

TNA ACTIVITY REPORT

Building an R Package and Web Application to Interact Within a Digital Ecosystem for Literary Studies

Author: Pozdniakov, Ivan

Current position: Independent researcher

Affiliation: none

Host institution: University of Potsdam, Germany

Mentor(s): Prof. Peer Trilcke

Period of stay: 2022-04-15 to 2022-06-14

Overview of my TNA stay

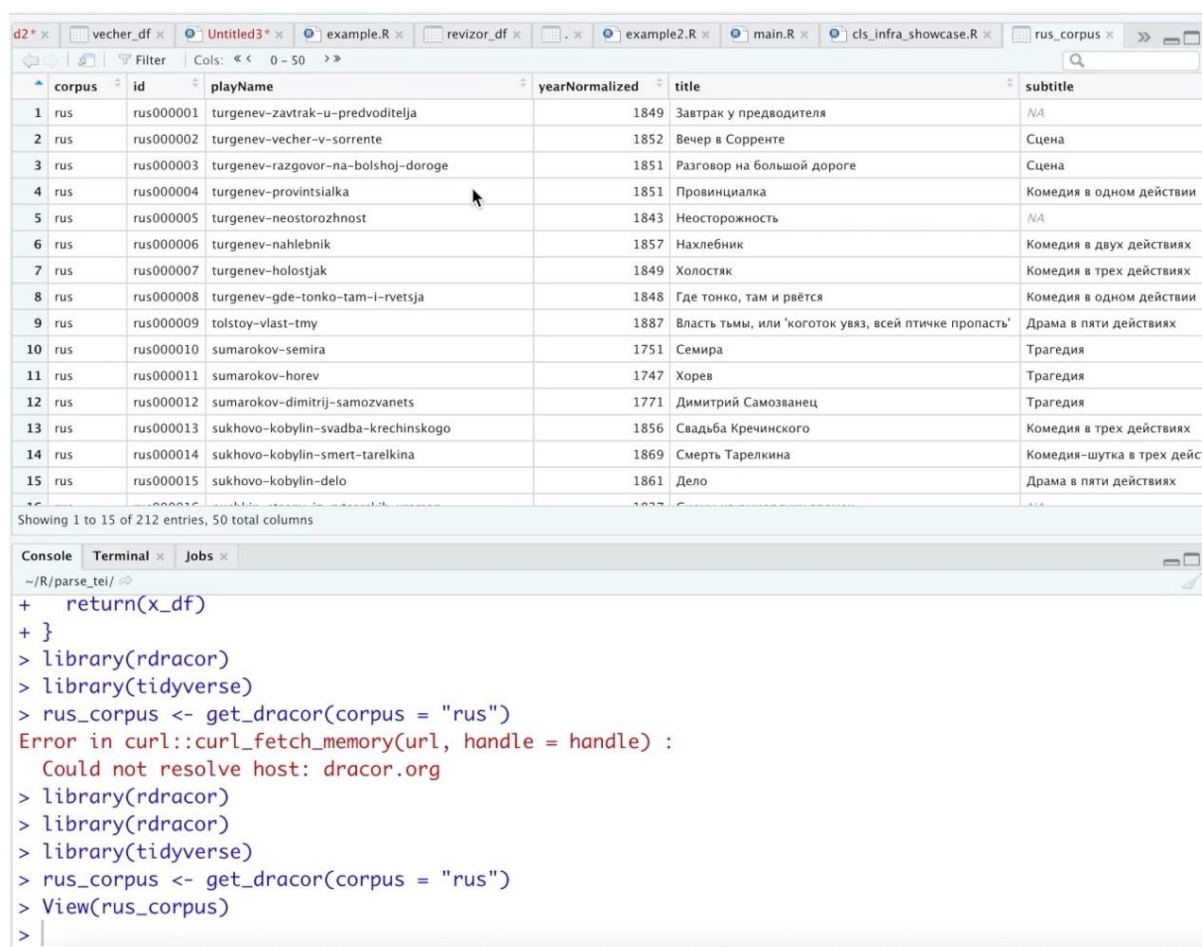
My stay at the University of Potsdam, Germany from April 15th, 2022 to June 14th, 2022 was dedicated to the development of “rdracor”, a package in the programming language R, that could be used to analyze the literary data provided by the Drama Corpora Project (DraCor). DraCor is a prototype of the Programmable Corpora concept that is explored within the CLS Infra Project in Work Package 7 led by the University of Potsdam and provides data on theater plays in several European languages. Every play is encoded using TEI that helps to analyze different plays and the whole corpora by access to the project’s API. I am excited by insights given by the distant reading approach and I believe that providing a complex but easy-to-use infrastructure for computational research in Literary Studies will not only make it less challenging and faster but also will lead to new hypotheses and ways to test them. Thus my particular interest was to develop a package that could be used by the R community within CLS in native way to the programming language and build on commonly used data structures, like datatables. For the package I could build upon an early prototype that I created in the course of my involvement in a student research group at my University some years ago.



