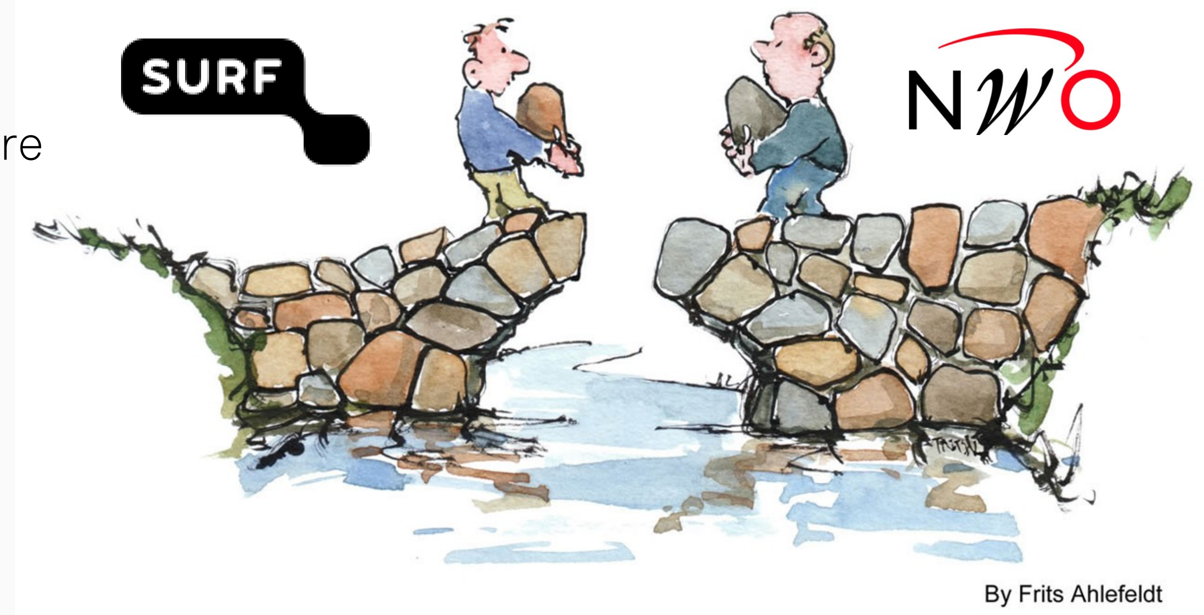
An introduction to XENON

Jason Maassen, Arnold Kuzniar, Jurriaan Spaaks, Stefan Verhoeven, Johan Hidding, ...

netherlands eScience center

netherlands eScience center ?

research
computer
infrastructure



By Frits Ahlefeldt

research
funding

We are an institute that provides **funding + RSEs** to research projects.

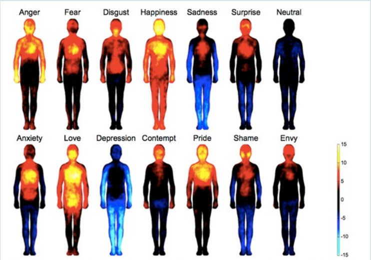
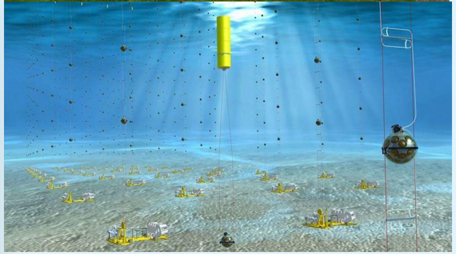


Fig. 2. Bodily topography of basic (*Upper*) and nonbasic (*Lower*) emotions associated with words. The body maps show regions whose activation increased (warm colors) or decreased (cool colors) when feeling each emotion. ($P < 0.05$ FDR corrected; $t > 1.94$). The colorbar indicates the t-statistic range.

so far: ~130 projects

(on many different topics)

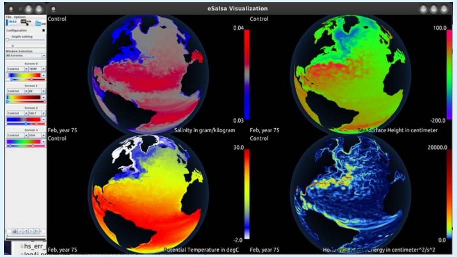


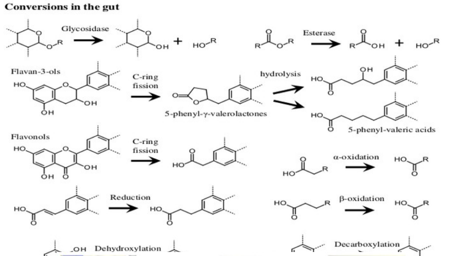
Humanities & Social Sciences

incl. SMART cities, text analysis, creative technologies

Physics & Beyond

incl. astronomy, high-energy physics, advanced materials


eSalsa Visualization
Control
Feb, year 75
Salinity in gram/kilogram
Surface Height in centimeter
Potential Temperature in degC

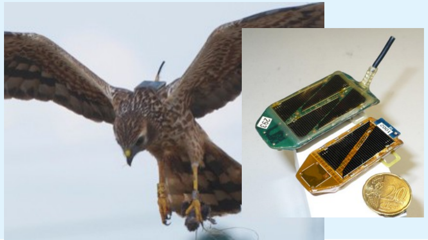


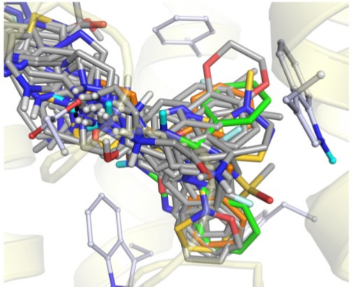
Sustainability & Environment

incl. climate, ecology, energy, logistics, water management

Life Sciences & eHealth

incl. bio-imaging, next generation sequencing, molecules





One of our goals: software re-use

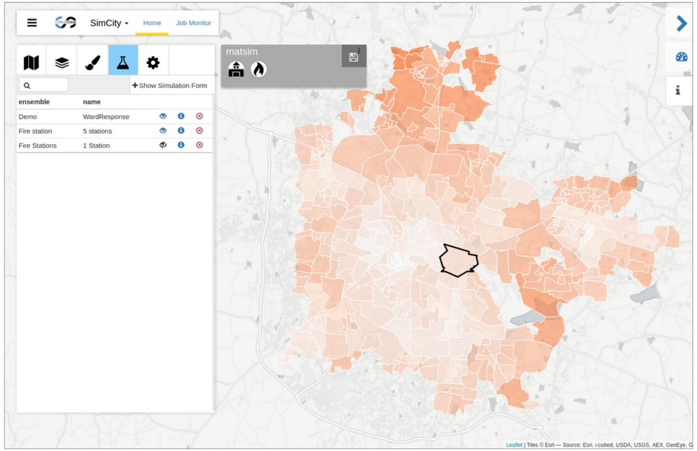
We try to re-use research software between projects

spreads cost of maintenance and development
re-use expertise of the engineers
increases user community
increases sustainability → more bang for buck!

Works best for software low on the software stack

generic tools are easier to reuse

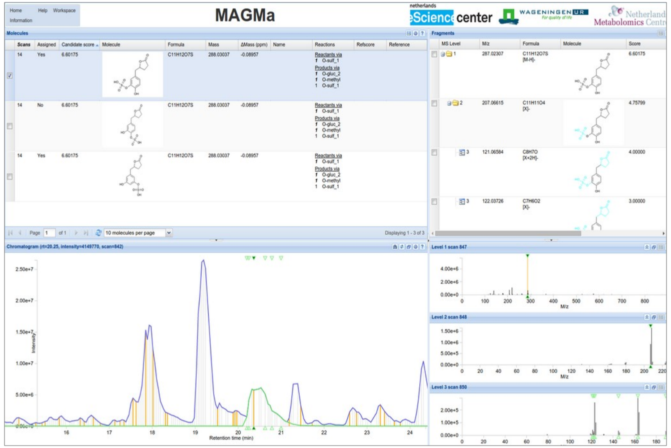
Examples applications



SimCity
Home
Job Monitor
Show Simulation Form
matsim
Leaflet | Tiles © Esri — Source: Esri, i-cubed, USDA, USGS, AEX, GeoEye, G

ensemble	name
Demo	WardResponse
Fire station	5 stations
Fire Stations	1 Station

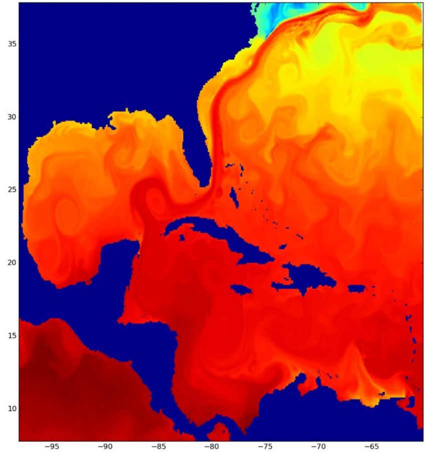
SIM-CITY



	Scans	Assigned	Candidate score	Molecule	Formula	Mass	ΔMass (ppm)	Name	Reactions	Refscore	Reference
☑	14	Yes	6.60175		C11H12O7S	288.03037	-0.08957		Reactants via 1 O-sulf_1 Products via 1 O-gluc_2 1 O-methyl 1 O-sulf_1		
□	14	No	6.60175		C11H12O7S	288.03037	-0.08957		Reactants via 1 O-sulf_1 Products via 1 O-gluc_2 1 O-methyl 1 O-sulf_1		
□	14	Yes	6.60175		C11H12O7S	288.03037	-0.08957		Reactants via 1 O-sulf_1 Products via 1 O-gluc_2 1 O-methyl 1 O-sulf_1		

	MS Level	M/z	Formula	Molecule	Score
□	1	287.02307	C11H12O7S [M-H]-		6.60175
□	2	207.06615	C11H11O4 [X]-		4.75799
□	3	121.06584	C8H7O [X+2H]-		4.00000
□	3	122.03726	C7H6O2 [X]-		3.00000

MAGMa



AMUSE / OMUSE

A recurring theme in many of our projects is **easy access to (remote) compute and storage.**

