Package ‘r4subtrace’

May 9, 2026

Title Traceability Engine for Clinical Submission Readiness

Version 0.1.0

Description Quantifies and explains end-to-end traceability between clinical submission artifacts (ADaM (Analysis Data Model) outputs, derivations, SDTM (Study Data Tabulation Model) sources, specs, code). Builds trace models from metadata and mapping sheets, computes trace levels, and emits standardized R4SUB (R for Regulatory Submission) evidence table rows via 'r4subcore'.

License MIT + file LICENSE

URL `https://github.com/R4SUB/r4subtrace`

BugReports `https://github.com/R4SUB/r4subtrace/issues`

Depends R (>= 4.2)

Imports cli, dplyr, r4subcore, rlang, stringr, tibble

Suggests igraph, testthat (>= 3.0.0)

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.3.3

NeedsCompilation no

Author Pawan Rama Mali [aut, cre, cph]

Maintainer Pawan Rama Mali `<prm@outlook.in>`

Repository CRAN

Date/Publication 2026-03-03 21:20:16 UTC

Contents

`build_trace_model` *Build a Trace Model*

Description

Constructs a directed trace model (nodes + edges + diagnostics) from ADaM metadata, SDTM metadata, and an optional mapping sheet.

Usage

```
build_trace_model(
  adam_meta,
  sdtm_meta,
  mapping = NULL,
  spec = NULL,
  config = trace_config_default()
)
```

Arguments

`adam_meta`	A data.frame of ADaM variable metadata. Must contain `dataset` and `variable` columns.
`sdtm_meta`	A data.frame of SDTM variable metadata. Must contain `dataset` and `variable` columns.
`mapping`	An optional data.frame describing ADaM-to-SDTM mappings. Must contain `adam_dataset`, `adam_var`, `sdtm_domain`, `sdtm_var`.
`spec`	Reserved for future use (ADaM spec ingestion).
`config`	A `trace_config` object from `trace_config_default()`.

Value

A list of class `"trace_model"` with elements:

- `nodes`: tibble of asset nodes (datasets and variables)
- `edges`: tibble of relationships between nodes
- `diagnostics`: list of tibbles (`orphans`, `ambiguities`, `conflicts`)
- `config`: the configuration used

Examples

```
adam_meta <- data.frame(
  dataset = "ADSL", variable = c("STUDYID", "USUBJID", "AGE"),
  label = c("Study ID", "Unique Subject ID", "Age")
)
sdtm_meta <- data.frame(
  dataset = "DM", variable = c("STUDYID", "USUBJID", "AGE"),
  label = c("Study ID", "Unique Subject ID", "Age")
)
map <- data.frame(
  adam_dataset = "ADSL", adam_var = c("STUDYID", "USUBJID", "AGE"),
  sdtm_domain = "DM",   sdtm_var = c("STUDYID", "USUBJID", "AGE")
)
tm <- build_trace_model(adam_meta, sdtm_meta, mapping = map)
tm$nodes
tm$edges
```

`compute_trace_levels` *Compute Trace Levels for ADaM Variables*

Description

Assigns a traceability level (L0–L3) to each ADaM variable in the trace model based on available mapping, derivation text, and confidence scores.

Usage

```
compute_trace_levels(trace_model)
```

Arguments

`trace_model`	A `trace_model` object from `build_trace_model()`.

Details

Trace levels:

- **L0**: No mapping and no derivation text.
- **L1**: Derivation text present but no SDTM mapping.
- **L2**: Mapping to SDTM variable/domain exists.
- **L3**: Mapping exists AND (confidence >= threshold OR derivation text present alongside mapping).

Value

A tibble with columns: `adam_dataset`, `adam_var`, `trace_level`, `has_mapping`, `has_derivation_text`, `n_candidates`, `max_confidence`.

Examples

```
adam_meta <- data.frame(
  dataset = "ADSL", variable = c("STUDYID", "USUBJID", "AGE", "AGEGR1"),
  label = c("Study ID", "Unique Subject ID", "Age", "Age Group")
)
sdtm_meta <- data.frame(
  dataset = "DM", variable = c("STUDYID", "USUBJID", "AGE"),
  label = c("Study ID", "Unique Subject ID", "Age")
)
map <- data.frame(
  adam_dataset = "ADSL", adam_var = c("STUDYID", "USUBJID", "AGE"),
  sdtm_domain = "DM",   sdtm_var = c("STUDYID", "USUBJID", "AGE"),
  confidence = c(1.0, 1.0, 0.9)
)
tm <- build_trace_model(adam_meta, sdtm_meta, mapping = map)
compute_trace_levels(tm)
```

`print.trace_model` *Print Trace Model*

Description

Print Trace Model

Usage

```
## S3 method for class 'trace_model'
print(x, ...)
```

Arguments

`x`	A `trace_model` object.
`...`	Ignored.

Value

Invisibly returns `x`. Called for its side effect of printing a summary of the trace model (ADaM variable count, SDTM variable count, edge count, orphan count, and ambiguity count) to the console.

`trace_config_default` *Default Trace Configuration*

Description

Returns a list of default configuration values for trace model building and evidence emission.

