# CROSS-LINGUISTIC LINKED DATA

## Dateninfrastruktur für Diversity Linguistics

Robert Forkel

18.9.2015

Max-Planck-Institut für Menschheitsgeschichte

# Diversity Linguistics

- Umfasst Teile von Historischer Linguistik, Typologie und Sprachdokumentation
- beschäftigt sich mit **vielen** oder **kleinen** Sprachen
- typische Daten:
  - Wortliste
  - Grammatikskizze
  - Phoneminventar
  - meist graue Literatur
  - traditionelle Publikationskanäle funktionieren kaum noch

# Cross-Linguistische Daten - ein Beispiel

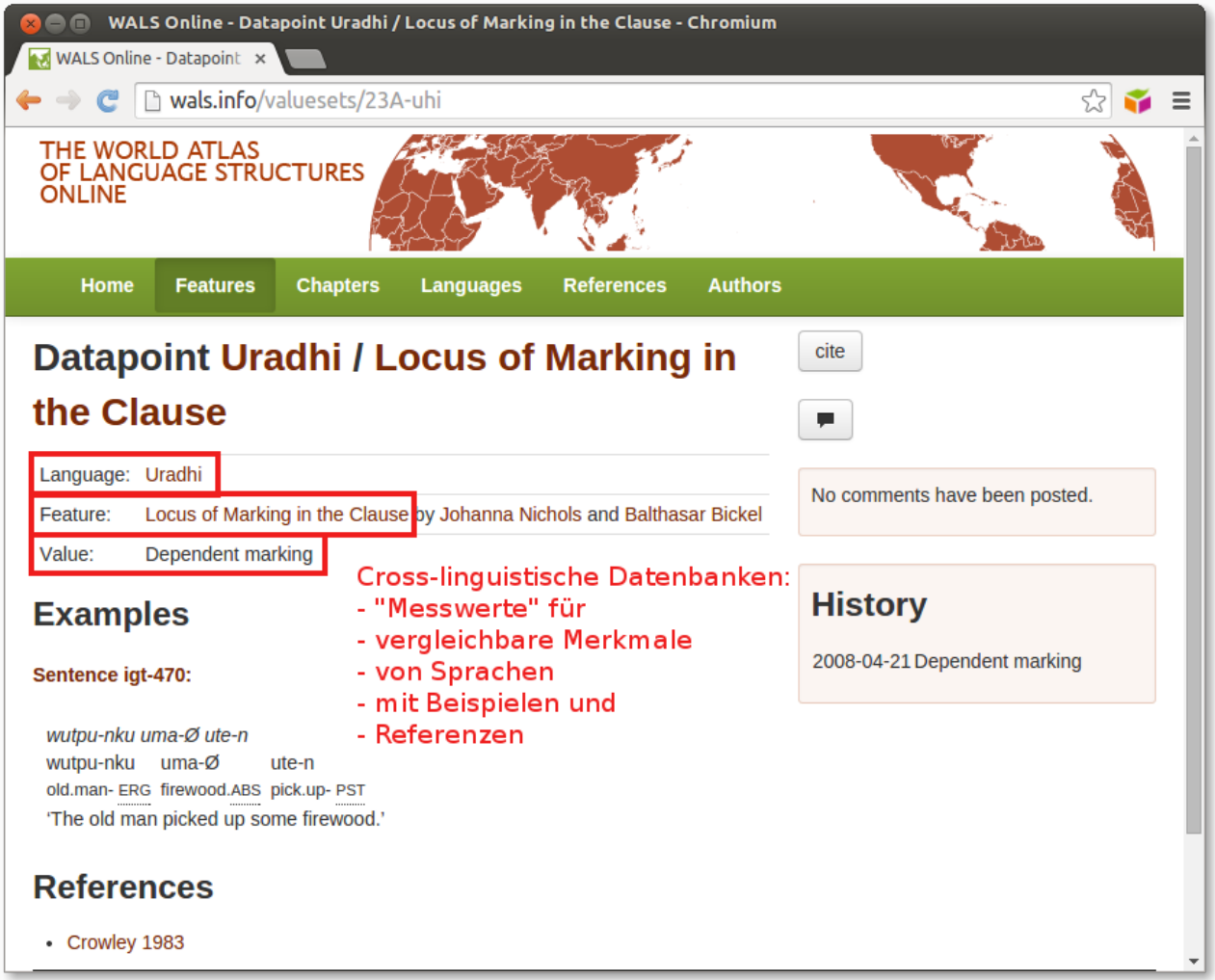



**Figure:** Ein WALS datapoint illustriert typische cross-linguistische Datensammlungen.

# Cross-Linguistische Daten

- Wir leben also fast im Naturwissenschaftlichen Shangri-La der Forschungsdaten – simple Messreihen
- abgesehen von der Kalibration der ”Messgeräte” :), die zwar hochsensibel sind, aber zu Inkonsistenz neigen
- und dem Umstand, dass Datengeber und Datennehmer sich kaum überschneiden

# Das CLLD Projekt

- Von der MPG für 4 Jahre gefördert, anfangs am MPI EVA in Leipzig, jetzt am MPI SHH in Jena.
- Brücke zwischen Datengebern und Datennehmern
- Datenpublikationsplattform für Diversity Linguistics:
  - Referenzdatenbanken: Sprachkatalog und Bibliographie
  - Datenjournals: Dictionaria für Wörterbücher und JCLD für Datenbanken
  - Typologische und lexikalische ”standalone” Datenbanken
  - Publikationsformen sind an traditionellen Vorbildern orientiert: Journal, Buchreihe