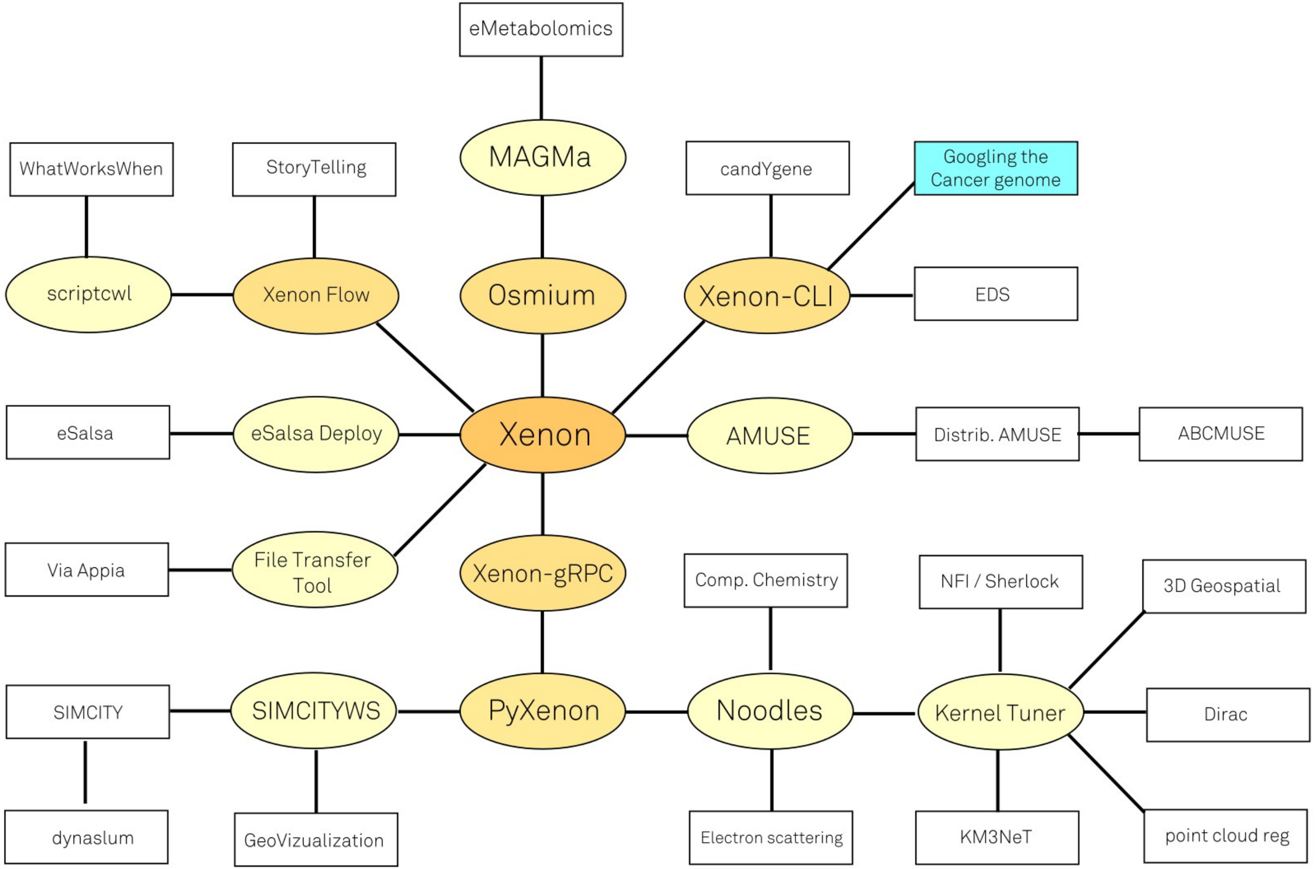
What is the problem ?

For many applications we need to copy data and submit jobs to remote systems. There are many ways to do this:

FTP, SFTP, WebDAV, S3, Hadoop, GridFTP, iRods, ...
Slurm, GridEngine, PBS, Torque, Amazon-Batch, ...

There are libraries and CLIs for each of these, however:

- you have to figure out how they work
- picking one will lock-in your solution
- using all of them is way too much work!



xenon

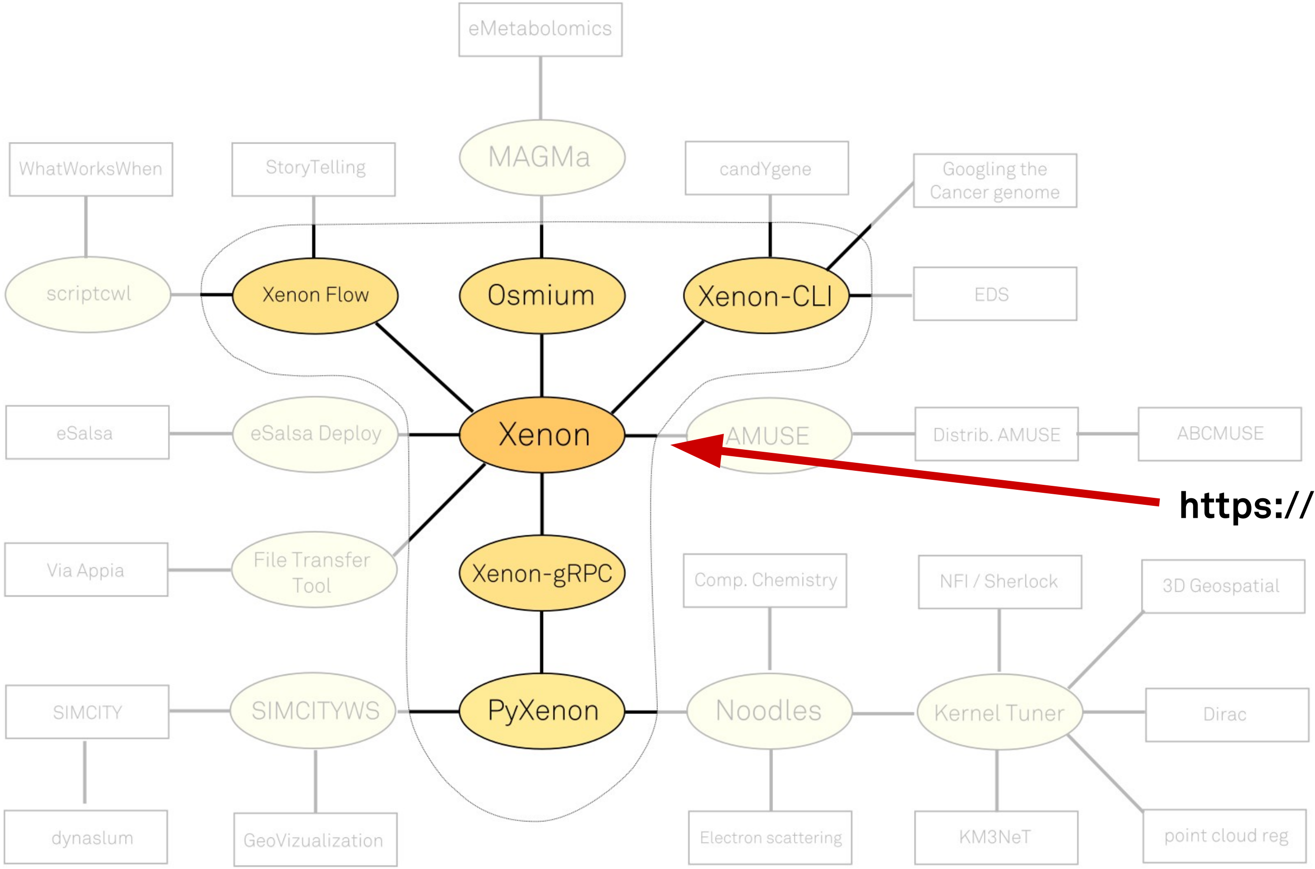
(and friends)

A collection of libraries and tools that provide easy access to (remote) compute and storage resources

Used via-via in many other tools and projects.

xenon

(and friends)



Core packages at:

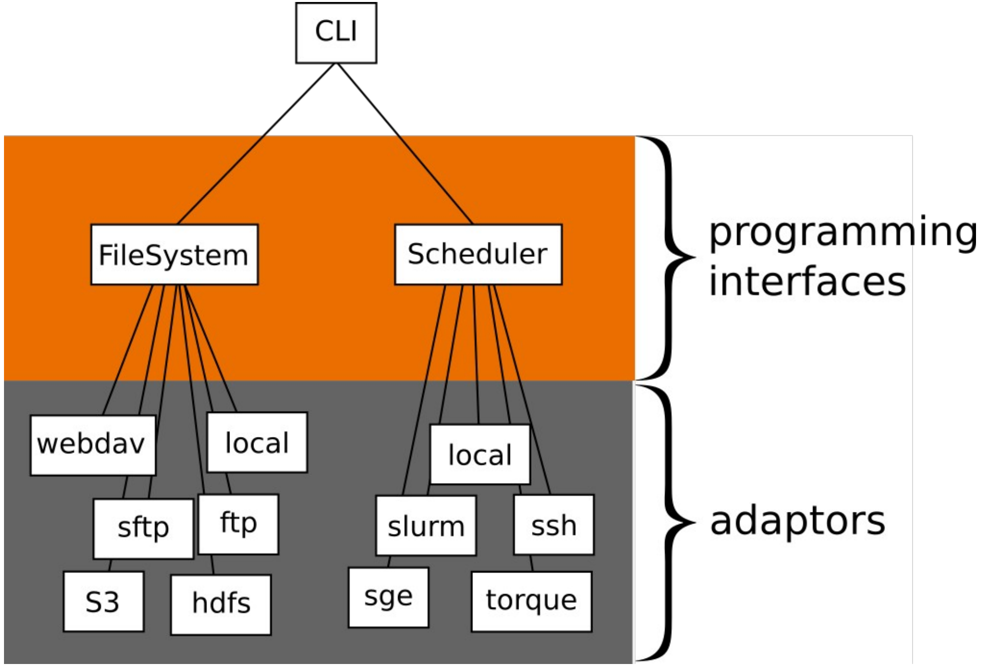
https://github.com/xenon-middleware

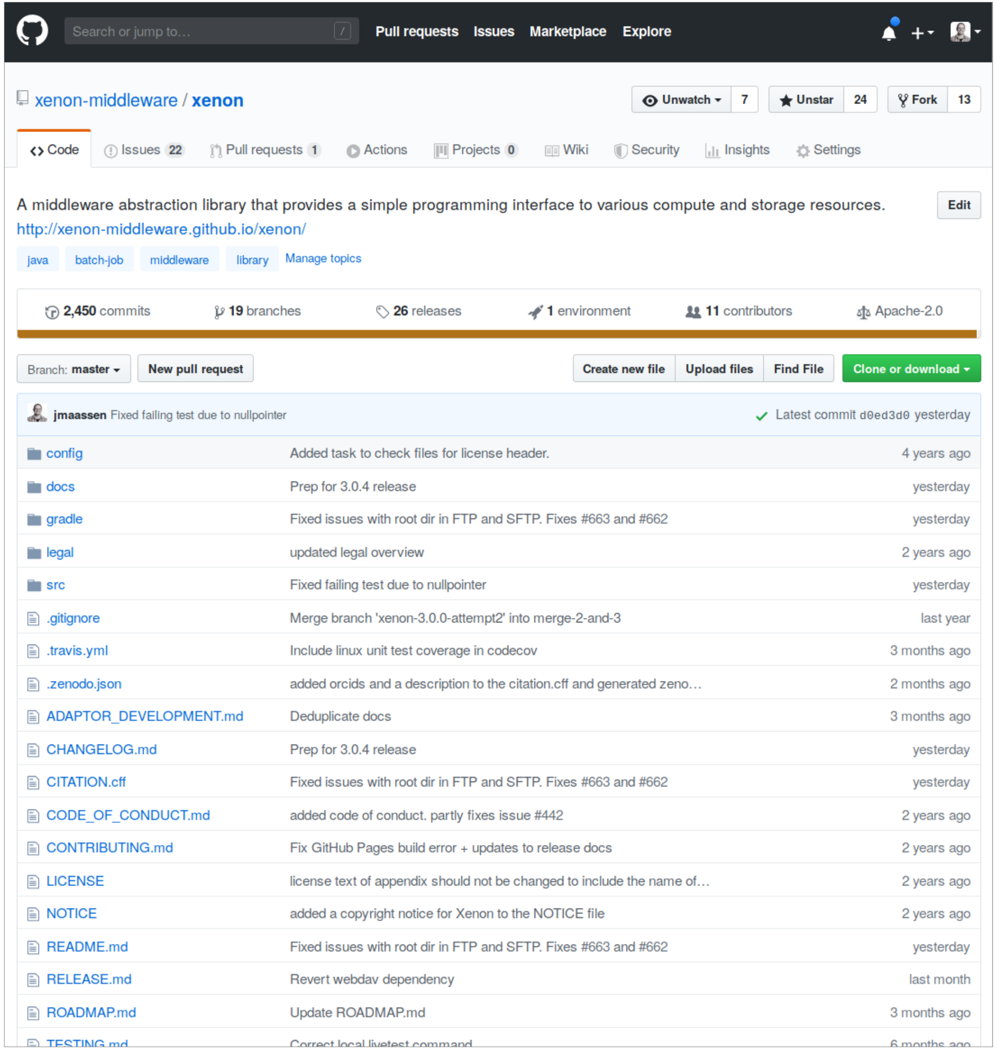
How does xenon help ?

Xenon offers simple programming interfaces to access remote file systems and schedulers.

Adaptors implement the functionality for different backends.

A CLI allows for easy use in scripts.





A middleware abstraction library that provides a simple programming interface to various compute and storage resources.

http://xenon-middleware.github.io/xenon/

jmaassen Fixed failing test due to nullpointer — Latest commit d0ed3d0 yesterday

config	Added task to check files for license header.	4 years ago
docs	Prep for 3.0.4 release	yesterday
gradle	Fixed issues with root dir in FTP and SFTP. Fixes #663 and #662	yesterday
legal	updated legal overview	2 years ago
src	Fixed failing test due to nullpointer	yesterday
.gitignore	Merge branch 'xenon-3.0.0-attempt2' into merge-2-and-3	last year
.travis.yml	Include linux unit test coverage in codecov	3 months ago
.zenodo.json	added orcids and a description to the citation.cff and generated zeno…	2 months ago
ADAPTOR_DEVELOPMENT.md	Deduplicate docs	3 months ago
CHANGELOG.md	Prep for 3.0.4 release	yesterday
CITATION.cff	Fixed issues with root dir in FTP and SFTP. Fixes #663 and #662	yesterday
CODE_OF_CONDUCT.md	added code of conduct. partly fixes issue #442	2 years ago
CONTRIBUTING.md	Fix GitHub Pages build error + updates to release docs	2 years ago
LICENSE	license text of appendix should not be changed to include the name of…	2 years ago
NOTICE	added a copyright notice for Xenon to the NOTICE file	2 years ago
README.md	Fixed issues with root dir in FTP and SFTP. Fixes #663 and #662	yesterday
RELEASE.md	Revert webdav dependency	last month
ROADMAP.md	Update ROADMAP.md	3 months ago
TESTING.md	Correct local livetest command	6 months ago

xenon

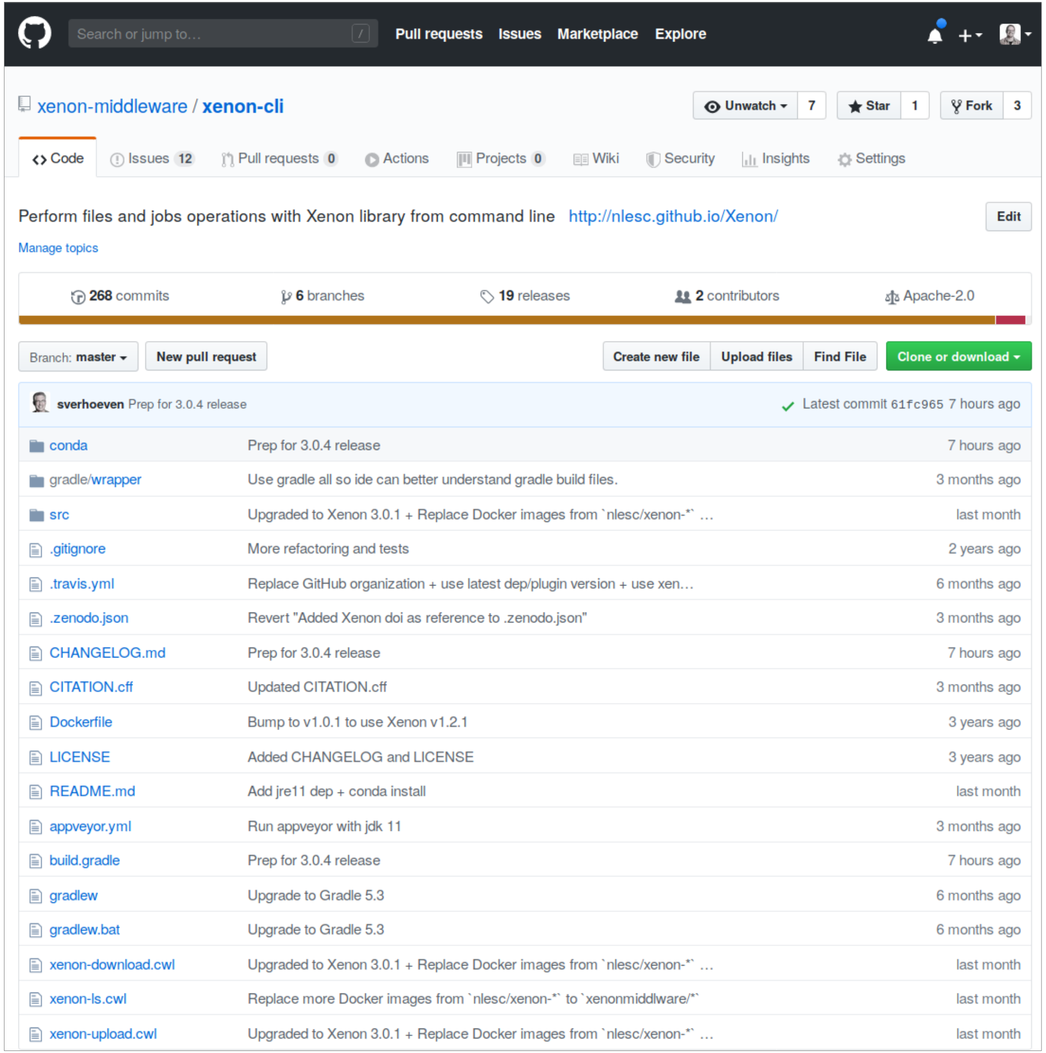
(the library)

Target audience is developers creating tools and (other) libraries.

Programming interface is kept simple, just focus on basic tasks:

- submit a job
- copy files

Good enough for 90% of the cases



xenon-middleware / xenon-cli

Perform files and jobs operations with Xenon library from command line http://nlesc.github.io/Xenon/

268 commits · 6 branches · 19 releases · 2 contributors · Apache-2.0

sverhoeven Prep for 3.0.4 release — Latest commit 61fc965 7 hours ago

conda	Prep for 3.0.4 release	7 hours ago
gradle/wrapper	Use gradle all so ide can better understand gradle build files.	3 months ago
src	Upgraded to Xenon 3.0.1 + Replace Docker images from `nlesc/xenon-*` …	last month
.gitignore	More refactoring and tests	2 years ago
.travis.yml	Replace GitHub organization + use latest dep/plugin version + use xen…	6 months ago
.zenodo.json	Revert "Added Xenon doi as reference to .zenodo.json"	3 months ago
CHANGELOG.md	Prep for 3.0.4 release	7 hours ago
CITATION.cff	Updated CITATION.cff	3 months ago
Dockerfile	Bump to v1.0.1 to use Xenon v1.2.1	3 years ago
LICENSE	Added CHANGELOG and LICENSE	3 years ago
README.md	Add jre11 dep + conda install	last month
appveyor.yml	Run appveyor with jdk 11	3 months ago
build.gradle	Prep for 3.0.4 release	7 hours ago
gradlew	Upgrade to Gradle 5.3	6 months ago
gradlew.bat	Upgrade to Gradle 5.3	6 months ago
xenon-download.cwl	Upgraded to Xenon 3.0.1 + Replace Docker images from `nlesc/xenon-*` …	last month
xenon-ls.cwl	Replace more Docker images from `nlesc/xenon-*` to `xenonmiddlware/*`	last month
xenon-upload.cwl	Upgraded to Xenon 3.0.1 + Replace Docker images from `nlesc/xenon-*` …	last month

xenon-cli

(the command line tool)

