

Web Cartography with Web Standards:  
Teaching, Learning, and Using Open Source Web Mapping Technologies

By

Richard Gardiner Donohue II

A dissertation submitted in partial fulfillment of  
the requirements for the degree of

Doctor of Philosophy

(Geography)

at the

UNIVERSITY OF WISCONSIN-MADISON

2014

Date of final oral examination: 8/05/2014

The dissertation is approved by the following members of the Final Oral Committee:

Robert Emmett Roth, Assistant Professor, Geography

James E. Burt, Professor, Geography

Robert J. Kaiser, Professor, Geography

Ian Muehlenhaus, Assistant Professor, Geography

Molly Steenson, Assistant Professor, Journalism & Mass Communication

# Table of Contents

<b>List of Figures and Tables .....</b>	<b>iii</b>
<b>Acknowledgments .....</b>	<b>v</b>

## Chapter One: Introduction

- 1.1 Project significance
- 1.2 Problem statement
- 1.3 Chapter outline and intent of research studies
  - 1.3.1 Background and literature review (Chapter 2)
  - 1.3.2 Analysis of emerging mapping solutions and assessment of current mapping practices and needs (Chapter 3)
  - 1.3.3 Implementing web standards solutions to meet mapping needs (Chapter 4)
  - 1.3.3 A pattern library for cartographic interface design: toward reusable, modular interface development solutions (Chapter 5)
  - 1.3.4 Conclusion (Chapter 6)

## Chapter Two: Background review

- 2.1 What is the web map of today?
  - 2.1.1 A brief history of the technical enablements of web mapping
  - 2.1.2 Rich Internet Applications: from proprietary plugins to the Open Web Platform
  - 2.1.3 A watershed moment: Web 2.0 and Google Maps.
  - 2.1.4 Moving Beyond Web 2.0 and Google Maps: Toward the Web Map of Tomorrow
- 2.2 How do we teach to the web map of today?
  - 2.2.1 The Body of Knowledge and Beyond
  - 2.2.2 Geography and Information Science & Technology's Engagement with Curriculum and Instruction

## Chapter Three: Competitive Analysis Study and Online Survey

- 3.1 Competitive analysis study
  - 3.1.1 Objectives and methods
  - 3.1.2 Results and discussion
- 3.2 Needs assessment survey
  - 3.2.1 Objectives and methods
  - 3.2.2 Results and discussion
- 3.3 Conclusions: implications for Web Cartography education

## Chapter Four: Diary Study

- 4.1 A mapping scenario to evaluation web mapping technologies and process
  - 4.1.1 Objectives and methods
  - 4.1.2 Results and discussion
- 4.2 Exit Survey
- 4.3 Conclusions: reflections on the research process and the establishment a prototypical web mapping workflow within the full stack to serve Web Cartography education

## Chapter Five: A Web Mapping Pattern Library

- 5.1 Design patterns and pattern libraries: a solution for web mapping education?
- 5.2 Design Patterns and Pattern Libraries Across Computing Fields
  - 5.2.1 Design Patterns in Software Engineering

- 5.2.2 Design Patterns in HCI
- 5.2.3 Design Pattern in web design and development
- 5.2.4 Application of design pattern approaches across disciplines
- 5.3 Approaching a pattern library for Web Cartography education
  - 5.3.1 Establishing a pedagogical model for web mapping instruction
  - 5.3.2 Core heuristics that guide web mapping pattern library development

**Chapter Six: Conclusion**—Executive Summary and Future Research Directions

<b>Appendix A: Protocol for Online Needs Assessment Survey .....</b>	<b>139</b>
<b>Appendix B: Protocol for Diary Study .....</b>	<b>145</b>
<b>Appendix C: Protocol for Exit Survey to the Diary Study .....</b>	<b>153</b>
<b>References .....</b>	<b>163</b>

## List of Figures and Tables

---

### Figures

**Figure 2.1** Synchronous interaction pattern compared with asynchronous pattern.

**Figure 3.1** The importance of different qualities of web mapping technologies when choosing an appropriate technology or set of technologies.

**Figure 4.1** Example solutions for the energy web mapping scenario resulting from the diary study.

**Figure 4.2** Overview of the diary study results.

**Figure 4.3** The participants' emotional experience with each of the four candidate technologies.

**Figure 4.4** Screenshot of DOM hierarchy of a Leaflet map accessed through using the Firebug web development tool in Firefox.

**Figure 4.6** Typical conception of the layers of the 'full stack' development environment.

**Figure 4.7** The prototypical web mapping workflow.

**Figure 5.1** Alexander et al.'s (1979) diagram of the "open alcove".

**Figure 5.2** Tidwell's (1999) pattern 'Navigable Spaces.'

**Figure 5.3** Welie *et al.*'s (2003) diagram indicating the hierarchical structure of related design patterns through different levels.

**Figure 5.4** Welie *et al.*'s (2008) 'Accordion' pattern, organized under a 'Navigating around' category.

**Figure 5.5** Starbucks style guide for the 'Promo Layout A' demonstrating the constituent elements of a single page layout while emphasizing their branding.

**Figure 5.6** Code for America style guide, demonstrating pre-rendered HTML in relation to rendered output of elements.

**Figure 5.7** Github's CSS style guide illustrating their recommended structure for creating a tabbed navigation.

**Figure 5.8** The Zurb Foundation's template options.

**Figure 5.9** The layout of the Yahoo Pattern Library, illustrating common patterns

**Figure 5.10** MailChimp's Pattern Library.

**Figure 5.11** The Barebones Pattern 'Primer.'

**Figure 5.12** The Pears 'Pattern Pairings' interface, rendered through the Wordpress content management system.

**Figure 5.13** Patternry's pattern output, coupled with pre-rendered HTML, CSS, and JavaScript.

**Figure 5.14** Frost's 'atomic design' workflow.

**Figure 5.15** Frost's Pattern Lab interface, in this example showing an example for an 'Accordion' pattern, which comprises list elements. Note that the pre-rendered structural HTML is shown, but the CSS rules and accompanying JavaScript are not.

**Figure 5.16** Current educational challenge of integrating cartographic enablements with technical enablements to transform geographic data into a map.

**Figure 5.17** Conceptual schema of how scope and sequence relate in web mapping education.

**Figure 5.18** Conceptual example of how a first learning module works through a sequence of topics at a shallow depth of scope.

**Figure 5.19** Conceptual example of a successive learning module, reinforcing the HTML and CSS topics at a deeper level of depth of scope while introducing JavaScript.

**Figure 5.20** Conceptual example of a third successive learning module, now integrating HTML, CSS, and JavaScript into learning a web mapping library's API.



**Figure 5.21** Conceptual example of a fourth learning module, departing from further depth of HTML and CSS scope to focus on interaction between JavaScript and the web mapping library's API.

**Figure 5.22** Conceptual walkthrough of a sequence of learning modules approximating a laboratory assignment in an advanced-level web mapping course.

**Figure 5.23** Pre-rendered code within the prototype web mapping pattern library showing the solitary JavaScript code loading in an external data resource with no complimenting CSS or HTML.

**Figure 5.24** The navigation menu of the prototype web mapping pattern library, organizing available design patterns according to 'getting started,' 'data,' 'representation,' 'interaction,' and 'map elements.'

**Figure 5.24** The navigation menu of the prototype web mapping pattern library, organizing available design patterns according to 'getting started,' 'data,' 'representation,' 'interaction,' and 'map elements.'

**Figure 5.26** Screenshot of the architectural structure of the prototype web mapping pattern library. Individual patterns are stored as independent HTML files and loaded into the library.

**Figure 5.27** Conceptual model of a web mapping workflow to create a web map with a tiled 'slippy' basemap with proportional symbols drawn atop that support a dynamic label and filter functionality.

**Figure 5.28** Conceptual walkthrough of a sequence of learning modules approximating a laboratory assignment in an advanced-level web mapping course.

## Tables

**Table 3.1** The representation and interaction codes used to compare the collected suite of open web mapping technologies.

**Table 3.2** Results of the competitive analysis study.

**Table 3.3.** The frequency with which participants in the online needs assessment survey use geographic information, make print maps, and develop web maps as part of their daily work.

**Table 3.4** The level of engagement with the set of web mapping technologies gathered through the competitive analysis study.

**Table 4.1** The representation and interaction requirements of the diary study.

**Table 4.2** The participants' overall emotional experiences during the diary study.

**Table 4.2** The participants' overall emotional experiences during the diary study.

**Table 4.5** The participants' rating of the quality of learning materials available by technology.

**Table 5.1** Structural components with which Alexander *et al.* (1979) presented their design patterns.

**Table 5.2** Criteria for evaluating makeup of various approaches to pattern libraries.

**Table 5.3** Comparison of SE approaches to design patterns and pattern libraries.

**Table 5.4** Comparison between guidelines and design patterns, in favor of design patterns, by different HCI authors.

**Table 5.5** Comparison of HCI approaches to design patterns and pattern libraries.

**Table 5.6** Comparison of web design and development solutions approximating design patterns and pattern libraries.

**Table 5.7** Summary comparison of web design and development solutions approximating design patterns and pattern libraries.

**Table 5.8** Synoptic comparison of design patterns and pattern libraries across the four application domains: (1) Software Engineering, (2) Human-Computer Interaction, (3) web design and development, (4) web mapping

**Table 5.9** Comparison of web design and development pattern solutions approximating educational concepts applied to web mapping.

## Acknowledgements

I owe it all to Robert Roth, who always believed in me and never let me give up. And to Linnzi Hodel, who carried the weight with me until the end. Special thanks to my dissertation writing group composed of Adam Mandelman, Jake Fleming, Chris Limburg, Mark Cooper, and Anna Zeida, from whom I learned how to be academic.

And I owe particular gratitude toward Carl Sack, John Czaplewski, Sam Matthews, Issac Dorsch, Tim Wallace, and Tanya Buckingham.

## Chapter One: Introduction

### Web Mapping on the Open Web Platform

---

#### Overview

Designing maps that deliver a positive user experience is among the highest goals for both educators and practitioners of Cartography. Web cartographers successfully have paired the traditional focus within Cartography on representation design with tenets of interaction design drawn from the fields of Human-Computer Interaction and Usability Engineering. Yet, good web map design requires an understanding of the available technologies suited for the job, as well as the technical skills needed to implement these technologies. In this sense, development must serve design. Unfortunately, the technical demands of development can impede good web map design, particularly for students at a novice level or those lacking a strong programming background. This dissertation addressed this problem by researching how we learn, practice, and teach web mapping within the current web development environment.

#### 1.1 Project significance

For over a decade, web designers—web cartographers included—successfully employed standalone, proprietary technologies requiring specific browser plug-ins to publish high-quality, interactive graphics on the web (Hu 2008). The Adobe Flash product, a multimedia software platform originally designed for creating web-based animated vector graphics, was especially well-suited for web mapping (Lienert *et al.* 2012; Muehlenhaus 2013). However, the broader web development community shifted away from such solutions in the late 2000s and toward a set of JavaScript-driven open source technologies natively supported within modern web browsers (Jobs 2010). These technological changes fundamentally affected how web maps are produced, disseminated, and consumed (Woodruff 2011). Educators too must reexamine how to effectively teach students the skills, concepts, and background knowledge needed to build effective cartographic interfaces within the new medium. Our research investigated this transition

and embraced the opportunity to employ modern web design standards—collectively known as the *Open Web Platform*—to expand the ways in which maps are designed and used online.

Given the continuous technological innovation in computing and the web, it is surprising that we are able to identify specific watershed or transitional moments. Yet, in the years from 2010-2012, it became clear that a cartography curriculum reliant upon a single, proprietary technology was neither tenable nor sustainable. We needed a better solution. Fortunately, this recognition coincided with the release of an abundant number of alternative open source technological solutions built using open web standards, most of which were freely available for download and use on the web. While these options yielded an embarrassment of riches for web cartographers, few resources to date effectively collected, organized, and summarized the technologies; fewer still provided insight into how best to use the technologies individually or in combination across web mapping contexts. Specifically, it was unclear which technologies were suitable for use within university-level cartography courses. Beyond this, if we were able to identify viable candidates, how then were we to implement them given that their technical architecture differed so dramatically from the integrated development environment that Adobe Flash provided? A decade of institutional knowledge and practice in web mapping was suddenly in peril, and it was unclear if the new approach could support higher-order cartographic objectives.

This project aimed to delineate a new technical landscape for Web Cartography while establishing a *process* to integrate ongoing technological change within teaching and learning environments. The immediate pedagogical and curricular needs of the University of Wisconsin–Madison (UW) Cartography program constituted the practical goals of the research. The modern web development workflow emerged as an appropriate model for reevaluating web mapping practice and education today, while prompting questions of where and how Web Cartography diverges from this workflow. The resulting knowledge was used to establish a generalizable web mapping workflow informing laboratory instruction, exercises, and assignments. Finally, drawing from various computing fields, we propose a novel pedagogical approach

called a web mapping design pattern library to promote good design and development solutions in the instruction of web mapping.

## **1.2 Problem statement**

The move to the Open Web Platform challenges practitioners and educators to bring conventional cartographic practice to a shifting technical landscape, as well as how to best harness novel modes of representation and interaction. Our research was designed to address these problems and approached four research questions, which range from practical concerns of immediate program needs to longer-term conceptual questions working toward a deeper understanding of Web Cartography and its relationship with the wider web community:

- (1) What technologies currently are available for web mapping and how do they vary?
- (2) What are the important characteristics of web maps that should inform the selection of web mapping technologies?
- (3) How should web mapping be taught in higher education?
- (4) How can we better cope with continued evolution in web mapping technologies?

To this end, the research reported within this dissertation connected cartographic representation and interaction design with specific instructional techniques for web development using the Open Web Platform. The result was a better understanding of the current state of web mapping and the ways in which teachers and students can best negotiate this conceptually and technically demanding terrain. Through this work, we generated insight into how the workflow of web mapping relates to that of web design and development in general, as well as how our instruction of Web Cartography benefits from articulating the critical points of intersection and divergence between the two.

### 1.3 Chapter outline and intent of research studies

This dissertation employed four empirical studies to address the research questions posed above. Design of the process followed the *convergent methods paradigm*, which prescribes administration of multiple, often qualitative social science methods (Buttenfield 1999). Each study then is conducted in a discount manner (e.g., leveraging secondary sources, recruiting only a small number of participants) to ascertain input and feedback quickly (Nielson 1994). Reliability of the project as a whole is maintained through triangulation of insights generated across the studies. Such a multi-stage process has been applied with success to the design of custom map symbol libraries (Robinson et al., 2012, Robinson et al., 2013) as well as the usability evaluation of interactive maps (Slocum et al., 2003, Robinson et al., 2005). We triangulated insights across four studies in total: (1) a competitive analysis of existing web mapping technologies, (2) a needs assessment survey with web map designers and developers, (3) a diary study tracking the implementation of the same web map using a candidate subset of technologies identified from the first two studies, and (4) an exit survey collecting opinions from the diary participants. Altogether, these studies represent a repeatable process that can recur in the future, and thereby be used to update educational strategies and analyze the evolving state of Web Cartography. The chapters within this dissertation subsections detail the method design of each of the four studies included in the process and the findings. Finally, we proposed a set of heuristics to inform a collection of design patterns (i.e., a ‘pattern library’) to bridge expert and novice experience in Web Cartography, as well as higher-order cartographic design principles with technical development solutions. Altogether, the dissertation offers a substantial contribution to current cartographic education needs and establishes a research agenda for further work.

