Rapid Converter Framework
An approach to ultra-fast converter development

Paul-James Jones, Justin Van Duine & Bo Du
Fall, 2020

Our Inspiration: Henry J. Kaiser
The Current State (problem)
The Current State (problem)

- Converter building is currently a craft process, requiring arcane knowledge and skill.
The Current State (problem)

• Converter building is currently a craft process, requiring arcane knowledge and skill.

• It is slow, precisely because it is a craft process, and requires arcane knowledge
The Current State (problem)

• Converter building is currently a craft process, requiring arcane knowledge and skill.

• It is slow, precisely because it is a craft process, and requires arcane knowledge.

• In order to move beyond the hobbyists’ workshop model we need to be inspired by Kaiser’s example.
The Objective

• Building on our collaborative experience in developing converters, we are laying the groundwork for a “Rapid Converter Framework” that will potentially let us build brand new converters in days, not weeks or months.
The Objective

• Building on our collaborative experience in developing converters, we are laying the groundwork for a “Rapid Converter Framework” that will potentially let us build brand new converters in days, not weeks or months.

• It is based on Kaiser’s core principles:
  ➢ Ruthless Simplification
  ➢ Ruthless Standardization
  ➢ Ruthless Separation (into tiny, elemental steps)
The Objective

- Wherever we can, we’re letting software automate the production of converters...
The Objective

- Wherever we can, we’re letting software automate the production of converters...
  - By building data models
  - By *writing code* (modules)
  - By testing the output
The Objective

• Wherever we can, we’re letting software automate the production of converters...
  
  ➢ By building data models
  ➢ By *writing code* (modules)
  ➢ By testing the output

• What you will see here is not necessarily the final form of the Rapid Converter Framework that we might implement, but it is intended to show what’s possible and where we’re going...
Let's Start
In the beginning...
In the beginning...
If you can’t read the vendor file, there is no realistic chance to ever build a converter.

In the beginning...
As soon as you can read the vendor file, we can “feed” the Data Model Creator to automatically build (i.e. write code for) a minimalistic, first-version “Consensus Data Model” for us.
As soon as you can read the vendor file, we can “feed” the Data Model Creator to automatically build (i.e. write code for) a minimalistic, first-version “Consensus Data Model” for us.
We have decided to standardize on CSV as a “messaging” format between the components of the Rapid Converter Framework. This is not necessarily a final design decision: JSON also possible...
One side benefit to this particular step is that you get to stress-test your File Parser against the diversity of vendor file format variations that you might reasonably expect to encounter in the wild.
Once you have your Consensus Data Model, you can start producing your first files immediately. The Consensus Data Model Creator *writes the code* the CSV Creator needs to implement the data model right away. Just compile and go.
The process

• Of course, the “Consensus Data Model” is the bare minimum we need to create standard-format files. But using the “Consensus Data Model” approach allows us to decouple the much slower “Final Data Model” development from converter creation.
The process

• Of course, the “Consensus Data Model” is the bare minimum we need to create standard-format files. But using the “Consensus Data Model” approach allows us to decouple the much slower “Final Data Model” development from converter creation.

• This allows our Companies to get much-needed near-term ROI by getting real data into standard-format files quickly so our scientists can poke it, prod it, and get used working with to it.
Final Data Model Creator

Consensus Data Model Creator

Vendor File Parser

CSV Creator

Data Model Integrity Checks

CSV to TTL format conversion

TTL file

Aggregator (Pfizer)

Data Cube

Data Description

Data Package

Standard-format files

community

File Parser (metadata)
The CSV creator allows us to immediately populate the data cubes— at no extra cost or effort!
The Pfizer tool also allows us to directly put whatever we want into the data package...
Typical operational scenario for the Rapid Converter Framework in production... (w/ Consensus Data Model)
Elapsed time from the very beginning  < 1 day
Using flat files as a transfer medium lets us easily intercept / modify the interprocess communication so that we can easily prototype new ideas that are not directly or immediately supported by the codebase.

(for example, novel hybrid full-graph/leaf node data models...)
Using flat files as a transfer medium lets us easily intercept / modify the
interprocess communication so that we can easily prototype new ideas that
are not directly or immediately supported by the codebase.

(for example, novel hybrid full-graph/leaf node data models...)

This is also very useful where you don’t know the best way, *a priori*, to
populate files... You have the ability to “experiment” easily....

(for example: peaks in Data Description vs. peaks in Data Cubes...)

Using flat files as a transfer medium lets us easily intercept / modify the interprocess communication so that we can easily prototype new ideas that are not directly or immediately supported by the codebase.

(for example, novel hybrid full-graph/leaf node data models...)

This is also very useful where you don’t know the best way, *a priori*, to populate files... You have the ability to “experiment” easily....

(for example: peaks in Data Description vs. peaks in Data Cubes...)

or

(for example: metadata as JSON vs. TTL in the Data Package...
Example: Incorporating manually-produced leaf-node, full-graph or hybrid data models

CSV Creator

Vendor File Parser

TTL file
(metadata)

Vendor Proprietary file

Example: Incorporating manually-produced leaf-node, full-graph or hybrid data models

Data Cube

Data Description

Data Package

Aggregator (Pfizer)

Standard-format files

CSV file(s) – one per data cube
(data)

Vendor Proprietary file

CSV Creator

Vendor File Parser
Current Status

• This modular approach lets us mix & match components – like Lego® blocks - the compartments of the files are populated asynchronously and discontinuously as needed.
Current Status

• This modular approach lets us mix & match components – like Lego® blocks - the compartments of the files are populated asynchronously and discontinuously as needed.

• We can rapidly prototype many different file configurations quickly (example: data cube structure/content) – with little or no coding.
Current Status

• This modular approach lets us mix & match components – like Lego® blocks - the compartments of the files are populated asynchronously and discontinuously as needed.

• We can rapidly prototype many different file configurations quickly (example: data cube structure/content) – with little or no coding.

• Once the file parser is in hand, the turnaround to being able to produce the first complete (DD + DC + DP) files for a new type of vendor file is typically on the order of a day or two.
Current Status

• This modular approach lets us mix & match components – like Lego® blocks - the compartments of the files are populated asynchronously and discontinuously as needed.

• We can rapidly prototype many different file configurations quickly (example: data cube structure/content) – with little or no coding.

• Once the file parser is in hand, the turnaround to being able to produce the first complete (DD + DC + DP) files for a new type of vendor file is typically on the order of a day or two.

• We employ extensive instance data checking at several places in the process in order to minimize the amount of post-hoc validation required.
Next Steps

• The elements of the Rapid Converter Framework are currently stand-alone console applications.
Next Steps

• The elements of the Rapid Converter Framework are currently stand-alone console applications.
• We are currently optimizing the inter-module communication.
Next Steps

• The elements of the Rapid Converter Framework are currently stand-alone console applications.
• We are currently optimizing the inter-module communication.
• The thought is to ultimately redeploy this network as a collection of pluggable REST microservices to support massive, rapid converter development & deployment across the enterprise.
“Start swinging!”