From disconnected elements to a harmonious ecosystem

The Epiverse-TRACE project

Hugo Gruson

Senior Research Software Engineer, data.org

Piping system on a chemical tanker, by Hervé Cozanet, CC BY-SA.

Modern pipelines are not made of fully interoperable pieces.

Lots of data wrangling is required to:

- Make successive pieces fit well with one another
- Swap one piece for another equivalent one

Piping system on a chemical tanker, by Hervé Cozanet, CC BY-SA.

Modern pipelines are not made of fully interoperable pieces.

Lots of data wrangling is required to:

- Make successive pieces fit well with one another
- Swap one piece for another equivalent one

Our community spends a lot of time working on individual pieces, but little time on the ecosystem as a whole.

Piping system on a chemical tanker, by Hervé Cozanet, CC BY-SA.

Modern pipelines are not made of fully interoperable pieces.

Lots of data wrangling is required to:

- Make successive pieces fit well with one another
- Swap one piece for another equivalent one

Our community spends a lot of time working on individual pieces, but little time on the ecosystem as a whole.

Epiverse-TRACE project 2021-2024(-2026)

Epiverse-TRACE project 2021-2024(-2026)

Main goals:

- Harmonise the ecosystem of epidemiology tooling
- Make existing pieces interoperable
- Develop of sustainable community of users

Epiverse-TRACE project 2021-2024(-2026)

Main goals:

- Harmonise the ecosystem of epidemiology tooling
- Make existing pieces interoperable
- Develop of sustainable community of users

Secondary goals:

- Spread good practices in software development
- Advocate for the development of RSE career paths

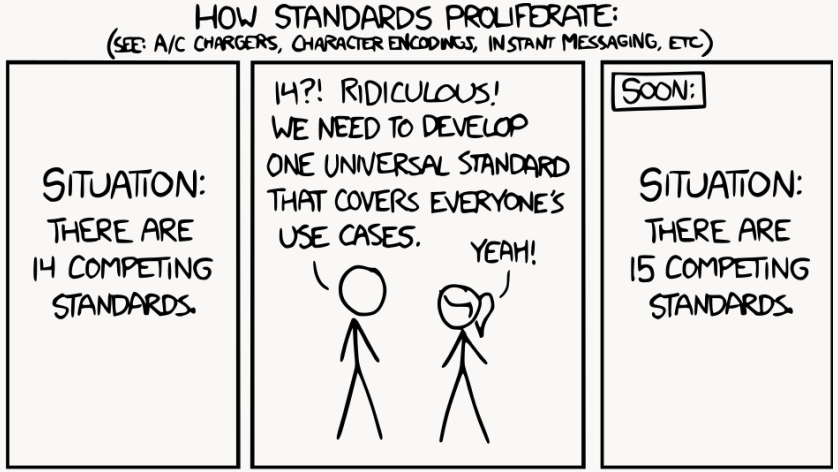
What we don’t want:

What we don’t want:

- Reinvent the wheel
- Exclude existing community members & developers
- XKCD 927

What we don’t want:

- Reinvent the wheel
- Exclude existing community members & developers
- XKCD 927



XKCD 927: *Standards*, by Randall Munroe, CC BY-NC

Diversity is good but we also need interoperability!

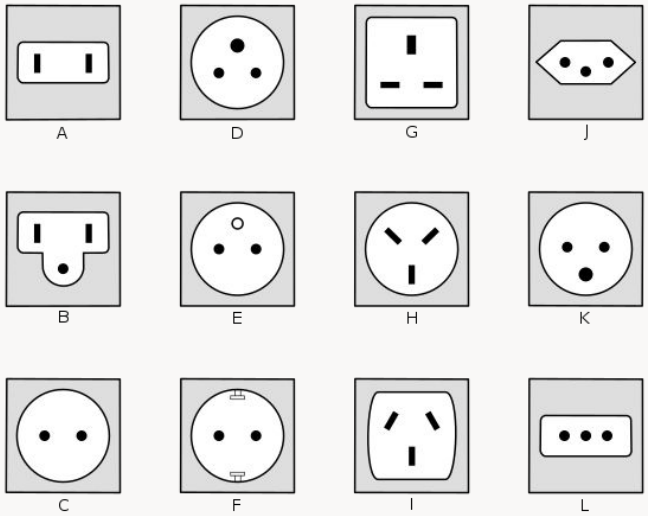
Diversity is good but we also need interoperability!