Usage

```
trace_config_default(
  severity_by_level = c(L0 = "high", L1 = "medium", L2 = "low", L3 = "info"),
  result_by_level = c(L0 = "fail", L1 = "warn", L2 = "warn", L3 = "pass"),
  confidence_threshold_L3 = 0.8,
  uppercase_datasets = TRUE
)
```

Arguments

`severity_by_level`	Named character vector mapping trace levels to severity.
`result_by_level`	Named character vector mapping trace levels to result.
`confidence_threshold_L3`	Numeric threshold for L3 classification. A mapping must have confidence >= this value to qualify for L3.
`uppercase_datasets`	Logical; if `TRUE`, dataset and domain names are uppercased during canonicalization.

Value

A list of class `"trace_config"` with elements: `severity_by_level`, `result_by_level`, `confidence_threshold_L3`, `uppercase_datasets`.

Examples

```
cfg <- trace_config_default()
cfg$severity_by_level

# Override a single setting
cfg2 <- trace_config_default(confidence_threshold_L3 = 0.9)
```

`trace_indicator_scores` *Compute Trace Indicator Scores*

Description

Computes summary metrics from evidence rows generated by `trace_model_to_evidence()`. Returns key traceability indicators.

Usage

```
trace_indicator_scores(evidence)
```

Arguments

`evidence`	A data.frame of evidence rows (must contain `indicator_id` and `metric_value` columns).

Value

A tibble with columns: `indicator`, `value`, `description`.

Examples

```
library(r4subcore)
ctx <- r4sub_run_context(study_id = "TEST001", environment = "DEV")
adam_meta <- data.frame(
  dataset = "ADSL", variable = c("STUDYID", "AGE", "AGEGR1"),
  label = c("Study ID", "Age", "Age Group")
)
sdtm_meta <- data.frame(
  dataset = "DM", variable = c("STUDYID", "AGE"),
  label = c("Study ID", "Age")
)
map <- data.frame(
  adam_dataset = "ADSL", adam_var = c("STUDYID", "AGE"),
  sdtm_domain = "DM",   sdtm_var = c("STUDYID", "AGE")
)
tm <- build_trace_model(adam_meta, sdtm_meta, mapping = map)
ev <- trace_model_to_evidence(tm, ctx = ctx)
trace_indicator_scores(ev)
```

`trace_model_to_evidence`

Convert Trace Model to R4SUB Evidence

Description

Emits evidence rows compatible with `r4subcore::validate_evidence()` for each ADaM variable's trace level, plus diagnostic rows for orphans, ambiguities, and conflicts.

Usage

```
trace_model_to_evidence(
  trace_model,
  ctx,
  source_name = "r4subtrace",
  source_version = NULL
)
```

Arguments

`trace_model`	A `trace_model` object from `build_trace_model()`.
`ctx`	An `r4sub_run_context` from `r4subcore::r4sub_run_context()`.
`source_name`	Character; the name of the evidence source.
`source_version`	Character or `NULL`; version of the source.

Value

A data.frame of evidence rows passing `r4subcore::validate_evidence()`.

Examples

```
library(r4subcore)
ctx <- r4sub_run_context(study_id = "TEST001", environment = "DEV")
adam_meta <- data.frame(
  dataset = "ADSL", variable = c("STUDYID", "AGE"),
  label = c("Study ID", "Age")
)
sdtm_meta <- data.frame(
  dataset = "DM", variable = c("STUDYID", "AGE"),
  label = c("Study ID", "Age")
)
map <- data.frame(
  adam_dataset = "ADSL", adam_var = c("STUDYID", "AGE"),
  sdtm_domain = "DM",   sdtm_var = c("STUDYID", "AGE")
)
tm <- build_trace_model(adam_meta, sdtm_meta, mapping = map)
ev <- trace_model_to_evidence(tm, ctx = ctx)
r4subcore::validate_evidence(ev)
```

`validate_mapping` *Validate Trace Mapping*

Description

Checks that a mapping data.frame contains the required columns (`adam_dataset`, `adam_var`, `sdtm_domain`, `sdtm_var`) and canonicalizes names, trims whitespace, and optionally uppercases dataset/domain names.

Usage

```
validate_mapping(df, uppercase_datasets = TRUE)
```

Arguments

`df`	A data.frame describing ADaM-to-SDTM variable mappings.
`uppercase_datasets`	Logical; if `TRUE`, uppercases `adam_dataset` and `sdtm_domain`. Default `TRUE`.

Value

A tibble with canonicalized column names and values.

Examples

```
map <- data.frame(
  ADAM_DATASET = "adsl", ADAM_VAR = "AGE",
  SDTM_DOMAIN = "dm", SDTM_VAR = "AGE"
)
validate_mapping(map)
```

`validate_metadata` *Validate Dataset Metadata*

Description

Checks that an ADaM or SDTM metadata data.frame contains the required columns (`dataset`, `variable`) and canonicalizes column names to lowercase.

Usage

```
validate_metadata(df, kind = c("adam", "sdtm"))
```

Arguments

`df`	A data.frame of dataset metadata.
`kind`	Character; `"adam"` or `"sdtm"`. Used in error messages only.

Value

A tibble with canonicalized column names.

Examples

```
meta <- data.frame(DATASET = "ADSL", VARIABLE = "SUBJID", LABEL = "Subject ID")
validate_metadata(meta, kind = "adam")
```

Index