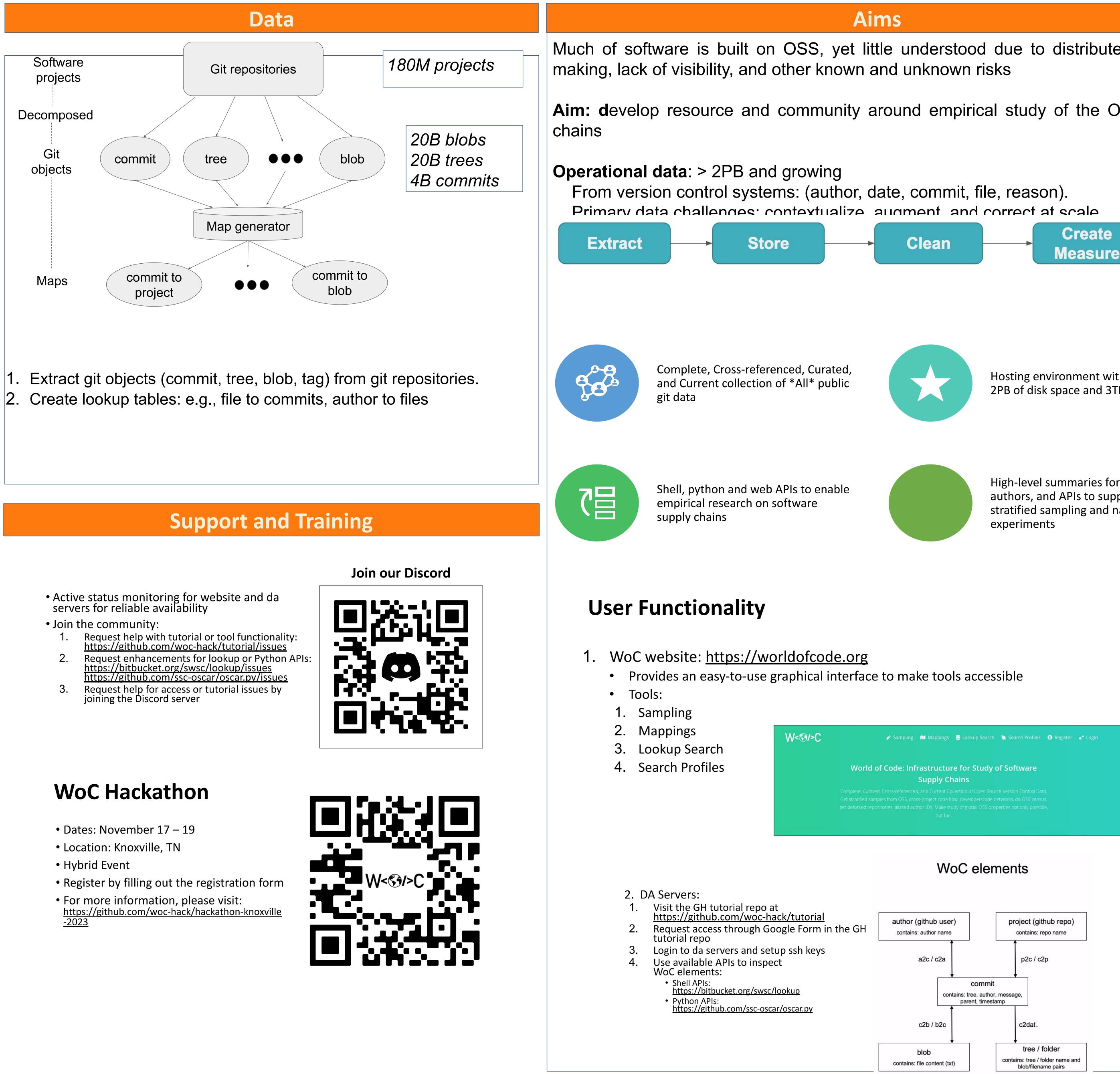


2120429 World Of Code (WoC): The development of curated code resource to support research in software engineering Audris Mockus¹, Jim Herbsleb² and Bogdan Vasilescu² ¹University of Tennessee, Knoxville; ²Carnegie Mellon University Curation: Data Correction^{IV} Data Aims Much of software is built on OSS, yet little understood due to distributed decision **Correcting Developer Identity Errors in VCS** 180M projects making, lack of visibility, and other known and unknown risks Operational data for software supply chain is of low quality esp. Aim: develop resource and community around empirical study of the OSS supply identities of individuals. For example, names and email addresses of users of people on networks are often misspelled. chains 20B blobs **Data:** 820M commits from 31M projects with 14M author IDs blob 20B trees **Challenge**: The 3Ds – Deal with lack of ground truth, Disambiguate, **Operational data**: > 2PB and growing 4B commits From version control systems: (author, date, commit, file, reason). De-anonymize Primary data challenges: contextualize augment and correct at scale **Method**: Supervised classification methods to predict if a pair of IDs are a match i.e. belong to the same physical entity. Use an Active Create Extract Store Clean Learner to handle lack of training data Measure commit to blob • Author distances based on 'fingerprints': ►I ← Phase 2 ← ►I ← Complete, Cross-referenced, Curated, Hosting environment with 8 servers Define and Current collection of *All* public Create Attach 2PB of disk space and 3TB of RAM predictors pairwise ACTIVE fingerprint for string comparison Compare.