During my stay in Potsdam, in several meetings with my TNA mentor, Prof. Peer Trilcke and team members of CLS INFRA project and the Potsdam Network for Digital Humanities we went through the functions provided by the DraCor API and discussed which of them needed to be represented in the package. I then updated some of the legacy code to the current version of API, but also focused on making the code more independent of possible future changes of the API by implementing a generic function that could send requests not only to the current production version of the platform, but also to local instances running in a Docker container.

First impressions of rdracor (still in development)

The main feature of rdracor is that it translates the DraCor data to R data objects which are familiar to R users and convenient to process in the R environment. For example, the metadata for all plays in each corpus can be retrieved as a dataframe with a single 'get_dracor' command:



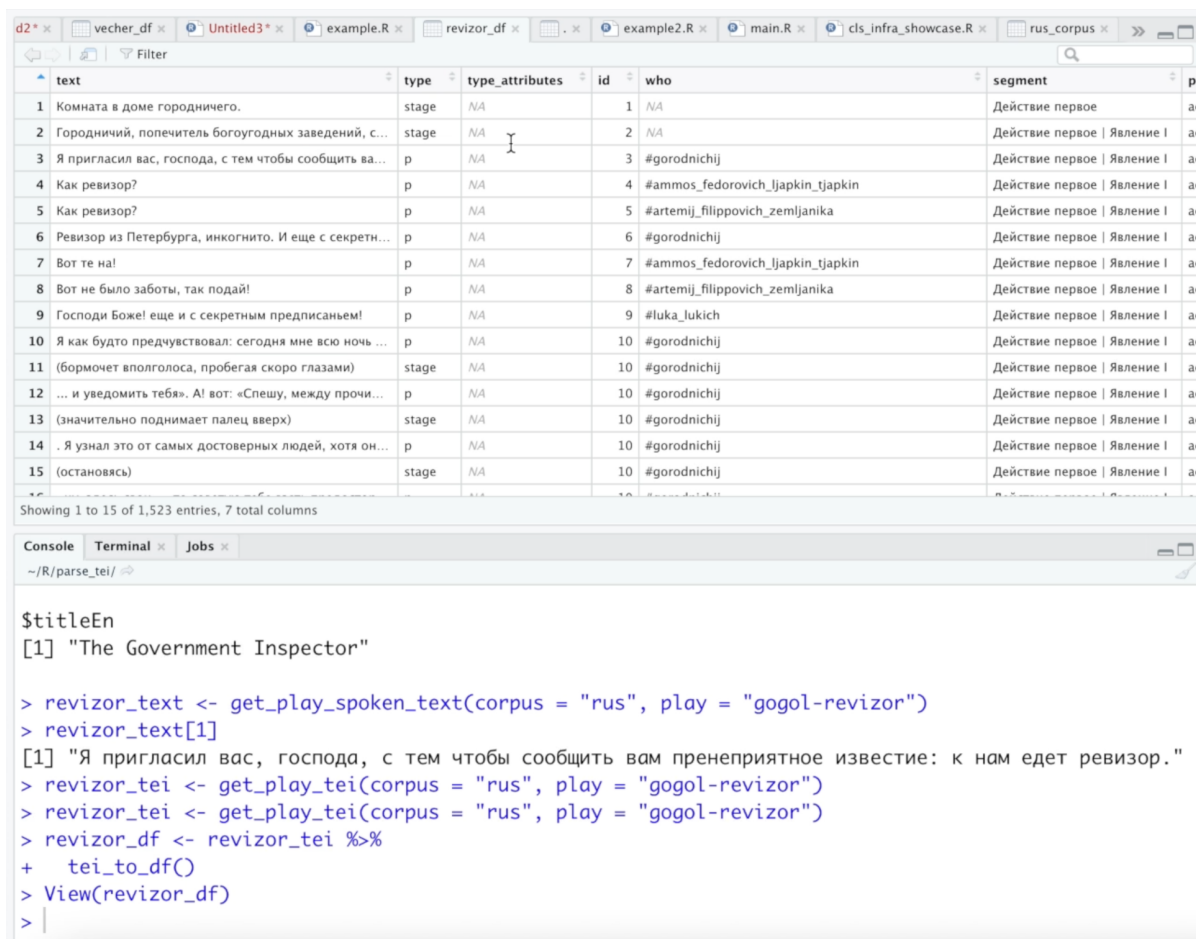
The screenshot shows an RStudio window with a dataframe containing play metadata. The dataframe has columns: corpus, id, playName, yearNormalized, title, and subtitle. Below the dataframe, the console window shows the following R code and error message:

```

+   return(x_df)
+ }
> library(rdracor)
> library(tidyverse)
> rus_corpus <- get_dracor(corpus = "rus")
Error in curl::curl_fetch_memory(url, handle = handle) :
  Could not resolve host: dracor.org
> library(rdracor)
> library(rdracor)
> library(tidyverse)
> rus_corpus <- get_dracor(corpus = "rus")
> View(rus_corpus)
>
  
