

KelownaGigs

Constructing a Social Publishing Application
to Support a Local Music Community

by

Geoff Appleby

A thesis submitted in partial fulfilment of the requirements for the degree of

Bachelor of Science Honours

in

The Irving K. Barber School of Arts and Sciences

(Honours Computer Science Major Computer Science)

The University of British Columbia Okanagan

April 2010

© *Geoff Appleby, 2010*

Abstract

KelownaGigs is a website initially built in 2003 to support Kelowna's local music community, including artists, bands, venues, promoters, and fans. With the previous system all site content had to be validated and inputted manually by a limited number of users with administrative access. This project focused on migrating the site to a social publishing application with the goals of automating as many tasks and processes as possible, and providing each user increased access to manage the content relevant to themselves. The existing content aggregation tools were deemed insufficient for the site's specific needs, and so information extraction techniques were researched in order to implement a custom solution for importing data from remote web sites. Throughout the project access logs and user statistics were kept to monitor changes in user behaviour as site changes were implemented. Additional feedback was solicited from users via surveys available throughout the development period, including questions tailored to their role within the community. Overall the site received positive feedback from users and traffic metrics increased throughout the duration of the project. Administrative effort was also reduced despite the increase in site usage and content.

Table of Contents

Abstract	i
Illustration Index	iv
Acknowledgements	v
1 Introduction	1
1.1 Motivation	1
1.2 Goals and Objectives	1
2 Background	3
2.1 Social Media	3
2.2 Web Information Extraction	4
2.3 Content Management	5
2.3.1 Drupal	6
2.3.2 Existing Information Extraction Tools	6
3 Site Overview	8
3.1 System Architecture	8
3.2 Site Construction	8
3.3 Database	10
3.4 Data Migration	11
4 Web Information Extraction	12
4.1 Sources	12
4.2 Chosen Approach	13
4.2.1 Fetcher	13
4.2.2 Parser	14
4.2.3 Processor	14
4.3 TicketMaster	14
4.4 Facebook	17
4.4.1 Fetcher	18
4.4.2 Parser	21
4.4.3 Processor	22
5 Content Access	25
6 Metrics	27

6.1 Access Logs	27
6.2 Surveys	29
6.2.1 Users	29
6.2.2 Artists	31
6.3 Content and Users	33
7 Discussion and Conclusions	35
7.1 Future Work	35
References	38

Illustration Index

Figure 1: Simplified RSS Structure Tree.....	7
Figure 2: Simplified Data Diagram.....	10
Figure 3: Information Extraction System Data Flow.....	14
Figure 4: TicketMaster Query Results.....	15
Figure 5: TicketMaster Event Object Properties.....	16
Figure 6: Public View of a Facebook Event.....	17
Figure 7: Event Import Form.....	18
Figure 8: Administrator Facebook Import Queue.....	18
Figure 9: Facebook Event XHTML Code Sample.....	19
Figure 10: Public View of a Facebook Page's Upcoming Events.....	20
Figure 11: Facebook Event List XHTML Code Sample.....	21
Figure 12: Parsing the Event Date from XHTML via XPath and Regular Expressions.....	21
Figure 13: Attempting to Import an Event that has Previously Been Imported.....	22
Figure 14: Form Populated with Parsed Data.....	23
Figure 15: An Import with an Item that could not be Matched.....	24
Figure 16: Field for Managing Accounts with Access.....	25
Figure 17: Weekly Visits and Unique Visitors, July through March.....	27
Figure 18: Pages per Visit, July through March.....	28
Figure 19: Bounce Rate, July through March.....	28
Figure 20: Duration of User's Social Network Account Memberships.....	29
Figure 21: User's Participation in Music Events and their Usage of KelownaGigs.....	29
Figure 22: User's Request for Features Versus Interest in Contribution.....	30
Figure 23: Duration of Artists' Social Network Account Memberships.....	31
Figure 24: Artists' Requests for Features Versus Interest in Contribution.....	32
Figure 25: Events Submitted to the Site, by Month.....	33
Figure 26: User Registrations, by Month.....	34

Acknowledgements

I would like to thank Dr. Ramon Lawrence, my supervisor for this project, for his valuable feedback and guidance throughout. Over the past four years, he and the rest of the UBC-O Computer Science faculty have also shared their knowledge and fostered my interest in computer science and programming such that undertaking this project was possible.

Many of my fellow students also provided encouragement, and spurred me on by sharing their own interests and successes.

1 Introduction

KelownaGigs.com was started around 2003 as a static web page for listing local events featuring live bands. At the time, event details were largely spread by word of mouth and via fliers at other events. This limited the spread of knowledge and made it difficult for new people to find out about events. KelownaGigs provided a central location for people to post and find events within the city, without requiring that they be constantly involved in events or part of a social group that is. However, as a static web page it frequently fell out of date as the creator's available time fluctuated. My involvement with the site began after creating a simple PHP based application to manage the listing of events, allowing multiple people to easily update the site and enabling dynamic updates of the site information.

1.1 Motivation

With the large growth of online social networks such as MySpace and Facebook, it has been easier for bands to spread information among each other and to fans. MySpace especially is used by bands, due to its separation of 'user' and 'band' pages, the latter of which allowed bands to upload their music for visitors to listen to within a Flash applet. This allowed bands to share their music but not allow visitors to download the music file, and consequently many musical artists, from local and unknown to large chart toppers, began using the site.

While social networking sites have removed the physical limitations of spreading information about events, they have taken a step backwards in how the information is spread. Rather than providing an improved, central repository of information they again rely on spreading through social groups and though the effect is reduced, information is often limited to existing, connected social groups. Other sites have been created as repositories of information but they lack the social aspect, resulting in them often being limited to only events that feature better known musical artists or take place in large venues.

1.2 Goals and Objectives

The focus of this project is to merge aspects of both social networking and traditional publishing websites into a social publishing framework. The site should be an effective repository of information, allowing all users access to content of interest, but also allow users to freely contribute and share content. This should improve the site's value as a resource,

while also reducing the requirements of site administrators to control all content. In addition to implementing tools to assist site operations, a significant aspect of this project is to determine how these changes affect users interactions with the site, and the value of such tools to site users. Hopefully the approaches taken in this project can be utilized in other media, to improve value, increase user interaction, and reduce administrative overhead, and the metrics gathered can provide insights into the value of producing these tools.

The thesis will begin by describing some of the background of social networks, information extraction, and content management systems. In Section 2 there is an overview of the system and more specific information on the software stack utilized behind the scenes. In Section 3 the approach and implementation of information extraction used to aggregate content for the site is discussed, and Section 4 covers the approach to user access control. Finally the thesis closes with an analysis of some of the metrics gathered over the course of the project, including survey responses and traffic logs.

2 Background

2.1 Social Media

Social networking sites are one of the fastest growing online mediums and contain an increasing collection of information. Facebook is one of the fastest growing social networking sites available, increasing from 150 million to 400 million active users between January 2009 and February 2010 [1]. Initially built primarily for sharing photos between friends, it has grown as an extensive distribution medium for information, notable here for its use in sharing event information: over 3.5 million events are posted to Facebook each month [2]. Facebook allows users to organize in Groups, with administrators able to distribute information to members, as well as providing a message board for members to communicate with each other. Additional features can be added, such as photographs, videos, events listings, and forums. Pages are a similar system, created as a way for people such as businesses and celebrities to create profiles, and mix many of the features of user profiles and groups. For most users both are functionally equivalent, however Pages support additional reporting features and are favoured by businesses for this reason. Groups were present in the initial development of Facebook, with Pages being a more recent introduction.

MySpace is one of Facebook's main competitors, and prior to April 2008 had more monthly unique visitors [3], but has since had significantly slower growth. While MySpace has many features similar to Facebook such as groups, events, and photo albums, its primary use is commonly for bands seeking to distribute their music. MySpace also typically has a younger demographic of users, usually in high school [4] rather than university which was Facebook's initial target market.

While social networking sites focus primarily on their users and the connections between them, social publishing focuses on the content people produce. The largest social publishing sites are usually focused on news, allowing all site users to write articles which will appear alongside articles written by site authors. NowPublic.com is an entirely user-authored news site, where employee contributions are given equal exposure to those of other users. Users can submit news stories on their own or in collaboration with others, or submit accessory content such as photos and videos for other users to include in articles. In addition site

content is linked to media available through other sources, such as stock prices and related Twitter topics.

2.2 Web Information Extraction

Information Extraction (IE) [5] is the process of automatically obtaining structured data from a collection of unstructured or semistructured data, and has gained increasing relevance with the growth of data available on the World Wide Web . There has been increasing interest in descriptive web formats such as Resource Description Framework (RDF) to include machine readable metadata to assist parsing general data, or data specific formats such as Really Simple Syndication (RSS) for the distribution of syndicated content such as news articles. The large majority of content on the web, however, is solely focused on display for end users with few semantic hints.

In many instances, particularly with newer websites, an Application Programming Interface (API) is provided to allow other programs to access a website's information in a programmatic manner. For example, through the Flickr API a program can search for photographs tagged with a set of keywords and incorporate them into its own output. Both the data request and response are prepared in a machine parseable format such as XML. While the number of organizations providing such interfaces is increasing, many popular sites have yet to provide them, or have not made them adequate for accessing all needed information [6].

Tools for IE vary from simple systems providing a query language to extract page elements, to complex systems which find and label individual elements without user interaction. They also range from parsing data from a single page to populating a database with content extracted from an entire site. These methods all have varying effectiveness on different extraction domains based on the nature of the content. Some important criteria to consider are how complex the extraction method's configuration is, how it deals with variance within and across data sources, and how much user interaction is required for the method to complete operation.

