StackTraces – Problems

R Analysis document

Boris Baldassari – Castalia Solutions

Contents

Introduction

About this dataset

The Automated Error Reporting (AERI) system retrieves information about exceptions. It is installed by default in the Eclipse IDE and has helped hundreds of projects better support their users and resolve bugs.

This dataset is a dump of all records over a couple of years, with useful information about the exceptions and environment.

- **Generated date**: Mon Feb 12 08:52:00 2018
- **First date**: 2014-09-12 08:47:00
- **Last date**: 2016-12-13 14:55:09
- **Number of problems**: 50872
- **Number of attributes**: 15

Terminology

- **Incidents** When an exception occurs and is trapped by the AERI system, it constitutes an incident (or error report). An incident can be reported by several different people, can be reported multiple times, and can be linked to different environments.
- **Problems** As soon as an error report arrives on the server, it will be analyzed and subsequently assigned to one or more problems. A problem thus represents a set of (similar) error reports which usually have the same root cause – for example a bug in your software. (Extract from the AERI system documentation)

This dataset targets only the Problems of the AERI dataset. There is another dedicated document for the Incidents.

Privacy concerns

We value privacy and intend to make everything we can to prevent misuse of the dataset. If you think we failed somewhere in the process, please let us know so we can do better.

The AERI system itself doesn't gather much private information, and takes a great care of it. Ths dataset goes a step further and removes all identifiable information.

- There is **no email address** in this dataset, **nor any UUID**.
- People not willing to share their traces to the AERI system can tick the private option. This choice has been respected, and all classes that do not belong to public hierarchy have been hidden thanks to an anonymisation mechanism.

The anonymisation technique used basically encrypts information and then throws away the private key. Please refer to the documentation published on github for more details.

About this document

This document is a R Markdown document and is composed of both text (like this one) and dynamically computed information (mostly in the Anaysis section below) executed on the data itself. This ensures that the documentation is always synchronised with the data, and serves as a test suite for the dataset.

Structure of data

The plugin collects a lot of useful information. We only use a subset of it, as required by research interest and privacy protection concerns.

The Problems dataset comes in two flavours: `All problems`, in JSON format, and `Problems extract`, in CSV format.

All problems (JSON)

All problems is the most complete dataset, with all attributes, stacktraces and bundles. Since the stacktraces and bundles structures are too complex for CSV, only the JSON export contains them. The dataset comes as a quite large compressed archive, with one JSON file per problem. This represents a total of 50872 files (problems).

The structure of a problem file is examplified below:

```
{
  "summary": "NoStackTrace in RedeliveryErrorHandler.logFailedDelivery",
  "kind": "NORMAL",
  "v1status",
  "osgiArch": "x86_64",
  "osgiOs": "MacOSX",
  "osgiOsVersion": "10.9.4",
  "osgiWs": "cocoa",
  "createdOn": "2014-09-14T05:39:21.554Z",
  "modifiedOn": "2014-09-14T05:39:21.554Z",
  "savedOn": "2016-05-23T07:22:10.479Z",
  "eclipseBuildId": "4.4.0.I20140606-1215",
  "eclipseProduct": "org.eclipse.epp.package.standard.product",
  "javaRuntimeVersion": "1.8.0-b132",
  "numberOfIncidents": 0,
  "numberOfReporters": 74,
  "products": [
    { product },
    { product }
  ],
  "bundles": [
    { bundle },
    { bundle }
  ],
  "stacktraces": [
    [ "stacktrace for incident" ],
    [ "stacktrace for cause" ],
    [ "stacktrace for exception" ]
  ]
}
```

The structure used in the mongodb for stacktraces has been kept as is: it is composed of fields with all information relevant to each line of the stacktrace. Each stacktrace is an array of objects as shown below:

```
[
  {
    "cN": "sun.net.www.http.HttpClient",
    "mN": "parseHTTPHeader",
    "fN": "HttpClient.java",
    "lN": 786,
  }
]
```

Bundles have the following format:

```
{
  "bundleFrequency": 1,
  "bundleName": "org.eclipse.egit.core",
```

```
  "bundleVersion": "4.1.1.201511131810-r"
},
```

Products have the following format:

```
{
  "buildId": "4.5.2.M20160212-1500",
  "frequency": 3,
  "productId": "org.eclipse.epp.package.jee.product"
}
```

Problems extract (CSV)

The **Problems extract** CSV dataset provides the same information as the full JSON dataset, excluding complex structures that cannot be easily formatted in CSV: stacktraces, bundles, products.

Attributes are: `summary`, `numberOfReporters`, `numberOfIncidents`, `v1status`, `kind`, `createdOn`, `modifiedOn`, `savedOn`, `osgiArch`, `osgiOs`, `osgiOsVersion`, `osgiWs`, `eclipseBuildId`, `eclipseProduct`, `javaRuntimeVersion`.

Examples are provided at the end of this file to demonstrate how to use it in R.

Attributes

Summary

- Description: A short text summarising the error.
- Type: String

Number of reporters

- Description: The number of people who reported this incident or problem.
- Type: integer

Statistical summary:

- Range [0 : 57748]
- 1st Quartile 1
- Median 2
- Mean 25.87223
- 3rd Quartile 6

Number of incidents

- Description: The number of times this problem was identified in incidents.
- Type: Integer

Statistical summary:

- Range [0 : 24934]
- 1st Quartile 0
- Median 0
- Mean 11.32645
- 3rd Quartile 3

- NAs 1

V1 Status

- Description: The status of the problem attached to the error report.
- Type: Factors

The possible values found in the dataset for this attributes are:

- CONFIRMED (count: 522)
- DUPLICATE (count: 380)
- FIXED (count: 1048)
- INVALID (count: 585)
- NEEDINFO (count: 110)
- UNCONFIRMED (count: 48227)

Note: The name of this attribute in the original file is `v1status`.

Kind

- Description: The type of error recorded, as identified by the AERI system.
- Type: Factors

The possible values found in the dataset for this attributes are:

- FREEZE (count: 4198)
- NORMAL (count: 29305)
- OOME (count: 2472)
- SOE (count: 1205)
- THIRD (count: 13293)
- THIRD_FREEZE (count: 399)

Notes

There are different kinds of incidents described in the official documentation:

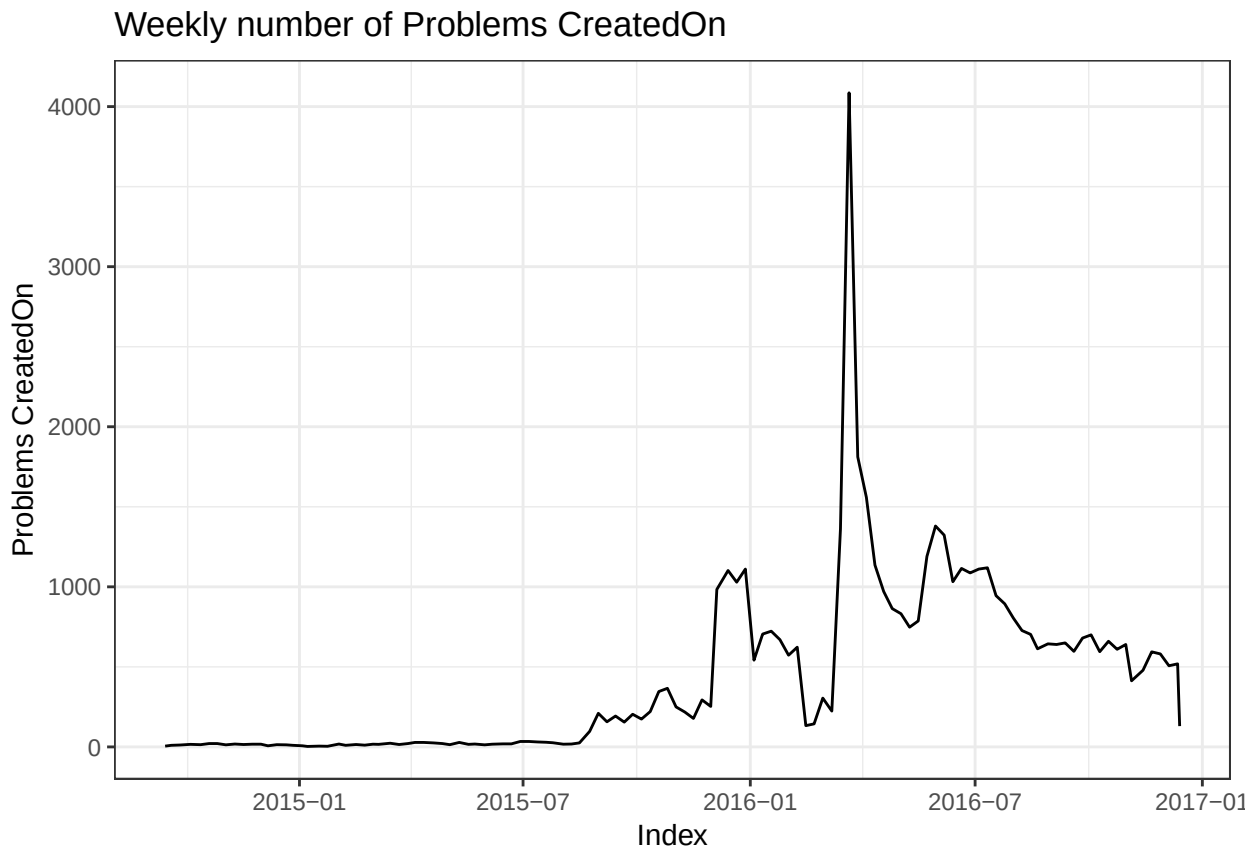
- Normal Error: Normal errors are all exceptions that were reported by a client but that are not of kind defined below. Common examples of a normal error are a `NullPointerException` or `IllegalArgumentException`.
- An `OutOfMemoryError` is a special kind of exception. Unlike for normal errors, the stack frame (implicitly) throwing the exception is only sometimes indicative of the root cause of the problem.
- A `StackOverflowError` is a special kind of exception, whose unique characteristic is a repeating pattern of stack frames near the top of the stack trace.
- UI Freeze: A UI freeze is caused by a long-running operation or even a deadlock on the UI thread.
- Third-Party Error: Third-party errors are reports that were received by the Codetrails Error Analytics Server, which deemed neither the configured projects nor their dependencies at fault.
- Third-Party UI Freeze: Third-Party UI Freezes are UI freezes that were received by the Codetrails Error Analytics Server, which deemed neither the configured projects nor their dependencies at fault.

