Choice Matters: Contrasting Package Manager User Experience

Syful Islam

Raula Gaikovina Kula

Software Engineering Laboratory
Graduate School of Information Science
Nara Institute of Science and Technology

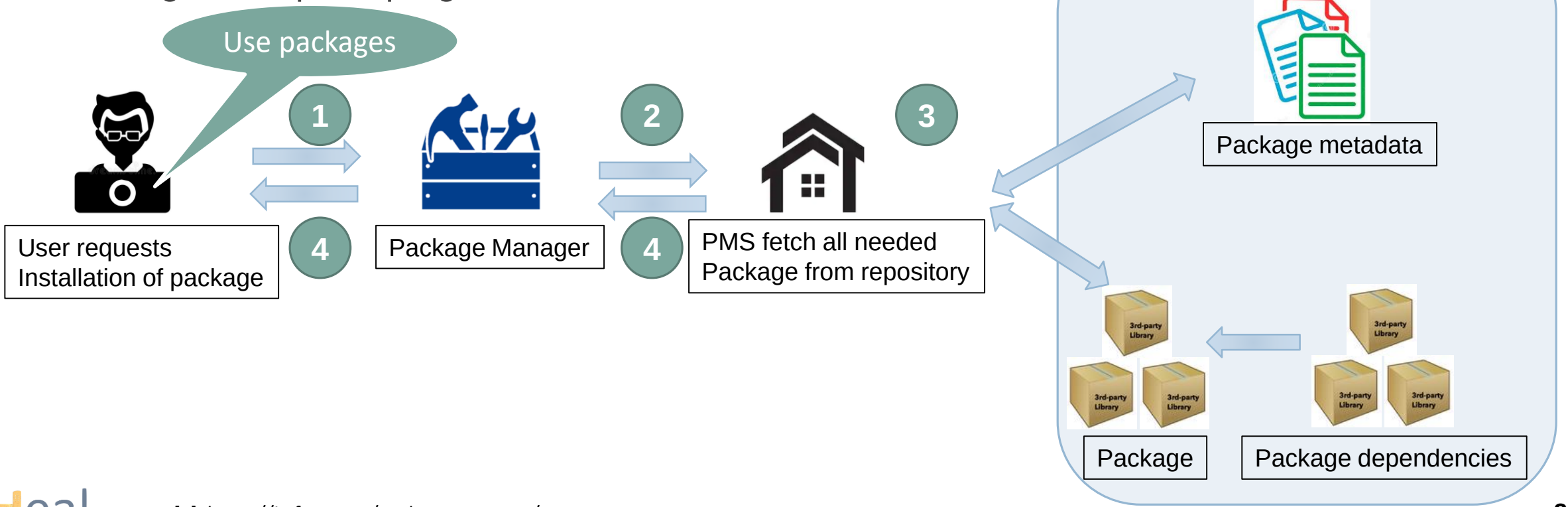
Nara Institute of Science and Technology NAIST®

NAIST

SoHeal

Package Managers are Crucial to Most Technology Stack

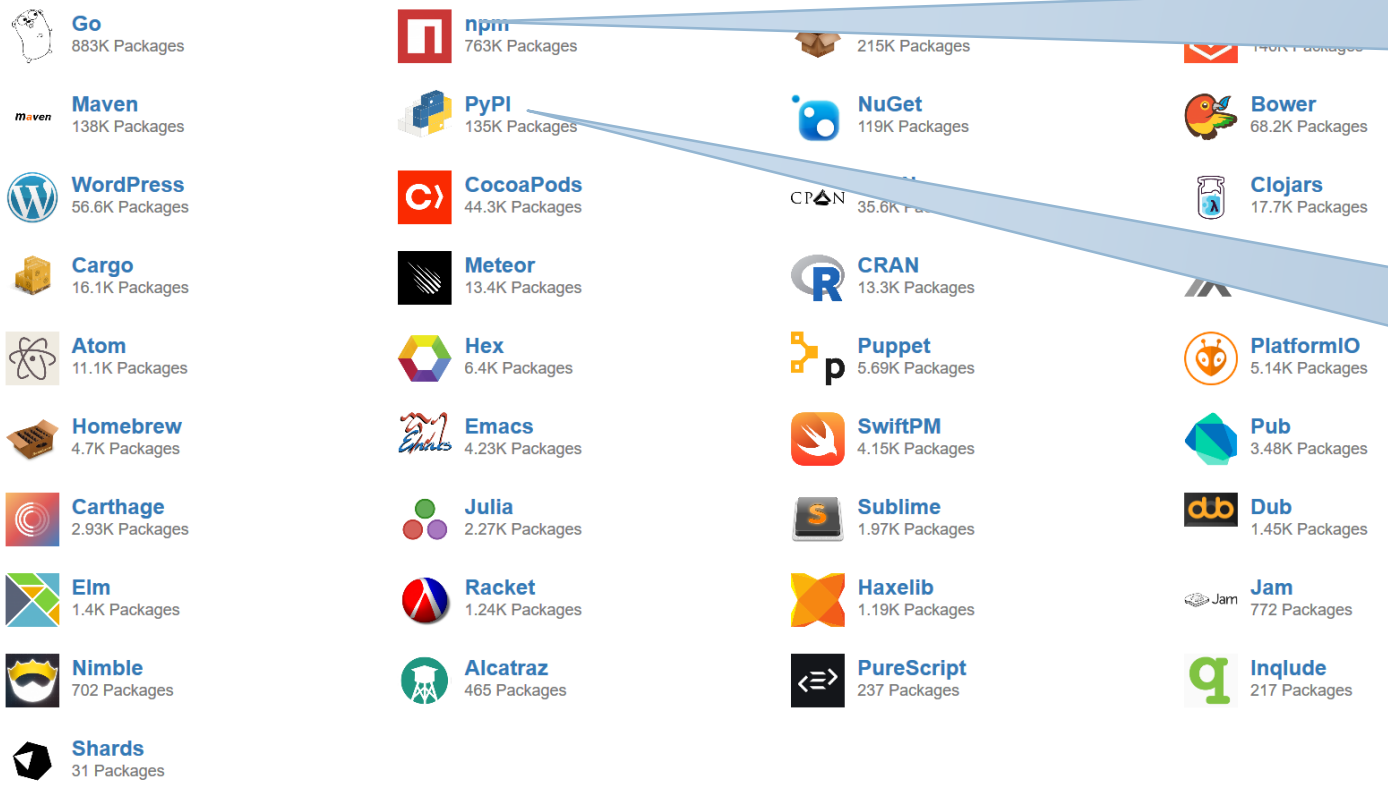
- Automates the process of
  - installing
  - upgrading
  - configuring and
  - Removing of computer programs



[1]. https://itsfoss.com/package-manager/

Diversity of Technology Stack has Led to Variety of Package Manager

- **Libraries.io** monitors open source packages across from **37** different package managers [1]