Wir müssen also in vielen Fällen ein grundlegendes Problem wissenschaftlicher Datenbanken lösen, nämlich Zitierfähigkeit (also Zugang zu älteren Versionen) mit Aktualisierbarkeit in Einklang zu bringen.

# RESEARCH DATA MANAGEMENT MIT GITHUB

# “Data is code”

Linguistische Forschungsdaten haben genug mit code gemein, um tools, workflows und best practices aus der Open Source Software-Entwicklung zu borgen.

Unsere Daten sind

- Text
- oft in zeilenbasierter Form (etwa CSV)
- kleine Datenmenge
- häufig offen zugänglich

# git und GitHub

**git**
- source code management tool
- Ähnlich wie *CVS* oder *Subversion*, aber *distributed*
- $\Longrightarrow$ jeder checkout ist voll funktionsfähiges repository

**GitHub**
- Hosting Plattform für git repositories
- Ergänzt git mit zusätzlichen Kollaborations-tools, insbesondere pull requests
- webhooks erlauben Integration mit anderen Services, etwa
  - Archivierung via ZENODO
  - continuous integration via Travis-CI

# KOLLABORATIVE DATENPFLEGE MIT GITHUB

# Data is code - Übersetzungshilfe

Best practices der open source Softwareentwicklung übertragen auf Datenpflege:

| | |
|---|---|
| **commit history** | audit trail für alle Änderungen |
| **merge** | Prozedur zum Zusammenführen von Änderungen mehrerer Autoren |
| **pull request** | Prozedur für Einreichung und open peer review neuer Daten |
| **release** | Publikation |
| **continuous integration** | Validierung |
| **fork** | Transparenter Wechsel der Verantwortlichkeiten |