Created On

- Description: The time of first appearance of the problem in an incident.
- Type: Date (ISO8601)

Dates range from 2014-09-12 08:47:00 to 2016-12-13 14:55:09.

```
xts.createdOn <- as.xts(apply.weekly(myp.xts.createdOn, sum))
autoplot(xts.createdOn, geom='line') +
  theme_bw() + ylab("Problems CreatedOn") + ggtitle("Weekly number of Problems CreatedOn")
```

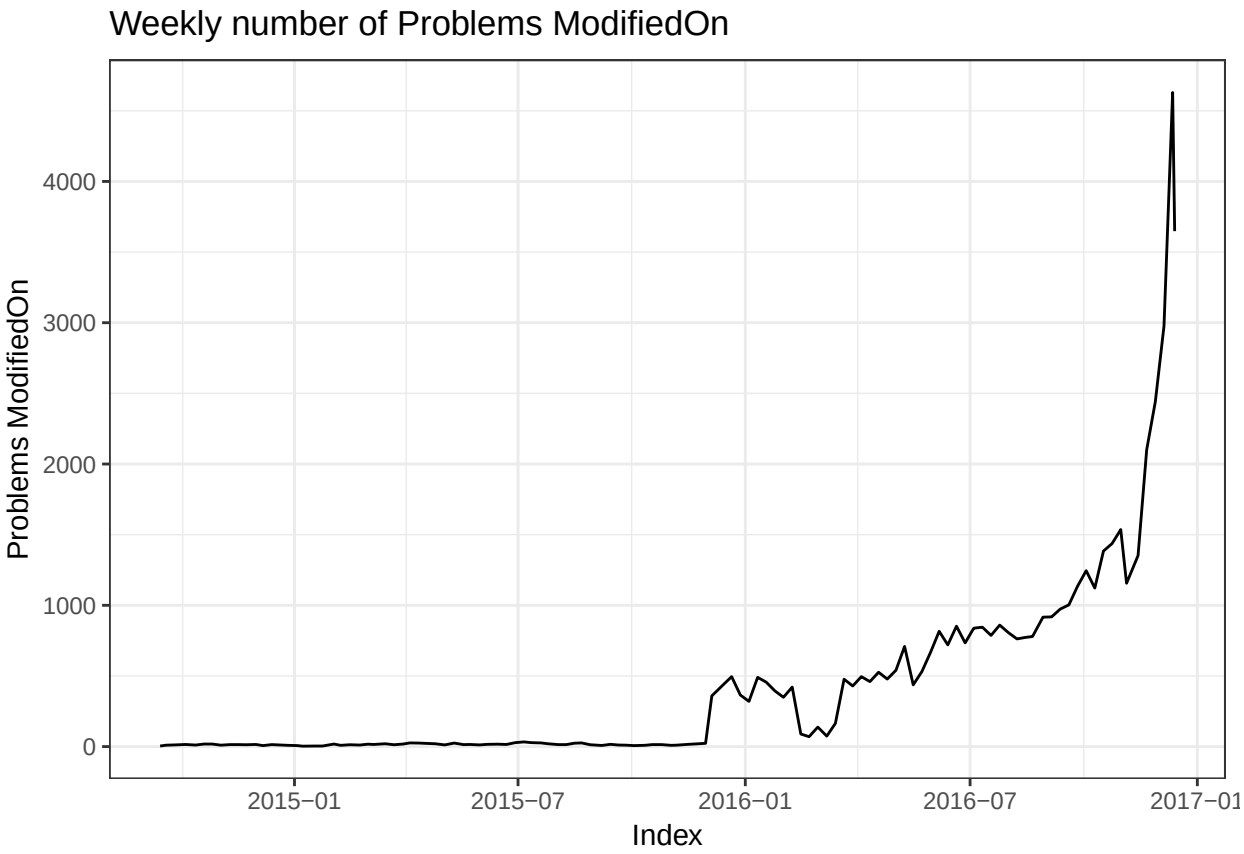


Modified On

- Description: The time of last update of the problem in an incident.
- Type: Date (ISO8601)

Dates range from 2014-09-12 17:41:11 to 2016-12-13 15:17:45.

```
xts.modifiedOn <- as.xts(apply.weekly(myp.xts.modifiedOn, sum))
autoplot(xts.modifiedOn, geom='line') +
  theme_bw() + ylab("Problems ModifiedOn") + ggtitle("Weekly number of Problems ModifiedOn")
```