Brokers packages that run in Node.js environment and written in JavaScript

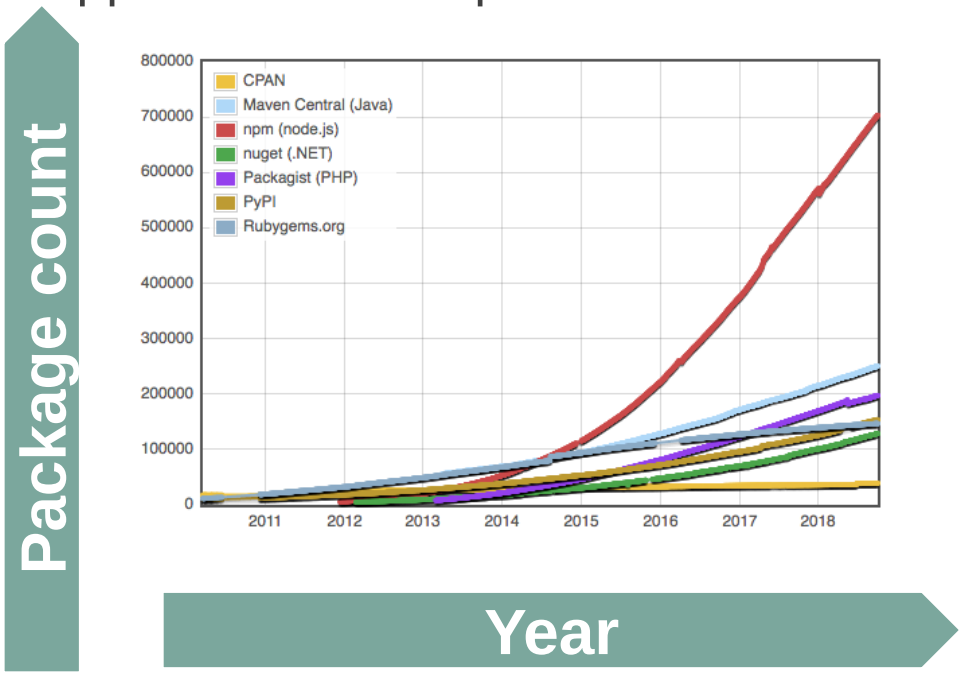
Brokers python package

SoHeal

[1] https://libraries.io/

Package Managers are Automated Solution for Applications that Heavily Rely on Third-party Packages

- Package Managers act as a broker of packages
  - Web building &
  - Mobile application development



Serve over 5 million open source packages

Ensuring the integrity and authenticity of the package

Grouping packages by function to reduce user confusion

Earlier in 2020, GitHub acquired the npm, which is the largest PM serving over 1.3 million packages to roughly 12 million developers, and constantly growing each day

SoHeal

[2]. https://blog.npmjs.org/post/180868064080/this-year-in-javascript-2018-in-review-and-npms

Related Research on Package Manager

- Studies on package dependency update
  - Bogart et al. investigates the reasons why developers do not update dependency [1]
  - Kula et al. found that 69% of the developers are unaware of the need to update dependency and perceived to extra workload [2].
  - Dietrich et. al, reported that developers are facing challenge on which version of package to depend [3].
- The common assumption is
  - Package manager itself does not factor when developers manage their dependencies.

1. Bogart, Christopher, et al. "How to break an API: cost negotiation and community values in three software ecosystems." Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering. 2016.

2. Kula, Raula Gaikovina, et al. "Do developers update their library dependencies?." Empirical Software Engineering 23.1 (2018): 384-417.

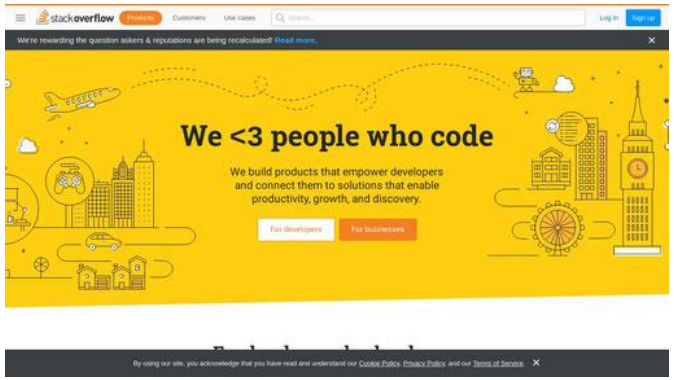
3. Dietrich, Jens, et al. "Dependency versioning in the wild." 2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR). IEEE, 2019.

Study Design: Objective and Data Source

- Objective
  - Mining information need on package manager
    - Characterize package manager issues faced by users
    - Investigate how the choice of package manager impacts the users experience of software developers.
- Data Source
  - Libraries.io
    - 16 Package managers
      - Dedicated webpage, archieve and not a operating system manager
  - Stack Overflow
    - Developers question posts

