

Key Objectives for Analysis Results Standard Project

- Develop a technical specification to prospectively leverage Analysis Results Metadata to drive automation
- Develop a structure to represent Analysis Results as data
- Develop an integrated logical model to support a technical specification and an analysis results dataset
- Illustrate and exercise with a set of common data displays

Identified separate sets of metadata components needed to describe analyses and represent analysis results, with significant overlap.

The diagram illustrates the relationship between Analysis Results Metadata and Analysis Results Data. A large blue arrow points from the 'Analysis Results Metadata' box to the 'Analysis Results Data' box.

Analysis Results Metadata

Identifiers		Analysis Group			Result Variable		
Name	Title	Dataset	Variable	ue	Variable	Value	Label
Table 2	Baseline Demographics and Clinical Characteristics, Safety Population	ADSL	TR01X	Drug Name Dosage X	SEX	M	Male
Table 2	Baseline Demographics and Clinical Characteristics, Safety Population	ADSL	TR01X	Drug Name Dosage X	SEX	M	Male
Table 2	Baseline Demographics and Clinical Characteristics, Safety Population	ADSL	TR01X	Drug Name Dosage X	SEX	F	Female
Table 2	Baseline Demographics and Clinical Characteristics, Safety Population	ADSL	TR01X	Drug Name Dosage X	SEX	F	Female

Analysis Results Data

Results Statistic		
Value	Name	Label
53	Count	n
61.6	Percent	%
33	Count	n
38.4	Percent	%

Business Problem

Need to formalize a data model for expressing analysis results and the analyses used to generate them. There are no non-commercial modeling tools that can do this. We wanted to use a proven open source solution that would facilitate both the development and the implementation of the model.



- The diagram illustrates the LinkML ecosystem components, organized into three main sections:

 - LinkML Core:** Represented by the LinkML logo (a stylized 'M' with a green circle) and a list of features:
 - Open Source
 - Simple YAML-based model development
 - Already used for other CDISC projects
 - Powerful modeling features
 - LinkML Utilities:** Represented by a list of tools:
 - Availability of LinkML Utilities
 - Generators
 - Converters
 - Schema linter
 - LinkML Documentation:** Represented by a QR code and the text "LinkML Documentation".

- The diagram illustrates the LinkML ecosystem components, organized into three main sections:

 - LinkML Core:** Represented by the LinkML logo (a stylized 'M' with a green circle) and a list of features:
 - Open Source
 - Simple YAML-based model development
 - Already used for other CDISC projects
 - Powerful modeling features
 - LinkML Utilities:** Represented by a list of tools:
 - Availability of LinkML Utilities
 - Generators
 - Converters
 - Schema linter
 - LinkML Documentation:** Represented by a QR code and the text "LinkML Documentation".



- ARS schema developed in YAML
- Schema linter used to check modeling best practices
- Generators used to re-represent the model in other frameworks (JSON-Schema, GraphQL, SQL DDL, OWL, Mermaid ER diagram)

