Longitudinal Analysis of Collaboration Graphs of Forked Open Source Software Development Projects

Amirhosein (Emerson) Azarbakht
School of Electrical Engineering and Computer Science
Oregon State University
azarbaka@oregonstate.edu

Work Under Supervision of Prof. Carlos Jensen
Ph.D. Committee Members:
Prof. Alex Groce
Prof. Chris Scaffidi
Prof. Drew Gerkey
Prof. A. Morrie Craig

What is this about?

- Developing complex software systems is complex
- Software Developers (SD) interact
- Developers' interactions can be modeled as graphs
- Graphs change through time
- Can these changes reflect the climate happening during run-up to fork

What is forking?

- Forking:
  “when a part of a development community (or a third party not related to the project) starts a completely independent line of development based on the source code basis of the project.”

- Sometimes Undesirable:
  - Dilution of workforce
  - Flaming
  - Redundant work

Why Projects Fork?

Reason for forking	**Example forks**
Technical (Addition of functionality)	Amarok & Clementine Player
More community-driven development	Asterisk & Callweaver
Differences among developer team	Kamailio & OpenSIPS
Discontinuation of the original project	Apache web server
Commercial strategy forks	LibreOffice & OpenOffice.org
Experimental	GCC & EGCS
Legal issues	X.Org & XFree

Related Work

- Identifying knowledge brokers
- Sustainability
- Post-forking porting of new features or bug fixes
- Visual exploration of collaboration networks
- Social structures and dynamics
- Communication patterns
- Identifying “key” events
- Tension between diversity and homogeneity
- Age of users and survival rate patterns

