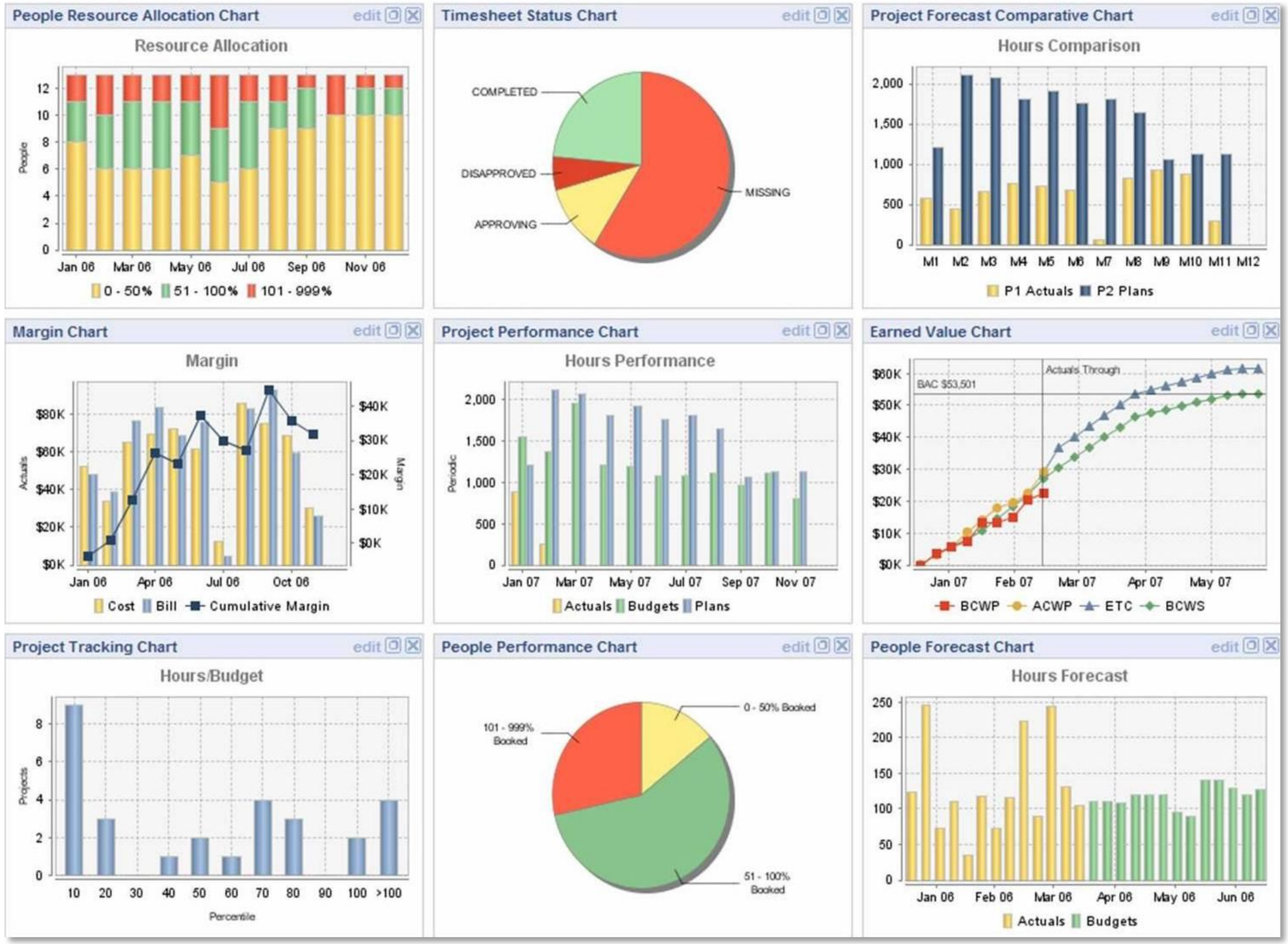
Validating the Use of Topic Models for Software Evolution

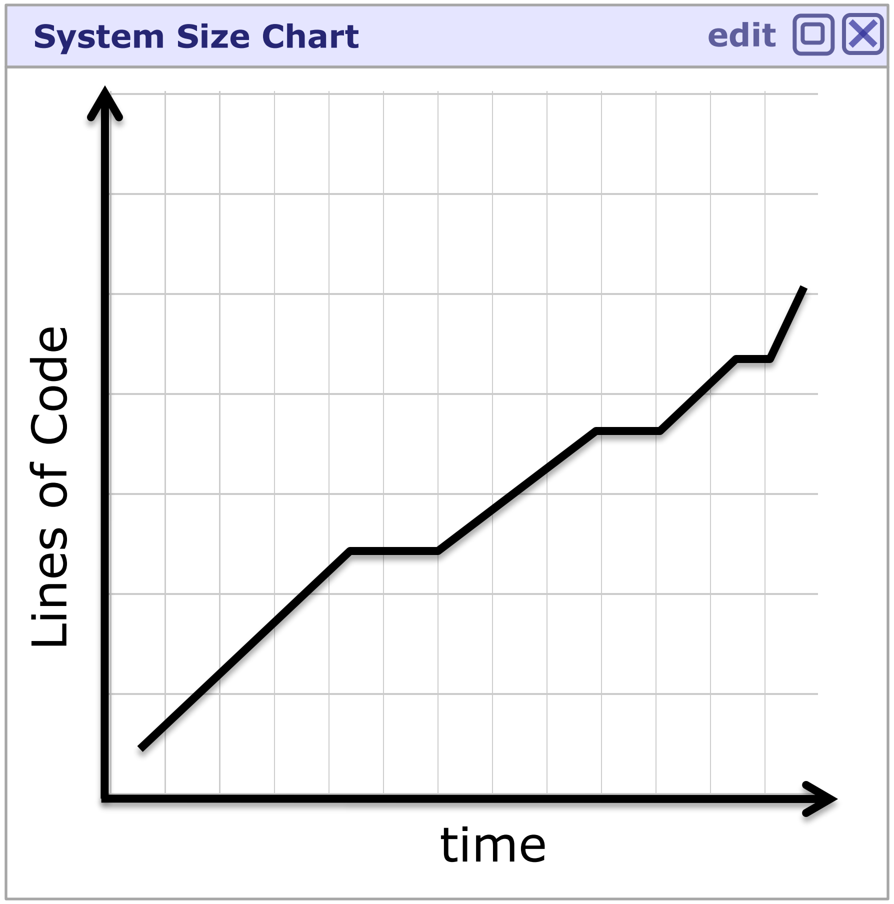
Stephen W. Thomas, Bram Adams,
Ahmed E. Hassan, and Dorothea Blostein

Software Analysis and Intelligence Lab (SAIL)
School of Computing, Queen's University, Canada

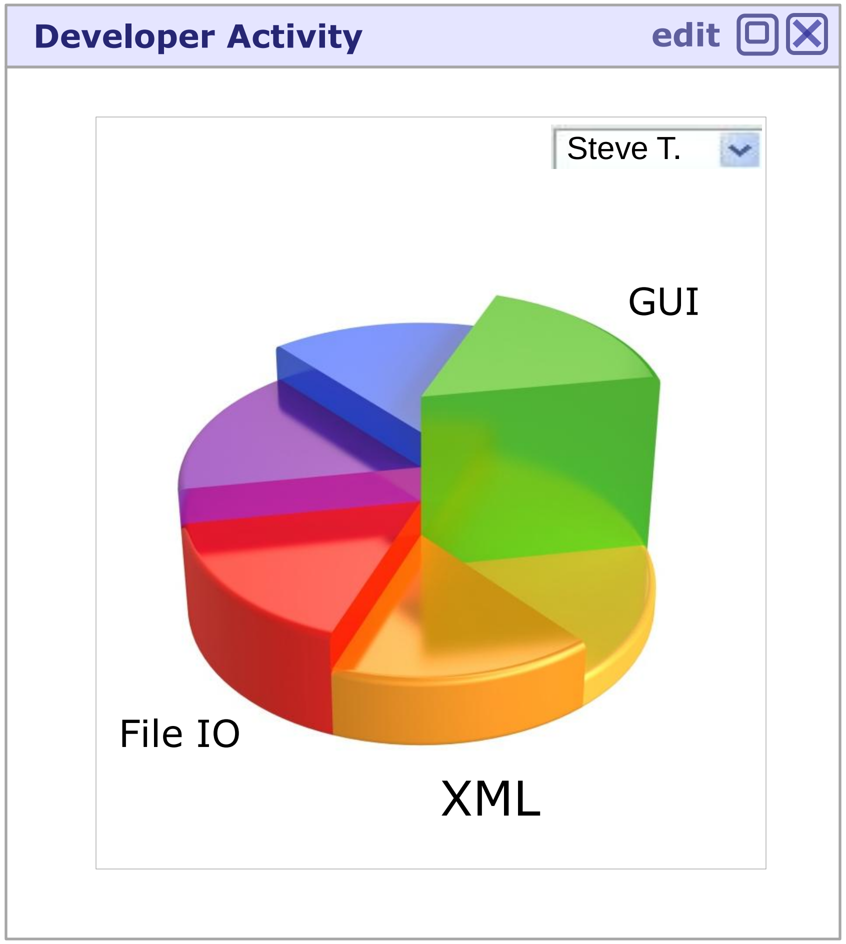


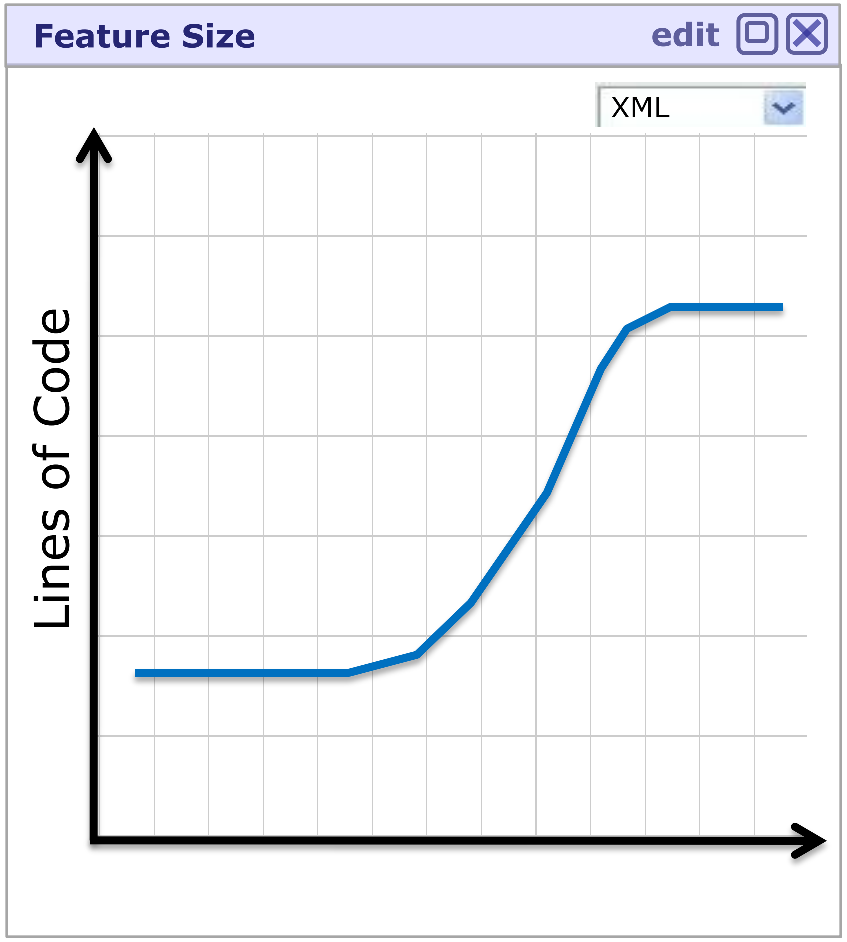
People Resource Allocation Chart
Resource Allocation
People
Jan 06 Mar 06 May 06 Jul 06 Sep 06 Nov 06
0 - 50% 51 - 100% 101 - 999%
Timesheet Status Chart
COMPLETED
DISAPPROVED
APPROVING
MISSING
Project Forecast Comparative Chart
Hours Comparison
M1 M2 M3 M4 M5 M6 M7 M8 M9 M10 M11 M12
P1 Actuals P2 Plans
Margin Chart
Margin
Actuals
Margin
Jan 06 Apr 06 Jul 06 Oct 06
Cost Bill Cumulative Margin
Project Performance Chart
Hours Performance
Periodic
Jan 07 Mar 07 May 07 Jul 07 Sep 07 Nov 07
Actuals Budgets Plans
Earned Value Chart
BAC $53,501
Actuals Through
Jan 07 Feb 07 Mar 07 Apr 07 May 07
BCWP ACWP ETC BCWS
Project Tracking Chart
Hours/Budget
Projects
Percentile
People Performance Chart
0 - 50% Booked
101 - 999% Booked
51 - 100% Booked
People Forecast Chart
Hours Forecast
Jan 06 Feb 06 Mar 06 Apr 06 May 06 Jun 06
Actuals Budgets
edit

