Overview

The R package tracelib contains functions for capturing data workflow metadata, in particular traceability information and storing this metadata to a metadata repository in a database or to a .json file.

Moreover action and file metadata can be connected to metadata of the project/activity where they belong to.

Workflows are considered to be action-file-networks / graphs.

Files of interest are just those which are stored into a version control system (just SVN in version 1). Temporary files are of minor interest.

Actions can mainly be seen as executions of functions which consume and produce files.
R Functions on an appropriate level can be marked for capturing their execution as an action – this will be explained in detail later.

File metadata can be captured by using wrapper function for reading and writing files. Alternatively functions for capturing the metadata can be called explicitly using arguments like filepath or the file metadata itself.

Remarks on the metadata

The metadata is collected in global objects, which can be written to a database or to a .json file.

As long as the files are stored and checked in a local SVN folder, the required SVN repository metadata can be captured automatically and metadata can be stored directly to the database; otherwise the metadata must be stored into a .json file during execution of the workflow and after check in of the output files an additional function needs to be called to add the SVN specific metadata and store it to the database.

The metadata of projects/activities is supposed to be stored directly in the table activity in the database including a activity specific SVN repository path, which can be used to link the file and action metadata automatically to the corresponding activity.

Detailed description

All tracelib functions to be used by the R programmer have the prefix t (for traceability).

There are 3 groups of “t-functions”:

- Functions to be called once per data workflow
- Functions to define actions
- Functions to capture reading and writing of files

In the following these functions are described. For an illustrative example see the next section.

Functions to be called once per data workflow

In the begin of a data workflow performed by a R function (which can call an arbitrary hierarchy of subfunctions) once an initialization method **tStartMetadataCapture** must be called, which initializes global objects for file, action and system infos.

Two parameters need special explanation, see package documentation for other parameters.

By the boolean parameter **metaDataCapture** (default: true) can be controlled, if all tracelib functions capture metadata or not. E.g. for validation purposes this parameter can be set to false to prove, that results of the users code are not affected by calling the tracelib functions.

By the parameter **filePath** a path to a file or folder inside the project/activity specific folder should be passed to allow identification of the corresponding activity for actions of this data workflow (using the table activity).

Like function calls also actions can be organized hierarchically, and by execution of **tStartMetadataCapture** a top level action is implicitely created, which groups all actions created in this workflow.

Counterpart to **tStartMetadataCapture** is **tEndMetadataCapture**, which must be called once at the end of the data workflow. By this method is done some housekeeping, all collected metadata is written and in case of errors within tracelib an errorLog is written.

By the parameter **storageMode** (= "DB" or "File") can be controlled, whether the metadata are written to the database or to a file.

By the parameter **outputFolder** a folderPath can be passed where in case of errors the errorLog and in case of storageMode = "File" the .json file is written.

For documentation of other parameters see package documentation.

Functions to define actions

As mentioned before R functions on an appropriate level can be marked for capturing their execution as an action. The level of the function is appropriate, if within the function (directly or by subfunctions) one or more files are read and used to produce and write another (nontemporary) file(s).

At the beginning of such a function the function **tStartAction** should be called and a **actionType** should be passed (one of the strings "DataImport", "AnalysisFileGeneration", "Simulation", "Analysis", "TLFGeneration", "ReportGeneration", "Edit","Copy","Move", "Run", "Other").

At the end of the function **tEndAction** should be called – without any argument.

By these calls the corresponding metadata will be captured and collected during execution.

As already mentioned actions can be organized in a hierarchical way, so for instance a workflow "Run" action can contain/call multiple sub-actions of type "AnalysisFileGeneration", "Simulation", "Analysis", "TLFGeneration".

Functions to capture reading and writing of files

As mentioned before file metadata can be captured by using wrapper function for reading and writing files, or functions for capturing the metadata are explicitly called passing a filepath or the metadata itself.

When reading or writing any files of interest (mostly files within an SVN checkout folder), the programmer should call the function **tStoreFileMetaData** and pass a parameter **access** = "read" or "write" respectively.

Often it is sufficient to pass the parameter **filepath**, then inside the function the metadata for the file including the SVN related metadata can be captured, the current action is determined and the file and action metadata are linked.

In some cases, e.g. when accessing files from or to different data repositories the metadata cannot be extracted so easily and must be passed explicitly to that function, see package description for those parameters.

For convenience some wrapper functions are implemented to cover the most often used read and write operations.

