Transforming Introductory Computer Science Projects via Real-Time Web Data

Austin Cory Bart
Virginia Tech
acbart@vt.edu

Eli Tilevich
Virginia Tech
tilevich@cs.vt.edu

Clifford A. Shaffer
Virginia Tech
shaffer@cs.vt.edu

Tony Allevato
Virginia Tech
allevato@vt.edu

Simin Hall
Virginia Tech
thall57@vt.edu

Abstract

Computing is becoming increasingly distributed. To better prepare students for the challenges of the workplace, computing educators need to introduce issues pertaining to distributed computing. Unfortunately, programming projects in introductory classes are mostly divorced from the students day-to-day computing experiences. These experiences entail interacting with real-time Web-based data from sources including weather reports, news updates, and restaurant recommendations. The disconnect between student experience and the content of their programming projects is known to drive some students away from computing. To address this problem, we have created an architectural framework that makes real-time web data accessible for introductory programming projects. The framework effectively introduces important real-time distributed computing concepts without overwhelming them with the low-level details that working with such data typically requires. Our preliminary results indicate that our approach can be effective in the context of a typical CS 2 data structures class. We are currently working on extending, improving, and fine tuning our toolchain to make our framework accessible to other classes in the curriculum.

Categories and Subject Descriptors K.3.2 [*Computer and Information Science Education*]: Computer Science Education

General Terms Design, Human Factors, Reliability, Experimentation

Keywords real-time data, learning enhancement, projects, introductory

1. Introduction

Most peoples experience with computing today involves a confluence of distributed Web architectures and continuously updated, remote data sources. College students are immersed in consuming real-time Web-based data on their mobile devices. Examples abound. Weather information, traffic data, stock values, shopping deals—all these real-time data sources are being made available as remote Web-based services to be integrated into various applications written for a panoply of computing devices. Interacting with Web-based real-time data influences the students perception of what Computer Science is, the perception with which they embark on studying the discipline.

Because of these developments, computing educators strive to introduce issues related to real-time web data in the curriculum as early as possible for two reasons. First, they would like to meet their students expectations that Computer Science studies exciting topics that are relevant to the students experiences as computer users. Second, since distributed real-time data has entered the mainstream of the majority of computing domains, students should be introduced to the corresponding technical issues to prepare for the modern IT workforce. Indeed, the new ACM/IEEE Computer Science Curriculum 2013 advises 10 hours of material on a new dedicated *Networking and Communication* Knowledge Area [9].

Introducing real-time data is not only about increasing engagement. Perhaps more important, it is an avenue for exploring computing in a social context. The CS Curriculum 2013 emphasizes the importance of Social Issues [9], prescribing 16 hours on topics such as Social Context and Data Privacy. The ability to access real-world data streams enables introductory learners to deal with real-world problems. Consider assigning students to explore how geological data about earthquakes can be used to aid disaster relief. Another project might have students mine political data to find evidence of corruption. Yet another project, perhaps controversially, would involve analyzing social media data provided by services including Twitter or Facebook, to identify the cases of private data being left publicly available unintentionally. Tying actual class projects to such topics can powerfully convey computing's role in society to the introductory learner.

Alas, despite its ubiquity, distributed computing remains hard. Even the seasoned software developer finds working with network protocols and parsing binary data streams conceptually complex, particularly in the presence of overwhelming technical issues that range from handling partial failure to dealing with the distributed components evolving independently from each other. Not surprisingly, a common CS curricular design strategy is to leave the challenges of teaching distributed computing for later courses, when students will have accumulated sufficient technical expertise and tolerance for engineering non-trivial system designs.