# Data is code - Beispiel Tsammalex

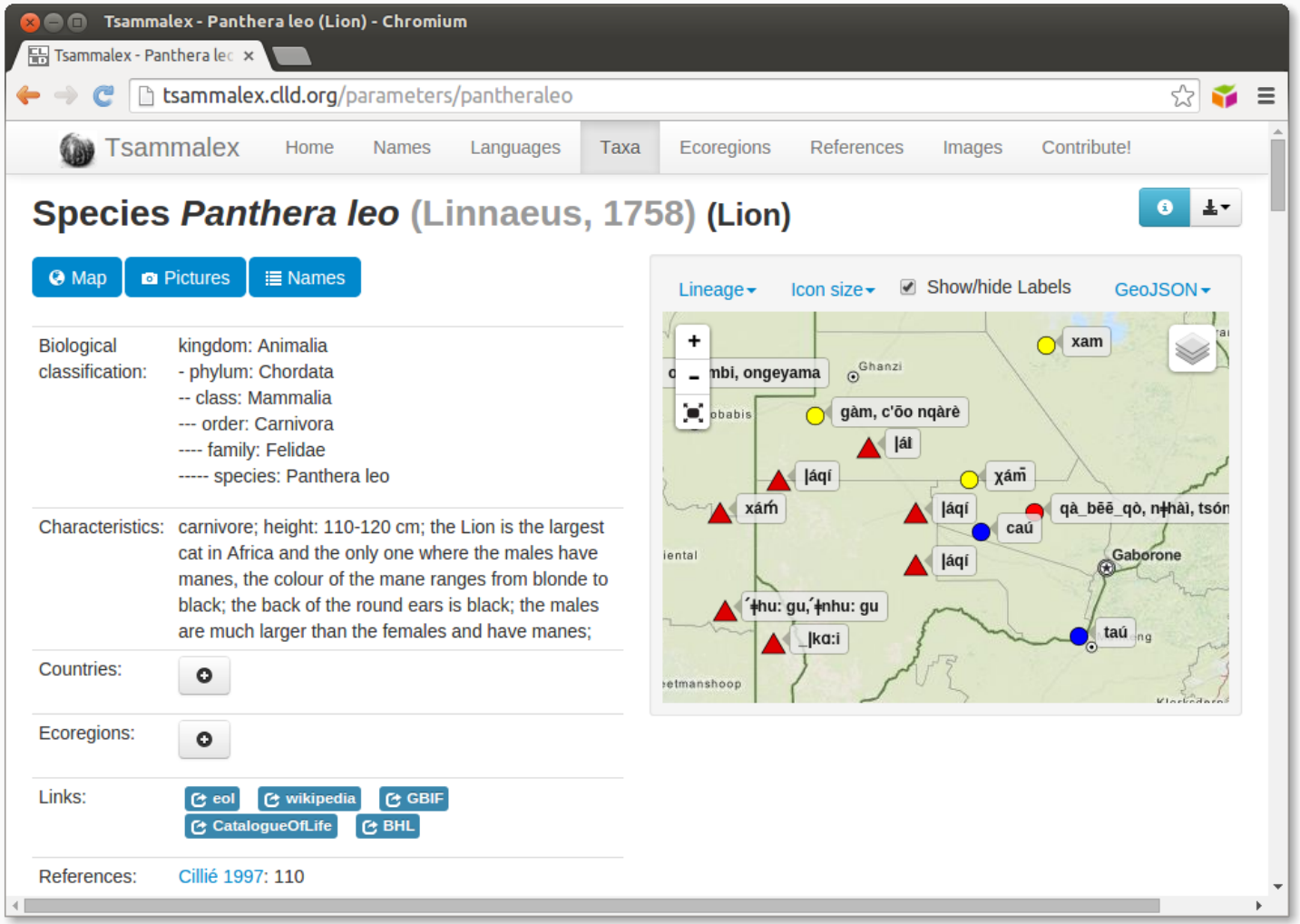



Figure: Tsammalex Applikation

# Data is code - Tsammalex data repository

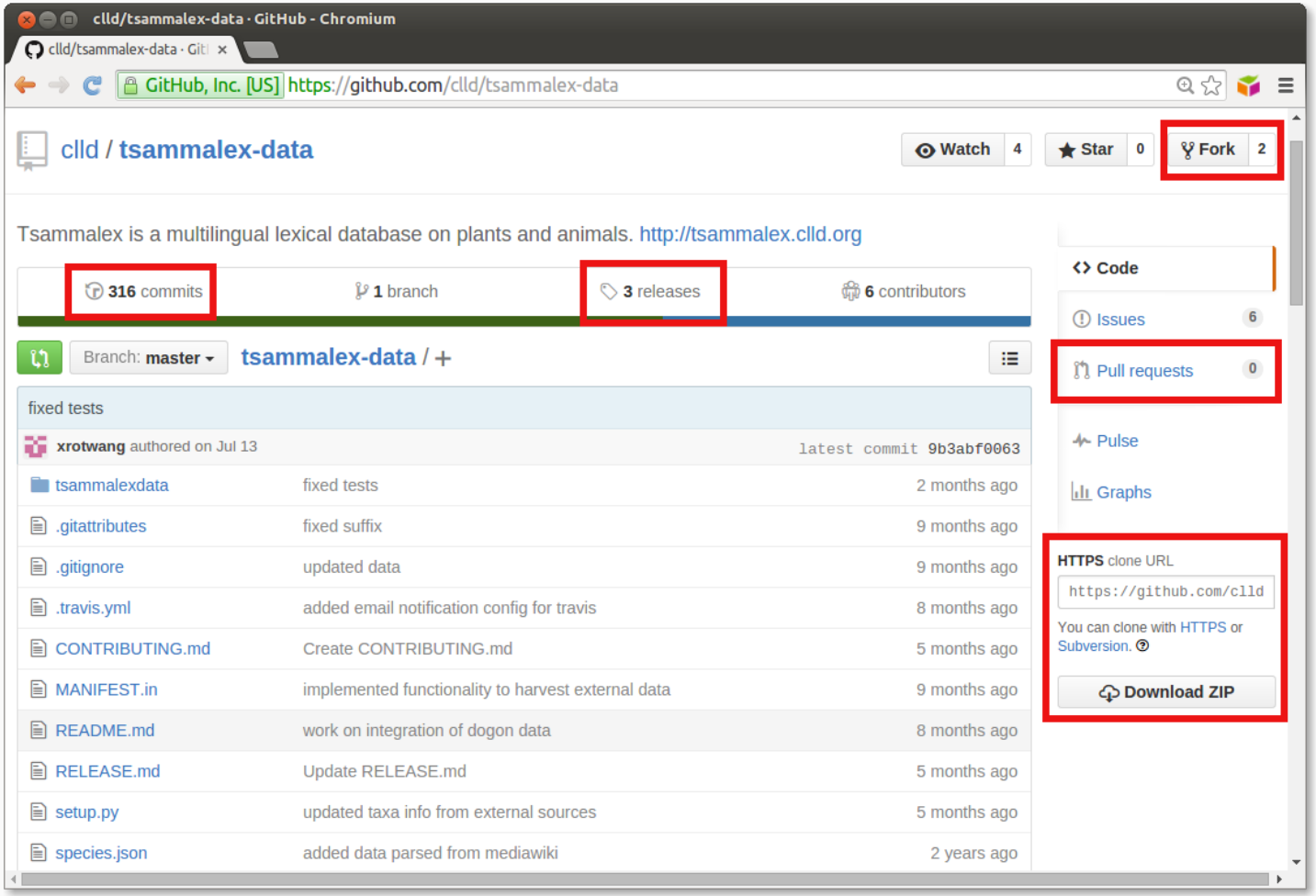



Figure: Tsammalex data repository

