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

Reference Documentation for 0.8 - Beta

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
3/30/2020

Hello, All,

I've finished a beta version of our 0.8 documentation. I've attached the PDF here, and a zip of the markdown source and images.

Please review and let me know if you find errors. I think there is still a need for more visuals, but right now the content seems to have stabilized enough to call this a beta in a very casual sense.

Future effort will also go into a new "introduction" document that talks more in depth about the overall project, but I've got a nice introduction here that states the purpose of the standard, that I left in; it'll have to stand alone to some degree for distribution as a whitepaper. I'll also rename files to reflect the version and so on.

Will also add URLs and so on as I get up to speed with the web side of things.

-- Edited: Updated to 0.7.2.1

Attachments

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
3/30/2020

Nice! I will review when I can.

I added the "Introduction" to the website just to show how that works. I also added a google translate version of it to our French content.

https://sandpiper.onrender.com/docs/introduction/ (https://sandpiper.onrender.com/docs/introduction/)

I left the sample Doc sections so you can see how things are done, but feel free to remove them. We can always find them on the Docsy website docs.

Make a new directory for each heading you want to show in the table of contents on the left. That index is actually built (and ordered) according to the "Front Matter" found in each `_index.md` file found in those directories. The Front Matter looks like this:

```
---
title: "Introduction"
linkTitle: "Introduction"
weight: 1
description: >
  What is Sandpiper and why was the project started
---
```

"linkTitle" is what shows in the index. "weight" is for ordering the sections.

Put any images in the directory where it is used. There are lots of other features, but that should get you started.

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

Like

Reply (/forums/post?tid=5857&ReplyPostID=5868&SpaceID=127) Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
4/1/2020

I removed the requirement for grain keys to be UUIDs, as discussed in the other documentation thread. This added a lot of overhead for little benefit at this time. Here's a diff:

```
--------------------------------- Sandpiper.md ---------------------------------
index 1b379ea..402a8d6 100644
@@ -159,8 +159,6 @@ To structure this data and aid the creation of shared scope be
tween actors, the

 The node is a single Sandpiper instance or system^[While a server might run multi
ple concurrent copies of the Sandpiper software and thus represent multiple nodes,
for simplicity we just refer to the node as a system, as this is the most common u
se case.]. It has a *Controller* responsible for, though not necessarily the origi
nator of, its operation and contents.

-Nodes also contain a list of part numbers, their creators, and a UUID assigned to
them that will be used for all operations referencing a part number.
-
 Note: a human interacting at Level 1-1 is technically a node, though their data s
tate is unknown after retrieval.

 ##### Pools
@@ -187,7 +185,7 @@ A slice defines the single type, format, and version of the da
ta it contains (e.

 A slice is broken into *Grains*, each representing one unit of meaning or scope.
 It can be thought of as the element level of the data.

-Grains have a *Grain Key* containing an unambiguous UUID, to safely and atomicall
y operate on the data in pieces smaller than a whole slice.
+Grains have a *Grain Key* containing a single text value, to safely and atomicall
y operate on the data in pieces smaller than a whole slice. Ideally this is a [UUI
D](#UUIDs) but a non-compounded single unicode value can be used.

 The grain is the smallest unit on which a Sandpiper node can act, and can only be
directly addressed in Level 2 and higher transactions.

@@ -374,7 +372,7 @@ Grain
 : An addressable and self-contained unit within the slice

 Grain Key
-: A reference to a single UUID that can be used by Sandpiper to operate on data
+: A reference to a single key value that can be used by Sandpiper to operate on d
ata on a grain-by-grain basis
```

Like

Reply (/forums/post?tid=5857&ReplyPostID=5870&SpaceID=127) Answer

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

4/1/2020

Okay, this one made me smile (but I have no idea what it means):

> `"a non-compounded single unicode value"`

Like

Reply (/forums/post?tid=5857&ReplyPostID=5871&SpaceID=127) Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
4/2/2020

> **On 4/1/2020 10:50 PM, Doug Winsby said:**
>
> Okay, this one made me smile (but I have no idea what it means):
>
> > `"a non-compounded single unicode value"`

Ah, I should clarify that in the text itself too. I meant, this is not meant to be a place to store some packed multiple value key, like "ABC|123|Brown". That's a nightmare for parsing and the intent is to provide a spot to enter one unique, unambiguous key value.