Current State of Practice

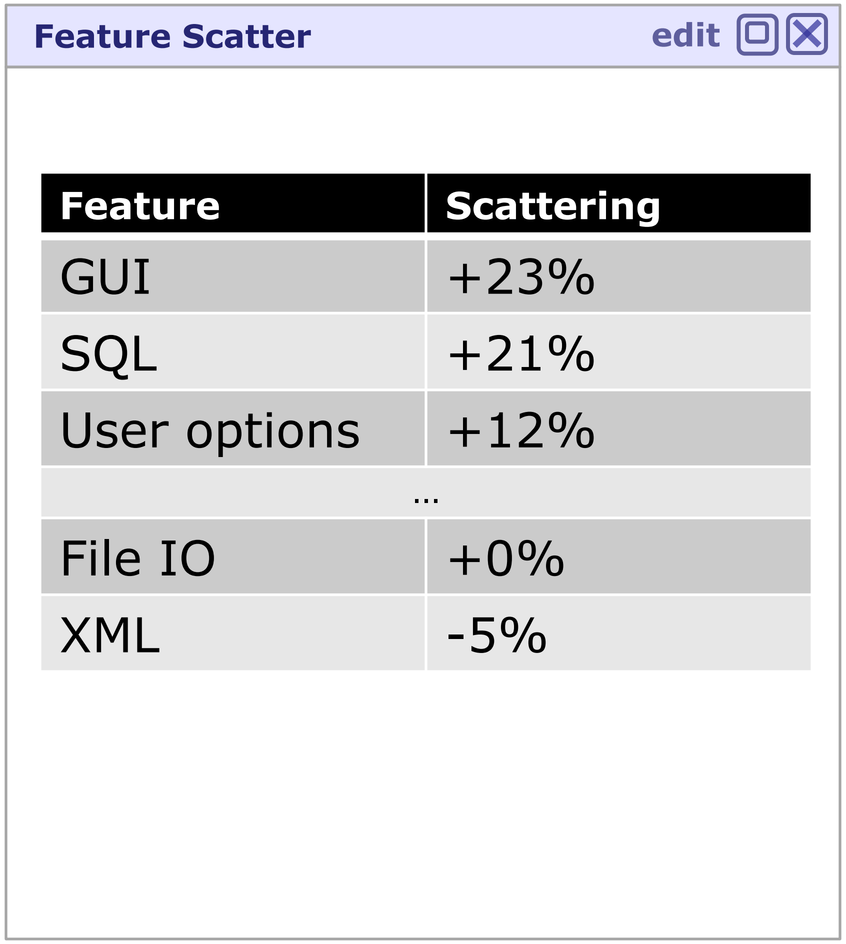


Our Vision

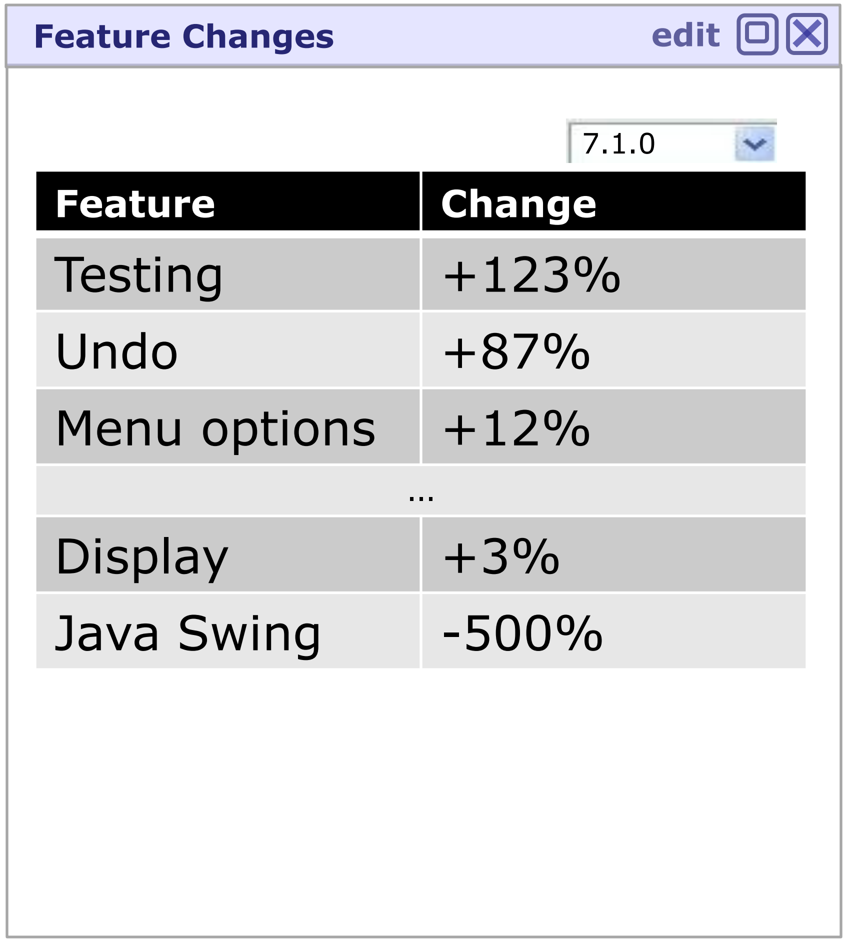

Developer Activity
edit
Steve T.
GUI
File IO
XML


Feature Size
edit
XML
Lines of Code

Our Vision



Feature	Scattering
GUI	+23%
SQL	+21%
User options	+12%
…	
File IO	+0%
XML	-5%



Feature	Change
Testing	+123%
Undo	+87%
Menu options	+12%
…	
Display	+3%
Java Swing	-500%

Detected by →

→ *Used in*

Software Changes
(bug fixes, refactorings, new features)

Project Dashboards

How to automatically detect software evolution?

Hypothesis:

- ***Topic evolution models***, borrowed from the text-mining domain, can automatically detect software evolution

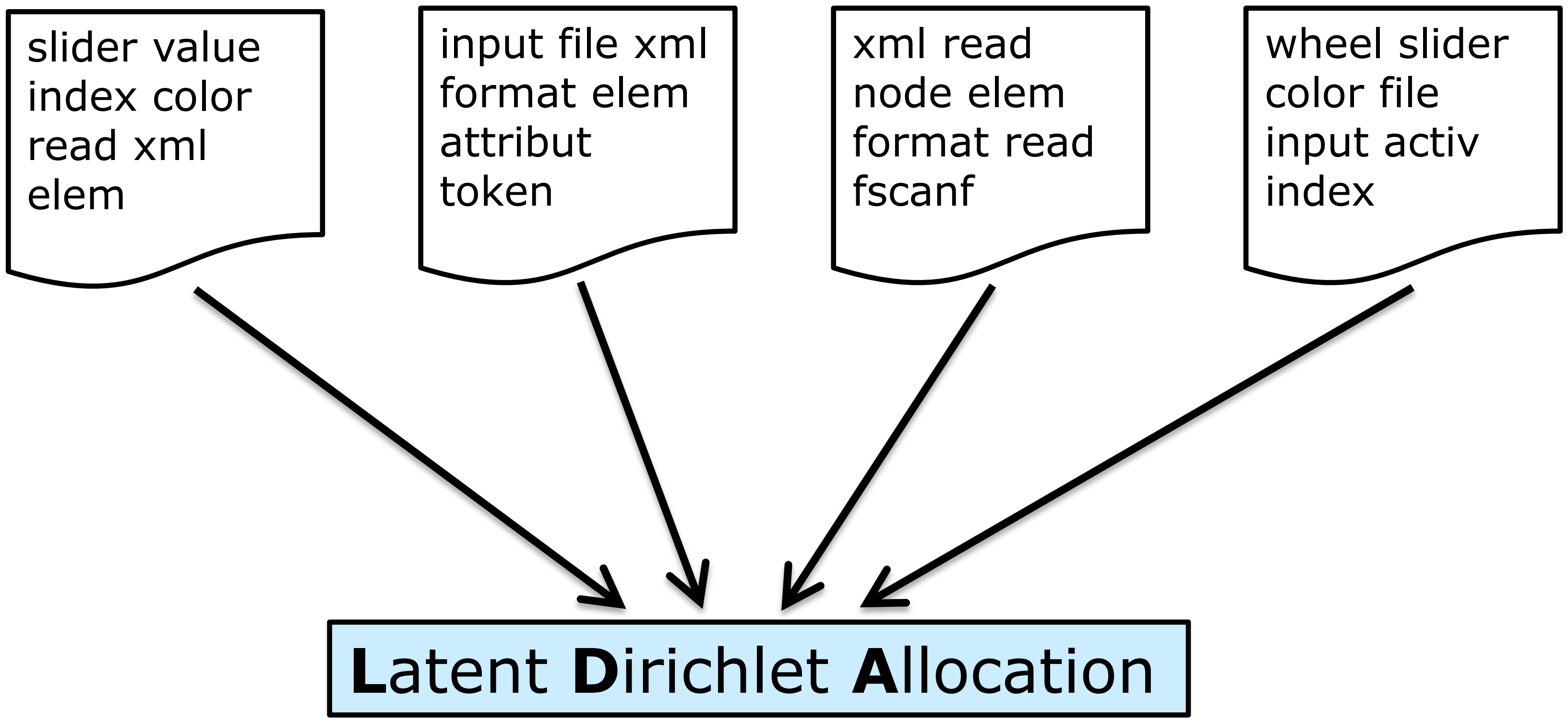
This paper:

- Initial qualitative study to validate hypothesis

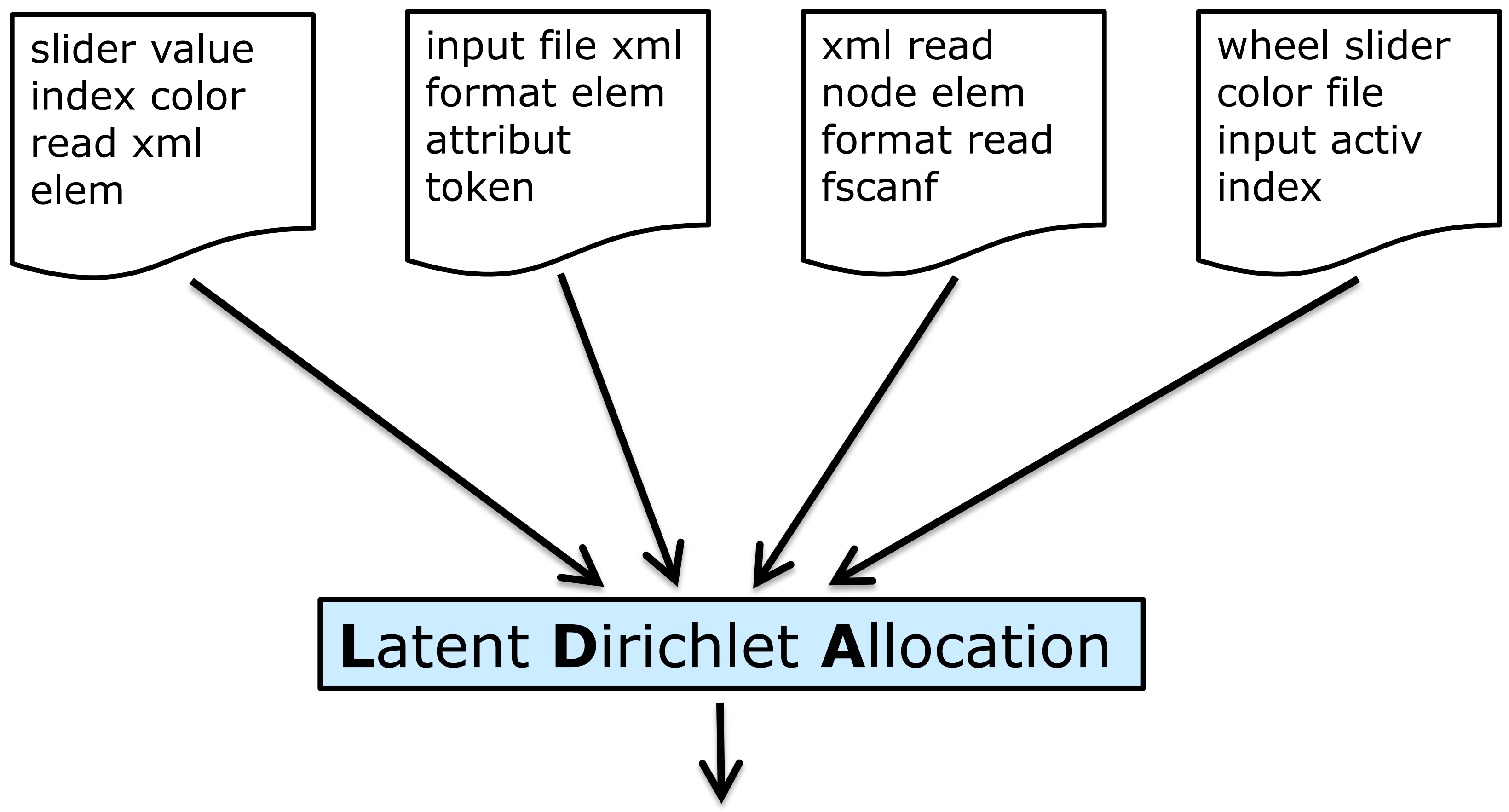
Topic Models

Latent **D**irichlet **A**llocation

Topic Models



Topic Models

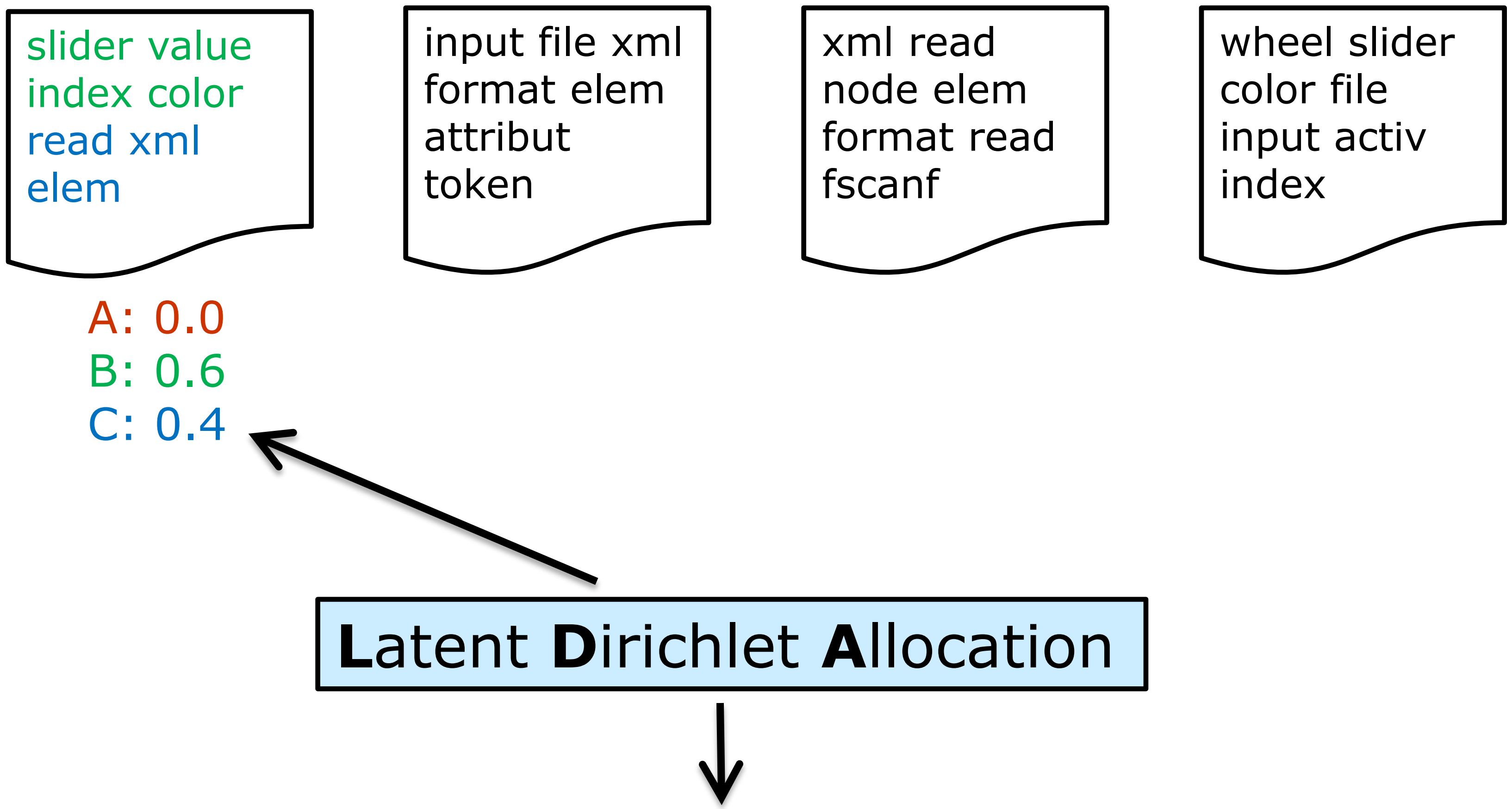

slider value index color read xml elem
input file xml format elem attribut token
xml read node elem format read fscanf
wheel slider color file input activ index
Latent **D**irichlet **A**llocation