#### 1.3.1 Background review (Chapter 2)

Chapter 2 provides an overview of modern web mapping and situates educational challenges and strategies within it. First, we briefly characterize the current state of the art by asking, *What is the web*

*map of today?* Changes in web technologies within the past decade have created a stunning array of mapping solutions, thereby increasing the flexibility afforded to cartographers in both applying conventional cartographic principles and implementing novel functionality within web maps. The rise of mobile mapping and intensified map integration with diverse web services further complicate the ontological status of today's web map, and therein how to best teach web mapping. Cartographers today are presented with an undetermined set of practices and tools ranging from widely accessible but limited "point-and-click" options to code libraries allowing for fully customized cartographic interfaces. These options need to be considered in terms of the educational objectives of GIScience and Cartography.

After establishing the web map of today, we follow with a second question, *How do we teach to this web map?* We acknowledge the seminal effort to formalize Geographic Information Science and Technology (GIS&T) education within the UCGIS *Geographic Information Science and Technology: Body of Knowledge (BoK)* (UCGIS 2006). The *BoK* is a catalogue of the depth and breadth of GIScience and provides support for curricular development. However, it offers little by way of prescribing learning processes that facilitate understanding and has yet to be updated for modern web mapping. A review of educational literature grounded in Geography and Cartography introduces four concepts that potentially redress this gap between practice and education. **Scope** and **sequence** are useful for determining the depth of and order in which new skills and concepts should be introduced. Web Cartography education also may prioritize identifying students' ***misconceptions*** that hinder learning, as well as the ***threshold concepts*** or 'conceptual gateways' that lead to new ways of understanding. Further research is needed to integrate the modern web map with cartographic education.

### **1.3.2 Analysis of emerging mapping solutions and assessment of current mapping needs and practices (Chapter 3)**

Chapter 3 reports on two studies designed to assess currently available web mapping options and how they meet the educational needs at the undergraduate and graduate levels. The first study used the

*competitive analysis* method, or a systematic comparison of a suite of related tools or technologies based on their relative merits (Nielsen 1994). The competitive analysis was conducted to identify, evaluate, and compare a wide array of open source web mapping technologies. This study established the range of available web mapping options, as well as their relative strengths and weaknesses in supporting the functionality desired for cartographic representation and interaction. In total, thirty-five (n=35) web mapping technologies were surveyed and compared by two independent coders according to supported representation and interaction design techniques.

Results from the competitive analysis were strengthened by an online survey study aiming to clarify the tools that currently are used by web mapping practitioners within the UW System and the general level of awareness of alternative options, as well as to generate insights into the training and education processes operating within their respective workflows. The online survey acted as a *needs assessment*, as the purpose of the survey was to elicit past experiences with the collected technologies as well as to identify future or currently unmet web mapping needs (Wiggins and French 1991). In total, twenty-one (n=21) UW System faculty, staff, and students participated in the online survey. Together, the two studies described in Chapter 3 helped to answer the first two questions posed within the project statement above, with the insights used to identify the subset of web mapping technologies that held potential for subsequent evaluation and adoption within the UW curriculum.

### **1.3.3 Implementing web standards solutions to meet mapping needs (Chapter 4)**

Building upon the results of Chapter 3, Chapter 4 reports on two additional studies designed to evaluate a subset of candidate web mapping technologies against a typical set of cartographic requirements. The first study used the *diary* technique, a variation of participant observation that requires participants to self-observe as they complete an activity (Marsh & Haklay 2010). Four student participants representative of the targeted user group developed the same web map using a different candidate technology (four in total) and recorded their progress in a diary every hour for a total of forty hours. The requirements of the web



map itself were based on the cartographic representation and interaction techniques leveraged for the competitive analysis reported in Chapter 3. To improve reliability in the diary entries, while remaining cognizant of participant fatigue, I completed the same web mapping scenario with all four technologies. As a result, there were eight ( $n=8$ ) diaries total, two for each of the four candidate technologies. An independent coder interpreted the diaries according to the implemented representation and interaction requirements.

Results from the diary study were supplemented by a final exit survey with the four student participants. The exit survey was designed to elicit additional feedback about their assigned technology and the overall learning and mapping processes. This pair of studies helped to identify two web mapping technologies that currently support the UW Cartography curricular needs; the resulting laboratory exercises are included as appendices to the dissertation. Importantly, this pair of studies also generated new insights into the process of making maps using the Open Web Platform, as well as the requisite skills and knowledge to do so. Together, the two studies described in Chapter 4 helped to answer the third research question posed above, leading to a greater understanding of a workflow for development using the Open Web Platform.

### **1.3.4 A pattern library for cartographic interface design: toward reusable, modular interface development solutions (Chapter 5)**

Chapter 5 introduces the concept and practice of design patterns to Web Cartography education and practice. Software Engineering (SE) and Human-Computer Interaction (HCI) both have leveraged design patterns and pattern libraries for software development and interface design. Professional web designers and developers also have employed design patterns to aid novice designers, as well to bridge design and development. While Web Cartography successfully has applied principles and practices from these fields for web mapping, design patterns remain an unexplored solution to many of the challenges in Web Cartography education identified in Chapters 3 and 4. This chapter reviews and analyzes design patterns

and offers recommendations for creating and maintaining design patterns within a pattern library to serve web mapping education and practice.

## 2. Background review

---

### Introduction

Neumann (2012) provides a useful terminological distinction between Web Cartography and web mapping. *Web Cartography* is a research thrust that addresses the conceptual and theoretical aspects of cartographic design, including the application of usability and cognitive psychology to mapmaking in a digital, distributed environment. *Web mapping* is encompassed within Web Cartography and constitutes the more technical aspects of designing, deploying, and using maps through the web. To date, research on Web Cartography has extensively dealt with questions of *cartographic representation*, or how maps are seen, understood, and imbued with meaning within a distributed digital context (Harrower 2004; Jenny *et al.* 2008). To complement research into cartographic representation, Roth (2013, 64) offers a substantial contribution to the science of *cartographic interaction*, defined as a “dialogue between human and a map mediated through a computing device”. Research into map manipulation by the user can be further distinguished from that of an individual *cartographic interface*, or the “set of digital tools through which the cartographic interaction occurs” (Roth 2013, 66). This literature review approaches a gulf between web mapping (in Neumann’s sense) and the educational approaches for building cartographic interface solutions (in Roth’s sense).

### 2.1. What is the web map of today?

It is easy, accurate, and obvious to say that Digital Cartography has changed greatly over the past twenty to thirty years due to rapid advancements in computing technologies, the Internet, and the web (Monmonier 1985; Kraak and Brown 2003; Harrower 2004).<sup>1</sup> There is little reason to suspect this will

---

<sup>1</sup> This manuscript maintains a distinction between the *Internet*, defined as the global computer network born from the ARPAnet and composed of interconnected networks using standardized communication protocols to distribute information to users (*Concise Oxford English Dictionary* 2008) and the *web* which, while connected to and facilitated through the Internet, also includes the broader networked applications that help constitute online maps (Tsou 2011). Skarlatidou (2010) also notes that the web is technically distinct from the Internet in that the TCP/IP protocol that supports HTTP runs on top of the Internet (246).

not continue, for change is, as Michael Peterson asserts, “inevitable when it comes to maps and the Internet” (Peterson 2008, 7). The unstable nature of this emerging discipline is in part evidenced by the wide variability of language used to name and describe the phenomena itself. Terminology for the current state of web mapping within the literature includes the *geospatial web* (Scharl and Tochtermann 2009; Elwood 2010), the *geoweb* (Haklay *et al.* 2008; Elwood 2011), *online mapping* (Crampton 1999), *Internet Mapping* (Peng and Tsou 2003), *web mapping* (Haklay *et al.* 2008; Skarlatidou 2010), *Web Cartography* (Kraak and Brown 2003; Tsou 2011; Neumann 2012), *cybercartography* (Taylor 2005), *web-based multimedia GIS* (Hu 2003), *maps 2.0* (Crampton 2009), *GIS/2* (Miller 2006), *neocartography* (Jobst and Dollner 2008; Liu and Palen 2010; Kraak and Ormeling 2011; Cartwright 2012), *neogeography* (Turner 2006; Hudson-Smith *et al.* 2009), *locative media* (Rheingold 2002), *digiplace* (Zook and Graham 2007), *spatial crowdsourcing* or *geocollaboration* (Hopfer and MacEachren 2007), *map hacking* (Gibson and Erle 2006), and *countermapping* (Harris and Hazen 2006). These terms point toward a range of research questions into novel forms of geographic information visualization, the role of user interactivity, the integration of disparate digital tools and information sources, the growing role of mobile devices and location-based services, and critical questions of power, place, and identity.

### 2.1.1 A brief history of the technical enablements of web mapping

Web mapping began with the invention of the World Wide Web itself by Tim Berners-Lee in 1991, which immediately made the distribution of maps to users easier and changed the nature of cartographic practice and research (Crampton 1999; Cartwright 2003; Peterson 2003; Haklay 2010; Peterson 2012). The “hypertext project” that has come to be known simply as the *web* offered a networked system of hyperlinked documents written in a plain text ASCII protocol scripting language, *HyperText Markup Language (HTML)* (Harrower 2004; Hu 2008).<sup>2</sup> HTML documents are shared across the web via a *HyperText Transfer Protocol (HTTP)*, the primary means for the transfer and exchange of hypertext across the Internet (Skarlatidou 2010). The adoption of the HTML and HTTP protocols by the *World*

---

<sup>2</sup> See <http://www.w3.org/Proposal.html> for the original proposal for the specification.

*Wide Web Consortium (W3C)* was followed by the release of the first *web browser* supporting a graphical user interface (GUI), the Mosaic browser in 1993.<sup>3</sup> Several browsers have vied for the market share of users since, and achieving *consistency* in and *accessibility* to the display and interaction functionality across these browsers continues to be a general problem for web authoring.<sup>4</sup> Despite such challenges, the web provided a new means for distributing maps to the public, “stimulated the public’s demand for maps,” and invited cartographers to create new ways to share geospatial information (Harrower 2004, 39).

Geographic information is encoded in one of two data models: either as gridded units composed of rows and columns (i.e., *raster*) or as points, lines, polylines, and polygons (i.e., *vector*). Both raster and vector formats remain important to Cartography and GIS today, as they serve different purposes for geographic representation (Slocum *et al.* 2009). While desktop GIS systems and software accounted for both of these models before the invention of the web, cartographers have grappled with ways to efficiently transfer and render information encoded within these two formats on the web. The web browser—a *client-side* technology—was initially able to interpret basic formatting codes in HTML to display images in addition to text. The distributed network of computers designed to support network services—known as servers, or *server-side* technology—made the dissemination of maps across the web possible from the outset, albeit limited to digitally-scanned, analog maps (Harrower 1997; Cartwright 2008; Hu 2008). These raster image formats tend to be larger files, thus constituting a greater challenge in terms of bandwidth and transfer than features encoded within a vector format. However, support for raster images within initial web specifications made them an obvious mapping solution early on. Vector data files, while potentially smaller in file size, require more complex computation, are slower to render within a browser, and have lacked standardized support in web browsers.

---

<sup>3</sup> The W3C is the primary international standards organization for the World Wide Web led by the creator of the web itself and promotes the development of web standards. See <http://www.w3.org/Consortium/>.

<sup>4</sup> Most differences in the appearance, features, and functionalities between various web browsers is the result of different designs as software applications. The important differences for performance within the GUI itself reside in the underlying layout engines. Additionally, variability exists across different operating systems (OS).

While HTML alone supported sharing of the traditional analog map between computers in the form of an image file, an original specification of the World Wide Web known as the *Common Gateway Interface (CGI)* further enabled client-server interaction (Plewe 2007; Tsou 2011).<sup>5</sup> This provided the technical means to serve maps based on the user's request, a process that would come to be known as *on-demand mapping*, that went beyond merely distributing a pre-existing map by allowing users to request maps 'on the fly' (Harrower 2004; Cartwright 2008). GeoSystem's MapQuest was the noteworthy pioneer of this approach and helped establish these client-to-server interactions as a "basis of modern cartography" with the launch of the first consumer-focused interactive mapping site on the Web in 1996 (Peterson 2008, 4).<sup>6</sup> Opportunities afforded through this client-server architecture encouraged cartographers to explore possibilities beyond the transfer of simple static maps and foreshadowed the robust interaction capabilities to come.

### 2.1.2 Rich Internet Applications: from proprietary plug-ins to the Open Web Platform

Additional technologies expanded the limited capabilities of HTML to deliver *Rich Internet Applications (RIAs)*, which provided web users with graphic capabilities similar to those supported by desktop software applications. Such high performing web graphics most commonly were achieved through the use of a *plug-in*, an independent software component facilitating content distribution and consistent graphic representation and user interaction across a variety of platforms and browsers. While plug-ins have existed since the mid-1970s, they became more prevalent within web browsers in the mid-1990s and expanded not only the functionality of the browser, but also supported a growing number of non-

---

<sup>5</sup> The CGI specification (<http://tools.ietf.org/html/rfc3875>) in his most basic form initially allowed for command line execution of processes on the server from the client.

<sup>6</sup> This concept of dynamic interaction between client and server also encouraged efforts toward a distributed GIS or Web-based GIS to perform GIS-related analysis tasks (Andrienko *et al.* 2003; Hu 2008). Map applications have tended toward more of a general consumer end rather than such commercial/reference applications and GIS software has been slow to provide online mapping solutions (Peterson 2008). Notable exceptions to this include current products such as "geotools" and webweb process services (WPS). See <http://docs.geotools.org/latest/userguide/unsupported/wps.html>

standardized file formats and types (Plewe 2007; Tsou 2011).<sup>7</sup> Prominent examples include the QuickTime Player, which handles video, audio, and image formats, and the Java plug-in, which activates a Java *applet* (an abbreviated term for *application*). These ‘hypermedia’ technologies allowed cartographers to explore new possibilities such as three-dimensional mapping, animation, and user interaction while freeing the cartographer from “conventional design constraints” (Harrower 2004, 35).

In 1997, Macromedia introduced the web market to Macromedia Flash (subsequently released as Adobe Flash), which provided a desktop application known as an *integrated development environment (IDE)* for producing interactive and animated vector graphics encoded within a binary *shockwave movie format (swf)* file rendered in the client browser through the Flash plug-in.<sup>8</sup> Broadly accepted within the graphics industry as a standard for excellence, the authoring functionality of Flash-built applications was extended through writing code in the ActionScript programming language, though the Adobe Flash Professional IDE itself provided novice users with the means to produce RIAs without sophisticated programming skills.<sup>9</sup> Flash greatly increased the potential for high-quality cartographic interaction and representation on the web (Hu 2008; Leinert *et al.* 2012) and at the time was “heralded as the future of online interactivity and mapping” (Muelenhaus 2014, 197). Flash boasted high market penetration, and its compiled binary format made it small and efficient for transfer to the client compared to its text-based alternative, the *Scalable Vector Graphic (SVG)* (Peterson 2008; Lienert *et. al* 2012). Microsoft Corporation responded in kind with their equivalent of Flash, a cross-platform, cross-browser framework called Silverlight. While both Flash and Silverlight offered numerous advantages for web cartographers,

---

<sup>7</sup> The EDT text editor is a pre-Internet example of a plug-in that extended the capabilities of a mainframe operating system to support basic text editing. *EDT Text Editor Reference Manual*, Cinnaminson, New Jersey: Unisys Corporation, 1975.

<sup>8</sup> Compiled swf files were relatively small, thereby providing for vector-based mapping solutions accessible across platforms provided the Flash plug-in was installed.