Like

Reply (/forums/post?tid=5857&ReplyPostID=5872&SpaceID=127) Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
4/2/2020

Updated the language around the grain key to reflect a more clear definition of "single value". Diff:

```
-------------------------------- Sandpiper.md --------------------------------
index 402a8d6..8d66aab 100644
@@ -185,7 +185,7 @@ A slice defines the single type, format, and version of the da
ta it contains (e.

 A slice is broken into *Grains*, each representing one unit of meaning or scope.
 It can be thought of as the element level of the data.

-Grains have a *Grain Key* containing a single text value, to safely and atomicall
y operate on the data in pieces smaller than a whole slice. Ideally this is a [UUI
D](#UUIDs) but a non-compounded single unicode value can be used.
+Grains have a *Grain Key* containing a single text value, to safely and atomicall
y operate on the data in pieces smaller than a whole slice. This value must be a s
ingle unicode key that directly references one key value within the data, e.g. a p
art number or a [UUID](#UUIDs). It must not be an artificially packed or delimited
set of values referring to more than one key within the data.

 The grain is the smallest unit on which a Sandpiper node can act, and can only be
directly addressed in Level 2 and higher transactions.
```

Like

Reply (/forums/post?tid=5857&ReplyPostID=5873&SpaceID=127) Answer

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

4/2/2020

The way I've been thinking of the "grain-key" is purely as a convenience for the PIM. I currently have it defined as "not null" (i.e. required), but I'm not sure it needs to be. I originally added it for two reasons:

1. The original design did not restrict what grain types could be in a slice. This meant, for example, you could have two "aces-file" grains in the same slice, one for "Brake Pads" and another for "Brake Shoes". We needed a way to uniquely identify a grain for replacement (other than by the UUID). So the grain-key, in this cases, would be "Brake Pads" or "Brake Shoes".
2. Digital assets were a special case because they were file-based, but you could have lots of them. It was natural, then, to include the Part Number (or other PIM-assigned identifier) as the key. (I think this case still holds true with the current one-grain-type-per-slice design.)

The reason for the key in both of these cases, however, was for the PIM to manage the grains directly without some added index that it would need to maintain externally. I don't believe it was ever intended for the receiver to use it explicitly. It is not something they could depend on being

used consistently from one supplier to the next.

So the takeaway here, I think, is that it is an optional field available for the publishing PIM to use as it sees fit.

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

Like

Reply (/forums/post?tid=5857&ReplyPostID=5874&SpaceID=127) Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
4/2/2020

> **On 4/2/2020 8:56 AM, Doug Winsby said:**
>
> The way I've been thinking of the "grain-key" is purely as a convenience for the PIM. I've got it defined as "not null" (i.e. required), but I'm not sure that it needs to be.

I think you're right, it's not going to be used by the Sandpiper client -- it's for something further up or down in the chain to use. I'm trying to put some fences around the intended use to make sure people don't abuse it by inserting quadruple-primary-key indexes, or coded values to use elsewhere in the processing chain.. also leaving the door open for the proper implementation of a parser that can work with the sandpiper repository independent of a PIM (I could see this being a very useful tool, just not something we are going to develop here).

I think a section of the documentation discussing the role of the client software vs. PIM vs. others will be good, I'll add that and a visual.

Like

Reply (/forums/post?tid=5857&ReplyPostID=5875&SpaceID=127) Answer

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
4/2/2020

My last response above was referring to L1 slices. For L2 slices (e.g. "aces-items"), the grain-key might be more important and would probably always be something that relates to the "item" (part number) for any of the L2 *-items slice-types. But again, only as a publishing

PIM convenience (to decide what to replace on an update).

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

Like

Reply (/forums/post?tid=5857&ReplyPostID=5876&SpaceID=127) Answer

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
4/2/2020

Or we could drop the grain-key entirely and keep the update mechanics completely separate from Sandpiper. Thoughts **Luke Smith (https://autocare.communifire.com/people/autopartsource)** ?

separation-of-concerns (/spaces/127/sandpiper/searchresults?keyword=%22separation-of-concerns%22&searchtags=1)

Like

Reply (/forums/post?tid=5857&ReplyPostID=5877&SpaceID=127) Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
4/2/2020

I think the update itself is important to have handled by Sandpiper. For Level 2 it'll be a bit more complex but having the "source of truth" regarding what should be current based on what was sent, I think is prime.