The net effect of these curricular design choices is that the content of introductory courses remains isolated from issues pertain-

SPLASH-E '13 Oct 26–31, 2013, Indianapolis, Indiana, USA.
Copyright © 2013 ACM [to be supplied]. . . $15.00

ing to Web-based real-time data. Programming projects are particularly vulnerable to this omission. When working on programming projects, introductory students find themselves disconnected from familiar data from social media, news outlets, and local business. Instead, they find themselves tackling abstract, toy problems with limited context in their lives. Divorcing the content of programming projects from the students experiences as computer users has a debilitating effect on motivation and engagement [2]. In fact, competent students are known to leave the major, having been discouraged by the lack of relevance for what they learn [1]. This problem is particularly acute for female students, for whom real-world application has been identified as the primary motivation for studying the discipline [6]. Our vision is that introductory courses can overcome the technical barriers inherent in working with real-time web data, enabling us to introduce novel, relevant projects to students.

2. Prior Work

Insulating beginner programmers from complicated systems while still providing the ability to manipulate interesting data is not a novel concept. For example, most novice-targeted programming environments feature convenient methods for manipulating images and sounds. Indeed, Racket [7] treats images as a primitive data type, avoiding the complex, low-level programming that is typically required. While images and sounds have an obvious appeal, they lack the direct real-world connection afforded by real-time data. However, there has been little work to create similar student-oriented interfaces for real-time data sources.

Despite this, working with real-time data has been shown to have an impact on student learning. A project at Stanford in a Geological Sciences course had students working with real-time data on earthquakes [5]. Although students did not use a programmatic approach, scientific computational tools were used to analyze the data and reach conclusions. The instructor reported that students became significantly more engaged as they worked with data that had relevance to them. For instance, many students became excited when they discovered that there were earthquakes happening in their region all the time. The final assignment had students chose a city that they wanted eventually to live in and determine the geological risks of the area. The instructor reported that, even after the course had ended, students applied similar analysis to other geographical regions relevant to their lives. This assignment contextualized the learning experience for the student, and created ”a personal connection and positive affect that motivates their future learning” [5]. Corresponding projects, perhaps even using the same data source, are ripe for a programming class.

Similar projects have been used in statistics courses [3] and data mining courses [10]. Although other domains seem ready and willing to bring real-world data into the classroom, the research literature on the topic is scarce. One recent exception is a project conducted by Dr. Marc Waldman that introduces realistic open data into upper-level database courses [11]. Although the cited benefits are similar, Waldman’s work takes advantage of the complicated nature of real-world data to challenge experienced learners. While we wish to avoid scaring novices, it is also desirable to appropriately challenge more advanced users, enabling them to develop higher-level skills.

3. Our Approach

We have created a software architectural framework (named ”RealTimeWeb”) that provides introductory programming students with an easy way to manipulate distributed real-time data. Our approach offers technical scaffolding for the students to gradually ease into (or completely circumvent if appropriate) some of the most vexing complexities of distributed computing. At the heart of our project

Business Service Connects to the Yelp API to gather information about businesses.

- Search Businesses
  - **Consumes** a string representing a location.
  - **Produces** a list of businesses near that location.
- Get Business Data
  - **Consumes** a Business.
  - **Produces** a new Business with more detailed information.

Weather Service Connects to the National Weather service to get forecasts.

- Get Weather
  - **Consumes** a pair of numbers indicating a geo-coordinate.
  - **Produces** data about the current weather at that location.
- Get Forecasts
  - **Consumes** a pair of numbers indicating a geo-coordinate.
  - **Produces** a list of forecasts for that location.

Reddit Service Connects to the link-sharing site Reddit to get user-submitted content.

- Get Posts
  - **Consumes** Consumes an optional string indicating the subreddit to filter by.
  - **Produces** Produces a list of the top Posts in that subreddit.
- Get Comments
  - **Consumes** a Post.
  - **Produces** a heirarchical list of comments associated with that post

Figure 1. Sample Client Libaries

are carefully engineered client libraries through which students can access the data provided by real-time web services. These libraries are readily available through an online curated gallery, designed to be quickly adapted to instructors specific academic needs. This gallery also provides a tool for rapidly prototyping new libraries based on our framework.