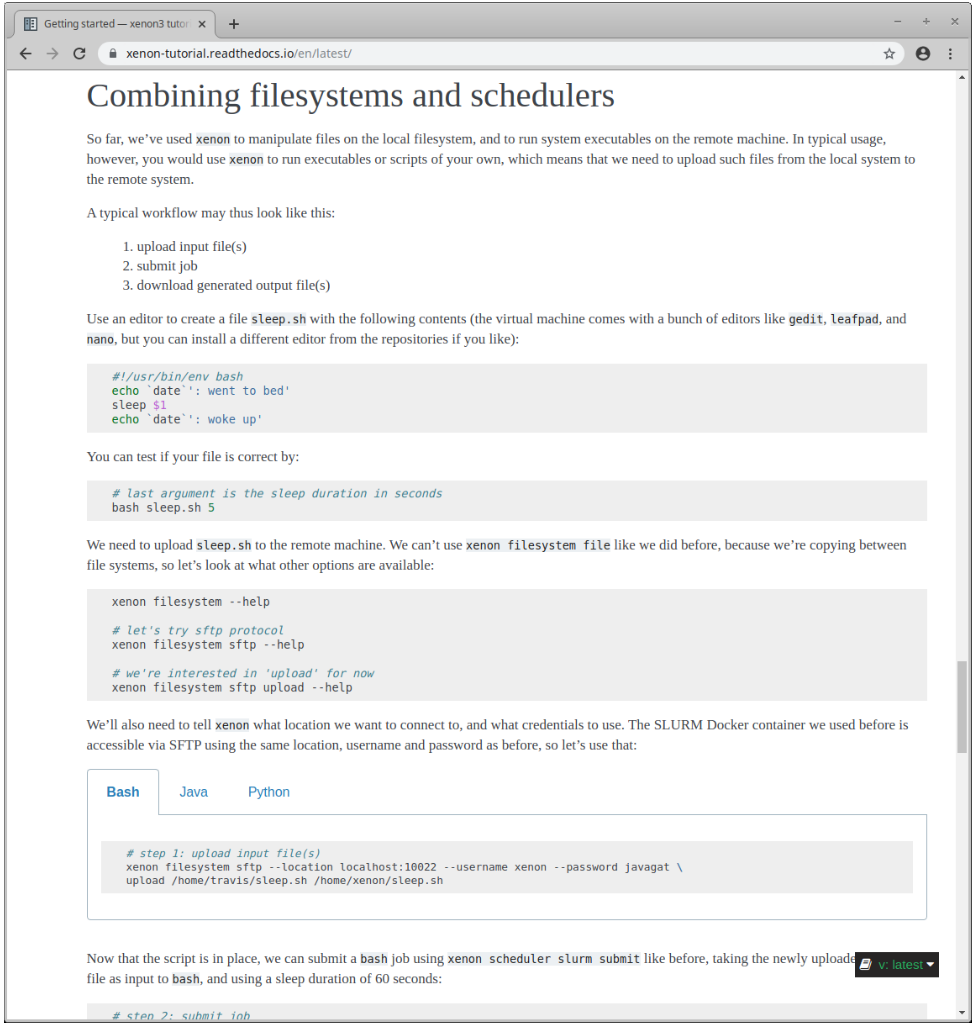
Target audience is users creating scripts and workflows

Offers uniform syntax to use different platforms with focus on basic tasks:

- submit a computation
- copy files

Good enough for 90% of the cases

tutorial


Getting started — xenon3 tutor
xenon-tutorial.readthedocs.io/en/latest/
v: latest

Combining filesystems and schedulers

So far, we've used `xenon` to manipulate files on the local filesystem, and to run system executables on the remote machine. In typical usage, however, you would use `xenon` to run executables or scripts of your own, which means that we need to upload such files from the local system to the remote system.

A typical workflow may thus look like this:

1. upload input file(s)
2. submit job
3. download generated output file(s)

Use an editor to create a file `sleep.sh` with the following contents (the virtual machine comes with a bunch of editors like `gedit`, `leafpad`, and `nano`, but you can install a different editor from the repositories if you like):

```
#!/usr/bin/env bash
echo `date`': went to bed'
sleep $1
echo `date`': woke up'
```

You can test if your file is correct by:

```
# last argument is the sleep duration in seconds
bash sleep.sh 5
```

We need to upload `sleep.sh` to the remote machine. We can't use `xenon filesystem file` like we did before, because we're copying between file systems, so let's look at what other options are available:

```
xenon filesystem --help

# let's try sftp protocol
xenon filesystem sftp --help

# we're interested in 'upload' for now
xenon filesystem sftp upload --help
```

We'll also need to tell `xenon` what location we want to connect to, and what credentials to use. The SLURM Docker container we used before is accessible via SFTP using the same location, username and password as before, so let's use that:

Bash　Java　Python

```
# step 1: upload input file(s)
xenon filesystem sftp --location localhost:10022 --username xenon --password javagat \
upload /home/travis/sleep.sh /home/xenon/sleep.sh
```

Now that the script is in place, we can submit a `bash` job using `xenon scheduler slurm submit` like before, taking the newly uploade file as input to `bash`, and using a sleep duration of 60 seconds:

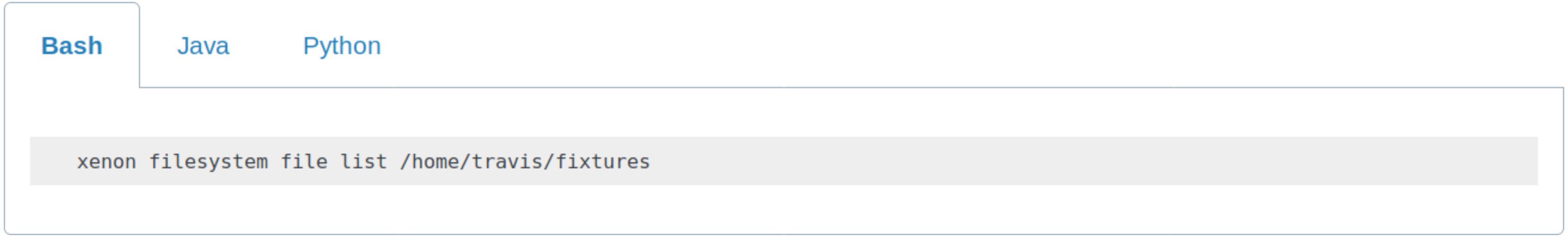
```
# step 2: submit job
```

https://xenon-tutorial.readthedocs.io

Explains how to use the
- Python API
- Java API
- Command Line Interface

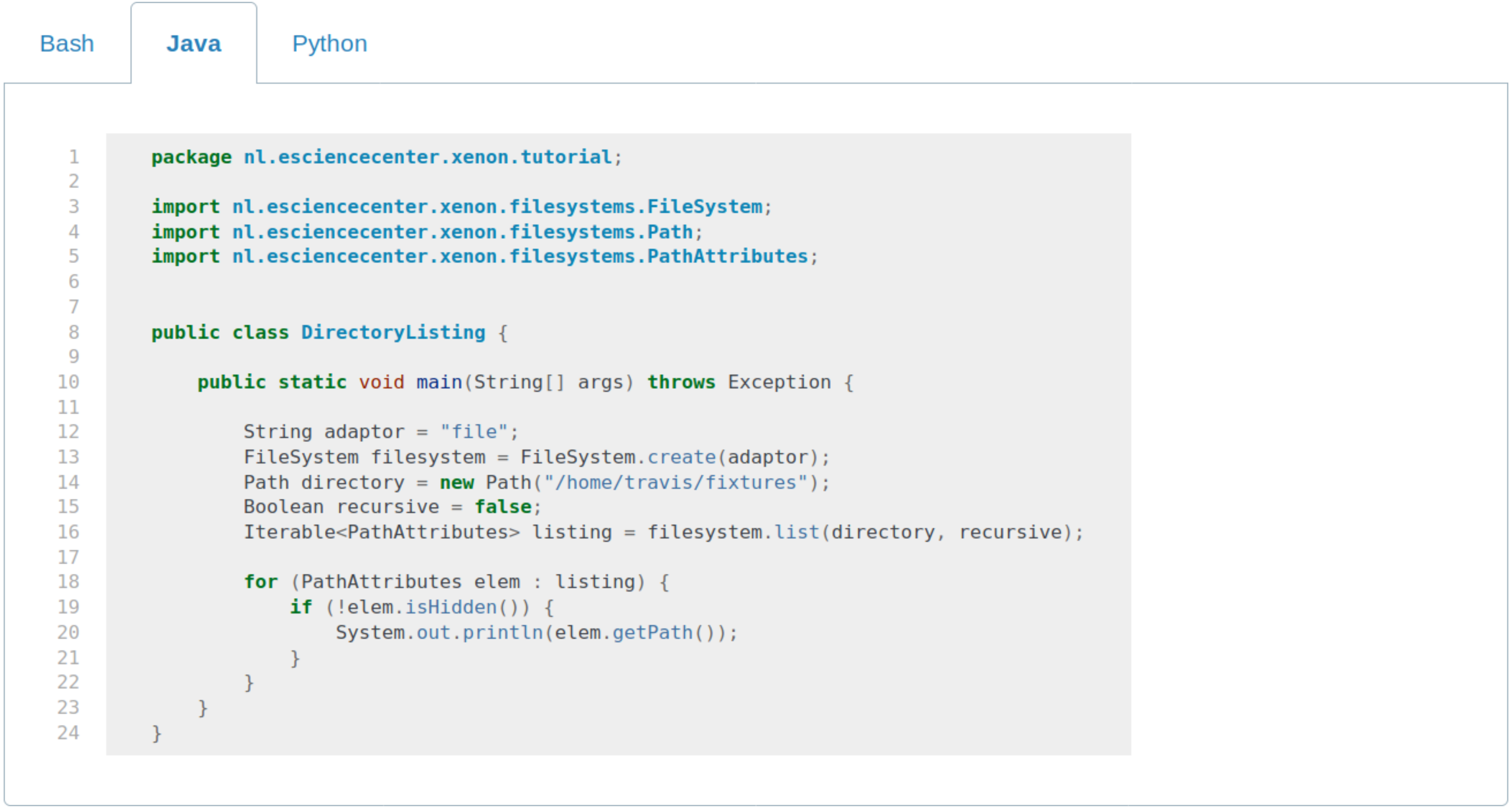
Provide Linux VM image with
- pre-installed xenon
- docker containers (slurm, sftp)