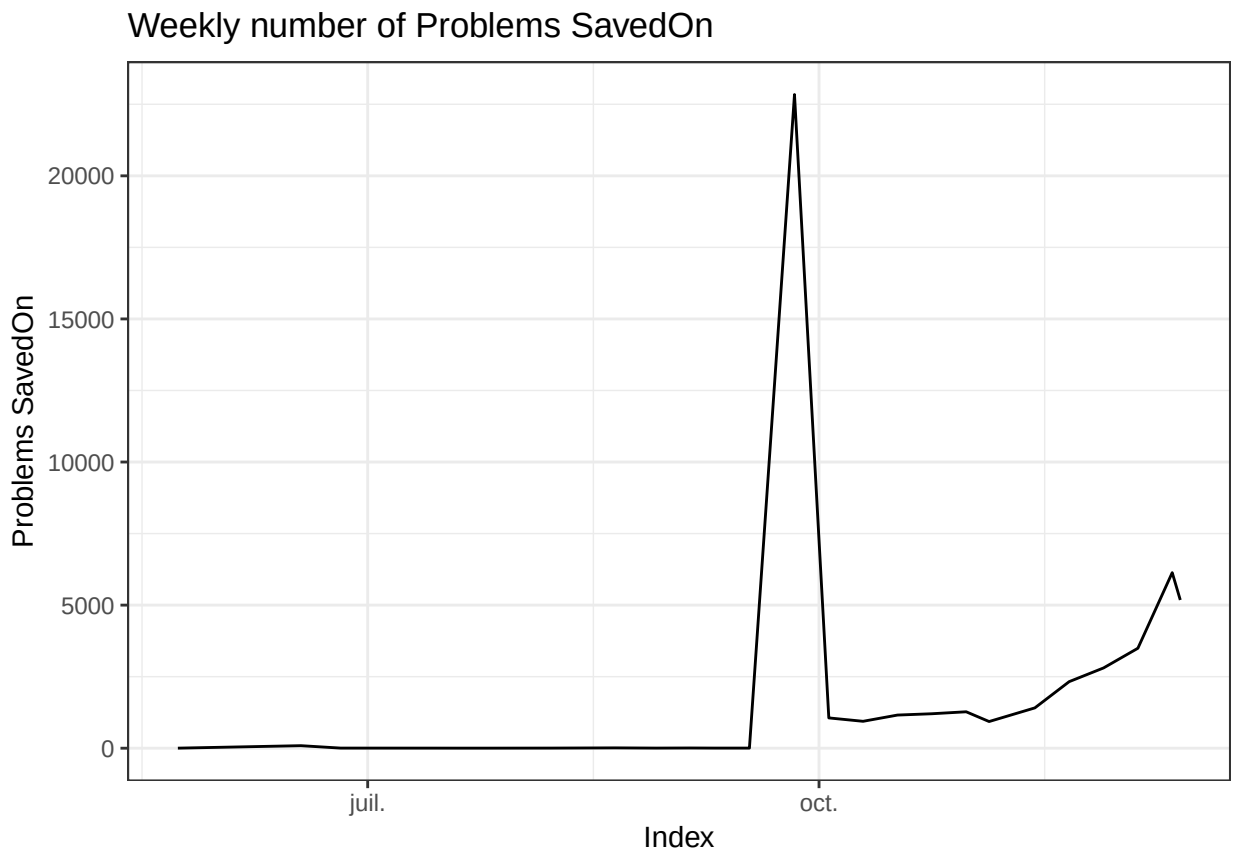


Saved On

- Description: The time of last save of the problem.
- Type: Date (ISO8601)

Dates range from 2016-05-23 07:22:10 to 2016-12-13 15:17:45.

```
xts.savedOn <- as.xts(apply.weekly(myp.xts.savedOn, sum))
autoplot(xts.savedOn, geom='line') +
  theme_bw() + ylab("Problems SavedOn") + ggtitle("Weekly number of Problems SavedOn")
```



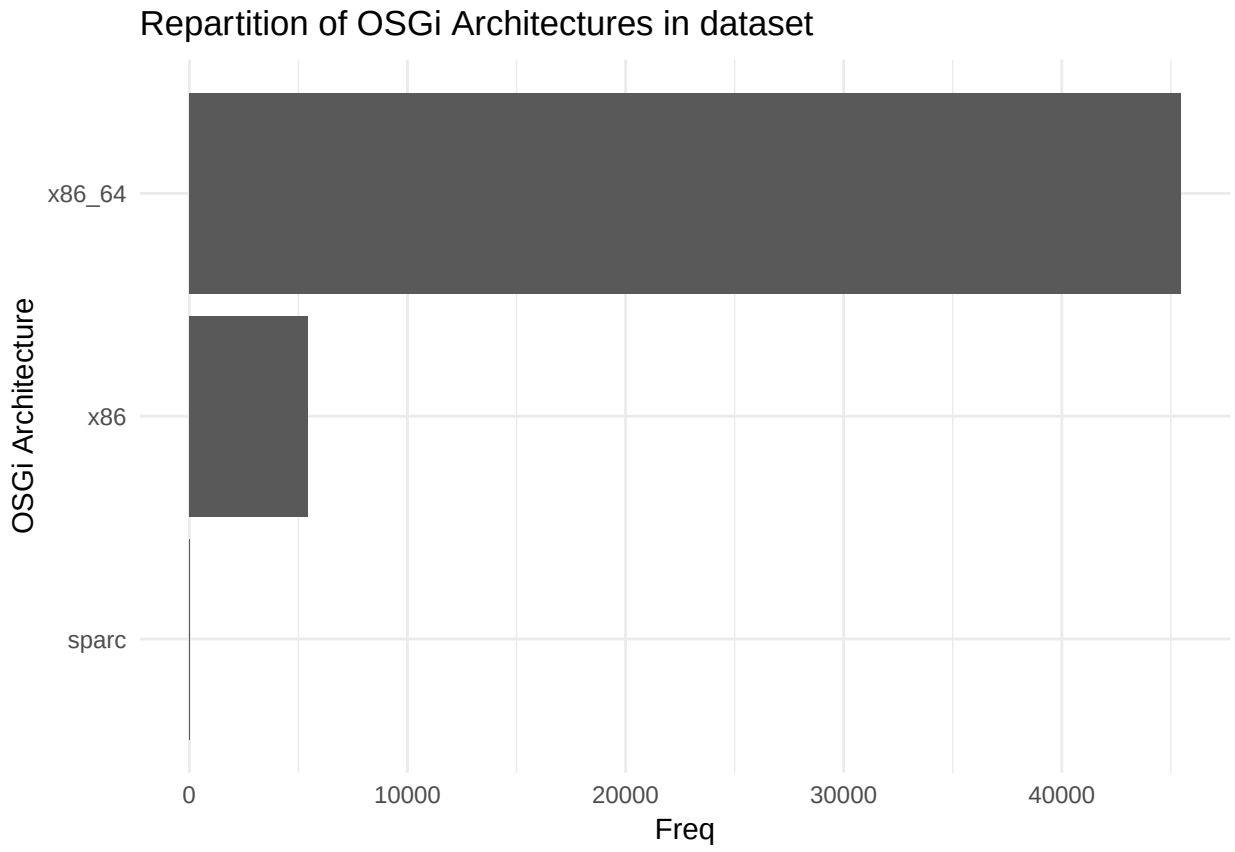
OSGi Architecture

- Description: The architecture of the host, as specified in the OSGi bundle definition.
- Type: Factors

Possible values found in the dataset for this attribute are:

- sparc (count: 2)
- x86 (count: 5413)
- x86_64 (count: 45457)

Repartition of architectures:



OSGi OS

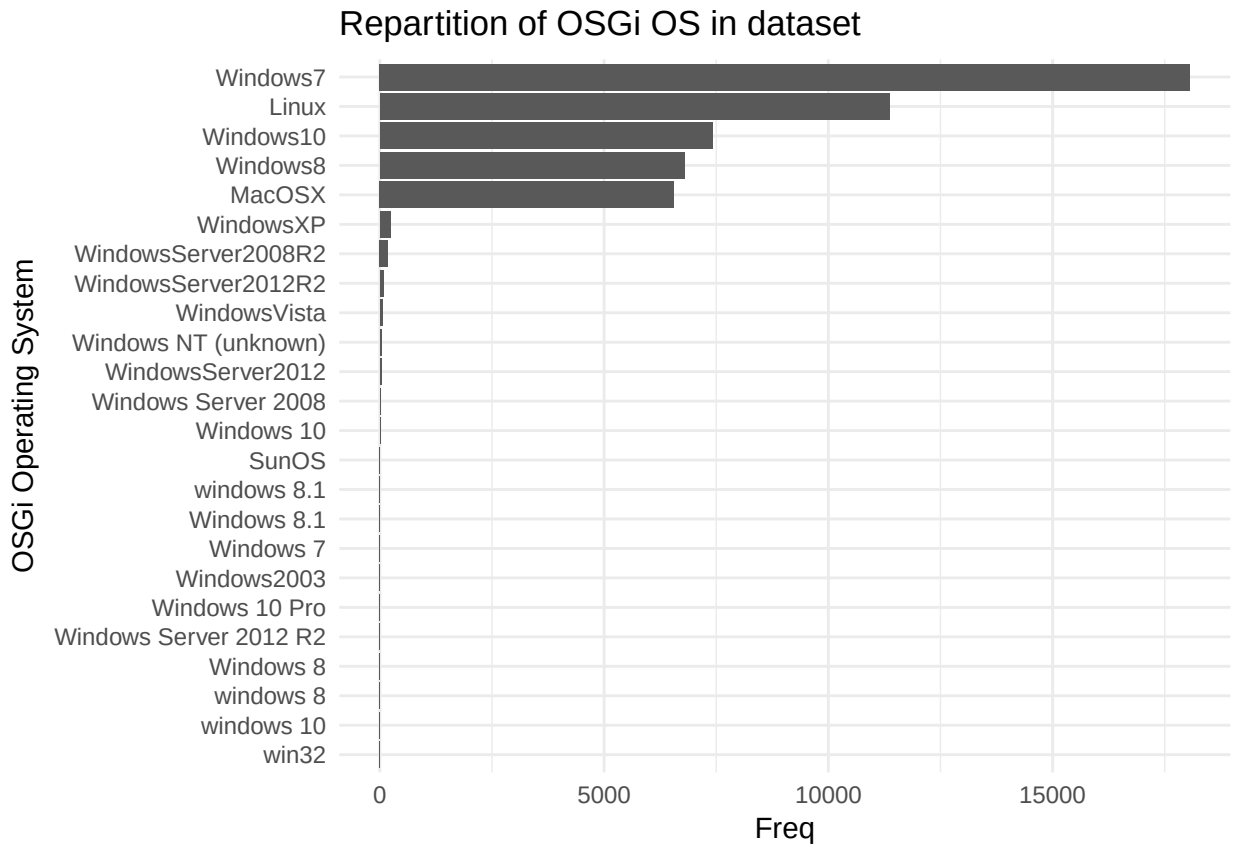
- Description: The host operating system, as reported in OSGi.
- Type: Factors