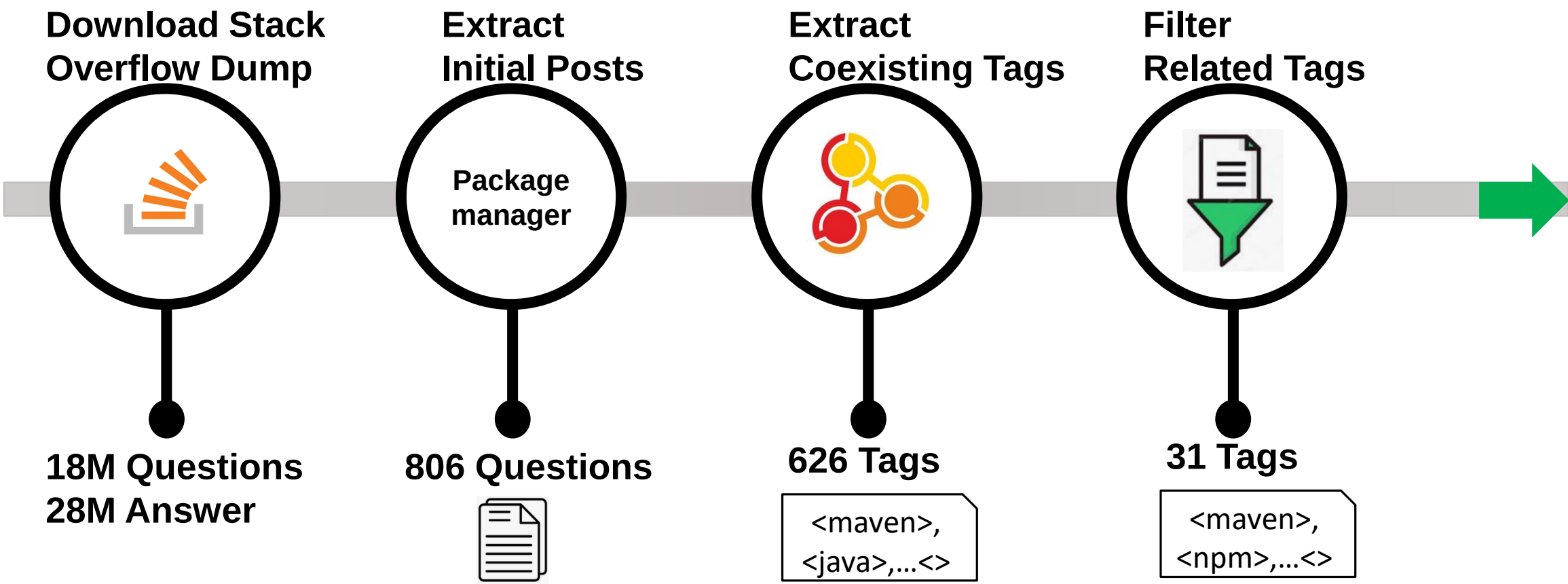
Developers

Communicate issues

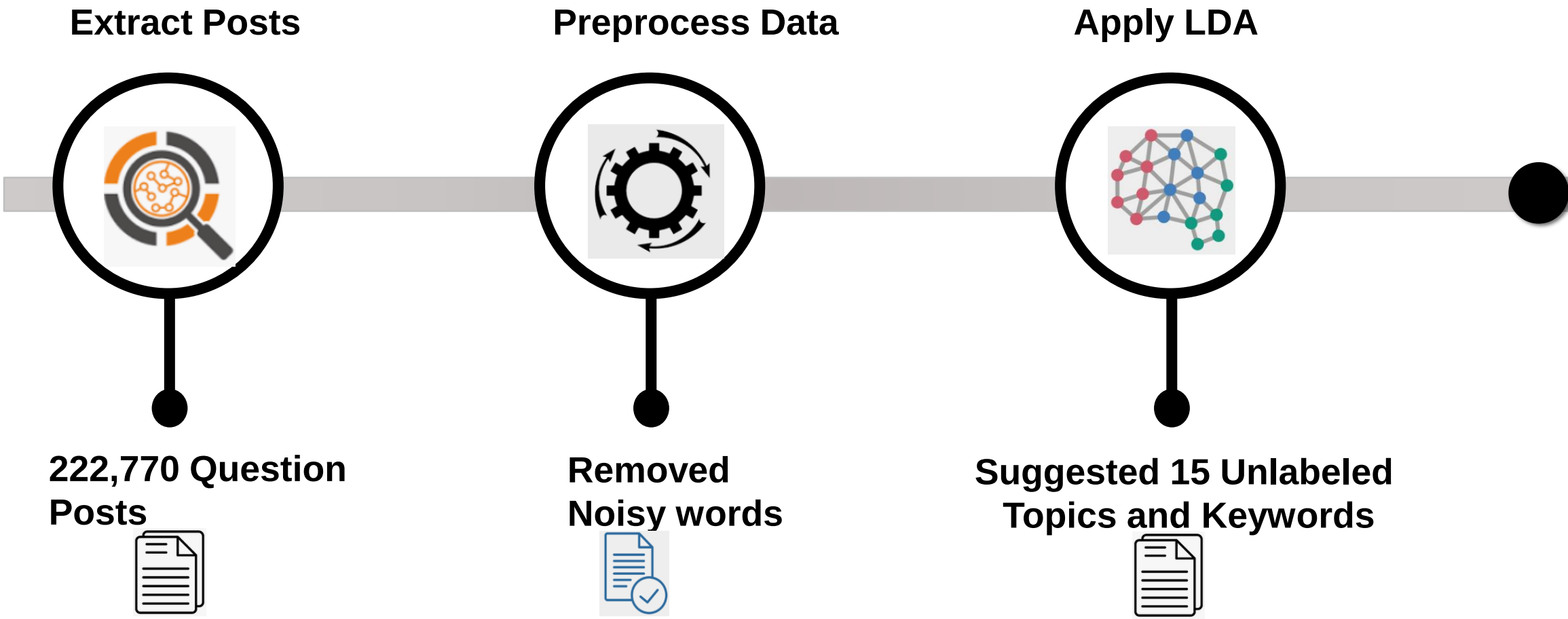


Stack Overflow

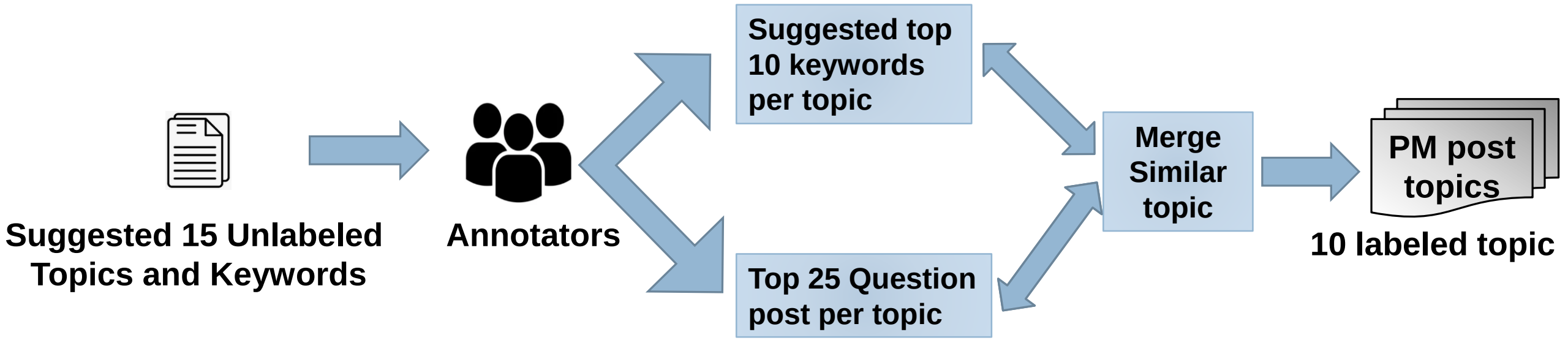
Collecting Package Manager Related Posts from Stack Overflow [1/2]



