UPPSALA
UNIVERSITET

Maintaining Nuclear Data Evaluation tools – collaboration, portability, and continuity – Uppsala University experience

Georg Schnabel, Henrik Sjöstrand
Uppsala University

UPPSALA
UNIVERSITET

It is mostly about communication

UPPSALA UNIVERSITET

Steps in ND evaluations

- Retrieval/Selection of experimental data
- Weighting/Corrections/Adjustment of experimental data
- Fitting a physics model to experimental data
- Generation of ENDF file / random files / libraries
- Validation

UPPSALA UNIVERSITET

Complexities

- Formats / Data retrieval / Data storage (EXFOR, C4, CSV, ...)
- Algorithms (UMC-G/B, Kalman, BMC, ...)
- Variety of interacting codes (TALYS, EMPIRE, TEFAL, checking codes, ...)
- Variety of interacting programming languages (bash, Python, R, Perl, ...)
- Variety of interacting IT systems (Windows, Linux, hardware, clusters)

Choices / Opinions / Lock-in

UPPSALA UNIVERSITET

Questions

- How can we ascertain the correct implementation?
- How can we facilitate deploying/sharing a pipeline for users?
- How can we facilitate development?
- How can we make a system adaptable/extendable and future-proof?
- How can we accelerate the building process of a pipeline?