The possible values found in the dataset for this attributes are:

- Windows7 (count: 18060)
- Linux (count: 11360)
- Windows10 (count: 7420)
- Windows8 (count: 6788)
- MacOSX (count: 6562)
- WindowsXP (count: 236)
- WindowsServer2008R2 (count: 183)
- WindowsServer2012R2 (count: 77)
- WindowsVista (count: 62)
- Windows NT (unknown) (count: 42)
- WindowsServer2012 (count: 32)
- Windows Server 2008 (count: 14)
- Windows 10 (count: 10)
- SunOS (count: 6)
- windows 8.1 (count: 5)
- Windows 8.1 (count: 4)
- Windows 10 Pro (count: 2)
- Windows2003 (count: 2)
- Windows 7 (count: 2)

- win32 (count: 1)
- windows 10 (count: 1)
- windows 8 (count: 1)
- Windows 8 (count: 1)
- Windows Server 2012 R2 (count: 1)

Visualisation of the various operating systems used in the dataset:



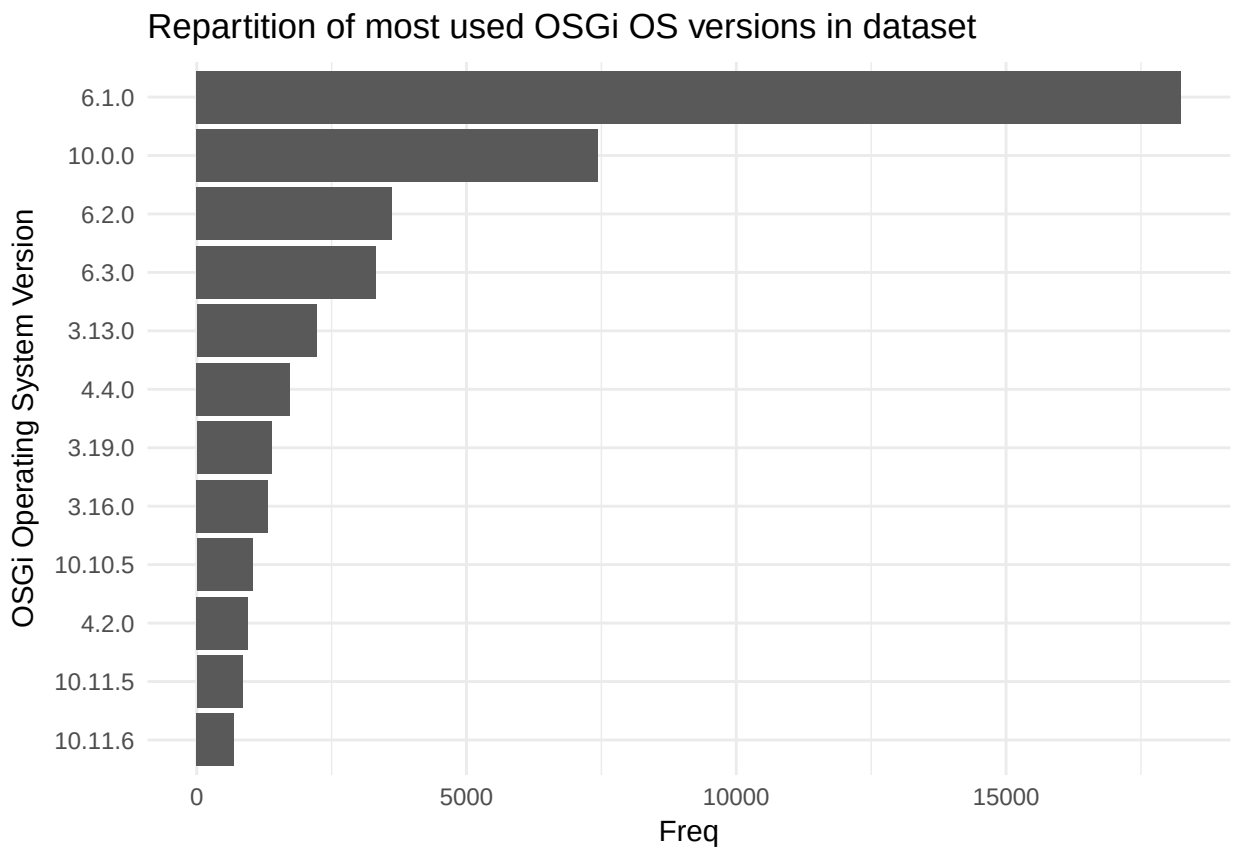
OSGi OS Version

- Description: The host operating system version, as reported in OSGi.
- Type: Factors

The possible values found in the dataset for this attributes are:

- 6.1.0 (count: 18246)
- 10.0.0 (count: 7437)
- 6.2.0 (count: 3623)
- 6.3.0 (count: 3325)
- 3.13.0 (count: 2219)
- 4.4.0 (count: 1728)
- 3.19.0 (count: 1388)
- 3.16.0 (count: 1316)
- 10.10.5 (count: 1035)
- 4.2.0 (count: 948)
- 10.11.5 (count: 850)
- 10.11.6 (count: 690)

Visualisation of the various operating system versions used in the dataset:



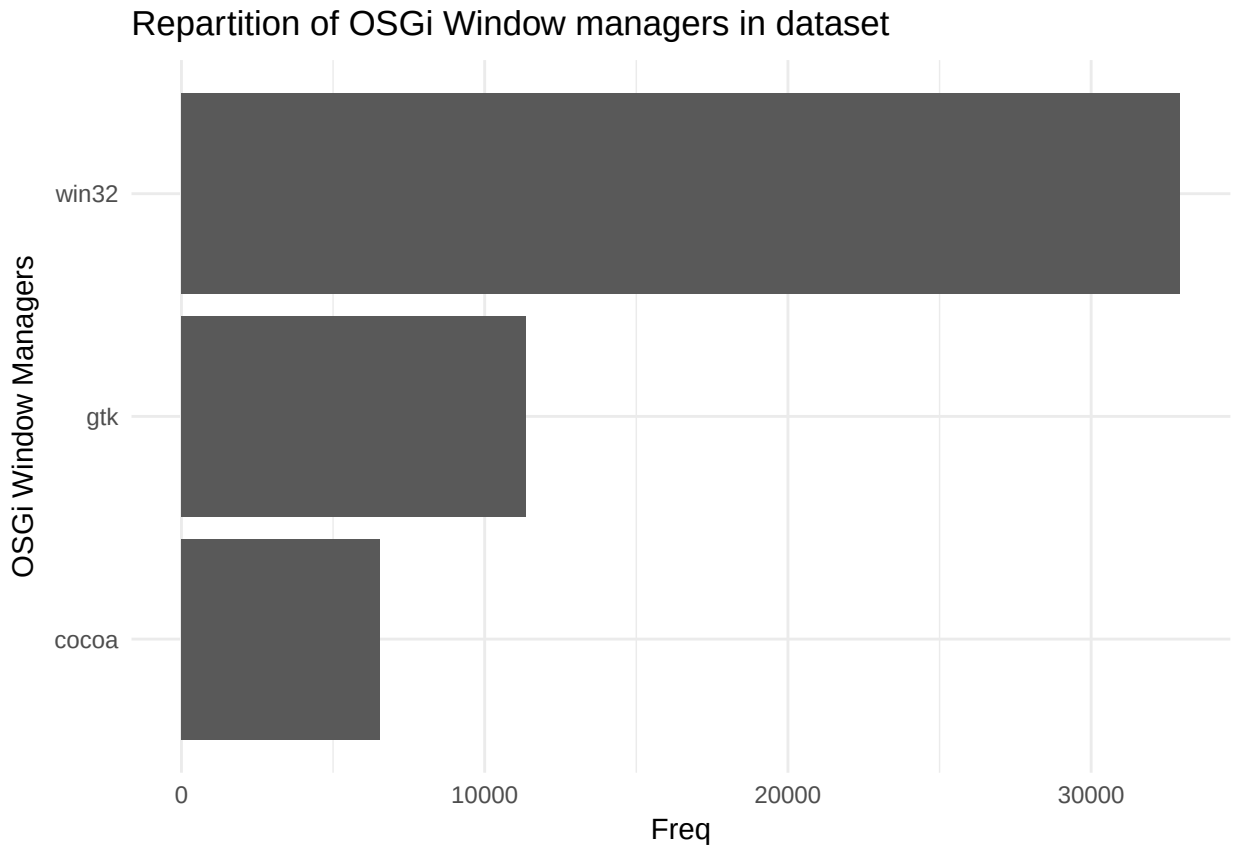
OSGi Window Manager

- Description: The Window Manager used by the host, as reported in OSGi.
- Type: Factors