IE methods aim to extract a set of attributes out of a source document, which may have various properties. Each attribute may consist of one or multiple values, with the possibility of having empty attributes that may contain no value in some instances. The attributes

themselves may occur in multiple orderings depending on their context, or have differing formats. All of these factors contribute to the complication of building a comprehensive IE system, and are even more significant when dealing with multiple sources.

Approaches to generating IE tools vary in their level of automation from requiring a user to construct parsing rules manually in a general programming language to simply returning extracted content based on calculated rules. While manual extractors require greater user knowledge, they can be constructed for the greatest range of data sources. Automated systems may adapt to differing data sources more easily, but will potentially require user intervention in order to ascertain what data it is able to extract is actually relevant. The user can provide hints to the system prior to its analysis of source data, or the system can attempt to extract data and allow the user to specify any modifications afterwards. In general, automated approaches may be better suited to data that may change in format over time due to the simplified process of updating the extraction rules, whereas the manual approaches may require significant investment from a user with greater programming knowledge and would be more effective for static data formats.

2.3 Content Management

Content Management Systems (CMS) control the collection and availability of content, frequently providing various levels of access control to content and the features needed to organize it. CMS complexity can range from personal blogs, which only support one type of content produced by one author, to large media websites, which produce many types of content by many authors. Early content management systems were usually custom built for particular organizations, but with the growth of the Internet packaged systems configurable for a variety of implementations have been developed. Many open-source alternatives have developed large communities and have active development cycles.

The main systems considered included Drupal, Joomla and Wordpress. Systems based on programming languages other than PHP were not considered due to developer familiarity and the greater availability of hosting services for PHP applications. As an application primarily focused on blogs or article based sites, Wordpress did not offer the granular content storage and control required. Joomla and Drupal are two of the most well known open source CMS frameworks and offer many similar features. Of the two, Drupal appeared to have an

architecture with a greater focus on supporting application development and a more cohesive library of existing extension modules. A key feature of Drupal not present in many other systems was its focus on enabling community websites.

2.3.1 Drupal

Drupal [7] is one of the largest open source, PHP based content management systems available. Initially released in 2001, Drupal is currently on its sixth major release and version seven is scheduled for release sometime in 2010. With a focus on code quality and development flexibility, a large developer community, and an extensive selection of modules available, it was selected as the base for the new system. While a notable amount of time was required to adequately understand the architecture in order to extend it, the system handles the basic tasks, allowing designers to focus on system improvements and higher level operations, rather than implementing many features from the ground up. Security is also an important consideration for any software system, and utilizing Drupal and other open source applications provides a thoroughly tested framework which is continually scrutinized by a large community for security issues.

As a large open-source project, Drupal has an extensive library of contributed modules that were utilized to provide many of the features used on the site. The core Drupal system only supports simple content types consisting of a title and body field, and Content Construction Kit (CCK) [8] extends this to provide a framework to create content types with any number of content fields, with custom storage formats, validation, and display. Views [9] provides a graphical query builder to allow custom displays of site content, including arguments provided statically, through URL paths, or user-specified parameters. These modules provided the basic information and presentation structure of the site. Additional features produced over the course of the project were implemented within additional extension modules for the Drupal system.

2.3.2 Existing Information Extraction Tools

Drupal has two modules, Feeds API [10] and its recent successor Feeds [11], created for importing data from remote sites. The initial focus of these two modules was primarily importing RSS feeds, and as such they expect data in a simple format with an item hierarchy such as in Figure 1. Additional parsers for other formats, such as iCal for calendar items and

KML for geographic data, have been created but also expect a similar, relatively flat structure in a standardized format. All these formats also share the common attribute that they are designed for sharing data across systems, rather than representing the data directly to the user as is the purpose with general HTML and XHTML web pages.

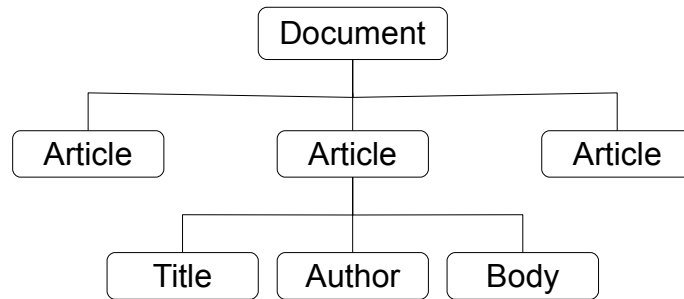


Figure 1: Simplified RSS Structure Tree

While Feeds provides a promising path for future development of more customizable and extensible options, its current restrictions provided too great an obstacle for its use as either a solution in itself or a framework for extension. The two sources utilized in this project provide data in very different formats, but share the common attribute of a complex, nested structure to the data they provide which was not easily navigable by existing solutions. A significant complication is also that Feeds expects an item's data to be encapsulated in one request, and so cannot collect related data from several pages into one content item, as occurs in some of the sources considered.

PHP includes built in classes for parsing and navigating XML and HTML documents. The DOMDocument class [12] is capable of parsing XML and HTML strings into a Document Object Model data structure for navigation and manipulation. SimpleXML [13] provides a restricted but accessible interface to provide easier access to data and querying on the structure through XPath queries. DOMDocument provides an advantage in that it can parse HTML documents that are not well-formed, which SimpleXML cannot handle, but requires more code to find and extract data items.

3 Site Overview

The site is primarily structured around listings of events, with ancillary data including venue and artist listings. Other content of lesser prominence currently includes classified ads and general articles. Users are able to create new content of any type on the site, but may be restricted in what existing content they have access to edit. Event listings are currently limited to those occurring within Kelowna, however artist listings exist for artists from across North America.

3.1 System Architecture

KelownaGigs was initially run in a shared hosting environment, but with the database and memory requirements of the Drupal framework, other resources were required. During this project, the application was migrated to a low resource Virtual Private Server running Ubuntu Server. The application stack consisted of the Apache web server with PHP module and a MySQL database server. All applications used are available under Open Source licenses [14].

3.2 Site Construction

The site was constructed with an initial focus of duplicating the content and features of the existing site. Utilizing the core content type system and extended with CCK data fields, content types were created for Events, Classified Advertisements, Venues, and Artists. In the previous system events were already linked to venue listings, however artists were provided in a simple text box. With the new content type, both the venue and artist entries for events are linked to related content on the site. An additional content type was added for site news, which was previously entered into a single text field on the old site and repeatedly overwritten, and for articles which did not exist previously.

Drupal's Taxonomy system was used to store the location information for artists, previously divided among Kelowna, Okanagan, and other regions. Due to the ease of configurability this was extended with additional terms to include other cities within the Okanagan and each of the Canadian provinces and territories. Further granularity for other regions outside the Okanagan was not provided at this time in order to reflect the locality of the site, and simplify the interface for users. An additional vocabulary was added for genres, in order to provide users an additional way to classify and find artist information. While the regions vocabulary

is currently fixed and only an administrator is able to modify it, the genres vocabulary is classified as a 'tagging' vocabulary, allowing any user to specify new terms.

The previously created events listings, such as upcoming, newly added, and past events, were duplicated on the new site by creating pages with the Views module to aggregate the relevant events. Similarly, classified ads were shown on their own pages and were easily filtered by date to only show recent posts – a feature that had to be performed manually previously, as out of date entries were not removed.

The previously available artists listing on the front page was initially duplicated to show all artists available within the database. This was soon found to be excessive, however, especially after additional artists listings were created from past events – over 1000 individual artists were stored. Because artists were now linked to events, it was possible to only display artists that had participated in an event within a defined period. The Views module by default only supports one-way connections between content linked as events and artists are, initially requiring a query selecting recent events and then their associated artists. An additional module was found that supports connections in the alternate direction, allowing selecting artist profiles and then filtering them based on their last performance. The second method also allows the artists to be filtered on additional criteria, such as the creation date of their profiles, which was not possible with the first query method.

Many additional modules were explored to enhance features of the site. The user experience was improved by adding additional menus, customizing the URLs of site content, and providing redirects from pages on the old site to corresponding pages on the new system. In order to mitigate automated spam, a web service was utilized to screen some of the content posted to the site, and a filter was utilized to obfuscate email addresses being posted to the site in order to hide them from email harvesters. Other modules increased search engine exposure by automatically generating site maps and providing key terms.

Throughout deployment it was at times difficult to find modules solving a particular issue, and other times deciding between more than one module that contained similar features. In many instances a popular module was available and well supported by the Drupal developer community, but less common features often required lesser used modules with few contributors. The latest stable major version of Drupal, version 6, though available since

February 2008 contained many significant architectural changes and extension modules required significant changes to match. Some of the modules used on KelownaGigs were still in the process of updating features from previous versions, and many modules explored were only compatible with Drupal version 5. As with any software program bugs were found throughout deployment, resulting in several fixes being contributed back to the projects.

3.3 Database

The core Drupal system as well as the CCK module have robust, automatic database structure definitions that handle data used within the system well. This has been utilized for the system's core content types such as events, venues, and artists, as illustrated in Figure 2. Additional tables were created for additional storage needs required in constructing the site, such as storing entity identifier relationships between content on the new system and other data sources. The Facebook extractor also implements a content cache in the database in order to reduce network traffic.

To ensure data consistency and security, data manipulation was performed through APIs exposed from Drupal or other extension modules rather than direct queries to the database. This allows any other modules that manage related data to be notified of changes and perform any necessary operations. For performance, read only operations were performed directly on the database where possible, with subsequent write operations still using exposed APIs to maintain data consistency.