# Data is code - Updates

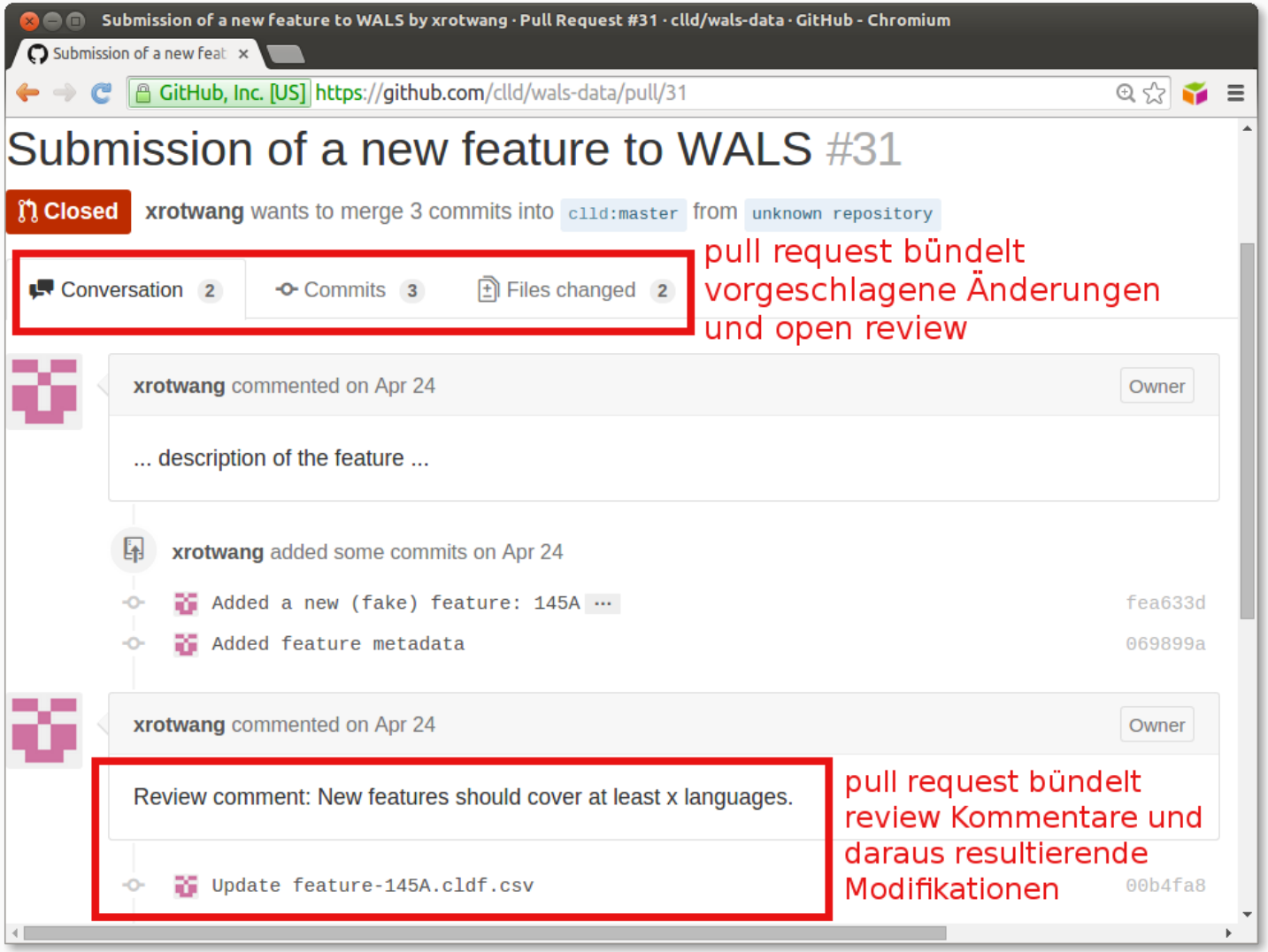



**Figure:** Pull request: Transparentes update mit review

# DATENBANK ON-DEMAND

# Data **and** code

Oft ist eine Datenbank mehr als nur ein dump der Daten: Die Applikation

- implementiert die Standardinterpretation der Daten und
- vermittelt Standardzugang zu den Daten (die API)

Wie können wir Zugänglichkeit der API von früheren Bearbeitungsständen der Datenbank ermöglichen?

