the issue. So the issue is now that in ontology, we have these, let me show it one more. We have this array parameter type, and this is a subclass of parameter type.

So both of them have their own implementation defined here. So there is implementation here, parameter type, and for array parameter type. So, important question I think for Brian mostly.

Can I filter if I have a class with implementation for example of array parameter type and it's a subclass of parameter type.

I want to filter out everything that's not more specific types, and if there is only one more specific type, then use this implementation. Would that be okay? And did I explain it good enough?

Brian: Yeah, I don't quite follow.

Georgy: Ok.

Brian: You have most specific type, and you have all of the types, and you want to use one or the other? The answer, I guess, would be yes. Use whichever one is appropriate. I don't know.

Georgy: Yeah, you know how this configuration rdf parser works, right?

Brian: Vaguely.

Georgy: Yes, so by the URI of the class it finds the Java URI, so Java class, but if it finds more than one Java class, then it confuses and throws an error.

And my idea is just to try to filter out it here somehow, so not to throw an error.

In case we have two classes, two implementations, and just select the most specific implementation here. The problem is in Java I can create a class and I can make a subclass of that. So for example this parameter type and array parameter type.

Array parameter type is a subclass and parameter type is not an abstract class so it's used. But this implementation of configuration that the parser doesn't like when it finds two real implementations.

But in ontology, at the same time, we are able also, we are able to define subclasses and use it as we would like it in ontology, so for ontology purposes.

So, yeah, I think. Ask questions.

Brian: I still don't understand why it would be, so when the parser is running, why would find two, aren't you specifying the class in the RDF that you wanted to instantiate? Why would it find two?

Georgy: Okay, so I think you can see on this page, so there is a definition of array parameter type, and it's a subclass of parameter type, right? And I use array parameter type, so this file is loaded with it, so this one.

And don't look at this one. It shouldn't be here. I removed that. But basically what it does, it infers that array parameter type is subclass of this types parameter type. Types is here. So it's Java and path to the Java implementation class.

So it's a subclass of two Java implementations. Because of inheritance. So this array parameter type and parameter type. One has one Java implementation, another has another Java implementation. But this one is also subclass of this one.

Brian: Right.

Georgy: Yeah. I don't want to have an abstract father ascendat, I want to still be able to use these Java implementations more freely, I would say, right?

So can I just distinguish, so if I have for this class, this RDF parser finds me these two Java implementation, can I filter out this one?

Brian: But what I still don't understand is why the, I mean, it seems like the actual RDF instance data that it's parsing and then populating the Java classes with. I mean, shouldn't that RDF only be using concrete classes, not abstract classes?

And I don't really know why. It seems like if you're not specifying an instantiable class, it should just throw an error, it seems to me, shouldn't it?

Georgy: Yes, but for some cases, in practice, I found that I need to define, so for example, this one, so there is a component serialization, primitive serialization type, and one of that is an abstract class, and I had to provide that there is an abstract class.

Without it, it throws an error. So for interfaces and abstract classes, I have to define that. But in case both of that, so my father of that class, right? And the class are both real Java implementations.

I get an error because it cannot distinguish what the Java implementation should be. Should it be this one or should it be this one? And I think that's the limitation of this configuration RDF parser.

Ivan: Is there a way maybe to annotate classes like primary like you have in Spring Boot?

Georgy: No, it's homemade.
Ivan: Maybe you can incorporate some library that helps you with that. I don't know. Maybe we can do a little bit of research and see if there's something that can help you.

Georgy: This thing traverses the graph and I don't think, you know, the previous idea was to avoid using anything else. It's not a problem that it's instantiate new things. There is a problem that it should specifically understand the graph that it uses to load Java instances. So it should know what Java instance to use to load some object.

Ivan: But maybe you can create something similar with reflection.

Georgy: Let's say it uses reflection.

Ivan: It uses reflection, yeah. Maybe you can make an annotation, like primary or something like that, and annotate parameter type or array parameter type. And then if you found a parameter type, you check for the existence of that annotation. And if the annotation is existing, you take that implementation. Maybe that is the solution. You're?

