

# Design Specification (Internal)

## Yahoo! Downloads v1.0

---

Client: Yahoo!

Version 1.2

21<sup>th</sup> Aug 2010



Copyright © Monvia, All rights reserved.

# Contents

- Actors ..... 3**
- Use Cases ..... 3**
  - 1. Fetch Data Process ..... 3**
    - 1.1. Software Data ..... 3
    - 1.2. Meta Data ..... 6
  - 2. File Creator Process ..... 6**
    - 2.1. Software List CSV file ..... 6
    - 2.2. Category List CSV file..... 7
    - 2.3. RSS feed XML files ..... 7
  - 3. Data Show/Hide Toggle View ..... 7**
  - 4. Create Feat. Content ..... 8**
  - 5. Dynamic State View ..... 9**
    - 5.1. Platform View (Home) ..... 9
    - 5.2. Category List View..... 10
    - 5.3. Most Popular List View ..... 10
    - 5.4. Newest List View ..... 11
    - 5.5. Top Rated List View..... 11
  - 6. Software Details View ..... 11**
  - 7. Thank you View ..... 12**
  - 8. Static State View..... 12**
  - 9. Search/Results View ..... 12**
  - 10. RSS View..... 13**

## Actors

1. General Users
2. Y! Admin
3. IY! (Inside Y!)

## Use Cases

### 1. Fetch Data Process

The data source for the system is to be extracted from an XML feed provide by Tucows. The request requires authentication and processed as instructed by the provided examples. Any content received from any external sources need to be validated prior to being persisted into the system. In this case there are two validation processes that need to happen. First the data XML needs to be validated against an XSD to validate the correctness of the response format. Once the format has been validated the data contained in the response itself need to be validated for acceptable values. The first step in doing this is to run all data values through YIV (Y! Input Validation) that will strip out any inappropriate content as established by Y! standards. Thereafter we can run the data through any other custom validations as required. In the case where the feed fetch cycle resulted it an error owing to network outage, validation failure against XSD, YIV or custom validation the following actions must be taken.

- a. If the error was a retrieval error: The request should be retried for a set number of times (configurable through system properties). If all the retries results in failure, the URL being invoked (inclusive of all parameters) should be logged into the data store with the following information: Invocation Time, Feed Type, URL and Error Dump
- b. If the error was due to a validation failure (e.g.: XSD validation) which results in not being able to process the whole feed itself, then that too should be logged. Only difference here however is that since the feed was fetched successfully, the whole response should be logged. Since the response may be quite large, it should be written to the feed\_error\_files directory (configurable through system properties) and its file name stored in the data store. Therefore, the following should be stored for this case: Invocation Time, Feed Type, URL, Error Dump and Error File Name

#### 1.1. Software Data

This is the primary data source for the system. There are about 200,000 software titles, 2000 categories/taxonomy and 50 distributions/OS versions that need to be processed. The daily load however would not be this extensive, as not everything will be updated on a daily basis. However, there could be a case where the data is too much to be processed in one go. To

overcome this potential constraint the responses can be paginated to be processed in blocks. This also gives a workaround to fetch any block independently in case the fetch cycle was not successful for some reason (e.g. network outage). The order for fetching the data is;

1. Category/Taxonomy
2. Distributions
3. Software Titles

Following is the request and response model for this purpose.

REQUEST:

```
<FEED_URL>?start=12345&end=67890&records_per_block=1000&block_number=1
```

*start*: The start timestamp (date in unix since epoch) of the data (when it was last updated). On a running system, this would be when the data was fetched successfully the last time round.

*end*: The end timestamp (date in unix since epoch) of the data (when it was last updated). On a running system, this would be the present scheduled fetch cycle that is running.

The reason for having these start and end limits is to facilitate any data loss, so that we can resume from where we left off in case a fetch cycle was a failure or to fetch only the subset of data that were modified (excluding download counts and ratings) in any way between the start and end times.