- Gap: **Run-up to forks** seldom studied

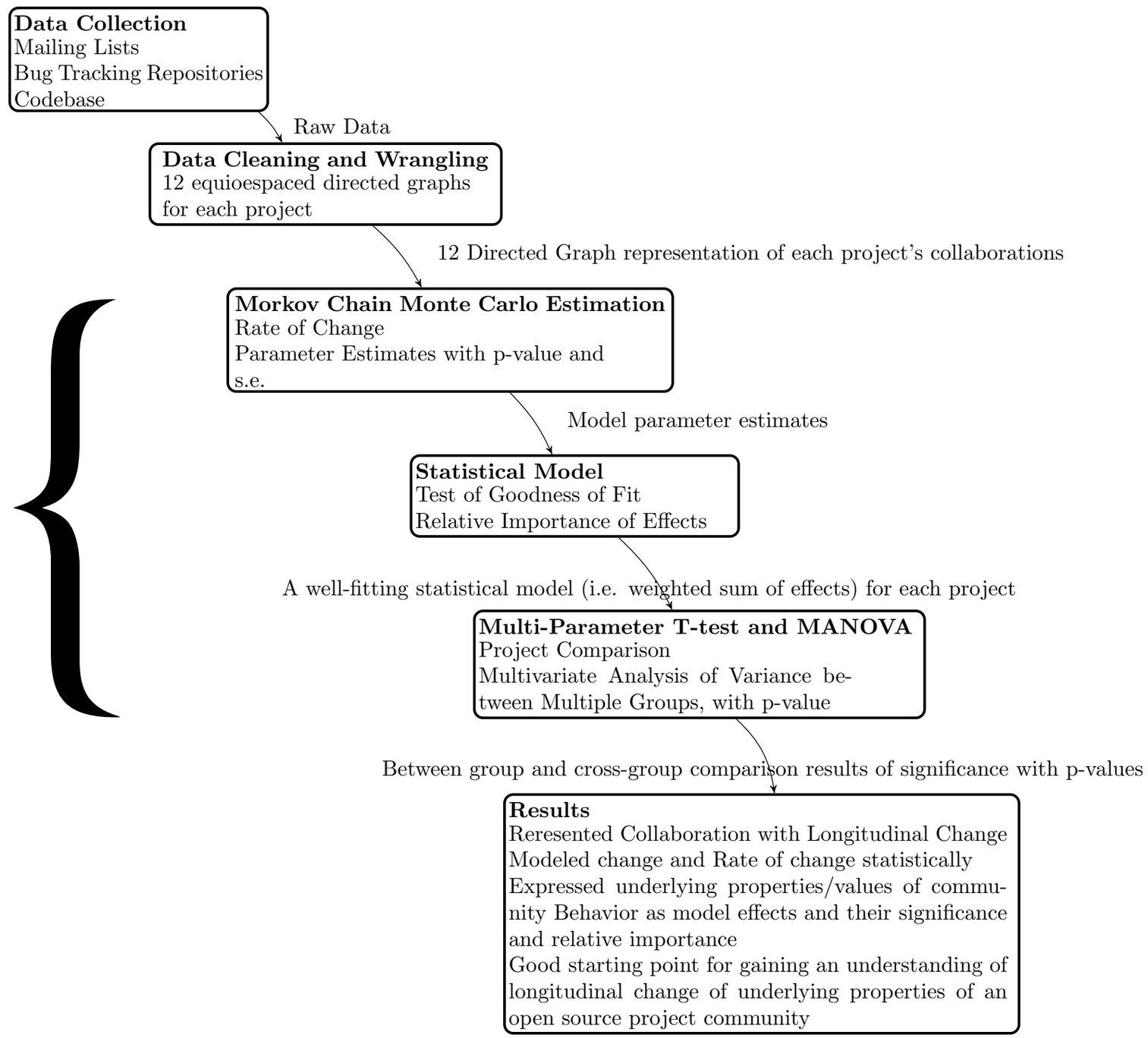
Research Goals

- Do forks leave traces?
- Do different forks leave different traces?
- Key indicators?
    - Validate