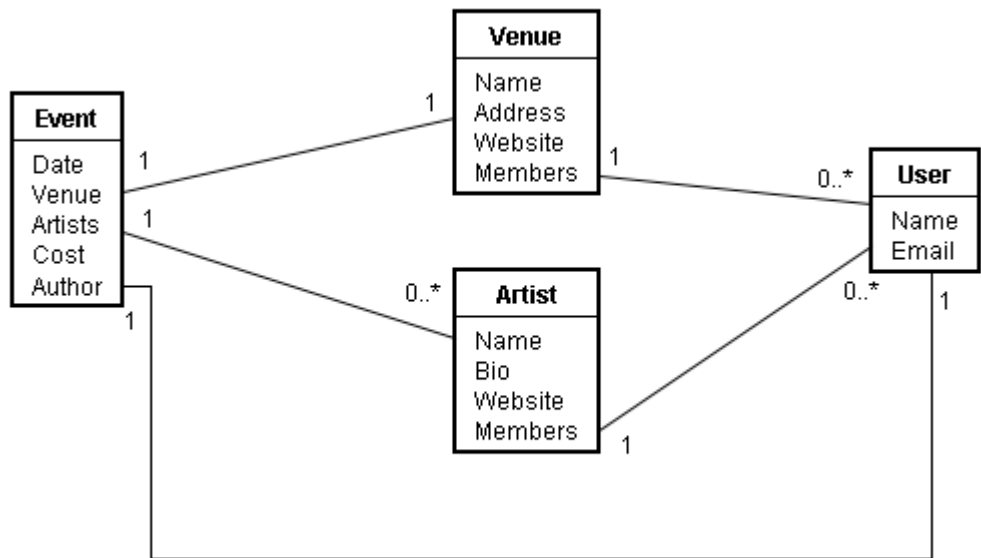


Figure 2: Simplified Data Diagram

3.4 Data Migration

The first development task was migrating the data from the legacy system into the new framework. In utilizing a soft launch of the new system, the data from both databases had to be linked to enable periodic updates of content without duplicating previously migrated data. A separate processor was created for each of the three types of content that were to be migrated: events, artists, and venues. Each processor could be called manually as needed, with the events processor performing the most work. Since the old system was to be disabled after the full launch of the new site, content was only updated on the new system and new and updated content on the new system did not affect the old one.

4 Web Information Extraction

Information Extraction methods were explored in order to supplement the information added to the site by administrators and users, as well as make the process of adding content to the site from other sources easier for users. The two uses of IE techniques were in automatically aggregating content for the site from external sources without user intervention, and providing an interface for users to duplicate content from other sites onto KelownaGigs with reduced effort. In combination, these uses seek to increase the amount of content stored by KelownaGigs and increase user interaction and contributions.

4.1 Sources

Two sources, TicketMaster and Facebook, were used for this project as they provide a large majority of the data that is currently added to KelownaGigs. TicketMaster primarily services large events that occur less frequently; the Kelowna Community Theatre, which uses TicketMaster for most music events, hosted 5 events from January through March 2010, whereas smaller venues such as music clubs, restaurants, and pubs will feature 1 to 3 performances each week. Facebook is increasingly being used by venues and artists to spread information about their performances. Of all events posted to KelownaGigs during 2009, 42% included a link to an event listing on Facebook, up from 28% in 2008 and 15% in 2007.

These two sources also provide different challenges for extracting information due to the formats in which they provide data. TicketMaster provides a listing of events for a specified venue or artist in Javascript Object Notation (JSON) format, which is a simple data-interchange format designed for communication of data between systems, and can be easily parsed into a native PHP data structure. Facebook only provides their event pages as template generated XHTML pages, which are focused on visual display for a user rather than communication of data between computer systems.

Other sources considered included MySpace, CBC Radio3 and individual venue websites. Due to the increasing overlap of data provided between Facebook and MySpace and the similarities in approach required to parse each site, MySpace was given a lower priority. CBC Radio3 has an extensive repository of events featuring lesser-known acts, but again some overlap is present with events currently available through Facebook. CBC Radio3 provides

their data in a JSON format similar to TicketMaster, and so similar approaches could be used. Due to the variance in formats of venue websites, the relative benefit of parsing such sources was deemed much lower than the return of mediums which aggregated content from multiple artists or venues. As well, since many venues are increasing their use of Facebook and other social networking sites to promote their events, the same event information can often be obtained through aggregate sources in addition to their custom websites.

4.2 Chosen Approach

The information gathered from other sites for inclusion on KelownaGigs is relatively small compared to the volume available. As such, for each site a focused information extraction module was implemented. Record-level and page-level extractors were used to either extract information for an individual item or a set of items, depending on the format returned from the data source. Each module consists of three stages: a fetcher, which retrieves the data from the remote site; a parser, which translates the retrieved data into a usable data structure; and a processor, which stores the parsed data structure in the KelownaGigs database. Additionally, the parser may find additional relevant content which the processor will then instruct the fetcher to retrieve on its next run if necessary.

Each module is coded specifically for the given source and data format, so it requires a higher level of domain knowledge in order to create and configure. They are also susceptible to changes in the retrieved data, as the parsers expect a specific structure and cannot deal with any changes or ambiguities without modification. The risk of changing data formats is relatively minor for the major data sources utilized, and so a more extensive extraction approach that did not rely on parsing a specific structure was not deemed necessary.

4.2.1 Fetcher

For the web sources utilized in this project, the fetcher primarily performs a HTTP request with necessary parameters to retrieve the required data. If data is not parsed and processed immediately, the fetcher also caches the retrieved content in the database for later access. It may optionally optimize the retrieved data by removing unnecessary portions before storage. The fetcher component is also responsible for scheduling the periodic retrieval of content from some sources.

4.2.2 Parser

The parser retrieves data of interest from the raw fetched data, and places it into a simple data structure. This can be aided by parsing the raw data itself into a utilizable data structure, as is possible with JSON, XML, and HTML data. In some cases regular expressions are also utilized to extract relevant portions of data out of a larger string. Data is simply found at this stage, and the parser has little knowledge about the data retrieved other than its format.

4.2.3 Processor

The processor handles translating the data from the format returned by the parser into the format required for storage in the database. This may include simple translation of attribute identifiers between the two sources, converting formats of the value (e.g. date format), or matching parsed values to existing items in the database. Since the parser has no knowledge of the validity or usefulness of the data it returns, it is up to the processor to determine if each value is relevant and useful and it may not store all values given to it by the parser. By separating the processing function from the parser, changes to either the source or destination format will have less effect on the system as only one function deals with each source.

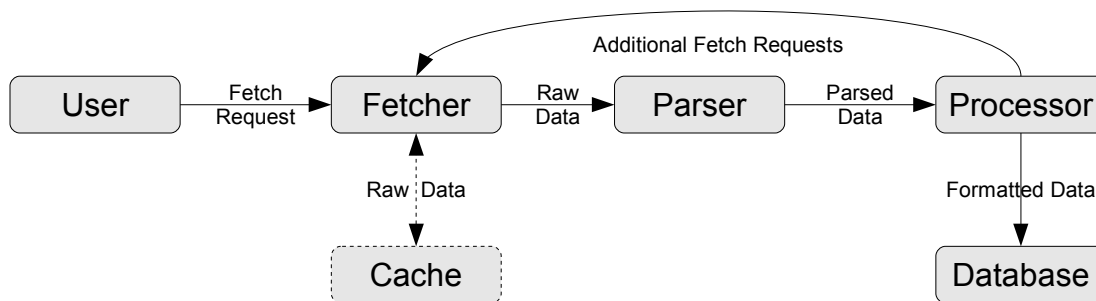


Figure 3: Information Extraction System Data Flow

4.3 TicketMaster

Due to the nature of events serviced through TicketMaster, information is infrequently provided to KelownaGigs by users of the site, and rarely in a timely manner. As a small channel KelownaGigs is overlooked by the promoters of these events, and the current site

demographic may either not be interested in the events or have no incentive to provide information.