TOPIC A: *{file input read fscanf format …}*

TOPIC B: *{color index slider wheel …}*

TOPIC C: *{elem attribut read token xml …}*

Topic Models

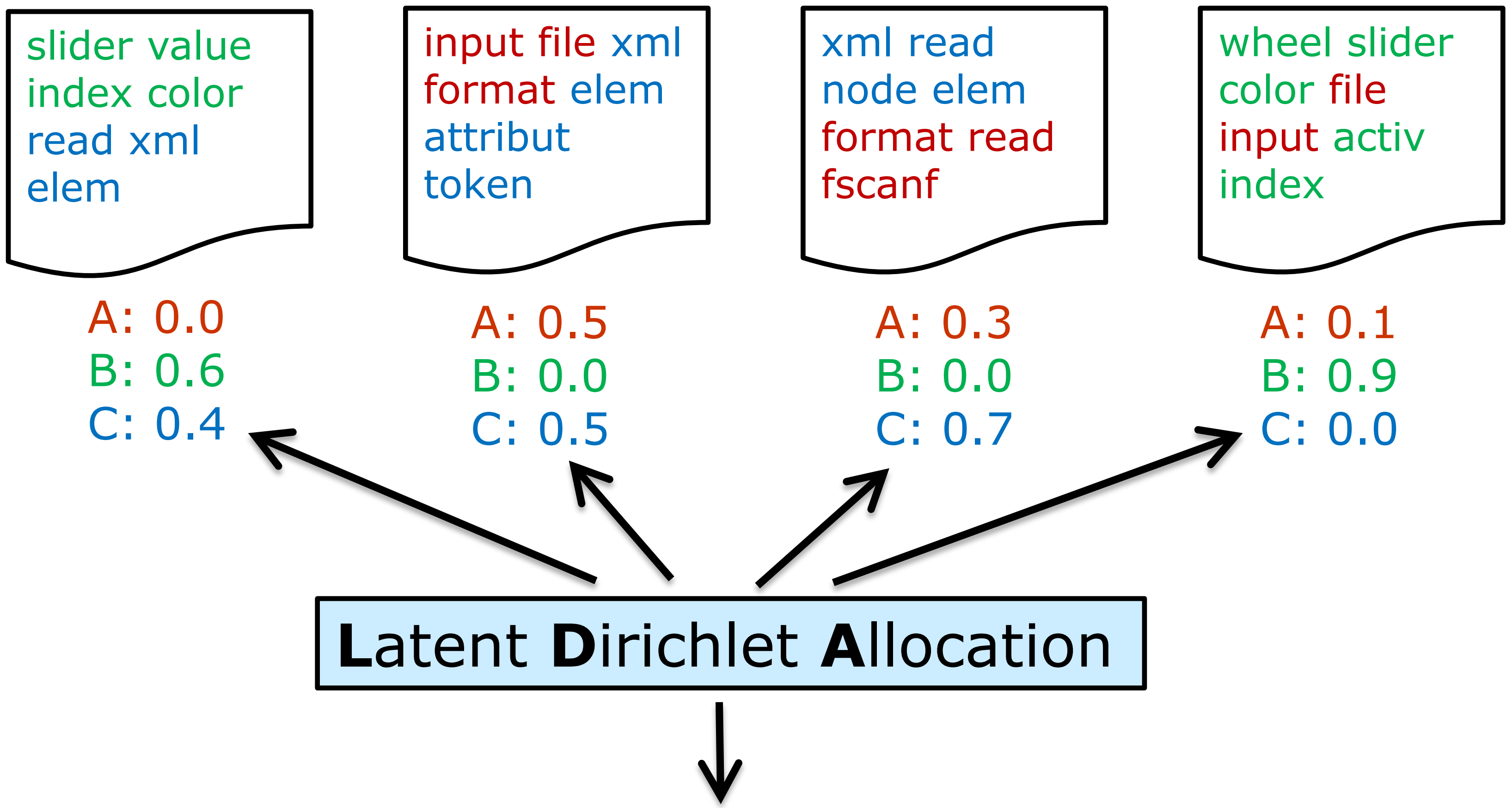


TOPIC A: *{file input read fscanf format …}*

TOPIC B: *{color index slider wheel …}*

TOPIC C: *{elem attribut read token xml …}*

Topic Models

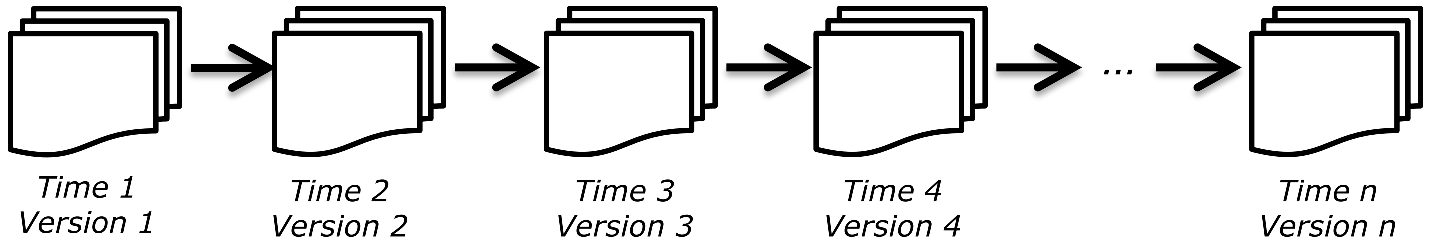

slider value index color read xml elem
A: 0.0
B: 0.6
C: 0.4
input file xml format elem attribut token
A: 0.5
B: 0.0
C: 0.5
xml read node elem format read fscanf
A: 0.3
B: 0.0
C: 0.7
wheel slider color file input activ index
A: 0.1
B: 0.9
C: 0.0
Latent **D**irichlet **A**llocation