<sup>9</sup> ActionScript is an object-oriented programming language derived from ECMAScript and shares similar syntax and semantics with the now more popular JavaScript programming language.

both were proprietary software applications dependent upon a single vendor for the preservation of the codebase (Lienert *et. al* 2012).<sup>10</sup>

In contrast with this plug-in approach is the use of non-proprietary ‘open’ technology specifications written by the W3C, supported natively within web browsers and known collectively as the *Open Web Platform*.<sup>11</sup> In recent years, the majority of web browsers increasingly have adopted these standards to provide consistent and accessible authoring and deployment tools that rival the sophistication only previously achieved through the use of a plug-in (Pulsifer *et al.* 2008, 166). Standards supported by these “modern” web browsers (most recently subsumed under the term *HTML5*) include:

- **HTML** (Hypertext Markup Language, the markup language that structures a web document)
- **CSS** (Cascading Style Sheets, the style rules that govern the layout and aesthetic appearance of HTML elements within a web page)
- **SVG** (Scalable Vector Graphics, a text-based web standard vector graphic derived from XML that describes vector shapes, as well as text and embedded raster graphics)
- **JavaScript** (a prototype-based server-side scripting language based on ECMAScript), and
- **Web Application Programming Interfaces (APIs)** including the **Document Object Model (DOM)**, which describes the relations between all entities within a rendered web page, as well as the **Canvas** element, which supports drawing of bitmap raster images.

The web map of today is therefore increasingly authored and consumed within the modern web browser, composed of these web standards and enabled under the scope of the open source movement (Pulsifer *et al.* 2008).<sup>12</sup>

The web map of today is rarely written using pure, ‘vanilla’ JavaScript. Rather, cartographers often make use of collectively authored, royalty-free JavaScript code *libraries* shared openly on the web that offer foundational support for such common tasks of data deserialization, display, and manipulation (Bostock and Davies 2013). These libraries are implemented and extended to build custom interfaces by more technically savvy authors. Other elements of the Open Web Platform are used in conjunction to tailor

---

<sup>10</sup> The decline of Adobe Flash across the graphic industry is commonly attributed to its lack of support on the iOS operating system of Apple Corporation’s iPhone and iPad products, stemming from an open letter written by Steve Jobs in 2010. See <http://www.apple.com/hotnews/thoughts-on-flash/>.

<sup>11</sup> See [http://www.w3.org/wiki/Open\\_Web\\_Platform](http://www.w3.org/wiki/Open_Web_Platform).

<sup>12</sup> The use of the term *modern* is drawn colloquially from the web community and means state-of-the-art or current. A *modern* web browser has come to mean one that supports the W3C specified web standards.



design aspects such as layout, visual hierarchy, colors, textures, typography, dynamic change, and user interaction. Recent advancements in web standards support among modern web browsers have helped promote a wave of these code libraries capable of meeting the representation, interaction, and user experience (UX) needs of web mapping, and offer the potential to retain traditional cartographic conventions. They support a variety of data types and formats, features, and functionality to meet a wide-ranging set of user-anticipated needs and presupposed prescriptions of what a modern web map should do.

### 2.1.3 A watershed moment: Web 2.0 and Google Maps.

Around the turn of the century, a new set of technical enablements fundamentally changed the relationship between web content authoring, dissemination, and consumption. The highly popularized though somewhat indefinite term *web 2.0* originated at a conference hosted by the prominent web-related publisher O'Reilly Media. The conference sought to characterize shifts in design patterns, business models, and web tools “for the next generation of software” (O'Reilly 2005). The term thereby refers to a range of technologies and practices made available by greater access to high-speed and inexpensive bandwidth, wireless and cellular Internet connections, an increasing use of the web for personal publishing, the advent of ‘social media’, and a growing commercialization of web services (Hu 2008).<sup>13</sup> Pillars of web 2.0 saw the web as a *platform* built from flexible and lightweight software designed for ‘hackability’ and ‘remixability’ upon which user experience occurs, rather than merely a mechanism for the consumption of information. At this time *user-generated content*—a precursor to the interests in ‘big data’ that would come to challenge the web geospatial infrastructure in the late 2000s—gained importance as technologists herald the potential to “harness collective intelligence” and the “wisdom of crowds” (O'Reilly 2005, online). The implications of web 2.0 for cartography accentuated an already shifting ontology of the prototypical map from primarily analog forms to various expressions within

---

<sup>13</sup> Web services provide a “loosely coupled,” platform-independent model for encoding and exchanging data over the web or Internet (Kralidis 2007, 224) and help constitute what comes to be referred to as the “cloud.”

digital technologies, including the web as the primary dissemination mechanism for maps and high levels of user interaction (Roth 2013).

The launch of Google Maps and the employment of the tile-based ‘slippy’ maps in 2005 is broadly understood as a watershed moment in Web Cartography, ushering in a set of standards and assumptions we now take for granted in contemporary web mapping (Crampton 2008). Three prominent aspects of this technology include *AJAX*-requested tiles, an *Application Programming Interface (API)*, and (as a direct implication of the API) the map *mashup*. Together these technologies supported more intuitive interaction and a superior user experience than the previous generation of MapQuest-like forms of cartographic interfaces. Google Maps were quickly complemented by such immersive mapping environments as NASA Whirlwind, which became Google Earth (Plewe 2007; Tsou 2011), and effectively “transformed the online mapping landscape” (Peterson 2008, 6). How did the technological breakthroughs demonstrated within these applications contribute to the web map of today?

While applets and similar component-oriented tools expanded the solutions for building web interfaces, web cartographers are now able to achieve a more instantaneous user experience through the employment of a technology known as *Asynchronous JavaScript and XML (AJAX)*.<sup>14</sup> This technology improves the client-server communication beyond the earlier CGI scripting from which it emerged. Rather than requiring users to wait for server requests to load additional data or code—a process necessitating a significant time delay and re-loading the entire web page—the AJAX web application model introduced an intermediary *AJAX engine* between the server and client (see Figure 1). This engine handles server requests *asynchronously* with user interaction, decreasing response time and providing uninterrupted control of the user interface. Google Maps utilizes AJAX to load sets of tiled basemap images into the browser, providing seamless panning and zooming of multiscale basemap graphics (Tsou 2011; Peterson 2012). Because download speeds still varied considerably across users’ Internet connections, large raster

---

<sup>14</sup> *Extensible Markup Language (XML)* is another W3C specification for encoding data in a structured, standard format. Although XML may be used in AJAX techniques it is not required. Increasingly the *JavaScript Object Notation (JSON)*, an open specification interchange format, is used instead.

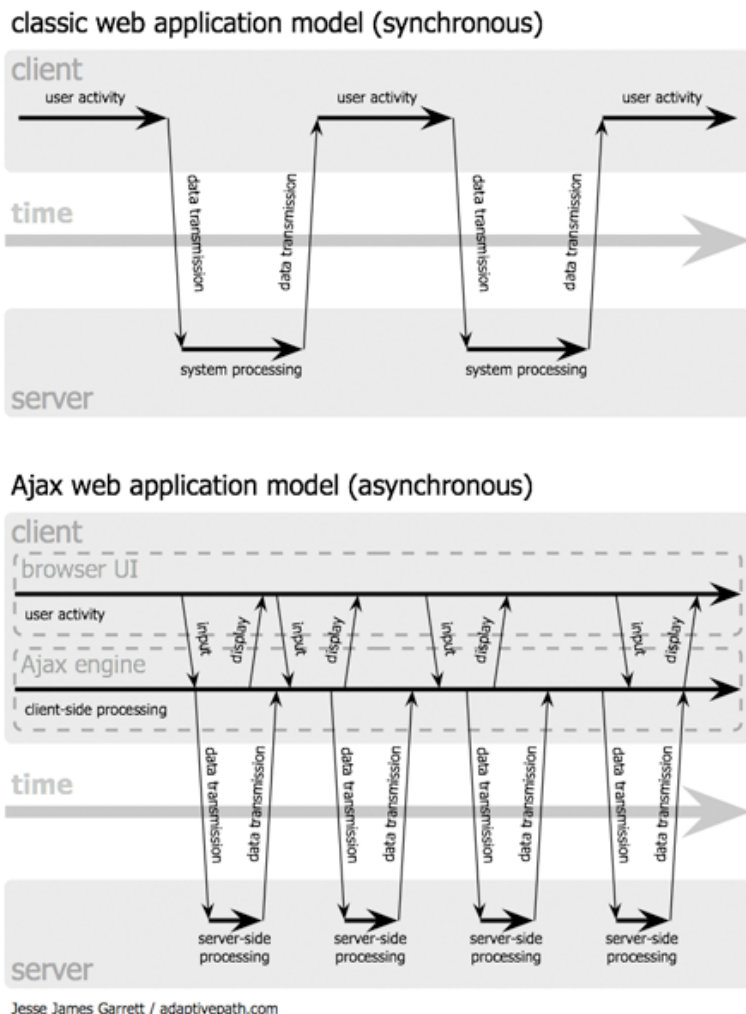
maps were broken into smaller sets of tiled images at various zoom levels, with a small subsets of these tiles loaded into the browser when necessary using an AJAX request.

Google Maps introduced the public and aspiring web cartographers to another fundamental component of contemporary web mapping: the *Application Programming Interface (API)*. The API allows various software and hardware components to interact or ‘talk’ with one another, letting users access and control more complicated computational routines through a higher-level, more simplified scripting or programming language (Peterson 2008; Skarlatidou 2010; Peterson 2012).<sup>15</sup> In part a branding mechanism for the company (Gale 2013), the Google API offered limited free access for rendering features of the map tiles on their servers and for loading, rendering, and interacting with user-generated maps in the client. The API became a “hook” for further manipulation of the map (Crampton 2009, 94) and increasingly came to define online mapping, a transformation of cartography characterized by a shift from a “passive to active enterprise” (Peterson 2012, 4). The API facilitated building user-defined maps and allowed for the overlay of additional information (Crampton 2009; Peterson 2012).<sup>16</sup> Google’s decision to ‘open up’ part of the API to the public was followed in suit by Yahoo!, Microsoft Live Local, and Virtual Earth and would become a defining feature of web mapping solutions built on the Open Web Platform.

---

<sup>15</sup> Most APIs are not transferable across different code libraries. The Mapstraction API is one exception allowing for switching between APIs, but only for basic functionality (Peterson 2012).

<sup>16</sup> This increase in the power of map-making to the public and amateurs also constitutes a new avenue of academic research, often captured under notions of “do-it-yourself” cartography or the “democratization of cartography” (Tsou 2011).



**Figure 2.1** Synchronous interaction pattern compared with asynchronous pattern, from Garrett (2005), <http://adaptivepath.com>

Google’s API-enabled maps paved the way for a new form of map—the *mashup*—a hybrid web of various web applications and services that allow users to integrate content from multiple, disparate sources and display that information on a set of (often proprietary) tiled basemap images (Pulsifer *et al.* 2008; Peterson 2012). Mashups helped popularize the integration of user-generated content with online maps and demonstrated that, “given access to the tools, users from a wide range of backgrounds will create Web mapping applications that link location to a variety of data sets” (Rouse *et. al* 2007, 156). Initially these ‘do-it-yourself’ or *DIY* mashups were frequently composed of point locations symbolized by the iconic ‘push-in’ symbol, which at least for a while during the late 2000s became ubiquitous within popular web map development (Wallace, forthcoming). More importantly than the limitations of thematic

representation, however, mashups helped mobilize amateurs, volunteers, and hobbyists to generate various content made accessible by the web 2.0 infrastructure (Tsou 2011).

#### 2.1.4 Moving Beyond Web 2.0 and Google Maps: Toward the Web Map of Tomorrow

Turner (2006) dubs current web mapping practices *neogeography* to underscore how markedly different this new set of techniques and tools is from traditional Geographic Information Systems (GIS). Examples such as interactive displays supporting queries of geolocated data features, the integration of spatial and temporal data from multiple sources, and the increased production of new data from the public are no longer impressive, but assumed (Crampton 2009). A more integrated relationship between web mapping and the wider technical field of the web itself shifted web-based GIS to the “geospatial web” or the “Geoweb” (Haklay 2008). The map mashup spawned various forms of *crowdsourcing*, a “form of emergent collaboration in which multiple people work together on a common project” enabled by increasingly low access barriers to the web itself (Crampton 2009, 95). Sometimes referred to as *volunteered geographic information (VGI)*, crowdsourced data fostered the enrollment of the public in networked, collaborative mapping practices and raised deeper questions as to how geospatial web engagement can break barriers between expert and lay, redefining the role of democratic citizenship (Goodchild and Li 2010, 13).<sup>17</sup>

Like the advent of the web itself, the rise of *mobile mapping* devices in recent years introduces both new opportunities and challenges for Digital Cartography and, curiously, re-introduces some older obstacles for distributed mapping.<sup>18</sup> Mobile mapping on phones and devices requires consideration of new ways of interacting with the map (as through touch interfaces), new functions and capabilities (such as location-aware mapping applications), and new content (Tsou 2011). Immersive, user-centered mobile maps

---

<sup>17</sup> One critical issue stemming from this is that information is not always “volunteered” willingly or knowingly, as is the case with data provided from such commercial products as Facebook or Twitter.

<sup>18</sup> This ambiguous role of mobile in web mapping is a great illustration of a flawed approach to understanding the technological purely in terms of development, which often implies continual progress and improvement. See Mosco’s (2004) *The Digital Sublime* for an account of the myths of technological breakthrough and promise.

(Meng and Reichenbacher 2005), context-aware wayfinding services (Huang and Gartner 2008, Spek 2009), geo-identification (Delikostidis and van Elzakker 2008), and *location-based services (LBS)* for enhancing landscape experience (Ligtenberg *et al.* 2009) signal only the tip of the proverbial iceberg of new mapping use and research.

Increased interoperability and greater data sharing standards now allow users to access applications, software, and infrastructure more efficiently through “the cloud” as web services, rather than as conventional downloadable data files or applications (Pulsifer *et al.* 2008). One goal of such environments is to create more seamless computing processes where users can make use of virtual servers and online computing platforms without the expertise or hassle of configuration themselves. (Cartwright 2008; Tsou 2011).<sup>19</sup> *Web mapping services* built by experts offers cleaner, more intuitive interfaces and powerful backend computational processes that can help mitigate the need to write code or use a mapping API. Users could instead, for example, upload raw data in any format and display these data thematically upon a set of publically distributed basemap tiles or vector layers (Gale 2013).<sup>20</sup> Such a goal aims to make web mapping easier for common users at the expense of limiting customized interface solutions and map aesthetics.

Contrasting with such a “point-and-click” approach to web mapping is a growing number of JavaScript options and online cartographic tools “only previously available as part of commercial software packages” (Crampton 2009, 94). The increasing array of options for map authors is characterized by a trend toward a “set of programming methods for producing interactive asynchronous web applications” (Tsou 2011, 252). While the availability of open web mapping tools and technologies has great potential, questions arise as to the capacity of users to leverage this potential. As Gale (2013, 157) cautions, “even if you provide a way to customize something, only a small percentage of people will generally take

---

<sup>19</sup> Tsou (2011) also argues these recent shifts parallel one from geovisualization to user-centered design (i.e., a shift from traditional expert-based GIS to general users of location based services, etc.).

<sup>20</sup> CartoDB (<http://cartodb.com/>), for example, attempts to provide a low barrier to access web service allowing non-technically savvy users such as data journalists a tool from which to quickly upload data into a database, run queries on that data, and construct visually appealing maps to be shared or exported.

advantage of that facility”. Therein lies one of the most urgent challenges for cartographic education and web mapping today.