TicketMaster uses Javascript requests to populate the pages of its site. The initial download consists only of the page framework, including the styles and Javascript code. A Javascript method is then called to fetch the relevant data for the page and format it to display to the user. The TicketMaster import module utilizes the same Javascript request methods to fetch the relevant data, which is returned in Javascript Object Notation (JSON). This is a machine readable format, which can easily be parsed with PHP, and the relevant fields can be copied into the KelownaGigs database. Since TicketMaster supplies no user-generated content, returned values can almost always be trusted to contain valid and relevant data.

```
{
  "PromoId": ["n0yd9g"],
  "VenueSEOLink": "/Kelowna-Community-Theatre-tickets-Kelowna/venue/139432",
  "VenueId": "139432",
  "LocalEventMonthYear": "December 2009",
  "PostProcessedData": {
    "SuppressWireless": true,
    "Onsales": {
      "unmodified_epdate": null,
      "expire": "Fri, 12/11/09<br>07:30 PM",
      "onsales": [
        {
          "suppress": 0,
          "onsale_type": 1,
          "interval": {
            "end": "Fri, 12/11/09<br>07:30 PM",
            "start": "Mon, 10/05/09<br>10:00 AM"
          }
        }
      ],
      "event_date": {
        "event_date_type": 5,
        "date": "Fri, 12/11/09<br>07:30 PM",
        "date_range": null,
        "suppress_time": 0,
        "onsale_status": "1"
      }
    },
    "VenueCity": "Kelowna",
    "MajorGenreId": [10002],
    "ExpirationDate": null,
    "VenueCityState": "Kelowna, BC",
    "AttractionSEOLink": ["/The-Nutcracker-tickets/artist/804130"],
    "Type": ["Event"],
    "EventType": 0,
    "LocalEventShortWeekday": "Fri",
    "EventName": "The Nutcracker",
    "AttractionId": [804130],
    "ArtsBrowseGenre": ["All Arts & Theater", "Ballet and Dance"],
    "LangCode": "en-us",
    "AttractionName": [The Nutcracker],
    "OnsaleOn": "2009-10-05T17:00:00Z",
    "LocalEventShortMonth": "Dec",
    "VenuePostalCode": "V1Y 1J4",
    "DMAId": [518, 521],
    "SearchableUntil": "2009-12-12T07:59:59Z",
    "AttractionImage": [""],
    "MinorGenreId": [12],
    "LocalEventWeekdayString": "Friday",
    "Host": "VAN",
    "EventDate": "2009-12-12T03:30:00Z",
    "LocalEventDateDisplay": "Fri, 12/11/09<br>07:30 PM",
    "LocalEventDay": 11,
    "PresaleOff": ["2009-10-05T06:00:00Z", "2009-10-05T06:00:00Z", "2009-10-05T06:00:00Z"],
    "Timezone": "America/Los_Angeles",
    "LocalEventYear": 2009,
    "PresaleOn": ["2009-09-16T17:00:00Z", "2009-09-16T17:00:00Z", "2009-09-16T17:00:00Z"],
    "MinorGenre": ["Ballet and Dance"],
    "MarketId": [110],
    "timestamp": "2009-11-30T14:55:35.59Z",
    "DocumentId": "Event+1100432A8DBB9095+en-us+1",
    "LocalEventMonth": 12,
    "VenueAddress": "1375 Water Street",
    "VenueName": "Kelowna Community Theatre",
    "VenueImage": "/dbimages/kct_van_139432_4Z.gif",
    "Id": "1100432A8DBB9095",
    "Genre": ["Ballet and Dance"],
    "VenueCountry": "CA",
    "MajorGenre": ["Arts & Theater"],
    "VenueState": "BC",
    "EventId": "1100432A8DBB9095",
    "OnsaleOff": null,
    "search-en": "The Nutcracker Kelowna BC British Columbia Kelowna Community Theatre December 2009 Friday V1Y 1J4 Ballet and Dance thenutcracker"
  }
}
```

Figure 4: TicketMaster Query Results

Information to be extracted is bold

The parser and processor components of the module are tightly coupled, providing a simpler implementation at the risk of requiring more extensive changes if either component requires modification. The first step in parsing is to transform the JSON encoded string into a native PHP data structure through a built in decoding function. Within the resulting data structure is an array of event objects related to the requested artist or venue. Each item has its genre and location values checked for validity (e.g. Figure 4 has a 'MajorGenre' value of "Arts & Theatre", and so would not be imported), and then its values are mapped to an event object for storage on KelownaGigs. Unique identifiers are included within the retrieved data, such as the `attractionId` attribute which corresponds to the `attractionName` attribute, allowing the processor to match content based on these unique ids if available rather than on string content which may contain variations. If matching on identifiers fails, such as the `attractionId` representing an artist that is not known in the KelownaGigs system, it falls back to content matching. If the content can be matched to an existing item in the KelownaGigs database the existing item is updated with the unique identifier, otherwise it is assumed that it is new data and a new database entry is created with the known information including its unique identifier and name. The next time the automated fetcher runs it will also include the new data items in its queue.

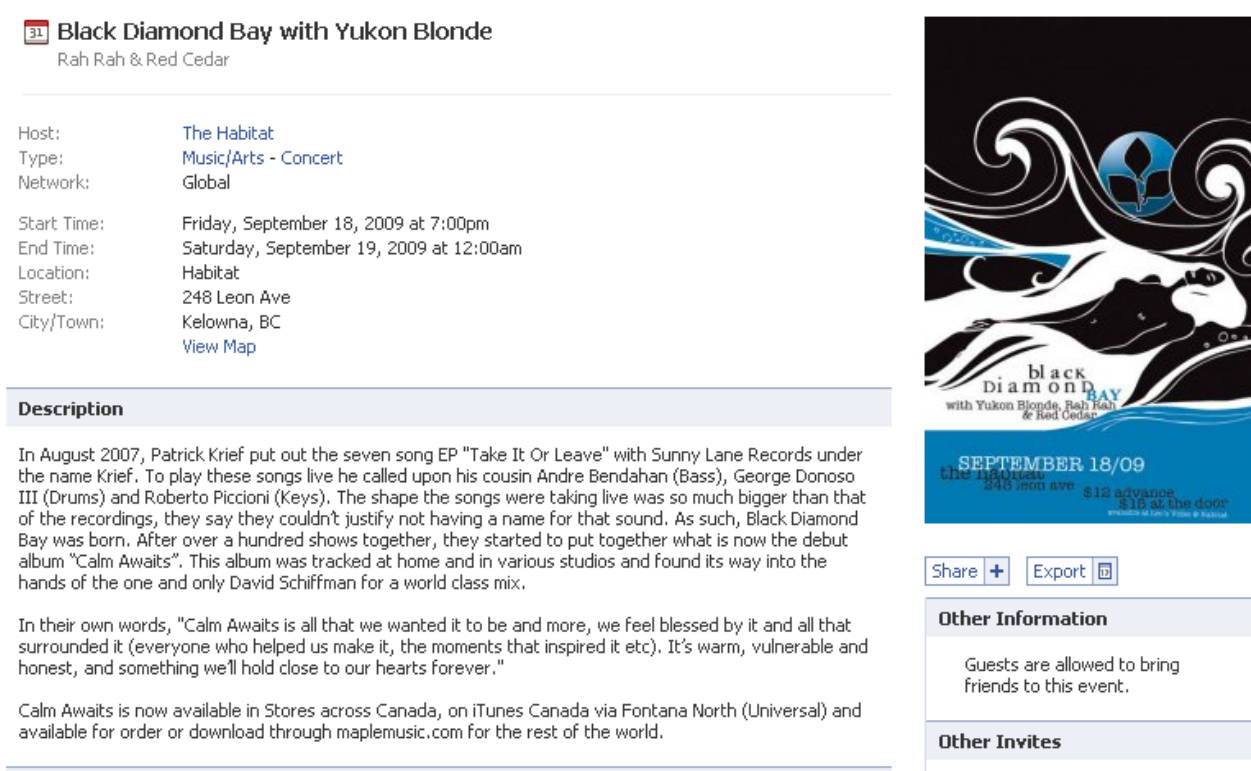
```
'EventId' => '1100432A8DBB9095'  
'MajorGenreId' => [10002]  
'MajorGenre' => ['Arts & Theater']  
'EventDate' => '2009-12-12T03:30:00Z'  
'VenueId' => '139432'  
'VenueCity' => 'Kelowna'  
'VenueAddress' => '1375 Water Street'  
'VenueName' => 'Kelowna Community Theatre'  
'VenuePostalCode' => 'V1Y 1J4'  
'VenueState' => 'BC'  
'VenueCountry' => 'CA'  
'AttractionId' => ['804130']  
'AttractionName' => ['The Nutcracker']
```

Figure 5: TicketMaster Event Object Properties

Items are in the order that they are parsed and processed from the JSON object. Values surrounded in square brackets indicate that multiple values may be present.

4.4 Facebook

As a medium built on user-generated content, Facebook provides additional challenges not present in parsing the TicketMaster feeds. All pages on Facebook are rendered in well-formed XHTML which aids in the parsing of content fields, but the use of user-generated content complicates the extraction of data from those fields. As an example, while the date of the event can easily be found by searching for its label and the date itself will be displayed in a known format, there is no consistent or fixed way to display the artists performing. Facebook provides three free-form fields where the user may input the data: the title, tagline, and description. While some events are listed with all of the bands in the title, others have a name for the event, with the artists in the tagline. Others split up the main and supporting acts between the title and tagline.



The screenshot shows a Facebook event page for "Black Diamond Bay with Yukon Blonde" hosted by "Rah Rah & Red Cedar". The event details include the host, type (Music/Arts - Concert), network (Global), start and end times (September 18-19, 2009), location (Habitat, 248 Leon Ave, Kelowna, BC), and a "View Map" link. The description section contains two paragraphs: the first describes the band's history and their debut album "Calm Awaits"; the second is a quote from the band about their music. Below the description are "Share" and "Export" buttons. The "Other Information" section states that guests are allowed to bring friends. The "Other Invites" section is partially visible at the bottom.

Black Diamond Bay with Yukon Blonde
Rah Rah & Red Cedar

Host: [The Habitat](#)
Type: [Music/Arts - Concert](#)
Network: [Global](#)

Start Time: [Friday, September 18, 2009 at 7:00pm](#)
End Time: [Saturday, September 19, 2009 at 12:00am](#)
Location: [Habitat](#)
Street: [248 Leon Ave](#)
City/Town: [Kelowna, BC](#)
[View Map](#)

Description

In August 2007, Patrick Krief put out the seven song EP "Take It Or Leave" with Sunny Lane Records under the name Krief. To play these songs live he called upon his cousin Andre Bendahan (Bass), George Donoso III (Drums) and Roberto Piccioni (Keys). The shape the songs were taking live was so much bigger than that of the recordings, they say they couldn't justify not having a name for that sound. As such, Black Diamond Bay was born. After over a hundred shows together, they started to put together what is now the debut album "Calm Awaits". This album was tracked at home and in various studios and found its way into the hands of the one and only David Schiffman for a world class mix.