eScience center



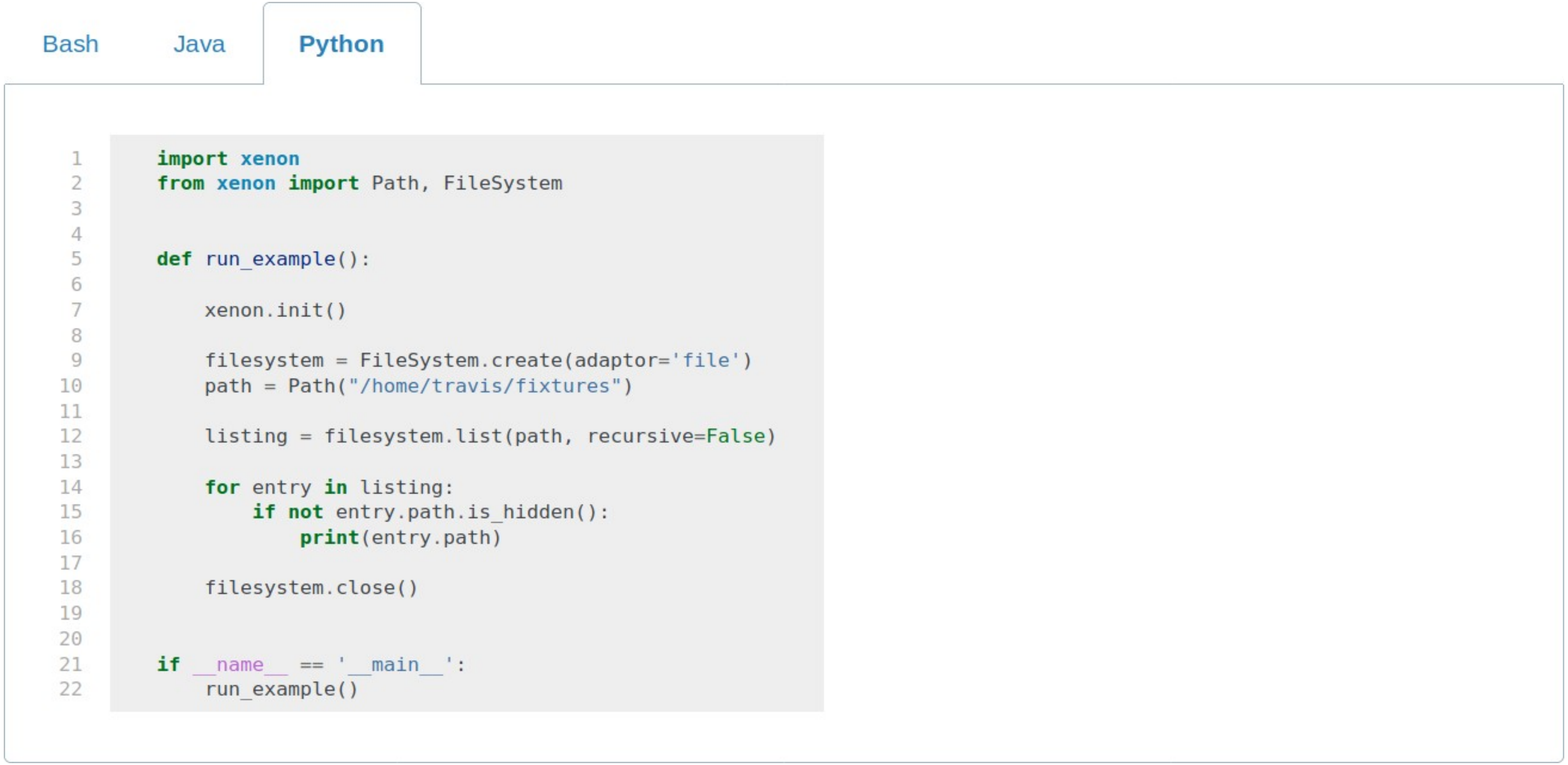
```
xenon filesystem file list /home/travis/fixtures
```

https://xenon-tutorial.readthedocs.io



```java
package nl.esciencecenter.xenon.tutorial;

import nl.esciencecenter.xenon.filesystems.FileSystem;
import nl.esciencecenter.xenon.filesystems.Path;
import nl.esciencecenter.xenon.filesystems.PathAttributes;


public class DirectoryListing {

    public static void main(String[] args) throws Exception {

        String adaptor = "file";
        FileSystem filesystem = FileSystem.create(adaptor);
        Path directory = new Path("/home/travis/fixtures");
        Boolean recursive = false;
        Iterable<PathAttributes> listing = filesystem.list(directory, recursive);

        for (PathAttributes elem : listing) {
            if (!elem.isHidden()) {
                System.out.println(elem.getPath());
            }
        }
    }
}
```

https://xenon-tutorial.readthedocs.io



```python
import xenon
from xenon import Path, FileSystem


def run_example():

    xenon.init()

    filesystem = FileSystem.create(adaptor='file')
    path = Path("/home/travis/fixtures")

    listing = filesystem.list(path, recursive=False)

    for entry in listing:
        if not entry.path.is_hidden():
            print(entry.path)

    filesystem.close()


if __name__ == '__main__':
    run_example()
```

https://xenon-tutorial.readthedocs.io

Portable HPC workflows based on

Snakemake + CONDA + XENON

Arnold Kuzniar

netherlands eScience center

Googling the cancer genome project

Identification and prioritization of cancer-causing structural variants (SVs)

UMCU team

Jeroen de Ridder (PI)

Wigard Kloosterman

Luca Santuari

Carl Shneider

eScience team

Lars Ridder (coordinator)

Arnold Kuzniar

Sonja Georgievska

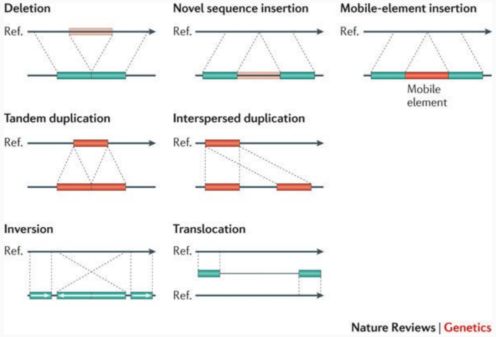
Jason Maassen

Stefan Verhoeven

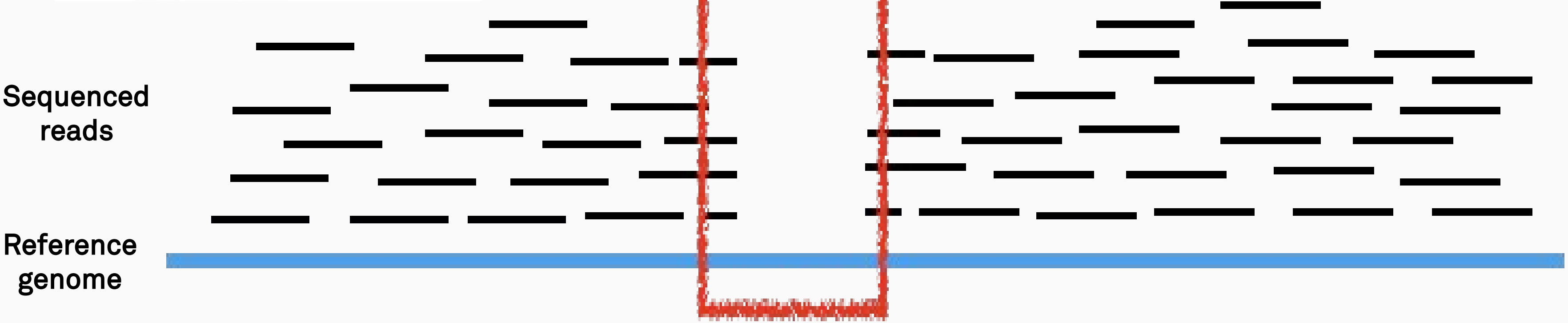
UMC Utrecht

netherlands eScience center

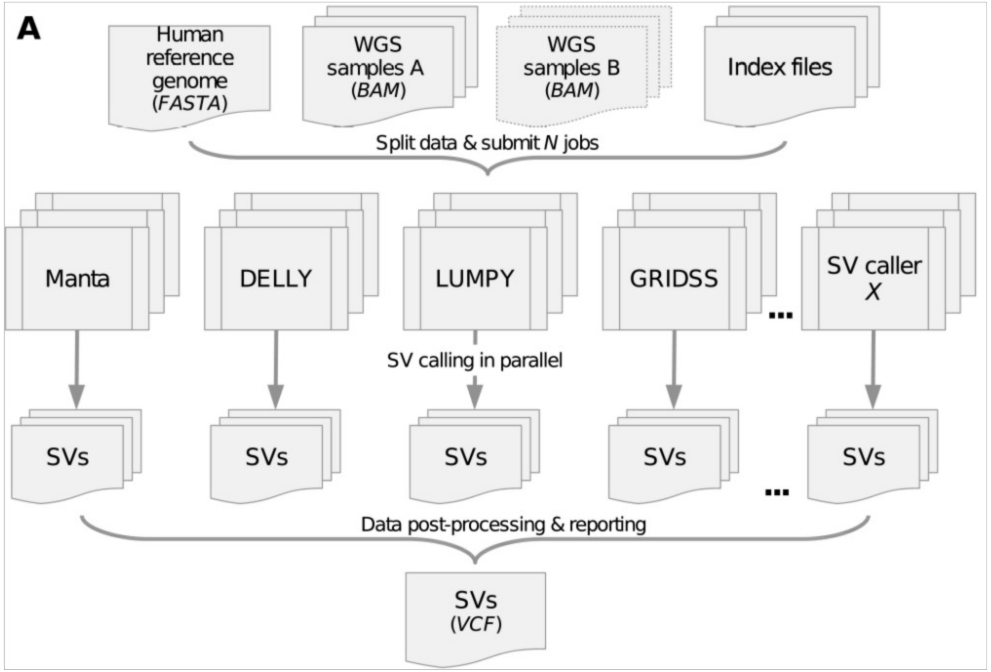
Challenge: reliable detection of SVs

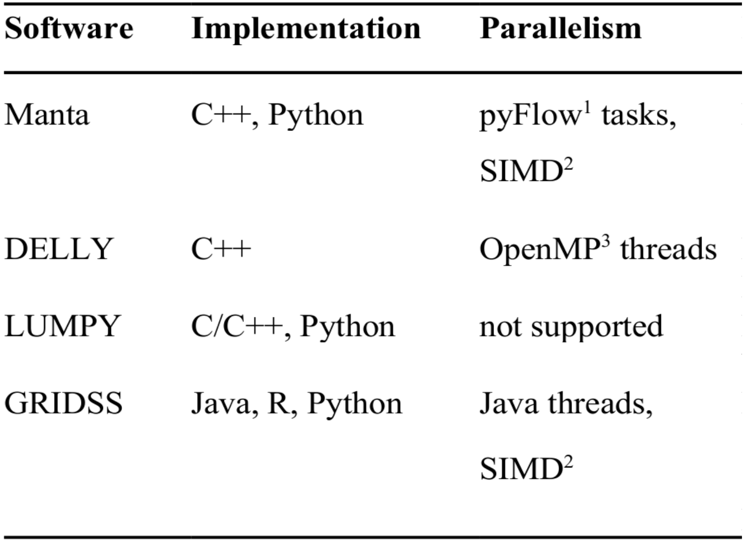


Nature Reviews | Genetics



sv-callers workflow

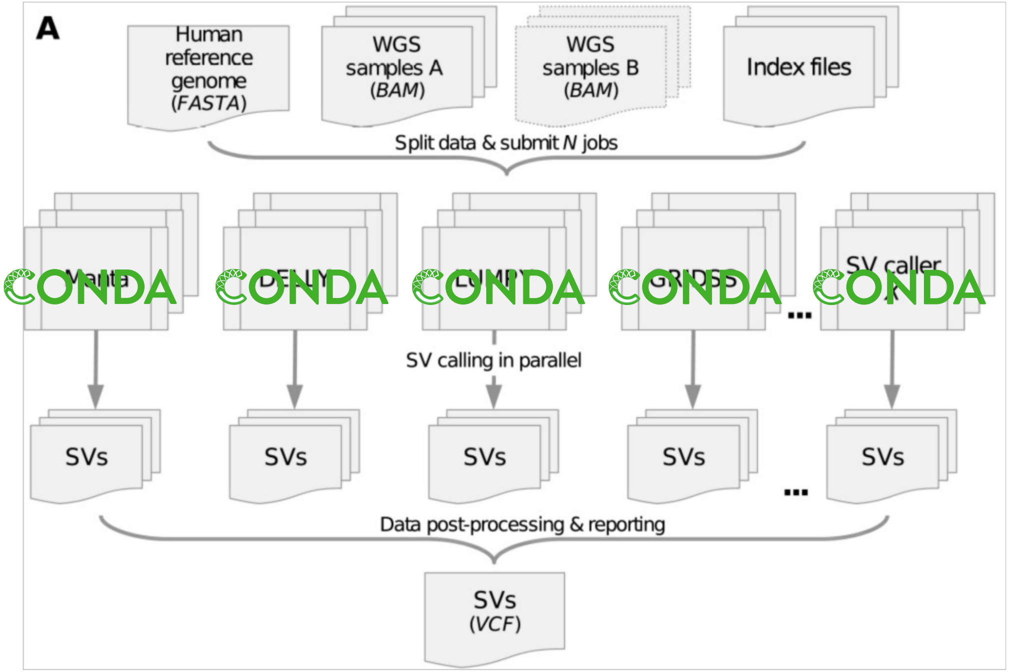




Software	Implementation	Parallelism
Manta	C++, Python	pyFlow[1] tasks, SIMD[2]
DELLY	C++	OpenMP[3] threads
LUMPY	C/C++, Python	not supported
GRIDSS	Java, R, Python	Java threads, SIMD[2]