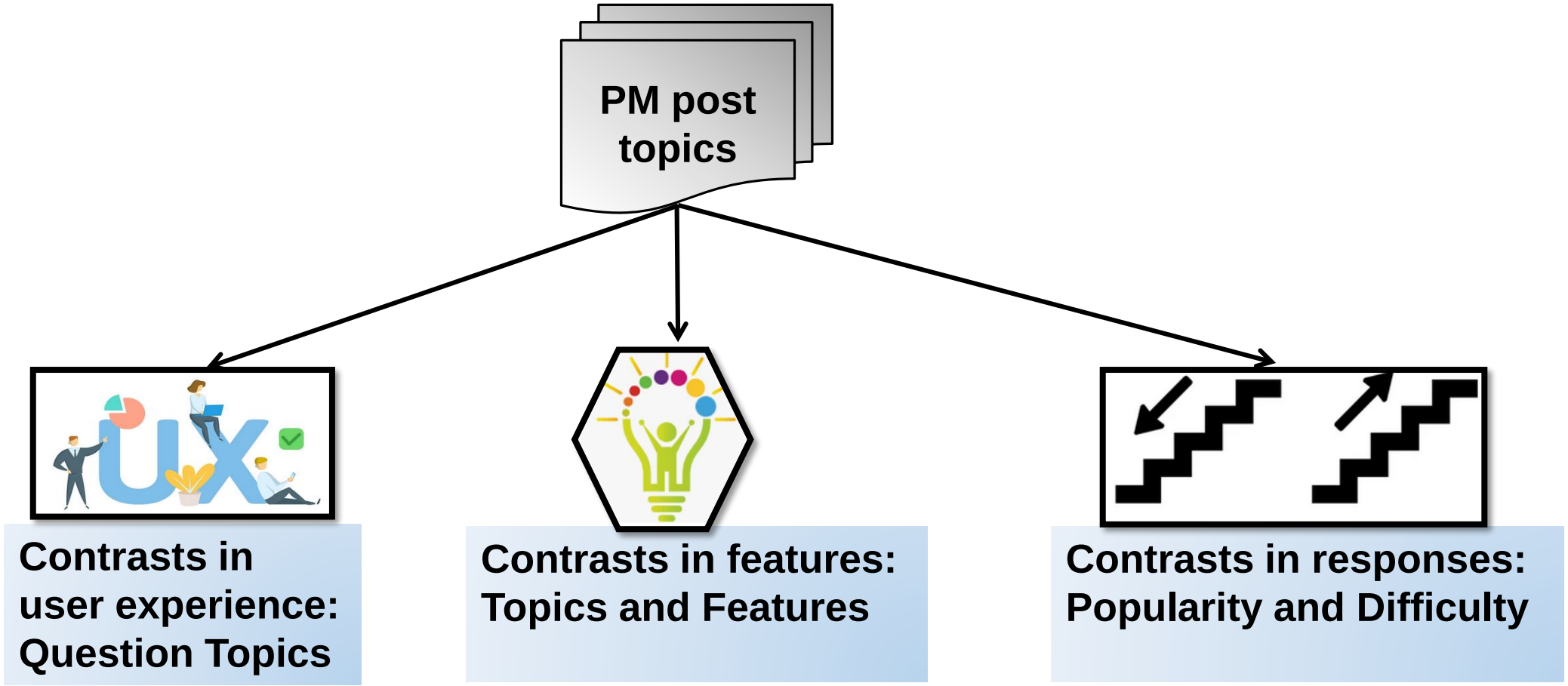
Topic Modeling on Package Manager Posts from Stack Overflow [2/2]