In their own words, "Calm Awaits is all that we wanted it to be and more, we feel blessed by it and all that surrounded it (everyone who helped us make it, the moments that inspired it etc). It's warm, vulnerable and honest, and something we'll hold close to our hearts forever."

Calm Awaits is now available in Stores across Canada, on iTunes Canada via Fontana North (Universal) and available for order or download through [maplemusic.com](#) for the rest of the world.

[Share](#) [Export](#)

Other Information

Guests are allowed to bring friends to this event.

Other Invites

Figure 6: Public View of a Facebook Event

In this example performing artists are present in both the title (Black Diamond Bay, Yukon Blonde) and tagline (Rah Rah, Red Cedar). The artists are also separated by different delimiters in each instance ('with' and '&').

Content can be imported from Facebook in two ways: a form is available to any site user to import the event, and a bot runs in the background to parse Pages and Groups for events. In both instances the event page is fetched and cached. If a user enters an event to import, they are immediately taken to a form for creating an event listing, with the parsed values populated into the relevant fields. A database table stores associations of existing content on KelownaGigs to Pages and Groups on Facebook, which the spider uses as crawling targets. If the spider finds any events, they are added to a queue which an administrator can use to import the events in the same manner as any other user. The queue displays a link for the administrator to view the event on Facebook, and then either import the event's information or ignore the event and mark it as not to be imported.

Figure 7: Event Import Form

Event ID	Title	View	Import	Ignore
296733497193	Magic 4 Haiti	View	Import	Ignore
330662974789	Sherman Doucette - ON YOUR COMPUTER...LIVE!	View	Import	Ignore
334029587877	Joel Strauss - ON YOUR COMPUTER...LIVE!	View	Import	Ignore
348896057867	Adaline- ON YOUR COMPUTER...LIVE!	View	Import	Ignore
350481065821	Share - ON YOUR COMPUTER...LIVE!	View	Import	Ignore
352965012730	Ian Kelly with Camaromance - ON YOUR COMPUTER...LIVE!	View	Import	Ignore

Figure 8: Administrator Facebook Import Queue

4.4.1 Fetcher

The fetcher consists of two components to either retrieve a page for a single event or a listing page containing partial information for many events. The fetcher utilizes built-in methods of Drupal to retrieve an HTTP request for the event page. The returned text is cleaned up to remove unneeded elements, such as Javascript and style sheets, and cached in the database. This reduces the network requirements of the fetcher as the content only has to be obtained once, and subsequent requests can be served locally from the database. This also improves performance for site users since items already in the import queue have had their content

cached by the spider, and when a user later imports the event through either interface no network requests are required.

```
...
<div class="event_profile_title"><h3>Black Diamond Bay with Yukon Blonde</h3>Rah
Rah & Red Cedar </div><div class="event_profile_information"><table
id='Event Info' class="profileTable info_table" cellpadding="0" cellspacing="0">
<tr><td class="label">Host:</td>
<td class="data"><div class="datawrap"><a
href="http://www.facebook.com/group.php?gid=4807438539">The
Habitat</a></div></td></tr>
<tr><td class="label">Type:</td>
<td class="data"><div class="datawrap"><a href="/search/?
o=4&sfxp=1&c1=5">Music/arts</a> - <a href="/search/?
o=4&sfxp=1&c1=5&c2=4">Concert</a></div></td></tr>
<tr><td class="label">Network:</td>
<td class="data"><div class="datawrap">Global</div></td></tr>
</table>
<table id='Time and Place' class="profileTable info_table" cellpadding="0"
cellspacing="0">
<tr><td class="label">Start Time:</td>
<td class="data"><div class="datawrap">Friday, 18 September 2009 at
19:00</div></td></tr>
<tr><td class="label">End Time:</td>
<td class="data"><div class="datawrap">Saturday, 19 September 2009 at
00:00</div></td></tr>
<tr><td class="label">Location:</td>
<td class="data"><div class="datawrap">Habitat</div></td></tr>
<tr><td class="label">Street:</td>
<td class="data"><div class="datawrap">248 Leon Ave</div></td></tr>
<tr><td class="label">Town/City:</td>
<td class="data"><div class="datawrap">Kelowna, BC</div></td></tr>
...
```

Figure 9: Facebook Event XHTML Code Sample
(see Figure 6: Public View of a Facebook Event)

The fetcher also has to deal with the possibility of transfer errors and content not being accessible to unauthenticated users. In such cases an error is presented to the user indicating that the content requested cannot be accessed and they will have to enter it manually. The database stores if a retrieval request failed due to authentication issues in order to prevent the system from repeatedly trying to access restricted content.



Displaying events for Habitat.

[Back to page](#)

10 April



Yukon Blonde

Type: [Music/arts - Concert](#)
Where: [Habitat](#)
When: [10 April from 20:00 to 00:00](#)
Your RSVP: [Maybe Attending \(edit\)](#)

[View guest list](#)
[Remove from my events](#)

14 April



Matthew Barber

Type: [Music/arts - Concert](#)
Where: [Habitat](#)
When: [14 April from 20:00 to 00:00](#)

[RSVP](#)
[View guest list](#)

16 April



Jon and Roy

Type: [Music/arts - Concert](#)
Where: [The Habitat](#)
When: [16 April from 20:00 to 00:00](#)
Your RSVP: [Maybe Attending \(edit\)](#)

[View guest list](#)
[Remove from my events](#)

02 May



Plants and Animals

with [Said The Whale](#)

Type: [Music/arts - Concert](#)
Where: [Habitat](#)
When: [02 May from 20:00 to 00:00](#)
Your RSVP: [Maybe Attending \(edit\)](#)

[View guest list](#)
[Remove from my events](#)

[Show Past Events...](#)

Figure 10: Public View of a Facebook Page's Upcoming Events

```

<div class="timeline">10 April</div><div class="partyrow"><table><tr><td
class="tunaimage"><a href="/event.php?eid=105198302847152&index=1"></a></td><td class="info"><table class="infotable"><tr><td class="eventtitle"
colspan="2"><h3><a href="/event.php?eid=105198302847152&index=1"
class="etitle">Yukon Blonde</a></h3></td></tr><tr valign="top"><td
class="label">Type:</td><td><a href="/search/?
o=4&sfxp=1&c1=5">Music/arts</a> - <a href="/search/?
o=4&sfxp=1&c1=5&c2=4">Concert</a></td></tr><tr valign="top"><td
class="label">Where:</td><td>Habitat</td></tr><tr valign="top"><td
class="label">When:</td><td>10 April from 20:00 to
00:00</td></tr></table></td><td class="actions" nowrap="nowrap"><a
href="/ajax/social_graph/dialog/popup.php?id=105198302847152" rel="dialog">View
guest list</a><a href="/ajax/events/remove_confirm.php?eid=105198302847152"
rel="dialog">Remove from my events</a></td></tr></table></div>

```

Figure 11: Facebook Event List XHTML Code Sample

(see Figure 10: Public View of a Facebook Page's Upcoming Events)

4.4.2 Parser

The Facebook parser utilizes PHP's DOMDocument class to parse the fetched page into a DOM data structure. Since no modifications to the DOM object are performed, it is translated into a SimpleXML object to provide easier querying on the structure through XPath queries. The parser attempts to clean up and divide the title and tagline on common separators and stores them in an array of potential artists. It is then left to the processor to match these potential values against actual values in the database. Figure 12 shows an example of extracting the event's date and time from the XHTML page. Initially, the table cells containing the desired information are extracted using known structure (body and div elements) and attributes (@id and @class qualities). Further queries obtain the desired piece of information by utilizing the known content labels ('Date:' and 'Time:'). The final step for the time is to use a regular expression to extract a single value, as often it contains a time range with two values. All entries are stored in an array as strings to be passed to the processor.

```

$xmlnodes = $sxml->xpath(
    "body//div[@id='content']//div[@class='event_profile_information']/table"
);
$date = $xmlnodes[1]->xpath("./tr[td='Date:']/td/div");
$data['date'] = (string) $date[0];
$time = $xmlnodes[1]->xpath("./tr[td='Time:']/td/div");
if($time){
    preg_match('/([\d]{1,2}:[\d]{2}(:[ap]m)?)/i', (string) $time[0], $matches);
    $data['time'] = $matches[1];
}

```

Figure 12: Parsing the Event Date from XHTML via XPath and Regular Expressions

4.4.3 Processor

The processor maps the parsed data values to fields in the event creation form and attempts to match the potential artists found from the parser against artists already existing in the KelownaGigs database. If a complete match is available it is used; otherwise the user is prompted to either enter the data themselves if it is valid, or ignore it if not. The 'Location' field is also matched against data stored within the database. If no matches are found, the ID located in the 'Host' field link, as well as the link text, are used to match against database values if possible. An example extraction where all items were matched is shown in Figure 14, whereas in Figure 15 “On your computer... Live” followed the artist in the title field of the event, separated by a hyphen, and did not match a known item.

Once the event information has been processed and stored within the KelownaGigs database, the cache record is updated to include the unique content id, and the cached page data is flushed. If another user attempts to import the same event, it will be found in the cache table as already imported, and the user will be shown a message providing a link to the existing event listing as in Figure 13. The system does not currently check for updates to event information after the initial import, instead relying on user feedback to notify of event changes.



Figure 13: Attempting to Import an Event that has Previously Been Imported

Please confirm the event information and click 'Save'

Create Event

Title:

eg., CD Release

Date: *

Format: March 14, 2010

Venue:

Type part of the venue's name to view some suggestions. For best results, do not include 'the' in your search.

If a listing does not currently exist for the venue, one will be created when you submit the event information.

Artists:

Type part of the artist's name to view some suggestions.