3.1 Client Libraries

To connect students to real-time data sources, we have designed client libraries with features intended for novice programmers. Each library offers a selection of method calls to request, parse, and return real-time data. Presently, we have created libraries to provide business reviews, weather forecasts, and content from the link-sharing site Reddit. Figure 1 summarizes the functionality available in the existing libraries. We have plans to create additional libraries for stock trading information, publicly released political data, and sports statistics. In theory, any publicly available real-time data source can be targeted by our framework.

3.1.1 Scaffolding

The instructor can determine how data is returned, so as to to provide varying levels of scaffolding. Choices include raw strings of data, semi-structured hashes and lists, or object-oriented classes. In this way, how much work students must perform to manipulate the data can be carefully chosen. This means that the libraries can be used in a wide variety of educational scenarios.

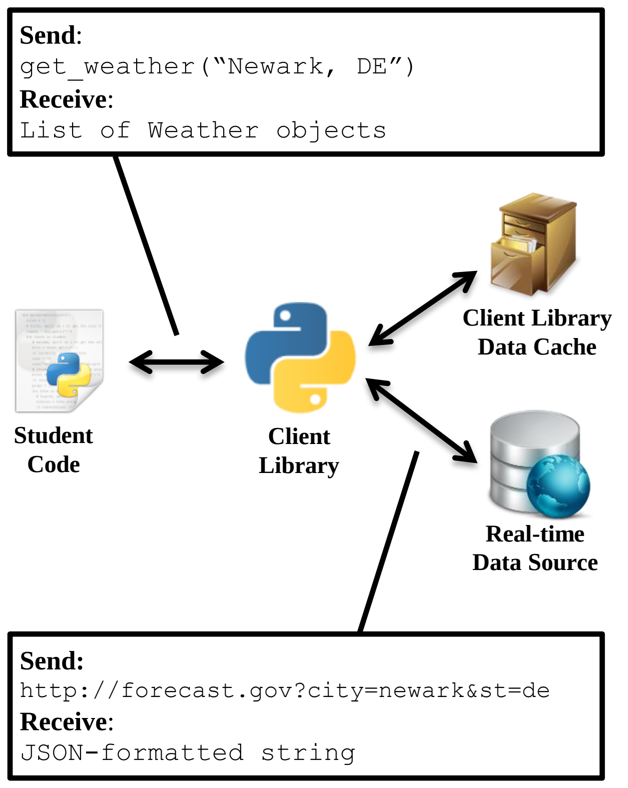

Figure 2. Client Library Architecture

3.1.2 Language

As the Computer Science Curriculum 2013 Ironman Draft [9] points out, there are many different programming languages used in introductory courses. To account for this, we implemented each library in a number of common beginner languages, including Python, Java, and Racket. We have also made an effort to provide compatibility on key platforms, including Android.

3.1.3 Threading

To address the needs arising in an assortment of situations, the libraries offer the flexibility of returning the results of API calls either synchronously or asynchronously. Concurrent programming, although an important skill for students to develop, can be overwhelming for beginners. Even as work to integrate multiprogramming into lower-level courses moves forward, scaffolding is still required if we want intro students to develop applications with parallel processing capabilities. Using the asynchronous return mode, the hard problems of threading can be avoided.

3.1.4 Internalized Data Cache

Perhaps the single most useful feature of the libraries is the *internalized data cache*. The cache can be used to avoid making requests directly to the actual data source, instead accessing a local, static data store. Figure 2 demonstrates the flow of data when using a client library. The cache option offers a number of advantages.

Idempotency By its nature, real-time data is subject to rapid temporal changes. Weather forecasts change on a daily basis, for example, and services like Reddit or Twitter change by the minute. Developing a program that uses such volatile data can be tricky, since it can change between runs of the program. To accommodate introductory students, the libraries make it possible to used cached local data, so consistency of input from run to run is guaranteed.