Topic A: *{file input read fscanf format …}*

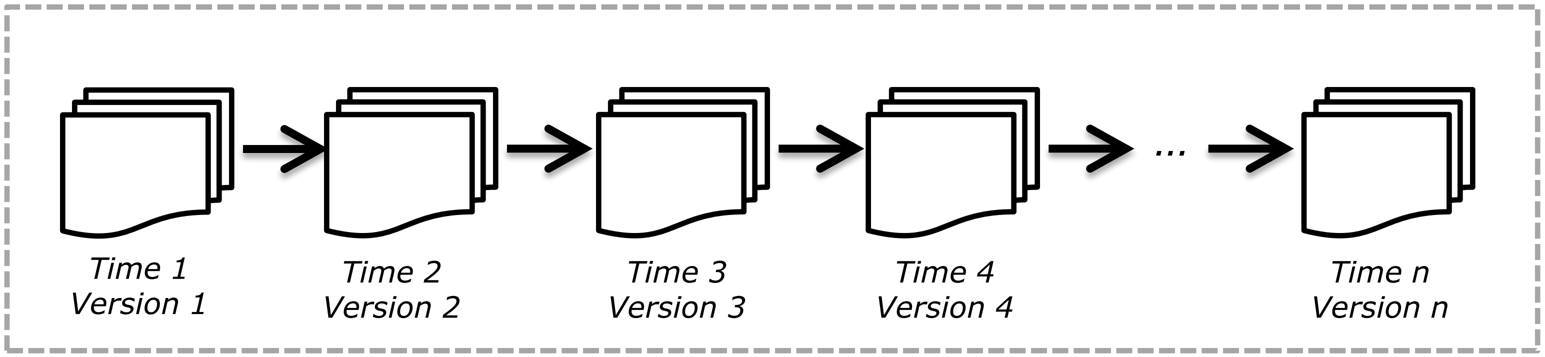
Topic B: *{color index slider wheel …}*

Topic C: *{elem attribut read token xml …}*

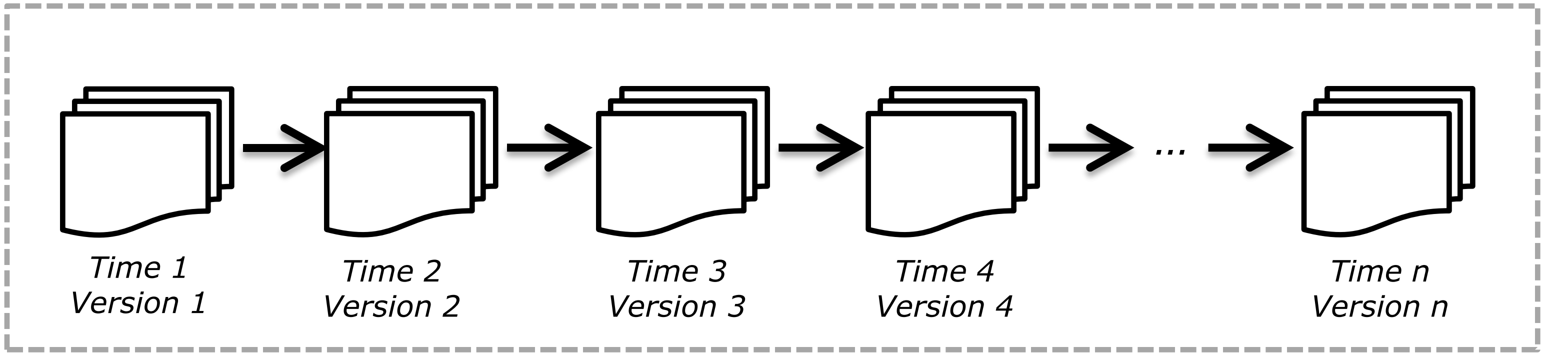
Topic Evolution Models

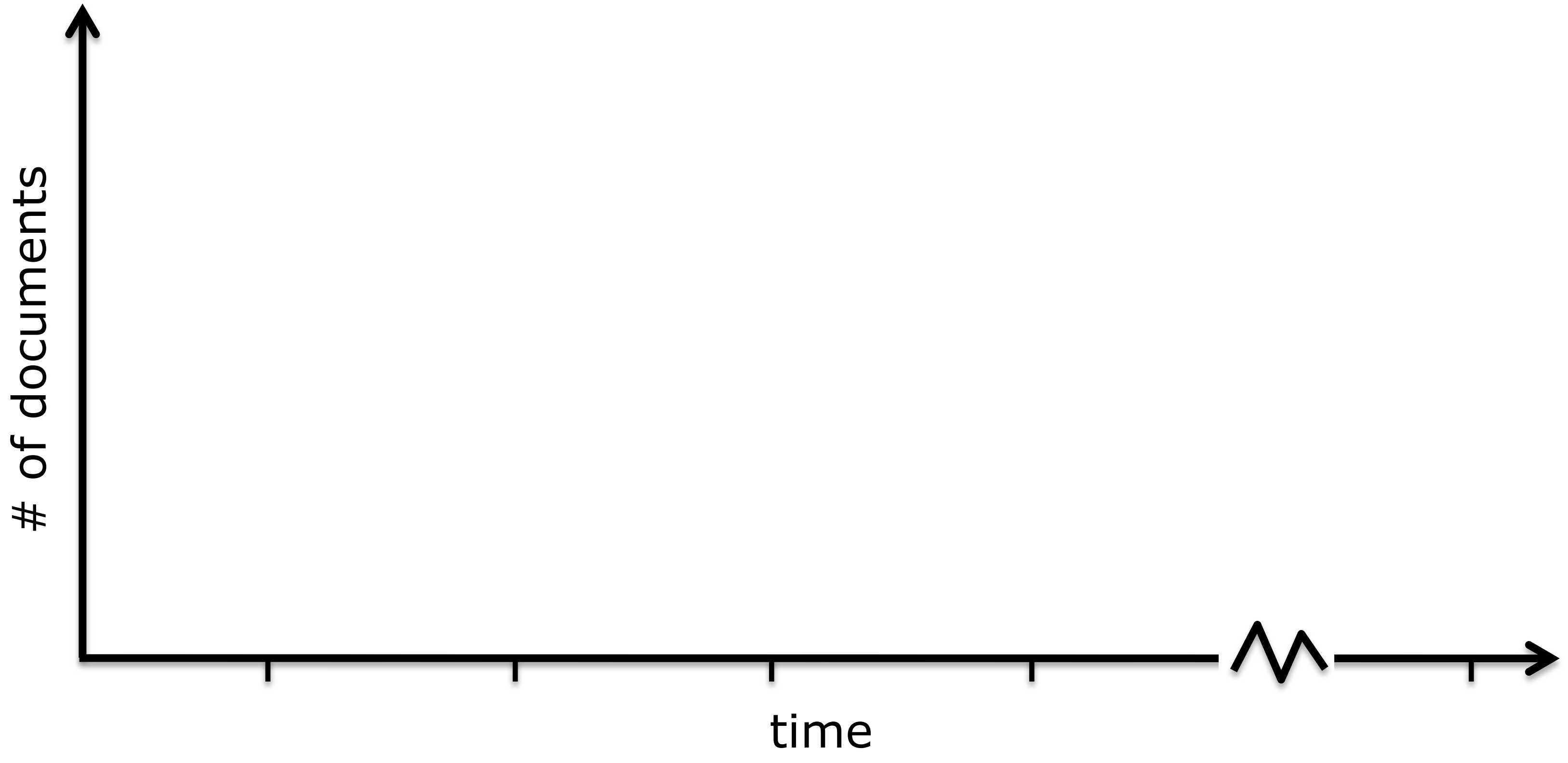


Topic Evolution Models

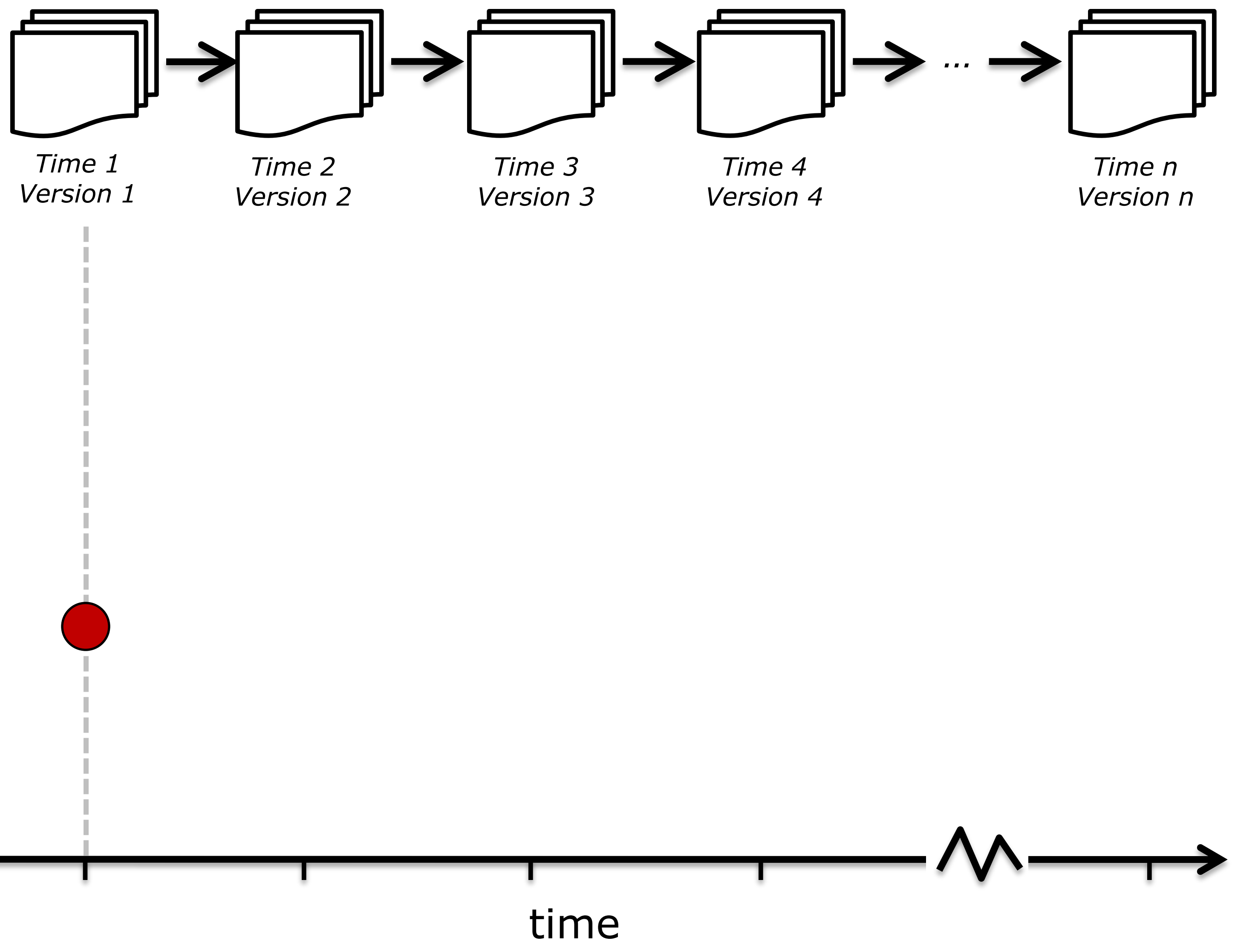


Topic Evolution Models

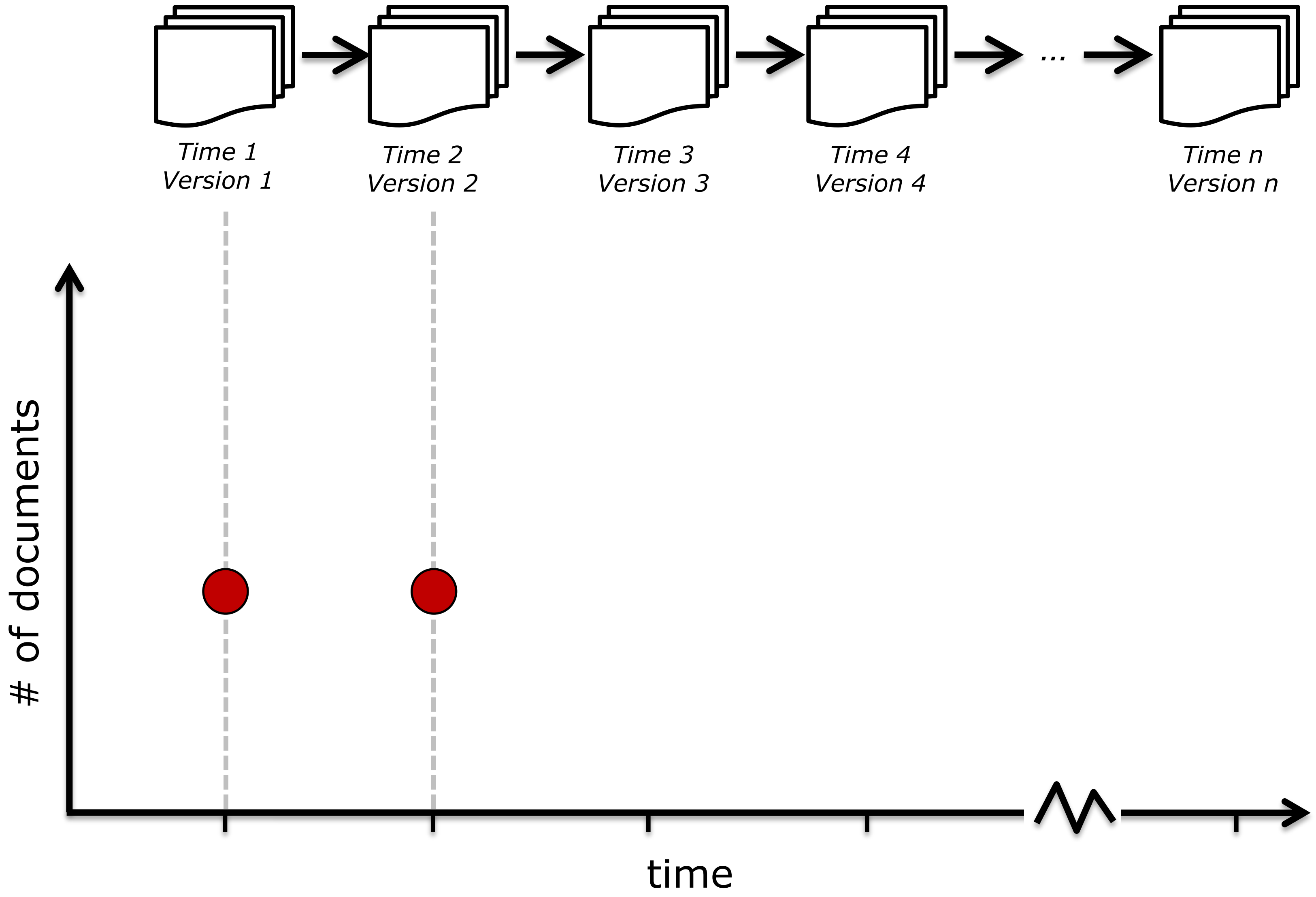




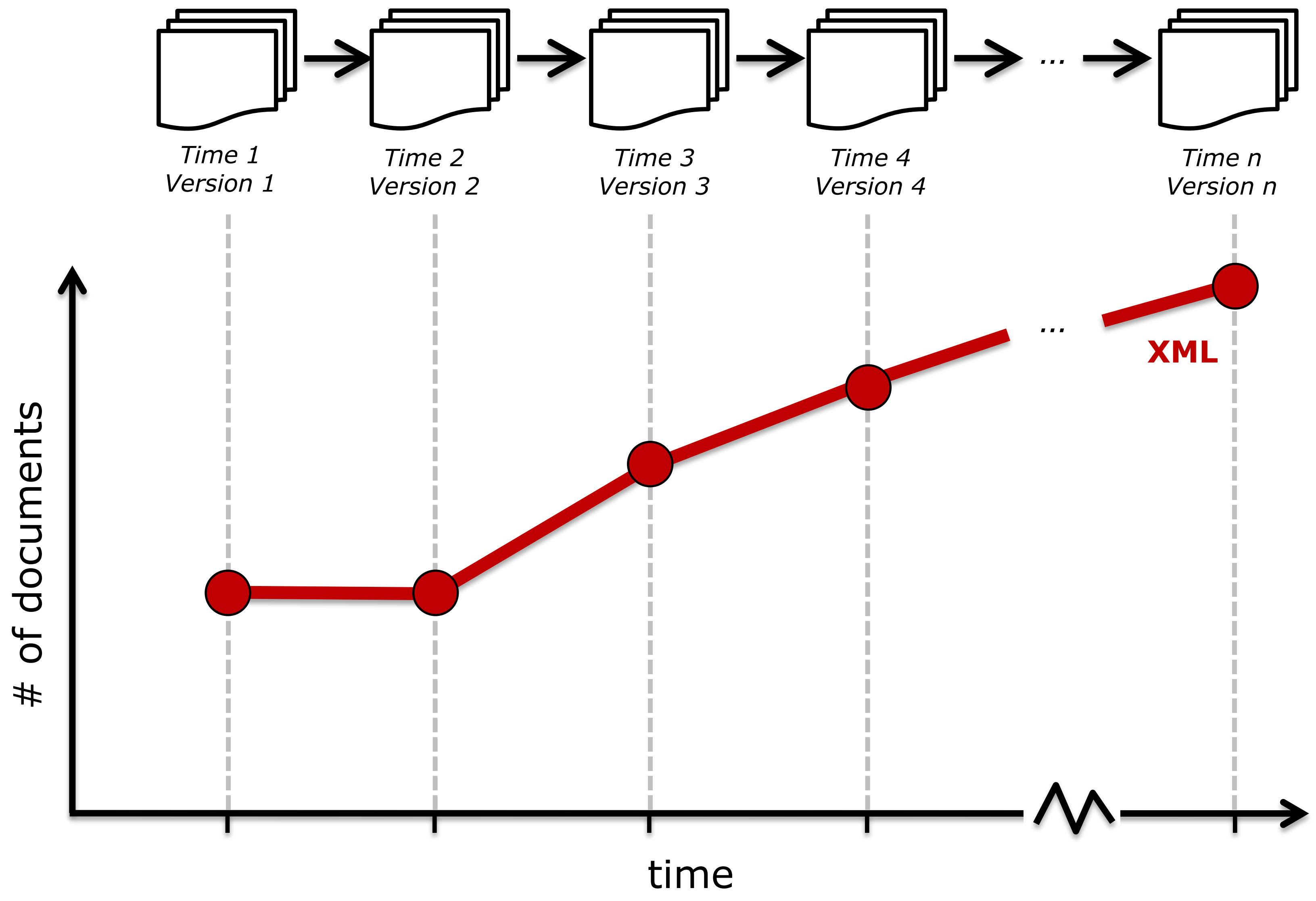
Topic Evolution Models



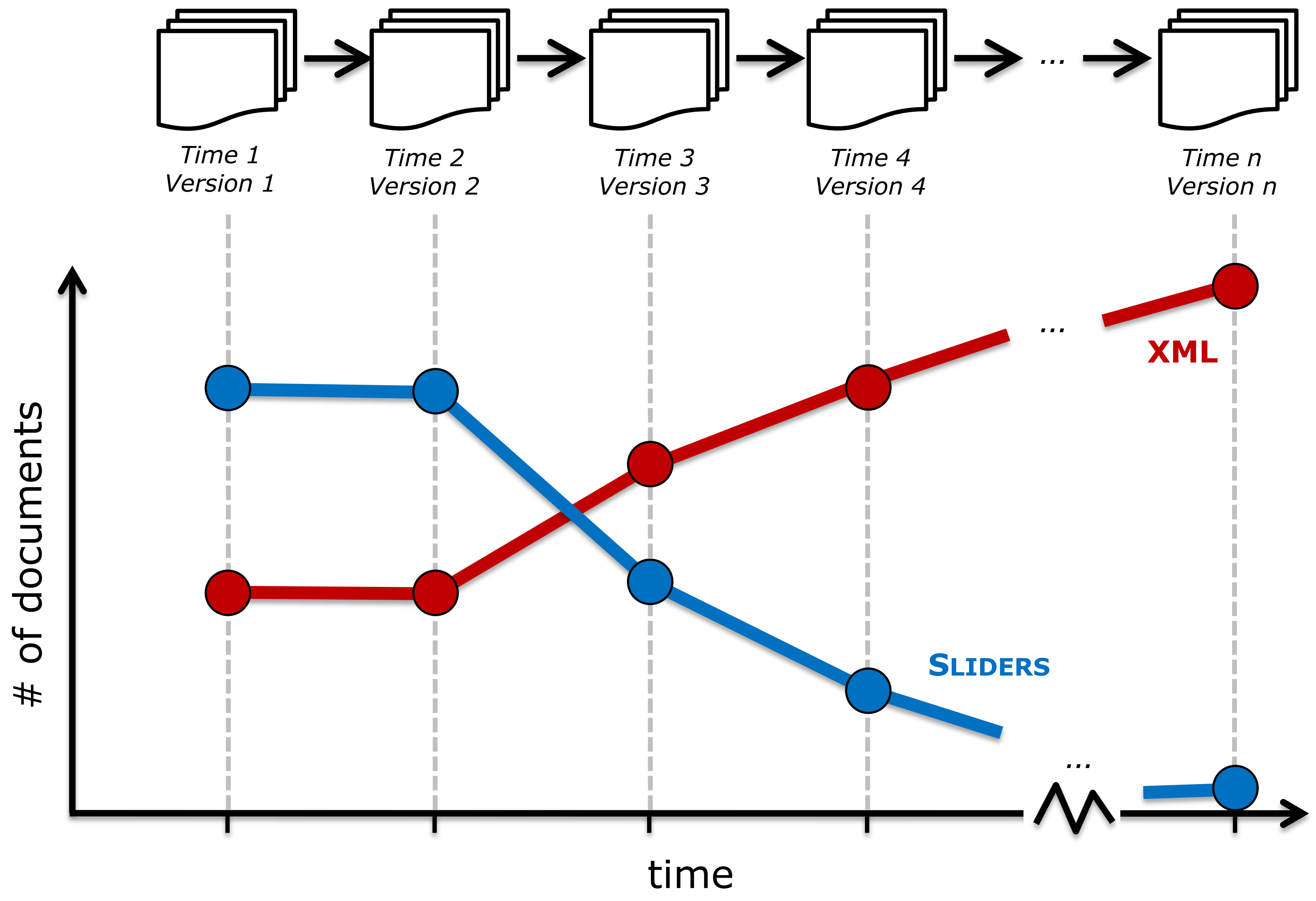
Topic Evolution Models



Topic Evolution Models



Topic Evolution Models



Topic Evolution Models

Program Comprehension

- When was the XML feature added?
- At what point did we switch to Java Swing?
- What features were affected by release 7.3?

Quality Assurance

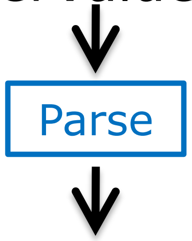
- What features are becoming too big?
- What features are becoming too scattered?
- What is being worked on right now?

Topic Models on Code

```
// Add a one if positive
if (sliderValue > 0){
  sliderValue++;
```

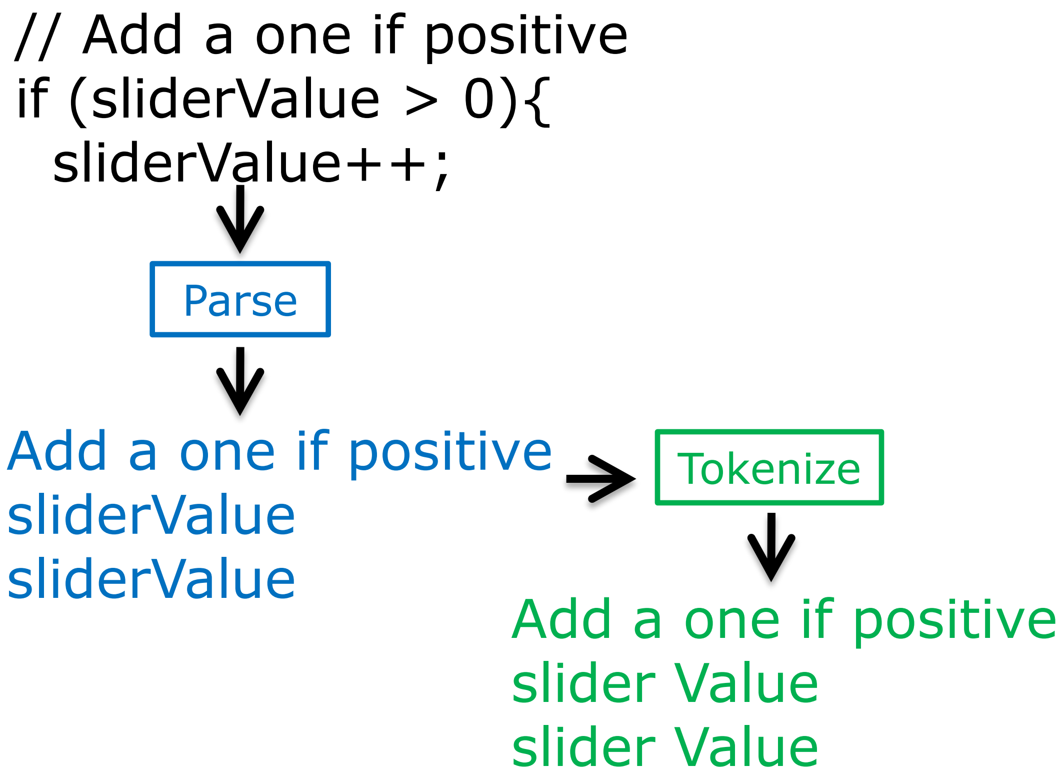
Topic Models on Code

```
// Add a one if positive
if (sliderValue > 0){
  sliderValue++;
```