The possible values found in the dataset for this attributes are:

- win32 (count: 32944)
- gtk (count: 11366)
- cocoa (count: 6562)

Visualisation of the various Window managers used in the dataset:

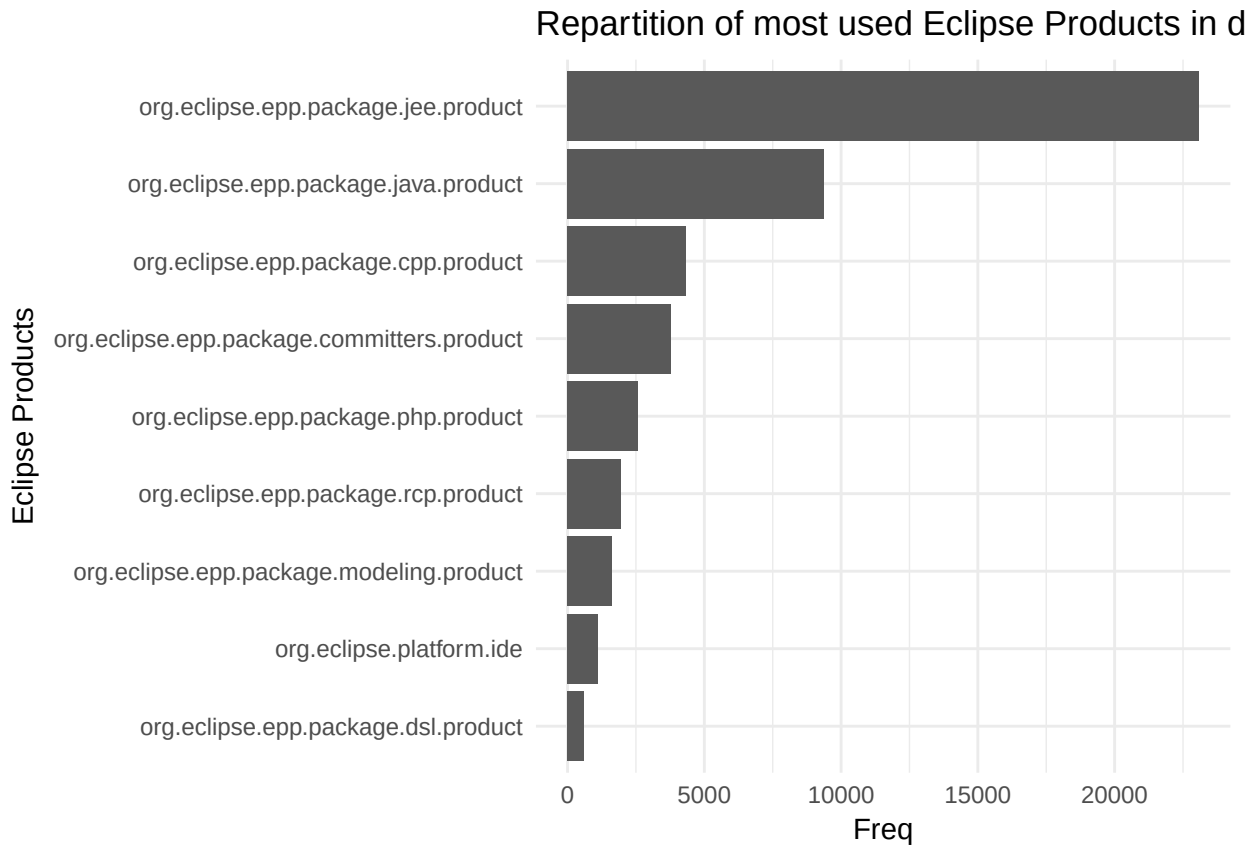


Eclipse Product

- Description: The Eclipse product impacted by the exception.
- Type: Factors

There are 31 different values found in the dataset for this attribute. The following table and bar plot only display the values with more than 500 occurrences:

- org.eclipse.epp.package.jee.product (count: 23078)
- org.eclipse.epp.package.java.product (count: 9374)
- org.eclipse.epp.package.cpp.product (count: 4343)
- org.eclipse.epp.package.committers.product (count: 3792)
- org.eclipse.epp.package.php.product (count: 2570)
- org.eclipse.epp.package.rcp.product (count: 1947)
- org.eclipse.epp.package.modeling.product (count: 1635)
- org.eclipse.platform.ide (count: 1124)
- org.eclipse.epp.package.dsl.product (count: 592)

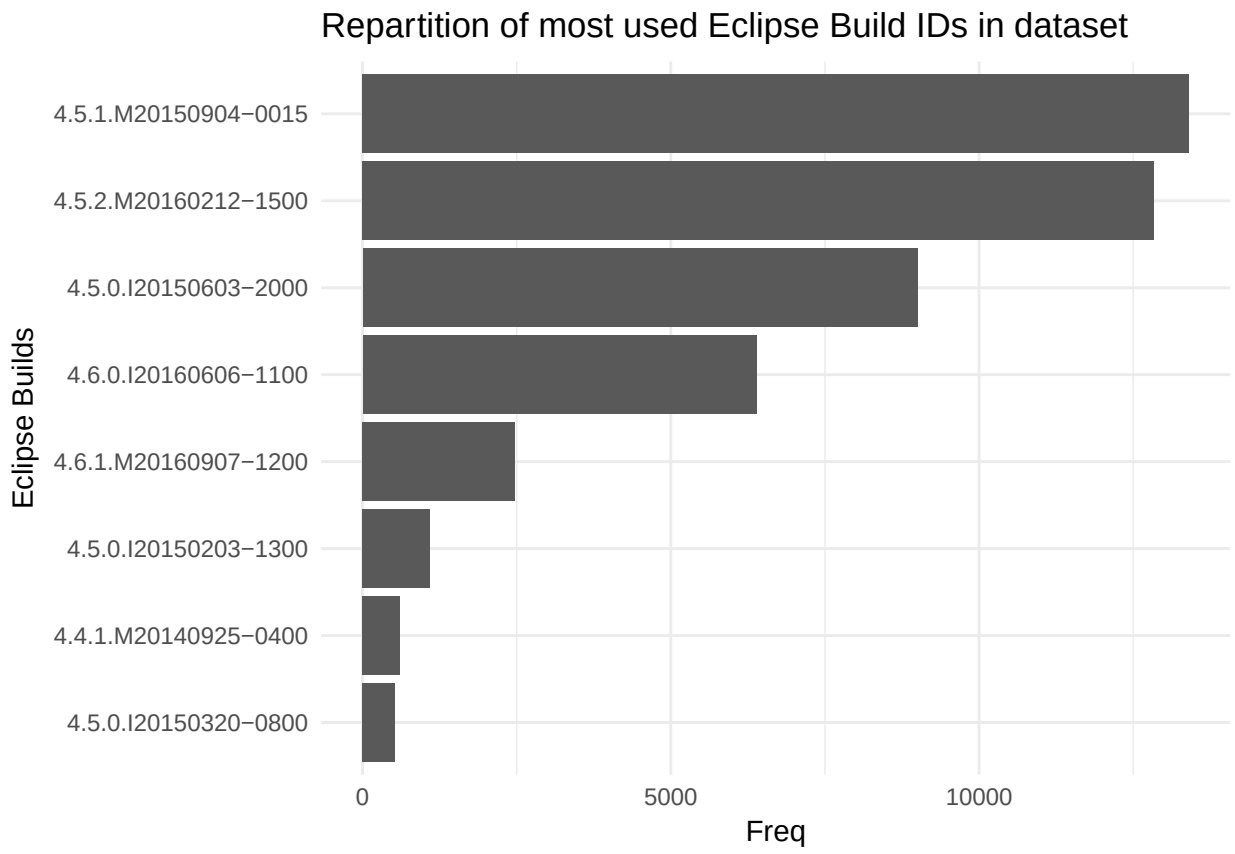


Eclipse Build ID

- Description: The Build ID of the Eclipse instance running when the exception occurred.
- Type: Factors