This section described the contemporary landscape of Web Cartography and web mapping technology. Advancement in web mapping technology continues to offer opportunities to enroll geographic information within a variety of deployment mechanisms and modes for engagement. Furthermore, web mapping technologies today are more integrated with open source technical solutions on a wider scale, and at a level that offers renewed promise for meeting high-end cartographic objectives. However, the challenge to educators within Cartography and GIS programs is no less daunting. The Open Web Platform requires a skill set stretching beyond what the curriculum has to date been accountable for, and it continues to innovate and shift beneath our feet. We are presented with uncertainty as to the best way forward. The good news is that GIScience educators are particularly committed to adapting to this change. What follows is an analysis of the most promising educational strategies for Geography and Information Science & Technology (GIS&T) instruction.

## **2.2. How do we teach to the web map of today?**

A shift in web mapping technologies toward what I’ve described as the Open Web Platform prompts a reevaluation of how we teach Web Cartography. Authoring tools in prior web mapping processes provided an encapsulated environment for map development that—while requiring technical knowledge of the development software and computer programming—helped constrain the range of skills and knowledge required to produce a web map. By contrast, the Open Web Platform of today consists of an assemblage of different technologies described above as *web standards* that require additional expertise and a more complicated workflow for bringing them together to produce the same map. In what ways does this workflow increasingly resemble that of a front-end web developer? The learning curve is steeper and the challenge greater for guiding students to a place where they can use these technologies to implement higher-order design tasks. How can we most effectively reconsider our pedagogical approach

in light of these changes and establish a roadmap for students to harness the practices and tools of modern web development to improve process of learning Web Cartography?

### **2.2.1 The *Body of Knowledge* and Beyond**

Given the above characterization of current web mapping practices, students today must gain competency in a range of technical terms, solutions, and practices. This in itself is nothing new, for as Foote (2011, 82) reminds us, “GIS&T is a demanding area of study involving a number of complex and interlocking concepts, theories and skills”. The application of GIS concepts within a web development environment only increases this range of required proficiency. Of note is that geographers and cartographers have contributed substantially to the pedagogy informing instruction within their courses, increasingly couched under the term *Geography and Information Science & Technology (GIS&T)*.<sup>21</sup> Somewhat unique to their discipline, cartographers and GIScientists have responded to rapid innovation in GIS&T with innovation in education seeking to serve the high demand for qualified graduates driven by GIS&T workforce needs within industry, government, and non-governmental organizations. Whether conceived of as a challenge or an opportunity, educators within Geography grapple with the question of “how best to reorganize and rethink traditional and sometimes hidebound disciplinary curricula and adopt new teaching methods in the context of this rapidly evolving field” (Foote *et al.* 2011, 5). Yet keeping pace with the changing technical landscape and emerging mapping solutions remains daunting.

UCGIS (2006) *Geographic Information Science and Technology: Body of Knowledge (BoK)* is a seminal work toward this goal, cataloging the depth and breadth of GIS&T education. Though not a curriculum itself, it is a means of supporting curricular development (Prager 2011). The *BoK* comprises ten knowledge areas, seventy-three units, 329 topics and over 1,600 formal educational objectives derived from Bloom’s (1956) taxonomy of learning objectives. Yet the *BoK* falls short of offering a comprehensive solution for teaching the web map of today. For one, the knowledge areas and topics have

---

<sup>21</sup> The addition of “& Technology” to the conventional “Geographic Information Science (GIScience)” acknowledges the growth of digital cartography from the mainframe GIS of the 1980s and desktop GIS of the 1990s into other web-based, distributed mapping technologies.



grown to encompass a wide array of web technologies and practices extending beyond what the *BoK* compiled less than a decade ago. This is less a limitation of the *BoK* itself, as periodic update of this material and the associated learning objectives is anticipated. However, there is more to effective education than simply covering content. Behavioral aspects such as *how to learn* (Prager 2012) and the integration of the *BoK*'s topics within a meaningful learning process that facilitates *understanding*—and not merely the acquisition of knowledge per se—are lacking (Foote 2011a). Fortunately, GIS&T educators since have responded to these deficiencies through a number of practical and conceptual contributions, many of which are drawn from the fields of Educational Psychology and Curriculum & Instruction.

### **2.2.2 Geography and Information Science & Technology's engagement with Curriculum and Instruction**

Foote (2011b) argues that articulating scope and sequence are critical for weaving foundational GIS&T topics and learning objectives into a curriculum designed to better promote student mastery. Foote (2011b, 81) defines *scope* as the “depth of knowledge about a given concept or skill” and *sequence* as the “the order in which concepts are introduced”. Both are valuable for improving the design of GIS&T curricula. The order in which new content is introduced is especially important for the instruction of web mapping, as it involves a large number of complex concepts and skills working in concert requiring significant time to understand and master. Logically and meaningfully organizing so much information benefits from careful consideration of “how these are introduced, at what level, and in how much detail” (Foote 2011b, 82). The extension of Digital Cartography into a web development workflow therefore requires thorough attention to scope and sequence. How do we begin to define this scope and sequence in terms of building a web cartographic interface?

While Foote's notions of scope and sequence help provide a roadmap to content mastery, Bampton (2012) articulates the closely linked notions of misconceptions and threshold concepts that play an important role

within student learning processes. *Misconceptions* involve beliefs students bring with them into the classroom that can lead to ‘troublesome’ knowledge; that is, incorrect or counterintuitive understandings that are difficult to right and therefore hinder further learning. Though student learning always will be shaped by their prior beliefs and experiences, identifying such misconceptions can help resolve some of the challenges to overcoming a steep learning curve involving a wide set of complex technical skills. Misconceptions are particularly unproductive, however, when they interfere with student gains concerning threshold concepts, or the “essential ideas which must be grasped for students to advance their understanding of particular topics” (Bampton 2011, 118). Distinct from what educators typically designate as “core concepts,” *threshold concepts* can be thought of as “conceptual gateways” that enable a qualitative shift in perspective and new understandings of the material (Bampton 2011: 120). Threshold concepts may be thought of as blockage points, yet it is more productive to conceive of them as akin to the transformative “aha” moments that promote integration of existing knowledge and allow for an opening up of a “new and previously inaccessible way of thinking about something” (Meyer and Land 2003, 1). While misconceptions may inhibit understanding a threshold concept, Bampton suggests threshold concepts may help students overcome misconceptions.

The roadmap to learning and teaching the web map of today then parallels what we know about GIS&T pedagogy in general and raises questions falling under three broad areas. First, topics and learning objectives must be identified. This involves acknowledging what we have gained from research into traditional, digital, and web mapping, and the solution space for designing a web map (see Table 3.1). It also requires consideration of current changes in the medium of the web itself and the associated challenges and affordances this evolving medium presents to educator, student, and web map developer. Second, these topics and learning objectives must be considered in terms of the scope of meaningful lessons and learning tasks, and the sequence in which they are taught. A fully functional web map involves the integration of many web standards working together, yet these web standards cannot be learned at once. A successful curriculum then may balance the introduction of each standard on its own

terms with its dependency upon other standards to help create a meaningful user experience and effective UI components. Furthermore, the order of that topics and learning objectives are introduced is important, *yet also* highly variable among students. While the ideal mapping process of a professional may be useful for informing the scope and sequencing of learning material, this process must be modified to meet a pedagogical goal. Third, the identification of scope and sequence as a means toward building lessons and learning tasks for students will be further strengthened through attention to misconceptions present in the learning environment and the threshold concepts which can be leveraged to overcome misconceptions and propel student learning gains.

Geographers' concerns with educational questions have shifted in recent decades from "how to educate an elite group of professional experts, to how to provide a basic level of understanding of GIScience principles to everyone" (Goodchild and Li 2010, 15). This goal, however, contrasts sharply with the requisite skill set needed for creating customized cartographic interfaces using the Open Web Platform in particular. As the authors of the popular Data Driven Documents (D3) JavaScript visualization library bluntly assert, "Simply put there is no substitute for writing code" (Bostock and Davies 2013, 133). Such a tension begs for wider consideration of curricular goals and whether a given educational program seeks to merely create educated tool users (such as conventional training using a desktop GIS application such as ArcGIS) or enable students to attain a higher level of technical literacy including geocomputing skills and advanced knowledge of web technologies. This research project aligns itself with the latter.

## **Chapter Three: Competitive Analysis Study and Online Survey**

### **Analysis of open mapping libraries and assessment of current mapping practices and needs**

---

#### **Overview**

This chapter reports upon two studies designed to assess currently available web mapping options and the ways in which they meet the educational needs at the undergraduate and graduate levels. These studies constitute the first two stages of a four-stage process designed to characterize the current landscape of open source web mapping technologies and provide a means for keeping pace with ongoing technological change. The first study used a competitive analysis to identify, evaluate, and compare a wide array of open source, JavaScript-based code libraries. This study provided knowledge of the range of available options, as well as their relative strengths and weakness in the functionality desired for cartographic representation and interaction design. A second study aimed to identify the tools that currently are used by web mapping practitioners within the UW System in order to further strengthen the interpretation of results from the competitive analysis study. This second study aimed to clarify the general awareness of alternative options and garner insights into the training and education processes operating within current web mapping workflows. Together, these two studies provided a deeper understanding of the technical landscape of contemporary web mapping and informed decisions about which subset of mapping libraries technologies may best meet the needs of the UW–Madison Cartography program. Execution of the process additionally generated broad, conceptual insights regarding current trends in web map design.

#### **3.1 Competitive analysis study**

Recent web standards across modern web browsers has promoted a virtual explosion of code libraries capable of meeting at least some of the representation, interaction, and user experience (UX) needs of web map design. Despite their potential, there are few resources that effectively collect and summarize these open web mapping technologies, and fewer that provide insight about how they are best leveraged

both individually and in combination across web mapping contexts. The goal of this study was to gain a better sense of these technologies and to narrow the viable web mapping solutions to those that best facilitate student learning about cartographic representation and interaction.

### **3.1.1 Objectives and methods**

A *competitive analysis study* is a systematic, critical comparison of a suite of related tools or technologies based on their relative merits (Nielson 1994). When the tools or technologies are compared according to established theoretical frameworks, a competitive analysis study is effectively a content analysis of secondary sources, common to archival research in social science. Completing a competitive analysis at the beginning of this project was essential due to the pace of technological change in web mapping and our relative unawareness of emerging options as we transitioned from the Flash-based solution to open alternatives. The competitive analysis represented the widest scoping stage in the process, as it assumed little or no existing knowledge of contemporary technologies and sought to characterize the range of emerging options for web mapping.

The competitive analysis study was designed to formulate initial recommendations of how best to make use of available open web mapping technologies within practical and educational settings. The objectives of this study were threefold: (1) identify the variety of open web mapping technologies currently available, (2) organize these technologies into a consistent and logical framework based on their intended purpose and structure, and (3) compare these technologies according to the functionality they support. To achieve these objectives, the competitive analysis study was completed in three phases: (1) collection, (2) coding, and (3) analysis.

The first phase aimed to identify and compile as many web mapping tools as possible, largely through Internet keyword searches, mapping and data-visualization specific blogs, and social media such as Twitter. This approach thereby made use of the online community of developers for collecting the range of potential web mapping solutions. Technologies were first recorded by name and URL into a shared

online spreadsheet. Given the curricular goals of teaching JavaScript specifically, along with web map design generally, an emphasis was placed on open libraries that can be combined with other JavaScript libraries (e.g., jQuery). We collected the primary webpages (i.e., the secondary sources included in the content analysis) for open source web mapping technologies over a two-week period in Spring 2012, making use of keyword searches, popular blogs, and social media for webpage collection.

After reaching saturation, the second phase analyzed and ranked each mapping option according to a predetermined coding scheme. We evaluated each mapping technology in terms of its purpose, its supported functionality, and its interoperability with other tools based on the documentation included within or linked from the websites. Given that many of the available web mapping technologies were developed by non-cartographers, it also was important to evaluate the technologies according to criteria derived from the GIS&T *BoK* and the pedagogical needs of Web Cartography. We therefore coded the technologies according to the supported *representation* techniques for graphically encoding information and the supported *interaction* techniques for building user interfaces to manipulate the representation. Representation codes included support for basemaps, vector overlays, and linked graphics/charts, as well as support for common thematic map types (Slocum et al., 2009). Interaction codes included support of interaction operators, or the generic kinds of interactive functionality available for manipulating maps and other visualizations (Roth, 2012, Roth, 2013a), as well as support for mobile and location-aware web maps. Twenty-seven (n=27) codes were used in total between the representation and interaction categories; Table 3.1 lists and defines the representation and interaction codes used for the competitive analysis.

To improve reliability, two independent coders applied codes based solely on the documentation included in the collected webpages (i.e., what the webpage promised of the technology) without experimenting with the technology itself (i.e., actually trying to use the technology). Coders followed a four point ordinal scale in their coding: (1) supported, (2) known work-around, (3) requires hack, and (4) not possible, with the average score between the coders ultimately used for a reliability comparison across

technologies. These coding instructions supported a discount, convergent approach to the overall process, the aim of which was to produce a broad understanding of the range of available options. This coding strategy was further justified by the targeted user group of students in the UW Cartography program, who need good documentation to learn and apply new technologies (which is true in part of any developer unfamiliar with a new technology).

Once all found mapping technologies were coded, analysis of these data ensued primarily along two modes. First, mapping tools were differentiated according to variance between supported features and functionality. This allows for identification of technologies that are better suited for particular cartographic tasks, such as creating a thematic map of a particular type versus producing a highly styled basemap. This analysis was useful for identifying open web mapping technologies well suited for the UW Cartography curricular needs, which requires support of the complete array of cartographic tasks rather than a specialized subset. Second, all technologies were analyzed in terms of their support for the breadth of cartographic requirements. This offered insight into the makeup of these emerging mapping tools generally in terms of what cartographic features have come to be expected by users. Together, this approach provided both specific practical information into individual technologies and a way to see the broader landscape of emerging web mapping technologies.

REPRESENTATION		
1	Map vs. Imagery	load different basemap tiles, such as road map, satellite imagery, etc.
2	Basemap Styling	adjust the styling of the basemap
3	Tile Rendering	generate and serve custom maps as tiles
4	Vector Overlays	draw and overlay additional vectors, including points, lines, and polygons
5	Choropleth	generate a choropleth map
6	Proportional Symbol	generate a proportional symbol map
7	Dot Density	generate a dot density map
8	Isoline/Surface	generate an isoline or surface map
9	Flow	generate a flow map
10	Cartogram	generate a cartogram
11	Bivariate/Multivariate	depict two or more statistical variables on the map
12	Animation	animate the map over a time series
13	Graphics/Charts	add additional information graphics or charts to the map
INTERACTION		
1	Arrange/Linked Views	manipulate the layout of the map and linked views
2	Reexpress	change the displayed map type
3	Sequence	generate an ordered set of related maps or change the map from the sequence that is shown
4	Resymbolize	change the design parameters of a map without changing the map type
5	Overlay/Toggle	adjust the feature types included in the map
6	Reproject	change the map projection
7	Pan	change the geographic center of the map
8	Zoom	change the scale or resolution of the map
9	Filter	alter the map to remove map features that do not meet one or a set of user-defined conditions/constraints
10	Search	alter the map to add/indicate a particular location or map feature of interest
11	Retrieve	request specific details about a map feature of interest
12	Calculate	derive new information about a map feature of interest
13	Mobile Support	support for viewing and interacting with the map on a mobile device
14	Location Aware	support for collecting and mapping information about the user's location

**Table 3.1** The representation and interaction codes used to compare the collected suite of open web mapping technologies.