Our goal:

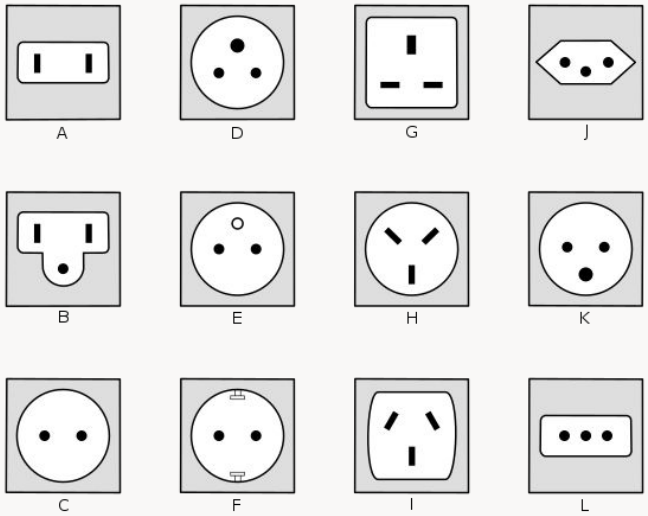
- Keep the diversity
- Buy-in from the community
- Collaborate to existing projects
- Be good open-source citizens

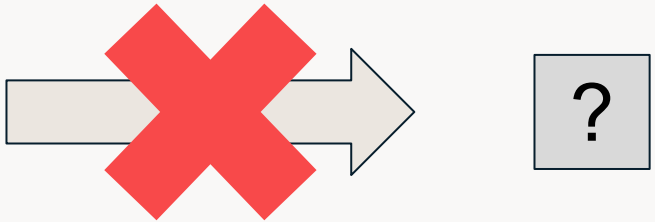
Create compatibility layers, not “one standard to rule them all”

Create compatibility layers, not “one standard to rule them all”

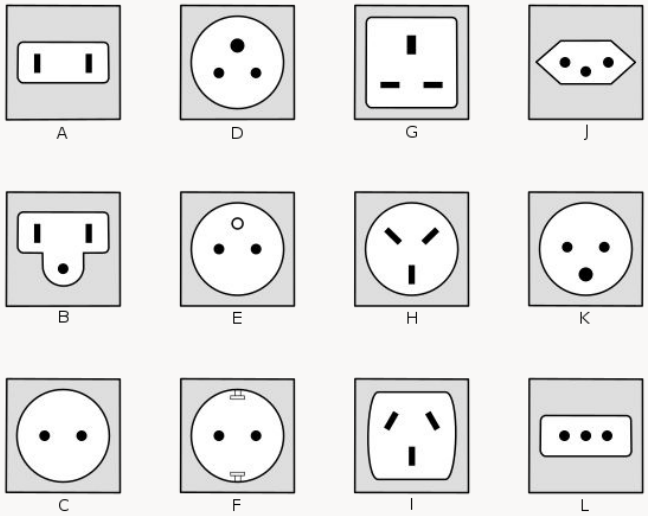


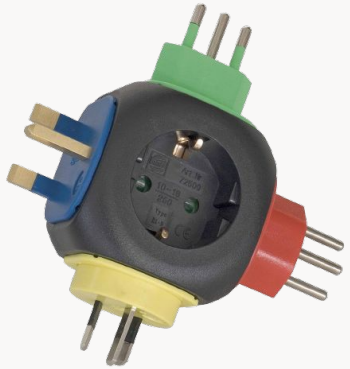
Create compatibility layers, not “one standard to rule them all”





Create compatibility layers, not “one standard to rule them all”

