Software Ecosystems
Mircea Lungu
Ecosystems
A software ecosystem is* a set of inter-connected, independently developed, co-evolving software systems.
Generators of ecosystems
Generators of ecosystems
The 30MLOC of PL/1 Code in a Bank…

http://scg.unibe.ch/archive/papers/Aesc13a-PL1Ecosystem.pdf
Software Engineering Abstraction Levels

Inter-connected Systems / Ecosystem

Architecture

Design

Code
Are software systems in an ecosystem co-evolving???

Let us investigate API deprecation!
How Do Developers React to API Deprecation? The Case of a SimulTalk Ecosystem

ABSTRACT

APIs are a key component of Software Engineering and have a high emotional value for their users. As a result, their deprecation and evolution can be a critical concern for developers. This paper presents a case study on the impact of API deprecation on a SimulTalk ecosystem.

Introduction

This work explores the development of a software ecosystem, focusing on the impact of API deprecation on the ecosystem's health. The study aims to understand the developers' reactions and the consequences of API deprecation.

Acknowledgments

This research is supported by the SimulTalk project and other relevant organizations.

1. INTRODUCTION

The study involves a comprehensive analysis of the ecosystem's evolution over the past 8 years, from its initial inception in 2014 to the present day. In this section, we present an overview of the study's methodology and key findings.

2.3 Contributors

2.5 Projects

95,000 Classes

110,000 Commits

600,000,000 LOC
RQ₂: Magnitude of ripple effects triggered by API deprecation?

Reacting projects

- Deprecations regularly impact the ecosystem
- The impact can be large
True Story

Why is my code broken today when it worked yesterday?
Opportunities
Standing on the shoulders of giants
Good programmers know what to write; great ones know what to rewrite and reuse.

(ES Raymond, The Cathedral and The Bazaar)
The Network Effect
The value of an ecosystem increases with the number of systems it contains
The Evolution of Gnome

Introduction

Evolution

Nautilus

Growth

Maturity
Better Tool and Developer Support
# Mining Trends in Library Usage

## Table 2: Switching back to older library versions for the period January 2007–January 2009

<table>
<thead>
<tr>
<th>Library</th>
<th># usages</th>
<th># switched back</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>junit 3.8.1</td>
<td>1501</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>junit 3.8.2</td>
<td>293</td>
<td>1</td>
<td>&lt;1%</td>
</tr>
<tr>
<td>junit 4.4</td>
<td>84</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>log4j 1.2.8</td>
<td>269</td>
<td>3</td>
<td>2%</td>
</tr>
<tr>
<td>log4j 1.2.14</td>
<td>114</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>log4j 1.2.15</td>
<td>7</td>
<td>4</td>
<td>57%</td>
</tr>
<tr>
<td>servlet-api 2.3</td>
<td>182</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>servlet-api 2.5</td>
<td>10</td>
<td>1</td>
<td>10%</td>
</tr>
<tr>
<td>derby 10.1</td>
<td>147</td>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>derby 10.2</td>
<td>31</td>
<td>0</td>
<td>0%</td>
</tr>
</tbody>
</table>
“Alphabetical ordering must die.”
–Jakob Nielsen
Geo-locating Knowledge Transfer in StackOverflow
Schenk, Lungu. SSE 2013
Usage of java.lang.Thread API in the Java Ecosystem

https://github.com/caracciolo/pangea/blob/master/demo/thread-stats.st
Overthrowing the Tyranny of Alphabetical Order in Documentation
Spasojević, Lungu, Nierstrasz, ICSME 2014
Challenges
Scale

Makes everything slower
Things that are affected:
build times, testing times, understanding, etc.
The industry-average productivity for a software product is about 10 to 50 of lines of delivered code per person per day (including all non-coding overhead).

Chapter 20.5
Trust

Not upset that you lied to me.
Upset that from now on I can't believe you.
“When systems depended on underlying systems, and those depended on things still older... it became impossible to know all the systems could do”
The Law of Leaky Abstractions

“All non-trivial abstractions, to some degree, are leaky”.

J. Spolsky
Designing a Run-Time Ecosystem…

How to encourage interaction while minimizing the required trust?

https://developer.chrome.com/extensions/content_scripts
Interdependence

Is a mixed blessing
Controlling the upstream propagation

Projects need to isolate themselves from the evolution of the others
Dependency Hell

1. Conflicting dependencies

2. Long chains of dependencies

3. Large Number of Dependencies
More Downsides of Inter-dependency…

Upstream evolution can be used as a strategy

Co-evolution can take a lot of effort. Must plan for co-evolution and put time aside.

(The Importance of Slack)
Awareness

Gets more difficult
Keeping up with the upstream is challenging
Survey of Information Needs in Microsoft

Find the relevant engineers for a feature
Find an expert on a given feature
Find all the resources related to a given feature, API, product
Find why a recent change was made
Being notified that a recent change affects an engineer’s work
Finding who might be affected by a given change to code/API
Survey of Information Needs in Open Source

http://scg.unibe.ch/scgbib?query=Haen14a
Participants: 75

Open JDK, Processing.js, jQuery, SciPy, NumPy, Pharo, Squeak, Seaside, Drupal, Core-audio, Apache Hadoop, Apache Cassandra, Google WebToolkit, Ubuntu, Soot and Zend Framework

Technologies
Developer Information Needs in Software Ecosystems

Haenni, Lungu, Schwarz, Nierstrasz, WEA 2014
I'm interested in

... the usability of my API

... which API methods are called

... unused methods and functionalities

... how the library is being used
**Upstream**

I'm interested in

... the usability of my API
... which API methods are called
... unused methods and functionalities
... how the library is being used

**Downstream**

I'm interested in

... the impact of changes.
... the estimated time to adapt to a new version...
... notifications about changes...

* I keep up to date with my upstream projects ...

Complete list in the paper...
What you should know!

> What is an ecosystem and why talk about it
> Opportunities associated with ecosystems
> Challenges that appear in live ecosystems
Can you answer these questions?

- What is *dependency hell*? What are some solutions?
- How would you mine library usage from the ecosystem?
- How would you approach detecting clones in a large ecosystem?
- What are the challenges for a developer working in an ecosystem?
- What are the benefits of software ecosystems?
Further Reading

Mandatory Reading

> **The Cathedral and the Bazaar**, Erik S. Raymond

Optional

> **The Law of Leaky Abstractions**, Joel Spolsky
> **Codebook: Discovering and Exploiting Relationships in Software Repositories**, Begel et al. 2010
You are free:
- to copy, distribute, display, and perform the work
- to make derivative works
- to make commercial use of the work

Under the following conditions:

**Attribution.** You must attribute the work in the manner specified by the author or licensor.

**Share Alike.** If you alter, transform, or build upon this work, you may distribute the resulting work only under a license identical to this one.

- For any reuse or distribution, you must make clear to others the license terms of this work.
- Any of these conditions can be waived if you get permission from the copyright holder.

Your fair use and other rights are in no way affected by the above.

http://creativecommons.org/licenses/by-sa/2.5/