### 3.1.2 Results and discussion

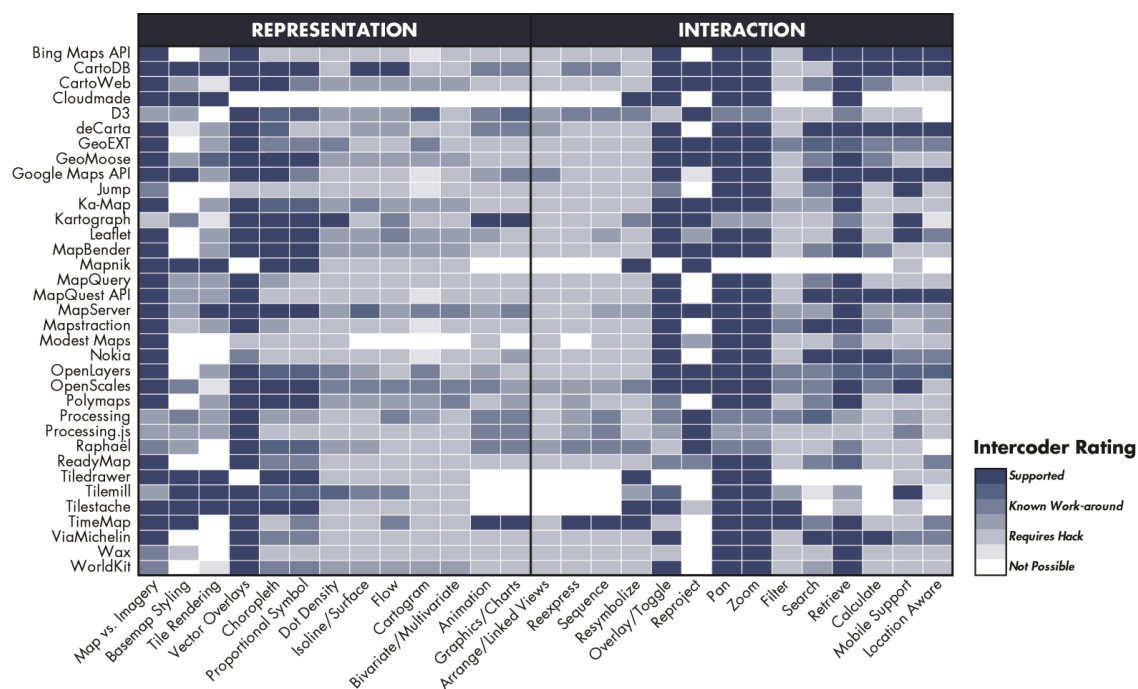
In total, thirty-five ( $n=35$ ) web mapping technologies were identified for the competitive analysis during this timeframe<sup>22</sup>. Results of the competitive analysis study are illustrated in Table 3.2. In the matrix, the darkest blue shading indicates a representation or interaction technique that was coded as ‘supported’ by both coders and the white shading indicates a technique that was coded as ‘not possible’ by both coders.

When interpreted horizontally, the matrix allowed us to identify similarities and differences in supported functionality across the 35 reviewed web mapping technologies. The competitive analysis revealed a

<sup>22</sup> Since the initial coding in Spring 2012, developers have released many additional technologies. In addition, several of the technologies evaluated have undergone significant changes and upgrades, and some have been deprecated.



basic distinction between specialist web mapping technologies designed to support a small subset of specific functions (e.g., Cloudmade Editor, Mapnik, Modest Maps) and multi-purpose web mapping technologies designed to support numerous functions (e.g., CartoDB, D3, the Google Maps API, Leaflet, MapServer, OpenLayers/OpenScales). Individual technologies generally fell into one of the following categories: (1) *frameworks* (n=10; 28.6%) providing a full stack of client- and server-side technologies (e.g., GeoMoose, MapServer, Processing), (2) *open libraries* (n=14; 40.0%) supporting client-side map rendering (e.g., D3, Leaflet, OpenLayers), (3) *closed APIs* (n=6; 17.1%) exposing a subset of functionality for creation of web map mashups (e.g., the Bing Maps API, the Google Maps API, the MapQuest API), and (4) *tile rendering services* (n=5; 14.3%) facilitating the rendering and serving of basemap tiles (e.g., Cloudmade Editor, TileMill, TileStache). The large majority of the reviewed technologies (n=28; 80.0%) leveraged JavaScript as the base programming language, with four (n=4; 11.4%) exclusively leveraging CSS or the CartoCSS variant used for tile rendering, one (n=1; 2.9%) leveraging Java, one (n=1; 2.9%) leveraging PHP, and one (n=1; 2.9%) leveraging ActionScript.



**Table 3.2** Results of the competitive analysis study. Collection and coding was completed in Spring 2012; therefore, the matrix is no longer complete nor accurate, although arguably it never can be given the speed of technological advancements in web mapping. The matrix does provide a snapshot in time of web mapping technology that is useful for understanding general patterns and emerging trends in web map design.

From the competitive analysis by technology, we identified open libraries implemented in JavaScript as the most suitable technological form for the curricular and institutional needs of the UW Cartography program. Open libraries can be combined flexibly with other non-mapping JavaScript libraries (e.g., jQuery) and can be extended more easily to implement non-natively supported representation and interaction functionality, two advantages that open libraries hold over closed APIs. Full stack frameworks, while more powerful and feature-complete than open libraries, are less approachable for a single, semester-long course and require background on server-side databases outside the scope of a course on Interactive Cartography and Geovisualization. Finally, the opportunity to teach and practice interaction design is limited with tile rendering services in comparison to the other forms of technologies.

When interpreted vertically, Table 3.2 provided us with a snapshot of trends in contemporary web map design. Widely supported representation functionality included custom vector overlays (n=29 supported; 82.9%), loading of map versus imagery basemaps (n=26; 74.3%), and choropleth (n=19; 54.3%) or

proportional symbol (n=16; 45.7%) thematic maps. Overall, the competitive analysis suggested a general focus on reference mapping over thematic mapping in existing web map technologies, as most of the reviewed technologies required a custom hack to implement advanced thematic map types beyond the choropleth and proportional symbol techniques. The lack of support for advanced thematic mapping is a real and significant gap between contemporary web mapping practice and traditional cartographic scholarship that should be addressed as web design and cartographic design continue to collide. Basemap styling and tile rendering exhibited the greatest variation in support across technologies; both were supported by eight (n=8; 22.9%) technologies, but not possible in thirteen (n=13; 37.1%) technologies. This variation was explained by inclusion of tile rendering services in the competitive analysis, rather than restriction to frameworks, libraries, and APIs.

Widely supported interaction functionality included panning (n=29; 82.9%), zooming (n=29; 82.9%), retrieval of details using an information window (n=25; 71.4%), and overlay of context layers (n=24; 68.6%). Arguably, these four interaction operators (overlay, pan retrieve, zoom) along with a multiscale reference basemap have coalesced to define the prototypical web map, an extension to the combination of panning and zooming explicit in the colloquial use of 'slippy map'. Tracking the evolution of the prototypical web map is useful for cartographers, as it exposes the expectations of non-specialist web map users and reveals potential gaps in contemporary design solutions due to technology constraints. Such gaps included support for reexpress (not supported natively by any technology; 0.0%), filter (n=2; 5.7%), and calculate (n=8; 22.9%). Reexpress and filter are considered important for exploratory visualization, while calculate is essential for advanced WebGIS. Dynamic reprojection exhibited the greatest variation across technologies, which was supported by sixteen (n=16; 45.7%) technologies but not possible in fifteen (n=15; 42.9%). Many of the technologies supporting reprojection were limited to a small set of cylindrical projections, further defining the prototypical web map as a reference map served as raster tiles. Finally, eleven (n=11; 31.4%) of the technologies natively included responsive mobile support, but only six (n=6; 17.1%) were location aware. Such a finding suggested that cartographic design for mobile

has garnered some attention in client-side web map design, but the implementation of location-based services using the Open Web Platform has been limited to-date.

### **3.2 Needs assessment survey**

Results from the competitive analysis study were further strengthened through triangulation with a needs assessment survey of the Cartography community across the University of Wisconsin (UW) System, a network of twenty-six university campuses across Wisconsin. The online survey acted as a *needs assessment study*, as the purpose of the survey was to elicit past experiences with the collected technologies as well as to identify future or currently unmet web mapping needs (Wiggins and French, 1991). We included the survey as the second step in the overall process in order to quickly acquire feedback about technologies collected in the competitive analysis from designers and developers outside of the project team. We chose the online survey format over interviews or focus groups given the discount, convergent approach to the overall process.

#### **3.2.1 Objectives and methods**

A *survey* is a user-based method that requires participants to respond to predetermined questions (Suchan & Brewer 2000). Survey participation was targeted toward UW System faculty who teach, train, and manage students studying web mapping as well as UW System employees (staff and graduate students) who perform a range of mapping and GIS-related tasks online. The survey design aimed to establish the current needs, practices, and values of the immediate mapping community, with the goal of comparing these insights to the trends in open web mapping technologies identified from the competitive analysis. Specifically, the following objectives guided the survey construction: (1) determine the web mapping technologies currently used within the UW System, and the motivation for implementing these technologies, (2) identify web mapping design and development needs that currently are not being met, (3) ascertain the level of familiarity with alternative web mapping technologies that might support these

needs, and (4) identify challenges in teaching and learning alternative web mapping technologies that restrict their adoption.


The survey question protocol consisted of four sections (Appendix A): (1) basic biographic information, (2) current use of the technologies identified in the competitive analysis, (3) aspects of technologies that should be considered when selecting a technology, and (4) overarching opinions on designing web maps and teaching web map design. The non-biographical sections of the survey included twelve questions in total, with four Likert scale questions and eight free response questions. The Likert scale questions aimed to identify aspects of web mapping technologies that must be considered when selecting an appropriate solution (see Appendix A for the complete survey protocol).


### 3.2.2 Results and discussion


Twenty-one (n=21) UW employees participated in the online needs assessment survey in the Spring of 2012. Participation was limited to individuals who either develop web maps as part of their work responsibilities or supervise individuals who develop web maps. Eight (n=8) of the participants were UW staff, eight (n=8) were UW graduate students, and five (n=5) were UW faculty. Table 3.3 describes the frequency with which the participants used geographic information, made print maps, and developed web maps as part of their daily work.


	How frequently do you:					
<i>use geographic information</i>	13	7	0	0	0	1
<i>make print maps</i>	2	6	4	3	4	2
<i>develop web maps</i>	4	5	6	1	2	3
	daily	weekly	monthly	yearly	never	I supervise this activity

**Frequency**

 n=13

 n=7

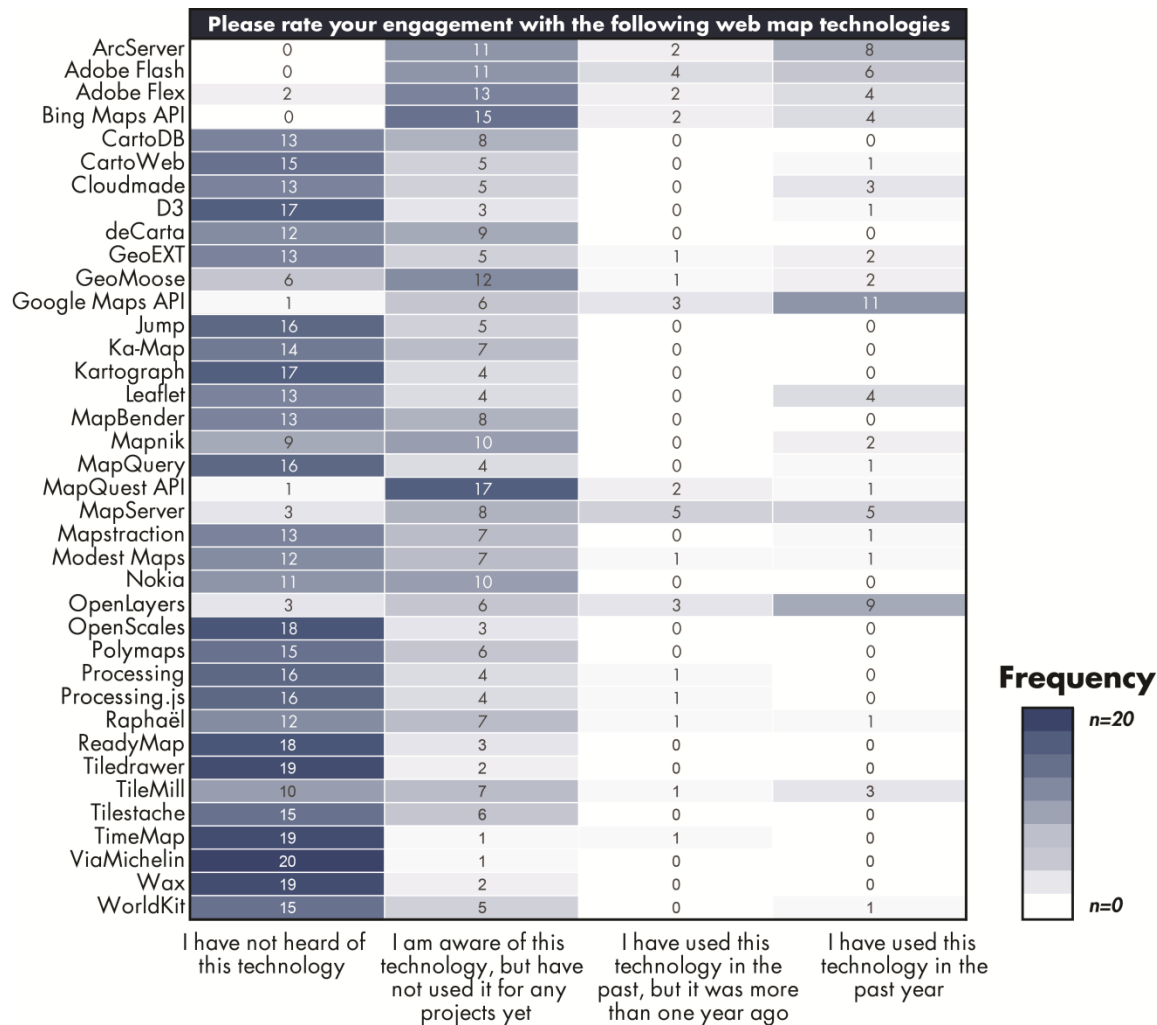
 n=6

 n=0

**Table 3.3** The frequency with which participants in the online needs assessment survey use geographic information, make print maps, and develop web maps as part of their daily work.

The first section of questions addressed existing use of the web mapping technologies that we collected through the competitive analysis. Table 3.1 presents the frequency that survey participants were aware of or had used the collected set of web mapping technologies. We listed three proprietary technologies in this section of the online needs assessment survey—Esri ArcGIS Server, Adobe Flash, and Adobe Flex—as a baseline against which to compare the collected set of open source web mapping technologies, resulting in evaluation of 38 technologies in total.

Survey participants had used just a subset of the collected technologies. Only the Google Maps API was used by a majority of participants in the past year ( $n=11$ ; 52.4%), with OpenLayers ( $n=9$ ; 42.9%), ArcGIS Server ( $n=8$ ; 38.1%), and Adobe Flash ( $n=6$ ; 28.6%) used in the past year by a large minority. There were several technologies that numerous participants were aware existed, but had never used themselves, most notably the MapQuest ( $n=17$ ; 81.0%) and Bing Maps ( $n=15$ ; 71.4%) APIs, the GeoMoose framework ( $n=12$ ; 57.1%), and the ArcServer ( $n=11$ ; 52.4%), Adobe Flash ( $n=11$ ; 52.4%), and Adobe Flex ( $n=13$ ; 62.0%) proprietary technologies. Overall, participants had not heard of the majority of the surveyed technologies, including technologies that have gained in popularity across the cartographic community since administering the survey, such as D3 ( $n=17$ ; 81.0%), Processing/Processing.js ( $n=16$ ; 76.2%), CartoDB ( $n=13$ ; 62.0%), and Leaflet ( $n=13$ ; 62.0%). Such a shift in awareness signals the fast pace of technological change in web mapping. This finding also justified our pairing of the competitive analysis study with the online needs assessment study, as reliance on an internal survey alone would have limited discussion to a small subset of available technologies (e.g., the Google Maps API, Open Layers, ArcServer) not fully representative of the trajectory of web mapping at the time.



