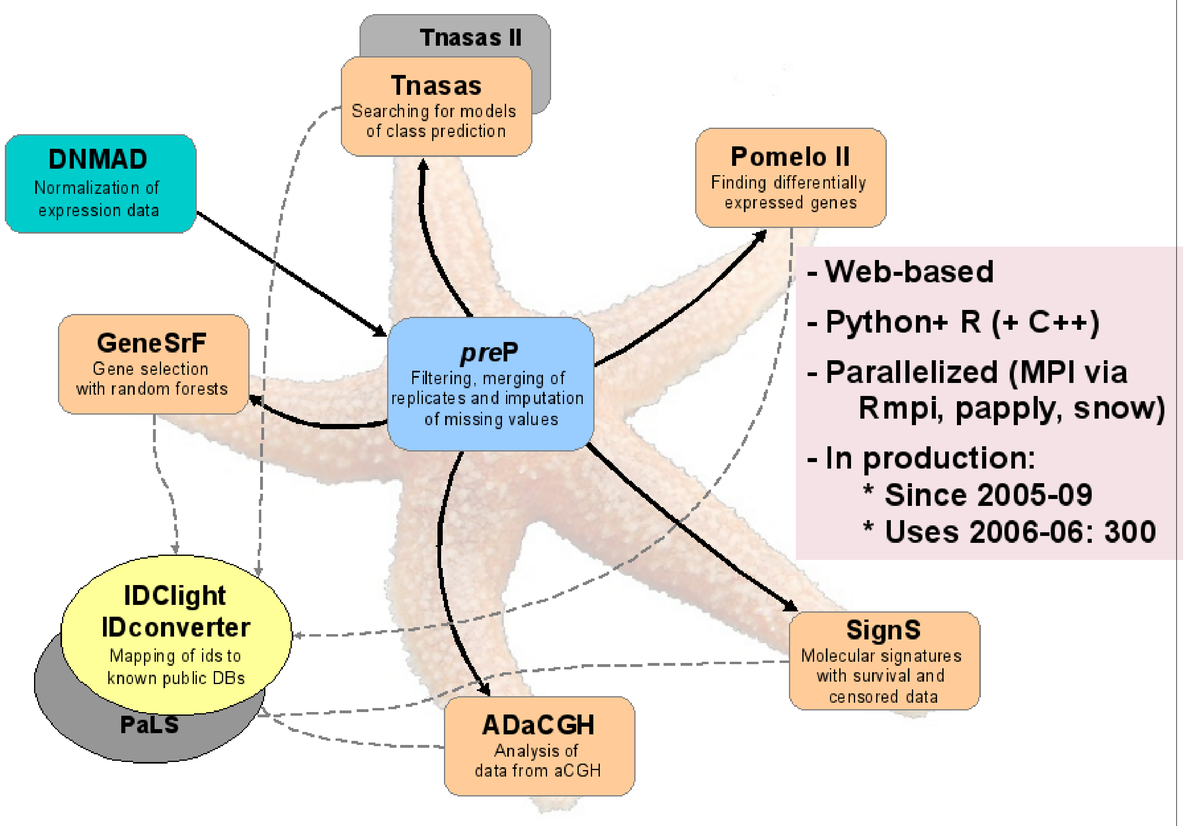
Asterias: an example of using R in a web-based bioinformatics suite of tools

Ramón Díaz-Uriarte, Andrés Cañada, Edward R. Morrissey,
Oscar Rueda, Andreu Alibés, David Casado
Statistical Computing Team
Centro Nacional de Investigaciones Oncológicas (CNIO)
Melchor Fernández Almagro, 3
Madrid, 28029, Spain
rdiaz@ligarto.org, http://ligarto.org/rdiaz

Asterias tools

- A suite of web-based tools for the analysis of genomic data
- You can send data from one application to another as shown by the arrows
- You can access each one directly



Availability

- Using it: http://www.asterias.info
- Project page: http://bioinformatics.org/asterias
- License: GNU GPL + Affero GPL.

Interlude: why is code availability important

- All the usual good reasons
- Web-based applications becoming a common platform for analyses:
  - How do we verify it does what it says . . .
  - . . . specially when results do not make sense?