Like

Reply (/forums/post?tid=5857&ReplyPostID=5878&SpaceID=127) Answer

Doug Winsby (https://autocare.communifire.com/people/dougwinsby)
4/2/2020

> **On 4/2/2020 9:13 AM, Krister Kittelson said:**
>
> I think the update itself is important to have handled by Sandpiper.

Absolutely, and the current API provides a way to add and delete grains by UUID.

I'm torn on whether or not to include the grain-key, though. I really like it for the asset-file case. And it seems natural to have it for L2 item-level grains as well. Anything more granular than item gets fuzzy as to what you might use it for.

There is some appeal to keeping it "clean" (making the PIM keep track of grain IDs). But it puts a slightly heavier burden on PIM implementations. They might need to have those kind of tracking tables anyway, in which case they just won't use our key (if supplied).

Like

Reply (/forums/post?tid=5857&ReplyPostID=5879&SpaceID=127) Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
4/2/2020

> **On 4/2/2020 9:23 AM, Doug Winsby said:**
>
> I'm torn on whether or not to include the grain-key, though. I really like it for the asset-file case. And it seems natural to have it for L2 item-level grains as well. Anything more granular than item gets fuzzy as to what you might use it for.

I still want to include it. I'm not sure I trust the PIM to handle all the key associations' storage -- Sandpiper can store for it, even if the PIM or a preprocessor is what's actually filling out that field. That way the grain UUID never gets handed off to someone else in a way that could allow a mismatch or confusion.