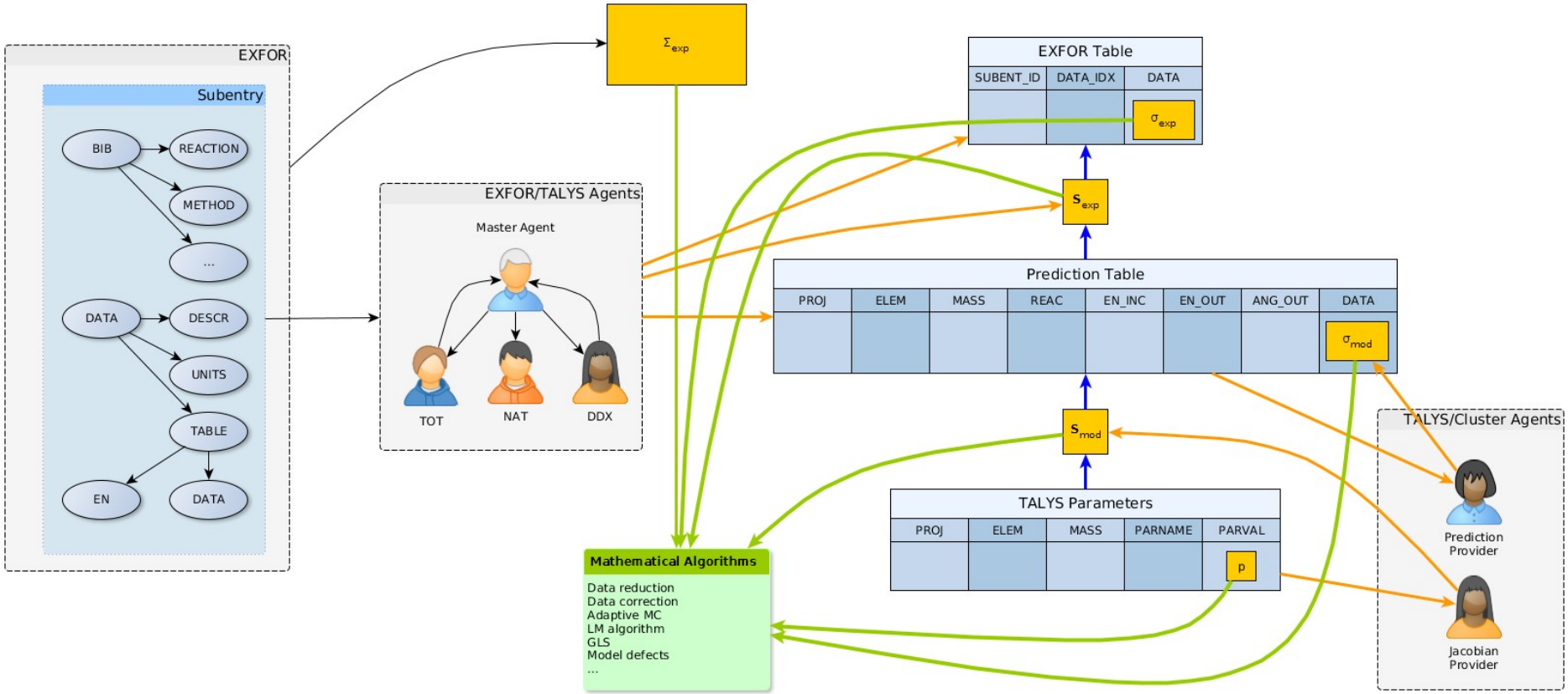
UPPSALA UNIVERSITET

Goals

- Quality assurance
- Streamlined deployment
- Good reusability
- Transparency

UPPSALA UNIVERSITET

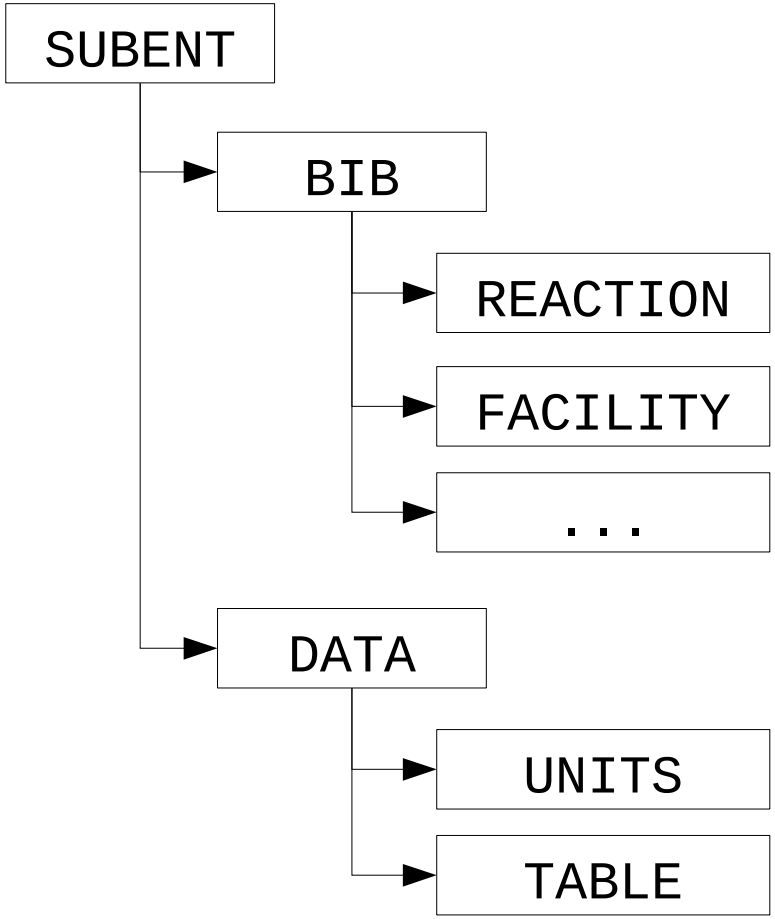
Overview of system developed at UU



UPPSALA UNIVERSITET

Interfaces / Example #1

```
SUBENT        23171003   20170913   20180129   20180126       2265
BIB                 11         26
REACTION   (26-FE-56(N,2N)26-FE-55,,SIG)
DECAY-DATA (26-FE-55,2.73YR)
FACILITY  1(VDG,2ZZZGEL) Van de Graaff accelerator at IRMM.
          2(ACCEL,2AUSIRK) For AMS (accelerator mass spectrometry)
            at VERA lab. for radionuclides 10Be, 14C, 26Al, 55Fe,
           (CCW,2GERDRE)
METHOD    1(ACTIV)  Activation technique combined with
          2(AMS)    mass spectrometric technique
SAMPLE      Natural iron samples were irradiated at TU Dresden
            and IRMM.
INC-SOURCE1(D-T) T(d,n)He-4 .
INC-SPECT 1 Quasi-monoenergetic neutrons with energies between
            13.4 and 14.8 MeV;
          2 from 13 to 20 MeV.
FLAG       (1.) Experiment in 2007 yr,   TUD/VERA
           (2.) Experiment in 2010 yr,   TUD/VERA
           (3.) Experiment in 2010 yr,   IRMM/VERA
ERR-ANALYS (EN-ERR) finite neutron energy distribution and the
                   absolute uncertainty in the neutron energy
                      (namely it is EN-ERR+EN-RSL).
STATUS     (PRELM) Preliminary results (decleared by A.Wallner,
                   2017-08-30) under STATUS).
           (TABLE) Data received from the author (2017-08-30)
HISTORY    (20170913R) SD: Data received from the author were
           added. FLAG was added. BIB update according to
           comments from author.
ENDBIB              26
NOCOMMON             0          0
DATA                 5         15
EN          EN-ERR     DATA       DATA-ERR   FLAG
MEV         MEV         MB          MB         NO-DIM
      13.35       0.15       300.       100.         3.
      13.49       0.04      322.4       16.1         1.
      ...
ENDDATA             17
ENDSUBENT           48
```