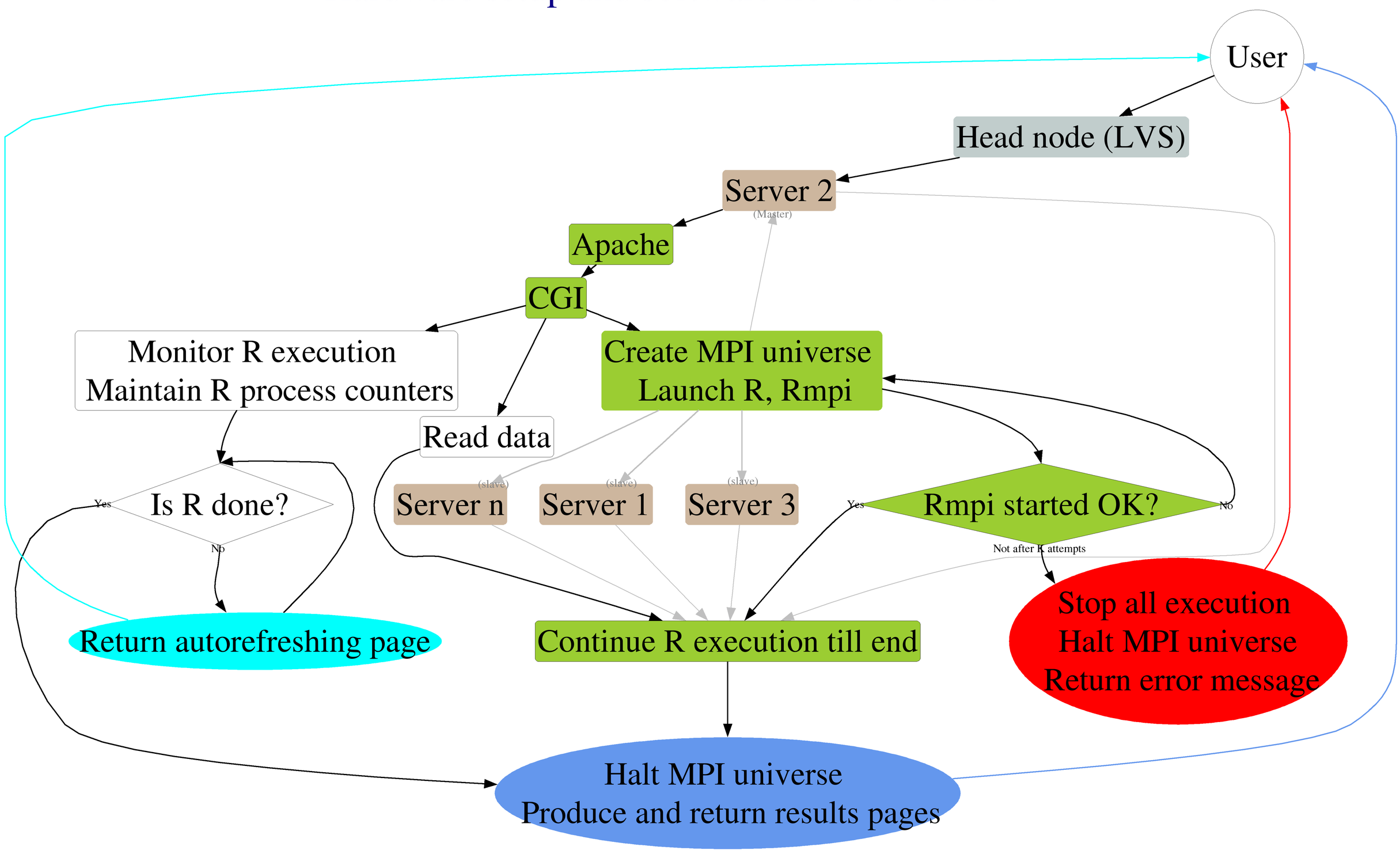
Future

- Simple installation
- Use virtual servers (Linux VServer, Xen) for security and deployment ease
- Further parallelization work: UPC, MPI + OpenMP
- Use of Grid
- Allow deployment as web services