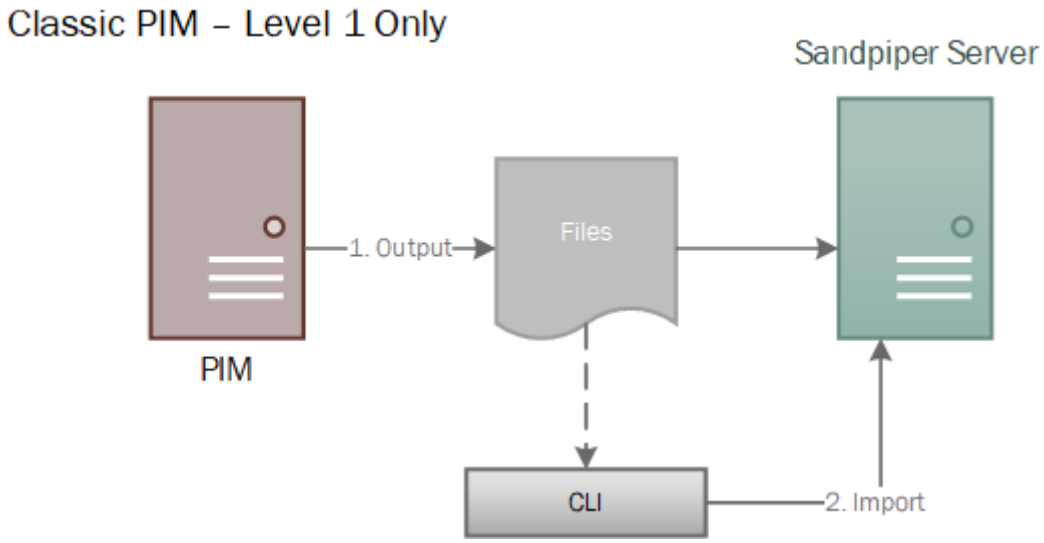
Like

Reply (/forums/post?tid=5857&ReplyPostID=5880&SpaceID=127) Answer

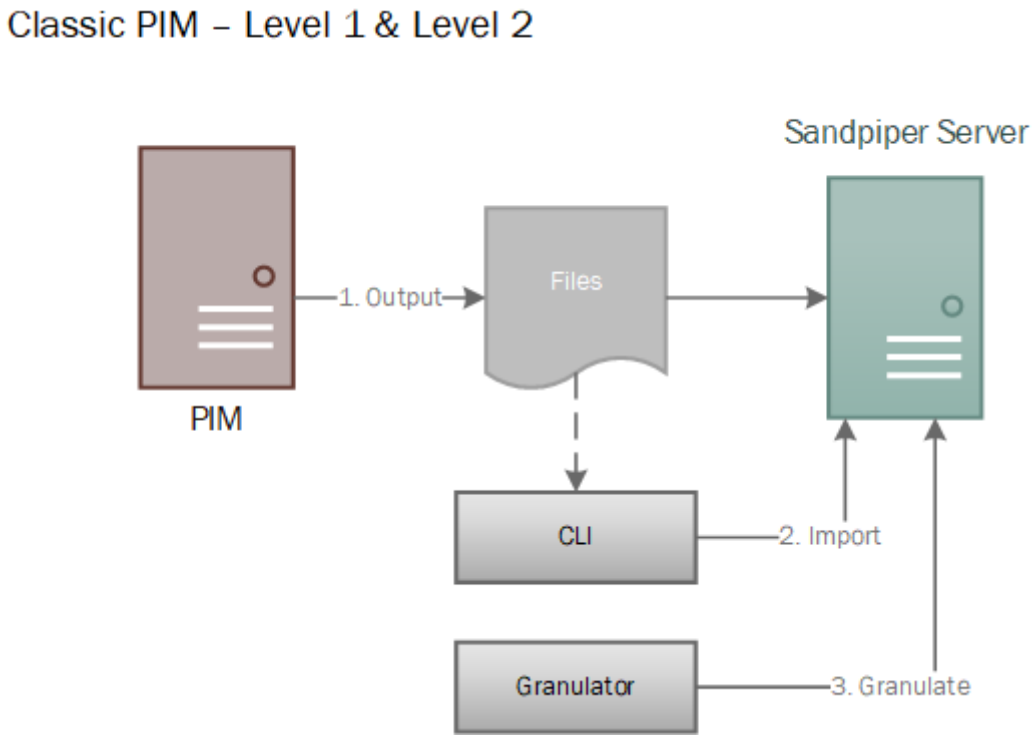
Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
4/2/2020

Here's something I'm going to add to the manual around workflow.. I'll clean up the wording but wanted to check the visuals with you guys.

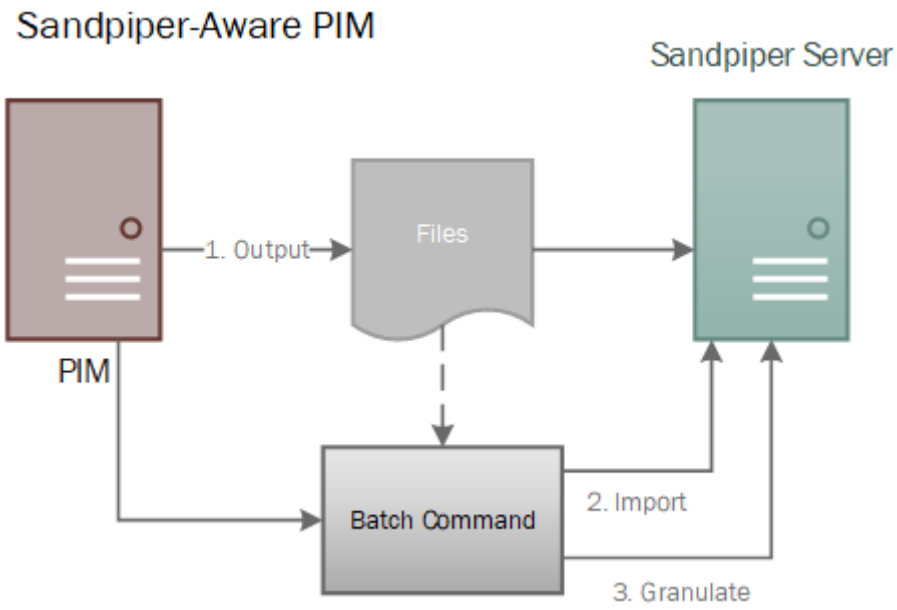
The PIM can output files to be imported by the Sandpiper server. Classic PIMs will require the user to execute a command to load them in:



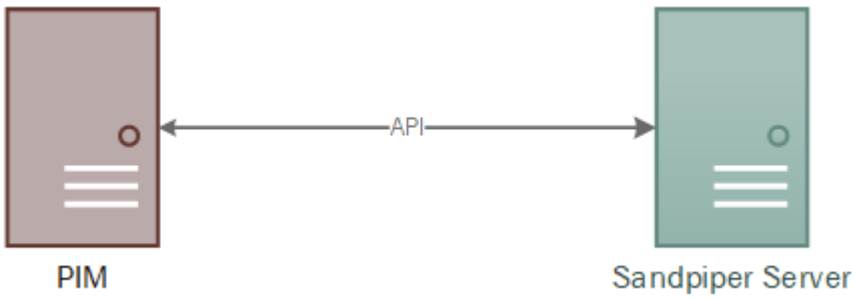
To proceed to level 2, you need grains. A processor has to create these since the Sandpiper server is content-agnostic; it provides an API and CLI to fill in the grain information, using the slice grain type and master association:



Some PIMs will be modifiable to be "Sandpiper Aware", triggering a batch script or similar to kick off this processing without needing the user to use the CLI:



A truly Sandpiper Capable PIM does not need these intermediary steps; it can use the API directly:



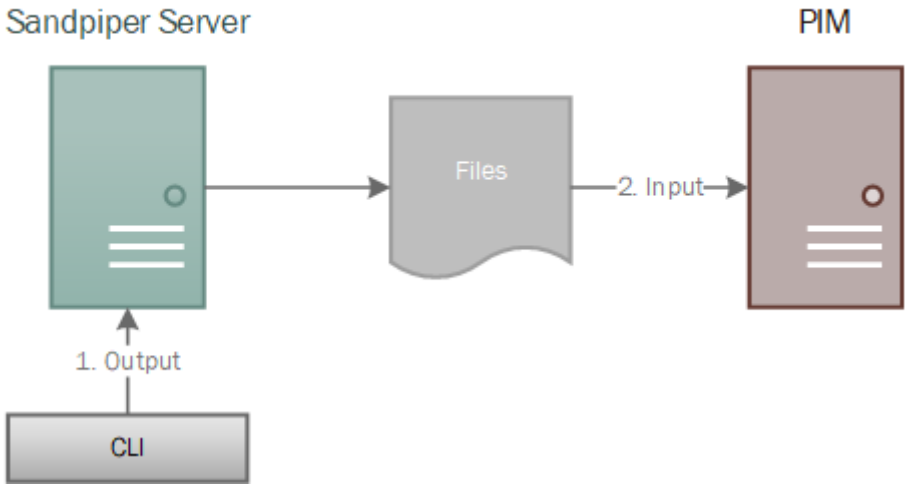
Like

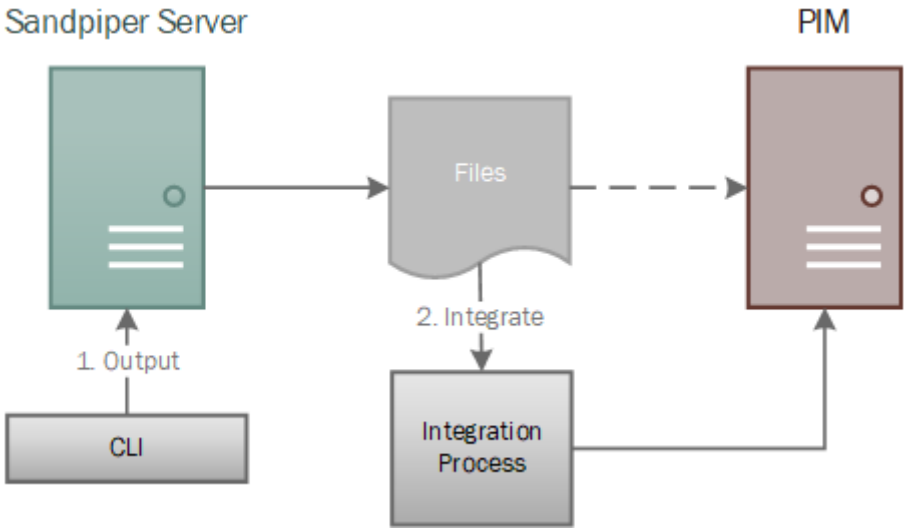
Reply (/forums/post?tid=5857&ReplyPostID=5881&SpaceID=127) Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
4/2/2020