Consistency The content and structure of the web is highly dynamic, as that of the web services composing it. Because service APIs commonly evolve at a dizzying pace, the libraries that depend on them must be updated accordingly. Even during a single semester, changes in the data source could be introduced that would result in an out-of-date library. However, this problem can be bypassed by keeping to the internal data cache until an updating fix is released, avoiding any serious delays in development.

Connectivity Although most campuses have gigabit internet connections, this is not universal. Additionally, many students live in off-campus settings with varying internet capabilities. Requiring students to develop an application in these settings can be a tall order. However, the the cache allows client libraries to be used offline, if necessary.

Efficiency Even assuming a fast and stable connection, the performance of many web services is limited by the number of connections that the server can handle. Student developers can be polite consumers by developing with client-side data, greatly reducing the number of calls to the online service. This is especially important since many APIs throttle the number of requests (e.g. 100 API calls per day).

Tests Instructors can use the cached data for testing, ensuring uniform coverage by all students. There is no need to worry about students missing out on edge cases, since they can be provided with the important cases. Since the cache is stored in a convenient JSON-based data format, it can easily be modified by instructors to return specific results. The ability to ensure consistent tests is particularly useful for automated grading systems, such as Web-CAT [4].

3.1.5 Open-source

The libraries are meant to be good examples of API design. As students gain mastery, they can be encouraged to read the source code to learn how network communication and data parsing is implemented. They can then modify, extend, and even re-implement the API as they see fit. If students choose to create their own API based on our model, they can submit it to our gallery and gain wider recognition for their work.

3.1.6 Examples

Figure 4 demonstrates using the Java version of the Reddit library. This sample program monitors new posts for a given keyword, and then updates when a title of interest appears. Note that if line 25 had been omitted, the library would have instead returned results from its internal cache. Figure 5 shows an example of using the Racket version of the Weather library. This program will check the weather forecast for a user-specified location, and then graphically render the expected temperatures as colored circles.

3.2 Curated Gallery

The completed libraries are showcased at `http://research.cs.vt.edu/vtspaces/realtimeweb/`. Besides the various language bindings available for a service (e.g. Python, Racket, Java), there are a number of other useful pieces of information:

- API documentation and student-oriented user guides for each language,

- alternative datasets for the internalized data cache (e.g., instead of business reviews from around Blacksburg, VA, there might be another dataset for Indianapolis, IN)
- Example assignments that use the library.

3.3 Prototyping Tool

An important goal for our project is to provide an online tool for rapidly prototyping new libraries. Most of the code used in our libraries follows the same pattern for any given language. First, requests are made to a web service and raw data is returned (typically as XML or JSON). Next, the data is parsed into some intermediary, semi-structured form using dictionary and list types that are native to the language. Finally, the data is encoded into a class or struct, depending on the disposition of the language. For example, beginner students using Racket might deal with structs, instead of classes. This is not true in object-oriented languages such as Python and Java. Because the data flow is consistent regardless of programming language or data source, we can leverage this similarity to fill out a template for the target languages based on a single, abstract meta-description (a "Client Library Specification"). Our current version of the prototyping tool enables a user to create and edit a Specification, from which it can generate corresponding, prototypical client libraries for Racket, Python, and Java. Ultimately, this tool will be a component of the gallery to allow instructors to quickly create new services for their students.

3.4 Using RealTimeWeb

The labelled arrows in figure 3 demonstrate how the complete RealTimeWeb framework is used by instructors and students.