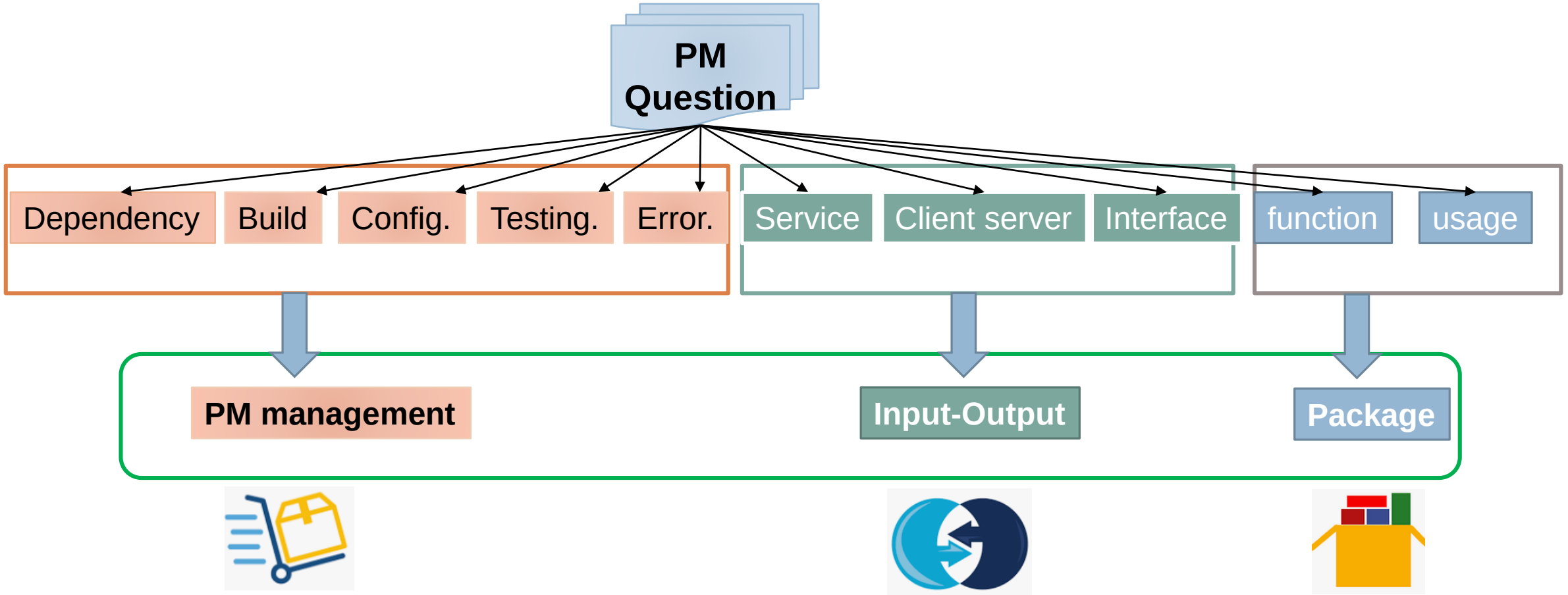
Naming Process of the Unlabeled Topics



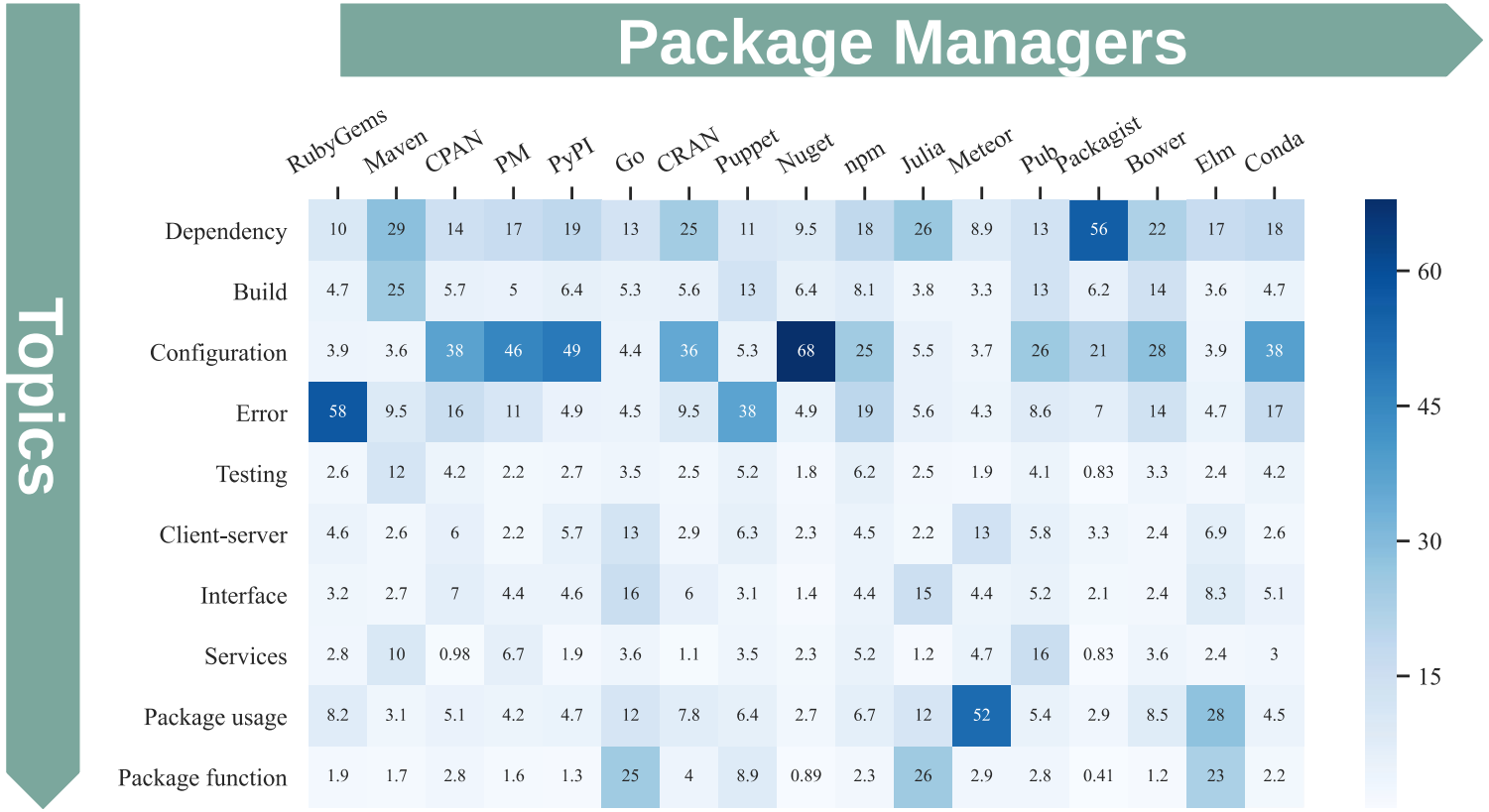
Characterizing Package Manager Issues



Contrasts in User Experience: Topics [1/5]



Contrasts in Features: Topics [2/5]



NuGet, CPAN, PyPI, Conda users have configuration related issues

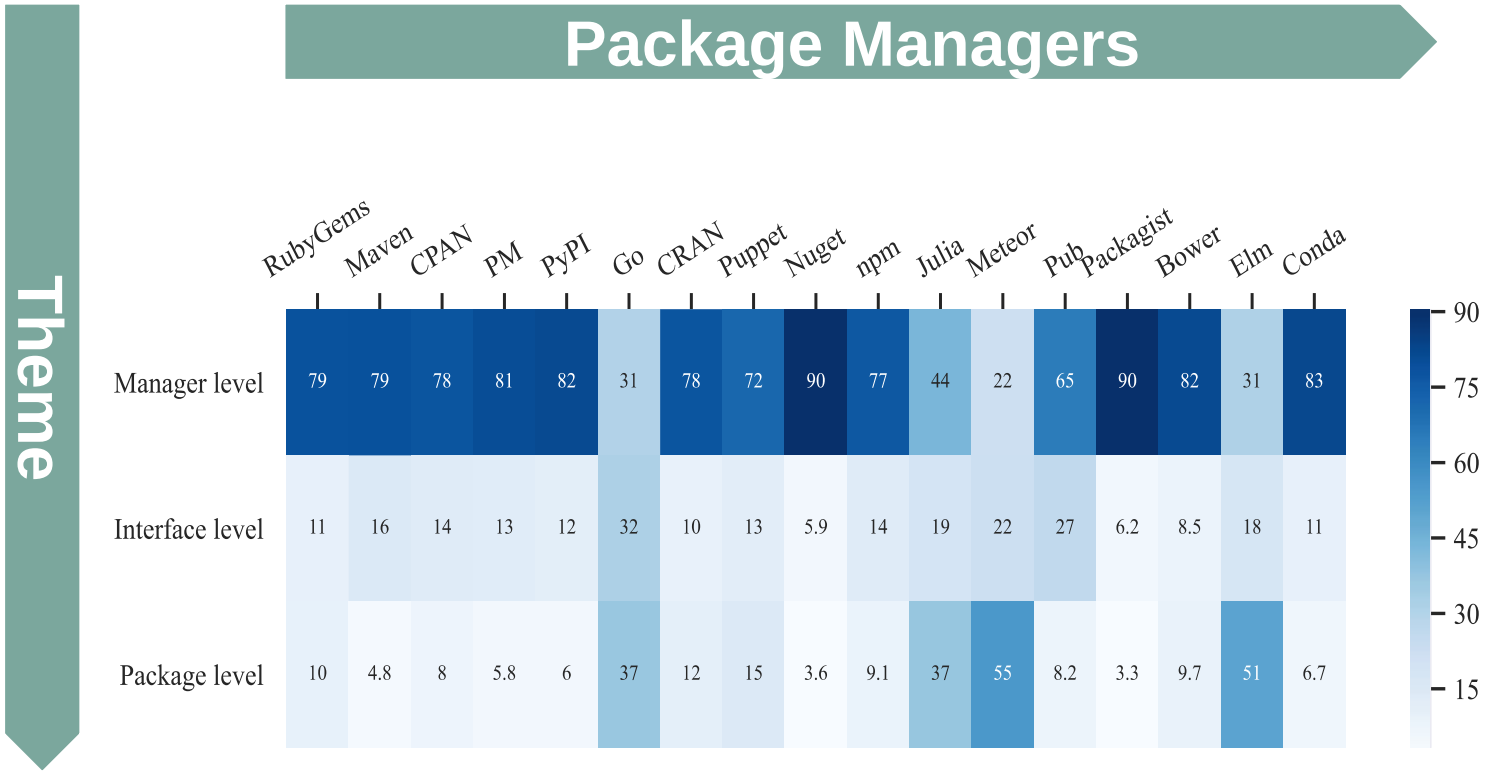
Packagist, Maven, CRAN users have dependency related issues

RubyGems, Puppet users have error related issues

GO, Meteor, and Elm users have package usage and functionalies related issues

SoHeal

Contrasts in Features: Theme [3/5]