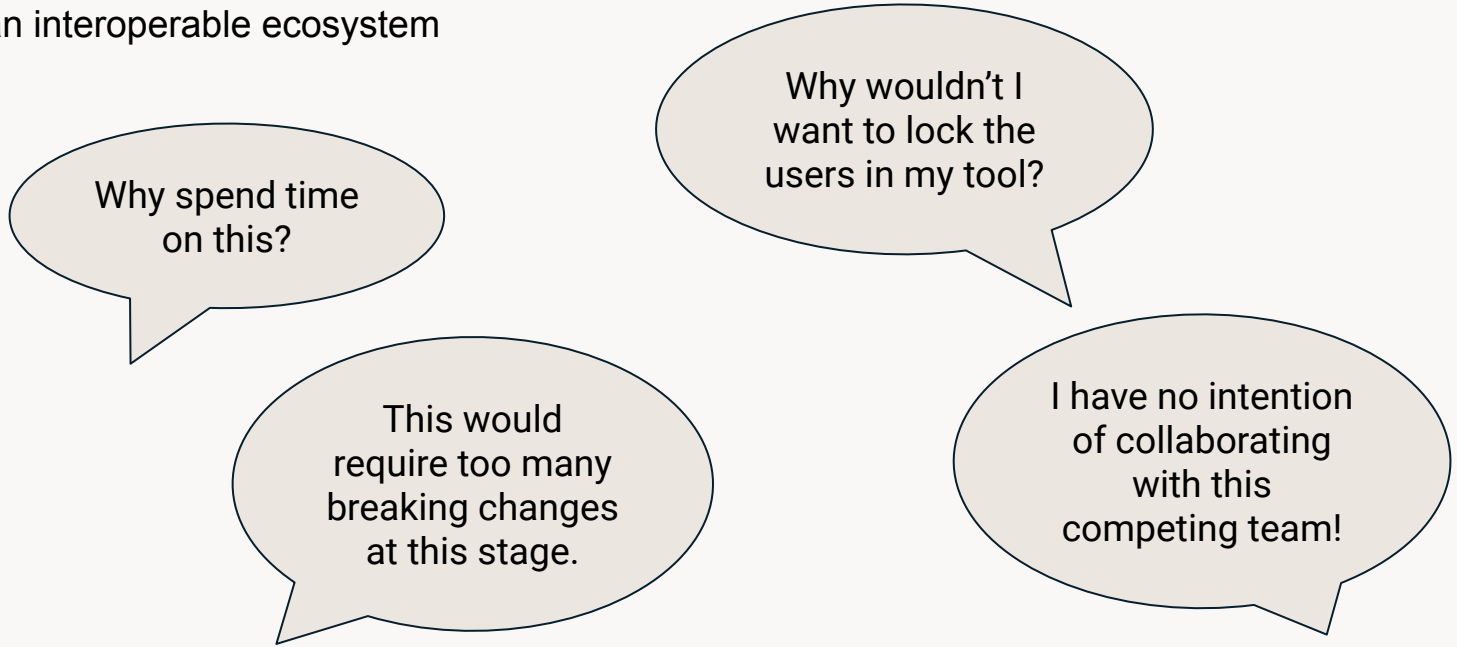




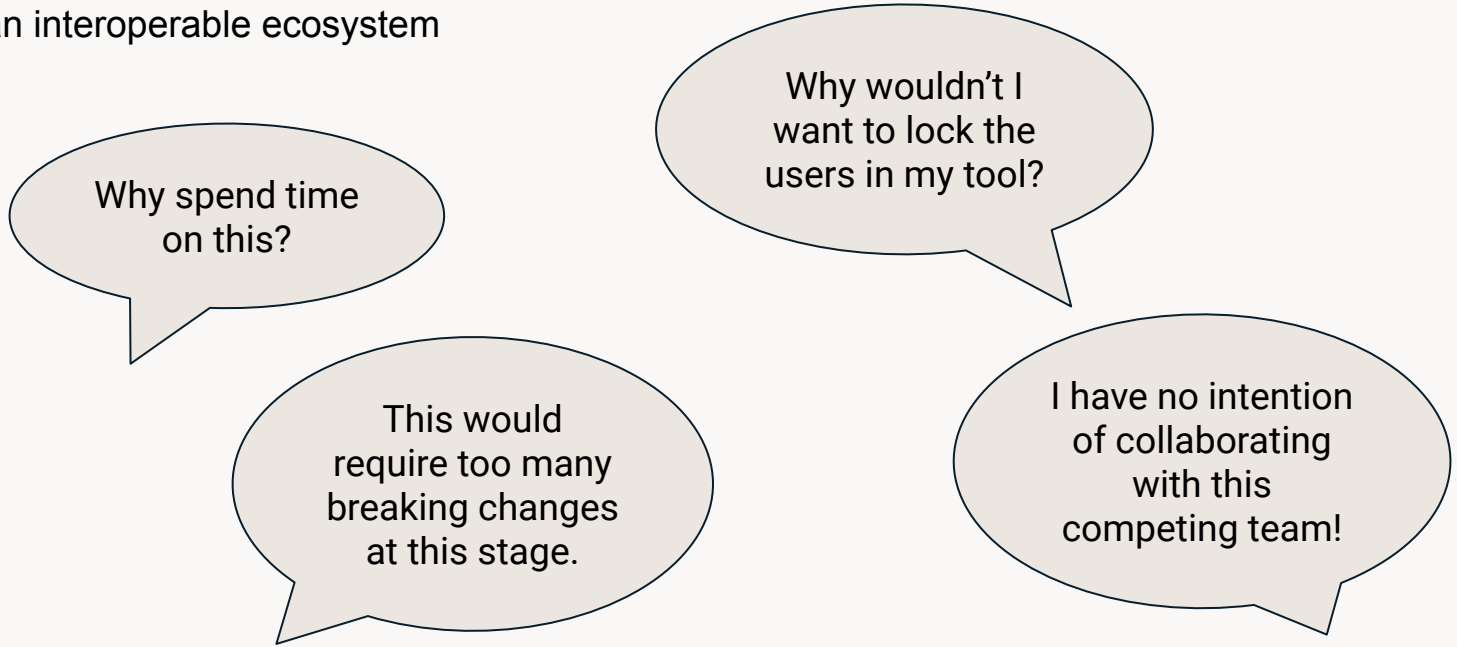
Reisestecker mit austauschbaren Adaptern für verschiedene Länder, by Cephira, CC BY-SA