Hierarchical format

UPPSALA UNIVERSITET

Interfaces / Example #1

```
library(MongoEXFOR)
db <- connectExfor("entries","exfor","mongodb://localhost")

queryStr <- makeQueryStr(and(
  'BIB.REACTION: { $regex: "26-FE-56.*SIG", $options: "" }',
  'BIB.REACTION: { $not: { $regex: "\\) *[+-*/] *\\(", $options: "" }}',
  'DATA.TABLE.DATA: { $exists: true }',
  'DATA.TABLE.EN: { $exists: true }'
))

resDt <- db$find(queryStr, {
  list(SUBENT = ID,
       REAC = BIB$REACTION)
})
```

```
   SUBENT                                          REAC
1: 10022010       (26-FE-56(N,P)25-MN-56,,SIG)
2: 10031005       (26-FE-56(N,P)25-MN-56,,SIG)
3: 10037004     (26-FE-56(N,EL)26-FE-56,,SIG)
4: 10037005             (26-FE-56(N,TOT),,SIG)
5: 10037015 (26-FE-56(N,INL)26-FE-56,PAR,SIG)
...
```

UPPSALA UNIVERSITET

Interfaces / Example #2

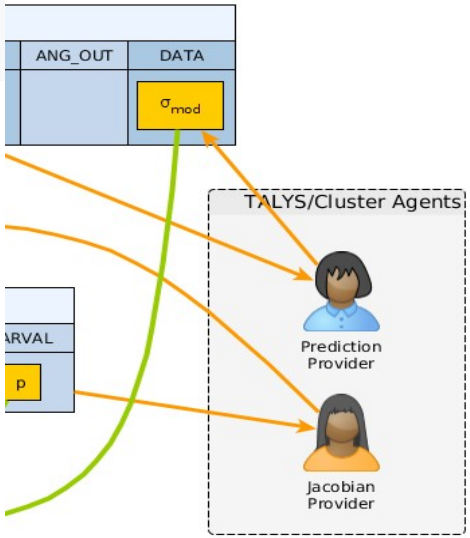
From https://github.com/gschnabel/clusterSSH:

Now everything is set up to apply a function to an input list in parallel. Let's create a simple function that takes the numbers in the input list and adds one to them:

```
parFun <- function(input) {
    lapply(input, function(x) x+1)
}
```

This function can be applied to some input list:

```
input <- list(1,10,20)
clusterHnd$eval(parFun, input, pollTime=5)
```



UPPSALA UNIVERSITET

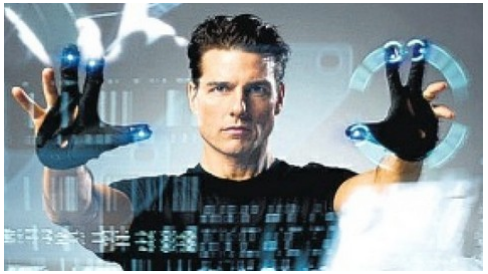
Conclusions about interfaces

While building systems, interfaces are more important than implementation

(Design by contract / Contract programming)

Advantages:

- Test-driven development / mockup (quality assurance)
- Coexistence of programming languages(?) (reusability, deployment)
- Broader expert participation(?)
- Higher level of abstraction



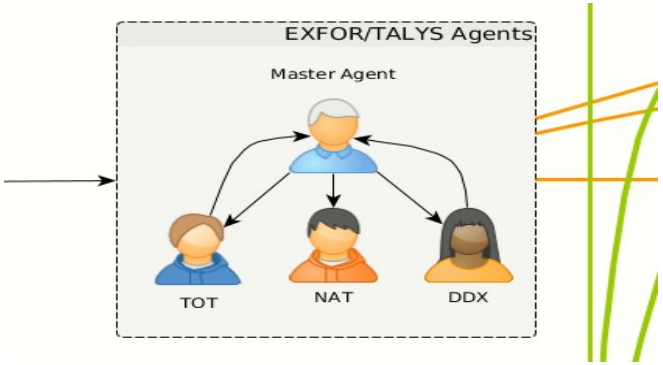
Minority report (2002)

UPPSALA UNIVERSITET

Where interfaces, there modules

Modules should be...