There are 146 different values found in the dataset for this attribute. The following table and bar plot only display the values with more than 500 occurrences:

- 4.5.1.M20150904-0015 (count: 13406)
- 4.5.2.M20160212-1500 (count: 12831)
- 4.5.0.I20150603-2000 (count: 9001)
- 4.6.0.I20160606-1100 (count: 6390)
- 4.6.1.M20160907-1200 (count: 2479)
- 4.5.0.I20150203-1300 (count: 1100)
- 4.4.1.M20140925-0400 (count: 610)
- 4.5.0.I20150320-0800 (count: 521)

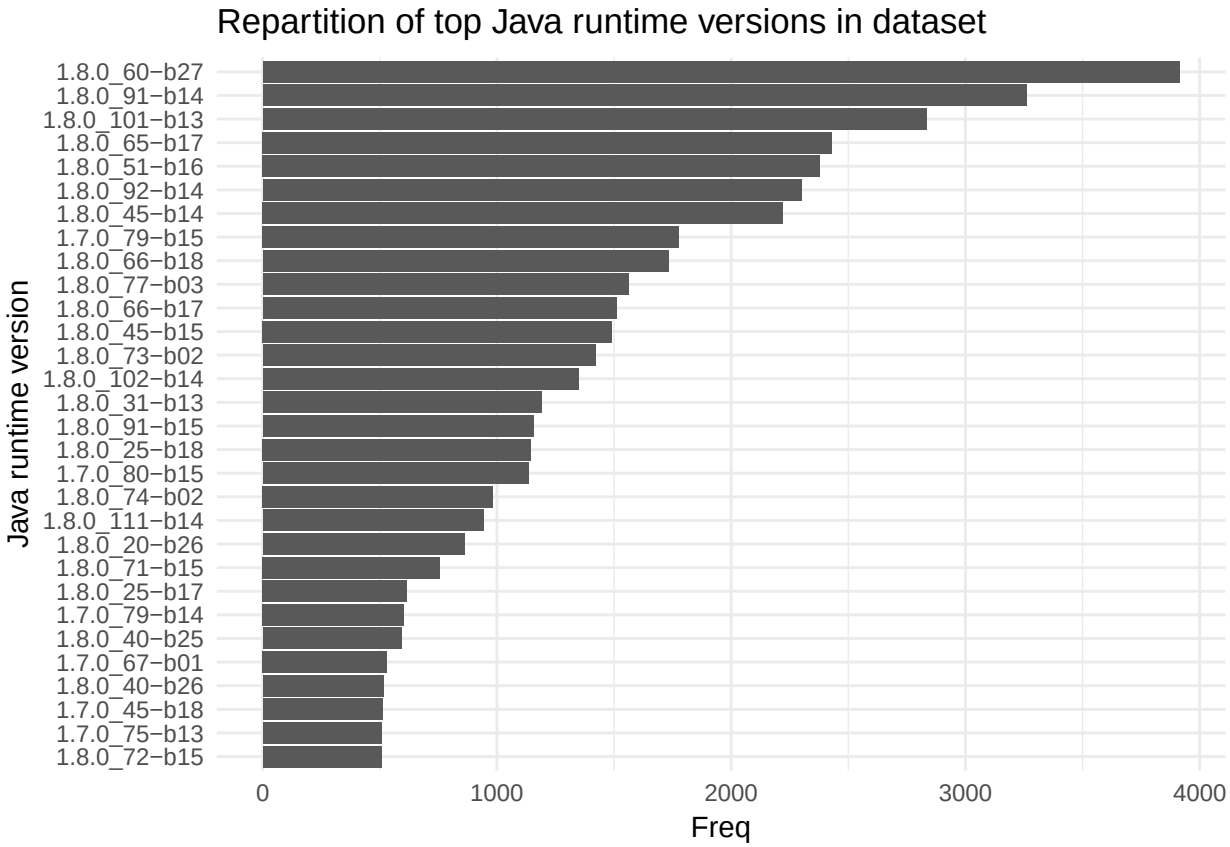


Java runtime version

- Description: The Java runtime of the host.
- Type: Factors

```
occurences.max.jrv <- 500
myjrvs <- data.frame(table(myproblems$javaRuntimeVersion))
myjrvs <- myjrvs[order(-myjrvs$Freq),]
myjrvs.top <- myjrvs[myjrvs[,c('Freq')] >= occurences.max.jrv,]
```

There are 452 different values found in the dataset for this attribute. The following bar plot only displays the values with more than 500 occurrences:



Using the dataset

Reading CSV file

Reading file from ../problems_extract.csv.

```
myproblems <- read.csv(file.in, header=T)
myproblems[,c("bug", "status")] <- NULL
```

There are 15 columns and 50872 entries in this dataset:

```
ncol(myproblems)
```

```
## [1] 15
```

```
nrow(myproblems)
```

```
## [1] 50872
```

Names of columns:

```
names(myproblems)
```

```
##  [1] "summary"           "numberOfReporters" "numberOfIncidents"
##  [4] "v1status"          "kind"              "createdOn"
##  [7] "modifiedOn"        "savedOn"           "osgiArch"
## [10] "osgiOs"            "osgiOsVersion"     "osgiWs"
```

```
## [13] "eclipseBuildId"     "eclipseProduct"     "javaRuntimeVersion"
```

Using time series (xts)

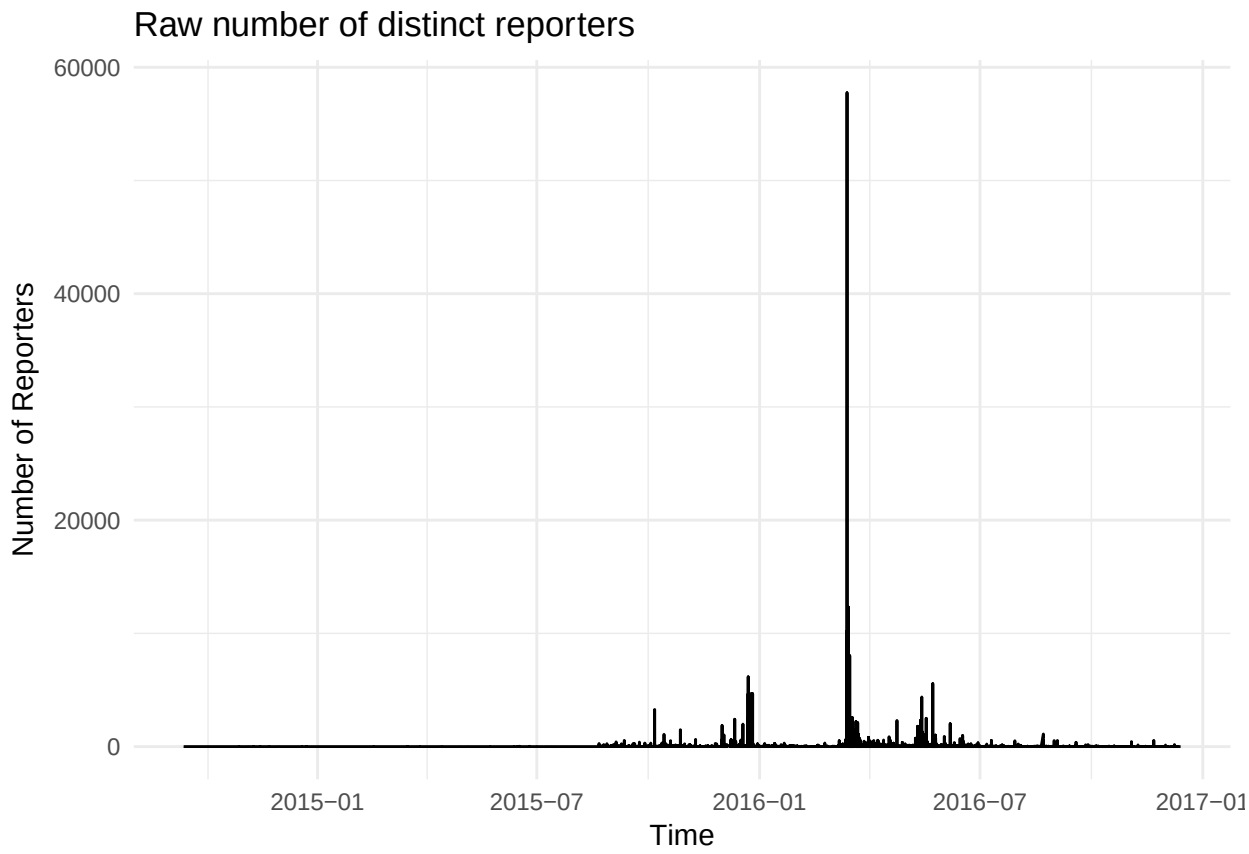
The dataset needs to be converted to a `xts` object. We can use one of the 3 dates

```
require(xts)
myp.xts <- xts(x = myproblems, order.by = parse_iso_8601(myproblems$createdOn))
```