*records\_per\_block*: The number of records that we can process at once. This is purely to manage the data load so that we don't have to wait for too long and process a lot of data between fetch cycles.

*block\_number*: The block number that is being requested. In the event where there is more than one block to process, this param's value will increment by one after each successful response. This also gives us a fall back to be able to process a subset of data, in case there was some unforeseen error in data processing (e.g. request time).

RESPONSE:

```
<feed name="Tucows Software Download" source_name="Tucows"
date="1280449275" start="12345" end="67890" records_per_block="1000"
block_number="1" total_blocks="20">
```

The response's root node would return all of the request parameters as is, with the exception of *total\_blocks*.

*total\_blocks*: Having provided the number of records we can process, Tucows will give us the total number of blocks which we will use to make subsequent requests in order to have the complete list of updates.

Sample of the of the response is as follows

```
<feed name="Tucows Software Download" source_name="Tucows" date="1280449275">
  <software>
    <software_update ts="1280278885" action="add" id="7419">
      <addl_required><![CDATA[]]></addl_required>
      <category_id>189</category_id>
      <cost>0.00</cost>
      <currency><![CDATA[]]></currency>
      <date_added>948725998</date_added>
      <date_last_modified>985699552</date_last_modified>
      <downloads_last_week>0</downloads_last_week>
      <downloads_total>1741</downloads_total>
      <downloads_yesterday>0</downloads_yesterday>
      <eval_period><![CDATA[]]></eval_period>
      <files>
        <file preferred="y" id="32501">
          <distributions>
            <distribution_id>1</distribution_id>
            <distribution_id>29</distribution_id>
          </distributions>
          <download_url>
tucows_download="yes"><![CDATA[http://www.tucows.com/get/7419_32501]]></download_url>

          <file_md5><![CDATA[e1565bd8555ce4883cc5370c7159bb0a]]></file_md5>
          <file_name><![CDATA[sredird-
1.1.8.tar.gz]]></file_name>
          <file_size>22216</file_size>
          <file_version><![CDATA[1.1.8]]></file_version>
          <revision_date>973832400</revision_date>
        </file>
      </files>

      <home_page><![CDATA[ftp://ftp.metalab.unc.edu/pub/Linux/system/serial/]]></home_pa
ge>

      <license_description><![CDATA[GPL]]></license_description>
      <long_description><![CDATA[Sredird is a serial port redirector that
is compliant with the RFC 2217 "Telnet Com Port Control Option" protocol.<p>This protocol
lets you share a serial port through the network. ]]></long_description>
      <min_requirements><![CDATA[]]></min_requirements>
      <note><![CDATA[]]></note>
      <program_name><![CDATA[Sredird]]></program_name>
      <publisher><![CDATA[]]></publisher>
      <rating>4</rating>
      <screenshots></screenshots>
      <short_description><![CDATA[Sredird is a serial port redirector
that is compliant with the RFC 2217 "Telnet Com Port Control Option" protocol.This
protocol lets you share a serial port through the network.]]></short_description>
      <software_type><![CDATA[Regular Program]]></software_type>
      <support_url><![CDATA[]]></support_url>
      <tags></tags>
    </software_update>
    <software_update ts="1280278885" action="add" id="7419">
      .....
      .....
    </software_update>
  </feed>
```

## 1.2. Meta Data

The Meta data is referred to any data that change on a daily basis that can be processed independently without having to process the whole context of the data item. Such identified data are total download, downloads in the last 7 days, downloads in the last 24hrs and ratings. This data feed needs to be processed only after processing the software titles as these attributes are dependent on the software titles. The request model is the same as that of the software data and the response is similar to that as well. Below is the response model.

```
<feed name="Tucows Software Metadata" source_name="Tucows" date="1280449275"
start="12345" end="67890" records_per_block="1000" block_number="1" total_blocks="20">
  <software>
    <software_update ts="1280278885" action="add" id="7419">
      <downloads_last_week>0</downloads_last_week>
      <downloads_total>1741</downloads_total>
      <downloads_yesterday>0</downloads_yesterday>
      <rating>4</rating>
    </software_update>
    <software_update ts="1280278885" action="add" id="7419">
      .....
    </software_update>
    <software_update ts="1280278885" action="add" id="7419">
      .....
    </software_update>
    .....
  </software>
</feed>
```