- easily available... (reusability)
- ...as open-source (transparency)
- under version control (quality assurance)



UPPSALA UNIVERSITET

GitHub / GitLab / Bitbucket / ...

launch TALYS calculations in parallel on a cluster and retrieve results from R

2 commits | 1 branch | 0 packages | 0 releases | 1 contributor | MIT

Branch: master | New pull request | Find file | Clone or download

gschnabel added README — Latest commit be8e8ba on Mar 27

R	first commit	8 months ago
man	first commit	8 months ago
.Rbuildignore	first commit	8 months ago
.gitignore	first commit	8 months ago
DESCRIPTION	first commit	8 months ago
LICENSE.MIT	first commit	8 months ago
NAMESPACE	first commit	8 months ago
README.md	added README	8 months ago

README.md

clusterTALYS - R package

The package `cluserTALYS` enables launching calculations in parallel over an SSH connection on computers with multiple processors, clusters of work stations, or scientific computing clusters.

Requirements

This package makes use of the functionality of the R package clusterSSH to communicate with the cluster, which must therefore be installed together with its dependencies. Some of the required packages currently use the commands `rsync`,

UPPSALA UNIVERSITET

Dependencies...

launch TALYS calculations in parallel on a cluster and retrieve results from R

2 commits | 1 branch | 0 packages | 0 releases | 1 contributor | MIT

Branch: master | New pull request | Find file | Clone or download

gschnabel added README — Latest commit be8e8ba on Mar 27

R	first commit	8 months ago
man	first commit	8 months ago
.Rbuildignore	first commit	8 months ago
.gitignore	first commit	8 months ago
DESCRIPTION	first commit	8 months ago
LICENSE.MIT	first commit	8 months ago
NAMESPACE	first commit	8 months ago
README.md	added README	8 months ago

README.md

clusterTALYS - R package

The package `cluserTALYS` enables launching calculations in parallel over an SSH connection on computers with multiple processors, clusters of work stations, or scientific computing clusters.

Requirements

This package makes use of the functionality of the R package clusterSSH to communicate with the cluster, which must therefore be installed together with its dependencies. Some of the required packages currently use the commands `rsync`,

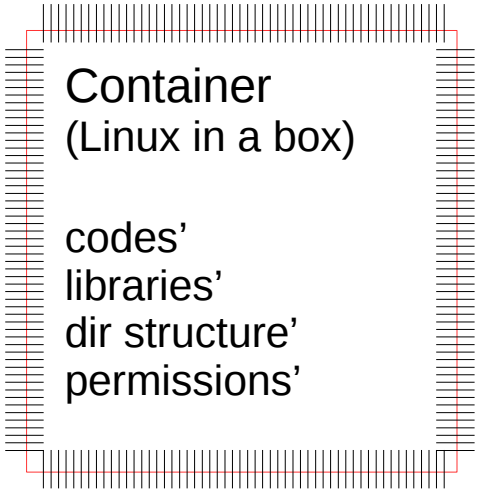
UPPSALA UNIVERSITET

Docker

Application/Mechanism to manage containers

Linux

codes
libraries
directory structure
permissions



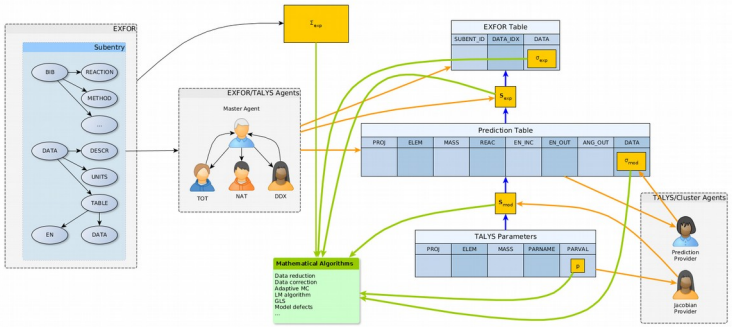
UPPSALA UNIVERSITET

Docker concepts

Dockerfile	Script with instructions to assemble an image	Technical drawing of a car
Image	prototype of a container (immutable)	car prototype (immutable)
Container	containers are running mini-computers (initially clones of the prototype but can be altered, e.g., installation of new software)	real cars used on the street (fresh out of the factory clones of the prototype but can be modified, e.g., installation of new radio)