Raw Reporters

Let's plot the number of reporters for each error report on a timeline.

```
xts.reporters <- xts(as.integer(myp.xts[,c("numberOfReporters")]), order.by = index(myp.xts))
autoplot(xts.reporters, geom='line') +
  theme_minimal() + ylab("Number of Reporters") + xlab("Time") + ggtitle("Raw number of distinct reporte
```

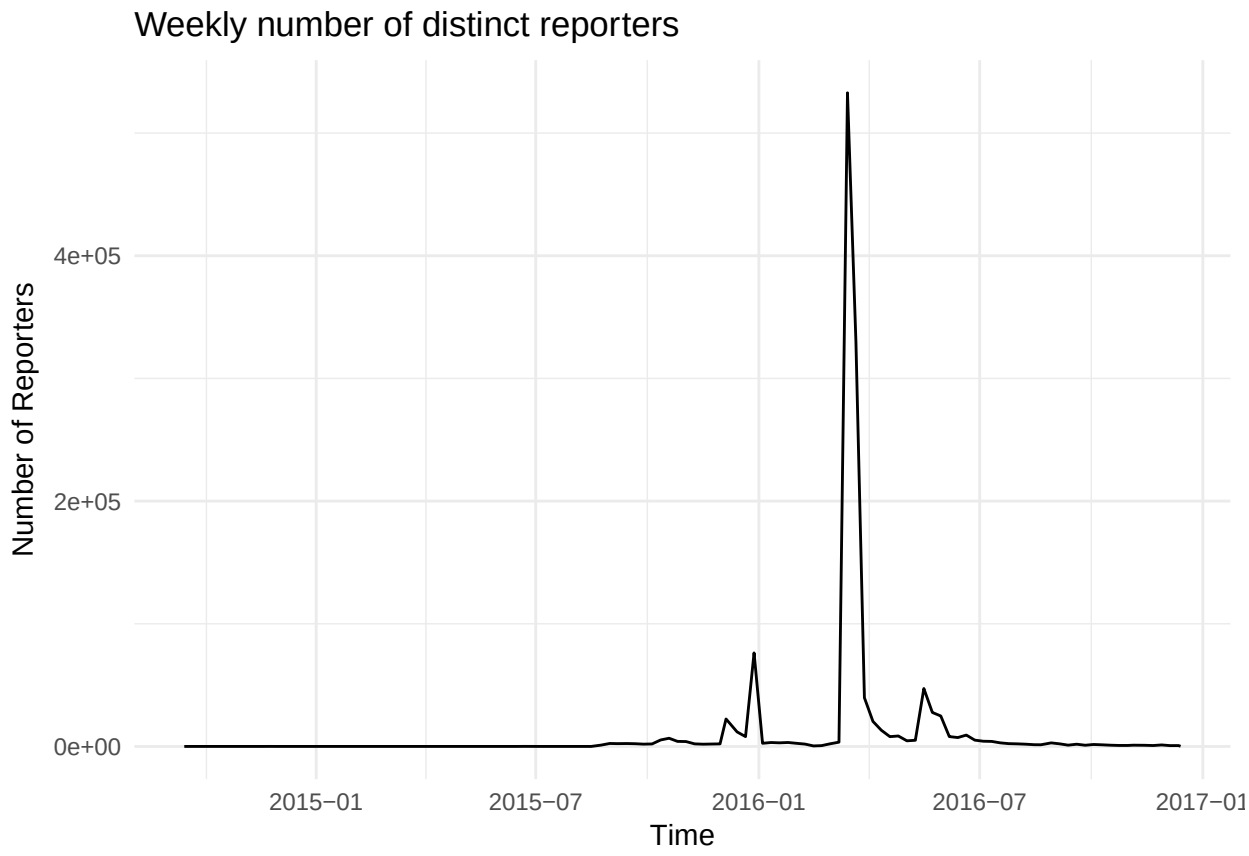


Weekly reporters

The previous plots used the `xts` object as it is, which is there is one point for each error report. When considering the timeline of the dataset, it can be misleading when there several submissions on a short period of time, compared to sparse time ranges. We'll use the `apply.weekly` function from `xts` to normalise the total number of weekly submissions.

Applied to the `numberOfReporters` attribute summed up with a week range, we get the following plot:

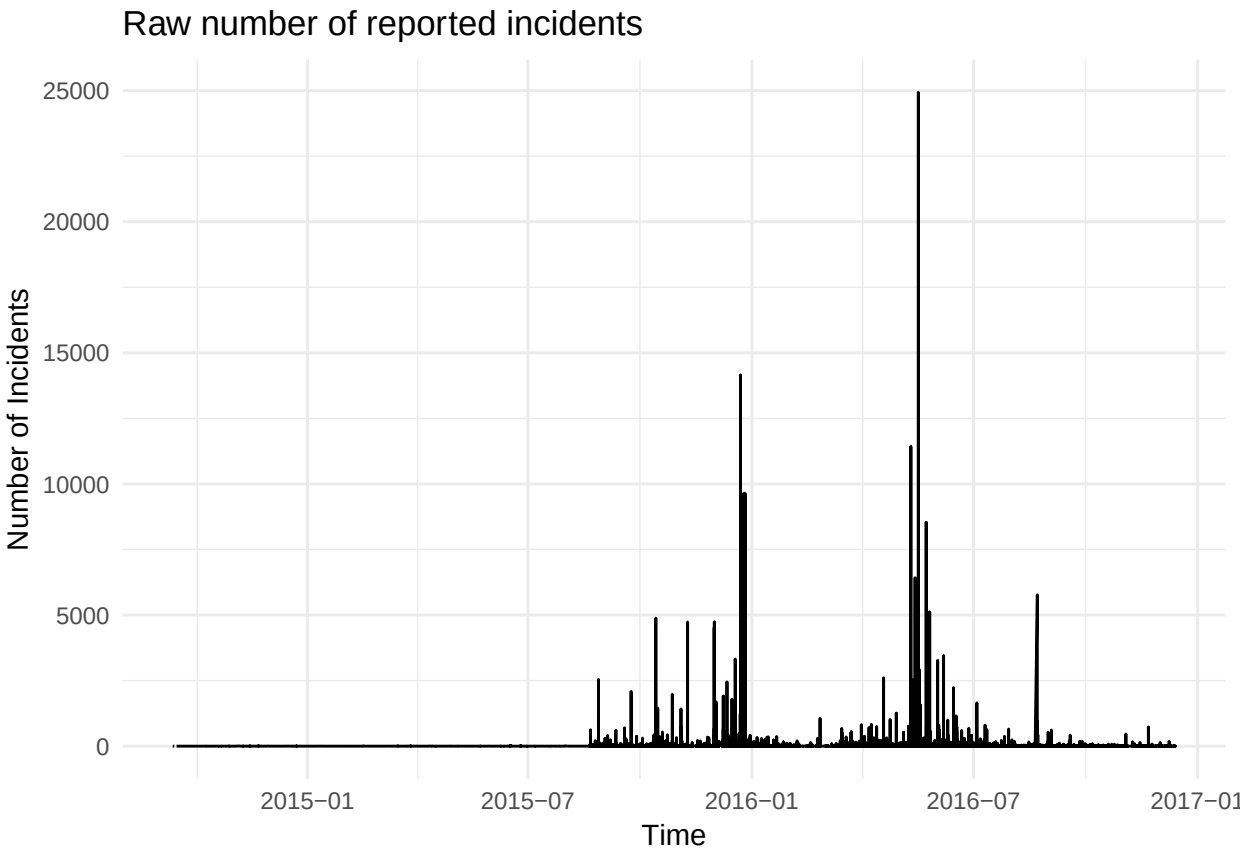
```
xts.reporters.weekly <- as.xts(apply.weekly(xts.reporters, sum))
autoplot(xts.reporters.weekly, geom='line') +
  theme_minimal() + ylab("Number of Reporters") + xlab("Time") + ggtitle("Weekly number of distinct repo
```