Georgy: No, not really, because both implementations are used, but used by different ontology instances based on what type these ontology instances have, individuals that define an ontology, so there are specific annotations used by this configuration RDF parser and configuration bin loader, the whole thing called configuration bin loader. There are a few annotations, homemade annotations created by this and it works pretty well, but in this case I think if I'm at least allowed to make a differentiation here, then we would be able to use this freely. Because, for example, I can maybe show you the dynamic API, I can show you the components, I suppose, and where is it?

Dynamic API conditions operations. No, not really one second. So that's the implementation. So this array parameter type just has this annotation that is used by configuration bin loader and it just allows to set type of the items that are stored in this array.

And I don't think it should be moved to this parameter type class, so that's the parameter type class and it's used mostly for most of the parameters that are not containers.

Ivan: One thing that comes to my mind, maybe not the most beautiful thing in the world, you can probably make that parameter type class abstract, and then have one class that is the implementation for the regular ones, and you have the implementation of the array parameter types.

Georgy: Yeah, I know. I thought about it but you know that's a kind of limitation, so each time we would like to model something in Java we are constrained by this.

And you can find this at the stage when you when for example some tests are working because you just don't use this configuration bin loader. So that's that's where it fails. It fails on configuration reloader when I create instances of the same parameter types that tests are passing. So I don't really want to do that thing.

Instead, I would like to do the opposite. I would like to be able to make more subclasses because already I had to maybe a little bit over-complicate the classes because of this limitation. And the code in Java will get worse and worse because of that, I think if it would be possible to filter that at this point, then we wouldn't see this type of errors, I suppose, at all. And as you know, configuration-bin-loader errors, I know if you have seen them, they are quite complicated.

So I think I'll try to filter this out. I'm not sure how. Maybe I'll just find some subclasses of this and try to remove that from this list. We'll see how it goes.

OK. So next thing, do we still have time? Okay.

We have a topic about dynamic data isolation and that's a thing also to think about. For example, we have different data.

We have, for example, ontology in our Dynamic API that we don't really use it right now because now we don't have an interface because now we don't have a web interface to use it. We just defined it to assign components and bind the components to Java implementation and for documentation purposes. But still that could evolve during time and I'm not sure how and where it should be placed for this data isolation.

Because this should be available for the user to work with web interfaces, but I'm not sure if it would be needed for the endpoints to be loaded with configuration binary loader, because configuration binary loader also uses some of that data.

Also this file, I don't know, maybe one second I'll move this. So this file, dynamic API implementation, I don't know if you have seen it before. So this file basically binds the ontology instances with some Java implementations.

And it should be. It's required for the loading of the data, so if we are loading some procedure in isolated graph, then the bindings for that procedure are required to be there.

And one way would be to have a shared bindings between all of the graphs and load all of them with the same ontology and with the same bindings to the Java implementation.

But I thought, what if we get to the point when we update some instances and ontology has changed or binding has changed? Or we need to find out if bindings on the system present in the data that we are going to import as a new functionality.

So the user would know that it just wouldn't work, so we can fail fast for that reason. And the default set of components and parameters, as we have, I think, in the folder.

Let me get there. In this Dynamic API A-Box folder, we have also default parameters type, model parameters, HTTP methods, serialization type, user groups, validators. So all that it might be used by some endpoints and required for this endpoint and this might change and this should be somehow be I think captured by what needs to be loaded for this graph and isolated in the graph.

So that's all things I think we should consider for that. Any points on that? Okay, then let's go further. So for the graph organization, as we talk on previous meetings, we need to define a graph URI from directory that currently is not existing.

So now we have, as far as I know, we have some hardcoded in the Java class, we have hardcoded names. That links to the graph names in specific triple stores and after that directory names like for example, let me show it here.

So here we have auth directory, display directory, display display directory, display tbox, dynamic API box and ttn. So all of that have a specific meaning and ttn, for example, it used to add data to the same name of graphs.

So you will see for each language here the name of the graphs, so display interface and things like that. And to introduce something new, we need to first acknowledge what's existing.

So we have this types of directories called everytime which as I remember it's something called that it loads into the memory each time Viva starts and it can't be edited as far as I remember, right?

Also we have a first time, first time as far as I remember, it's when you load it in real graph for the first time and then on each startup you compare the data in these files with the data that resides in your triple store.

I'm not sure if the first time have different behavior between triple store types, so for content and configuration, does it work the same?