**Table 3.4** The level of engagement with the set of web mapping technologies gathered through the competitive analysis study. The proprietary technologies ArcServer, Adobe Flash, and Adobe Flex are added to the top of the table to provide a comparison against open source technologies.

Open-ended comments regarding the technologies that participants continued to leverage versus those they completely abandoned revealed broad awareness of the technology transition in web mapping underway at the time. Overall, participants acknowledged the move towards the Open Web Platform and JavaScript, with one participant stating “In testing technologies for next generation of web apps, we’re quickly moving toward primarily JavaScript-based frameworks” and a second adding “I am going to transition to JavaScript.” This discussion provided further justification for narrowing our focus to JavaScript-based technologies and continued to signal JavaScript itself as an important threshold concept to address within future pedagogical considerations. Looking towards the future, several participants

suggested a move away from closed APIs and towards full-stack frameworks or client-side mapping libraries. One participant indicated that their program does not “employ programs like Bing and Google Maps API unless students are working on navigational aids,” and a second stated “I suspect the Google Maps API is on its way out.” A third participant gave justification for the move away from closed APIs, stating that the “advancement of many of these libraries/frameworks [provides] highly-customizable standard mapping interface components and interaction behaviors.” Thus, responses to the first section of questioning revealed a general preference for openness and extensibility, but an overall poor awareness of the emerging frameworks and libraries that could be used in place of closed APIs and proprietary technologies.

The second section of questions solicited feedback about the qualities of web mapping technologies that should be considered when selecting an appropriate technology or set of technologies. Figure 3.1 presents a series of box plots depicting participant responses to five-point Likert ratings ranging from ‘not important’ to ‘essential’. The box plots are organized according to three qualities of web mapping technologies: (1) design characteristics of the resulting web map; (2) technical considerations, emphasizing constraints in applying the technology associated with hardware or software; and (3) practical considerations, including other non-functional constraints when applying the technology.

Participants rated interactivity as the most essential characteristic of web maps that should be supported by a web mapping technology (mean=4.50; median=5), with no participant rating interactivity lower than a ‘3’ (‘important’). Such a finding justified inclusion of both representation and interaction functionality in the competitive analysis coding (Table 3.2), and reflected the growing importance of UI and UX design to web mapping specifically, and the discipline of Cartography broadly. Participants also listed interface design aesthetics (mean=4.00; median=4), multiscale (mean=3.95; median=4), and scalability (mean=3.95; median=4) as important aspects of web map design that must be supported in the underlying technology. Participants rated animation as the least essential property of web maps to consider when selecting a technology (mean=2.30; median=2), a surprising finding given the substantial body of

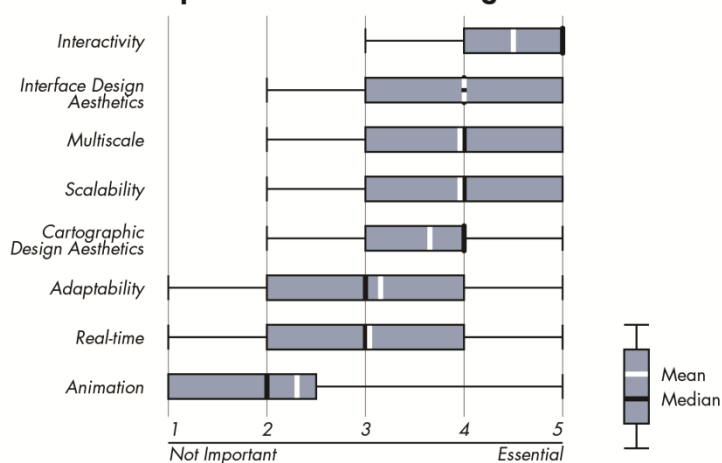


research on animation in the cartographic literature. Overall, Likert scale ratings on web map characteristics suggested an increase in importance on user-driven display changes (i.e., interactivity) and a decrease in importance on system-driven display changes (i.e., animation and real-time updates) in contemporary web map design.

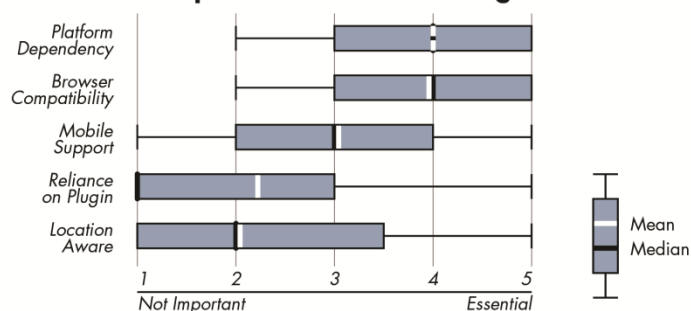
Participants rated platform dependency as the most important technical consideration for web mapping (mean=4.00; median=4), directly followed by browser compatibility (mean=3.95; median=4). As reviewed above, cross-browser and cross-platform compatibility were major advantages to using plugin-based technologies for web mapping through the mid-2000s, and the sharp decline in cross-platform compatibility, specifically mobile support, was an important driver away from plugins like Flash Player in the early 2010s. Participant responses regarding technical considerations indicated that cross-browser and cross-platform compatibility remain a high priority in web mapping, and provides further justification for leveraging frameworks and libraries that can be used in combination with other open libraries that enable cross-browser and cross-platform compatibility. Location awareness was rated as the least important technical consideration (mean=2.05; median=2), providing further evidence that implementation of in-browser location-based services was not common at the time of conducting the survey.

Finally, participants rated maintenance/stability as the most important practical consideration when selecting a web mapping technology (mean=4.05; median=4). Poor long-term maintenance and source code instability historically have been criticisms of open source technologies, but are improving as the open source web mapping community strengthens and matures, adding clout to the enforcement of open web standards. High quality code documentation (mean=3.70; median=4) and tutorials/examples (mean=3.50; median=3) also were listed as important practical considerations, both of which aid in learning a new technology as well as keeping one's skills up-to-date as the technology evolves.

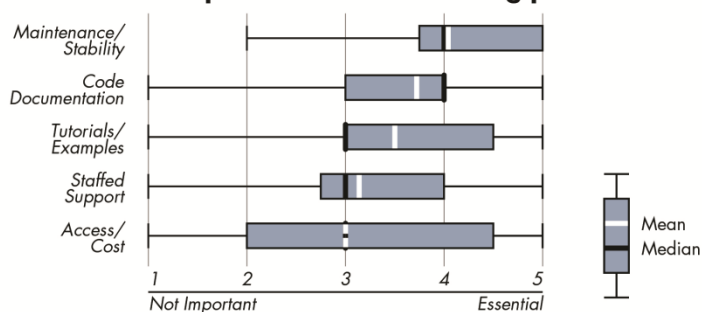
**A. Please rate importance of the following characteristics of web maps:**



**B. Please rate the importance of the following technical considerations:**



**C. Please rate the importance of the following practical considerations:**



**Figure 3.1** The importance of different qualities of web mapping technologies when choosing an appropriate technology or set of technologies: (A) web map characteristics; (B) technical considerations; (C) practical considerations.

Opinion was split across surveyed participants regarding the value of open source technology. Access/cost was rated as the least important practical consideration (mean=3.00; median=3), a finding that contrasts with the above participant comments about transitioning to open source frameworks and

libraries. Responses to the access/cost Likert scale revealed a divergence in opinion regarding open source web mapping technology, with nine (n=9) participants listing access/cost as ‘not important’ or only ‘somewhat important’ and eight (n=8) participants listing access/cost as ‘very important’ or ‘essential’. One participant shed light on this bimodal distribution in an open-ended response, stating “I personally think opens source is a great ideal...but not as important as people make it sound.” This participant went on to state that “There are many good open source products...there are many good closed products too. I will use whatever software is most user-friendly and easily adaptable. I don't care if it is open or closed.” A second participant stated “Increasingly it is a blurry line between commercial, open source, [and] cloud-based options and hybrid applications utilizing all of these are a growing trend.” Therefore, it is important to remember that good design matters more than novel tools, and that a robust Cartography curriculum should introduce students to a representative portfolio of industry-standard technologies, open and proprietary. Because of this feedback, we decided to include one closed-API in the diary study (see Chapter 4) to enable consideration across different degrees of openness.

The third and final set of questions in the online needs assessment survey solicited approaches used by the participants to keep pace with evolving web mapping technologies. The most common strategy listed for experimenting with new web mapping technologies was the completion of a pilot study or proof-of-concept prototype (n=6), followed by working through posted examples and tutorials (n=3), reading other developers’ experiences on forums and web blogs (n=2), and directly reviewing the available documentation (n=2). In particular, completion of a pilot study falls in line with our four-stage process, as the process represents a structured, repeatable approach to prototyping. Participants indicated that before they are willing to experiment with a technology, they need details about its cost (n=3), documentation (n=3), examples and tutorials (n=3), server requirements (n=3), development environment (n=3), functional capabilities (n=2), base programming language (n=2), security (n=1), stability (n=1), and supported data formats (n=1).

Some participants (n=3) indicated they rarely experiment with new technologies, with one participant stating “experimentation does not occur too much unless someone requests the change” and a second stating “we know what we know and use it and tweak it to the utmost...we only really evolve if we learn of a new software or plugin that fits with our current ecosystem.” A third participant indicated that experimentation is limited because “resources [are] committed to existing projects...and [we] don’t usually pick technologies on a project-by-project basis,” and went on to say that too much experimentation may lead the team to “become novices in many technologies instead of proficient in a few.” Therefore, constraints on resources and time may lead to path dependencies, with a program or firm leveraging the same web mapping technology long beyond its functional viability. Again, these comments fall in line with our recommended process for running a pilot study, as the convergent, discount approach enables effective use of resources and time. Additional barriers to learning new web mapping technologies listed by participants included poor or incomplete code documentation (n=4), poor or incomplete examples and tutorials (n=4), difficulty in knowing where or how to get started (n=2), limited awareness of available technologies (n=1), difficulty in working across a stack of technologies (n=1), and the prerequisite of learning a new programming language, such as JavaScript (n=1).

### **3.3 Conclusion: implications for Web Cartography education**

*Research Question #1: What technologies currently are available for web mapping and how do they vary?*

The surveyed open source web mapping technologies took one of four forms: frameworks, open libraries, closed APIs, and tile rendering services. The large majority of technologies surveyed in the competitive analysis leveraged JavaScript as the base programming language, and thus integrated with the Open Web Platform broadly. Multiple participants in the needs assessment survey noted the growing importance of learning and applying JavaScript-based technologies. Survey participants had used a small subset of the technologies identified through the competitive analysis study, and were unaware of a large majority of

these technologies. We ultimately identified open libraries as most appropriate for an advanced class on Interactive Cartography and Geovisualization, as frameworks required too many additional competencies, closed APIs were limited in their openness and extensibility, and tile rendering services provided limited opportunity to teach interaction design.

Insights from the competitive analysis and needs assessment were triangulated to identify four candidate technologies for inclusion in a scenario-based diary study and further consideration for integration within our advanced-level Interactive Cartography and Geovisualization course (Chapter 4). As described above, we placed an emphasis on open libraries implemented in JavaScript, given the curricular and institutional needs of the UW Cartography program, yet maintained one closed API for comparison:

- *OpenLayers* is an open library based in JavaScript supported by the OSGeo community (<http://openlayers.org/>). We selected OpenLayers (Version 2.12) for the diary study because it was the most robust in terms of supported functionality across the reviewed open libraries (Table 3.2) and was the most frequently used open library (n=9; 42.9%) by participants in the needs assessment survey, and second most frequently used technology overall, behind the Google Maps API. With its initial release in 2006, OpenLayers also had a level of long-term maintenance and stability uncommon to other open web mapping libraries reviewed in the competitive analysis study, an important consideration identified through the needs assessment survey.
- *Leaflet* is an open JavaScript library pioneered and maintained by Vladimir Agafonkin (<http://leafletjs.com/>). Leaflet supports SVG rendering within Internet Explorer 7 and 8, one advantage over most other open libraries using the SVG specification for client-side rendering. At the time of writing, Leaflet was considered among the best web mapping libraries when designing for mobile devices because of a small file size (28MB in Version 0.4) and support of touch-based interactions. However, Leaflet was among the newest technologies included in the competitive analysis study, and was not a commonly used technology among the needs assessment participants (Table 3.4). We included Leaflet in the diary study due to the above advantages, and because it was

the second most robust open library in terms of supported functionality, following the OpenLayers/OpenScale combination (Table 3.2).

- *D3* (Data Driven Documents) is an open JavaScript library pioneered and maintained by Mike Bostock (<http://d3js.org/>). We selected D3 (Version 2.0) over other open libraries that natively supported a similar amount of functionality because of its unique approach to client-side rendering and interaction. Unlike tile-based technologies, D3 explicitly supports dynamic projection of linework into a wide array of map projections, using SVG to draw the projected vectors in-browser. Further, D3 was designed to support rendering of any interactive visualization, not just maps, and therefore offers potential for multiview, coordinated geovisualization unavailable by alternative web mapping technologies.
- *The Google Maps API* is a JavaScript API made available by Google for the creation of slippy map mashups (<https://developers.google.com/maps/>). As reviewed in Chapter 2, the AJAX-based Google Maps, and its subsequent API release, was an important innovation in web mapping, giving rise to the multiscale, slippy map mashup. We selected the Google Maps API (Version 3.0) because it was the most robust in terms of supported functionality across the closed APIs reviewed in the competitive analysis study and was the most commonly used technology (n=11; 52.4%) by participants in the online needs assessment study. At the time of this writing, the Google Maps API was only partially open, and carried with it many usage restrictions, including a maximum number of website visits before Google charges for use of its web mapping service. The Google Maps API therefore served as a baseline in the diary study against which to compare the open libraries without usage restrictions.

*Research Question #2: What are the important characteristics of web maps that should inform the selection of web mapping technologies?*

The competitive analysis revealed notable patterns in supported representation and interaction functionality across open web mapping technologies. The majority of technologies natively supported reference maps served as a set of raster tiles in a cylindrical projection as well as panning, zooming,

retrieval of details, and overlay of context information. Altogether, this representation and interaction functionality defines the prototypical web map. The competitive analysis also revealed gaps between contemporary web mapping practice and traditional cartographic scholarship, including thematic mapping beyond choropleth and proportional symbol representation techniques and the calculate, filter, reexpress, and reproject interaction operators. Finally, participants in the needs assessment survey placed an emphasis on support for user-driven display changes (i.e., interactivity) when choosing a technology, and placed less importance on system-driven displays changes (i.e., animation and real-time updates).

## Chapter Four: Diary Study

### Implementing web standards solutions to meet mapping needs

---

#### Overview

This chapter builds upon results from the two studies reported in Chapter 3 to test a subset of open web mapping technologies against a common set of cartographic requirements. We selected a subset of four candidate technologies (see Chapter 3) and tracked the process of building a web map using each technology through a diary study. A final exit survey garnered additional feedback from the cumulative experience of the diary study. The pair of studies constituted the final two stages of a four-stage process designed to characterize the current landscape of open source web mapping technologies and provide a means for keeping pace with ongoing technological change. These two studies met the practical goals of identifying viable technologies for our curricular needs, and also generated new conceptual insights into the process of making maps on an unfamiliar development platform. In short, a greater understanding of a new web mapping workflow emerged, and we were able to isolate parts of the process where cartographic interface design and development occurs. While in many ways this new workflow comes to look more like that of a typical website developer, particularities to Cartography remain integral to the process of web mapping.

