Posted in: General (/spaces/127/sandpiper/forums/4997/general)

Reference Documentation for 0.8

Unsubscribe from this discussion
Subscribe to RSS (../../../../../../spaces-cf/forums/rss-space-posts.ashx?spaceID=127&topicID=5424&forumID=4997&key=rrer%2B1xDo%2BlxpC7jBRbM5w%3D%3D)

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
12/27/2019

Hello, All,

I have a very rough and incomplete document that I'm building to encapsulate 0.8 from a proposal standpoint. I'm attaching it here for your amusement and feedback as I continue to write.

Note that this is the technical document -- more user-friendly and higher-level material still needs to be written.

One thing I wrote, that we haven't discussed and I am not sure if it's correct, is that UUIDs should be of the RFC4122 version 4 form, which lays out how to generate a UUID from a random or pseudorandom number. It's here: https://tools.ietf.org/html/rfc4122#section-4.4 (https://tools.ietf.org/html/rfc4122#section-4.4) . I think we'll probably need to actually discuss it, in a separate post somewhere.

There's lots more to write but this also gives an overview of the data model, the interaction levels, and the implementation.. It needs visuals badly too, I have just the one that I put together for the data model.

Things are still unclear in many sections under the levels, I've only begun to hash out the different phases/approaches of each level. And I haven't gone into the add/delete functionality in Level 2 yet. I'll update this post with new versions.

📎 **Attachments**

👍 Like

Reply (/forums/post?tid=5424&ReplyPostID=5425&SpaceID=127)

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
12/28/2019

Naming things is really important, and I'm having trouble with the word "**grain**" to describe the data-object. It doesn't feel like something that has a payload. It feels more like something that can't be broken (a "grain" of sand, a "grain" of salt, etc.).

Does anyone like any of these better:

```
blob
capsule
carton
container
packet
```

Or we could make up some acronym like `sdo` (Sandpiper Data Object).

Any other suggestions???

This decision is time-sensitive, because I'm at a point where I need to implement it.

grain (/spaces/127/sandpiper/searchresults?keyword=%22grain%22&searchtags=1)

👍 Like

Reply (/forums/post?tid=5424&ReplyPostID=5426&SpaceID=127)   Answer

Luke Smith (https://autocare.communifire.com/people/autopartsource)
12/29/2019

I like "sdo" because It has no baggage from other areas of computer science.

👍 Like

Reply (/forums/post?tid=5424&ReplyPostID=5427&SpaceID=127)   Answer

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
12/29/2019

The problem with "sdo" is that it doesn't read well when you have plural ("sdos"?). If we go with a made up term, I might like "sdoc" (Sandpiper Data Object Container) better. The plural ("sdocs") reads better (and you can say it too!).

People with think we meant it to be a Sandpiper Document, but that's not all bad either.

naming (/spaces/127/sandpiper/searchresults?keyword=%22naming%22&searchtags=1)

Like

Reply (/forums/post?tid=5424&ReplyPostID=5428&SpaceID=127) Answer

Luke Smith (https://autocare.communifire.com/people/autopartsource)
12/29/2019

sdoc(s) is reasonable. If we feel that it needs to strongly convey the idea of an indivisible atomic piece of Content, I think we need to keep pondering.

Like

Reply (/forums/post?tid=5424&ReplyPostID=5429&SpaceID=127) Answer

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
12/29/2019

I'm going to use "grain" for the time being... Now I just need to resist the urge to create a "grain-silo" :)

grain (/spaces/127/sandpiper/searchresults?keyword=%22grain%22&searchtags=1)

Like

Reply (/forums/post?tid=5424&ReplyPostID=5430&SpaceID=127) Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
12/30/2019

I think you may be right that the idea of a grain does imply some atomicity. Perhaps we should move that way down in the hierarchy somewhere and, like you said, pick a name for this packet of one or more grains.

Could we use "Fragment" instead? That seems like a nice middle ground. Then the grain would replace what I've called the "Element" (which is way too general and used all over the XML space).