UPPSALA UNIVERSITET

Making Docker images using Dockerfiles



R interpreter
Sequence of R scripts
R packages
Linux tools (rsync, ssh)
MongoDB / EXFOR
TALYS
...

Dockerfile

Instructions to assemble image

comprises, e.g.:

download Linux image

download modules

configure system

etc.

web resources

GitHub: ClusterSSH

MongoDB website

TALYS website

EXFOR

https://github.com/gschnabel/eval-fe56-docker

docker build -t eval-fe56-img:latest .

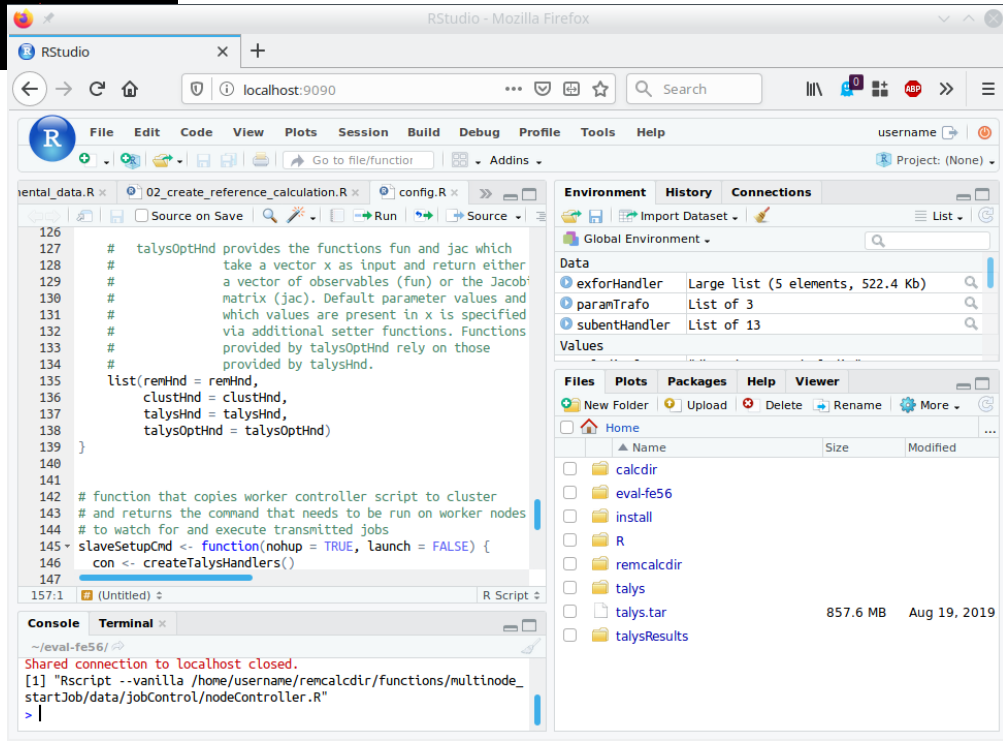
UPPSALA UNIVERSITET

Containers in practice

```
(base) georg@georg-UX32LN:~$
(base) georg@georg-UX32LN:~$ docker ps -a
CONTAINER ID        IMAGE                    COMMAND
 NAMES
d54a2e433806        eval-fe56-img:latest     "/bin/sh -c /home/st…"
 eval-fe56-cont
338e2a88f249        compexfor:latest         "/bin/sh -c /home/st…"
 compexfor-cont
(base) georg@georg-UX32LN:~$
(base) georg@georg-UX32LN:~$ docker start eval-fe56-cont
eval-fe56-cont
(base) georg@georg-UX32LN:~$ docker attach eval-fe56-cont
root@d54a2e433806:/#
root@d54a2e433806:/#
```