Add a one if positive
sliderValue
sliderValue

Topic Models on Code

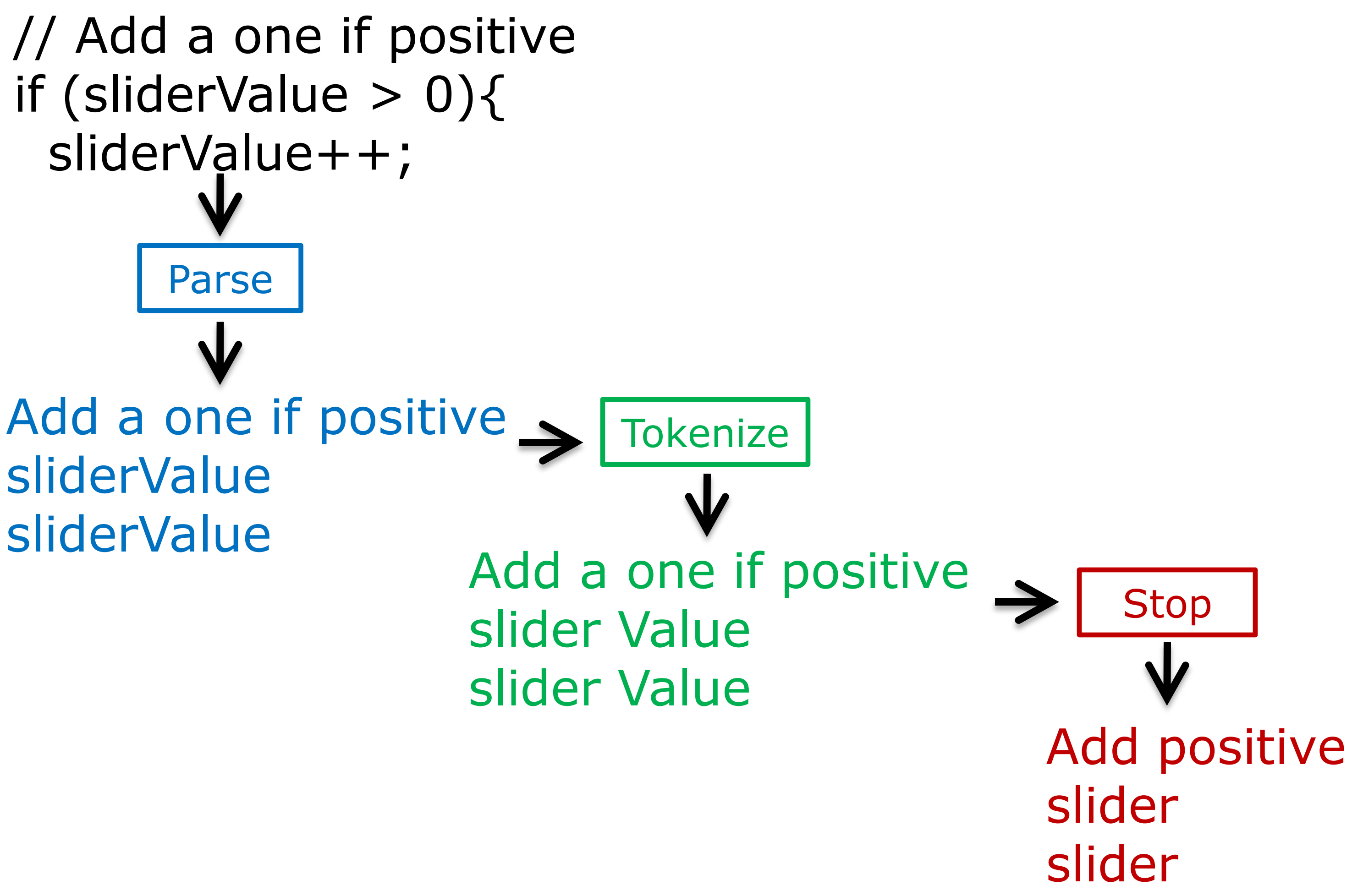


```
// Add a one if positive
if (sliderValue > 0){
  sliderValue++;
```

Add a one if positive
sliderValue
sliderValue

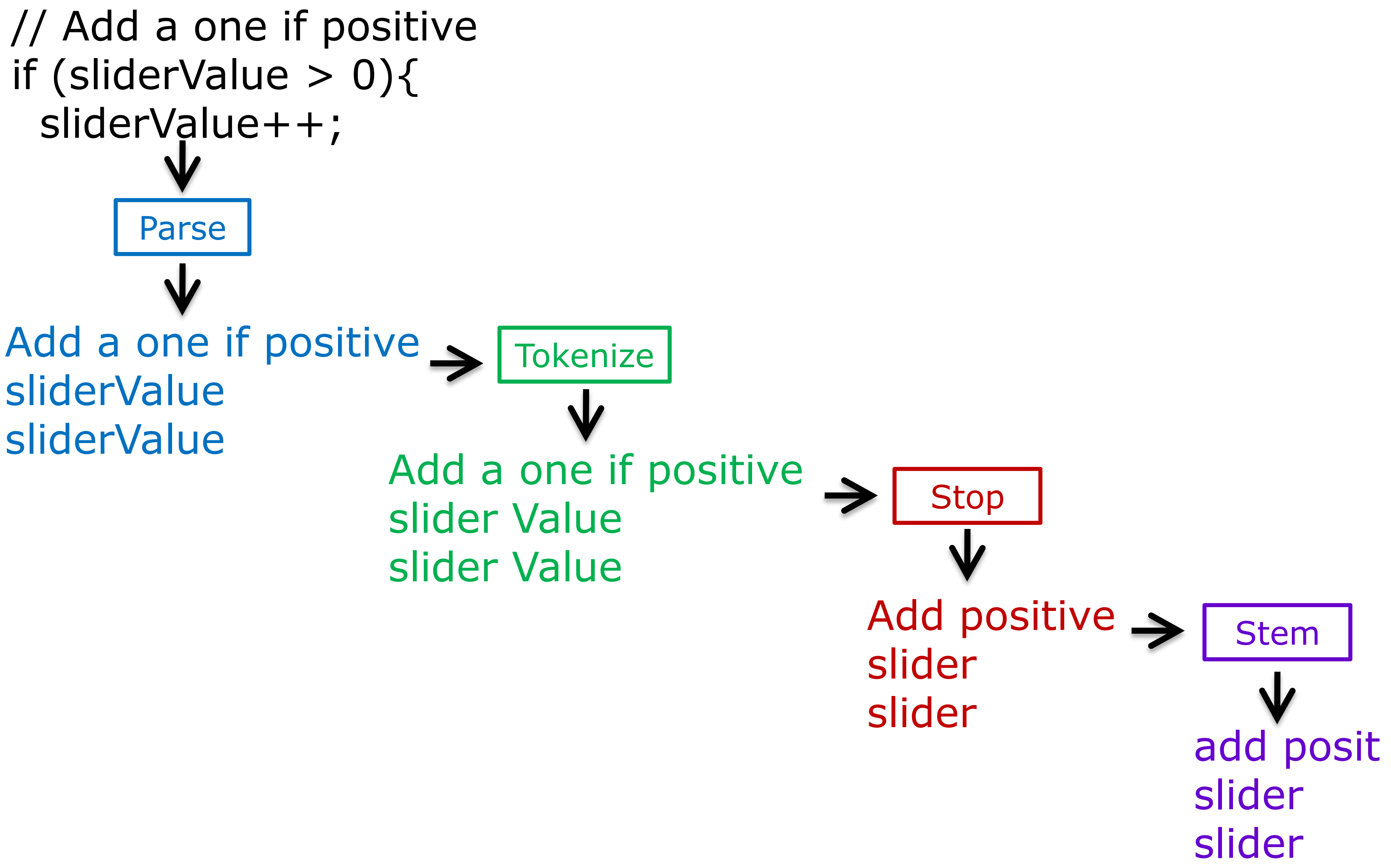
Add a one if positive
slider Value
slider Value

Topic Models on Code



```
// Add a one if positive
if (sliderValue > 0){
  sliderValue++;
```

Topic Models on Code



```
// Add a one if positive
if (sliderValue > 0){
  sliderValue++;
```

What causes topics to evolve in source code?

Developer *change activities*?

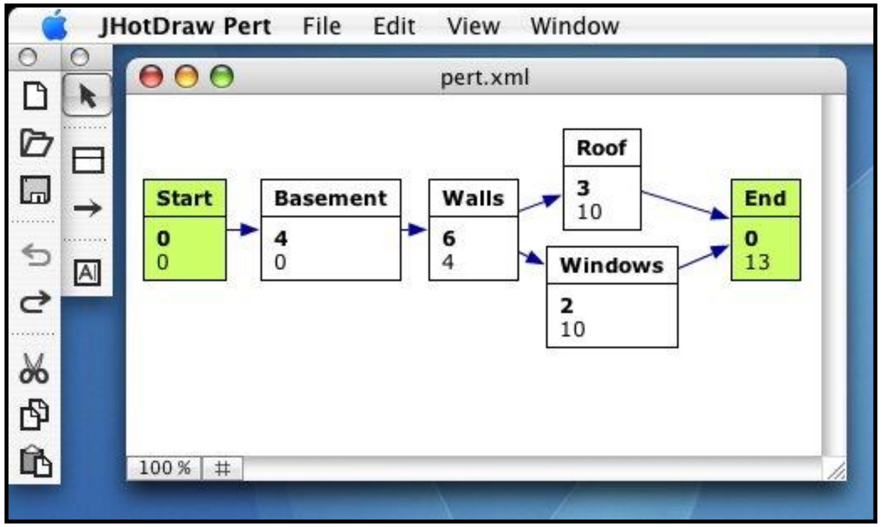
- Corrective evolution (bug fixes)
- Refactoring (new libraries, structure)
- New functionalities or features

Something else?

- Noise/error in model
- ???

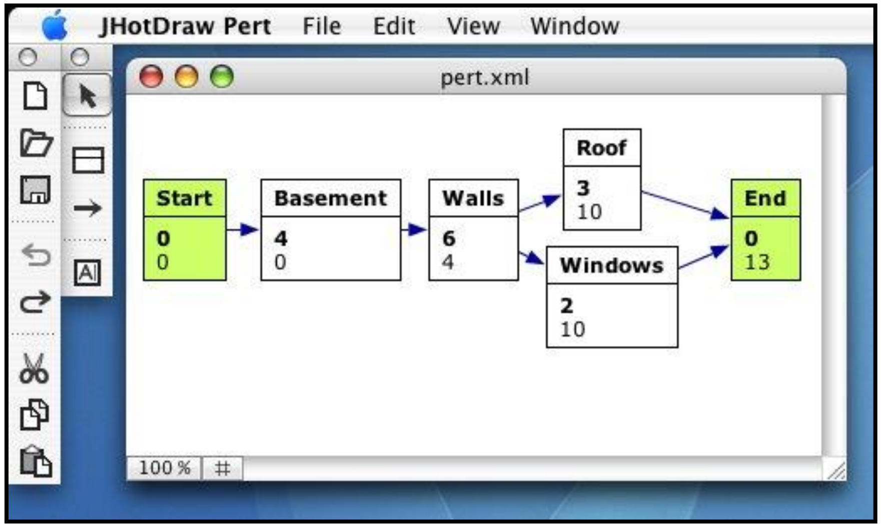
Case Study: JHotDraw

- **Medium-sized, open source drawing framework**
  - Well studied and documented
  - Manageable size
- **12 release versions**
  - 9 years
  - 17 KLOC → 127 KLOC
- **Used MALLET tool**
  - 45 LDA topics