Like

Reply (/forums/post?tid=5424&ReplyPostID=5431&SpaceID=127)   Answer

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
12/30/2019

"Fragment" sounds (to me) like it's broken. :)

I personally like "packet" because it indicates a container of some sort. It's the lowest level thing that we exchange (sort of like a TCP packet). You could think of them like those little condiment packets that you tear open and pour into your morning coffee.

I was trying to think of corollaries to a packet in the "sand" space, but the two I came up with (sand-bucket and sand-bag) didn't sound too good to me.

I'm getting more accustomed to "grain", the more I use it.

Like

Reply (/forums/post?tid=5424&ReplyPostID=5432&SpaceID=127)   Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
12/30/2019

How about we take "Grain" then, as is, and I'll change the "Element" entry to be "Particle" -- indicating that it truly is a sub-atomic piece?

That will leave us with:

1. Node
2. Pool
3. Set

4. Section

5. Slice

6. Grain

7. Particle

Like

Reply (/forums/post?tid=5424&ReplyPostID=5433&SpaceID=127) Answer

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
12/30/2019

```
1: slice -> grain -> element
2: slice -> grain -> particle
3: slice -> packet -> grain
4: slice -> packet -> particle
5: slice -> packet -> element
```

Let's vote.

Like

Reply (/forums/post?tid=5424&ReplyPostID=5434&SpaceID=127) Answer

Luke Smith (https://autocare.communifire.com/people/autopartsource)
12/30/2019

I like #3 (slice > packet > grain). Although I'd be perfectly happy with slice > grain. I'm not convinced we need an intermediate level between the slice and the atomic particle.

Like

Reply (/forums/post?tid=5424&ReplyPostID=5435&SpaceID=127) Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
12/30/2019

I'd be OK with #3 I think. I like #2 linguistically but I think packet does describe the "window"ness of it well.

Like

Reply (/forums/post?tid=5424&ReplyPostID=5436&SpaceID=127) Answer

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
12/30/2019

The data-object really does seem to be the smallest piece we will be dealing with. There's no reason I can see for Sandpiper to crack it open once it has been created. This does point to just one level below a slice. For the time-being, I'm using "grain" for this purpose.

Like

Reply (/forums/post?tid=5424&ReplyPostID=5437&SpaceID=127) Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
12/31/2019

All right, it sounds like a vote for Slice-Grain-Particle then. I'll update the doc.

The particle is something I added looking forward to future needs, not current -- I think it's going to be useful as an abstraction when systems are reporting individual sections of records being updated in realtime (though still the grain will be the add/delete, but having common language to resolve discrepancies between data) when we get to Levels 3 & 4 someday.

However I should make that clear in the documentation, so it's not something implementations feel they need to implement now.

Like

Reply (/forums/post?tid=5424&ReplyPostID=5438&SpaceID=127) Answer

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
12/31/2019

> **Elements**
>
> An element is a specific piece of product data within the grain – a single fitment record, for example, or a single attribute.

I think what Luke and I are saying is that we don't believe Sandpiper is responsible (or knows about) anything within the grain. It is a black-box payload that is delivered with some metadata, but without the ability to parse it further.

In Level 2, Sandpiper will be delivering these Elements (single fitment records, for example) as the grain's payload.

payload (/spaces/127/sandpiper/searchresults?keyword=%22payload%22&searchtags=1)

Like

Reply (/forums/post?tid=5424&ReplyPostID=5439&SpaceID=127) Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
12/31/2019

> **On 12/31/2019 10:27 AM, Doug Winsby said:**
>
> I think what Luke and I are saying is that we don't believe Sandpiper is responsible (or knows about) anything within the grain. It is a black-box payload that is delivered with some metadata, but without the ability to parse it further.

It is a good point and one I agree with. I'm thinking of the particle as an as-yet-unimplementable future reference that two nodes can use as a pointer into the grain, that non-Sandpiper systems and humans can react. This would be key to the feedback mechanisms for example.

So in Level 3 the receiving PIM could say "Hey, this data you sent me, I've processed it but I have a problem with this piece, particularly the drivetype", and the Sandpiper node passes that back to the sender saying, "Get back to me." The node can't understand the particle, only the grain, but it can help systems that *do* understand it to talk about it.