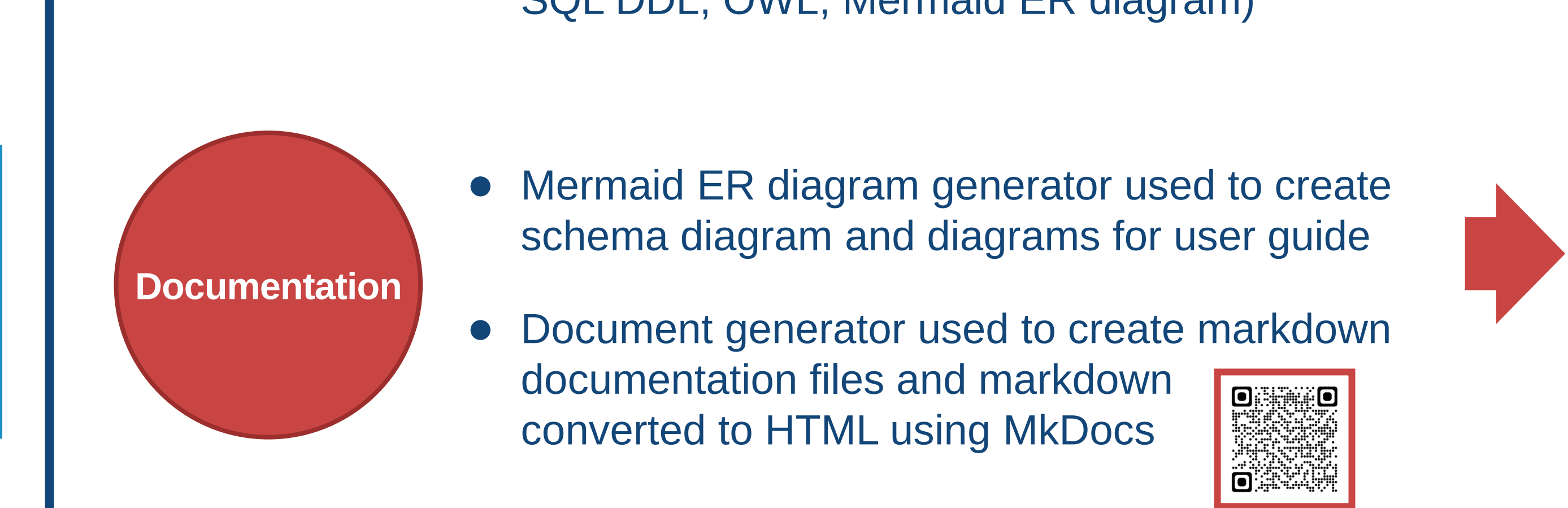
- 
- ARS schema developed in YAML
 - Schema linter used to check modeling best practices
 - Generators used to re-represent the model in other frameworks (JSON-Schema, GraphQL, SQL DDL, OWL, Mermaid ER diagram)

SQL DDL, OWL, Mermaid ER diagram)

- Mermaid ER diagram generator used to create schema diagram and diagrams for user guide
- Document generator used to create markdown documentation files and markdown converted to HTML using MkDocs

- Documentation

 - Mermaid ER diagram generator used to create schema diagram and diagrams for user guide
 - Document generator used to create markdown documentation files and markdown converted to HTML using MkDocs