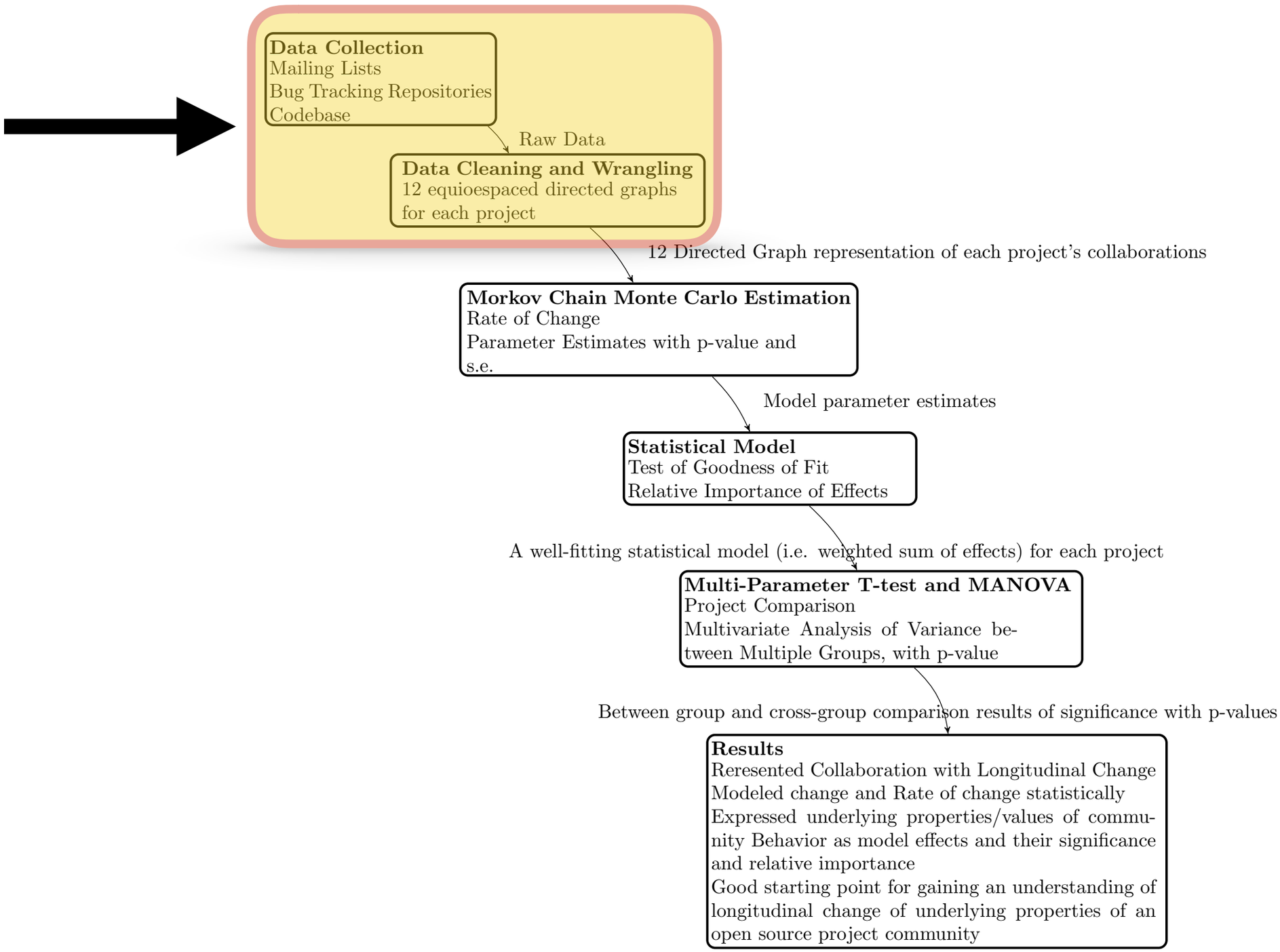
Methodology

Data Collection

Statistical Modeling



Methodology: Data Collection



Data Collection Categories

Table 6: The three types of projects for which data is collected in this study

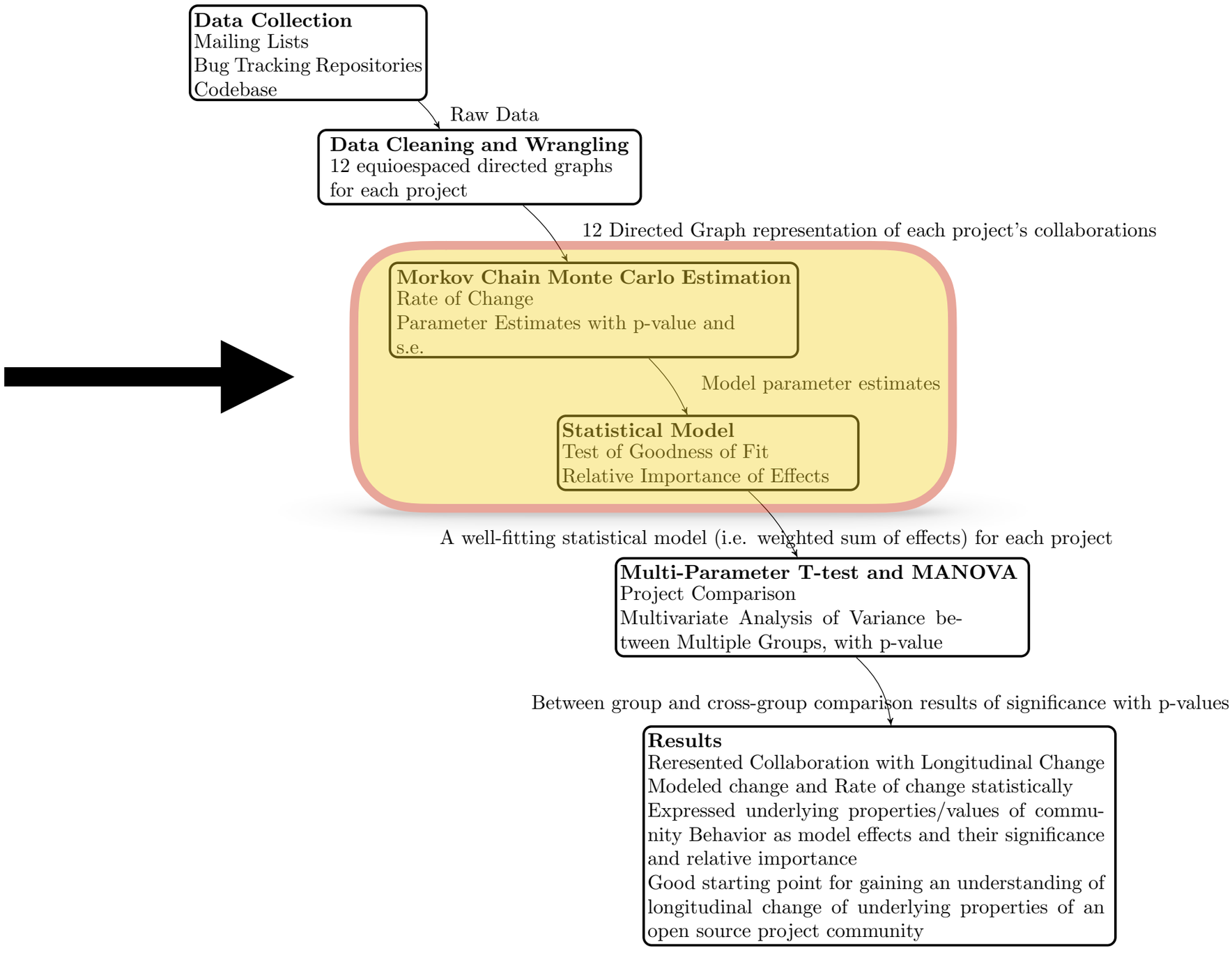
Type of forking	**Abbreviation**
Undesirable & socially-related forking	U.F.
Other socially-related forking	H.F.
No forking at all (as the control group)	No.F.

Data Collection Projects

Table 7: List of projects in U.F. H.F., and No.F. categories selected based on the criteria described in section 5.1 for our study

Projects	**Reason for forking**	**Year forked**	**Type**
Kamailio & OpenSIPS	Differences among developer team	2008	U.F.
ffmpeg & libav	Differences among developer team	2011	U.F.
Asterisk & Callweaver	More community-driven development	2007	U.F.
rdesktop & FreeRDP	More community-driven development	2010	U.F.
freeglut & OpenGLUT	More community-driven development	2004	U.F.
Amarok & Clementine Player	Technical (Addition of functionality)	2010	H.F.
Apache CouchDB & BigCouch	Technical (Addition of functionality)	2010	H.F.
Pidgin & Carrier	Technical (Addition of functionality)	2008	H.F.
MPlayer & MPlayerXP	Technical (Addition of functionality)	2005	H.F.
Ceph	Not forked	-	No.F.
Python	Not forked	-	No.F.
OpenStack Neutron	Not forked	-	No.F.
GlusterFS	Not forked	-	No.F.

Methodology: Statistical Modeling



Statistical Model

- Longitudinal
- Developer-oriented
  - Likelihood of forming ties, maintaining ties, …
- Stochastic
  - Optimize an objective function
  - Assume observed graphs as outcomes of a continuous-time Markov process
- Estimated

Statistical Model Observed Networks

Let the data for our statistical developer-oriented model be M repeated observations on a network with g developers. The M observed networks (at least two) are represented as directed graphs with adjacency matrices $X(t_m) = (X_{ij}(t_m))$ for $m = 1, ..., M$, where i and j range from a to g. The variable X_{ij} shows whether at time t there exists a tie from i to j (value 1) or not (value 0). Be definition, $\forall i, X_{ii} = 0$ (i.e. the diagonal of the adjacency matrices).