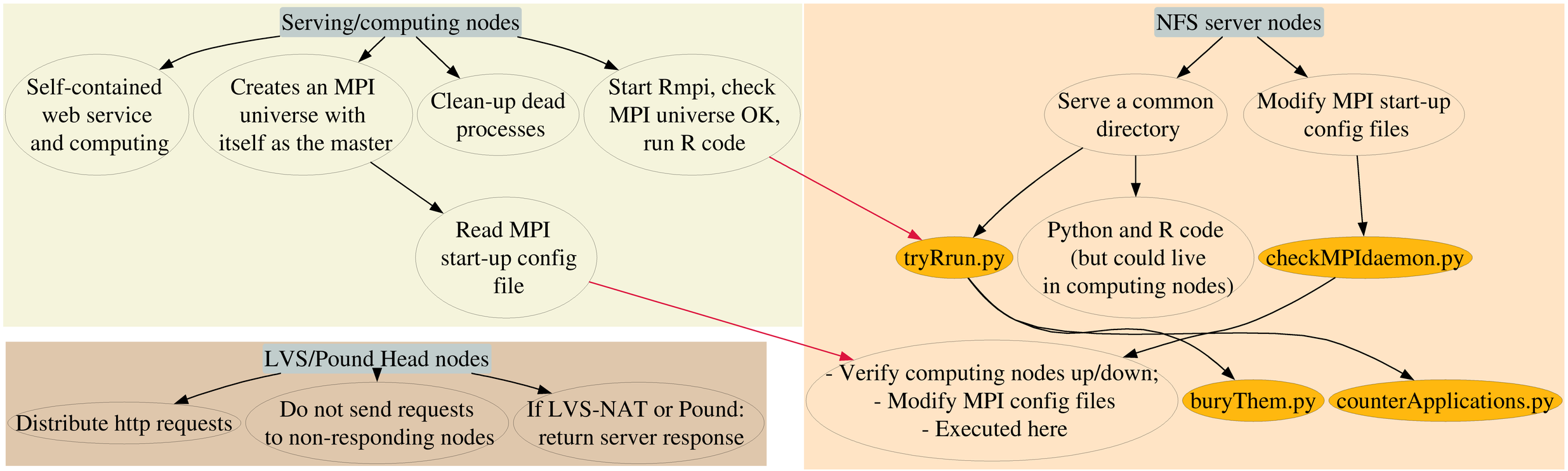
Acknowledgements

- Funding from Fundación de Investigación Médica Mutua Madrileña and Project TIC2003-09331-C02-02 of the Spanish Ministry of Education and Science.
- CNIO for infrastructure.
- Users and testers for bug reports.

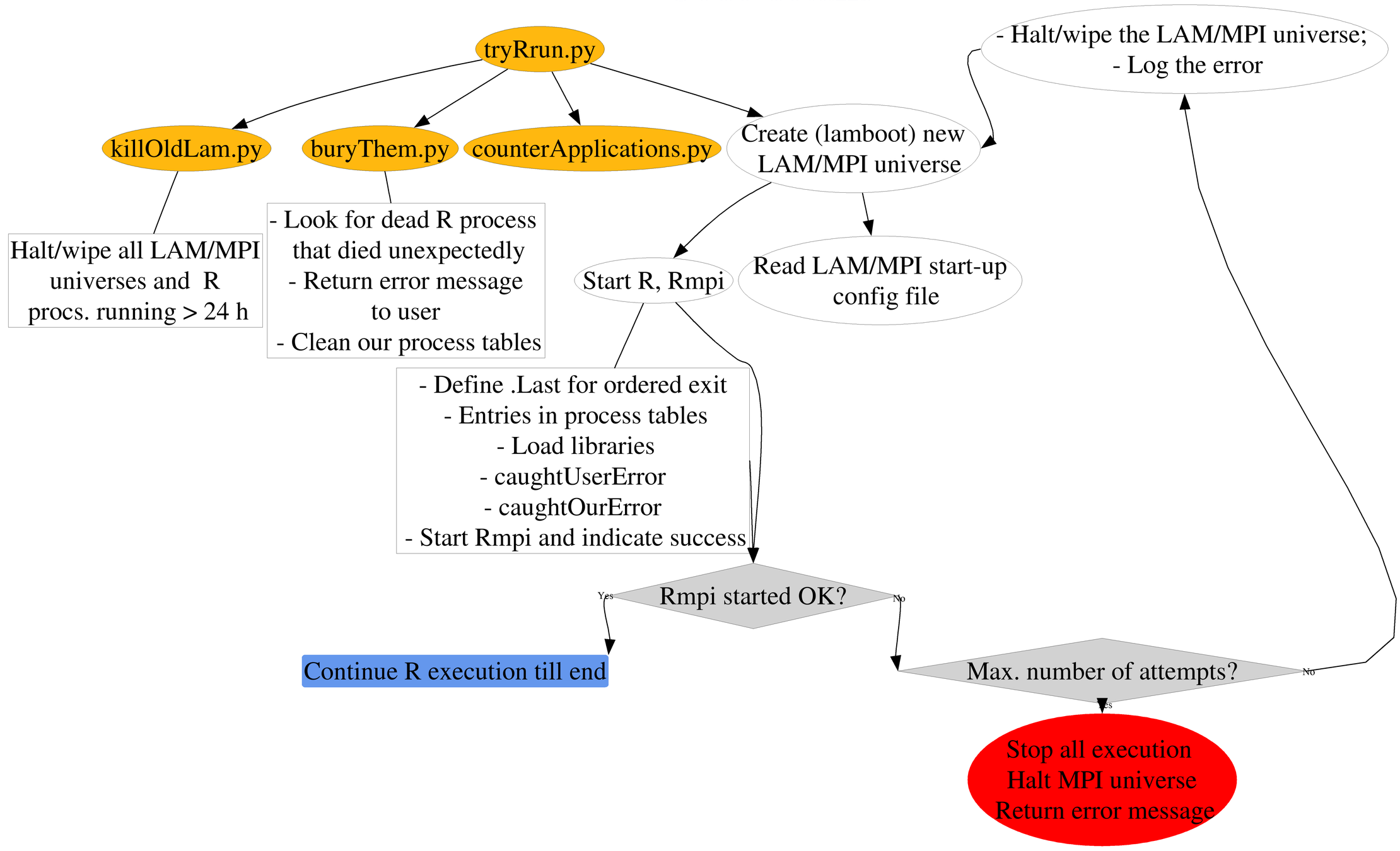
Hardware setup and software flows: overview