If a listing does not currently exist for the artist, one will be created when you submit the event information.

ADD ANOTHER ITEM

Door Cost:

Tickets:

Ticket Locations:

All Ages?:

N/A Yes No

Select if persons under the legal drinking age will be admitted to this event

Facebook Event:

TicketMaster ID:

Enter the ID for this event from the link to its webpage on TicketMaster.com

Poster:

Maximum Filesize: 1 MB

Allowed Extensions: png gif jpg jpeg

Images must be larger than 100x100 pixels

Additional Information:

Figure 14: Form Populated with Parsed Data

Please confirm the event information and click 'Save'

» Could not find artist "*ON YOUR COMPUTER...LIVE!*". If this is an artist, please add them.

Create Event

Title:

eg., CD Release

Date: *

Format: March 17, 2010

Venue:

Type part of the venue's name to view some suggestions. For best results, do not include 'the' in your search.

If a listing does not currently exist for the venue, one will be created when you submit the event information.

Artists:

Type part of the artist's name to view some suggestions.

If a listing does not currently exist for the artist, one will be created when you submit the event information.

ADD ANOTHER ITEM

Door Cost:

Tickets:

Ticket Locations:

All Ages?:

N/A Yes No

Select if persons under the legal drinking age will be admitted to this event

Facebook Event:

Figure 15: An Import with an Item that could not be Matched

5 Content Access

The goal of the content access system is to provide a middle ground of access between a private access system, where a content item can only be edited by one user who owns it, and a wiki, where there is no ownership and any user can edit any content. To prevent abusive access, the premise is to restrict access to users that have been authenticated in some manner, but then grant them access to any content relevant to them and their authorizations.

An existing module was utilized to allow basic access to content through a content field on each artist or venue listing. The field is only present when editing an item, and lists site users which should be granted access to modify that item's information. This field is automatically filled with the author's account when an item is created, and either a site administrator (who can edit all content) or the content's author can then grant more users access. If an author removes their account from the list, they will no longer have access to modify that piece of content.



The screenshot shows a web interface for managing access. At the top, it says "Members Accounts:". Below this is a list of two user accounts, each with a plus sign on the left and a radio button on the right. Below the list, there is a text prompt: "Select any users on this site who should have access to edit this profile." followed by a warning: "If you remove yourself from this list, you will no longer have access to edit the profile." At the bottom of the form is a button labeled "ADD ANOTHER ITEM".

Figure 16: Field for Managing Accounts with Access

Access to modify event information is more complex, since each event is relevant to more people; each event belongs to one venue and may feature several artists, each of which could have several members. No existing module thoroughly or efficiently handled inherited access control where people authorized to edit a venue or artists profile could then edit any related events. The implemented solution watches for updates on all three types of content: events, artists and venues. If an event is updated, the information for its referenced venue and artists is loaded, and any user they reference is granted access to modify the event information. If either an artist or venue is updated, all events which reference them are queried from the database, and each is put through the same update process as if the event itself was updated. This has the potential to require increasing processing overhead for each update as the amount of linked content grows, but is currently not a significant issue.

One issue that occurred during implementation is that when a request is made to save an artist or venue's information and update content permissions, the changes have not been committed to the database yet. Therefore, when the events that need to be updated are determined, they are updated based on the old data persisting in the database, rather than the new information to be saved. To solve this issue, the module which manages these permissions also implements a small cache that stores information that has not yet been saved to the database, so that other operations that require the newer information can retrieve it instead.

6 Metrics

Site metrics had been recorded prior to this project's implementation, and were continued through development. In addition to the access logs stored previously, upon release of the new site a set of surveys were made available to users.

6.1 Access Logs

In order to track user interaction with the site, Google Analytics was used to track page views. Statistics were available for the previous system, offering a consistent view of site traffic during the transition. Traffic was excluded from development machines by IP address exclusion, so to avoid skewing numbers.

As can be seen in the following graphs, there is a noticeable increase in site visitors starting when the new system was first made available in October. A significant dip in traffic occurred during the holidays, but traffic began to increase to previous levels in the first week of January. While the number of unique visitors began to decrease to previous levels in February, the number of visits did not decrease as much, indicating that visitors were returning to the site more often.

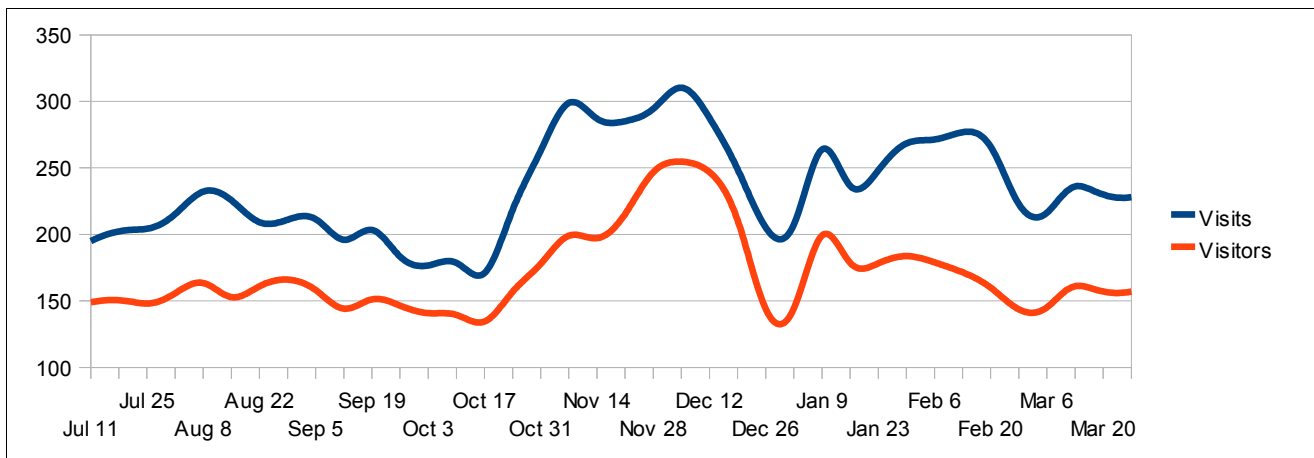


Figure 17: Weekly Visits and Unique Visitors, July through March

There is a more significant and continued increase in pages viewed per visit and average time spent on the site, indicating that users were consistently interacting with more site content on each visit. While both pages per visit and time spent on the site peaked in January, they remained at consistently higher values than prior to the site launch.

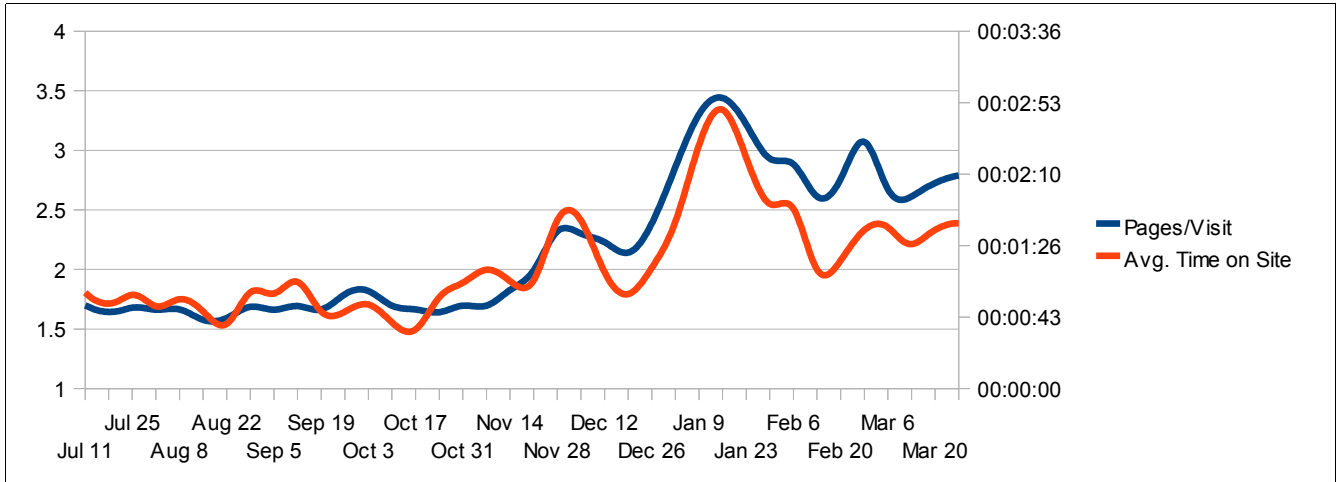


Figure 18: Pages per Visit, July through March

This is mirrored in a decrease in Bounce Rate, or the number of site visits that consist of only one page, by about 15%. With the previous system the majority of relevant content was available on the main page, and so users had little incentive to browse additional pages. The new site maintains a similar front page layout and content, but has links to significantly more content items which are stored on the new site which were previously not available, such as full artist and venue profiles.



Figure 19: Bounce Rate, July through March

6.2 Surveys

Throughout the duration of the project, surveys were available for site visitors with questions relevant to their role within the music community. A survey directed at general site users and fans included questions on how they access and spread information, and what features they would be interested in using or contributing to on the site. The final two surveys directed at artists and venues included questions on how they book and promote events, as well as new features they would like to see included on the site. All surveys included questions focused on determining the respondents use of social networking sites, and their importance in spreading information. The surveys directed at general site users and artists received approximately a dozen responses each, but the survey for venues only received one response. While the surveys provided insights into the users' usage patterns, they were obviously skewed with responses by the younger and more active members of the music community.