Yet to preserve reproducibility this would have to be limited to non-action communication -- "Delete this please" would not be allowed for example. So since this is too deep to go for what is meant as a stub, perhaps it makes sense to remove the reference and keep it in our back pockets for now.

Like

Reply (/forums/post?tid=5424&ReplyPostID=5442&SpaceID=127) Answer

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
12/31/2019

I see what you are saying.

Bringing up "Feedback" is important. We've really not addressed that yet. We should start another discussion on that topic. It's a big one.

feedback (/spaces/127/sandpiper/searchresults?keyword=%22feedback%22&searchtags=1)

Like

Reply (/forums/post?tid=5424&ReplyPostID=5443&SpaceID=127) Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
12/31/2019

I've revised the document and made these changes:

- Removed discussion of Element below grain, updated diagram to match
- Wrote lengthy section describing sychronization in full vs. partial changes and why Sandpiper handles them the way it does
- Shuffled order of some sections and headers to read better
- Minor updates to grammar and phrasing here and there

Attachments

Like

Reply (/forums/post?tid=5424&ReplyPostID=5444&SpaceID=127) Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)

12/31/2019

> **On 12/31/2019 11:16 AM, Doug Winsby said:**
>
> Bringing up "Feedback" is important. We've really not addressed that yet. We should start another discussion on that topic. It's a big one.

This, let's save for another round, because you're right -- it's quite large and it's one we don't intend to tackle for Sandpiper pre-1.0. However, if we want to have a discussion about the methods of feedback as error catching at the Sandpiper level rather than at the PIM level like the particle would enable, I can start that one up..!

Like

Reply (/forums/post?tid=5424&ReplyPostID=5445&SpaceID=127)   Answer

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
12/31/2019

"Out of scope" (I love it!)

feedback-out-of-scope (/spaces/127/sandpiper/searchresults?keyword=%22feedback-out-of-scope%22&searchtags=1)

Like

Reply (/forums/post?tid=5424&ReplyPostID=5446&SpaceID=127)   Answer

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
12/31/2019

Here is the data hierarchy included in the write-up:

>```
>1. Node
>2. Pool
>3. Set
>4. Section
>5. Slice
>6. Grain
>```

For the initial release, I was not planning to implement the Set and Section levels. Node and Pool are kind of implied (Node being the server and Pool the database). Just wanted to manage expectations on this point.

Also, the discussion of "Full Replacement" might be misleading. It becomes important now because it says it's only supported in Level1. This relates closely to one of the questions I asked here (https://autocare.communifire.com/spaces/127/sandpiper/forums/api/5410/-level-1-design) (when talking about what "keys" to use).

If "Full Replacement" means the ability to add/delete file-based grains, then I agree with your intent. But there really is no concept of what constitutes "Full" (full item, full product brand, full product-line?).

Basically the question was how the file-based CREATE api would know if it was creating a duplicate. For example, what constitutes a duplicate `aces-file` grain (within a slice)?

full-replacement (/spaces/127/sandpiper/searchresults?keyword=%22full-replacement%22&searchtags=1)

Like

Reply (/forums/post?tid=5424&ReplyPostID=5447&SpaceID=127) Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
1/6/2020

> **On 12/31/2019 1:04 PM, Doug Winsby said:**
>
> For the initial release, I was not planning to implement the Set and Section levels. Node and Pool are kind of implied (Node being the server and Pool the database). Just wanted to manage expectations on this point.

The set you might need to implement just because this is the only thing that dictates what kind of file is being transmitted. It also will contain all the references to VCdb PCdb etc. This becomes critical to the operation of the full replacement when doing machine-machine transfers.

I agree on the full replacement terminology -- I'll have to clarify that in the document. "Full" should mean every bit of data within that set (for a level 1 transaction with no grains) or grain (for a level 1 transaction using grain windowing).

So if we sent a full ACES recordset (a full file within the set) as an explicit ungrained transaction, it replaces all ACES data for all part numbers in that set. If we send a windowed update for PIES data for just one part number, it replaces all PIES data for that part number in that set; smaller grain. (This is why the part number UUID becomes important I think).

