Introduction to the Tool Shed: Automated tool installation and robust dependency control for reproducible analysis

Galaxy Australasia Workshop
March 2014
Ross Lazarus
Outline

- Galaxy tools
- Tool Shed: VCS for tools
- Strict dependency control
- Low impedance distribution
- Demonstration
- Commodity reproducible analysis
Galaxy: tools, data and results
Galaxy Framework is tool agnostic

- Framework: focus of dev team
- Galaxy = UI, histories, viz, data, jobs..
- So, new component
- Tool Shed: tool server(s) for Galaxy
- Galaxy and tool distribution *decoupled*
Where does Galaxy fit?

Developers, Statisticians

FreeBayes  SNPEff  BWA

Galaxy

Biologists, Clinicians

COMMAND LINE SKILLS BARRIER DO NOT CROSS
Historical tool distribution model

- Galaxy distributions: code + tool wrappers
- Executables always from local system
- eg: bwa wrapper → system “bwa”
- If system updated, redo → bad! different!
- Weakened reproducibility claim
Where does Galaxy fit?

Developers, Statisticians

Tool Wrappers and dependencies

FreeBayes
SNPEff
BWA
Bowtie
TopHat Cuffdiff

Tool Sheds

Galaxy

Biologists, Clinicians
New tool distribution model

- Tool sheds ← tools and dependencies
- Galaxy distribution – few or no tools
- Galaxy admin: sees valid tool shed tools
- One click install → with all dependencies
What are dependencies?

- OS projects: Tophat, Bowtie, bwa,..,etc
- Tool devs write wrapper → UI for tool
- Rapid development; regular updates
- eg bwa 0.5.9rc16 is updated to 0.6.2
- If dependency updated, different results
- Reproducibility ← dependency control
Dependencies – regular updates

Developers, Statisticians

Tool Wrappers and dependencies

Galaxy

Biologists, Clinicians

FreeBayes

SNPEff

BWA

Bowtie

TopHat Cuffdiff

COMMAND LINE SKILLS BARRIER DO NOT CROSS
Competition within task areas!
Old style dependencies

- Wrapper version # controlled in XML
- Tool executable was system version
- eg bwa 0.5.9
- If updated, tool calls new version silently
- System updates break reproducibility
- Researchers want latest versions
- Sysadmin under pressure...
New Tool Shed dependencies

- Tool devs create dependency packages
- Update as become available
- Tool devs write XML wrapper → UI for tool
- All dependencies specified in XML
- Strict tool/dependency version control
- Old jobs rerun with old versions
- New jobs run with latest version
Tool shed tools

- Wrapper plus specific dependencies
- Packaged as a tgz/zip/bz2 archive
- Uploaded to a new Tool Shed repository
- Changed wrapper version?
- New tool version available in Tool Shed
- Admin sees when updates available
- Click to install or update or upgrade
Tool Shed

- Independent server in Galaxy source
- Team runs main and test Tool Sheds
- Anyone can run a tool shed!
- Developers can upload to tool sheds
- Users can review and rate tools
- Every version maintained in a VCS
Demonstration Plan

- Silly example – but could be any code!
- Will use laptop local servers
- Install a tool from a toolshed
- Run the new tool
- Explore archive structure and code
- Show how tool was generated
- If time, show tool upgrade process
Outline

- Galaxy tools
- Tool Shed: VCS for tools
- Strict dependency control
- Low impedance distribution
- Demonstration
- Commodity reproducible analysis
Tool Shed: strict dep control

- Tool Shed is the app store for Tools
- Uses the mercurial VCS for storage
- Provides/accepts zip/tgz/gz archives
- Specific name/layout requirements
- Deps in tool_configuration.xml
- Galaxy maintains isolated versions
- Old jobs use same (old) version
Tool Shed

- App store for Galaxy tools
- Admin can click to install
- Explicit tool versioning
- Enhanced reproducibility
- Enhanced sharing of tools
VIVA LA EVOLUCIÓN

GALAXY

http://usegalaxy.org