Shell inside the container

Access GUI via web browser



UPPSALA UNIVERSITET

Communication with containers

Linux

codes
libraries
directory structure
permissions

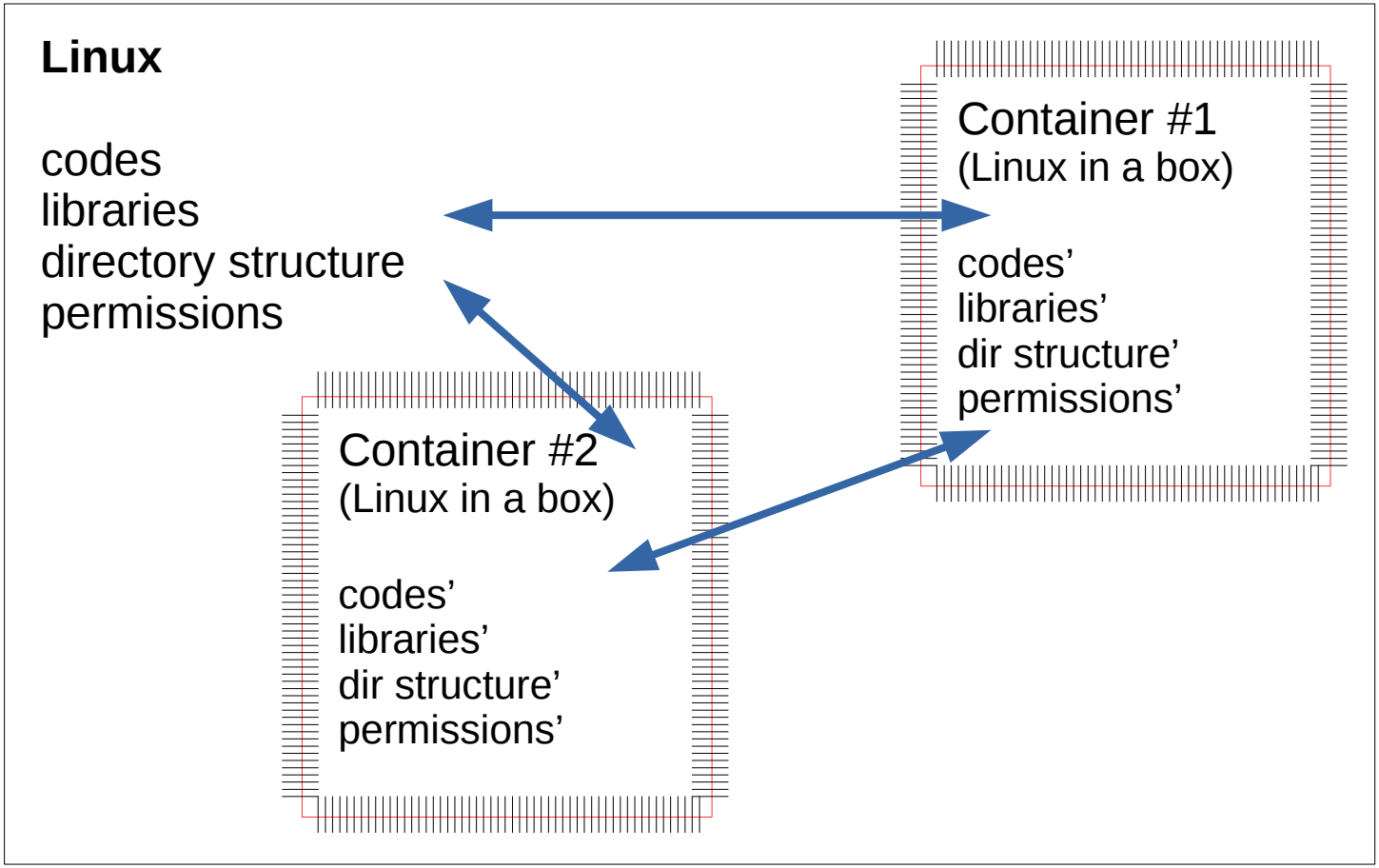
Comunication via:
shared folders
network protocols
(e.g. http, ssh, ftp)

Container
(Linux in a box)

codes'
libraries'
dir structure'
permissions'

