JDBC stack issue

Issues:

1. One Storage Engine to one database. If you need to store multiple data type that are supported by different SE, then it need to be stored in different DB. This is not a big issue is the Core support different SE for different signal which is easy to implemented.
2. Data Type is tied to SE and Plug-In. User can only read write strongly type data like int-wave using fixed interval int-wave type which is support by that particular plug-in. if they just want to store a matrix which is perfect for Matrix SE, they are not able to!

Analysis:

Ok let us look at file system. File allow user to store any file types on the driver. The interpretation of the file is done by specific apps, the disk just reads and writes bits to the disk.

But this is not what we are aiming for. What we are doing is integrating some of the functions of the client app as per the file system into our platform. E.g. user can specify a start and end time and decimation on our platform and get the date they want. But in file system, they need a app to read and seek the bit in the file to get the data.

So what are we really making? A storage “thing” that integrates apps for specific data types. Does this mean we cannot store anything that is not support by the integrated app? Short answer: yes. But its ok if the app can be installed. The app is called plug-in.

But, the plug-in does only support certain types of SE. this is like an app only support some kind of file system(FS). Yes different FS is different, that’s why we have POSIX FS APIs. Thus we have IMatrixStorageEngine.

It seems that we are doing great? No quite. What we are not doing so well since when we designed the SE and SE interface, we started by thinking a particular use case: MDSplus. So we ended up with a SE suitable for Matrix and a plug-in for vector (one row matrix) which is pretty limited. This means we have to not only make new plug-ins for new data types but also very likely needs new SE and new DB.

This is due to we are aiming for more performance than FS which just stores bits. We are using DB which provide more performance if we designed the data schema suitable for a specific date type. You heard it, a schema for a data type, which means a DB thus an SE for a data type. The problem is clear: we do not have a plug-in for some data types, but a SE for some data types.

Let us look at DB, DB does not store bits, they store object (here we are talking about NoSQL). But it is applicable for all kinds of applications. Why? Because the application defines the schema not the DB. SE is data type specific because it defines the schema!

Solution:

The solution is NOT very straight forward.

Let us say we keep the current stack, but embrace the concept that the SE is just like plug-in, we need to make multiple SE not just one. Which does not affect the current code. One bad thing is we have to deploy multiple DB if we support many data types.