## 2. File Creator Process

There are certain processes that create flat files or feed files beforehand solely for the purpose of not overloading the system at request time on a response that can be pre-processed in anticipation of receiving the request at some point in time. There are two such instances where the files based responses are set;

- CSV based flat file for IY! (Another Y! service that will be promoting the site on searches)
- XML file for RSS feeds. These feeds need to be generated for Top Rated (100), Most Popular (100) and Newest (100)

### 2.1. Software List CSV file

The software title is required to be created for the whole data collection, but for a sub set of values. The format in which the file needs to be prepared is as follows.

Term<tab>Name – software title<tab>URL

Term: The level 1/Main Platform ancestor of the category

Name: The title of the software

URL: The DYC URL of the destination page

Once the file has been created, it needs to be copied (SCP) to a remote location, and once the copy process is successful, a blank/zero-byte file with the same file name but with the file extension of “.ok” should be created in a URL accessible directory. For this purpose, we will store the confirmation files in a directory by the name “iy\_scp”. An example of this would be as follows: downloads.yahoo.com/iy\_scp/software\_list\_MMDDYYY.ok

## 2.2. Category List CSV file

This is similar to that of software list, but for categories. The format is the same as that of software list.

## 2.3. RSS feed XML files

This is for the RSS feeds of the items mentioned in “RSS View”. The RSS file will be pre-generated so that to minimize stress on the system. The format for the XML file will be that of the standard RSS format.

## 3. Data Show/Hide Toggle View

This is an Admin process where certain software titles may need to be excluded from the whole site. These exclusions are not data provider specific but rather Y! specific. This process is as simple as crossing off the software titles that need to be hidden/inaccessible on the site by the common audience. However grouping and displaying these software titles by categories makes traversing the list much easier. A separate list of the items excluded should also be displayed so that they may be re-enabled. Any items that are excluded by this process should not even be displayed anywhere else on the site, unless explicitly specified. Even though this use case would have an effect on the pre-created contents, we would not be doing any modification to those contents (processes described in the File Creator Process). However, if anyone did try to access these blocked off content pages they should be redirected to the corresponding parent category list page or the platform main page.

This show/hide toggle page however needs to be protected. This would be done by the Y! Bouncer configuration, hence there is no need to implement an authentication module.

#### Primary Flow

Admin opts to excluding some software titles from the systems. The view loads up all the categories and their corresponding software titles as well. There should also be option to filter out the software titles by child categories so as to make the process of identifying the software easier. The admin would then cross off the software to be excluded and save the list.

#### Alternate Flow

Including any excluded software would be similar to that of marking the software to be excluded. In this case, the list is reversed, where the list displays the excluded software which the admin would check off to include them again into the system.

### **4. Create Feat. Content**

This is an Admin functionality where downloads are pre-selected in order to promote them. These featured downloads are only for the Level 1 categories and Freeware list aka Main Platforms. The admin tool is only required to facilitate the management of creating, updating and removing these lists of items. Each category can have a maximum of 5 records; they can have less than 5 depending on the scenario and purpose.

This create featured content page however needs to be protected. This would be done by the Y! Bouncer configuration, hence there is no need to implement an authentication module.

#### Primary Flow

Admin opts to create the featured content list. The system will serve the user with the list of all level 1 categories/Main Platforms and the software associated with it and its child categories in a paginated list alphabetically with the option of filtering the software's by sub categories. In order to give the admin a clear understanding of how the software's are grouped, each software title will have to indicate their corresponding category. The admin would then check off the software that is to be featured and then save the list along with custom titles and descriptions.

