OpenShift.io Analytics

Actionable insights for the developer Community

Todd A. Mancini
Lead Product Manager

SriKrishna Paparaju
Sr. Principal Software Engineer
Today, developers have plenty of software components to choose ...
These software components grow everyday ...

<table>
<thead>
<tr>
<th>Component</th>
<th>Apr 21</th>
<th>Apr 22</th>
<th>Apr 23</th>
<th>Apr 24</th>
<th>Apr 25</th>
<th>Apr 26</th>
<th>Apr 27</th>
<th>Avg  Growth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clojars (Clojure)</td>
<td>18694</td>
<td>18706</td>
<td>18706</td>
<td>18720</td>
<td>18732</td>
<td>18735</td>
<td>18743</td>
<td>8/day</td>
</tr>
<tr>
<td>CPAN</td>
<td>35138</td>
<td>35140</td>
<td>35142</td>
<td>35147</td>
<td>35152</td>
<td>35158</td>
<td>35163</td>
<td>4/day</td>
</tr>
<tr>
<td>CPAN (search)</td>
<td>35138</td>
<td>35140</td>
<td>35142</td>
<td>35147</td>
<td>35152</td>
<td>35158</td>
<td>35163</td>
<td>4/day</td>
</tr>
<tr>
<td>CRAN (R)</td>
<td>10485</td>
<td>10487</td>
<td>10489</td>
<td>10500</td>
<td>10470</td>
<td>10480</td>
<td>10489</td>
<td>1/day</td>
</tr>
<tr>
<td>Crates.io (Rust)</td>
<td>8941</td>
<td>8952</td>
<td>8967</td>
<td>8974</td>
<td>8998</td>
<td>9015</td>
<td>9028</td>
<td>14/day</td>
</tr>
<tr>
<td>Drupal (php)</td>
<td>37234</td>
<td>37245</td>
<td>37252</td>
<td>37257</td>
<td>37268</td>
<td>37286</td>
<td>37299</td>
<td>11/day</td>
</tr>
<tr>
<td>DUB (dlang)</td>
<td>995</td>
<td>995</td>
<td>997</td>
<td>998</td>
<td>999</td>
<td>1000</td>
<td>1002</td>
<td>1/day</td>
</tr>
<tr>
<td>Gopm (go)</td>
<td>18949</td>
<td>18952</td>
<td>18953</td>
<td>18963</td>
<td>18964</td>
<td>18966</td>
<td>18969</td>
<td>3/day</td>
</tr>
<tr>
<td>Hackage (Haskell)</td>
<td>11213</td>
<td>11217</td>
<td>11220</td>
<td>11226</td>
<td>11237</td>
<td>11243</td>
<td>11247</td>
<td>6/day</td>
</tr>
<tr>
<td>Hex.pm (Elixir/Erlang)</td>
<td>4022</td>
<td>4028</td>
<td>4034</td>
<td>4039</td>
<td>4045</td>
<td>4058</td>
<td>4064</td>
<td>7/day</td>
</tr>
<tr>
<td>Julia</td>
<td>1344</td>
<td>1348</td>
<td>1347</td>
<td>1347</td>
<td>1351</td>
<td>1352</td>
<td>1356</td>
<td>2/day</td>
</tr>
<tr>
<td>LuaRocks (Lua)</td>
<td>1426</td>
<td>1428</td>
<td>1428</td>
<td>1428</td>
<td>1428</td>
<td>1430</td>
<td>1430</td>
<td>1/day</td>
</tr>
<tr>
<td>Maven Central (Java)</td>
<td>184100</td>
<td>184211</td>
<td>184267</td>
<td>184374</td>
<td>184558</td>
<td>184682</td>
<td>184789</td>
<td>115/day</td>
</tr>
<tr>
<td>MELPA (Emacs)</td>
<td>3607</td>
<td>3608</td>
<td>3609</td>
<td>3610</td>
<td>3612</td>
<td>3612</td>
<td>3612</td>
<td>1/day</td>
</tr>
<tr>
<td>npm (node.js)</td>
<td>438794</td>
<td>440564</td>
<td>441582</td>
<td>442183</td>
<td>443024</td>
<td>443826</td>
<td>444419</td>
<td>937/day</td>
</tr>
<tr>
<td>nuget (.NET)</td>
<td>77837</td>
<td>77926</td>
<td>77968</td>
<td>78022</td>
<td>78080</td>
<td>78178</td>
<td>78241</td>
<td>67/day</td>
</tr>
<tr>
<td>Packagist (PHP)</td>
<td>136774</td>
<td>136875</td>
<td>137006</td>
<td>137121</td>
<td>137241</td>
<td>137388</td>
<td>137528</td>
<td>126/day</td>
</tr>
<tr>
<td>Pear (PHP)</td>
<td>602</td>
<td>602</td>
<td>602</td>
<td>602</td>
<td>602</td>
<td>602</td>
<td>602</td>
<td>0/day</td>
</tr>
<tr>
<td>Perl 6 Ecosystem (perl 6)</td>
<td>810</td>
<td>810</td>
<td>811</td>
<td>814</td>
<td>812</td>
<td>813</td>
<td>815</td>
<td>1/day</td>
</tr>
<tr>
<td>PyPi</td>
<td>106353</td>
<td>106427</td>
<td>106490</td>
<td>106590</td>
<td>106636</td>
<td>106726</td>
<td>106822</td>
<td>78/day</td>
</tr>
<tr>
<td>Rubygems.org</td>
<td>131348</td>
<td>131377</td>
<td>131404</td>
<td>131431</td>
<td>131471</td>
<td>131515</td>
<td>131552</td>
<td>34/day</td>
</tr>
</tbody>
</table>
How can a developer choose from so many software components?
Maybe you the developer, choosing a software component that others in your organization are not currently using...
This is not a one time pain

How does a developer keep up with newer versions and the ever growing list of software components?
Enter OpenShift.io Analytics

Actionable insights from self learning, ecosystem agnostic analytics platform
OpenShift.io Analytics

- Actionable insights
- Architecture
- Analytics behind actionable insights
- Integrations
- Roadmap
Actionable Insights:

Current use cases
Actionable Insights:
For a software component

“I need help.”