Statistical Model Longitudinal Change

In order to model the network evolution from $X(t_1)$ to $X(t_2)$, and so on, it is natural to treat the network dynamics as the result of a series of small atomic changes, and not bound to the observation moment, but rather as a more of less continuous process. In this way, the current network structure is a determinant of the likelihood of the changes that might happen next [13].

Statistical Model Mini-steps

For each change, the model focuses on the developer whose tie is changing. We assume that developer i has control over the set of outgoing tie variables $(X_{i1}, ..., X_{ig})$ (i.e. the i^{th} row of the adjacency matrix). The network changes one tie at a time. We call such an atomic change a *ministep*. The moment at which developer i changes one of his ties, and the kind of change that he makes, can depend on attributes represented by observed covariates, and the network structure. The moment is stochastically determined by the *rate function*, and the particular change to make, is determined by the *objective function* and the *gratification function*. We cannot calculate this complex model exactly. Rather than calculating exactly, we estimate it using a Monte Carlo Markov Chain method. The estimated model is used to test hypotheses about the forked FOSS communities.

Statistical Model
Rate Function

The **rate function** $\lambda_i(x)$ for developer *i* is the rate at which developer *i*'s outgoing connections changes occur.

$$\lambda_i(x) = \lim_{dt \to 0} \frac{1}{dt} \mathrm{P}\big(X_{ij}(t+dt) \neq X_{ij}(t) \quad for \quad some \quad j \in \{i, ..., g\} | X(t) = x)\big)$$

It models how frequently the developers make mini-steps.

Statistical Model
Objective function

The **objective function** $f_i(s)$ for developer i is the value attached to the network configuration x.

$$f_i(\beta, x) = \sum_{k=1}^{L} \beta_k s_{ik}(x)$$

- Functions $s_{ik}(x)$ are the effects we define. Parameters $\beta = (\beta_1, \ldots, \beta_L)$ is to be estimated.

- Idea: Given the opportunity to make a change in his out-going tie variables $(X_{i1},\ldots,X_{ig})$, developer i selects the change that gives the greatest increase in the objective function.

Statistical Model Effects

- Reciprocity
- Closure
    - Transitive triplets
- Three-cycles
- in-in degree assortativity
- in-out degree assortativity
- Developer's level of contribution/activity (e.g. code commits per month, or mailing list posts per month)
- Developer's level of privilege/prestige (e.g. ad- min privilege-holder vs. core contributor vs. marginal devel- oper/user)
- Developer's level of negative experience (e.g. num- ber of rejected pull requests: especially in forks for more community- driven development)
- Developer's age (either birth age, (young 20-year- old developer vs. older 50-year-old developer) or their seniority as a development community member (i.e. how long they have been in the community)
- Developer's propensity for short-responses vs. long- responses. (under-communicator vs. over-communicator)
- Developer's communication sentiment (generally bitter in communication known as "a jerk", or has a positive communication prose)
- Developer's engagement style (i.e. dive-bomber- style contributor who jumps in, overcommits and leave, vs. a steady and increasingly engaged contributor who starts off slow and grows his/her commitment gradually)

Statistical Model Simulation and Estimation

- Using Method of Moments
- Markov Chain Monte Carlo (MCMC)

The method of moments (MoM) can be used in the following way [45]:

Let $x^{obs}(t_m), m = 1, ..., M$ be the observed networks, and the objective function be $f_i(\beta, x) = \sum_{k=1}^{L} \beta_k s_{ik}(x)$. In MoM, the goal is to determine the parameters β_k such that, summed over i and m, the expected values of the $s_{ik}(X(t_{m+1}))$ are equal to the observed values. In other words, MoM fits the *observed* to the *expected*. The observed target values are:

$$s_k^{obs} = \sum_{m=1}^{M-1} \sum_{i=1}^{g} s_{ik}(x^{obs}(t_{m+1})) \quad (k = 1, ..., L) \tag{9}$$