Content Development

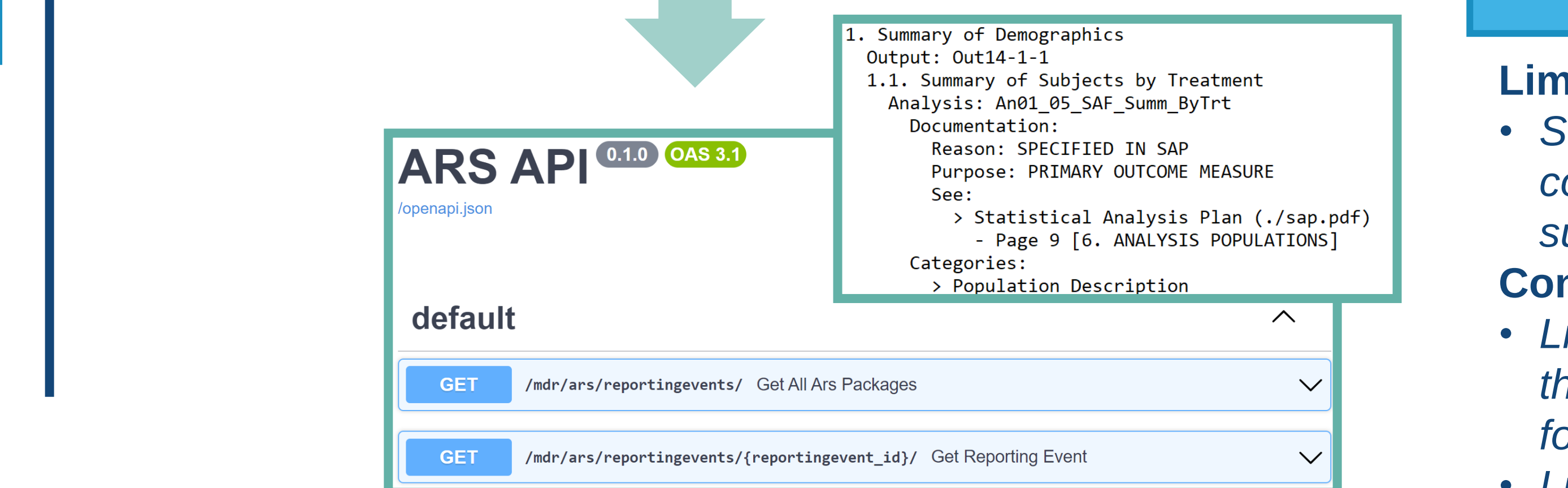
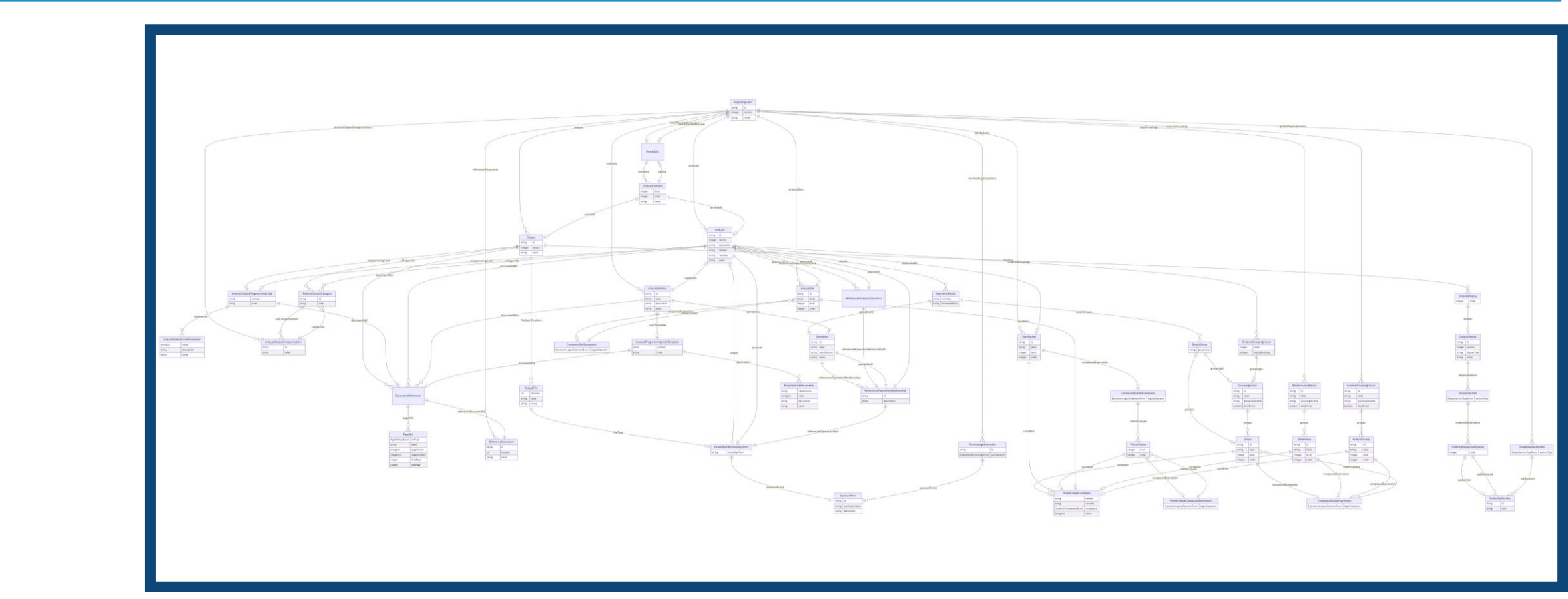