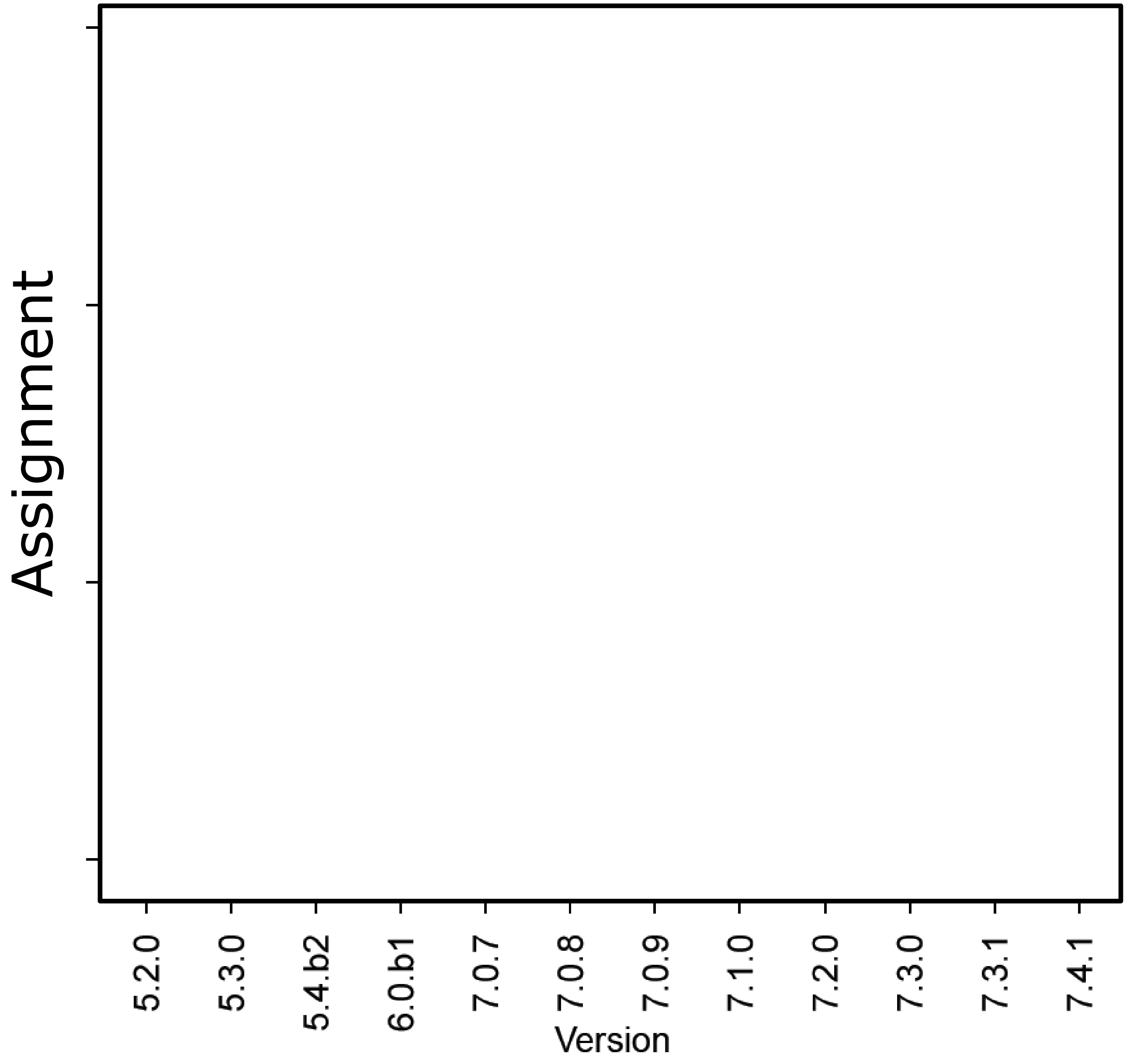


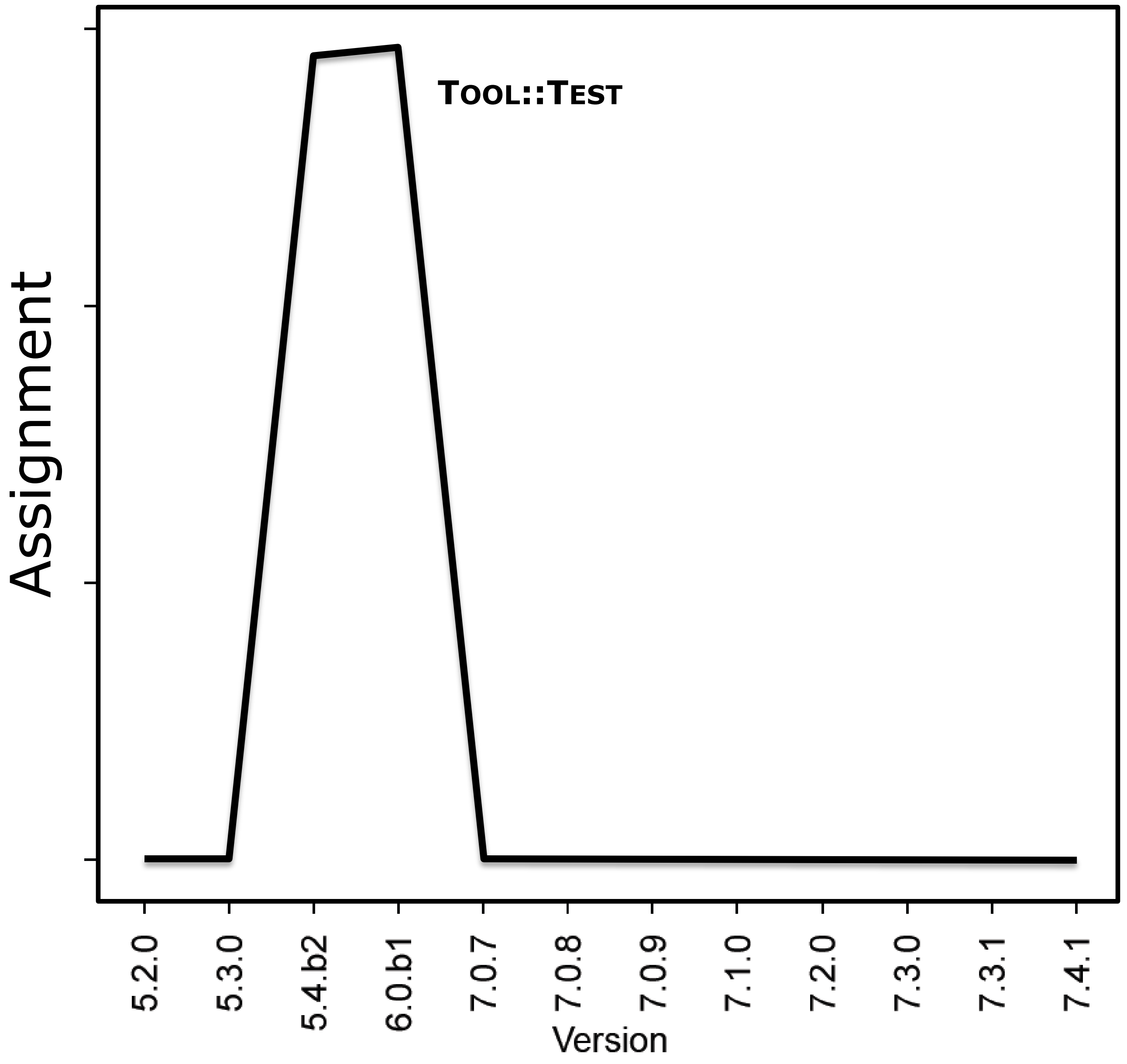
Case Study: JHotDraw

- **Medium-sized, open source drawing framework**
  - Well studied and documented
  - Manageable size
- **12 release versions**
  - 9 years
  - 17 KLOC → 127 KLOC
- **Used MALLET tool**
  - 45 LDA topics

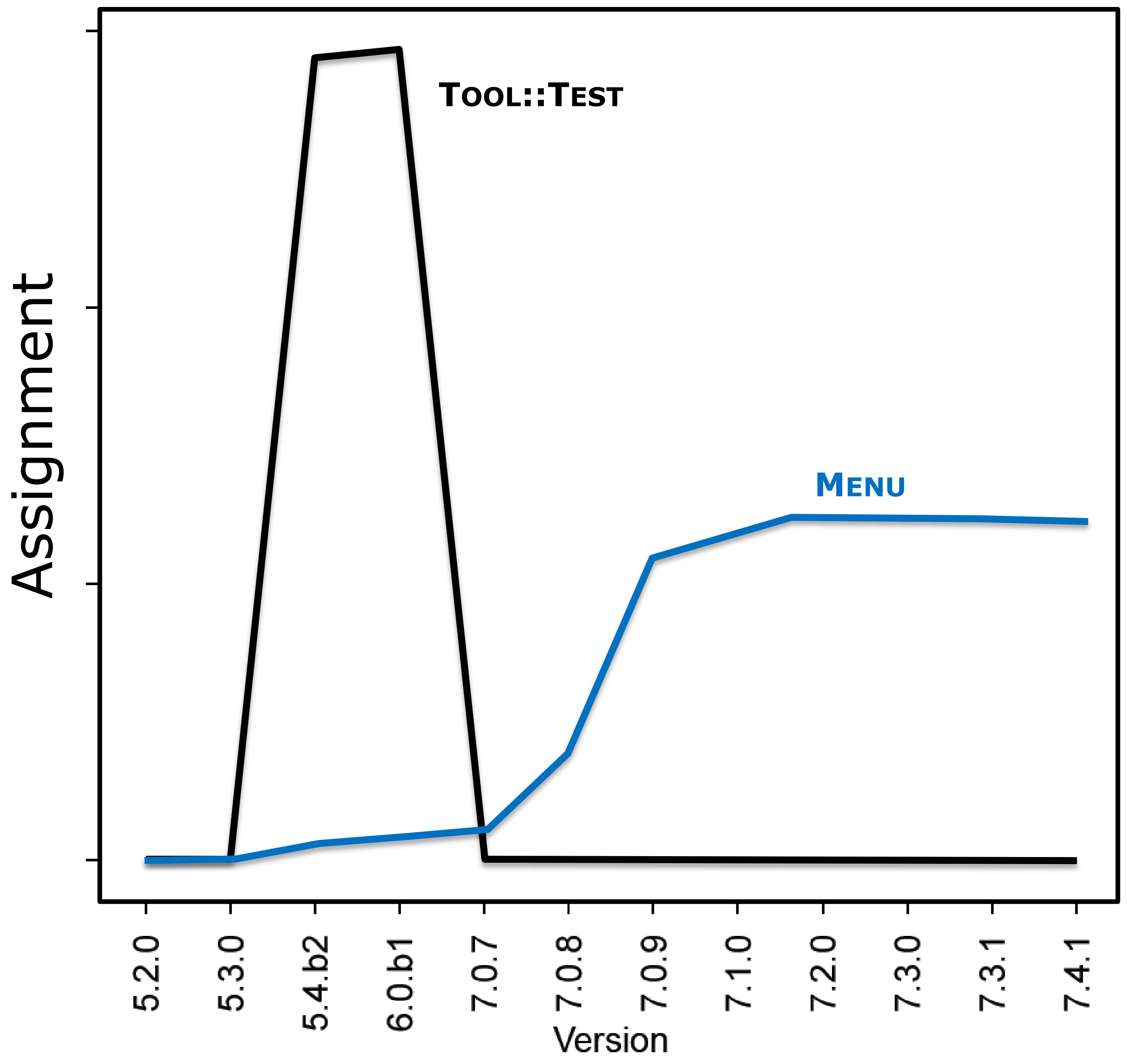


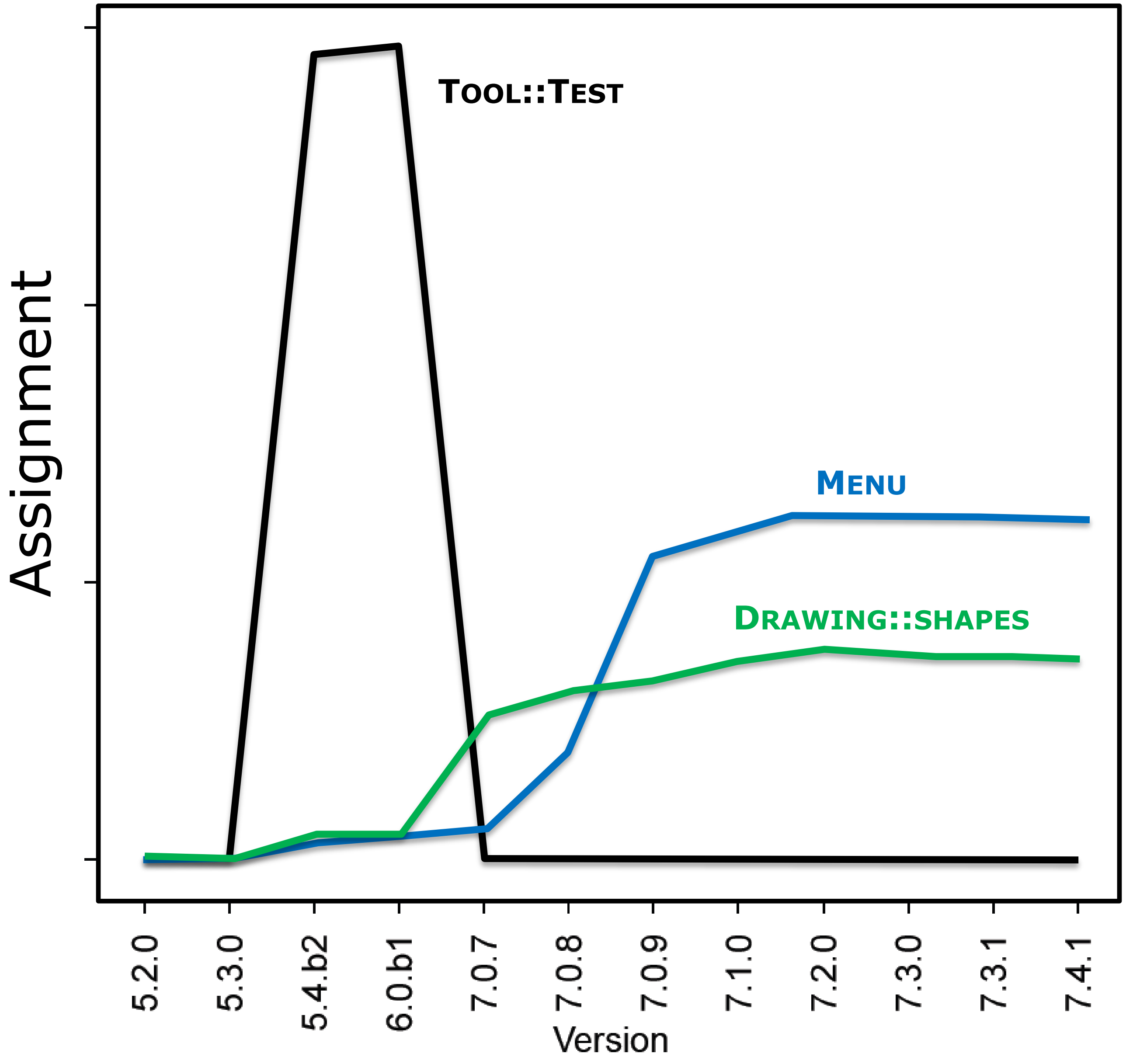
Are topic evolutions consistent with the code changes?


Assignment
5.2.0
5.3.0
5.4.b2
6.0.b1
7.0.7
7.0.8
7.0.9
7.1.0
7.2.0
7.3.0
7.3.1
7.4.1
Version

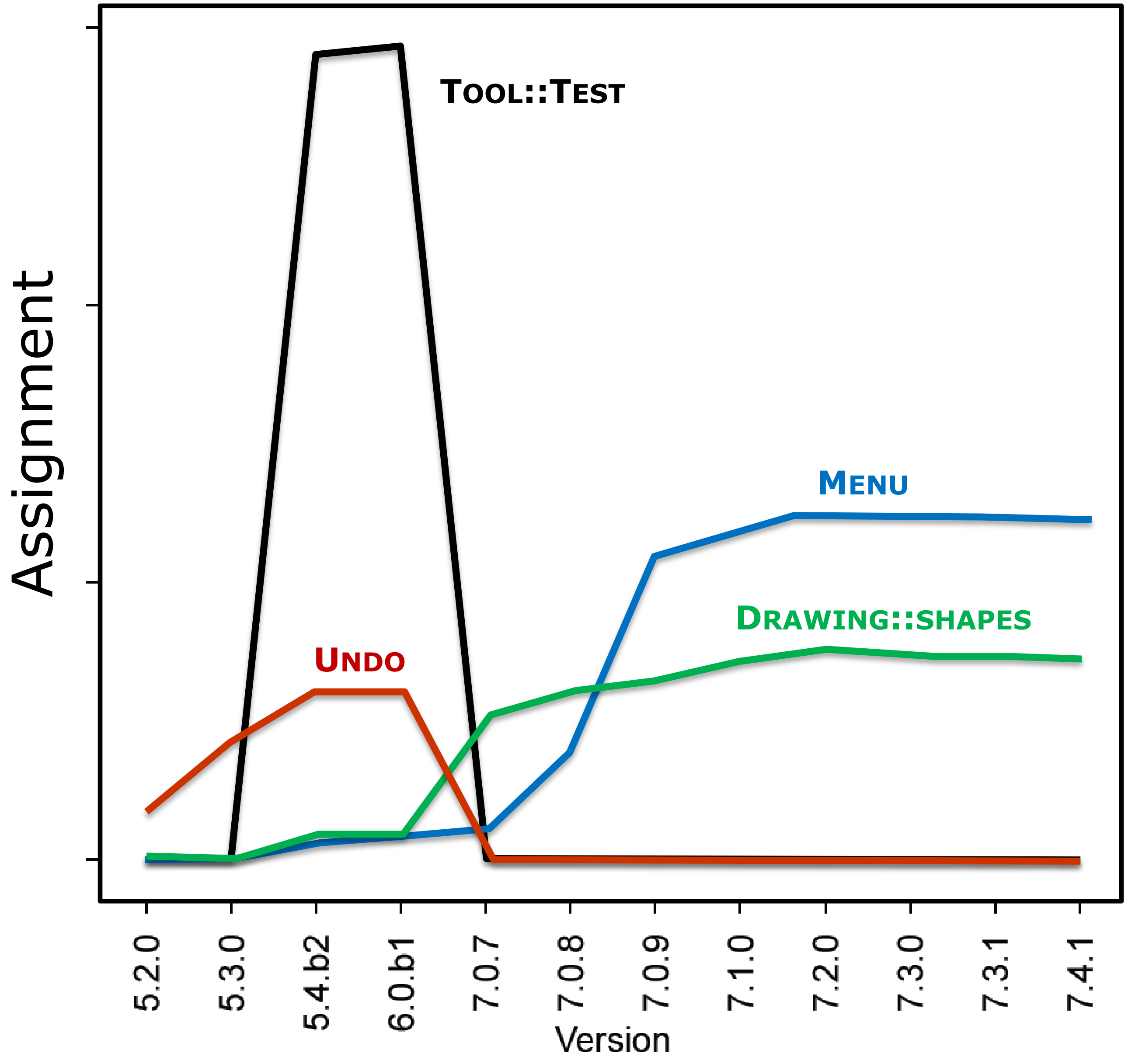


Tool::Test
Assignment
5.2.0
5.3.0
5.4.b2
6.0.b1
7.0.7
7.0.8
7.0.9
7.1.0
7.2.0
7.3.0
7.3.1
7.4.1
Version


Assignment
TOOL::TEST
MENU
5.2.0
5.3.0
5.4.b2
6.0.b1
7.0.7
7.0.8
7.0.9
7.1.0
7.2.0
7.3.0
7.3.1
7.4.1
Version



Assignment
Tool::Test
Menu
Drawing::shapes
5.2.0
5.3.0
5.4.b2
6.0.b1
7.0.7
7.0.8
7.0.9
7.1.0
7.2.0
7.3.0
7.3.1
7.4.1
Version


Assignment
TOOL::TEST
MENU
DRAWING::SHAPES
UNDO
5.2.0
5.3.0
5.4.b2
6.0.b1
7.0.7
7.0.8
7.0.9
7.1.0
7.2.0
7.3.0
7.3.1
7.4.1
Version

Are topic evolutions consistent with the code changes?