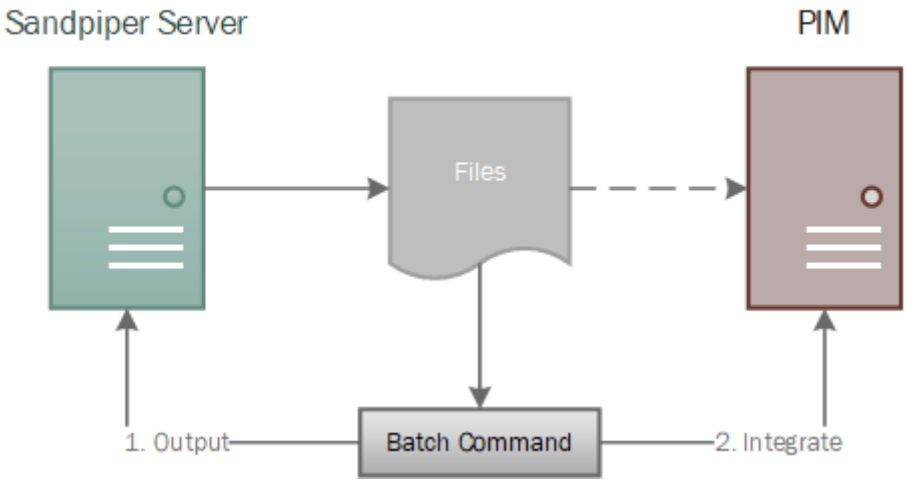
Here is the output process workflow in a similar vein:

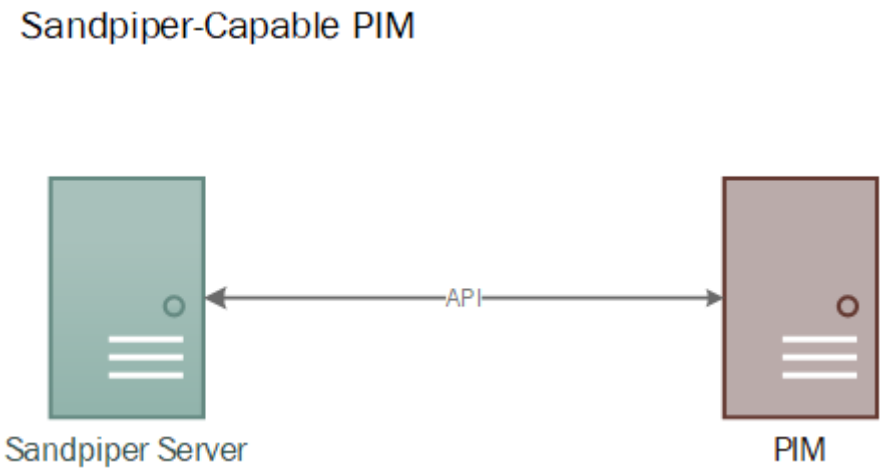
Classic PIM – Level 1 Only





Sandpiper-Aware PIM





Like

Reply (/forums/post?tid=5857&ReplyPostID=5882&SpaceID=127) Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
4/3/2020

Updated documentation:

- Added input/output implementation diagrams and descriptions
- Description of PIM and three different types: classic, Sandpiper Aware, and Sandpiper Capable
- Clarified "single key value" grain key statement
- Removed references to UUID requirement for grain key
- Overview of a granulator and what it does, and why the Sandpiper server does not do this
- Description of master slice linking and full file vs. granulated, along with handy example diagram

Attachments

Like

Reply (/forums/post?tid=5857&ReplyPostID=5886&SpaceID=127) Answer

Krister Kittelson (https://autocare.communifire.com/people/krister-kittelson)
4/3/2020

Updated with one change:

- Renamed "Grain Type" to "Slice Type" throughout

-- Edited: Added zip attachment

Attachments

Like

Reply (/forums/post?tid=5857&ReplyPostID=5894&SpaceID=127) Answer