In this kind of transaction, duplication should not be possible because it's not an explicit delete-add -- it's a strictly-controlled implicit "Drop it all and pull this in". The only valid content for a part-number-grained transaction entry would be data for that part number.

Like

Reply (/forums/post?tid=5424&ReplyPostID=5458&SpaceID=127)   Answer

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
1/6/2020

> **On 1/6/2020 9:19 AM, Krister Kittelson said:**
>
> The set you might need to implement just because this is the only thing that dictates what kind of file is being transmitted. It also will contain all the references to VCdb PCdb etc.

The way I have it now (looking at the ER diagram), "slice-metadata" includes all the reference-versions (VCdb, PCdb, etc). The kind of file being transmitted is currently defined by the "grain-type" (aka payload-type). Am I missing something big here?

Like

Reply (/forums/post?tid=5424&ReplyPostID=5470&SpaceID=127)   Answer

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
1/7/2020

> **On 1/6/2020 9:19 AM, Krister Kittelson said:**
>
> So if we sent a full ACES recordset (a full file within the set) as an explicit un-grained transaction, it replaces all ACES data for all part numbers in that set. If we send a windowed update for PIES data for just one part number, it replaces all PIES data for that part number in that set; smaller grain. (This is why the part number UUID becomes important I think).

Im confused by some of your terminology. As I understand it, a sandpiper transaction is all about syncing "grains". So I'm not sure what an "un-grained transaction" is. Do you mean sending file-based grains?

Also, when you talk about a "windowed update", do you mean an item-based grain? If so, I think it might be more clear to talk about "grain-type". That's currently a column in the grain table and describes the payload included in the grain.

I think adding a "grain-key" solves some of the problems of mixing grain-types within a "set". But the "set" (or any grouping of products), in my mind, is a PIM responsibility. We just want to give the PIM enough hooks to keep it from shooting itself in the foot (which would happen if it mixed file-based grains with overlapping item-based grains in the same slice).

I did bring up the idea of adding some structure to the pool in this post (https://autocare.communifire.com/spaces/127/sandpiper/forums/api/5279/proposal-for-slice-aggregation-terms-update?postid=5314#5314). I think the fallout of that discussion, though, was that it was out-of-scope for v1.

Like

Reply (/forums/post?tid=5424&ReplyPostID=5473&SpaceID=127)   Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
1/8/2020

> **On 1/6/2020 11:21 PM, Doug Winsby said:**
>
> The way I have it now (looking at the ER diagram), "slice-metadata" includes all the reference-versions (VCdb, PCdb, etc). The kind of file being transmitted is currently defined by the "grain-type" (aka payload-type). Am I missing something big here?

Each slice should be able to be compared to each other slice using the same criteria -- they are slices of a greater whole. The set is what provides this, and also the VCdb version, PCdb version, etc., so that within a set, all the data can be stored, analyzed, and integrated the same way.

I had thought we would have the grain occur as part of a slice. The slice belongs to the set. Therefore the grain doesn't actually have to have any of the metadata transmitted -- it's just "Here's the slice ID I'm part of." That way too it is never concerned with versions or details, and the framework can be extended without ever touching the grain implementation.

Like

Reply (/forums/post?tid=5424&ReplyPostID=5479&SpaceID=127) Answer

Page 1 of 2 (30 items)

1 (/spaces/127/sandpiper/forums/general/5424/reference-documentation-for-0-8/1)

2 (/spaces/127/sandpiper/forums/general/5424/reference-documentation-for-0-8/2)

Next » (/spaces/127/sandpiper/forums/general/5424/reference-documentation-for-0-8/2)

Posted in: General (/spaces/127/sandpiper/forums/4997/general)

Reference Documentation for 0.8

Unsubscribe from this discussion
Subscribe to RSS (../../../../../../../spaces-cf/forums/rss-space-posts.ashx?spaceID=127&topicID=5424&forumID=4997&key=rrer%2B1xDo%2BlxpC7jBRbM5w%3D%3D)

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
1/8/2020

> **On 1/7/2020 11:01 AM, Doug Winsby said:**
>
> Im confused by some of your terminology. As I understand it, a sandpiper transaction is all about syncing "grains". So I'm not sure what an "un-grained transaction" is. Do you mean sending file-based grains?