For example **tReadCsv** and **tWriteCsv** just call the functions read.csv and write.csv (with the same parameters like the original functions) and call in addition **tStoreFileMetaData**.

For storing metadata, which is just explicitly given, a function **tStoreActionMetadata** can be used to pass that metadata (metadata for the files can be stored within a dataframe created by **createFilesMetadataFrame**).
A specialized version for the explicit metadata of data file import from one data repository into another is **tImportFile**.

Example

The following example shows the usage of the described “t-functions”.

Assume there is some code for executing some computation based on two files which creates another file using a simple helper function for import of files:

```
add2FilesUsingLoad = function(inFile1,inFile2,outFile){
  tStartAction()

  allInputs <- load2Files(inFile1,inFile2)
  in1 = allInputs[["input1"]]
  in2 = allInputs[["input2"]]

  #### do some calculation ####
  outX = in1$x + in2$x
  outY = in1$y + in2$y
  outText = paste0(in1$text, in2$text)
  out <- data.frame(x=outX, y=outY, text=outText)
  ########

  tWriteCsv(out, outFile)

  success = 1

  tEndAction()
  return(success)
}
```

The helper function load_input_files, which calls t_read_csv , is not defined as an action.

```
load2Files = function(inFile1,inFile2){
  in1 = tReadCsv(inFile1)
  in2 = tReadCsv(inFile2)

  inputs = list( input1 = in1, input2 = in2)
  return(inputs)
}
```

And this code is called within the following data workflow function:

```
workflowAdd2UsingLoad <- function(dataFolder){
  tStartMetadataCapture(metaDataCapture = T , filePath = dataFolder)

  inFile1 <- paste0(dataFolder, "input1.csv")
  inFile2 <- paste0(dataFolder, "input2.csv")
  outFile1 <- paste0(dataFolder, "output1.csv")

  add2FilesUsingLoad(inFile1,inFile2,outFile1)

  tEndMetadataCapture(storageMode = "File",
                      outputFolder = dataFolder,
                      jsonFileName = "Workflow_add2_using_load”)
}
```