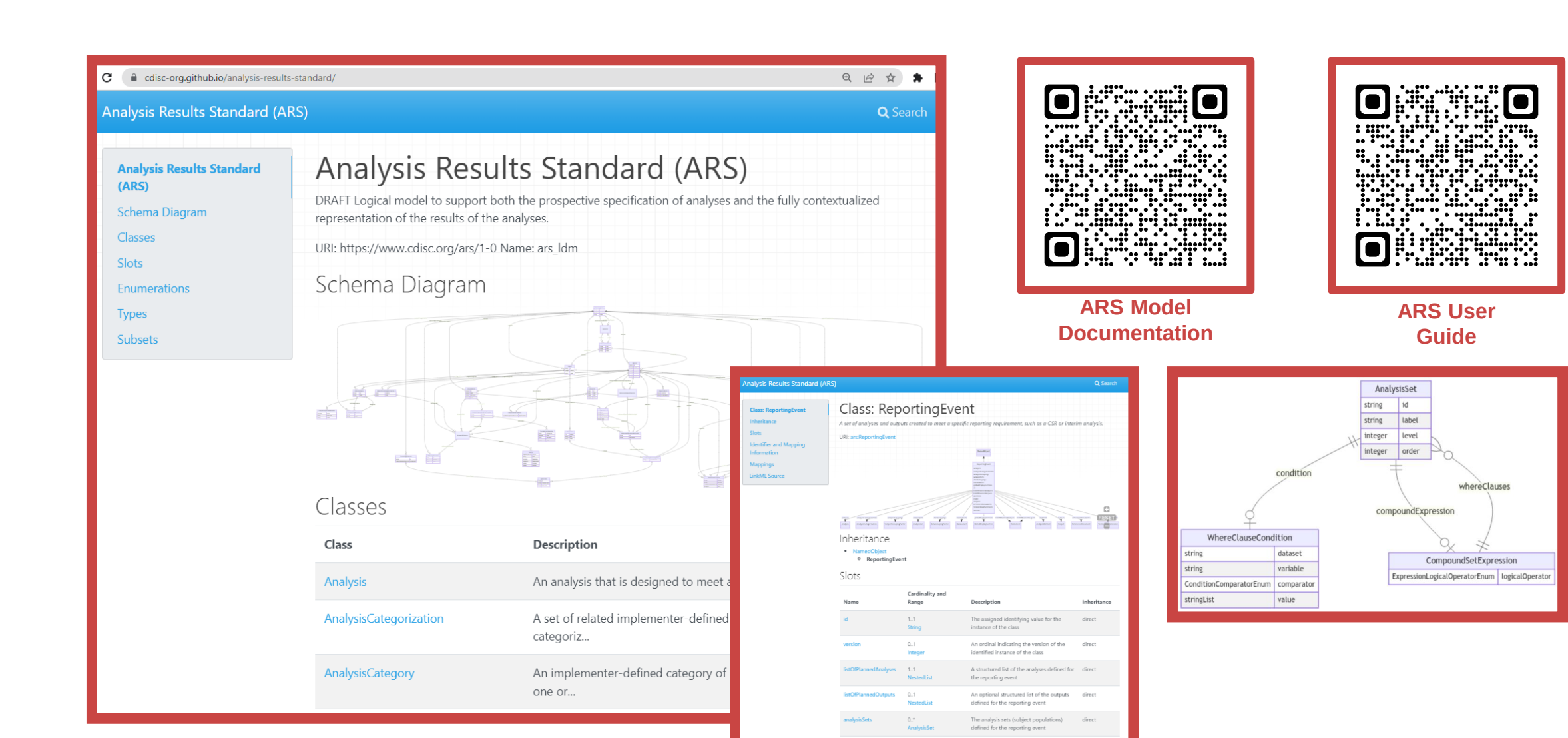
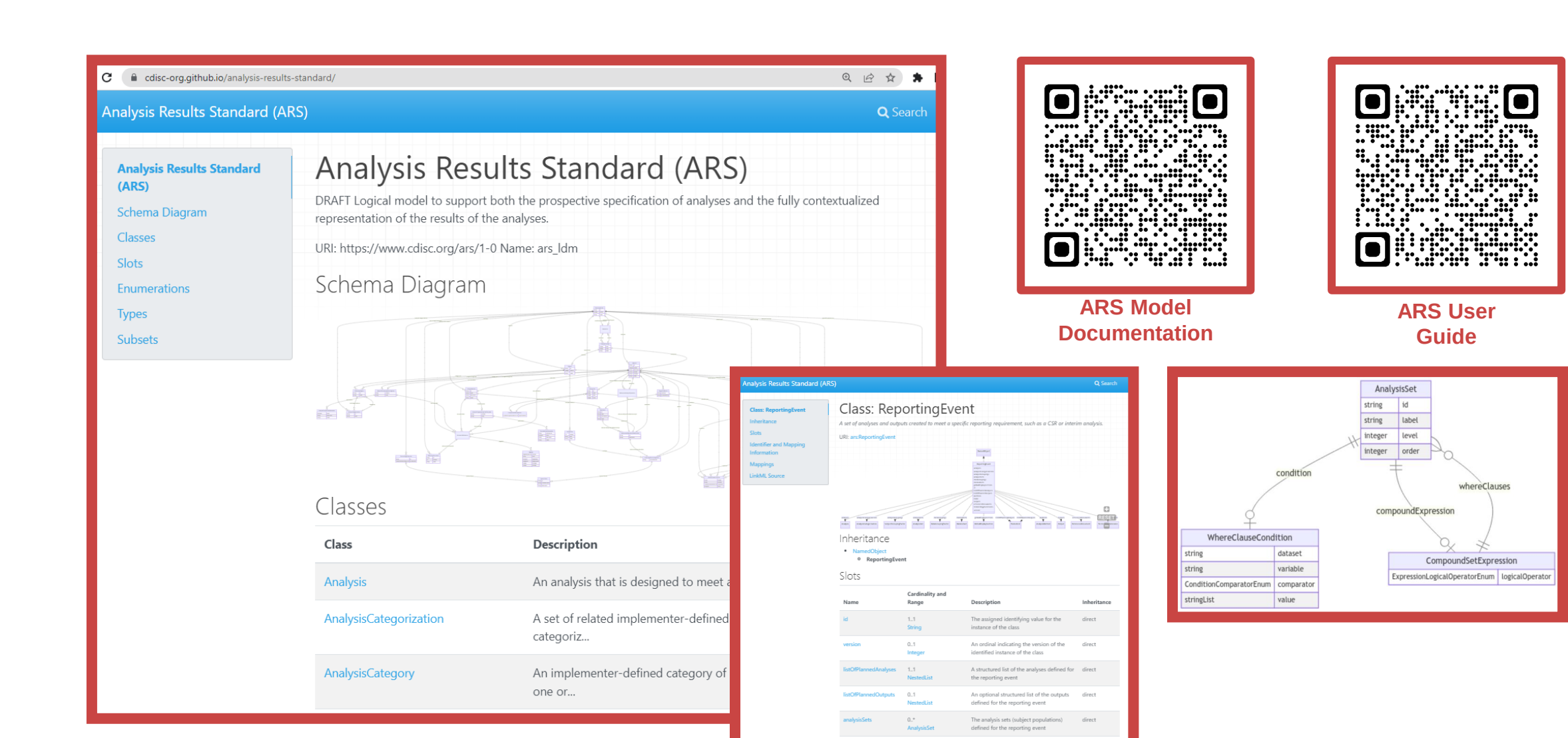
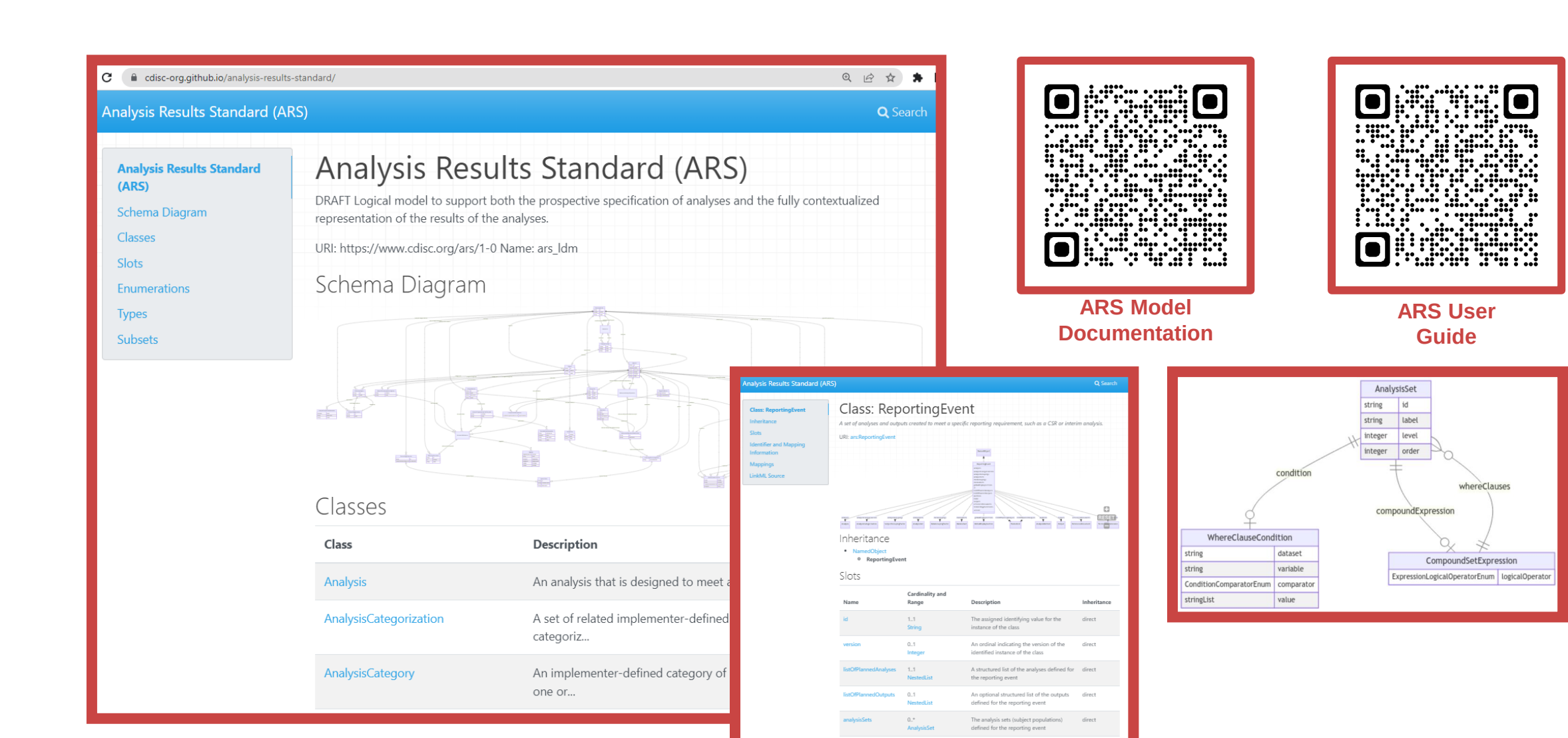