Building an interoperable ecosystem

Building an interoperable ecosystem



Building an interoperable ecosystem

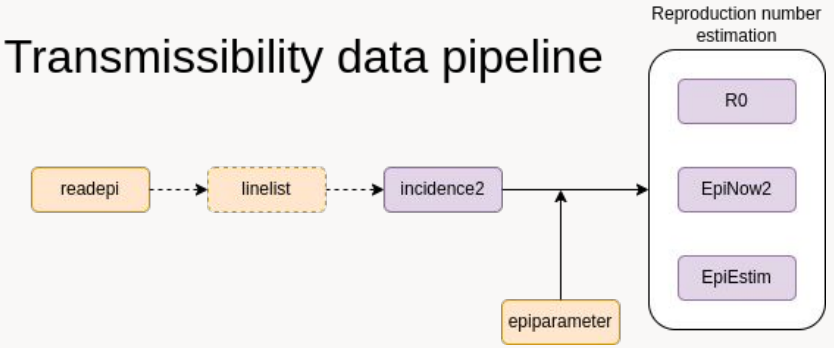


Interoperability is a socio-technical problem.

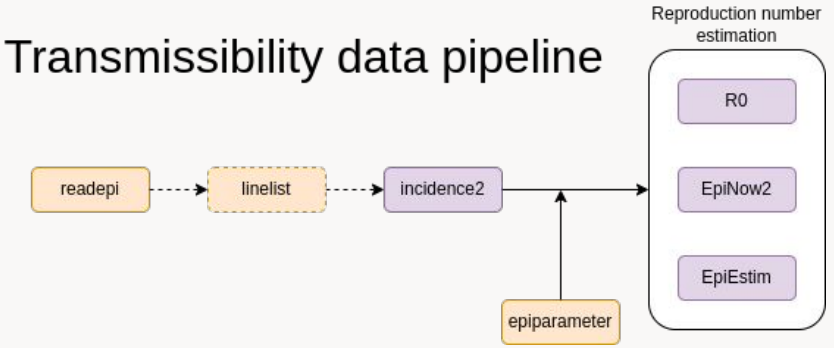
It's difficult to do right and not rewarded at all in career development.