#### Alternate Flow 1

Editing the featured software would be similar to that of create, where the selected software will be pre-checked in the list of software's of the selected category.

#### Alternate Flow 2

The manager console would typically list out the entire featured software list in order to view and edit them. This list should also comprise of the option for one click delete as well.

## 5. Dynamic State View

This use case describes how general users will interact with the system when viewing dynamic pages. These dynamic pages have the following URL format;

```
downloads.yahoo.com/<platform>/<level ... categories>/<software-title>
```

Based on the above structure every level 1 category is treated as a homepage (with the exception of Freeware).

### 5.1. Platform View (Home)

There are six different home pages for each identified platform that make up the site, which are served based on the platform from which the user is viewing the site. This segmentation not only applies to the home pages, but to the whole sites itself, unless where the user specifically makes a request to other platform's content. The platforms are;

- Windows (downloads.yahoo.com/windows),
- Linux (downloads.yahoo.com/linux)
- Mac (downloads.yahoo.com/mac)
- Web (downloads.yahoo.com/web)
- Mobile (downloads.yahoo.com/mobile)
- Freeware (downloads.yahoo.com/freeware)

Even though Web and Freeware are not OS based platforms, they are the two major classifications other than OS based platforms. Also note that freeware is not a Level 1 category, hence we will need to make an exception just for this case. Each of these pages will serve the following content (refer wireframe/mock page 1 for reference) that are relevant to their home page.

**List of Categories** – Display both all Level 2 categories and a max of 5 random Level 3 categories that belong to the Level 2 category

**Featured downloads** – Display content created by admin for the specified Level 1 category/platform

**Top rated (active tab)** – Display the top 10 software's that has the chosen segment/Level 1 category as its ancestor ordered by 'rating' attribute

**Newest (inactive tab)** – Display the top 10 software that has the chosen segment/Level 1 category as its ancestor ordered by 'latest\_download\_revision\_date' attribute

**Most Popular** – Display the top 10 software that has the chosen segment/Level 1 category as its ancestor ordered by 'downloads\_last\_week' attribute

#### Primary Flow

The primary scenario begins when a user accesses the site by the domain only. As the user requested for the non-specific home page, the request should be intercepted where the platform is identified and thereafter redirected to the appropriate home page. An example of this would be where the user only puts in the software title after the domain.

#### Alternate Flow 1

When the platform is unidentifiable the home page defaults to Windows.

#### Alternate Flow 2

The user specifically requests for a platform page, in which case there should not be any interceptions to deviate the user from what he/she is trying to access.

### 5.2. Category List View

These pages are based on the category hierarchy with the facility to drill up or down from Level n category to Level n+1 category and vice versa. The categories drill down/up functionality will be facilitate via the left rail, where the category tree can be expanded and collapsed. The drill up can also be performed via the breadcrumb on the page. The page will comprise of the following contents.

**Category Menu** – The expandable and collapsible menu displaying Level 2 – Level 4 categories

**Category Content** – Paginated list of all software that are part of the selected category and its sub-categories. The details to display are title, description (max of 100 characters), downloads last week, total downloads, rating and last updated date. The paginated list will display a max of 10 records per page

**Featured Downloads** – Display content created by admin for the specified level 1 category

**Most Popular/Top Rated/Newest** – The first 10 items for each of these topics, with a call to action link that will point to their corresponding pages.

#### Primary Flow

The user clicks on a particular category's call to action link/button on any of the pages.

### 5.3. Most Popular List View

Very similar to that of Category Listing above, this view would display software based on number of downloads in the last week. However, this list is limited to a max of 100 records that will be paginated at 10 records per view. The page will comprise of the following contents.

**Category Menu** – This would be the same category menu on the home page

**Popular content** – Paginated 10 records per view displaying a maximum of 100 records related to the Level 1 category

**Featured Downloads** – Display content created by admin for the Level 1 category  
**Top Rated/Newest** – The first 10 items for each of these topics, with a call to action link that will point to their corresponding pages.