High-level summaries for projects, authors, and APIs to support stratified sampling and natural

M1 ≠ M2

 $M2 \neq M3$

 $M1 \neq M3$

ACTIVE LEARNEF

Linkage ()

<u>M2</u>

With need for

Preliminary

Classifier

comparisor

Completeness (provenance), Ability to reconstruct past states of the world Support Supply-Chain Software Development Individual activity over projects Code/API spread over time **Create Universal Version History tools** Detect/Fix orphan vulnerabilities Identify license violations Establish quality of the code Curate LLM training data

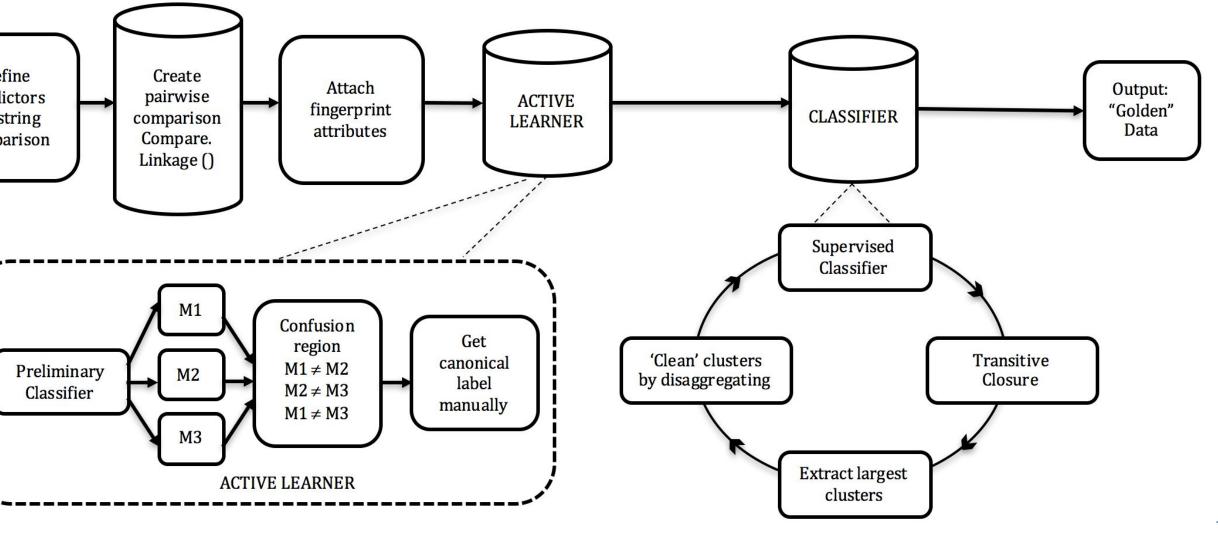
E.g., Orphan Vulnerabilities

- or package manager.
- available.
- Not in a linkable library
- Not in a package manager



• Name, Email, First name, Last name, User name & their Frequency.

• Files touched similarity, Time-zone similarity, Commit text similarity



Examples of Support ed Research

Better empirical research: escape convenience sampling

 Vulnerabilities in copied code that still exist in a project after they are discovered and fixed in another project. • Code is cloned and committed, not imported from a library

• Link to the original code does not exist or is not readily

• Often overlooked part of the software supply chain. • Becomes part of the new project