6.2.1 Users

Most survey respondents indicated that they were between the ages of 19 and 25, and had lived in the Okanagan more than five years. They were also very active in the music community, with the majority attending live music events for over five years. Respondents who had an account on MySpace typically had it longer than an account on Facebook, but more respondents did not have a MySpace account than did not have a Facebook account. Respondents participation in events varied, with an even amount of responses for having attended events from weekly to every couple of months. How often they visited KelownaGigs followed a similar distribution, but with increased frequencies.

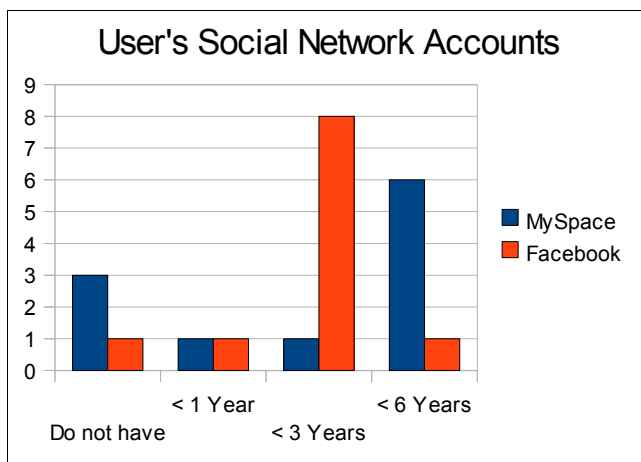


Figure 20: Duration of User's Social Network Account Memberships

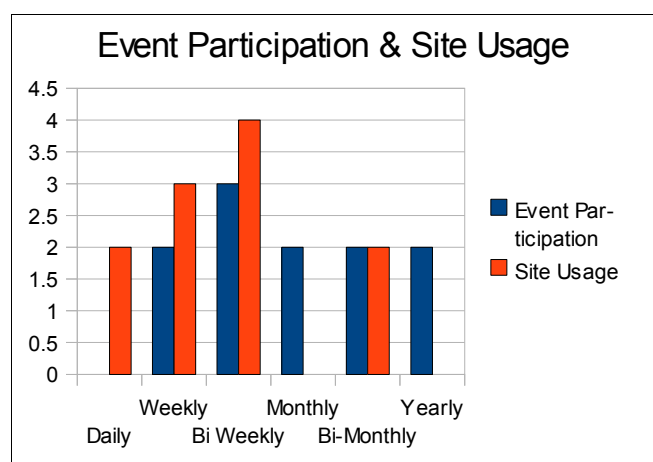


Figure 21: User's Participation in Music Events and their Usage of KelownaGigs

The primary means of finding information was through direct means such as personal communication, invitations through Facebook from friends (as opposed to receiving them from bands or venues), and posters in public spaces. More indirect methods such as event invitations from artists, and passive methods such as browsing artist's websites or profiles on other sites and newspaper or magazine listings were lesser but significant sources as well. When sharing information with others, again it was primarily through personal communication, followed by direct messages on Facebook.

While most respondents indicated that they would like to see more information on artists and photos of artists and events, less than half of those indicated that they would contribute those items to the site as well. Most respondents indicated that they would like to see interaction between KelownaGigs and their Facebook account, with fewer responding that they would like to be able to access content on KelownaGigs through a Facebook application.

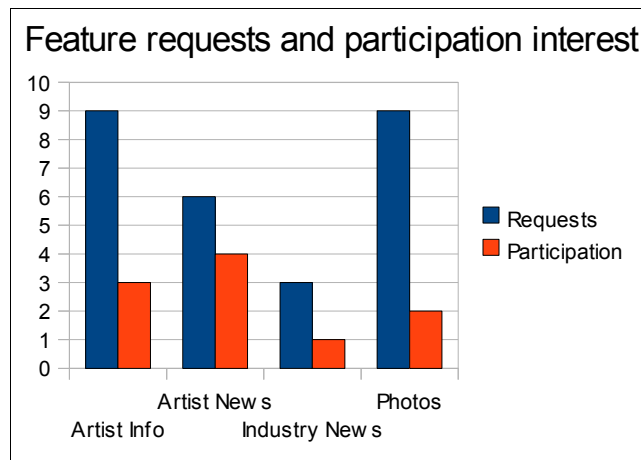


Figure 22: User's Request for Features Versus Interest in Contribution

6.2.2 Artists

Most survey respondents were Kelowna based artists who have been actively performing for more than four years. They primarily performed locally, ranging in frequency from weekly to monthly. They all used MySpace as a promotional method, and a significant number also had a Facebook Page. Less than one third had created their own websites for promotion. Most respondents have had their MySpace accounts for more than two years, whereas most created a Facebook Page or Group within the past year. The respondents were split between those who have visited KelownaGigs for more than two years and those who visited less than one year, with a significant number visiting for the first time within the past three months since the launch of the new version of the site.

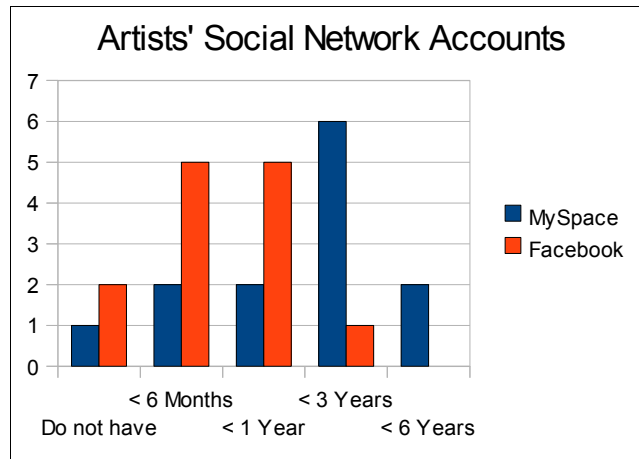


Figure 23: Duration of Artists' Social Network Account Memberships

Primary promotional methods used by artists were the creation of Facebook events listings, bulk messages through Facebook, general events listings on MySpace accounts, and physical fliers distributed to local businesses such as coffee shops and posted in public spaces. Less than half of respondents indicated that they posted event information on KelownaGigs, and only a few posted information on any other websites.

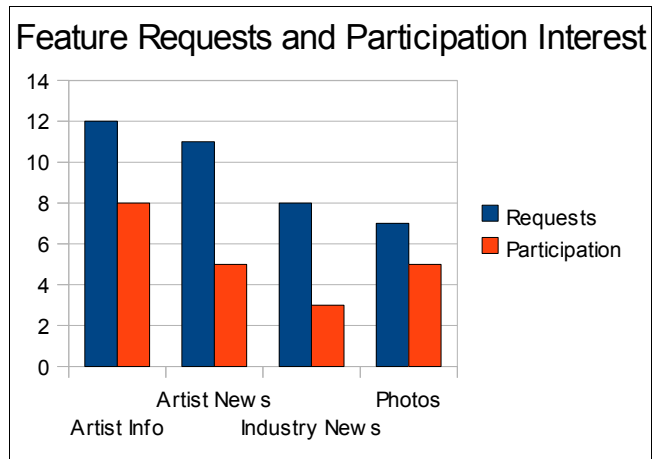


Figure 24: Artists' Requests for Features Versus Interest in Contribution

Similar to the responses by site users, artist's interest in contributing content was lower than interest in accessing it, though to a lesser degree.