What each hardware component does



Code details



MPI setup

- MPI files/processes know nothing about LVS or Pound
- MPI files/processes know nothing about NFS (start-up files live in NFS directory, but MPI knows nothing of this)
- *checkMPIdaemon.py* modifies MPI start-up configuration files when nodes fail or resurrect
- MPI configuration files: simpler (and safer?) if we use an LAN that is internal and private to the cluster
- MPI is started and halted by apache

What the R code knows (or doesn't know)

- Knows nothing about web service
- Knows nothing about LVS/Pound
- Knows nothing about NFS serving node
- (R will read and write files that, under normal operation, live in a remote, NFS-accessed directory, but R knows nothing of these details)
- Uses an existing MPI universe
- Knows nothing of other MPI universes
- Failure, halting, creation of other MPI universes/Rmpi processes has no effect on this one
- R code can be run directly and stand-alone

High availability

- Not a needed part of the system, but desirable
- **LVS**
  - **ldirectord**, **mon**, etc
  - We use LVS + ldirectord: two nodes, one virtual IP
- **NFS serving nodes**
  - **RAID + controler** with 2 access nodes, **drbd** (but we experienced problems), ldirectord in charge of moving NFS resources (we experienced problems)
  - We use: **testing_repository.py** with one (or more) checking nodes outside the cluster, and two NFS serving nodes

Separation of layers

- All nodes can be computing node (i.e., run Rmpi slaves/masters) (minor issues in the NFS serving nodes)
- For ease of tracking errors and minimizing interference: web serving/computing nodes $\neq$ LVS/Pound head nodes $\neq$ NFS serving nodes.

What if this part fails . . .?

- In all cases: admins receive email
- **Computing node**
  - *ldirectord* (or mon, or pound, etc) will notice non-responding http server
  - *checkMPIdaemon.py* modifies MPI config files
- **LVS head node**
  - *heartbeat* will transfer virtual IP
  - Only users with currently active connection will notice
- **NFS serving node**
  - *testing_repository.py*, running from a machine(s) outside the cluster: remount common storage on backup NFS server node
  - (This machine needs administrative access to cluster nodes, better if using internal cluster subnet).
  - Only users with currently active connection will notice

Exception and error handling

- CGI and Python pre-processing: catch all user errors
- R: Liberal use of `try`:
  - Orderly exit R (and Rmpi) ($\rightarrow$ whole appl.)
  - Error message (function call), traceback, figure
    - caughtUserError, caughtOurError
  - (Few, if any, should ever be user errors: caught before by Python code)

Testing

- Whole system and regression testing: FunkLoad `http://funkload.nuxeo.org/`
  - A suite per application/component
  - Allows simple stress-testing
- Hourly run one FunkLoad test
  - Test head node, servers, MPI; send email if failure
- Other checks (head nodes, NFS servers, each node)