1. A *Developer* (potentially an *Instructor*) **(1) designs** a new Client Library Specification based on a *Real-Time Data Source*.
2. The *RealTimeWeb Generator website* uses this Specification to **(2) generate** Client Libraries for the desired target languages (e.g. Python, Racket, Java, etc.). Developers can fine-tune these libraries as needed for the specific service, perhaps adding post-processing on API results or more complicated authentication.
3. The Developer can use one of the Client Libraries to collect and **(3) cache** data from the Data Source.
4. The finalized libraries, the default cache, and any alternative caches are **(4) submitted** to the gallery.
5. An *Instructor* can **(5) assign** a specific service, library, and data cache to *Students*.
6. Students **(6) use** the library and cache for a class project involving the data source.
7. The Client Library handles all **(7a) access** to the Data Source, insulating Students from complicated network protocols.
8. Alternatively, students can direct the Client Library to **(7b) access** the *Local Cache* instead of the online Data Source, in order to work offline.

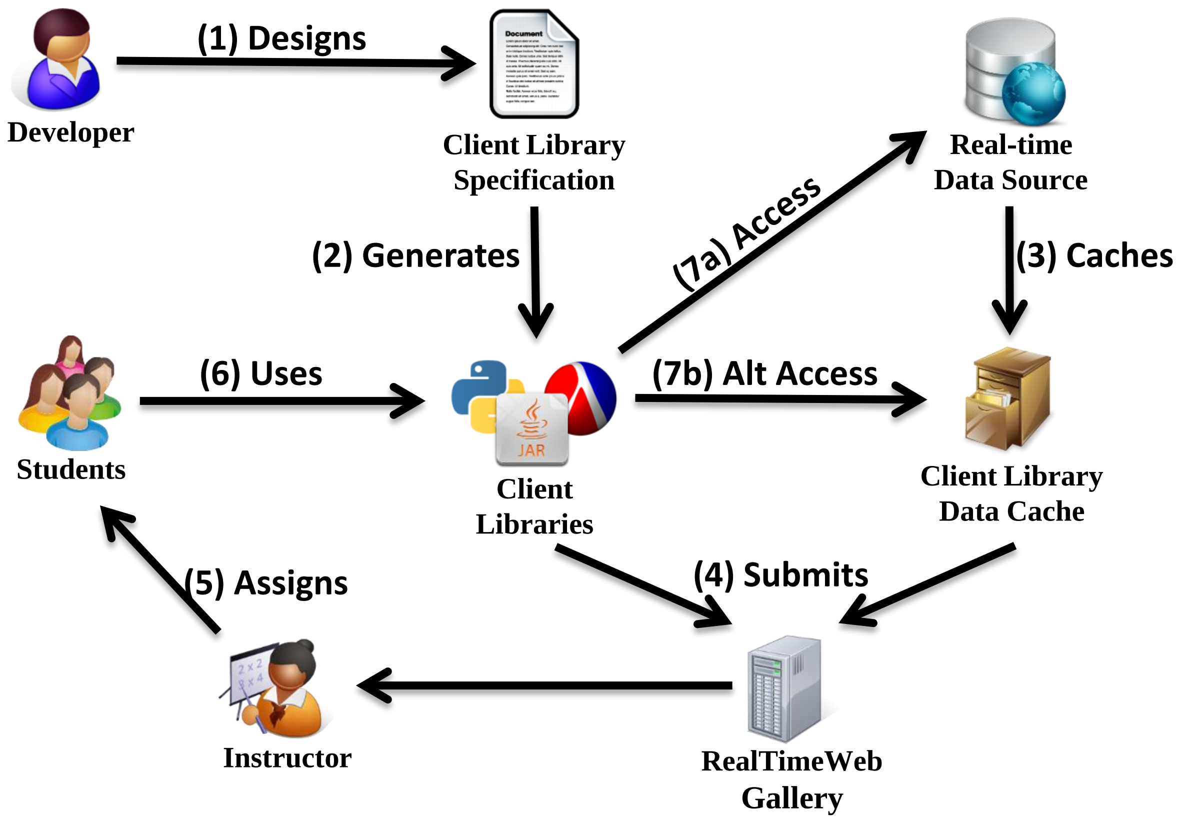

Figure 3. RealTimeWeb Architecture

Figure 4. A simple program demonstrating the Java Reddit library

```
package RedditServiceExample;

import java.util.List;
import java.util.HashSet;
import java.util.Scanner;
import realtimeweb.redditservice.main.RedditService;
import realtimeweb.redditservice.domain.Post;
import realtimeweb.redditservice.exceptions.RedditException;

public class RedditDemo
{

    public static void main(String[] args) throws RedditException
    {
        Scanner sc = new Scanner(System.in);
        System.out.println(”Enter a keyword”); // read user input
        String keyword = sc.nextLine().toLowerCase();

        trackKeyword(keyword);
    }

    public static void trackKeyword(String keyword) throws RedditException
    {
        RedditService rs = RedditService.getInstance(); // Create main object
        rs.connect(); // Use the actual Reddit service, not the local cache

        HashSet<Post> seenPosts = new HashSet<Post>();
        while ( true ) {
            List<Post> newPosts = rs.getPosts(); // Get a list of the top posts
            for (Post p : posts) {
                if (!seenPosts.contains(p.getId())) { // Check if it's new
                    String postTitle = p.getTitle().toLowerCase();
                if (postTitle.contains(keyword)) { // Check if it has our keyword
                    System.out.println(”New post with keyword: ” + p);
                    seenPosts.add(p.getId()) // Add it to seen posts
                }
            }
        }
        Thread.sleep(5000) // wait for 5 seconds
        } // while
    }
}
```

4. Courses Project Transformation

When integrated into a class project, our tools offer several features that positively impact student engagement. First, by offering libraries for a multitude of services, students have greater *autonomy* to control the general direction of their project. Second, this means that students can choose a data source that they find *relevant* to their interests. Third, the simple design of the libraries fosters a sense of *self-efficacy* and *competence* within the student. These aspects are all recognized as increasing intrinsic motivation and engagement [8].