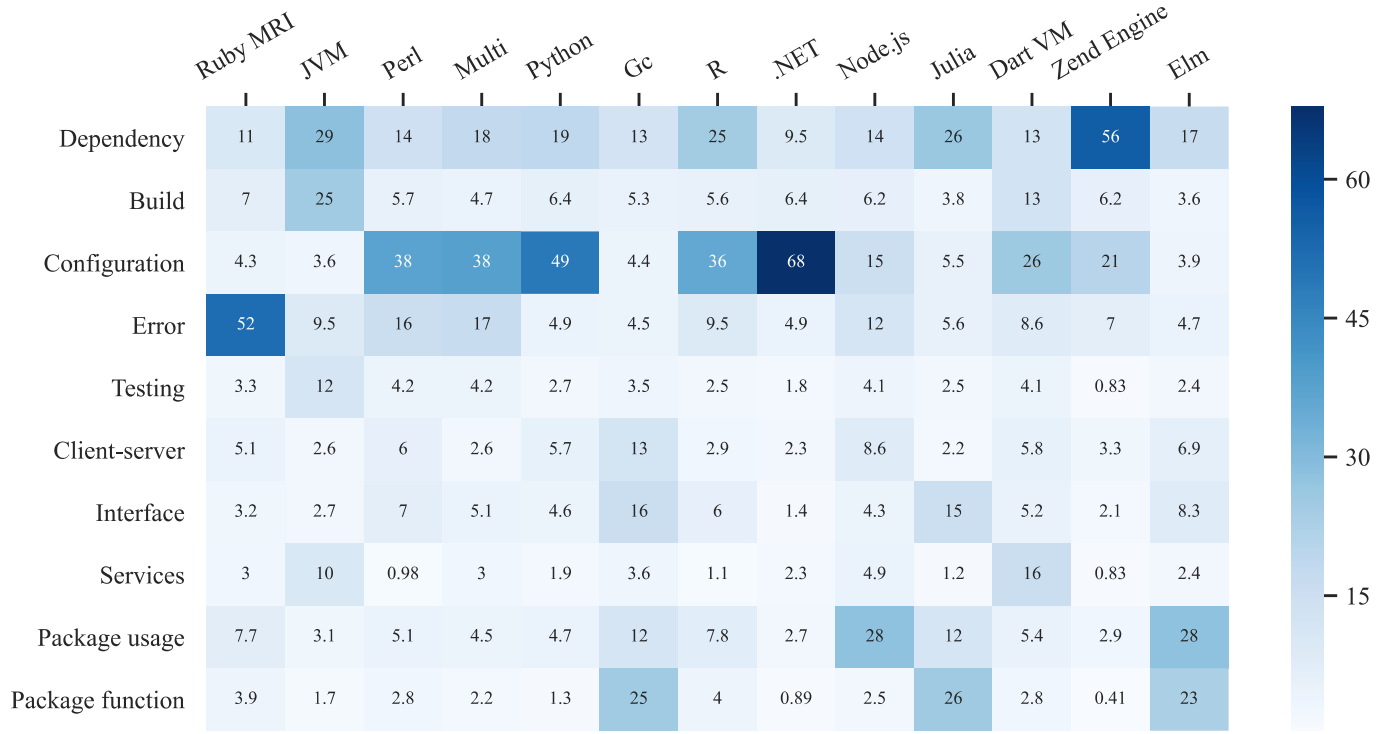
GO, Meteor, and Elm users may face different issues compared to other PM users

SoHeal

Contrasts in Features: Environment [4/5]

Topic

Package Managers Environment



.NET users face configuration related issues.

Ruby users face error related issues.

PhP users face dependency related issues.

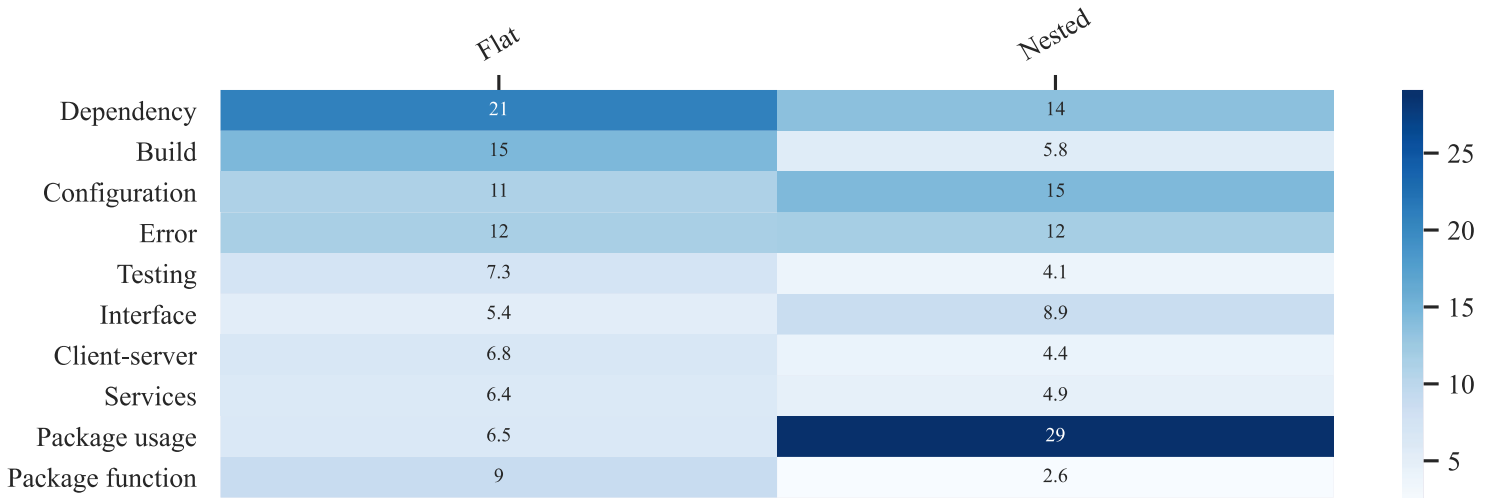
Go users face package function related issues.

SoHeal

Contrasts in Features: Dependency Tree [5/5]

Topics

Package Managers Dependency Tree



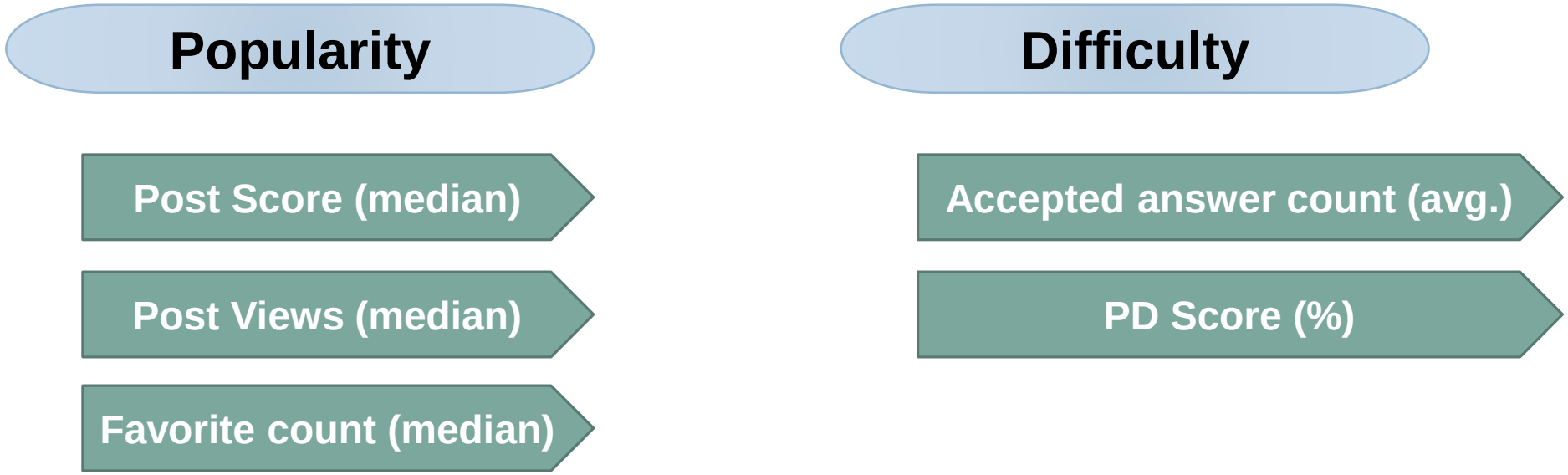
PM with nested dependency tree have high package usage issues