Developing community buy-in

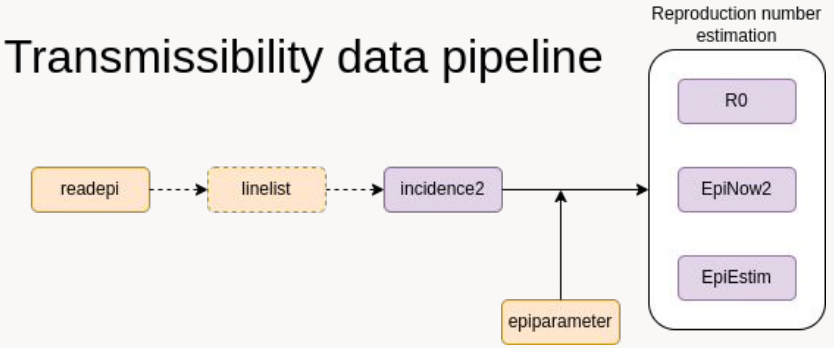
1. Write the pipelines

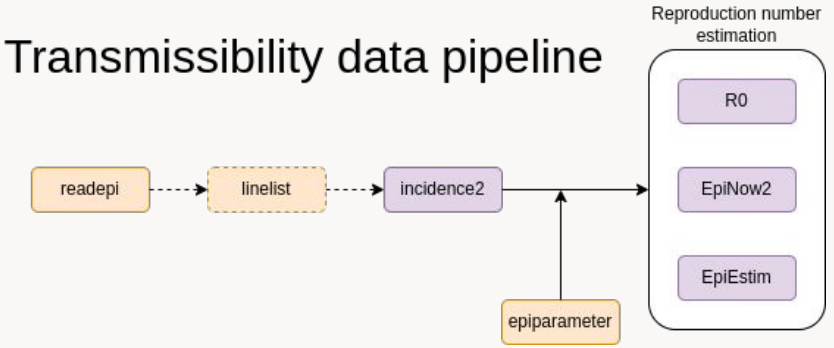


1. Write the pipelines
    - identify the alternatives & pain points
    - provide an interim solution

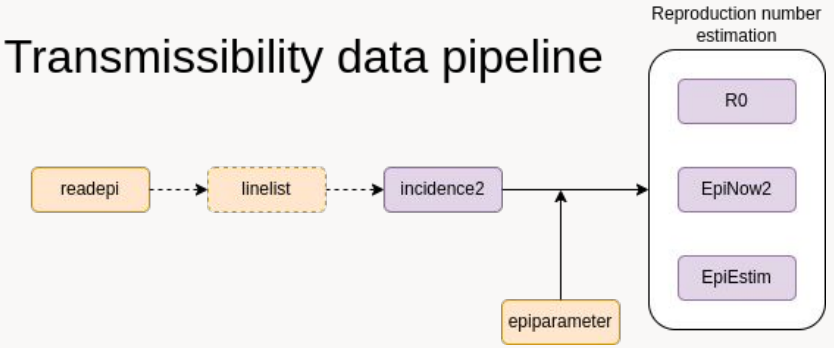


1. Write the pipelines
    - identify the alternatives & pain points
    - provide an interim solution
2. Organise meeting with devs of alternatives





1. Write the pipelines
    - identify the alternatives & pain points
    - provide an interim solution
2. Organise meeting with devs of alternatives
    - neutral party, no bad blood
    - get out of the “let’s regroup under my banner” mentality
    - (build trust by contributing to outstanding issues)



1. Write the pipelines
    - identify the alternatives & pain points
    - provide an interim solution
2. Organise meeting with devs of alternatives
    - neutral party, no bad blood
    - get out of the “let’s regroup under my banner” mentality
    - (build trust by contributing to outstanding issues)
3. Create a shared piece of underlying infrastructure to allow compatibility and simplify maintenance of shared tasks

Adding interoperability while minimizing breaking changes

Using R S3 object orientation to transparently operate the change.

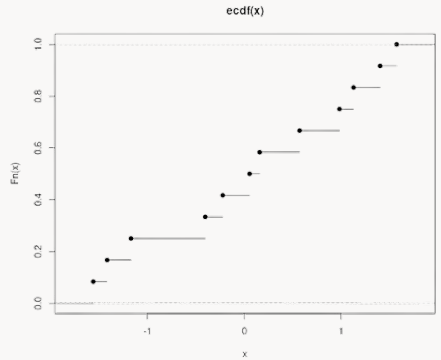
Using R S3 object orientation to transparently operate the change.

S3 is strange (like R):

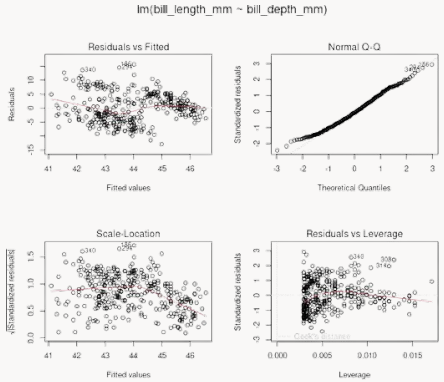
- "*S3 is informal and ad hoc, but there is a certain elegance in its minimalism: you can't take away any part of it and still have a useful OO system*." Hadley Wickham
- Mostly an advanced dispatch system based on the presence of the specific attribute
- Particularly useful because users don't need to understand anything about OO