The simulations run as follows [45]:

1. The distance between two directed graphs x and y is defined as
$$\|x - y\| = \sum_{i,j} |x_{ij} - y_{ij}| \tag{10}$$
and let the observed distances for m = 1, ..., M-1 be
$$c_m = \|x^{obs}(t_{m+1}) - x^{obs}(t_m)\| \tag{11}$$
2. Use the β parameter vector and the rate of change $\lambda_i(x) = 1$
3. Do the following for m = 1, ..., M-1
   (a) Define the time as 0, and start with $X_m(0) = x^{obs}(t_m)$
   (b) Simulate the model, as described in 5.3.5 until the first time point, where
   $$\|X_m(R_m) - x^{obs}(t_m)\| = c_m \tag{12}$$
4. For k = 1, ..., L, calculate the following statistics:
$$S_k = \sum_{m=1}^{M-1} \sum_{i=1}^{g} s_{ik}(X_m(R_m)) \tag{13}$$

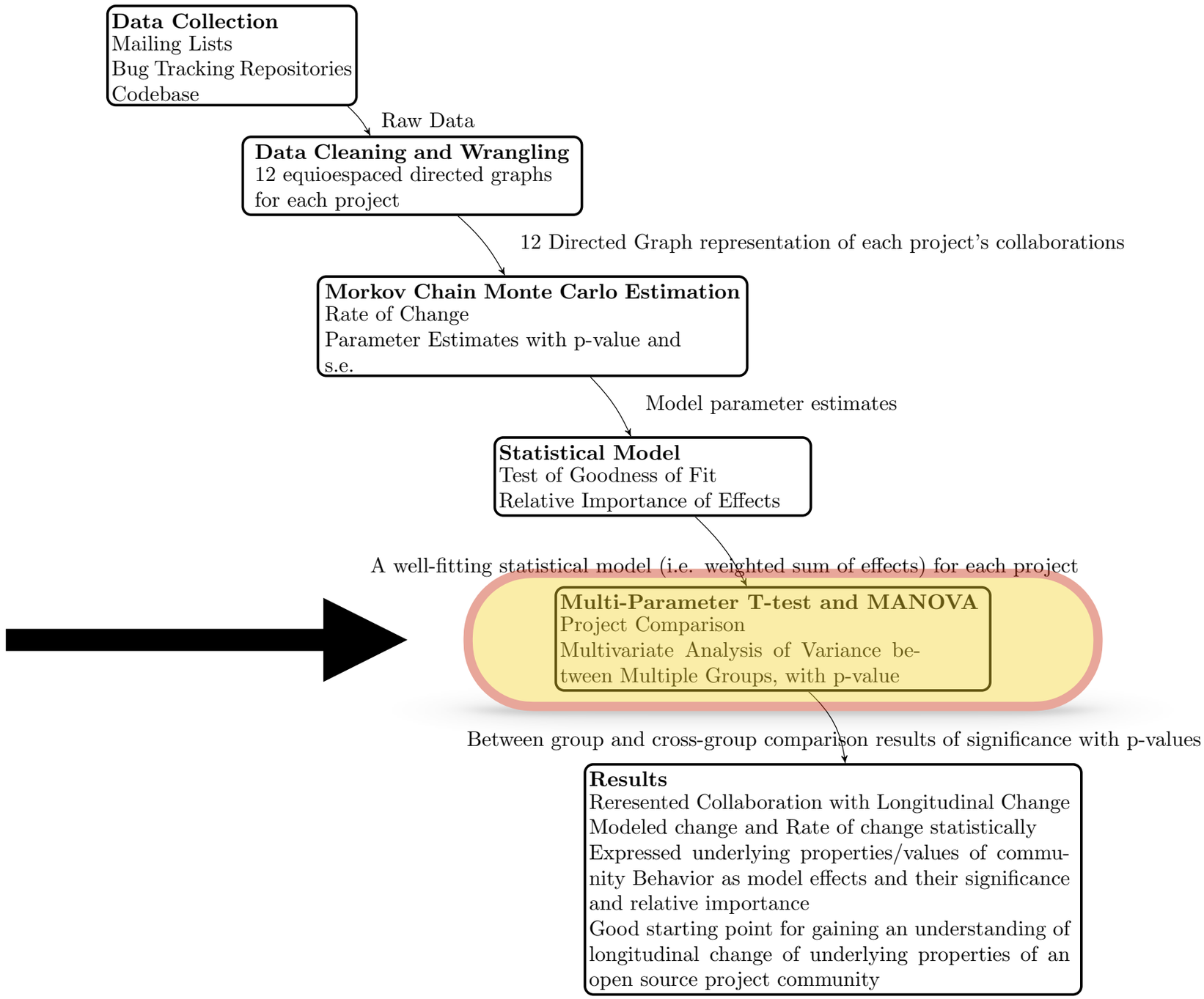
The simulation output will be the random variables $(S, R) = (S_1, ..., S_L, R_1, ..., R_{M-1})$ The desirable outcome for the estimation is the vector parameter $\hat{\beta}$ for which the expected and the observed vectors are the same.