Are topic evolutions consistent with the code changes?

495 change events

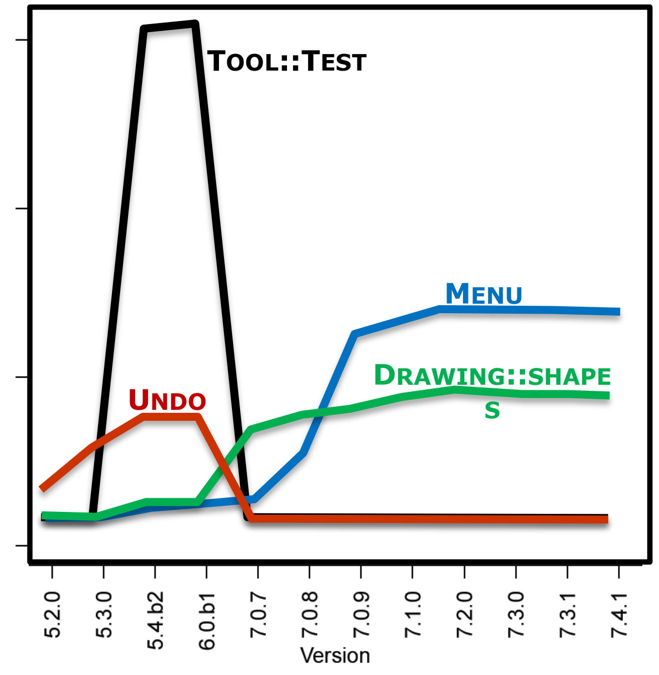
“24% increase in XML topic at version 7.1.0”

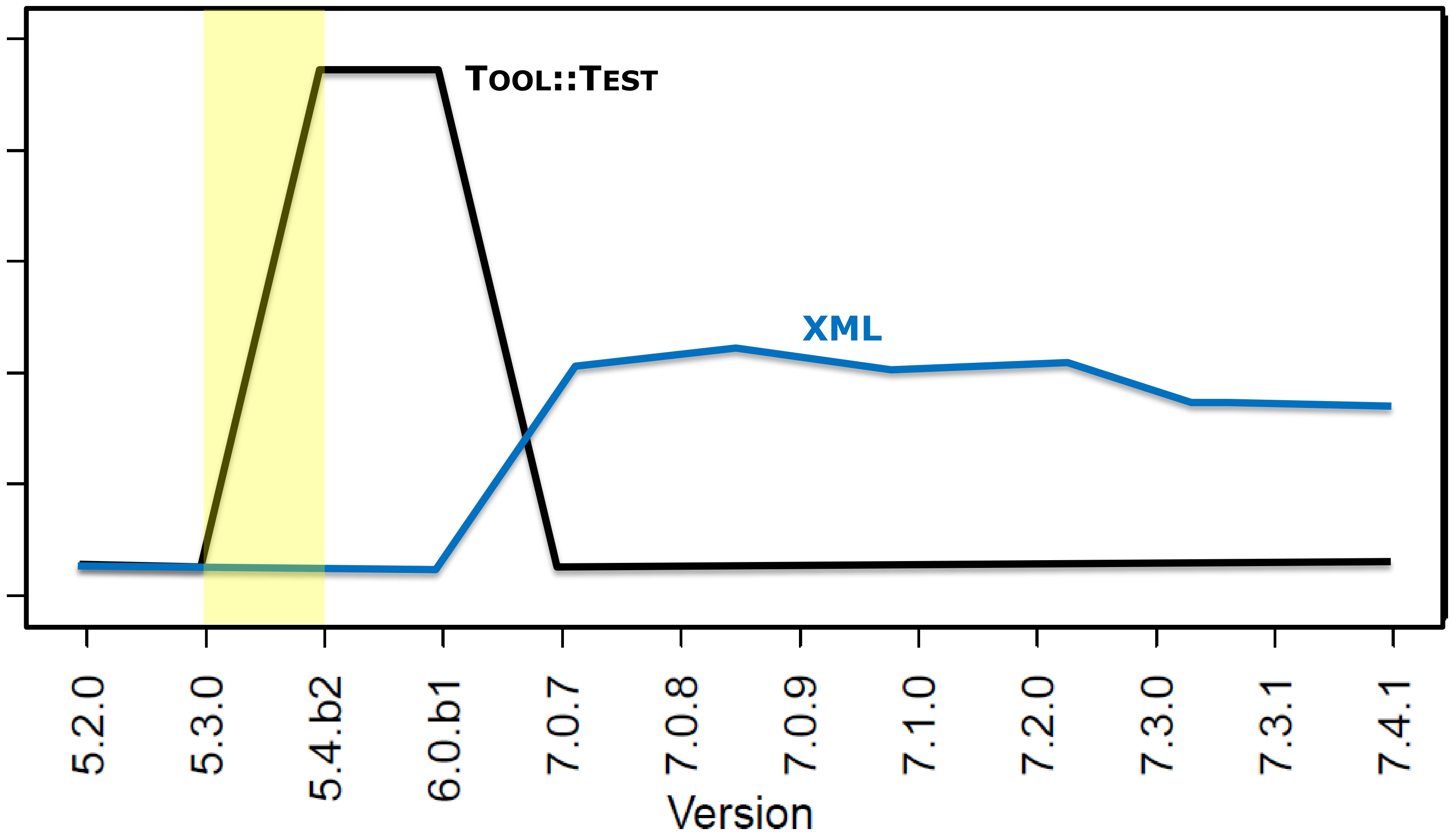
Selected 80 at random

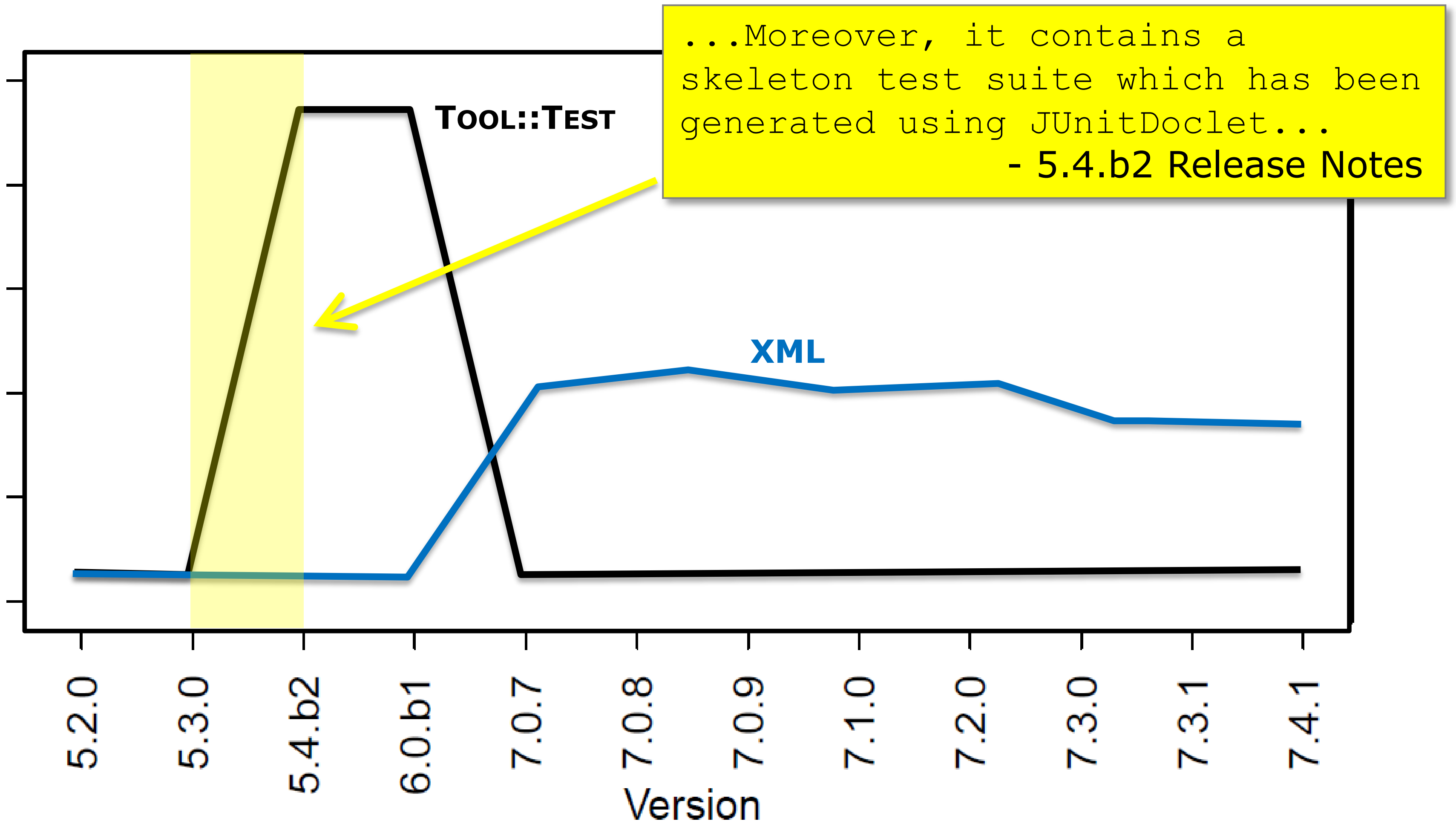
95% confidence interval, 5% error

Manually investigated project documentation

Change logs, release notes, commit logs

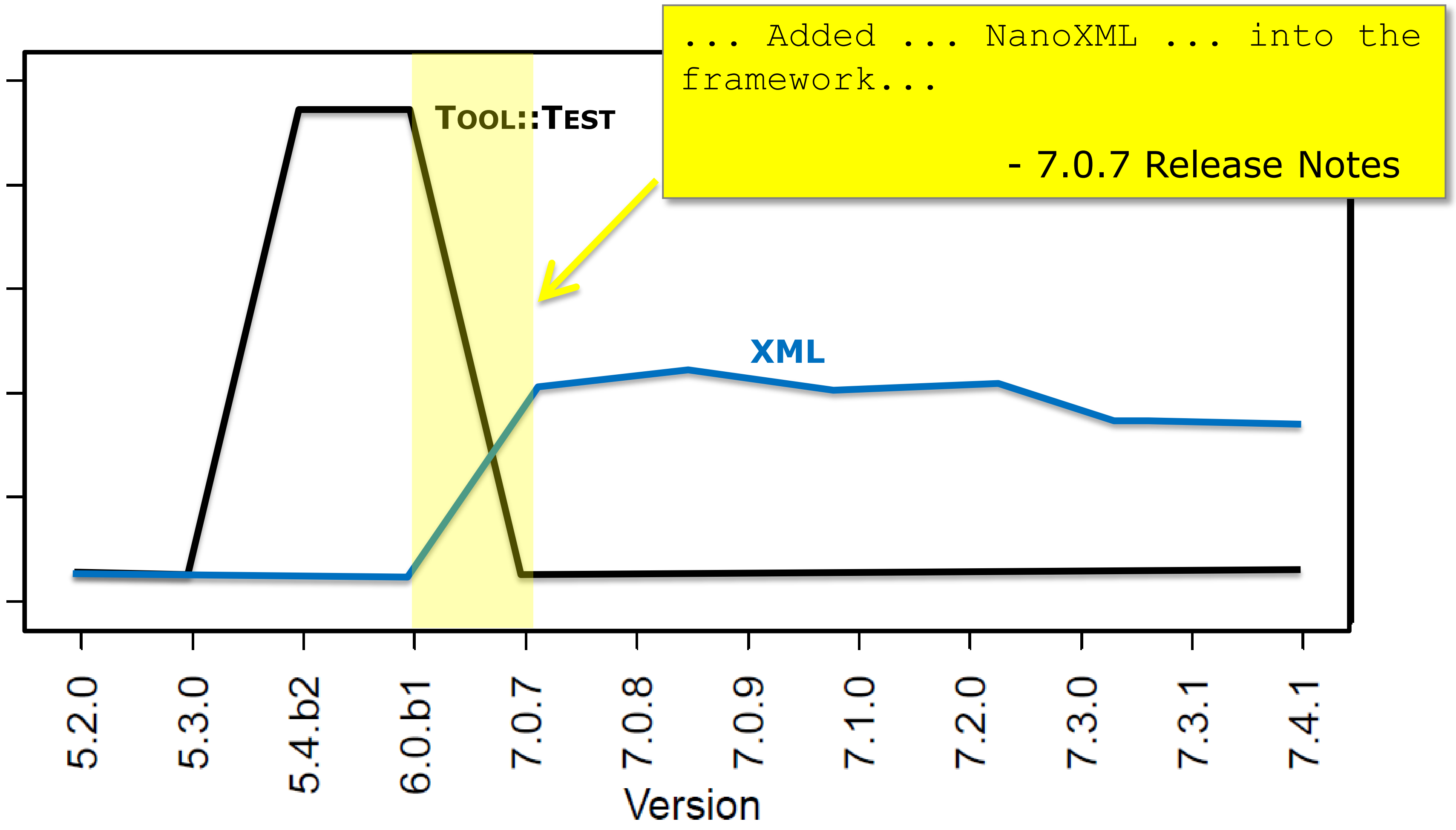



TOOL::TEST
XML
5.2.0
5.3.0
5.4.b2
6.0.b1
7.0.7
7.0.8
7.0.9
7.1.0
7.2.0
7.3.0
7.3.1
7.4.1
Version



