Introduction

The aerospace industry is heavily regulated. Data used to validate manufacturing processes is required by such regulations to be retained for many years.

Just retaining the data though, is not sufficient. It must also be readily accessible and consumable. Investigations arising from aircraft incidents/accidents require access to all relevant information (including manufacturing data) and may occur decades after production.

Twenty years ago, floppy-disks were in common use and Adobe Flash was the preferred format for creating expressive webpages. Both are now gone. Managing conflicting requirements of long-term retention and short-term capabilities presents challenges.

Primary data storage formats must fulfil three key values: longevity, simplicity and utility. Analytics layers can be leveraged to provide value-added capabilities and replaced if necessary, to respond to changes in the ecosystem.

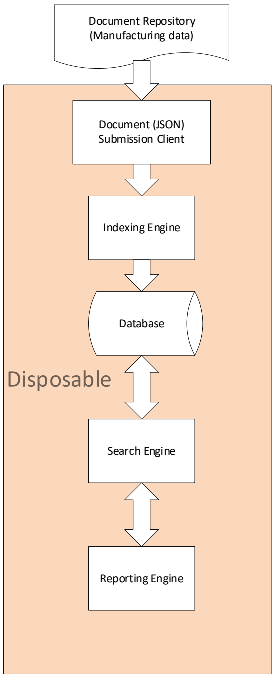
Data Obsolescence

IT systems have a history of changing rapidly. Leveraging proprietary storage/analysis solutions for long term data management is likely to lead to an escalating tech-debt burden.

Approaches for success

Using history as a guide, we can make informed guesses about how to create enduring data containers that evade obsolescence. Key tenets include:

- Use standard data formats, that are:
  - Simple
  - Self-contained
  - Human readable
  - Language agnostic
  - Stateless
  - Platform independent
- Use a layered stack – allow analysis layers to be swapped out as newer/better options become available.


Document Repository (Manufacturing data)
Document (JSON) Submission Client
Indexing Engine
Database
Disposable
Search Engine
Reporting Engine

Boeing Proprietary

Project

Your task contains two deliverables:

1. Design a stack to:
   - Consume the JSON data files
   - Index the various fields
   - Provision Search capabilities
   - Provide a UI (website) to allow users to execute search
2. Build one (or more) layers of the stack. You will be presenting your design and demonstrating the layer/s you build.

Proposal

Your design should consider:

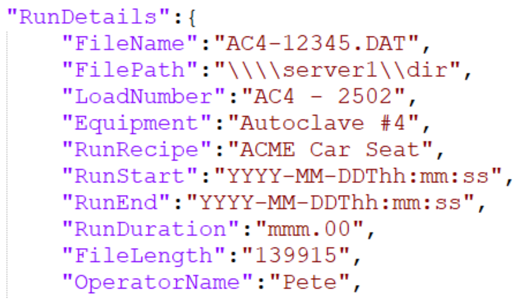
- Security (avoid leakage and ensure IP protection)
- Leverage open-source/COTS tools
- Adopt industry standard methods/protocols where possible
- Cloud deployable
- Scalability
- Minimise cost

Details

The provided data files are a JSON representation of the metadata associated with each autoclave cure. Each file has two top level blocks:

1. The *Run* section, which contains
   - File name,
   - File path
   - Load sequence,
   - Equipment identifier
   - Recipe
   - Start time
   - End time
   - Run duration
   - File Length
   - Operator name

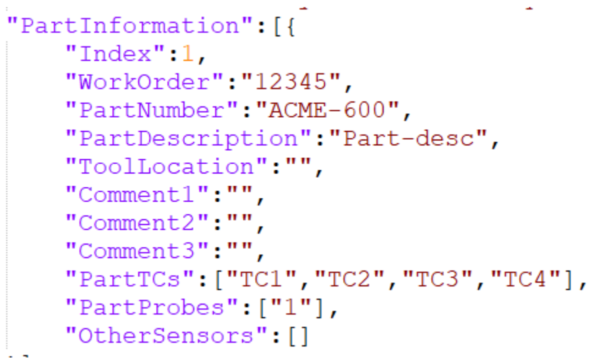
Example:



```
"RunDetails":{
    "FileName":"AC4-12345.DAT",
    "FilePath":"\\\\server1\\dir",
    "LoadNumber":"AC4 - 2502",
    "Equipment":"Autoclave #4",
    "RunRecipe":"ACME Car Seat",
    "RunStart":"YYYY-MM-DDThh:mm:ss",
    "RunEnd":"YYYY-MM-DDThh:mm:ss",
    "RunDuration":"mmm.00",
    "FileLength":"139915",
    "OperatorName":"Pete",
```

2. The *Part* section is an array of blocks; each block containing details of a part included in the cure. The details include:
   - Part ordinal
   - Workorder number
   - Part Number
   - Description
   - Location
   - Comment/s
   - Thermocouple ports (TCn)
   - Vacuum ports (MONn)
   - Other sensor identifiers

   Example:



```
"PartInformation":[{
    "Index":1,
    "WorkOrder":"12345",
    "PartNumber":"ACME-600",
    "PartDescription":"Part-desc",
    "ToolLocation":"",
    "Comment1":"",
    "Comment2":"",
    "Comment3":"",
    "PartTCs":["TC1","TC2","TC3","TC4"],
    "PartProbes":["1"],
    "OtherSensors":[]
```

The Solution

The stack must satisfy three separate capabilities:

1. A mechanism to accept/parse/index the data files, for example: A webservice that accepts client HTTP POST requests. The body of the request contains the JSON data for one cure.
   a. Receive the JSON text
   b. Parse the text
   c. Persist the data, preserving the structural relationships.

   Note: Your solution does not have to implement this example.

2. A website to facilitate cure search capabilities:
   a. Design a website to present a UI for cure search. Consider:
      i. Metadata fields to be included
      ii. Implementation of combinational logic
      iii. Layout of the page
      iv. UI elements for ease of use and reduction of input errors (dropdowns v. free-text etc.)
   b. Results:
      i. Design a results page to display list of cures satisfying the search criteria. Each item in the list should implement a hyperlink. When clicked, the hyperlink should display the full data for the associated cure.

3. A client tool to submit the submit JSON data files to the consuming service (1).

Recommendations

Design your stack using the values outlined in the proposal. Also consider factors such as learning-cure, complexity, capabilities, and estimated lifespan.

Solution Values

Visual appeal and usability/capability of the search website are key values with scalability and performance, secondary.