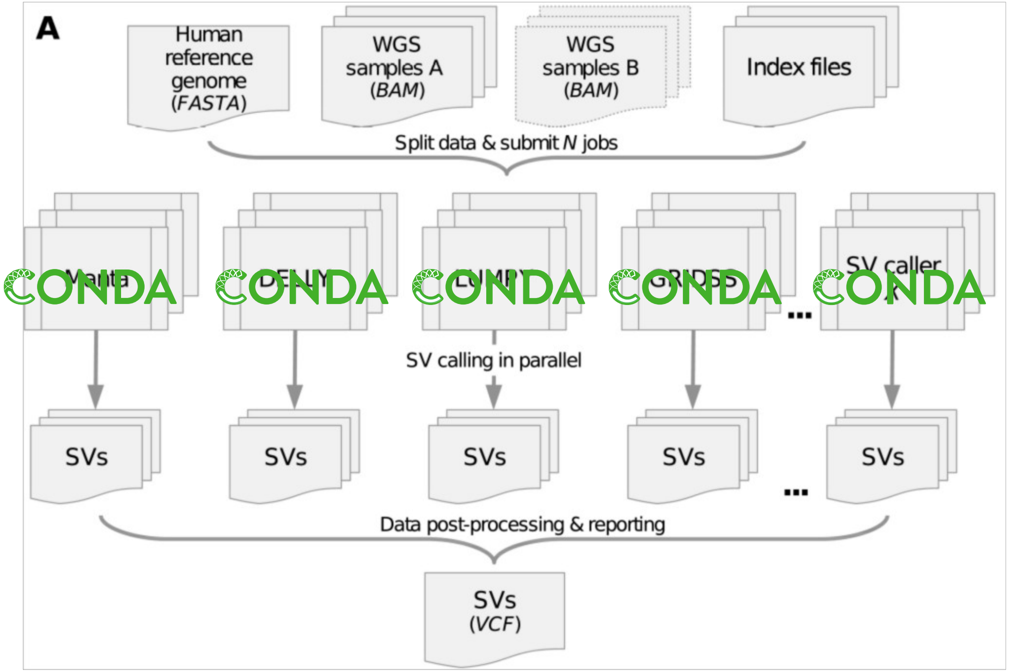
Combines existing SV detection tools into **a portable HPC workflow**

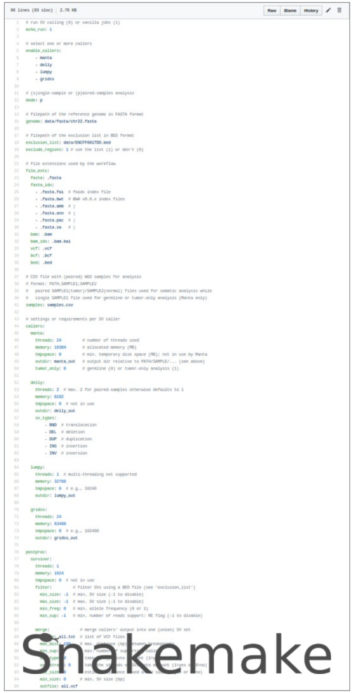
sv-callers workflow



Conda is used to enable automatic install of the tools on the target system

sv-callers workflow

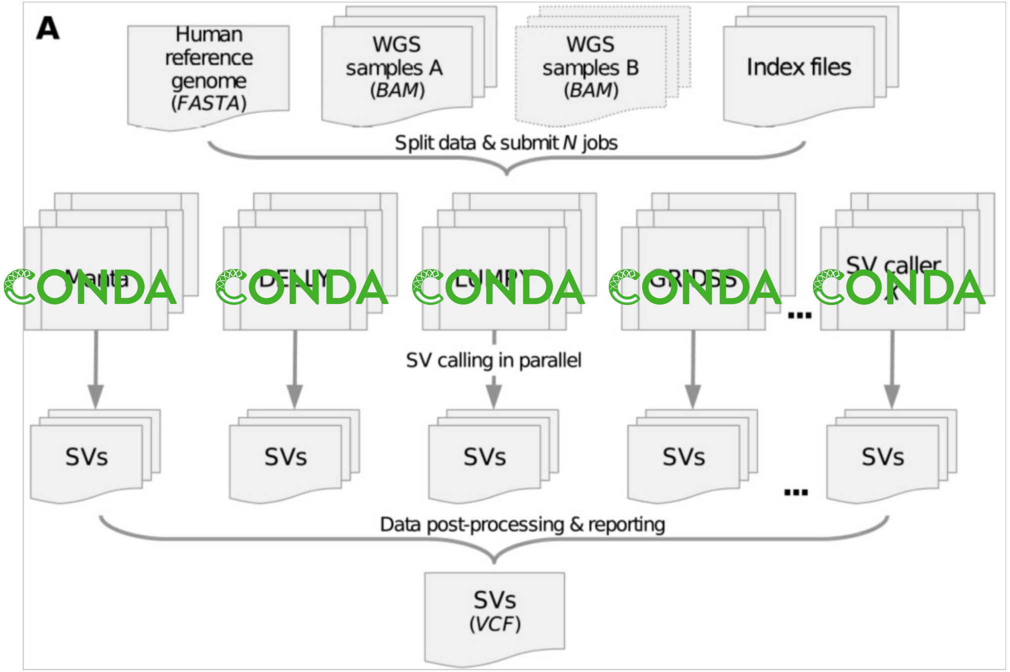


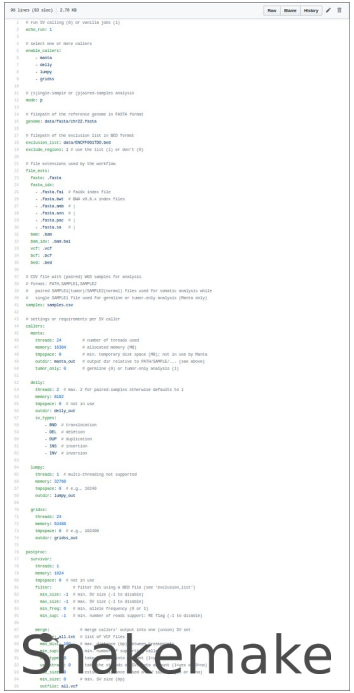


```
# run SV calling (0) or vanilla jobs (1)
echo_run: 1

# select one or more callers
enable_callers:
    - manta
    - delly
    - lumpy
    - gridss

# (s)ingle-sample or (p)aired-samples analysis
mode: p

# filepath of the reference genome in FASTA format
genome: data/fasta/chr22.fasta

# filepath of the exclusion list in BED format
exclusion_list: data/ENCFF001TDO.bed
exclude_regions: 1 # use the list (1) or don't (0)

# file extensions used by the workflow
file_exts:
  fasta: .fasta
  fasta_idx:
    - .fasta.fai  # faidx index file
    - .fasta.bwt  # BWA v0.6.x index files
    - .fasta.amb  # |
    - .fasta.ann  # |
    - .fasta.pac  # |
    - .fasta.sa   # |
  bam: .bam
  bam_idx: .bam.bai
  vcf: .vcf
  bcf: .bcf
  bed: .bed

# CSV file with (paired) WGS samples for analysis
# format: PATH,SAMPLE1,SAMPLE2
#   paired SAMPLE1(tumor)/SAMPLE2(normal) files used for somatic analysis while
#   single SAMPLE1 file used for germline or tumor-only analysis (Manta only)
samples: samples.csv

# settings or requirements per SV caller
callers:
  manta:
    threads: 24        # number of threads used
    memory: 16384      # allocated memory (MB)
    tmpspace: 0        # min. temporary disk space (MB); not in use by Manta
    outdir: manta_out  # output dir relative to PATH/SAMPLE/... (see above)
    tumor_only: 0      # germline (0) or tumor-only analysis (1)

  delly:
    threads: 2  # max. 2 for paired-samples otherwise defaults to 1
    memory: 8192
    tmpspace: 0  # not in use
    outdir: delly_out
    sv_types:
        - BND  # translocation
        - DEL  # deletion
        - DUP  # duplication
        - INS  # insertion
        - INV  # inversion

  lumpy:
    threads: 1  # multi-threading not supported
    memory: 32768
    tmpspace: 0  # e.g., 10240
    outdir: lumpy_out

  gridss:
    threads: 24
    memory: 63488
    tmpspace: 0  # e.g., 102400
    outdir: gridss_out

postproc:
  survivor:
    threads: 1
    memory: 1024
    tmpspace: 0  # not in use
    filter:         # filter SVs using a BED file (see 'exclusion_list')
      min_size: -1  # min. SV size (-1 to disable)
      max_size: -1  # max. SV size (-1 to disable)
      min_freq: 0   # min. allele frequency (0 or 1)
      min_sup: -1   # min. number of reads support: RE flag (-1 to disable)

    merge:            # merge callers' output into one (union) SV set
      [illegible]: all.txt  # list of VCF files
      max_dist: 100   # max. distance (bp) between breakpoints
      min_sup: [illegible]  # min. number of supporting callers
      [illegible]type: 0    # take [illegible] into [illegible]unt (1=y[illegible]
      use_strand: 0   # take the strands of SVs into account (1=yes or 0=no)
      [illegible]_size: 0   # esti[illegible]ance [illegible]sed on [illegible] size [illegible] or [illegible]no)
      min_size: 0     # min. SV size (bp)
      outfile: all.vcf
```