6.3 Content and Users

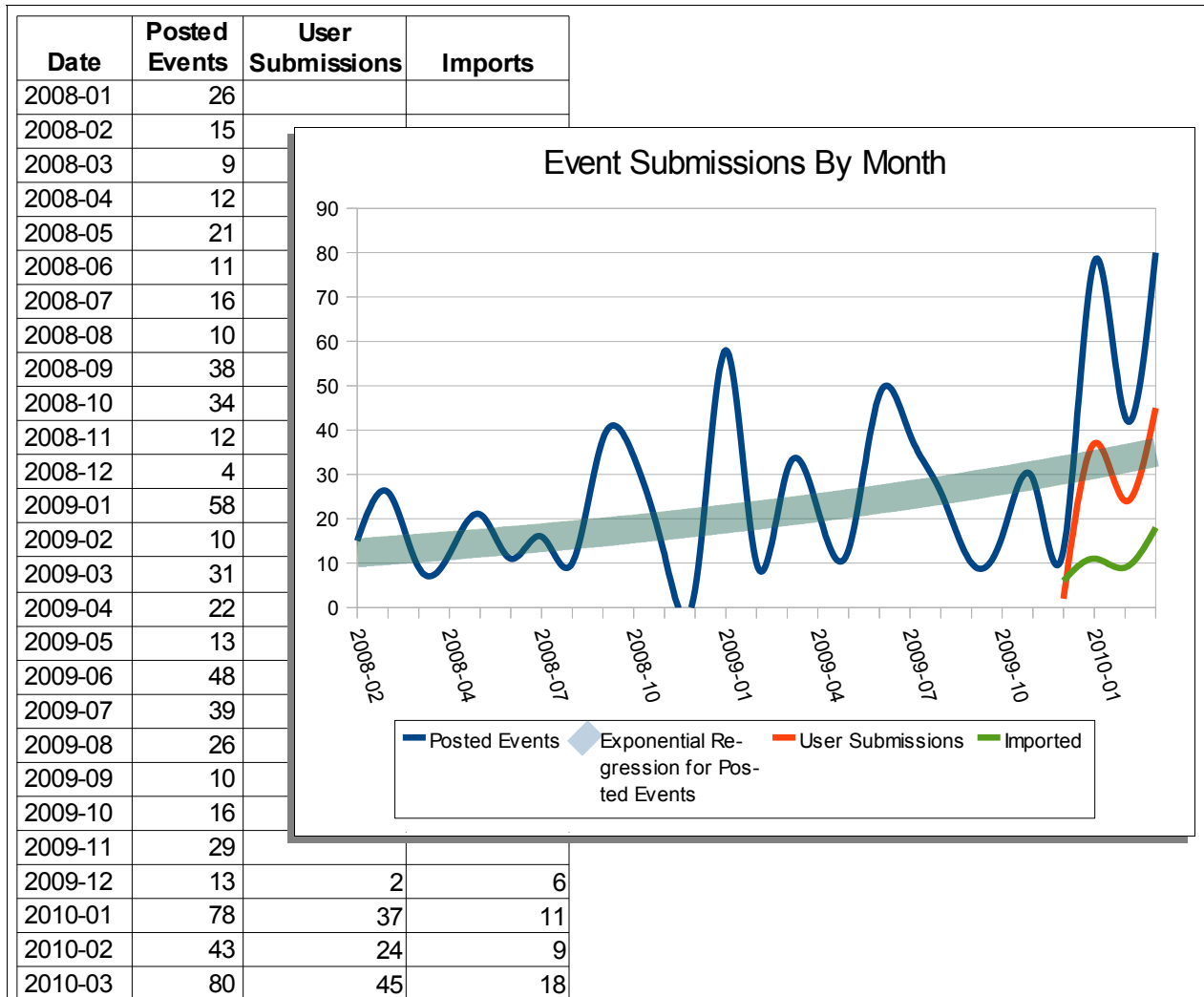


Figure 25: Events Submitted to the Site, by Month

The grey background indicates time before the new system was implemented.

Over the course of the project, a few new venues and several artists became involved in posting their own information to the website. A significant increase in events posted to the site can be seen from January through March 2010, with a large number of them being directly submitted by users. Some users did not realize that they were able to create content on the site, and so information for a few events was submitted through the site's contact form to be added to the site by an administrator. One fifth of the events posted to the site from

December through March were either automatically extracted without user intervention, or added by users with the assistance of the Facebook information extractor.

Most users registering on the site were affiliated with bands or venues, and registered in order to post events or gain access to modify their respective site listings.

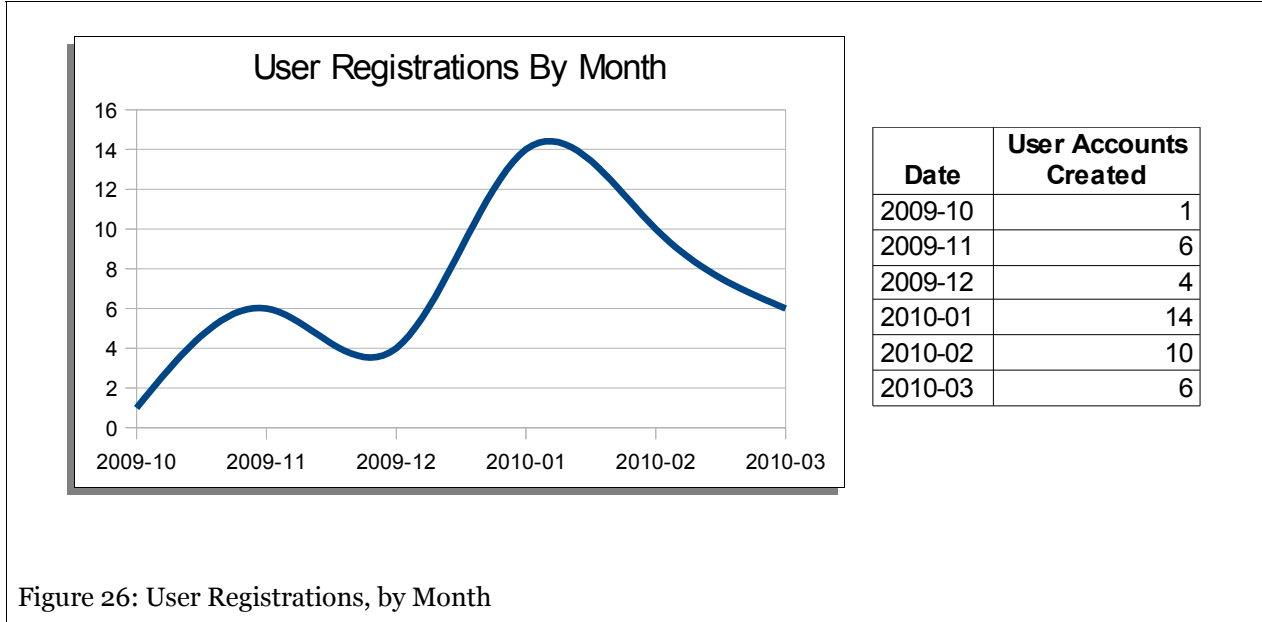


Figure 26: User Registrations, by Month

7 Discussion and Conclusions

The new site was well received at its introduction, both for its aesthetic changes and its new functionality. By allowing users to contribute their own content, as well as the automatic aggregation of content from a few sources, significantly more event listings were posted to the site from January through March 2010 than in previous months. Event submissions were concentrated from a small group of users, but produced a significant number of the total submissions.

While some aspects of user interaction were lower than expected, such as user registrations and contributions of information other than events, site traffic metrics showed both a general increase in traffic to the site and an increase in usage per individual user. One of the areas that unexpectedly received little user attention was the ability for users to comment on any content within the site. While it was apparent that some users did not realize that they could create accounts or author content items, “Add a new Comment” links were highly visible throughout the site, and did not require creating an account or logging in. New features were implemented in a relatively short time, and so it may simply take more time before users notice and make use of the new features on the site.

While some of the traffic gains were relatively small, they are encouraging progress for the site and may indicate that larger effects may be possible as the site expands to service users in more regions. Most survey respondents indicated that they used other social networking sites, and indicated interest in being able to integrate their accounts on other sites with KelownaGigs. Since social networking sites were a key medium for both general users and artists to find and share information, developing this integration could be a key feature in increasing user interaction and retention.

7.1 Future Work

The parsers for user-generated content are fairly rudimentary, and require user attention to validate their results. Currently parsed potential values are only presented to the user to validate, but the opportunity exists to use the existing knowledge within the database to automatically validate the parsed values. While this approach will be limited to recognizing matches with existing content, and new content will likely still require manual validation, by

applying a measure of certainty to the parsed results, user intervention would be required in fewer cases.

Each extractor currently only operates as a one time process and does not update existing information, instead relying on user contributions for changes. This could be improved to periodically check imported information for updates, either notifying administrators and users that changes are available or attempting to merge the updated information into the existing record.

With each parser implemented as its own module, very little code is reused between different extractors. While the Feeds module was not used as a base for the current implementations, it has the potential for a generic, configurable information extraction engine for more types of sources than currently possible. Others in the Drupal community have expressed a desire to build an XML parser for Feeds and the required tree processing capability could be used to parse other complex data formats, such as JSON, as well. These extensions could allow parsing new sources to be implemented more easily and changes to existing sources would have less effect. Automated information extraction methods could also be implemented to abstract away the structural information of data sources and make configuring the extraction tools simpler, and more accessible to non-technical users.

One of the researched methods for increasing user involvement that was not implemented included building a reputation system. Users would gain reputation points for making valuable contributions to the site, such as adding new events listings, writing articles, or posting photos. As they reach certain milestones they would be granted additional access to control site content, as well as visible recognition of their efforts. This would protect site content from negative modifications from new or anonymous users, as is a common weakness of wiki-like systems, but allow regular site users to participate in many of the moderation efforts that are currently performed by site administrators.

Many users throughout the history of KelownaGigs have expressed interest in similar sites for other regions, and for a short time a duplicate site was created to catalogue events occurring in Vernon. Reasons for choosing Drupal as the sites platform included its scalability in terms of handling more users, but also the potential for it to scale for distributing content for additional regions as well, ideally with little modification to the current design.

Current plans are to increase the regional influence of the site by aggregating more content relevant to cities throughout the Okanagan valley, and then further expand to other major regions.

References

- [1] Facebook | Timeline
<http://www.facebook.com/press/info.php?timeline>
- [2] Facebook | Statistics
<http://www.facebook.com/press/info.php?statistics>
- [3] <http://siteanalytics.compete.com/facebook.com+myspace.com/?metric=uv>
- [4] Ana M. Martinez Aleman, Katherine Lynk Wartman. **Online Social Networking on Campus**. Routledge New York, 2009. p63
- [5] Chia-Hui Chang, Mohammed Kayed, Moheb Ramzy Girgis, Khaled F. Shaalan. **A Survey of Web Information Extraction Systems**. IEEE Transactions on Knowledge and Data Engineering, Vol 18 No. 10, October 2006.
- [6] Hao Han, Takehiro Tokuda. **A Method for Integration of Web Applications Based on Information Extraction**. Eighth International Conference on Web Engineering, 2008.
- [7] <http://drupal.org>
- [8] Content Construction Kit (CCK) | drupal.org
<http://drupal.org/project/cck>
- [9] Views | drupal.org
<http://drupal.org/project/views>
- [10] FeedAPI | drupal.org
<http://drupal.org/project/feedapi>
- [11] Feeds | drupal.org
<http://drupal.org/project/feeds>
- [12] PHP: DOMDocument - Manual
<http://docs.php.net/manual/en/class.domdocument.php>
- [13] PHP: SimpleXML - Manual
<http://docs.php.net/manual/en/book.simplexml.php>
- [14] The Open Source Definition | Open Source Initiative
<http://www.opensource.org/docs/osd>