#### 4.1 A diary study tracking development across a web mapping scenario

We combined insights from the competitive analysis study and online needs assessment survey to identify a subset of four candidate technologies holding potential to meet the learning and design needs of the UW Cartography program (see Chapter 3). We then evaluated the four candidate technologies using a *diary study*, a variation of participant observation that requires participants to ‘self-observe’ as they complete an activity (Marsh and Haklay, 2010). In the diary study, participants developed a case study web map using one of the four candidate technologies and recorded their progress in an online journal. We selected a diary study for the third step in the process because it provided the deep development experience with a given technology needed to properly assess its advantages and limitations, but did so in a discount,



convergent manner by relying on prior methods to reduce the total number of technologies under consideration. As described in Chapter 3, we selected D3, the Google Maps API, Leaflet, and OpenLayers for inclusion in the diary study.

#### **4.1.1 Objectives and methods**

In order to implement and evaluate four emergent mapping solutions, we conducted an in-depth diary case study over the course of three months in Summer of 2012. The objectives of this study were threefold: (1) determine and weigh the relative strengths and weaknesses of the selected subset of candidate technologies against one another, (2) explore the way in which students engage with a given library, its API reference documentation, and other learning resources to understand and utilize its capabilities, and (3) better understand how the Open Web Platform is used as a platform for mapping development, as well as the workflow required to accomplish given mapping objectives.

Four students representative of the targeted user group were recruited to complete an example web mapping scenario, each using a different candidate technology. To improve reliability in the diary entries, while remaining cognizant of participant fatigue, I completed the same web mapping scenario with all four technologies. As a result, there were eight ( $n=8$ ) diaries total, two for each of the candidate technologies. All participants had taken one introductory course on cartographic design and one advanced course on web mapping. Participants also were required to complete the lynda.com video tutorials on HTML, CSS, and JavaScript before beginning the diary session.

At the start of the diary study, we introduced participants to a web mapping scenario representative of the kind of contract work completed at the University of Wisconsin-Madison Cartography Lab. We presented the web mapping scenario as a client request for a web map depicting energy consumption by country over the past thirty years and included a requirements document outlining the project scope (Appendix B). We provided the energy time series dataset to the participants as a CSV file. As with the competitive analysis discussed in Chapter 3, requirements for the web mapping scenario were split between

representation and interaction techniques. Representation requirements covered elements of effective design for classed choropleth and graduated symbol maps, including traditional cartographic design topics such as aesthetics, classification, typography, and visual hierarchy as well as emerging cartographic design topics enabled by digital media, such as animation, linked information graphics, and visual storytelling. The specific representation requirements included in the diary study deviated from the representation codes included in the competitive analysis due to the narrowed focus upon only two thematic map types. Interaction requirements were specific to the interaction operators included in the competitive analysis, and included one additional requirement for interface design aesthetics. We included 24 requirements in total in the web mapping scenario. Table 4.1 lists and defines the representation and interaction requirements used for the diary study.

REPRESENTATION		
1	Classed Choropleth	generate a classed choropleth map
2	Graduated Symbol	generate a graduated symbol map
3	Animation	animate the map over the included time series
4	Typography	label map features following typographic conventions
5	Classification	use an equal interval classification scheme for both maps
6	Legend	dynamically redraw a map legend to match the displayed map type
7	Highlighting	include a highlighted variant of each map feature to indicate selection
8	Information Graphic	include a line graph showing the signature of the selected country in comparison to the United States and the median value for the year
9	Visual Hierarchy	style the basemap to produce a strong visual hierarchy
10	Storytelling	provide a title and supplementary text to introduce the map subject and purpose
11	Cartographic Design Aesthetics	customize the look and feel of the map itself to fit the scenario
INTERACTION		
1	Reexpress	change the displayed map type between classed choropleth and graduated symbol
2	Sequence	include standard VCR controls (play, stop, step, back) to control the animation
3	Resymbolize	change the number of classes used for the classed choropleth or graduate symbol map
4	Overlay/Toggle	toggle between a road map and aerial image for the basemap
5	Reproject	set the map projection to equal area for the classed choropleth map and conformal for the graduated symbol map
6	Pan	change the geographic center of the map
7	Zoom	change the scale and/or resolution of the map
8	Filter	filter the map according to the attribute range using a two-thumb slider
9	Search	search for a specific country
10	Retrieve	highlight a probed map feature and activate an associated information window with details about the feature
11	Calculate	dynamically calculate deviation of the map feature from median (i.e., percentile)
12	Link	coordinate retrieve on the map with the line graph to show the selected feature on both graphics
13	Interface Design Aesthetics	customize the look and feel of the interface to the map to fit the scenario

**Table 4.1** The representation and interaction requirements of the diary study.

We gave participants a total of 40 hours of development time to complete as many of the 24 requirements as possible, mimicking constraints of a standard workweek. Given the lack of familiarity with the assigned technology—and development practices on the Open Web Platform in general—we did not expect participants to implement all requirements within the provided time period. Instead, we instructed participants to implement what they considered as ‘easy’ requirements before moving onto more difficult ones. Therefore, the requirements that participants ultimately implemented indicate functionality that likely was natively supported by the technology, rather than functionality needing a work-around or custom code solution. We did not allow participants to integrate multiple technologies into their web map solutions in order to identify the limitations of each technology in isolation.

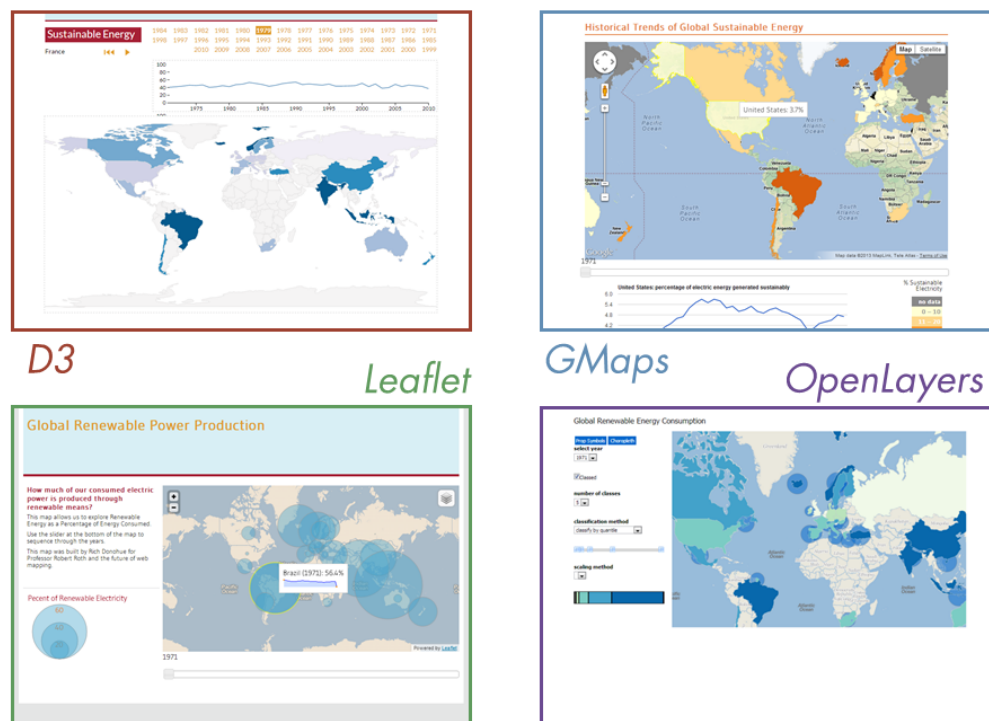
Participants logged a diary entry every hour across the 40-hour period. Within each diary entry, participants were asked to describe: (1) the requirement(s) they implemented in the past hour, (2) key frustrations or breakthroughs in the past hour, and (3) their current satisfaction with the web mapping technology drawing from a provided list of 125 emotions.<sup>23</sup> With regard to the latter, we selected the larger list of moods—rather than more terse taxonomies of affective or emotional experiences (e.g., Plutchik, 1980, Feldman-Barrett and Russell, 1998)—to give participants greater flexibility and precision in describing their emotional state. Participants also were required to capture a screenshot of the web map and a version of their code with each hour of the diary study (Haklay & Zafiri 2008).

#### **4.1.2 Results of the diary study**

Figure 4.1 presents example solutions to the energy web map scenario completed within the 40-hour time limit of the diary study, illustrating the relative affordances and constraints in web map design of the four candidate technologies.

---

<sup>23</sup> We derived the list of moods from an online technical product aiming to generate a hex value for a color based upon list psychological association, available at: <https://github.com/hazbo/moodswing2>



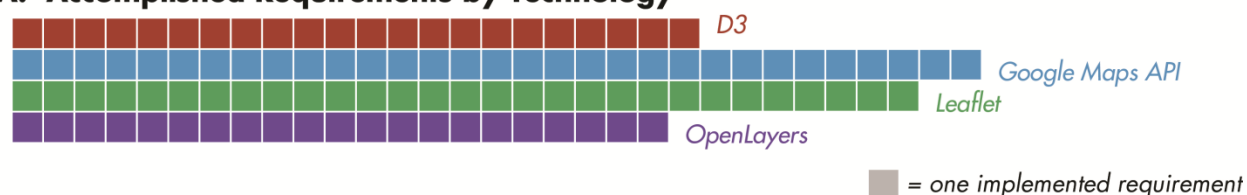
**Figure 4.1** Example solutions for the energy web mapping scenario resulting from the diary study: **(red)** D3; **(blue)** Google Maps; **(green)** Leaflet; **(purple)** OpenLayers.

Figure 4.2 presents an overview of the diary study results. Figure 4.2a illustrates the total number of diary study requirements by candidate technology. Again, each candidate technology had a pooled sample size of two, resulting in a maximum of 48 requirements per technology (2x24). On average, participants completed the most scenario requirements using the Google Maps API ( $n=31$ ; 64.6%), with Leaflet a close second ( $n=29$ ; 60.4%). Fewer requirements were accomplished with D3 ( $n=22$ ; 45.8%) and Open Layers ( $n=21$ ; 43.8%).

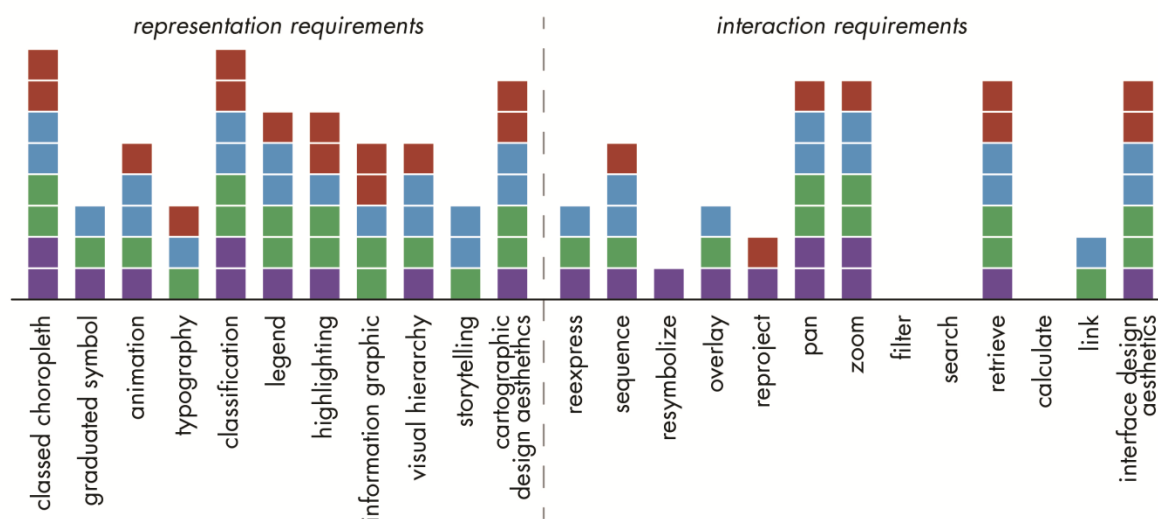
Figure 4.2b reorganizes the diary study results by individual scenario requirements. There was substantial variation in the final maps by individual requirement, with the choropleth map and dynamic classification the only features implemented in all eight ( $n=8$ ) diary sessions. Overall, many more representation requirements (59 total, or 7.4 per diary session) were implemented compared to interaction requirements (44 total, or 5.5 per diary session), despite the energy scenario including 11 representation requirements and 13 interaction requirements. Such a finding reflects the primacy of representation over interaction in the web development workflow, and potentially decreased native support for interaction versus

representation across the candidate technologies. This also suggests a recommended sequence for learning and implementing geographical information representation prior to applying interaction techniques. The operators pan ( $n=7$ ), zoom ( $n=7$ ), and retrieve ( $n=7$ ) were implemented in the large majority of diary sessions, but the fourth common operator overlay ( $n=3$ ) was implemented less frequently, likely in part due to its decreased relevance to the thematic mapping energy scenario. Looking at the absences, calculate, filter, and search were not implemented in any web map ( $n=0$ ), suggesting increased difficulty in implementing these operators across the candidate technologies (and perhaps in all web mapping technologies broadly). These deficiencies in native support among the web mapping technologies point toward unsupported aspects of web mapping scope requiring additional educational attention.

### A. Accomplished Requirements by Technology



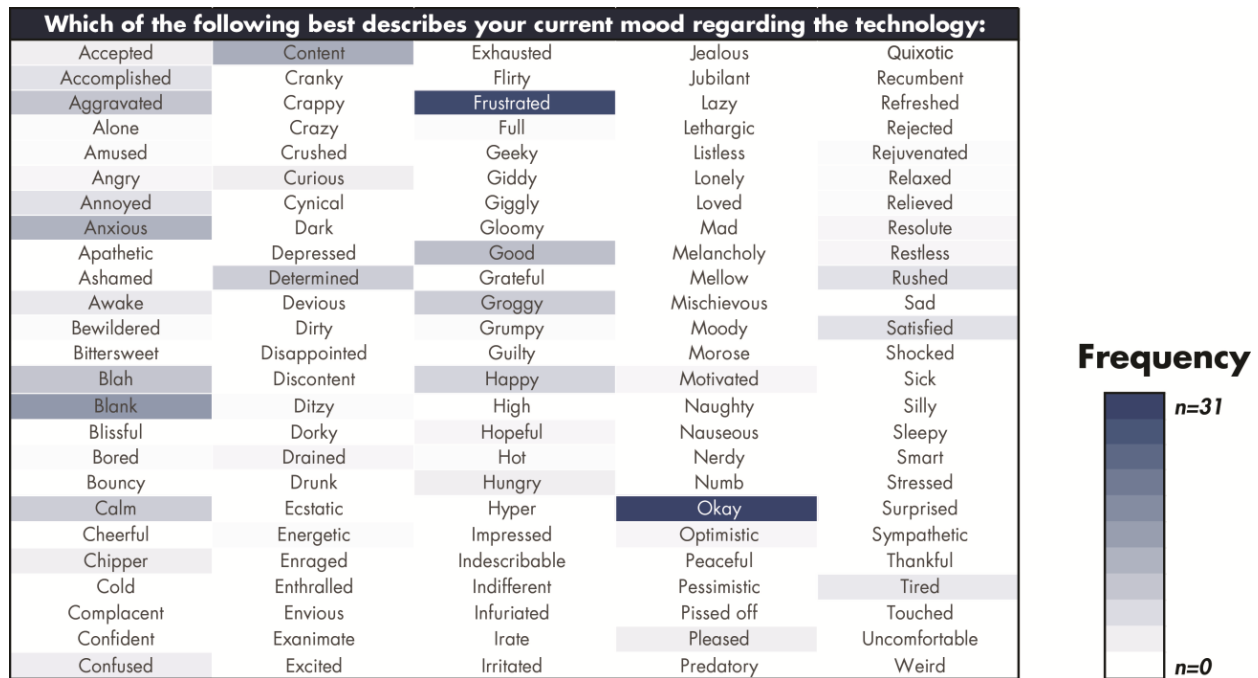
### B. Frequency by Individual Requirement



**Figure 4.2** Overview of the diary study results: (a) total frequency of accomplished requirements by technology; (b) frequency of individual accomplished requirements.