Snakemake is used to define the overall workflow

sv-callers workflow





```
# run SV calling (0) or vanilla jobs (1)
echo_run: 1

# select one or more callers
enable_callers:
    - manta
    - delly
    - lumpy
    - gridss

# (s)ingle-sample or (p)aired-samples analysis
mode: p

# filepath of the reference genome in FASTA format
genome: data/fasta/chr22.fasta

# filepath of the exclusion list in BED format
exclusion_list: data/ENCFF001TDO.bed
exclude_regions: 1 # use the list (1) or don't (0)

# file extensions used by the workflow
file_exts:
  fasta: .fasta
  fasta_idx:
    - .fasta.fai  # faidx index file
    - .fasta.bwt  # BWA v0.6.x index files
    - .fasta.amb  # |
    - .fasta.ann  # |
    - .fasta.pac  # |
    - .fasta.sa   # |
  bam: .bam
  bam_idx: .bam.bai
  vcf: .vcf
  bcf: .bcf
  bed: .bed

# CSV file with (paired) WGS samples for analysis
# format: PATH,SAMPLE1,SAMPLE2
#   paired SAMPLE1(tumor)/SAMPLE2(normal) files used for somatic analysis while
#   single SAMPLE1 file used for germline or tumor-only analysis (Manta only)
samples: samples.csv

# settings or requirements per SV caller
callers:
  manta:
    threads: 24         # number of threads used
    memory: 16384       # allocated memory (MB)
    tmpspace: 0         # min. temporary disk space (MB); not in use by Manta
    outdir: manta_out   # output dir relative to PATH/SAMPLE/... (see above)
    tumor_only: 0       # germline (0) or tumor-only analysis (1)

  delly:
    threads: 2  # max. 2 for paired-samples otherwise defaults to 1
    memory: 8192
    tmpspace: 0  # not in use
    outdir: delly_out
    sv_types:
        - BND  # translocation
        - DEL  # deletion
        - DUP  # duplication
        - INS  # insertion
        - INV  # inversion

  lumpy:
    threads: 1  # multi-threading not supported
    memory: 32768
    tmpspace: 0  # e.g., 10240
    outdir: lumpy_out

  gridss:
    threads: 24
    memory: 63488
    tmpspace: 0  # e.g., 102400
    outdir: gridss_out

postproc:
  survivor:
    threads: 1
    memory: 1024
    tmpspace: 0  # not in use
    filter:         # filter SVs using a BED file (see 'exclusion_list')
      min_size: -1  # min. SV size (-1 to disable)
      max_size: -1  # max. SV size (-1 to disable)
      min_freq: 0   # min. allele frequency (0 or 1)
      min_sup: -1   # min. number of reads support: RE flag (-1 to disable)

    merge:             # merge callers' output into one (union) SV set
      infile: all.txt  # list of VCF files
      max_dist: 100    # max. distance (bp) between breakpoints
      min_sup: [illegible]  # min. number of supporting callers
      use_type: 0      # take SV type into account (1=yes or 0=no)
      use_strand: 0    # take the strands of SVs into account (1=yes or 0=no)
      use_size: 0      # estimate distance based on SV size (1=yes or 0=no)
      min_size: 0      # min. SV size (bp)
      outfile: all.vcf
```

XENON

DAS 5

Utrecht Bioinformatics Center

SURF SARA

Snakemake then uses xenon to deploy on the available resources

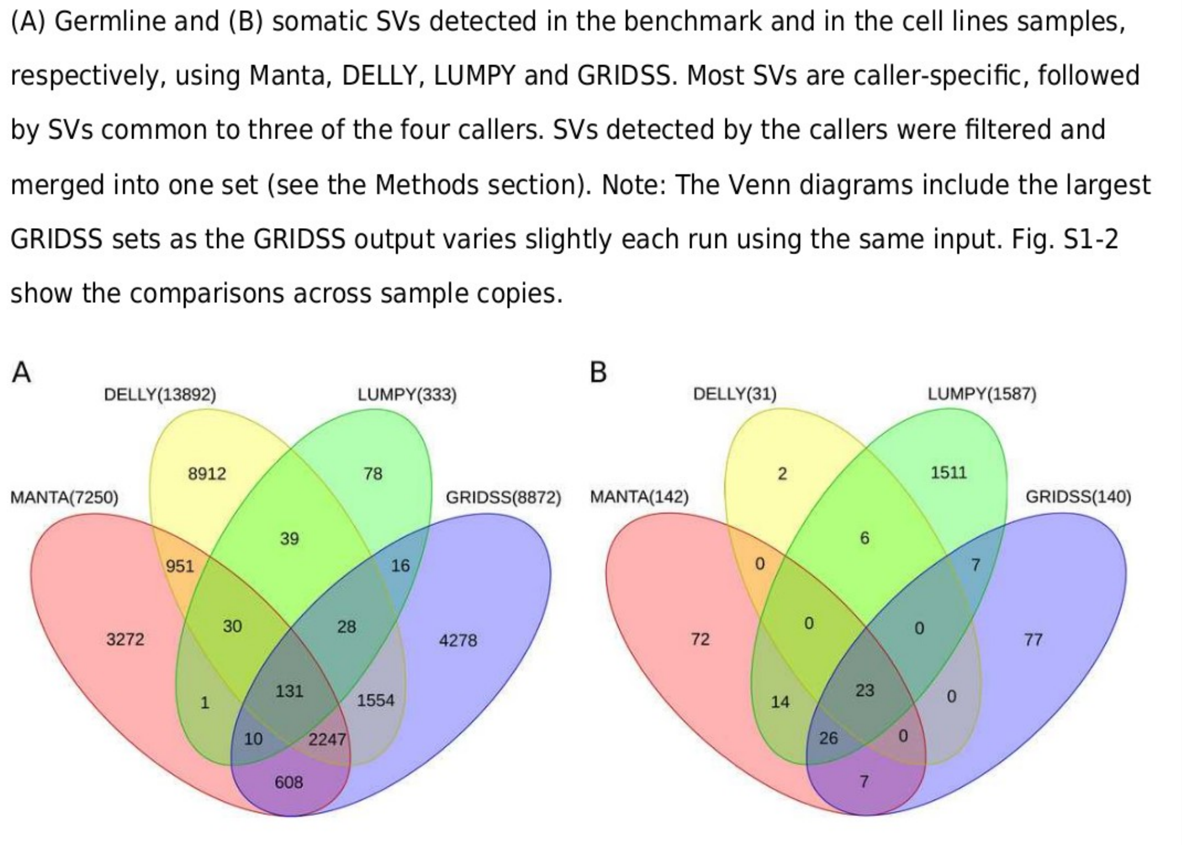
Snakemake → Xenon → GridEngine

```
snakemake -C echo_run=1 mode=p
enable_callers="['manta','delly','lumpy','gridss']"
--use-conda --latency-wait 30 --jobs 9
--cluster 'xenon scheduler gridengine
--location local:// submit --name smk.{rule} --inherit-env
--procs-per-node {threads} --start-single-process
--max-run-time 1 --max-memory {resources.mem_mb}
--working-directory . --stderr stderr-%j.log
--stdout stdout-%j.log' &>smk.log&
```

Snakemake → Xenon → Slurm

```
snakemake -C echo_run=1 mode=p
enable_callers="['manta','delly','lumpy','gridss']"
--use-conda --latency-wait 30 --jobs 9
--cluster 'xenon scheduler slurm
--location local:// submit --name smk.{rule} --inherit-env
--procs-per-node {threads} --start-single-process
--max-run-time 1 --max-memory {resources.mem_mb}
--working-directory . --stderr stderr-%j.log
--stdout stdout-%j.log' &>smk.log&
```

Results



(A) Germline and (B) somatic SVs detected in the benchmark and in the cell lines samples, respectively, using Manta, DELLY, LUMPY and GRIDSS. Most SVs are caller-specific, followed by SVs common to three of the four callers. SVs detected by the callers were filtered and merged into one set (see the Methods section). Note: The Venn diagrams include the largest GRIDSS sets as the GRIDSS output varies slightly each run using the same input. Fig. S1-2 show the comparisons across sample copies.

https://github.com/GooglingTheCancerGenome/sv-callers

**“sv-callers:
a highly portable parallel workflow for structural variant detection in whole-genome sequence data”**

Arnold Kuzniar, Jason Maassen, Stefan Verhoeven, Luca Santuari, Carl Shneider, Wigard P. Kloosterman, Jeroen de Ridder

Accepted for publication in PeerJ

eScience center

Xenon Roadmap

almost done: GridFTP, amazon-batch

in progress: azure-batch

ideas: iRODS (?), google storage (?), ...
language bindings for C++, C#, Go, etc.
better integration in snakemake,

suggestions are welcome!

Thanks!

https://github.com/xenon-middleware
https://xenon-tutorial.readthedocs.io
https://github.com/GooglingTheCancerGenome/sv-callers

xenon@esciencecenter.nl
j.maassen@esciencecenter.nl

www.esciencecenter.nl

Interested in Research Software ?

The Netherlands eScience Center is the Dutch national center of excellence for the development and application of research software to advance academic research.

Join the team !