# Data **and** code

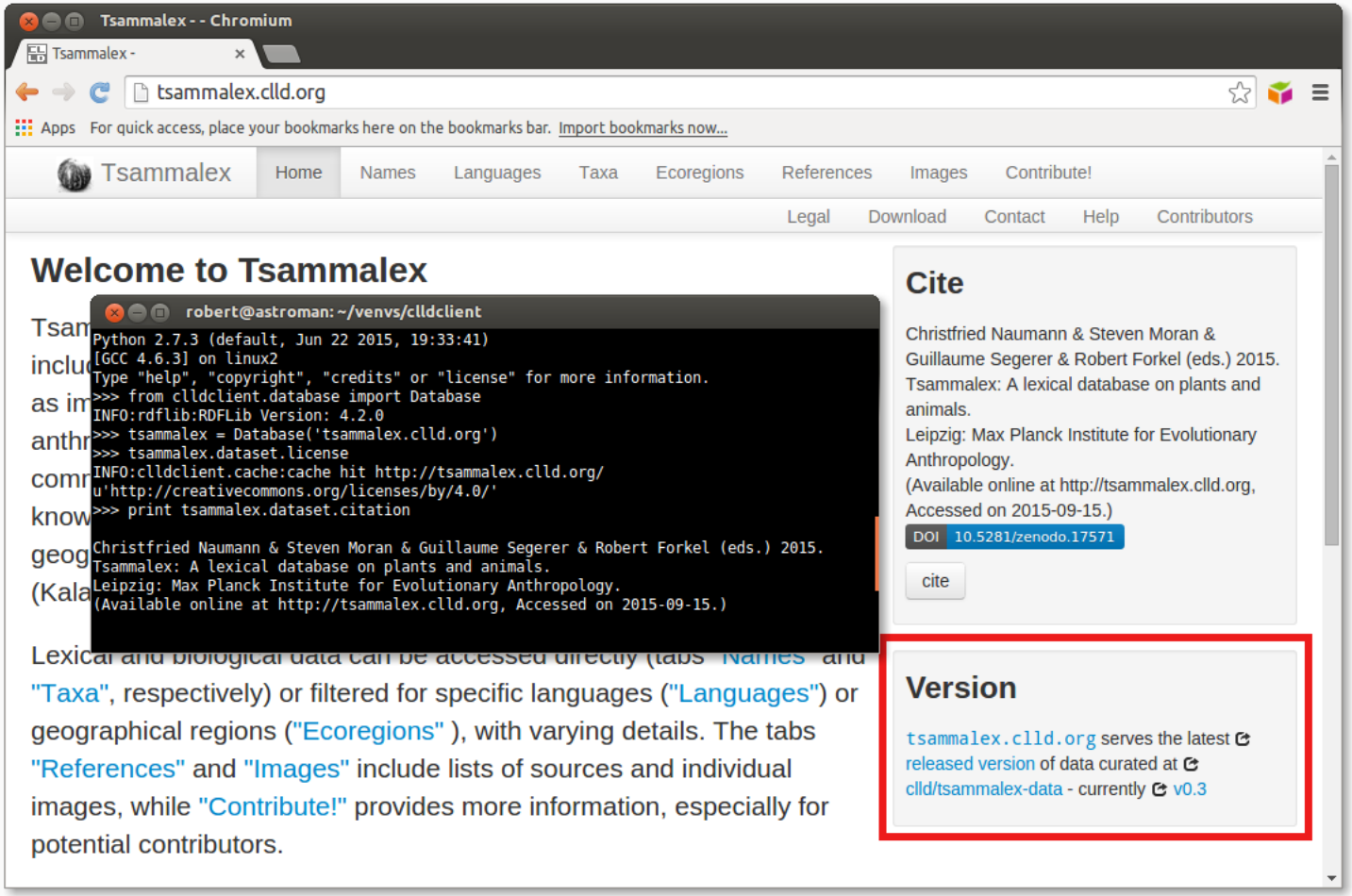
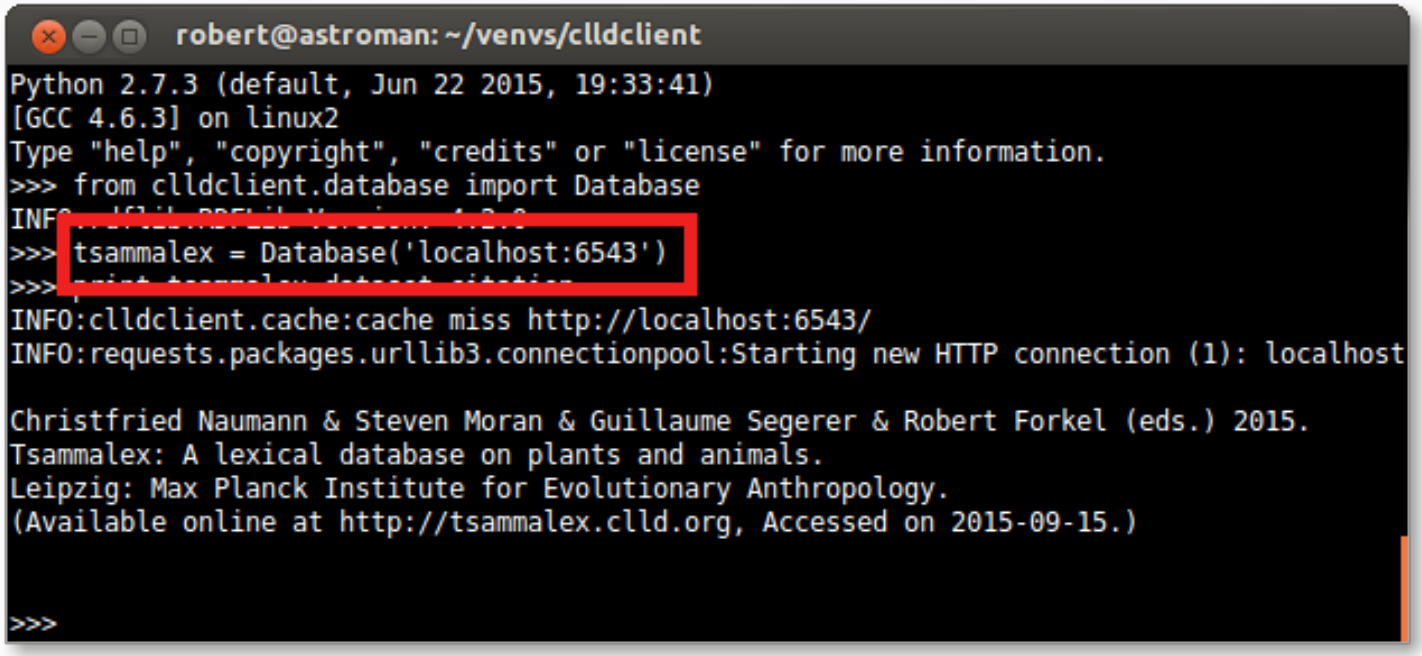