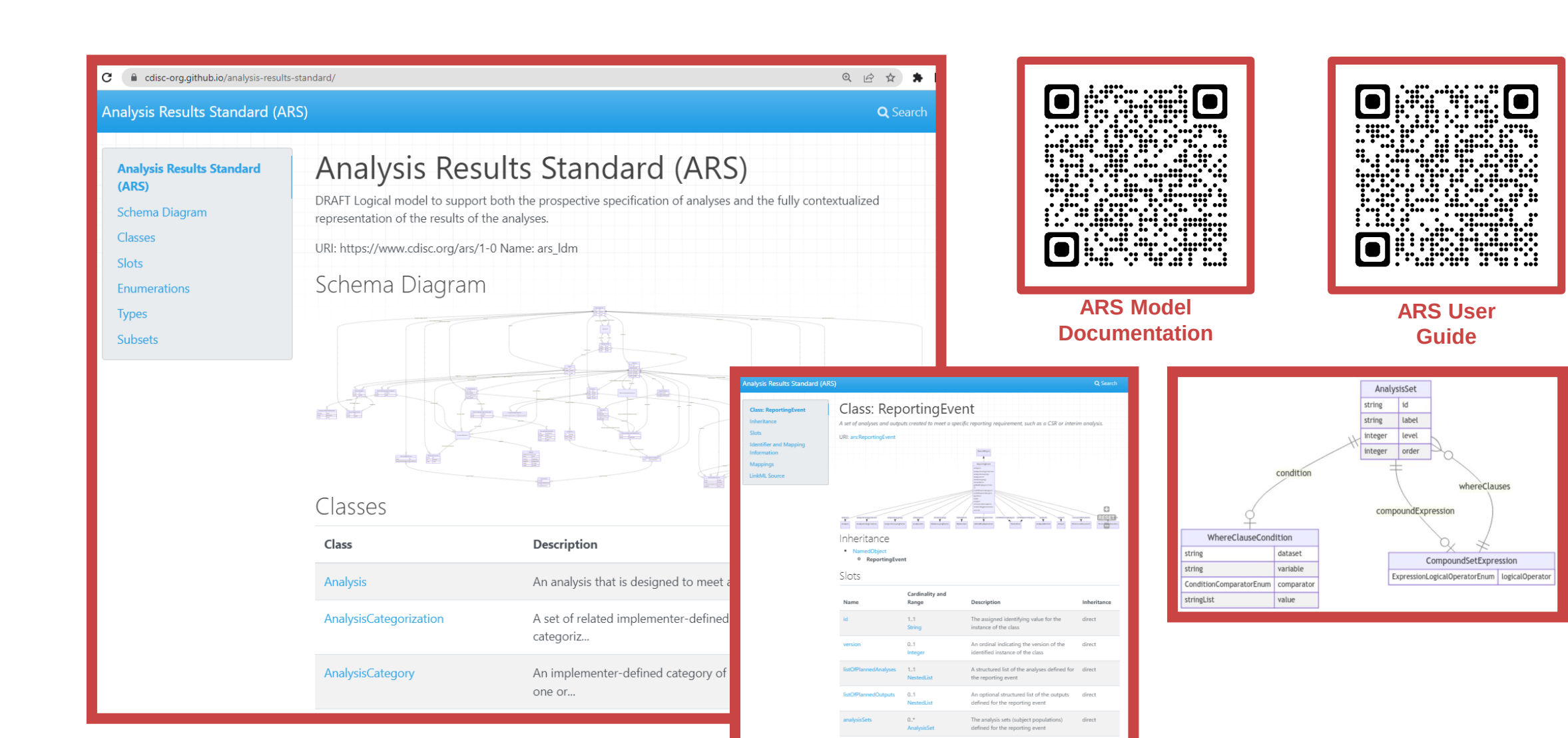
- Classes created by Python class generator used to convert Excel example content to YAML
- YAML to JSON converter used to create example JSON content