✉ n.renaud@esciencecenter.nl

netherlands eScience center

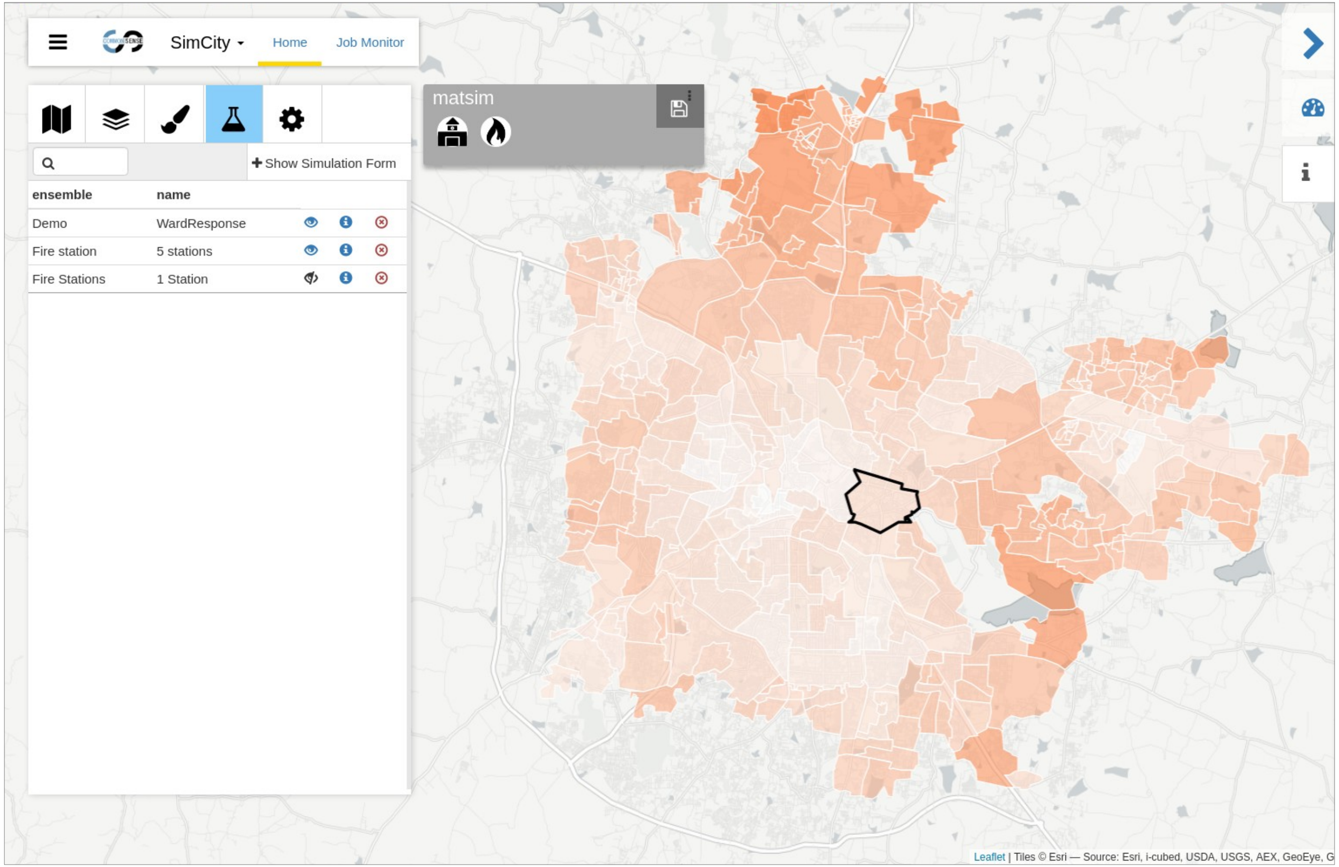
☎ +31 (0)20 460 4770

🌐 www.esciencecenter.nl

M blog.esciencecenter.nl

Extra slides

SIM-CITY


SimCity
Home
Job Monitor
Show Simulation Form
matsim
Leaflet | Tiles © Esri — Source: Esri, i-cubed, USDA, USGS, AEX, GeoEye, G

ensemble	name
Demo	WardResponse
Fire station	5 stations
Fire Stations	1 Station

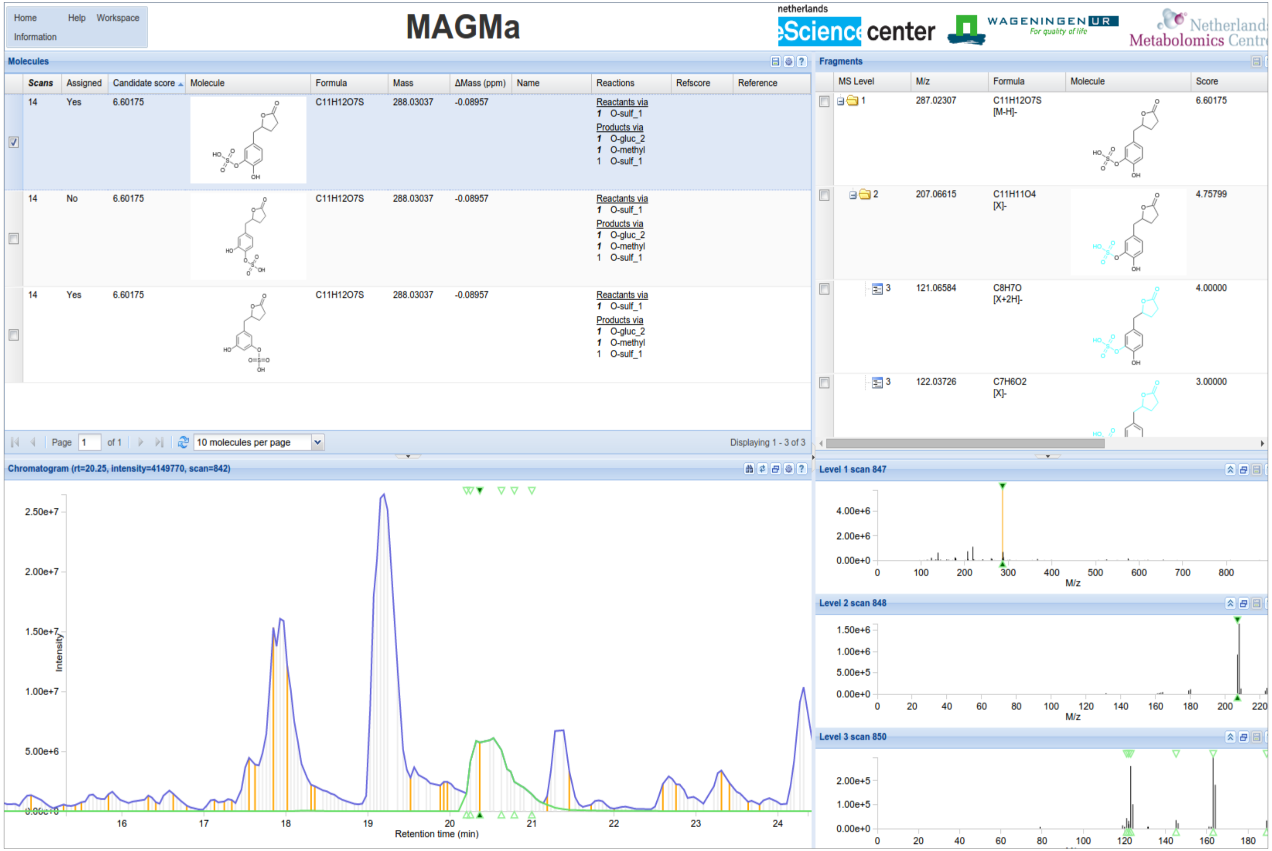
Goal: decision support for urban social economic complexity

Run urban planning simulations from an interactive user interface.

Example: location of fire stations

Simulations must be started on a remote compute cluster.

eMetabolomics



Home Help Workspace Information

MAGMa

Molecules

	Scans	Assigned	Candidate score	Molecule	Formula	Mass	ΔMass (ppm)	Name	Reactions	Refscore	Reference
☑	14	Yes	6.60175		C11H12O7S	288.03037	-0.08957		Reactants via 1 O-sulf_1 Products via 1 O-gluc_2 1 O-methyl 1 O-sulf_1		
□	14	No	6.60175		C11H12O7S	288.03037	-0.08957		Reactants via 1 O-sulf_1 Products via 1 O-gluc_2 1 O-methyl 1 O-sulf_1		
□	14	Yes	6.60175		C11H12O7S	288.03037	-0.08957		Reactants via 1 O-sulf_1 Products via 1 O-gluc_2 1 O-methyl 1 O-sulf_1		

Page 1 of 1 | 10 molecules per page | Displaying 1 - 3 of 3

Fragments

	MS Level	M/z	Formula	Molecule	Score
□	1	287.02307	C11H12O7S [M-H]-		6.60175
□	2	207.06615	C11H11O4 [X]-		4.75799
□	3	121.06584	C8H7O [X+2H]-		4.00000
□	3	122.03726	C7H6O2 [X]-		3.00000

Chromatogram (rt=20.25, intensity=4149770, scan=842)

Level 1 scan 847

Level 2 scan 848

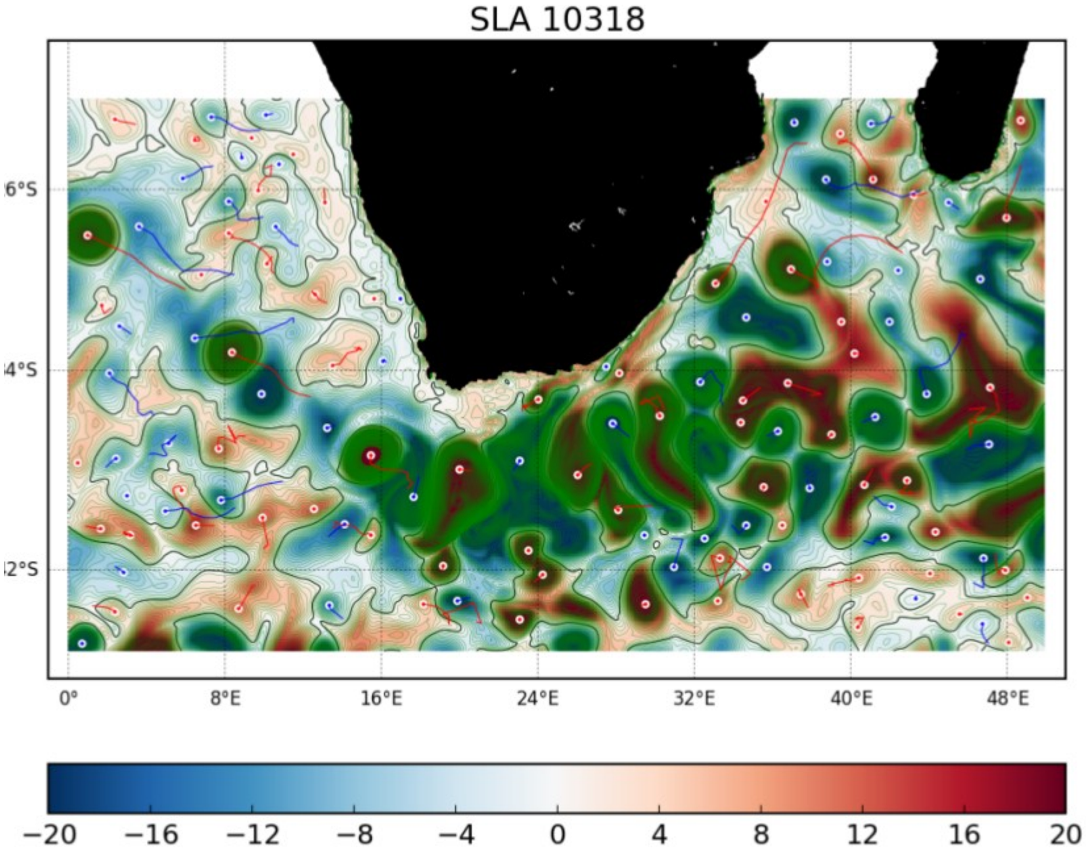
Level 3 scan 850

Goal: computer assisted metabolite profiling.

Determine how compounds are metabolized in the human body based on mass spectrometer data and metabolism rules.

Requires starting analysis computations on a server containing the metabolism database.

AMUSE / OMUSE



Goal: combine existing physics simulations into more complex models a simple python framework.

Originally for astrophysics (AMUSE), but now also for ocean modelling (OMUSE).

Requires starting simulations on multiple compute clusters and controlling them remotely.

eScience center