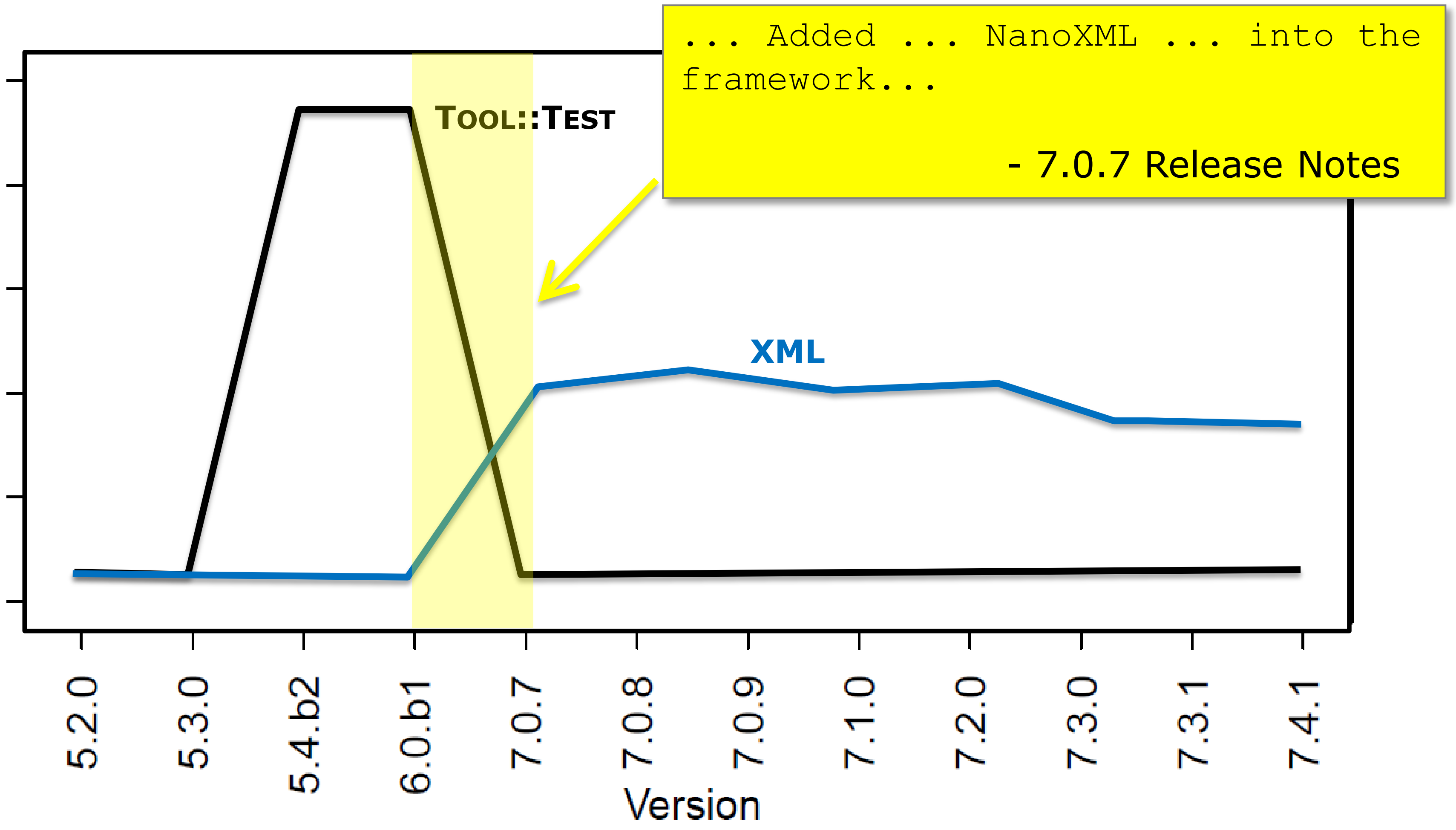
```
...Moreover, it contains a
skeleton test suite which has been
generated using JUnitDoclet...
```

- 5.4.b2 Release Notes



... Added ... NanoXML ... into the framework...

- 7.0.7 Release Notes



```
... Added ... NanoXML ... into the
framework...
```

- 7.0.7 Release Notes

Findings: 92% ± 5% of change events agree with documentation

Implications

Detected by →

Used in →

Software Changes

Dashboards

Implications

Detected by →

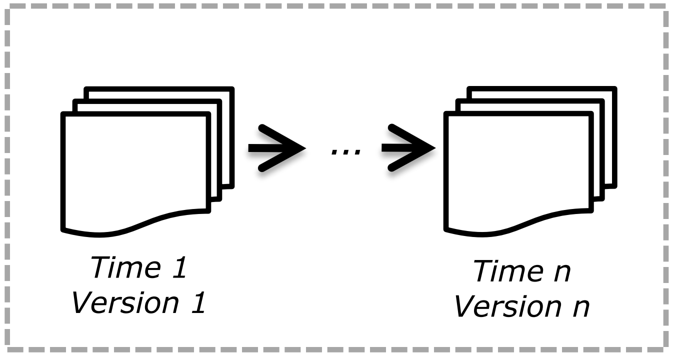
Latent **D**irichlet **A**llocation

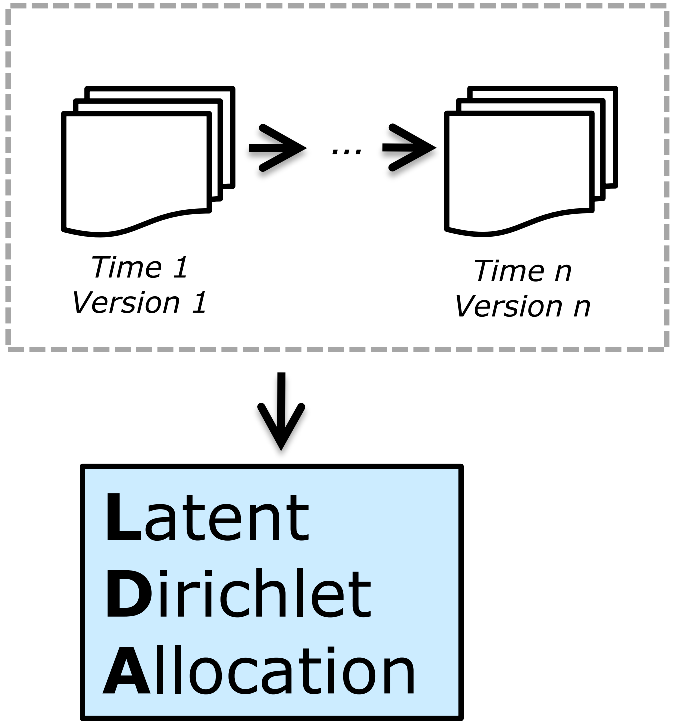
Used in →

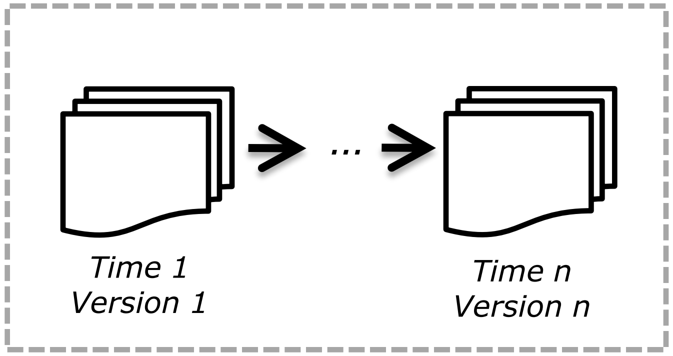
Software Changes

Topic Models

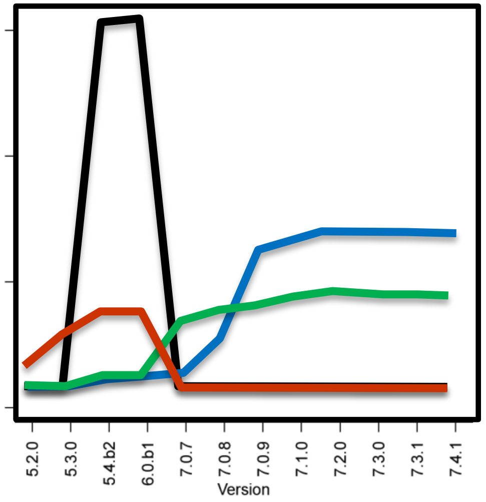
Dashboards

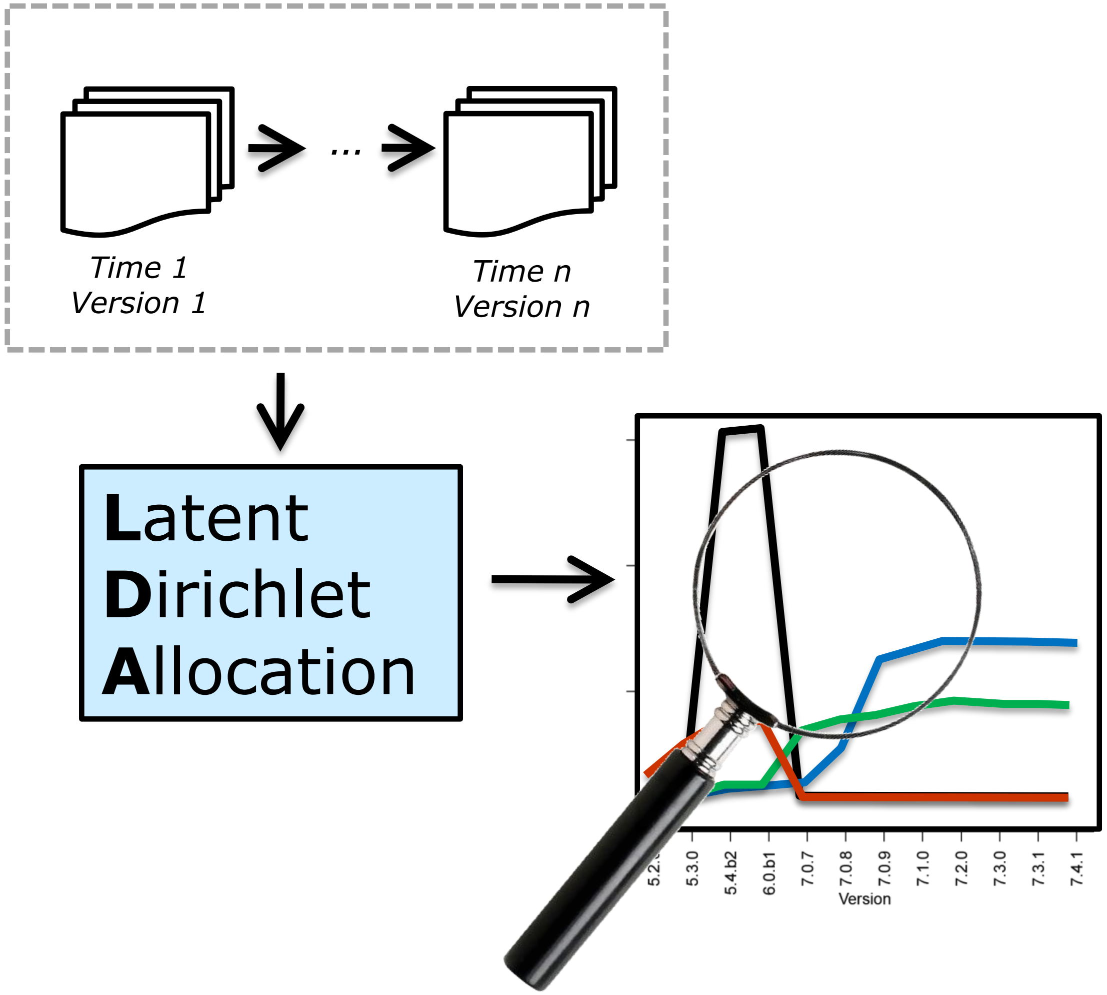

...
Time 1
Version 1
Time n
Version n

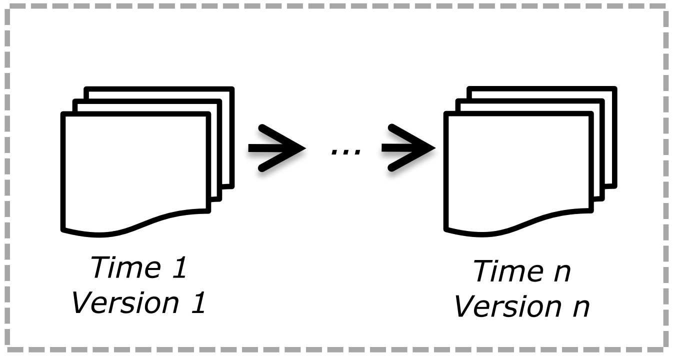

Time 1
Version 1
...
Time n
Version n
Latent
Dirichlet
Allocation



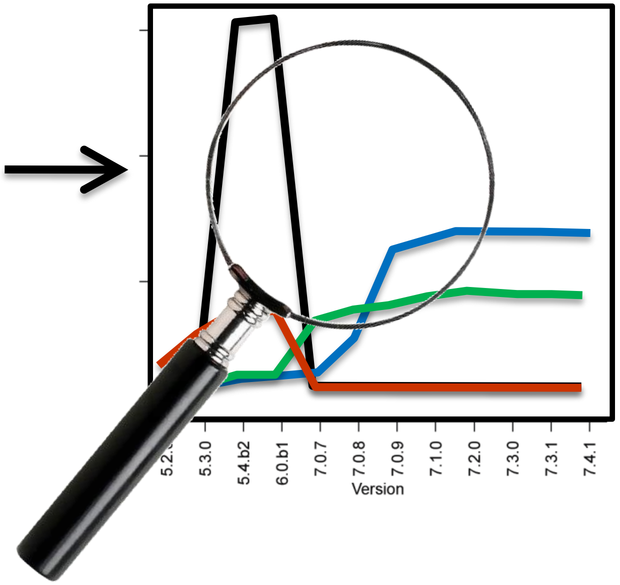
Latent
Dirichlet
Allocation

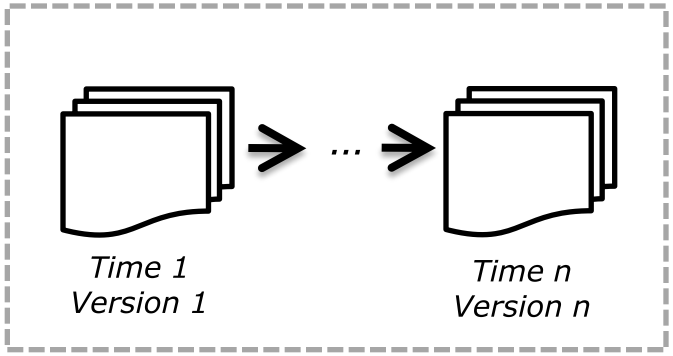



Time 1
Version 1
...
Time n
Version n
Latent
Dirichlet
Allocation
5.2
5.3.0
5.4.b2
6.0.b1
7.0.7
7.0.8
7.0.9
7.1.0
7.2.0
7.3.0
7.3.1
7.4.1
Version

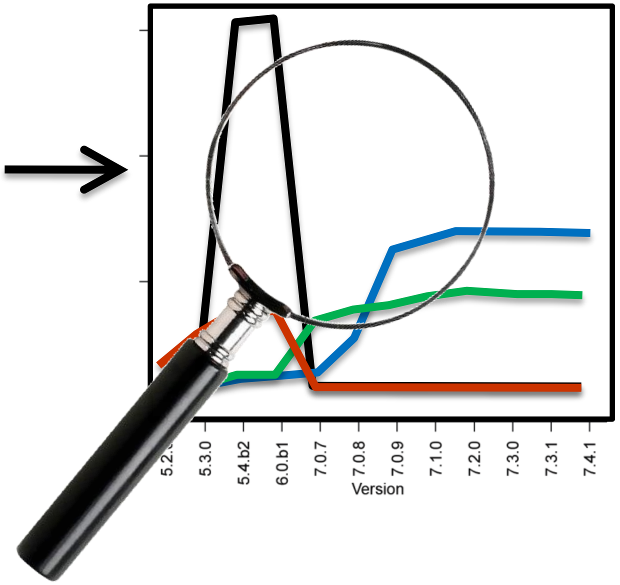


Latent
Dirichlet
Allocation





Latent
Dirichlet
Allocation



What do project managers want on the dashboard?

How to do recall for topic models?

Backups

Future Work

- **Investigate recall metric**
  - Do changes in source code result in changes in topic metrics?
- **Implement into dashboard**
  - Do managers actually like it?
- **More manual validation**
  - Larger and smaller systems
  - Closed-source systems
  - Systems from other domains