PM with flat dependency tree have high library dependency issues

SoHeal

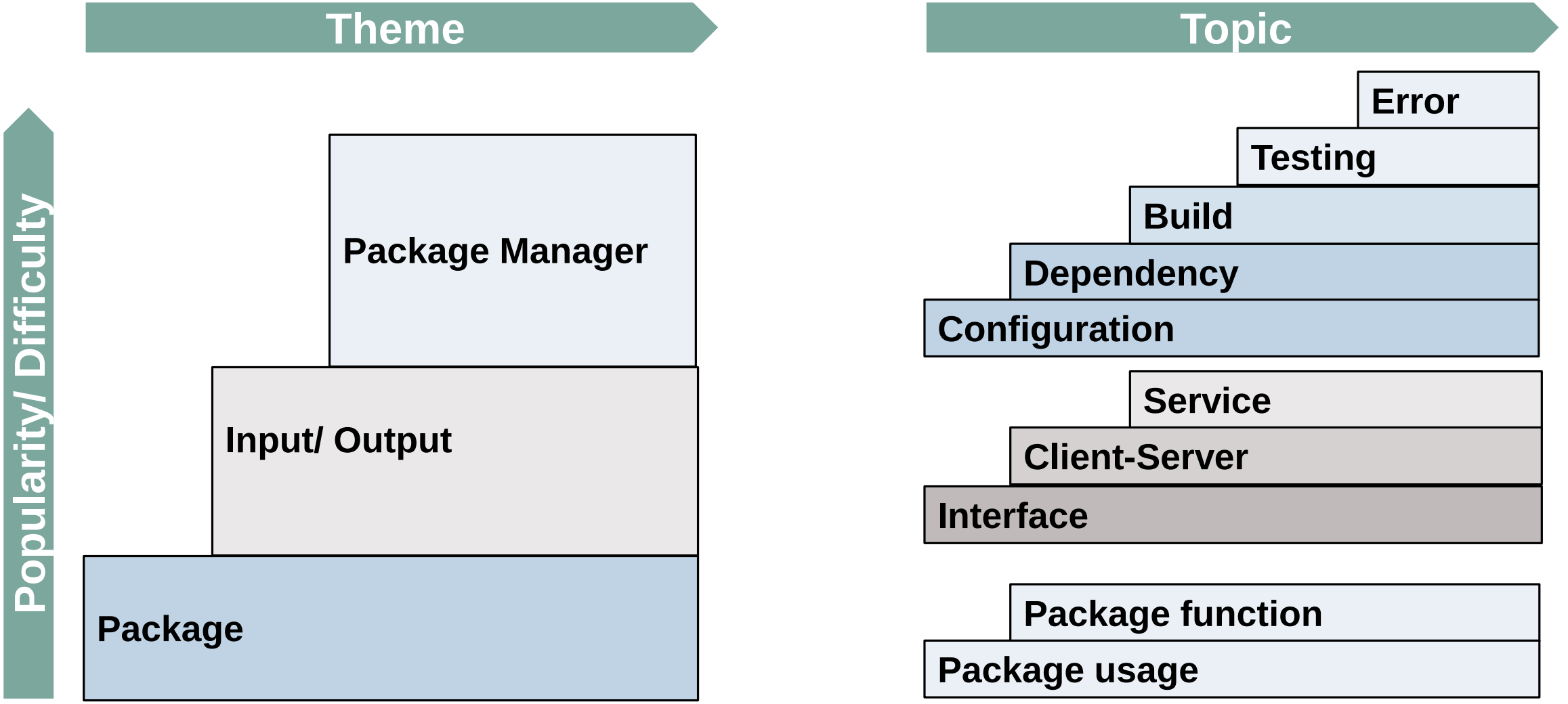
Contrast in Response: Popularity & Difficulty [1/2]

- Characterize PM topics popularity and difficulty based on Yang et al.[1].



1. Yang, Xin-Li, et al. "What security questions do developers ask? a large-scale study of stack overflow posts." *Journal of Computer Science and Technology* 31.5 (2016): 910-924.

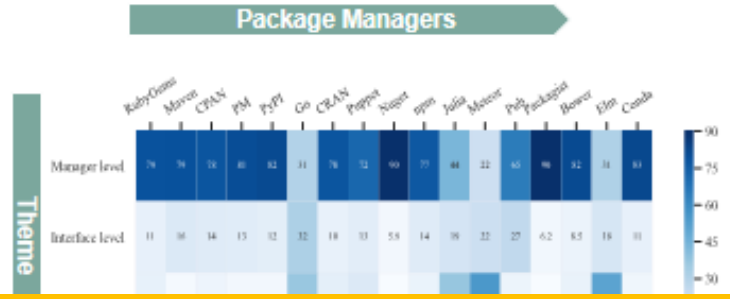
Contrasts in Response: Popularity & Difficulty [2/2]



Implications for Developers

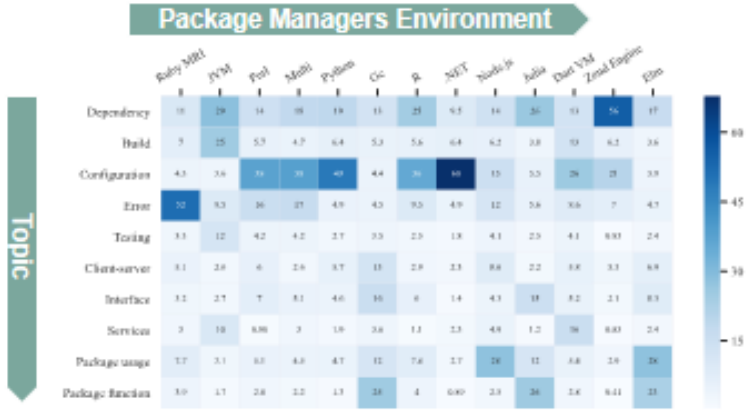
Contrasts in Features: Topics

Contrasts in Features: Theme

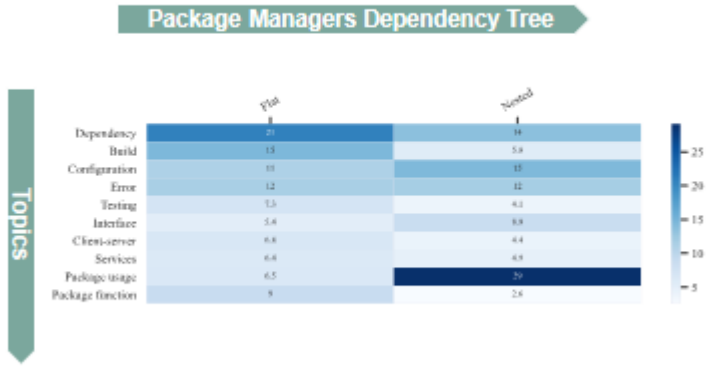


Developers should be conscious that their choice of a PM will impact user experience