Using R S3 object orientation to transparently operate the change.

```
fn <- ecdf(rnorm(12))
plot(fn)
```




```
m <- lm(bill_length_mm ~ bill_depth_mm, data = penguins)
plot(m)
```



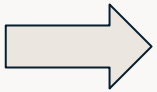
Fix both ends of each pipe:

- Adding dispatch based on input
- Adding a class to output

Fix both ends of each pipe:

- Adding dispatch based on input
- Adding a class to output

```r
#' @export
centroid <- function(coords, weights) {

  # ...

}
```

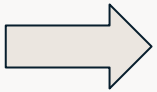


```r
#' @export
centroid <- function(coords, weights) {

  UseMethod("centroid")

}

#' @rdname centroid
#'
#' @export
centroid.default <- function(coords, weights) {

  # ...

}
```

Fix both ends of each pipe:

- Adding dispatch based on input
- Adding a class to output

```r
#' @export
centroid <- function(coords, weights) {

  # ...

}
```



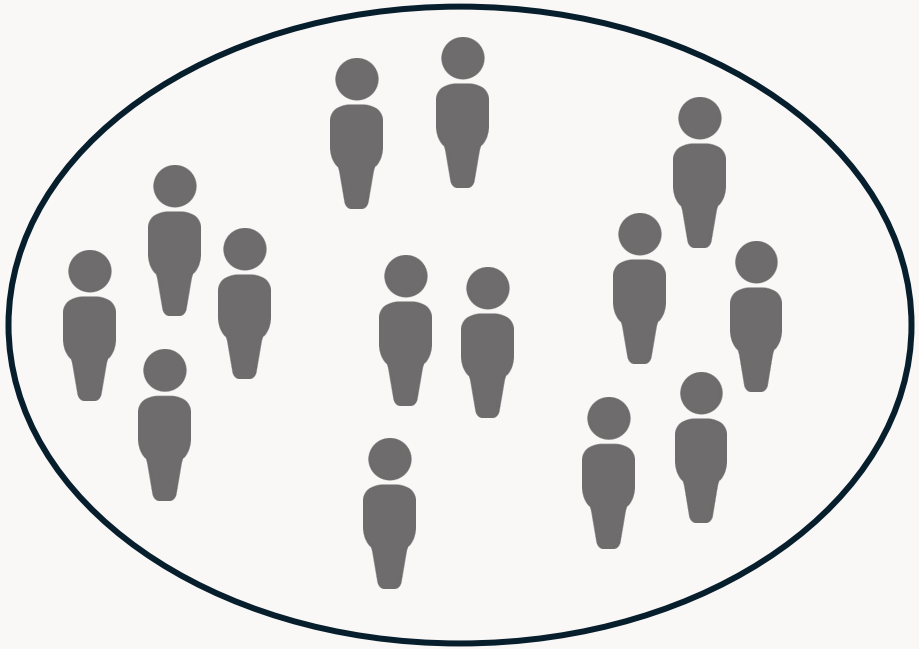
```r
#' @export
centroid <- function(coords, weights) {

  UseMethod("centroid")

}

#' @rdname centroid
#'
#' @export
centroid.default <- function(coords, weights) {

  # ...

}
```

Technical details & pitfalls at https://epiverse-trace.github.io/posts/s3-generic/