However, using real-world data does more than just engage students and increase their intrinsic motivation. Instructors can use the libraries to introduce real-world problems. Example project ideas include:

- Reddit Service: Social link-sharing website Reddit aggregates interesting content from around the web, including news and other real-time data sources. Students could be tasked with processing data from the site and finding interesting patterns in comments and posts. Even with only a rudimentary knowledge of string parsing, there are many interesting operations that can be performed, such as analyzing average comment length or finding a keyword.
- Weather Service: Planning a long-distance trip can be tricky, and preparing for weather is important. Students could investigate how to avoid bad weather between cities using different search algorithms to find best possible paths.
- Business Service: Business reviews can be a useful way to find out how good a restaurant is. Students could combine price data with user-reported ratings in order to make a restaurant suggestion application.

As a specific example of the potential educational benefits that RealTimeWeb can afford, consider two different sketches for a programming project that can be assigned to reinforce the topic of circular linked lists, a fixture of a typical data structures course.

Figure 5. A simple program demonstrating the Racket Weather library

```
; "weather.rkt" is the Racket Weather library
(require "weather.rkt")
(require 2htdp/image)

; 16 is a nice size
(define SIZE 16)

; forecast->color: forecast -> color
; Creates a color based on the temperature of the forecast.
; The higher the temperature, the redder the color.
(define (forecast->color a-forecast)
    (make-color (forecast-temperature a-forecast) 0 0))

; forecast->circle: forecast -> image
; Creates a solid circle with a color based on the temperature.
(define (forecast->circle a-forecast)
    (circle SIZE 'solid (forecast->color a-forecast)))

; forecasts->circles: non-empty-list-of-forecasts -> image
; Creates a series of circles based on a list of forecasts.
(define (forecasts->circles nelo-forecasts)
    (cond [(empty? (rest nelo-forecasts))
            (forecast->circle (first nelo-forecasts))]
           [(cons? (rest nelo-forecasts))
            (beside (forecast->circle (first nelo-forecasts))
            (forecasts->circles (rest nelo-forecasts)))]))

; report->circles: report -> image
; Creates a series of circles based on a report that indicate
; the temperature change over a series of time periods.
(define (report->circles a-report)
    (forecasts->circles (report-forecasts a-report)))

; Get user input
(define LATITUDE (read))
(define LONGITUDE (read))

; Render the circles
(report->circles (get-weather LATITUDE LONGITUDE))
```