Primary Flow

The user clicks on a particular category's call to action link/button on any of the pages.

5.4. Newest List View

Similar to that of Most Popular List

5.5. Top Rated List View

Similar to that of Most Popular List

**6. Software Details View**

This view would be displayed when user arrives directly or by navigating his/her way through the site in preparation to download the selected software. This page is to display pretty much all of the information about that software and a prominent call to action button that would point to the Tucows download URL provided in the feed. After a set time (configurable via system properties) has expired since the click action, the details page should automatically redirect to the thank you page detailed below. If the download URL is not a Tucows URL, then the click action should open the URL in a new window/tab. In the case where there are screen shots these need to be served as a carousel built using YUI (and no other JSUI library). The click action on these images should display them as an overlay on the page. The page will comprise of the following contents;

**Category Menu** – This would be the same category menu on the home page

**Software Details** – title, description, tags, release date, editorial rating, downloads last week, total downloads, publisher, license, cost (if applicable), trial period (if applicable), requirements, support destination and screenshots carousel.

Primary Flow

The user navigates his/her way or arrives directly to the page

## 7. Thank you View

This page would be the last view that user would see before the download begins. This page should also not be directly accessible and should be validated against a time based token that was parameterized from the details page. The content of this page comprises of;

**Category Menu** – This would be the same category menu on the home page

**Related Download** – Other software that are part of the same category. They are matched by tags for relevance and the order set by ratings and the release date.

### Primary Flow

The user arrives at the page by clicking on the call to action button on the Software details page and has a valid time sensitive token.

### Alternate Flow

The time based token that was passed is invalid in which chase the user should be take back to the Software details page to be issued a new token in order to complete his/her download.

## 8. Static State View

These are just simple HTML pages that are only for the purposes of informing the user of the site's limitations and constraints. Currently there are 4 such pages, they are;

Submit Download

Rating Policy

EULA

FAQ

## 9. Search/Results View

The search is performed on the Main Platform/Level 1 categories. Each Level 1 category page has a search bar that is below their corresponding tab. The search query that is submitted is then sent to the Y! Search API (to be provided) that will perform the search on the data store and return the result collection. This result collection is then displayed on the results page with the following options and contents.

**Search Filters** – There are two types of filters License Type and Category. These filters will not essentially trigger another search, but rather display only the relevant subset of results from the original search results.

**Results Display** – The results are displayed by relevance with the options to sort by name, release date, rating, downloads last week and total downloads. The result per view is 10 records, which are paginated to the entire result collection.

**Featured Downloads** – Display content created by admin for the Level 1 category

**Most Popular/Top Rated/Newest** – The first 10 items for each of these topics, with a call to action link that will point to their corresponding pages.

In order for the search to function the contents in the system that is searchable has to be first indexed. The details of this indexing process are yet to be understood (on configuring Vespa).

#### Primary Flow

The user searches by a particular keyword on a Level 1 category. And the results are displayed by relevance

#### Alternate Flow 1

The search result is sorted by the other available parameters.

#### Alternate Flow 2

The search results are filtered by Category or License Type

## 10. RSS View

All the list views have the facility of being able to provide RSS feeds. These RSS feeds unlike the page views will have the complete collection of data in the feed itself. These feeds are however not processed at request time, since the data collection is based on static data that the systems is aware of beforehand during the data retrieval from the external data provider, these RSS files will be pre-created. Hence the RSS view only has to serve these files that were pre-created. In the unlikely event of a feed file not existing, the feed view will create the required feed file, persist it to the feed repository and then serve that as a response. The process of creating are described as part of the “File Creator Process”

#### Primary Flow

The user subscribes for the RSS feed, which browser’s the RSS feed reader then fetches periodically on behalf of the user.

## Revision History

Revision	Date	Description
1.0	15-Aug-10	Initial version
1.1	21-Aug-10	Updated
1.2	07-Sep-10	Updated