I was thinking about this as well. It's not clear enough. I need to clarify the transaction and talk through it with you guys to see what we all think.

My thought is that every interaction between actors is called a transaction.

I'm going to start another thread on this and tag you and Luke to get your thoughts.

> **On 1/7/2020 11:01 AM, Doug Winsby said:**
>
> Also, when you talk about a "windowed update", do you mean an item-based grain? If so, I think it might be more clear to talk about "grain-type". That's currently a column in the grain table and describes the payload included in the grain.

I think this will be resolved with the discussion above. My thinking was that we had level 1 data transfers happening without ever using a grain, just passing URIs that the clients fetch from. Level 2 would be where the grain came in. But I could see the grain being used as an ever-present unit even when it's just a single grain passing a single large file.

Like

Reply (/forums/post?tid=5424&ReplyPostID=5480&SpaceID=127)   Answer

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
1/8/2020

There were early discussions about including "URI link" grain-types (in place of actual payload data), but allowing links would break our ability to track data changes (there's no easy way to know if the target of a URI has changed). By only supporting payload data, we ensure data integrity and version tracking.

link-payloads (/spaces/127/sandpiper/searchresults?keyword=%22link-payloads%22&searchtags=1)

Like

Reply (/forums/post?tid=5424&ReplyPostID=5481&SpaceID=127)   Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
1/8/2020

> **On 1/8/2020 9:24 AM, Doug Winsby said:**
>
> There were early discussions about including "URI link" grain-types (in place of actual payload data), but allowing links would break our ability to track data changes (there's no easy way to know if the target of a URI has changed). By only supporting payload data, we ensure data integrity and version tracking.

It makes sense for machine-machine interaction, since for the human/machine model (which really just means one side is a Sandpiper system and the other has to figure out how to get the data using URIs) they'll just use the plan alone along with a UI to help the negotiation process.

Like

Reply (/forums/post?tid=5424&ReplyPostID=5484&SpaceID=127)   Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
1/10/2020

OK, taking into account the Level 1 stuff above, I've written a new version of this document.

- Added interaction model description & diagram
- Added Grain Key, a UUID, as only method of criteria describing the data contained in the grain
- Added parts as list of part-creator with UUID contained in the node itself; still need to think about how this should be transmitted, maybe as a second XML file to the plan?
- Level 1 now divided into Level 1 Basic and Advanced, an either-choice (not both at once)
  - Basic is the human-machine or very basic machine-machine, they'll rely on file deliveries via URIs (no change tracking available, it's purely classic delivery/download model on a regular scheduled subscription). The Sandpiper server can hand this off to the OS FTP client / a file path / etc, if it wants; it just has to help get the ball rolling and drop files out someplace on a regular schedule. No synchronization, no grains, no transactions, only negotiation and confirmation when changes to the plan are needed
  - Advanced is the sandpiper systems actually connecting to one another and engaging in synchronization with transactions and grains
-

Attachments

Like

Reply (/forums/post?tid=5424&ReplyPostID=5511&SpaceID=127)   Answer

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
1/10/2020

Sandpiper human-to-machine (Level 1a) would entail:

- navigating to a sandpiper server landing page (via browser)
- logging in (authentication)

- listing your slices
- pressing a "download" button next to a slice

Sandpiper machine-to-machine (sync) is Level 1b.

Like

Reply (/forums/post?tid=5424&ReplyPostID=5517&SpaceID=127) Answer

Page 2 of 2 (30 items)

« Previous (/spaces/127/sandpiper/forums/general/5424/reference-documentation-for-0-8/1)

1 (/spaces/127/sandpiper/forums/general/5424/reference-documentation-for-0-8/1)

2 (/spaces/127/sandpiper/forums/general/5424/reference-documentation-for-0-8/2)