When calling `workflowAdd2UsingLoad` passing a SVN checkout folder the following metadata file is created for a toplevel action (call of `workflowAdd2UsingLoad`) and another action (call of `add2FilesUsingLoad`) which reads to input files and writes one output file.

```
{
 "actionInfos": {
  "1": {
   "actionId": [1],
   "actionName": ["workflowAdd2UsingLoad"],
   "actionType": ["Run"],
   "superActionId": ["NaN"],
   "scriptFileInfo": ["c33580238a48e681760386ef9754ff9e;testtracelib"],
   "executionStarted": ["2019-11-21T14:02:49"],
   "executed": ["2019-11-21T14:02:51"],
   "executedBy": ["USERNN"],
   "activityId": ["2"],
   "systemId": [1],
   "groupId": ["1"],
   "description": ["testtracelib:workflowAdd2UsingLoad/tracelib:tStartMetadataCapture"]
  },
  "2": {
   "actionId": [2],
   "actionName": ["add2FilesUsingLoad"],
   "actionType": ["Other"],
   "superActionId": [1],
   "scriptFileInfo": ["c33580238a48e681760386ef9754ff9e;testtracelib"],
   "inputFileInfos":
["md5:71b2bbc467b18bb4ba0dab92;C:/Projects/SVN/traceLib/trunk/tests/test_2/data/input1.csv",
"md5:bbe86b08323acb3b49eb14cd;C:/Projects/SVN/traceLib/trunk/tests/test_2/data/input2.csv"],
   "outputFileInfos":
["md5:564141d8bb57396860b4e03e;C:/Projects/SVN/traceLib/trunk/tests/test_2/data/output1.csv"],
   "executionStarted": ["2019-11-21T14:02:49"],
   "executed": ["2019-11-21T14:02:51"],
   "executedBy": ["USERNN"],
   "activityId": ["2"],
   "systemId": [1],
   "groupId": ["1"],
   "description":
["testtracelib:workflowAdd2UsingLoad/testtracelib:add2FilesUsingLoad/tracelib:tStartAction"]
  }
 },
 "fileInfos": {
  "c33580238a48e681760386ef9754ff9e;testtracelib": {
   "fileHash": ["c33580238a48e681760386ef9754ff9e"],
   "repoType": ["Package"],
   "repoPath": ["testtracelib"],
   "repoVersion": ["1.0.1"],
   "repoFileStatus": [""],
   "repoModified": ["2019-11-21T13:02:01"],
   "repoModifiedBy": ["Package Author NN"],
   "localFilePath": [""],
   "localModified": ["1970-01-01T01:00:00"],
   "localModifiedBy": [""],
   "fileType": ["Package"],
   "groupId": [""],
   "activityId": [""],
   "qualityChecked": [false],
   "qualityComputed": [false],
   "description": ["testtracelib 1.0.1 R 3.5.2; ; 2019-11-21 13:02:01 UTC; windows | "],
   "actionInfo": [""],
   "accessInfo": ["read"],
   "repoRoot": [""]
  },
  "md5:71b2bbc467b18bb4ba0dab92;C:/Projects/SVN/traceLib/trunk/tests/test_2/data/input1.csv": {
   "fileHash": ["md5:71b2bbc467b18bb4ba0dab92"],
   "repoType": ["SVN"],
   "repoPath": ["https://by-spmsvnprd/svn/traceLib/trunk/tests/test_2/data/input1.csv"],
   "repoVersion": ["11"],
   "repoFileStatus": [""],
   "repoModified": ["2019-08-22T13:02:07"],
   "repoModifiedBy": ["USERNN"],
   "localFilePath": ["C:/Projects/SVN/traceLib/trunk/tests/test_2/data/input1.csv"],
   "localModified": ["2019-07-09T11:38:15"],
   "localModifiedBy": [""],
   "fileType": ["Data"],
   "groupId": [""],
```

```
   "activityId": [""],
   "qualityChecked": [false],
   "qualityComputed": [false],
   "description": [""],
   "actionInfo": ["2 add2FilesUsingLoad"],
   "accessInfo": ["read"],
   "repoRoot": ["https://by-spmsvnprd/svn/traceLib"]
  },
  "md5:bbe86b08323acb3b49eb14cd;C:/Projects/SVN/traceLib/trunk/tests/test_2/data/input2.csv": {
   "fileHash": ["md5:bbe86b08323acb3b49eb14cd"],
   "repoType": ["SVN"],
   "repoPath": ["https://by-spmsvnprd/svn/traceLib/trunk/tests/test_2/data/input2.csv"],
   "repoVersion": ["11"],
   "repoFileStatus": [""],
   "repoModified": ["2019-08-22T13:02:07"],
   "repoModifiedBy": ["USERNN"],
   "localFilePath": ["C:/Projects/SVN/traceLib/trunk/tests/test_2/data/input2.csv"],
   "localModified": ["2018-08-20T10:36:19"],
   "localModifiedBy": [""],
   "fileType": ["Data"],
   "groupId": [""],
   "activityId": [""],
   "qualityChecked": [false],
   "qualityComputed": [false],
   "description": [""],
   "actionInfo": ["2 add2FilesUsingLoad"],
   "accessInfo": ["read"],
   "repoRoot": ["https://by-spmsvnprd/svn/traceLib"]
  },
  "md5:564141d8bb57396860b4e03e;C:/Projects/SVN/traceLib/trunk/tests/test_2/data/output1.csv": {
   "fileHash": ["md5:564141d8bb57396860b4e03e"],
   "repoType": ["SVN"],
   "repoPath": ["https://by-spmsvnprd/svn/traceLib/trunk/tests/test_2/data/output1.csv"],
   "repoVersion": ["11"],
   "repoFileStatus": [""],
   "repoModified": ["2019-08-22T13:02:07"],
   "repoModifiedBy": ["USERNN"],
   "localFilePath": ["C:/Projects/SVN/traceLib/trunk/tests/test_2/data/output1.csv"],
   "localModified": ["2019-11-21T14:02:50"],
   "localModifiedBy": [""],
   "fileType": ["Data"],
   "groupId": [""],
   "activityId": ["2"],
   "qualityChecked": [false],
   "qualityComputed": [false],
   "description": [""],
   "actionInfo": ["2 add2FilesUsingLoad"],
   "accessInfo": ["write"],
   "repoRoot": ["https://by-spmsvnprd/svn/traceLib"]
  }
 },
 "systemInfo": [
  {
   "systemId": [1],
   "systemName": ["ABCC012345"],
   "operatingSystem": ["Windows"],
   "osVersion": ["10 x64"],
   "RVersion": ["3.5.2"],
   "validated": [false],
   "description": [""]
  }
 ]
}
```