OpenShift.io Analytics

“T’m considering this software component.”

io.vertx:vertx-jdbc-client:3.1.0
Actionable Insights:
For an application stack

"I need help."

OpenShift.io Analytics
- vertx-jdbc-client:3.1.0
- hsqldb:2.3.3

"I’m looking at this stack."
Actionable Insights:

Demo
Architecture

- Cloud Native with CI, CD
- Micro service based architecture built on OpenShift
- With failover between two OSD clusters
- Zabbix monitoring, triggers failover
Architecture
A typical Big Data architecture

Data gathering
Data normalization

Analytics
Serving Layer

Presentation
Integrations
Architecture
Data gathering, normalization

Data Sources

Gather and Normalize

S3
RDS
SQS
Analytics behind actionable insights

Themes

- Intent of the User
- Evolve, Self learn
- OSIO Analytics Platform
- Ecosystem agnostic
- Adapt to Usage patterns
Analytics behind actionable insights

Themes: Adapt to usage patterns

Past user activity (eg: 1000)

SciKit learn

K-Means clustering

User profiles
Analytics behind actionable insights
Themes: How these user profiles are created?

Developer asking for Insights

Match to a User profile

User profile 1
User profile 2
User profile 3
Analytics behind actionable insights

Themes: Intent of the user

- Populate Probabilistic Graphs
- Step1: Populate a Reference architecture (Intent)
- Step2: Get the category a software component belong
- Step3: Assign probability based on data collected
- Step4: Create Probabilistic Graph models
Analytics behind actionable insights

*Step 1: Populate a Reference architecture (Intent)*

- E-Commerce
  - UI
  - NOSQL
  - Serving
Analytics behind actionable insights

Step 2: Get the category a software component belongs to.

- UI
  - Angular
- NOSQL
  - Cassandra
- Serving
  - Flask
  - NodeJS
Analytics behind actionable insights

Step 3: Assign probability based on data collected

E-Commerce

- UI
  - Angular: Probability of 1
- NOSQL
  - Cassandra: Probability of 1
- Serving
  - Flask: Probability of 1
  - NodeJS: Probability of 1
Analytics behind actionable insights

Step 4: Create Probabilistic Graph models (Bayesian network)
Analytics behind actionable insights

Step 4: Create Probabilistic Graph models (Bayesian network)

Intent Match → E-Commerce

- UI
  - Angular
  - ExpressJS

- NOSQL
  - MongoDB
  - Cassandra

- Serving
  - Flask
  - NodeJS

- ExpressJS
  - MongoDB
  - Flask
Analytics behind actionable insights

All themes together

Developer asking for Insights

3 models for user profiles
+ 3 models for graphs
Analytics behind actionable insights

Deployment view (Model training)

1. Trigger with Boto
2. Read data
3. Store the model
Analytics behind actionable insights
Deployment view (Model scoring)

1. Ask for Recommendations
2. Load the model if not loaded
3. Serve Recommendations

S3
OSIO Analytics

{dev}
Analytics behind actionable insights

Demo
### Demo flow

**Themes:** Compare two users (Software components)

<table>
<thead>
<tr>
<th></th>
<th>Pete</th>
<th>Kivy (UI framework)</th>
<th>Kishna</th>
<th>Sqlalchemy (ORM mapper)</th>
<th>Concurrent.futures (concurrency)</th>
<th>Kivy (UI framework)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Interaction --->
### Demo flow

**Themes:** Compare two users (Application stacks)

<table>
<thead>
<tr>
<th>Pete</th>
<th>Pandas</th>
<th>Scikit-learn</th>
<th>Flask</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Kishna</th>
<th>Pandas</th>
<th>Bumpy</th>
<th>Scipy</th>
<th>Pandas</th>
<th>scikit-learn</th>
<th>Flask</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Integrations

- Eclipse Che thru Language server protocol
- Jenkins
- SonarQube
- Free form query
- Integrate with REST
- Ansible
Integrations

SonarQube

SonarQube

OSIO Plugin

OSIO Analytics
Integrations
Free form query

Demo
RoadMap

- Expand the set of data sources
- Add more integrations
- Expand use cases for actionable insights
RoadMap

- Expand use cases for actionable insights

Future Development

- Cloud services
- Repository
- Pull Requests

Existing

- Software component
- Usage patterns
- Intent
We developers, can handle this
Thank you all .. team members

Arunkumar Srisailapthi
Bargava Subramanian
Bohuslav Kabrda
Frido Pokorny
George Acton
Geetika Batra
Harjindersingh Mistry
Jaivardhan Kumar
Jiri Popelka
Jyasveer Gotta
Manjunath Sindagi
Matthias Lubken

Michal Srb
Mitesh Patel
Pavel Kajaba
Saket Choudhary
Saleem Ansari
Samuzzal Choudhury
Sarah Masud
Shubheksha Jalan
Tomas Hrcka
Tuhin Sharma
Opening to the Community...

https://github.com/fabric8-analytics
THANK YOU

plus.google.com/+RedHat
linkedin.com/company/red-hat
youtube.com/user/RedHatVideos
facebook.com/redhatinc
twitter.com/RedHatNews