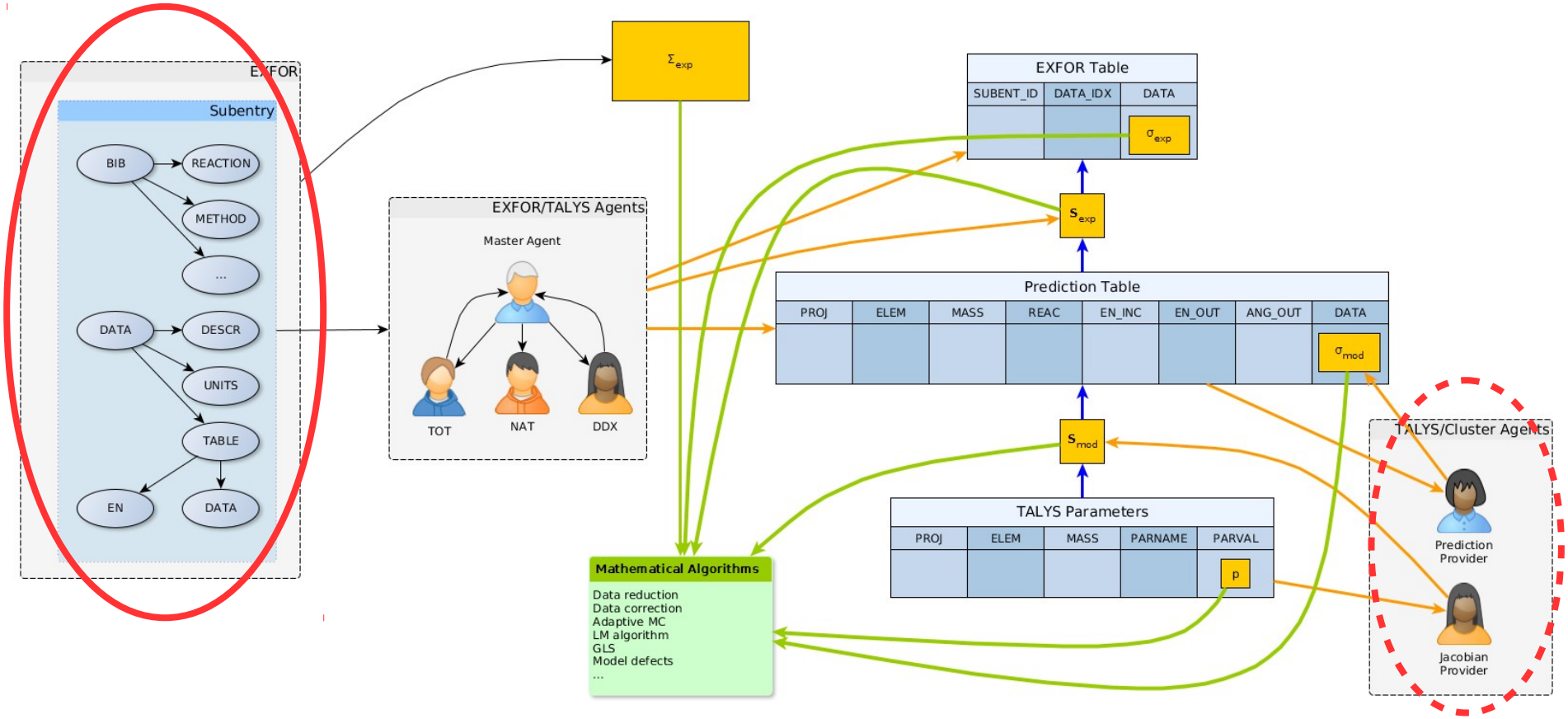
UPPSALA UNIVERSITET

Communication with containers



UPPSALA UNIVERSITET

Multiple containers



https://github.com/gschnabel/compEXFOR-docker

https://github.com/gschnabel/eval-fe56-docker

UPPSALA UNIVERSITET

Conclusions

- **Goals**: *quality assurance*, *streamlined deployment*, *good reuseability*, *transparency*
- Well-defined interfaces are important for quality assurance and reuseability
- Version control (e.g., git) is important for transparency and quality assurance
- Docker helps fast and streamlined deployment
- All these technical tools/utils have been beneficially applied for the creation of a nuclear data evaluation pipeline, which has already been successfully deployed at several research institutions on computers with Windows, iOs and Linux
- Pipeline has room for improvement, e.g.,
  - documentation of interfaces is not enough---in systems with several interacting modules/agents, it is equally important to document the data flows
  - creation of Docker containers to run on the cluster to facilitate the setup for distributed computing; let the individual tasks run in containers
  - semantic versioning of interface specifications and modules