- Content Development**

 - Classes created by Python class generator used to convert Excel example content to YAML
 - YAML to JSON converter used to create example JSON content

```
graph LR; A((Implementations)) --> B[• Classes created by Python API class generator used to create human-readable summary of example analysis definitions]; A --> C[• Classes created by Pydantic class generator referenced for API development]
```

- ```
graph LR; A((Implementations)) --> B[• Classes created by Python API class generator used to create human-readable summary of example analysis definitions]; A --> C[• Classes created by Pydantic class generator referenced for API development]
```



The diagram illustrates the relationship between a table and a list of planned analyses. On the left, a table with 5 columns (A, B, C, D, E) and 6 rows is shown. The first row has columns A, B, C, D, and E. The second row has columns A, B, C, D, and E. The third row has columns A, B, C, D, and E. The fourth row has columns A, B, C, D, and E. The fifth row has columns A, B, C, D, and E. The sixth row has columns A, B, C, D, and E. A blue arrow points from the table to a box on the right. The box on the right contains a list of planned analyses. The list starts with '3 < listOfPlannedAnalyses:' followed by '4 < listItems:' and then a list of items. The first item is '- name: Summary of Demographics' followed by '5 < level: 1', '6 < order: 1', '7 < outputId: Out14-1-1', and '8 < sublist:'. The second item is '9 < listItems:' followed by '10 < - name: Summary of Subjects' and '11 < level: 2'. A blue arrow points from this box to another box on the right. The second box contains a JSON-like structure for 'listOfPlannedAnalyses' with two items. The first item has 'name: Summary of Demographics', 'level: 1', 'order: 1', 'outputId: Out14-1-1', and 'sublist' which is a list of items. The second item has 'name: Summary of Subjects by Treatment', 'level: 2', and 'sublist' which is a list of items.

|   | A     | B                                | C     | D                      | E         |
|---|-------|----------------------------------|-------|------------------------|-----------|
| 1 | level | name                             | order | analysisId             | outputId  |
| 2 | 1     | Summary of Demographics          | 1     |                        | Out14-1-1 |
| 3 | 2     | Summary of Subjects by Treatment | 1     | An01_05_SAF_Summ_ByTrt |           |
| 4 | 2     | Age                              | 2     |                        |           |
| 5 | 3     | Summary by Treatment             |       |                        |           |
| 6 | 3     | Comparison by Treatment          |       |                        |           |

```

3 < listOfPlannedAnalyses:
4 < listItems:
5 < - name: Summary of Demographics
6 < level: 1
7 < order: 1
8 < outputId: Out14-1-1
9 < sublist:
10 < listItems:
11 < - name: Summary of Subjects
12 < level: 2