**Figure:** Applikation stellt API für Daten zur Verfügung

# Data **and** code

- Daten müssen zusammen mit code publiziert und archiviert werden.
- Code muss “bootstrapping” unterstützen, d.h. die Initialisierung einer lokalen Datenbank und Applikationsinstanz.




$\Longrightarrow$ Datenbank on-demand!

# Datenbank on-demand


Figure: DOI -> DB

# Datenbank on-demand

Der Teufel steckt im Detail (und externe Abhängigkeiten überall):

- Python-Pakete von PyPI
- Javascript vom CDN
- Base-layer für maps

Dennoch

- moderne Container- oder Virtualisierungslösungen (docker, ec2) ...
- mit standardisierten Applikationstemplates ...
- angeboten von Rechenzentren

könnte DOI -> DB als one-click Angebot für Wissenschaftler möglich machen.

# ZUSAMMENFASSUNG

# Zusammenfassung

## Für Datenerzeuger

Für code-ähnliche Daten lohnt ein Blick über den Zaun zur Software-Entwicklung – entsprechend John Nerbonne's Digital Humanities Motto:

> *Beg, buy, steal or borrow!* – https://twitter.com/TomKnieper/status/448820937446932480

## Für Humanities Data Center

"DOI->DB" könnte eine Antwort auf die "Mehr als wegspeichern?" Frage sein.

clld.org