The first version, uses abstract data, while the second one uses the data provided by the Business Service library that comes as part of the current RealTimeWeb catalog.
Original Version:

1. Create a circular list data structure capable of holding heterogeneous data.
2. Create a Data class that contains an integer id and a String description.
3. Place Data objects whose id fields represent consecutive integer values into each node of the list.
4. Remove all the nodes containing Data objects, whose id fields are even numbers.

RealTimeWeb Version:

1. Create a circular list data structure capable of holding heterogeneous data.
2. Download a list of nearby restaurants and place them into the nodes of the list, sorted by their price levels.
3. Remove all the restaurants that do not offer vegetarian options.

Although both versions reinforce the same topic of circular linked lists and require a comparable amount of student effort, the RealTimeWeb version has students manipulate data they are likely to encounter in their day-to-day experiences, thus potentially increasing motivation and engagement.

It's important to recognize that integrating these services needs to be more than skin deep. Students should be given a situated perspective on how this data interacts with the real world. Rather than just downloading an abstract list of data, assignments should lead to practical, interesting tools and results that the students can be proud of.

5. Intervention

To gather formative evaluations on the tools, we piloted our Business Service client library in a CS2-style course offered at Virginia Tech (CS 2114) in Spring 2013. This course covers topics typical to a second semester Computer Science course, including Object-Oriented concepts and the Java programming language. All students in the class are Computer Science majors and minors in their first or second year, with mostly limited prior programming experience. Every week, students use paired programming to complete

Statement	Strongly Disagree	Disagree	Somewhat Disagree	Somewhat Agree	Agree	Strongly Agree	Total
"I found the content of the... lab to be applicable to my day-to-day computing experiences."	0	0	1	9	5	2	17
"The classes in the API were clear to use based on their names."	0	0	1	4	7	5	17
"The methods of the API were clear to use based on their names."	0	0	0	5	12	0	17
"Overall, the API was easy to use."	0	0	0	2	14	0	16

Table 1. Results of the CS-2114 survey on the Business Service assignment

a lab assignment. There are also three large projects meant to be completed over the course of the semester.

In one lab session, students were assigned a multi-part problem that required the use of the Business Service library. To personalize their experience with the library, the internal data cache was filled with data from the Blacksburg area. The description of the lab began with an overview of the problems and advantages inherent in using real-time data, and then described how the Business Service library could be used. In the first task, students used the library to create a list of highly rated businesses in the area. In the second, they expanded that list with more detailed information. The primary intent of this lab was to familiarize students with the client library.

Students had a second opportunity to work with the library in the third project of the course, where they built a more complicated "Restaurant Guide" Android application. Students again search for a list of restaurants in a given region. This time, however, they are required to place the results in a custom-created Circular List data structure.

A voluntary survey was administered to the class after students had completed the project, and 17 responses were gathered. Overall, the results, while preliminary, were positive. All but one student indicated that they found the API applicable to day-to-day computing experiences, and all students thought that the API was easy-to-use. Qualitative feedback from the students highlighted clear, easy-to-figure-out methods and that they provided information that was easy to work with. Full numerical results from the survey are reported in Table 1.

6. Future Work

Currently, there are only three services available through our client libraries, each in three languages. Ultimately, we plan to have a diverse gallery that can cater to a wide range of student interests. We are particularly interested in soliciting feedback from the education community on other services that would be of interest and languages that should be supported.