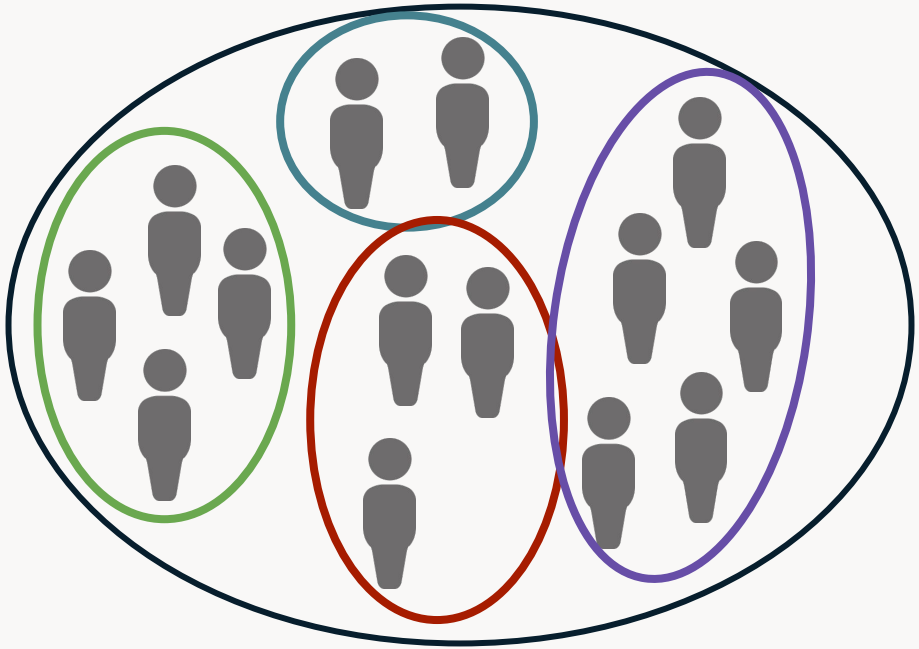
Examples & reception

Social contact data

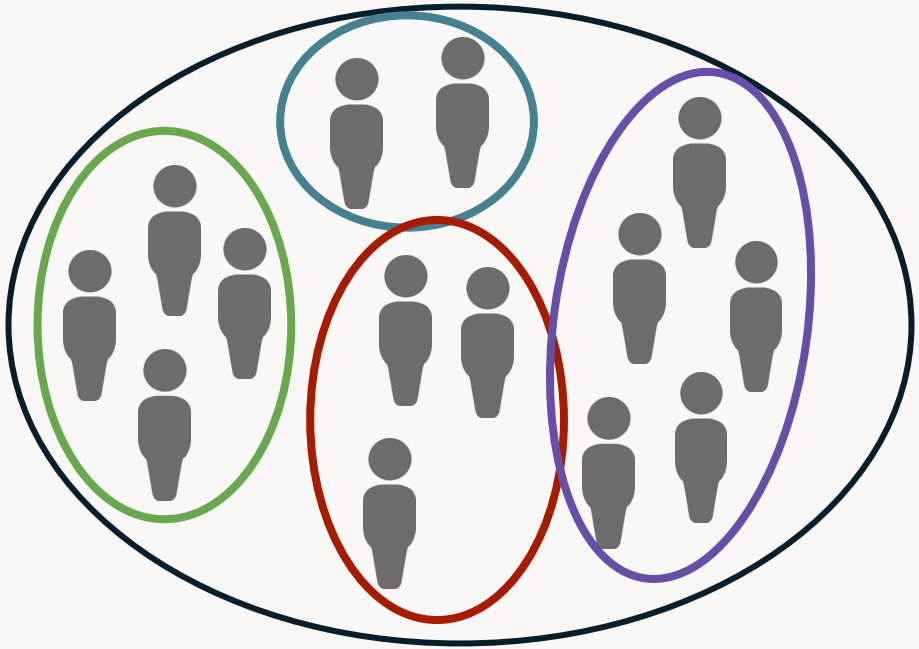
Social contact data



Social contact data

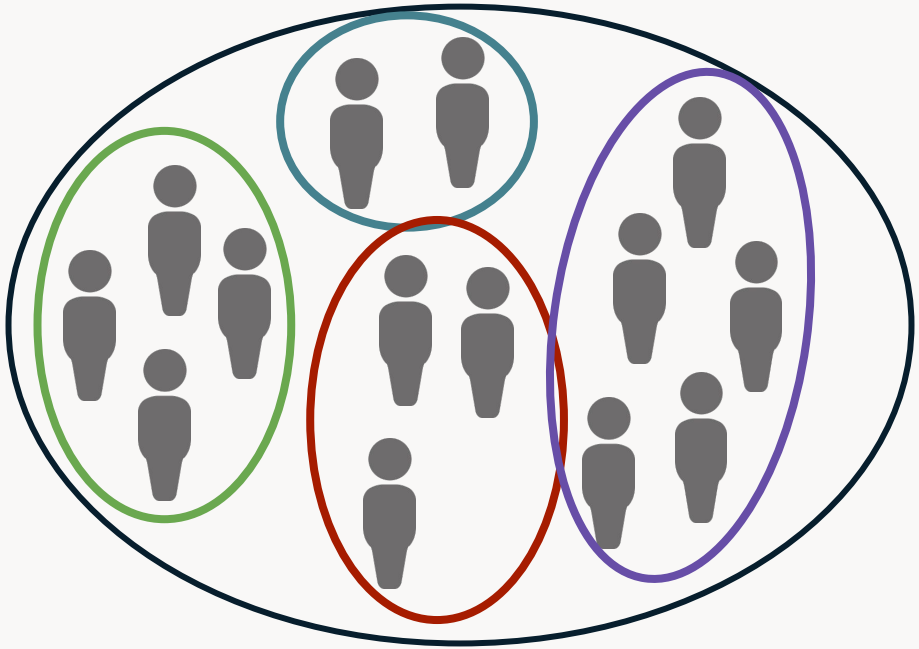


Social contact data



Quantify contact between groups

Social contact data



Quantify contact between groups

Used in mathematical modelling

Social contact data R packages

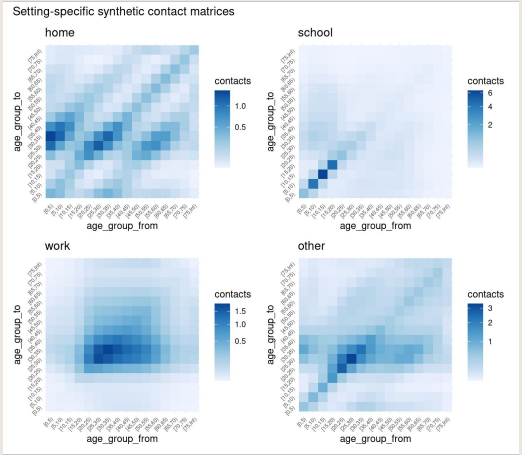
`conmat`

`socialmixr`

`contactdata`

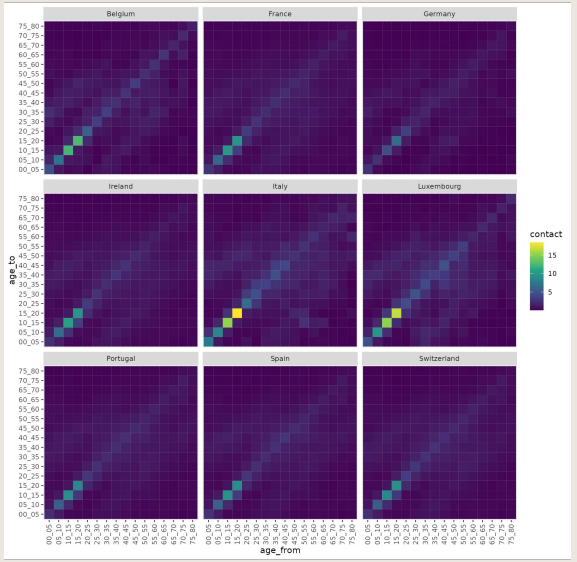
Social contact data R packages

conmat



socialmixr

contactdata



data.org

Social contact data R packages

`conmat` | `socialmixr` | `contactdata`

Shared `cm` class:

- plotting method
- printing method
- utilities (e.g., change age class breaks)

Social contact data R packages

`conmat` | `socialmixr` | `contactdata`

Shared `cm` class:

- plotting method
- printing method
- utilities (e.g., change age class breaks)

Downstream tools interact with the same class of object, no matter the exact method they use

Conclusion

Problem statement