Statistical Model

Table 8: Expectations for significance of the statistical model covariates based on intuition. Note that * indicates significantly-related; +/- indicates positively/negatively related.

Model parameter	**U.F. Pers. Dif.**	**U.F. More Comm.**	**H.F. Tech. Dif.**
Reciprocity	*		*
Transitive triplets	*		*
Number of developers at distance two		*-	
Three-cycles	*		*
out-out degree assortativity		*+	
Covariate-V1 [Developer's contribution level]-related popularity	*		*
Covariate-V2 [Developer's privilege level]-related dissimilarity		*-	*
Covariate-V3 [Developer's negative experience level]-related popularity		*	
Covariate-V3 [Developer's negative experience level]-related dissimilarity	*		
Covariate-V4 [Developer's age-seniority level]-related popularity	*		
Covariate-V4 [Developer's age-seniority level]-related dissimilarity	*	*	
Covariate-V5 and V6[Developer's under/over communicator & sentiment history]-related activity and dissimilarity	*		
Covariate-V7 [Developer's graduality level]-related activity	*	*-	
Covariate-V7 [Developer's graduality level]-related dissimilarity	*	*-	

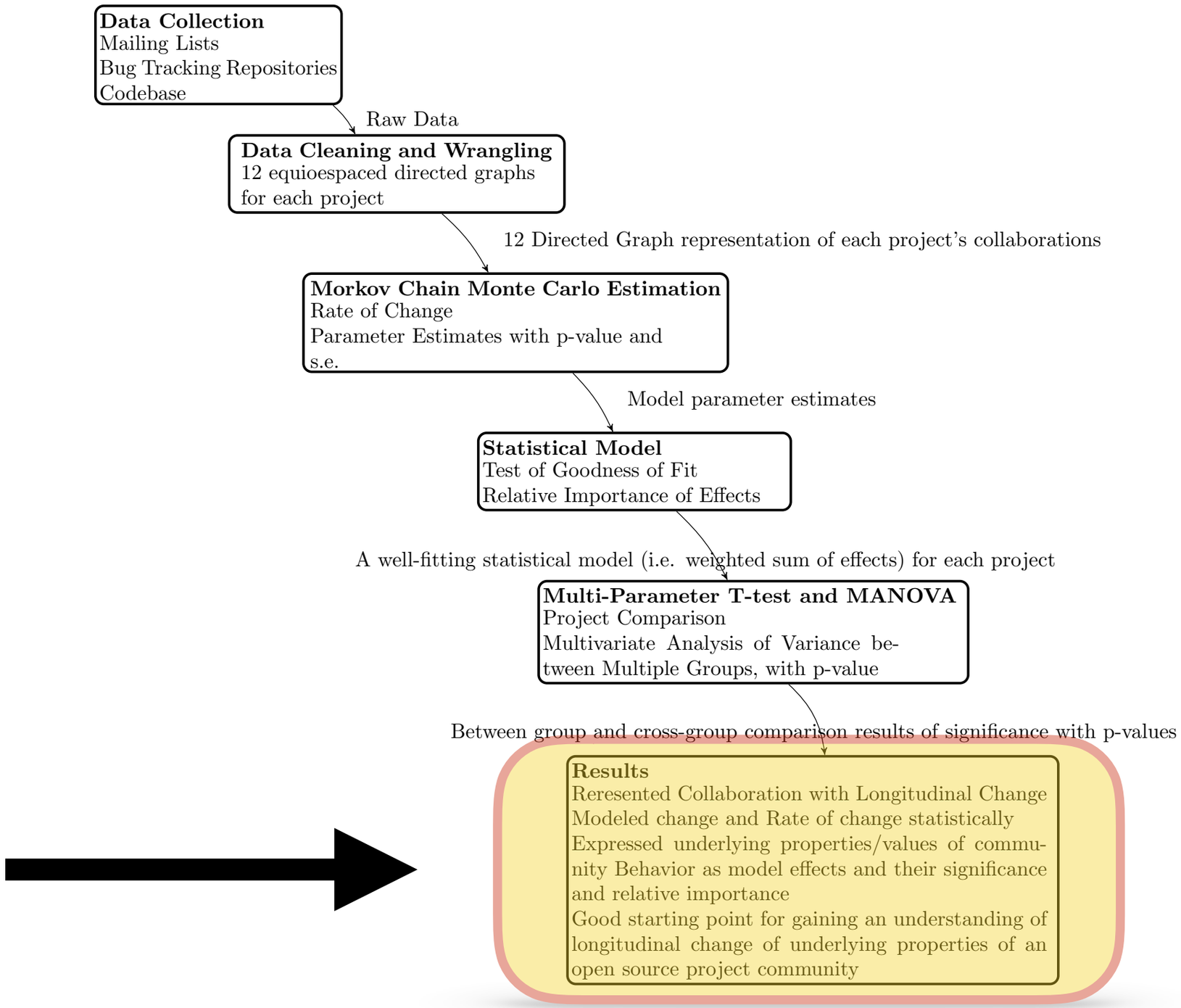
Methodology: Hypothesis Testing



Hypothesis Testing

- Single parameter tests
- Multi-parameter differences between groups
- Multivariate ANOVA

Methodology: Results



Methodology: Validation

- Qualitative
  - Interviews/surveys
    - Semi-structured interviews with contributors
  - Sentiment analysis
    - Contents of messages sent/received by top contributors

Conclusions

- Represented Collaboration with Longitudinal Change
- Modeled change and Rate of change statistically
- Expressed underlying properties/values of community behavior as model effects and their significance and relative importance
- Validated using qualitative methods
- Good starting point for gaining an understanding of longitudinal change of underlying properties of an open source project community

Timeline

TABLE 9 Timeline

Spring 2012	Literature review
Fall 2012	Literature review & data collection
Fall 2013	Data cleaning and wrangling
Winter 2014	Creating communication graphs
Spring 2014	Temporal visualization and temporal SNA
Fall 2014	Preliminary statistical analysis
Spring/Fall 2015	Planning and preliminary examination
Winter 2016	Prelim
Winter 2016	Data collection for issue tracking and source code
Spring/Summer 2016	Longitudinal modeling & qualitative study
Summer 2016	Thesis writing
September 2016	Defense (Hopefully)

Thank you!

Questions? Suggestions? Help?