```

Moreover, the TEI source for each play can also be retrieved as a line-by-line dataframe using Rdracor package. I believe this to be the most considerable advantage of the package,

since it converts an XML (which is usually not easy to process with data analysis tools) to a dataframe without any loss of information. In this dataframe, we still have all the data about who said what and in which order.



The screenshot shows an RStudio window with a dataframe containing 15 rows of play segments. The columns are: text, type, type_attributes, id, who, segment, and position. The terminal window below shows the following R code and output:

```

$titleEn
[1] "The Government Inspector"

> revizor_text <- get_play_spoken_text(corpus = "rus", play = "gogol-revizor")
> revizor_text[1]
[1] "Я пригласил вас, господа, с тем чтобы сообщить вам пренеприятное известие: к нам едет ревизор."
> revizor_tei <- get_play_tei(corpus = "rus", play = "gogol-revizor")
> revizor_tei <- get_play_tei(corpus = "rus", play = "gogol-revizor")
> revizor_df <- revizor_tei %>%
+   tei_to_df()
> View(revizor_df)
>

```

And as one of DraCor's main purposes is network analysis, one can of course use `rdracor` to retrieve networks of characters for the DraCor plays and analyze them making use of the rich network analysis infrastructure available in R. Here is an example of a network visualization with `rdracor`:

The screenshot displays the RStudio environment with the following components:

- Source Editor:** Contains R code for data retrieval and graph creation:


```

1 library(rdracor)
2 library(tidyverse)
3
4 rus_corpus <- get_dracor(corpus = "rus")
5 revizor_igraph <- play_igraph(corpus = "rus", play = "gogol-revizor")
6 plot(revizor_igraph)
7
8 get_play_metadata(corpus = "rus", play = "gogol-revizor")
9
10 revizor_text <- get_play_spoken_text(corpus = "rus", play = "gogol-revizor")
11 revizor_text[1]
12
13 revizor_tei <- get_play_tei(corpus = "rus", play = "gogol-revizor")
14

```
- Console:** Shows the execution of the code above. An error message is visible:


```

> library(rdracor)
> library(tidyverse)
> rus_corpus <- get_dracor(corpus = "rus")
Error in curl::curl_fetch_memory(url, handle = handle) :
  Could not resolve host: dracor.org
> library(rdracor)
> library(tidyverse)
> rus_corpus <- get_dracor(corpus = "rus")
> View(rus_corpus)
> revizor_igraph <- play_igraph(corpus = "rus", play = "gogol-revizor")
> plot(revizor_igraph)
>

```
- Environment:** Lists objects in the workspace:

revizor_ig...	List of 10
rus_corpus	212 obs. of 50 variables
- Viewer:** Displays a network graph with nodes of varying sizes and colors (green, purple, blue) connected by edges. Some nodes are labeled with Russian text:
 - Анна Каренина
 - Горький
 - Хлестаков