- Interoperability is hard, and seldomly encouraged
- Non-interoperable ecosystems generate frustration for users and duplicated developer efforts
- Some technical challenges but mostly a human problem

Looking to the future

- Think of software as tiny piece of a larger problem
  - Community
  - Interoperability
  - Sustainability
- Let’s collaborate more!
  - Split maintenance load over multiple maintainers
  - More coherent ecosystem for users
  - Fun!

Epiverse-TRACE Development Team

LSHTM / MRCG / UKHSA

- James Azam
- Bubacarr Bah
- Anna Carnegie
- Roz Eggo
- Sebastian Funk
- Pratik Gupte
- Adam Kucharski
- Joshua Lambert
- Edwin Van Leeuwen
- Karim Mané
- Tim Russell
- Carmen Tamayo
- Tim Taylor
- Andree Valle

UK Health Security Agency

PUJ / UniAndes

- Zulma Cucunubá
- Geraldine Gomez
- Julian Otero
- Jaime Pavlich-Mariscal
- David Santiago Quevedo
- Mauricio Santos-Vega
- Nicolás Torres
- Juan Daniel Umaña

Questions / Feedback?

Epiverse-TRACE

`epiverse-trace`

https://data.org/initiatives/epiverse

Hugo Gruson

`bisaloo`

https://hugogruson.fr