Brian: I believe so.

Georgy: File graph, I'm not sure at this point what it means, so I think it also laws the data, but Brian, do you remember how filegraph works?
Brian: Yeah, filegraph just derives a graph URI from the name of the file, and then it just compares the current state of that file to what's in that particular named graph, and then makes it so that the named graph matches what's in the file at startup there.

Georgy: Oh, so we already have then somehow graphs defined in the name of the file. Let me find some example. Maybe tbox. Yeah, file graph in tbox. But how it finds the name of the graph here for the vitro and vitro public, do you know?

Brian: So it uses some, I forget whether it uses the default namespace for the application or whether there's a file graph specific default namespace. I don't recall off the top of my head, but it just takes that file name. I think it chops off the extension and then just appends it to that namespace. It seems like it must be a file graph specific namespace, so it doesn't collide with whatever else you might be putting in your main namespace. I can't remember if that namespace is derived from your default namespace, or whether it's just one universal one. It's been many, many years since I've thought about it. But yeah, it takes that file name and appends it to something to make a URI, and then just uses that as the URI of the named graph that it synchronizes to.

Georgy: Thank you. Thank you, and I made a list of some configuration that we might need for the graph isolation, so we would need to have a GraphViewRise, and maybe that could be done the same way. We also need to have a graph relation, maybe, to be able to define that this graph is related to Dynamic API and some other graph loaded the same way, but not related to Dynamic API. We also might need to define somehow is it read-only or can be modified after it's loaded. Also, I'm not sure that there is any possibility to define right now the unions because all I have seen was in Java, so unions defined in Java. And also, yeah, we don't have an option to define triple store name, so the Java code that loads this data already knows where the graph should be. Should it be content triple store? Should it be configuration triple store? So, I don't know. Ideally, I think we should have some specific graph with that kind of configuration that would allow us to dynamically change that on the fly and not rely on Java code modification each time we would like to introduce something and have universal configuration. For example, now we're talking about the loading all these graph data from file system, but I suppose at some point somebody like we discussed on the Anthology meeting this week, it could be something like loaded from the internet where some ontologies are loaded and maybe stored in specific graphs and then compared to some previous versions of ontologies. So I think that kind of configuration and maybe there should be a specific ontology for that. I don't know if there exists already. And I told you to have more control over your graph and create, remove and work with graph dynamically. So Brian, what do you think? Do you have experience with something like that? For example, in this case, we might need to define more data about the graph. So is it related to what triple store it's related? Should it be read only or should it be modified? So some data that we might need to store in the triple store.

Brian: So this is metadata about named graphs?

Georgy: Yeah, I thought that right now we have this data in Java code. So in Java code we have names for the graphs and all of them loaded because they're put in configuration Java class and content. But what if we could define this dynamically? So define it in a special graph and load this data by these rules and don't have special cases for each case.

Brian: Yeah, that would be nice to have. Yeah, there isn't anything like that currently.

Georgy: Okay, so we're inventing something. Great. We will make a publication.

Brian: I can't say that somebody else hasn't already done that, but there hasn't been any attempt to do that so far in vivo.

Georgy: So, yeah, I think some research is needed anyway, but I think that would make it possible for dynamic API and at the same time improve state of the things in Vue in general. Okay, also at the end I just wanted to say that these Solr query operations are a little bit abandoned and should be updated and checked and covered with tests. Most likely with procedure tests where we loading something with configuration bin loader and see that it works and make some requests to that loaded procedures or endpoints. Also, one very nice thing would be to have HTTP client component to make lookups to some database like we have now lookups to some authorities, but all of that is a specific Java file and Yeah, sometimes we just need to get data, sometimes we need to pull some data somewhere. And that shouldn't be very complicated, I think. And yeah, there was only this cache implementation that we already discussed. And also, when Mark will join us, we'll see what's the progress with report generator UI. And yeah, that's all I prepared for today. Thank you very much. Do you have any questions? Do you want to discuss something else? How about your availability for the meetings? Is it okay to meet still on every other Friday? So once in two weeks on Friday at this time?

Ivan: Okay.

Milos: Okay.

Georgy: Good then I hope we will proceed in this way and if you find some free time and would like to work on something on that please let me know. That would be very appreciated and thank you very much and have a nice weekend.