



State of Changeset Evolution



I: Changeset Evolution?



What is Mercurial

- Distributed Version Control
- Core Experience Goal,
 - Simple,
 - Safe,
 - Powerful,
 - Flexible,

History rewriting

- A strong selling point for DVCS:
 - helps with testing,
 - helps with debugging,
 - helps with code review,
 - ...
- New complexities
 - different types of conflicts
 - "*mutable*" history record
 - new collaboration workflow



Changeset Evolution

- Phases
- Set of history-rewriting commands
- Topics
- Obsolescence History



Safe

- Preventing (phases, checks)
- Detecting (instability detection)
- Solving (evolve)



Simple

- No need to learn about everything at once
- Easy-to-use intermediate "bad" state
- Keep work scope well defined within *topic*
- Unified and predictable propagation



Powerful

- intermediate states are useful locally
 - commit-centric workflow
 - focus on current work, deal with consequences later,
- history of evolution can be audited
- painless distributed collaboration
- predictable propagation (*again*)



II: Progress since 2019

Workflow Improvement

- toward a "single head per name" world
 - obsolescence aware computation
 - single head enforcement
 - (multiple heads still possible)
- topic centric workflow
 - stack semantic and notations
 - new publishing and exchange semantic
 - strong integration in Heptapod

Safety improvement

- improvement pre-rewrite checks
 - phase divergence
 - content divergence
- reliable automatic evolution
 - orphan
 - phase divergence
 - content divergence
- *hg pull --confirm*

Other progress

- improved *hg obslog* command
- continue / abort support
- improved automatic evolution
- usable *hg rewind* command
- many small fixes and improvements,
- experimental *hg fixup* command



III: The Road Ahead



Area that need work

- Simple and Safe Workflow
- Automatic Resolution
- History Edition commands
- Performance

Simple and Safe Workflow

- Topic-Namespace solves many problems:
 - clear UI distinction:
your work / others' work,
 - filter pulling,
 - permission control for drive-by contributors,
 - safety around history rewriting,
 - server controlled configuration.



Automatic Resolution

- currently in a working reliable state
- some corner cases (e.g. splitting/folding) need improvement
- formal definition and analysis planned
- record and share merge-conflict

History edition commands


- we have a good core set of commands,
- *hg rewind* needs more work,
- improving advanced command use-case like *hg absorb* and *hg fixup*
- improving experience around *reordering/extraction/insertion* of changeset
- expect more fixes and polishing,



Performance

- lack of caching and incremental computation:
 - phase,
 - obsolescence,
 - orphan
 - phase divergence
 - content divergence
 - topic
- Obsmarkers Discovery
 - working prototypes since 2017
 - PHD in progress for an official version (2021-2024)

Non experimental ?

- core Semantic ✓
- basic command set
 - rewriting ✓
 - viewing ✓
 - undoing ✓!
- instability
 - detection ✓
 - resolution ✓
- pre-rewrite safety
 - unstability ✓
 - users insulation ✗
- performance
 - local ✗
 - exchange ✗
- upstreaming 



Conclusion



Close to "completion"

- no large unknown remains,
- topic namespace need experiment and adjustment,
- large performance work ahead,
- wrap up a consistent user experience by default.



Want to help

- Contributions:
 - User feedback,
 - Bug reports,
 - Patches,
- Funding,
- World Peace¹.

[1] or some geopolitical stability at least