```

```

4 "listOfPlannedAnalyses": {
5 "listItems": {
6 {
7 "name": "Summary of Demographics",
8 "level": 1,
9 "order": 1,
10 "outputId": "Out14-1-1",
11 "sublist": {
12 "listItems": {
13 {
14 "name": "Summary of Subjects by Treatment",
15 "level": 2,
16 "sublist": {
17 "listItems": {
18 {
19 "name": "Comparison by Treatment",
20 "level": 3,
21 "order": 3,
22 "outputId": "Out14-1-1",
23 "analysisId": "An01_05_SAF_Summ_ByTrt"
24 }
25 }
26 }
27 }
28 }
29 }
30 }
31 }
32 }

```

The diagram illustrates the relationship between a table and a list of planned analyses. On the left, a table with columns A, B, C, D, and E is shown. The rows are:

| A | B     | C                                | D     | E                      |
|---|-------|----------------------------------|-------|------------------------|
| 1 | level | name                             | order | analysisid             |
| 2 | 1     | Summary of Demographics          | 1     | Out14-1-1              |
| 3 | 2     | Summary of Subjects by Treatment | 1     | An01_05_SAF_Summ_ByTrt |
| 4 | 2     | Age                              | 2     |                        |
| 5 | 3     | Summary by Treatment             |       |                        |
| 6 | 3     | Comparison by Treatment          |       |                        |

A blue arrow points from the table to a list of planned analyses. The list is shown in a box with the following content:

```

3 ∨ listOfPlannedAnalyses:
4 ∨ listItems:
5 ∨ - name: Summary of Demographics
6 ∨ level: 1
7 ∨ order: 1
8 ∨ outputId: Out14-1-1
9 ∨ sublist:
10 ∨ listItems:
11 ∨ - name: Summary of Subjects
12 ∨ level: 2

```

A blue arrow points from the list of planned analyses to a JSON representation of the same data. The JSON is shown in a box with the following content:

```

4 "listOfPlannedAnalyses": {
5 "listItems": {
6 {
7 "name": "Summary of Demographics",
8 "level": 1,
9 "order": 1,
10 "outputId": "Out14-1-1",
11 "sublist": {
12 "listItems": {
13 {
14 "name": "Summary of Subjects by Treatment",
15 "level": 2,

```

The diagram illustrates the relationship between a table and a list of planned analyses. On the left, a table with columns A, B, C, D, and E is shown. The rows are:

| A | B     | C                                | D     | E                      |
|---|-------|----------------------------------|-------|------------------------|
| 1 | level | name                             | order | analysisid             |
| 2 | 1     | Summary of Demographics          | 1     | Out14-1-1              |
| 3 | 2     | Summary of Subjects by Treatment | 1     | An01_05_SAF_Summ_ByTrt |
| 4 | 2     | Age                              | 2     |                        |
| 5 | 3     | Summary by Treatment             |       |                        |
| 6 | 3     | Comparison by Treatment          |       |                        |

A blue arrow points from the table to a list of planned analyses. The list is structured as follows:

```

3 ∨ listOfPlannedAnalyses:
4 ∨ listItems:
5 ∨ - name: Summary of Demographics
6 ∨ level: 1
7 ∨ order: 1
8 ∨ outputId: Out14-1-1
9 ∨ sublist:
10 ∨ listItems:
11 ∨ - name: Summary of Subjects
12 ∨ level: 2

```

A blue arrow points from the list to a JSON representation of the list of planned analyses. The JSON is structured as follows:

```

4 "listOfPlannedAnalyses": {
5 "listItems": {
6 {
7 "name": "Summary of Demographics",
8 "level": 1,
9 "order": 1,
10 "outputId": "Out14-1-1",
11 "sublist": {
12 "listItems": {
13 {
14 "name": "Summary of Subjects by Treatment",
15 "level": 2,

```

## Summary

## Limitations

- ### Limitations
- *Some advanced modeling features such as “either/or” model components and built-in conformance constraints are not yet fully supported.*

## Conclusion

- ## Conclusion
- *LinkML is a simple, yet versatile and powerful modeling language and the associated utilities have facilitated and streamlined development for the ARS project.*
  - *LinkML's multi-platform toolchain will improve standards adoption.*