Contrasts in Features: Environment



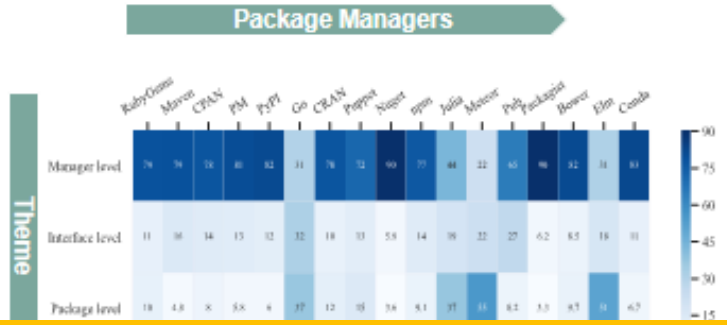
Contrasts in Features: Dependency Tree



Implications for PM Designers

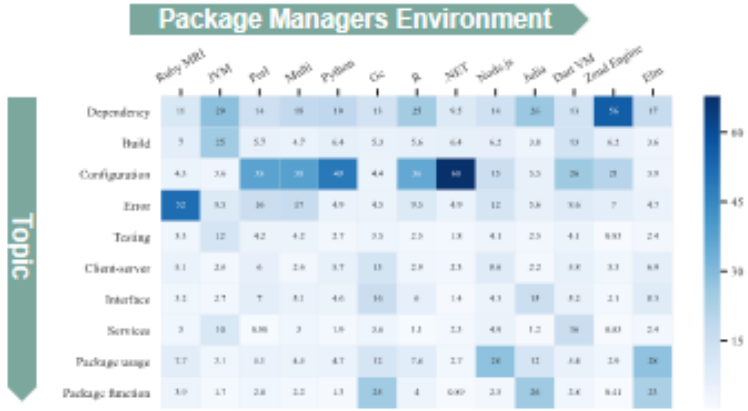
Contrasts in Features: Topics

Contrasts in Features: Theme

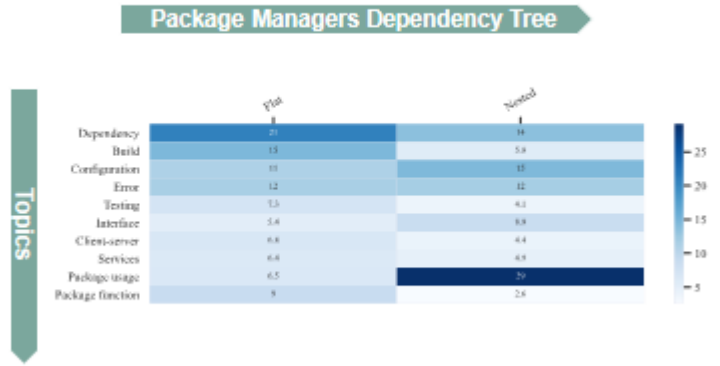


Designers should be proactive on issues frequently encountered by PM users.

Contrasts in Features: Environment



Contrasts in Features: Dependency Tree



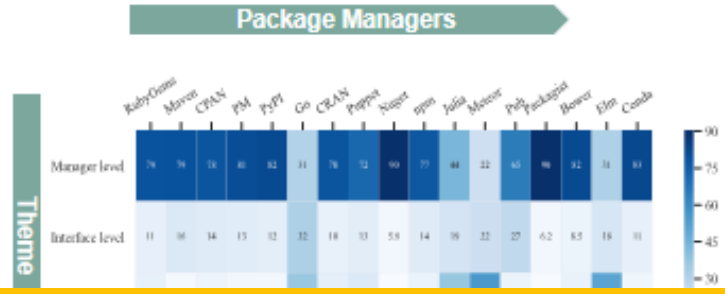
SoHeal

12

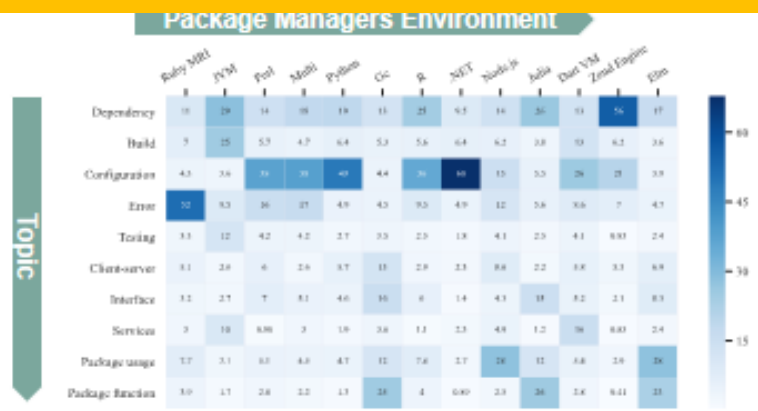
Implications for Researchers

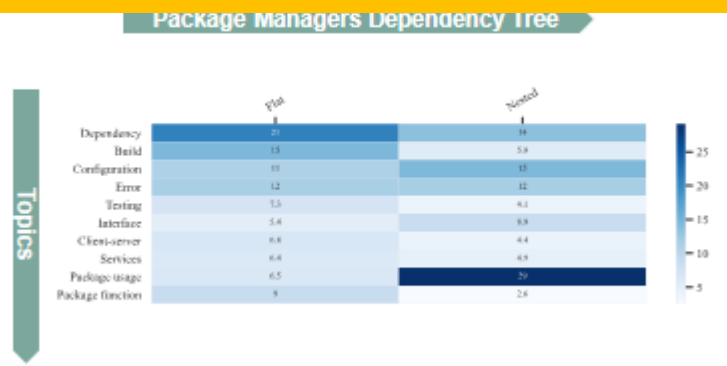
Contrasts in Features: Topics

Contrasts in Features: Theme



Researchers can investigate the trade-offs between design features and potential issues to understand what an ideal PM would look like.





SoHeal

12

Conclusions and Future works

- We explore 16 PMs
  - In terms of features correlate with user experience.
- Developers ask PM questions
  - 10 different topics,
  - 3 themes (Package management, Input/Output, Package).
- The next logical step is further exploration of PM into
  - Underlying causes and
  - Benefits and drawbacks