Besides the individual client libraries, there are planned improvements to the prototyping tool. Our intention is that instructors should be able to customize the libraries both for their desired data source and location. Currently, the tool only creates rudimentary implementations based on a given Client Library Specification. More sophisticated code generation for post-processing API results could be added by noting common transformations. For instance, cleaning up whitespace in strings or mapping a function across an array could be supported. Presently there is only one form of API authentication supported and only JSON data can be parsed. Supporting more complicated forms of data parsing, such as HTML, would enable a much broader set of client libraries. Finally, automated support for filling the data cache from the prototyping tool directly would greatly simplify the development process.

7. Conclusion

Learning how to handle real-time web data is an important skill for modern CS students to develop. This learning experience also makes introductory programming projects more relevant to the students day-to-day computing experiences, thereby increasing engagement and motivation. In order to reduce the technical barriers inherent in interacting with distributed systems, we have created publicly available client libraries that make it possible to connect to popular web services in an introductory student friendly manner, enabling a gradual introduction to the complexities of distributed computing. Results from an early pilot indicate that our libraries are easy to use by introductory students. The RealTimeWeb framework provide a promising way to better engage students while preparing them for the realities of the modern IT workplace .

References

[1] L. Carter. Why students with an apparent aptitude for computer science don't choose to major in computer science. In *Proceedings of the 37th SIGCSE technical symposium on Computer science education*, SIGCSE '06, pages 27–31, New York, NY, USA, 2006. ACM. ISBN 1-59593-259-3. doi: 10.1145/1121341.1121352. URL `http://doi.acm.org/10.1145/1121341.1121352`.

[2] D. I. Cordova and M. R. Lepper. Intrinsic motivation and the process of learning: Beneficial effects of contextualization, personalization, and choice. *Journal of educational psychology*, 88:715–730, 1996.

[3] A. B. Downey. *Think Stats*. O'Reilly Media, July 2011. ISBN 1449307116. URL `http://greenteapress.com/thinkstats/thinkstats.pdf`.

[4] S. Edwards. Using test-driven development in the classroom: Providing students with automatic, concrete feedback on performance. In *In Proceedings of the International Conference on Education and Information Systems: Technologies and Applications*, EISTA, 2003.

[5] A. E. Egger. Engaging students in earthquakes via real-time data and decisions. *Science*, 336(6089):1654–1655, 2012. doi: 10.1126/science.1214293. URL `http://www.sciencemag.org/content/336/6089/1654.short`.

[6] A. Fisher, J. Margolis, and F. Miller. Undergraduate women in computer science: experience, motivation and culture. In *Proceedings of the twenty-eighth SIGCSE technical symposium on Computer science education*, SIGCSE '97, pages 106–110, New York, NY, USA, 1997. ACM. ISBN 0-89791-889-4. doi: 10.1145/268084.268127. URL `http://doi.acm.org/10.1145/268084.268127`.

[7] M. Flatt and PLT. Reference: Racket. Technical Report PLT-TR-2013-1, PLT Design Inc., 2013. `http://racket-lang.org/tr1/`.

[8] R. M. Ryan and E. L. Deci. Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. *The American psychologist*, 55(1):68–78, Jan. 2000. ISSN 0003-066X. URL `http://view.ncbi.nlm.nih.gov/pubmed/11392867`.

[9] M. Sahami, S. Roach, E. Cuadros-Vargas, and D. Reed. Computer science curriculum 2013: reviewing the strawman report from the acm/ieee-cs task force. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education*, SIGCSE '12, pages 3–4, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1098-7. doi: 10.1145/2157136.2157140. URL `http://doi.acm.org/10.1145/2157136.2157140`.

[10] L. Torgo. *Data Mining with R, learning with case studies*. Chapman and Hall/CRC, 2010. URL `http://www.liaad.up.pt/~ltorgo/DataMiningWithR`.

[11] M. Waldman. Keeping it real: utilizing nyc open data in an introduction to database systems course. *J. Comput. Sci. Coll.*, 28(6): 156–161, June 2013. ISSN 1937-4771. URL `http://dl.acm.org/citation.cfm?id=2460156.2460186`.