Raw Number of Incidents

Let's plot the number of incidents for each error report on a timeline.

```
xts.incidents <- xts(as.integer(myp.xts[,c("numberOfIncidents")]), order.by = index(myp.xts))
autoplot(xts.incidents, geom='line') +
  theme_minimal() + ylab("Number of Incidents") + xlab("Time") + ggtitle("Raw number of reported incide
```



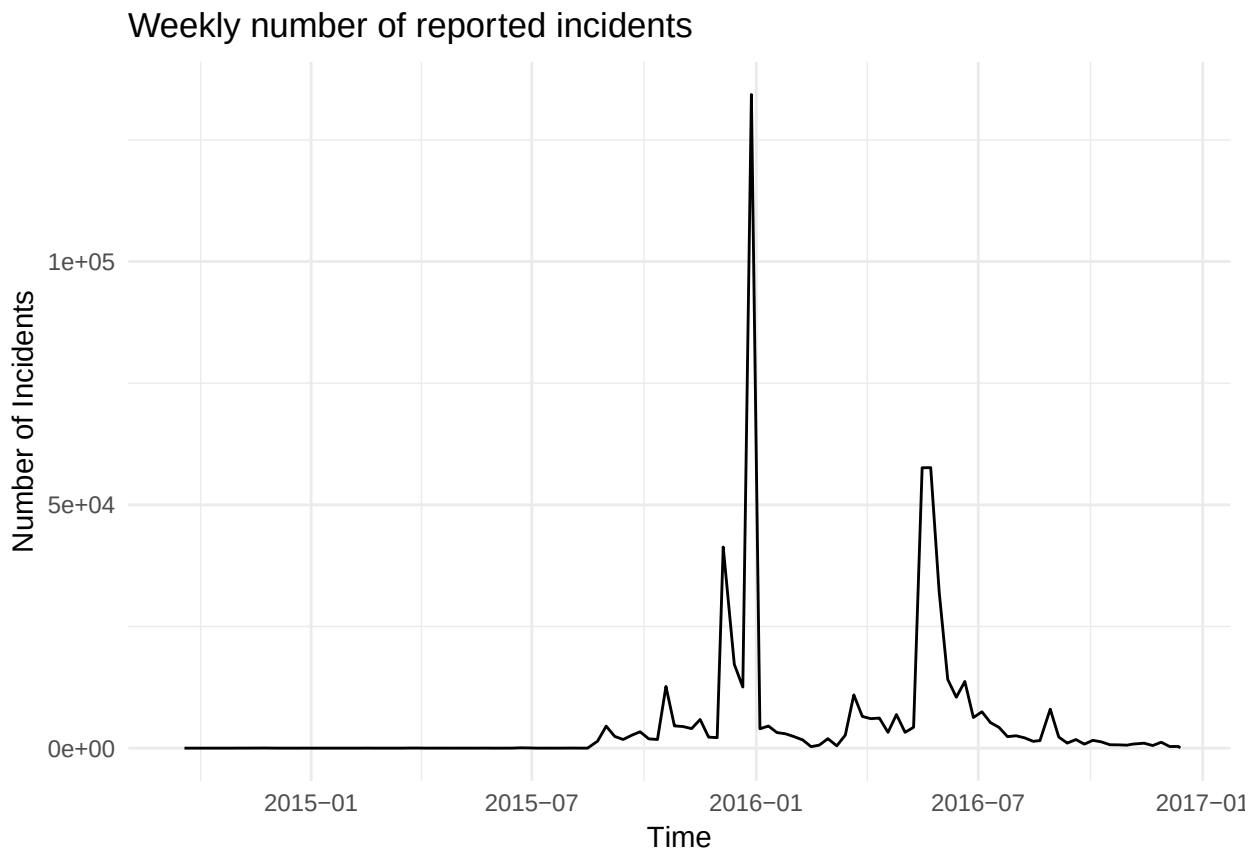
Weelky Number of Incidents

The previous plots used the `xts` object as it is, which is there is one point for each error report. When considering the timeline of the dataset, it can be misleading when there several submissions on a short period of time, compared to sparse time ranges. We'll use the `apply.weekly` function from `xts` to normalise the total number of weekly submissions.

Applied to the `numberOfIncidents` attribute summed up with a week range, we get the following plot:

```
xts.incidents.weekly <- as.xts(apply.weekly(xts.incidents, sum))
autoplot(xts.incidents.weekly, geom='line') +
  theme_minimal() + ylab("Number of Incidents") + xlab("Time") + ggtitle("Weekly number of reported inci
```

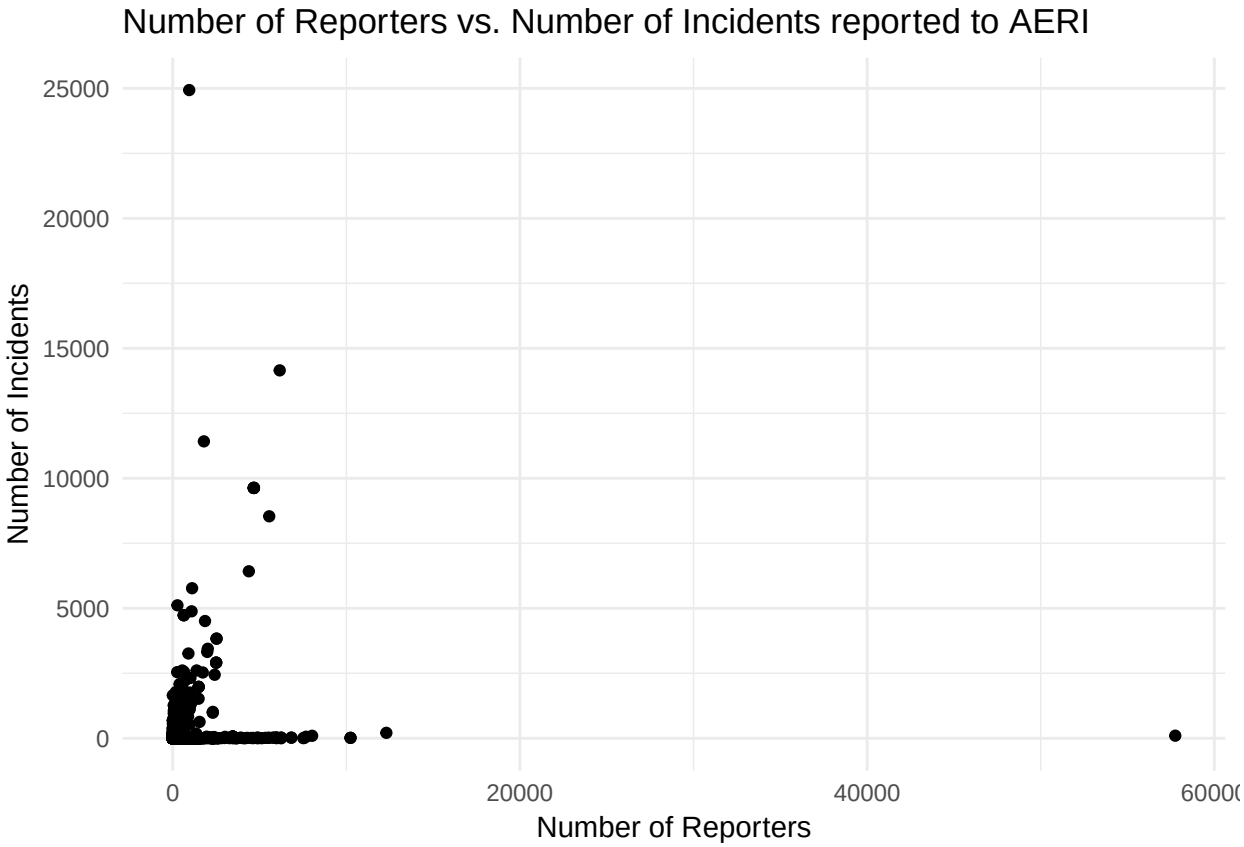
```
## Warning: Removed 1 rows containing missing values (geom_path).
```



Scatter plot

A scatter plot that compares the number of incidents reported and the number of distinct reporters.

```
qplot(myproblems$numberOfReporters, myproblems$numberOfIncidents) +
  theme_minimal() + ylab("Number of Incidents") +
  xlab("Number of Reporters") + ggtitle("Number of Reporters vs. Number of Incidents reported to AERI")
```


Number of Reporters vs. Number of Incidents reported to AERI
25000
20000
15000
10000
5000
0
Number of Incidents
0
20000
40000
60000
Number of Reporters