There were several differences across candidate technologies that suggest their relative affordances and limitations. Visual storytelling and live linkage between graphics were implemented using the Google Maps API and Leaflet, but not D3 or OpenLayers. The reproject operator was implemented in D3 and OpenLayers, but not the Google Maps API or Leaflet. The resymbolize operator was implemented using D3 only. There also were several gaps in which a requirement was implemented in three of the four technologies, suggesting a limitation of the absent technology. Such gap requirements included the graduated symbol map and the reexpress operator for D3 and typography, a linked information graphic, and the overlay operator for OpenLayers.

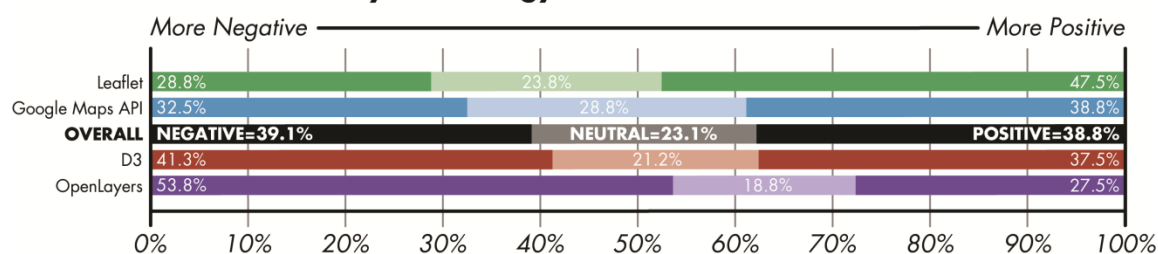
The recorded moods listed in the diary entries were coded according to their valence (positive, neutral, and negative) for subsequent analysis. Table 4.2 provides an overview of the participants' emotional experience while working with their candidate technologies. Overall, participants used 65 of the 125 unique terms to describe their emotional status across the eight diaries, supplying these 65 moods a total of 320 times across the diary study (8 diaries, each with 40 entries). OpenLayers yielded the most unique terms ( $n=37$ ), followed by the Google Maps API ( $n=35$ ), D3 ( $n=29$ ), and Leaflet ( $n=28$ ), perhaps indicating that participants had a more diverse emotional experience when using OpenLayers compared to Leaflet, although this variation also may be attributed to individual differences across participants in experiencing and describing their moods. The most commonly supplied mood was the neutral 'Okay' ( $n=31$ ; 9.7% of all supplied emotions), followed closely by the negative 'Frustrated' ( $n=30$ ; 9.4%). Other frequently supplied moods across the eight diaries included 'Blank' ( $n=17$ ; 5.3%), 'Confused' ( $n=15$ ; 4.7%), 'Content' ( $n=14$ ; 4.4%), 'Excited' ( $n=14$ ; 4.4%), 'Anxious' ( $n=13$ ; 4.1%), 'Good' ( $n=11$ ; 3.4%), 'Blah' ( $n=10$ ; 3.1%), and 'Aggravated' ( $n=10$ ; 3.1%).



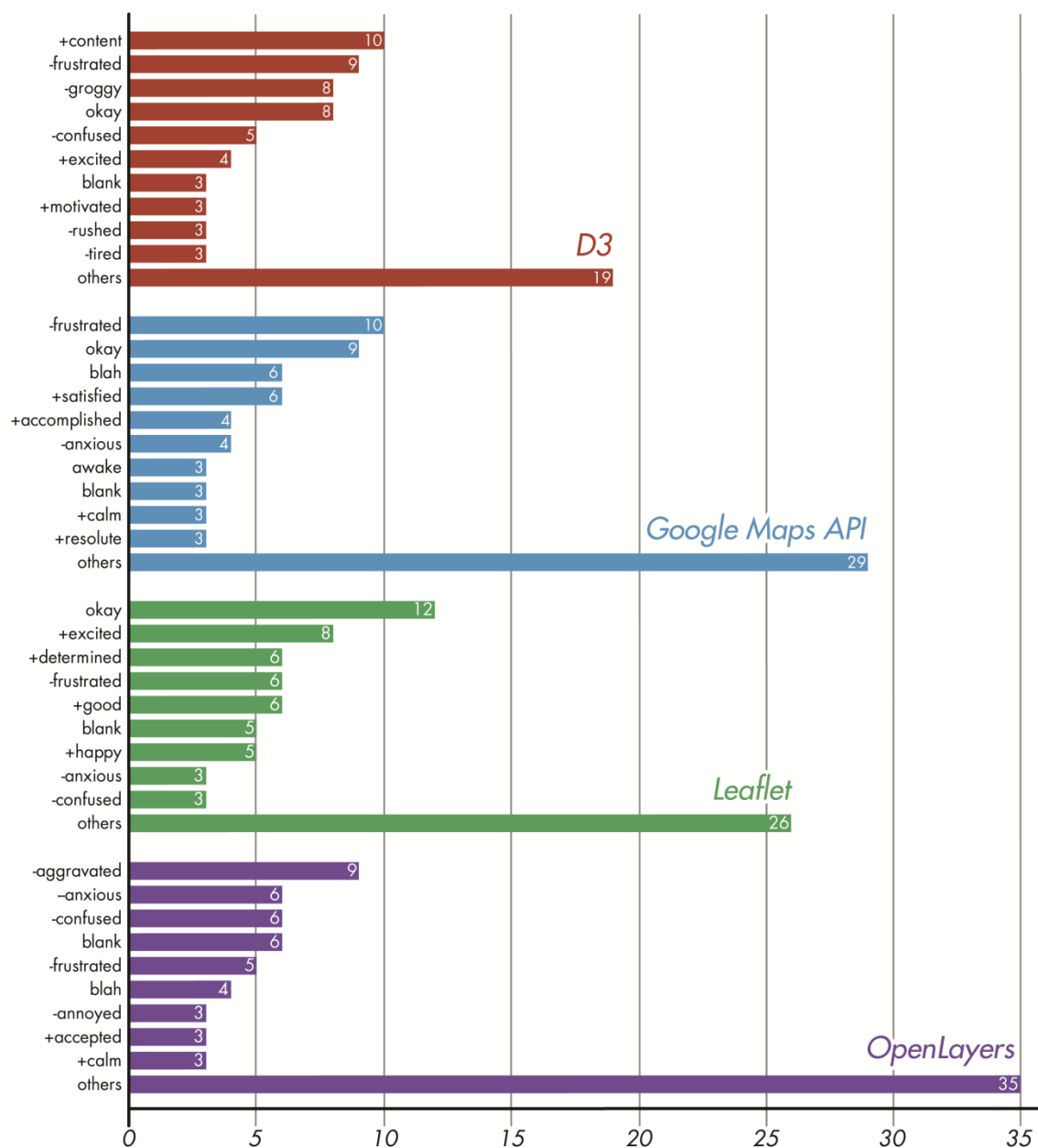
**Table 4.2** The participants' overall emotional experiences during the diary study. Participants used 65 of the provided 125 moods across the eight diaries. A single mood was supplied for each diary entry, totaling 320 moods across the diary study (8 diaries by 40 work hours).



### A. Valence of Moods by Technology



### B. Most Common Moods by Technology



**Figure 4.3** The participants' emotional experience with each of the four candidate technologies: (a) valence of emotional descriptions by technology, organized by negative, neutral, and positive moods (percentages out of 80 for individual technologies)

and 320 for the overall summary); **(b)** all moods provided at least three separate times for each technology (maximum frequency of 80).

Figure 4.3 breaks down the participants' emotional experiences according to the four candidate technologies. Figure 4.3a compares the overall valence of supplied moods across technologies. Across all eight diaries, participants were balanced nearly perfectly in their valence, supplying 125 negative moods (39.1%), 74 neutral moods (23.1%), and 121 positive moods (38.8%). In comparison to the overall average, the valence was more positive with both the Google Maps API and Leaflet. The pair of participants working with the Google Maps API supplied 26 negative moods (32.5%), 23 neutral moods (28.8%), and 31 positive moods (38.8%), while the pair of participants working with Leaflet supplied only 23 negative moods (28.8%), 19 neutral moods (23.8%), and 38 positive moods (47.5%). This means that, when working with Leaflet, participants were in a positive emotional state nearly half of the 40-hour work sessions, while in a negative emotional state just over one-quarter of the session. Therefore, participant experiences with Leaflet were slightly more positive (8.7%) compared to participant experiences with the Google Maps API, despite the Google Maps API resulting in a slightly greater number ( $n=2$ ) of implemented requirements.

In contrast, the valence of supplied moods was more negative with D3 and OpenLayers compared to the overall average. The emotional experience with D3 was only slightly more negative than average, with the pair of participants supplying 33 negative moods (41.3%), 17 neutral moods (21.2%), and 30 positive moods (37.5%). However, the emotional experience with OpenLayers was considerably more negative in comparison to the overall average, as well as in comparison to any of the other three evaluated web mapping technologies. The pair of participants working with OpenLayers supplied 43 negative emotions (53.8%), 15 neutral emotions (18.8%), and only 22 positive emotions (27.5%). Thus, the experience with OpenLayers was the opposite of Leaflet, with participants working in a negative emotional state over half of the 40-hour work session and working in a positive state just over one-quarter of the time. Inspection of the most commonly supplied moods by technology provides further evidence of this emotional disconnect between OpenLayers and the other technologies (Figure 4.3b), as no positive mood was

supplied more than three times by participants using OpenLayers, whereas participants using the other technologies supplied an even mixture of negative, neutral, and positive emotions.

The above summaries of implemented functionality and emotional experience are specific to the four candidate technologies included in the diary study. By analyzing the individual diary entries themselves, we were able to expose broader characteristics of web mapping technologies—and the overall web mapping process—that helped to explain the variation in implemented functionality and emotional experience. We first coded the diary entries according to the technical and practical considerations surveyed in the needs assessment (Chapter 3). Across the 320 diary entries, 79 discussed available tutorials or examples (24.7%), 40 discussed code documentation (12.5%), 1 discussed browser compatibility issues (0.3%), and 1 discussed staffed support (0.3%). There was no discussion in the diary entries of the other technical or practical considerations.

The discussion on tutorials/examples versus code documentation revealed confusion among the participants over the best approach for getting started with their assigned technology. One participant working with Leaflet noted, “it's difficult right away to figure out what I need to read first, what is most important, and where to begin.” The many example maps using OpenLayers was overwhelming and did little to offer an obvious point of entry. The balance in discussion between tutorials/examples and code documentation indicated opposing strategies for getting started, with six diaries starting with code examples and two starting with a multi-hour review of the code documentation. Interestingly, the participants starting with code examples ultimately second-guessed this approach. A participant working with D3 stated, “I took one of the examples and decided to manipulate it to meet my needs...This seemed like a good idea until I realized I have no idea what the code does, which led to me spending inordinate amounts of time trying to understand my own code,” and went on to say “I can't decide if starting from scratch would have been more efficient or not.” Furthermore, pulling from various specific examples also led to difficult when getting them to work together and required a thorough understanding of the code to modify accordingly. Thus, while all four candidate technologies had sufficient code examples, most of

these examples were targeted toward more experienced developers and thus may not be appropriate for classroom education. Examples therefore emerged as a potentially common misconception in learning the web mapping process using respective mapping libraries that, while offering insight and guidance, often pulled student development and learning in undesired directions. Comments in the diary study instead suggested that beginning students would benefit from an initial, condensed overview of code documentation combined with simplified code examples to improve their competency and encourage more active learning. Thus such findings contribute directly to an effective sequencing of learning modules.

Notably, there was one example of ‘staffed’ support regarding D3, with the participant writing “After my frustration earlier in the day, venting on Twitter got the attention of Mike Bostock...I spent the hour uploading data for him to look at and troubleshoot,” with the conversation resulting in the participant having a much deeper understanding of how to implement D3. While anecdotal, this interaction between a student and the creator of D3 was indicative of the supportive atmosphere and availability of open web mapping community. This anecdote also provided an example of how online collaboration—through social media such as Twitter and forums such as Google Groups and Stack Overflow—has become part of the support process for open source web mapping technologies. As such, the developers of D3, Leaflet, and OpenLayers explicitly state that users are free to contact them with questions through these collaborative outlets. Future educational strategies may emphasize as a threshold concept the importance of learning how to utilize the open web community in this way.

Discussion around the best way to get started with a technology ultimately led to comments about the optimal web mapping workflow, with consensus to first format and load the dataset, then implement the representation requirements (e.g., symbolizing the dataset), and finally implement the interaction requirements (i.e., to build the user interface). While the scenario requirements focused on representation and interaction design, participants spent the largest portion of their time formatting and loading the energy dataset. Across the 320 diary entries, 125 (39.1%) primarily referenced the data, 122 (38.1%) the

representation requirements, and 73 (22.8%) the interaction requirements. Such a work distribution in the diary sessions signaled an importance of teaching to the *data->representation->interaction* workflow, and also revealed a blind spot in many of the existing code examples that limited their utility. A participant working with OpenLayers stated “All examples use online data sources for the basemap, making it very difficult to figure out how to use my own data...documentation and examples for importing data is very weak,” while a participant working with Leaflet stated “There just doesn't seem to be much information telling me how to get to the point of taking data out of [the dataset].” Therefore, it is critical to teach data formats and loading first—even if they are not the primary goals of the course—before teaching representation and interaction techniques. This key insight speaks volumes to our desired understanding of proper sequencing of the learning process across this emergent workflow.

Interestingly, one of the key advantages of the Google Maps API, leading to the high level of implemented functionality, was the documentation for loading data through Google Fusion Tables, with one participant stating he or she “was able to get started much more quickly on the representation, given how the [Google Maps] API makes loading and styling tiles straightforward” and the second stating he or she “plugged the data into Fusion Tables, from which point it was pretty easy to merge with the KML and get it to render as a layer.” However, the initial decision to use Fusion Tables for the dataset led to constraints in both representation and interaction design later in both diary sessions using the Google Maps API. This may suggest the use of Fusion Tables in the closed Google Maps API proved to lend toward misconceptions of an ideal way to store and access data in the running of the application. A participant also noted a problem with representation in the Google Maps API, stating “Each map is limited to five layers, and each layer is limited to five styles...this means I can only have five classes (including a null value class) and won't be able to allow users to reclassify the map,” while a second noted a problem with interaction, stating “tooltips fusion library has its own constraints...unsure if Fusion Tables are cool or not.” Ultimately, these initial data decision to use the Fusion Tables led to a large amount of code refactoring midway through the diary sessions to better support the representation and

interaction requirements, and ultimately to some of the dissatisfaction with the proprietary Google Maps API in comparison with the fully open and more flexible Leaflet (Figure 4.3a).

The importance of initial data formatting and loading on subsequent representation and interaction development was not specific to the Google Maps API. One participant working with Leaflet reformatted the dataset numerous times throughout the diary session, stating “working on building stats and all of a sudden I realized my data is actually completely incorrect...time to rebuild AGAIN!” This participant was unable to start on the representation and interaction requirements until the second half of the diary study due to issues with data formatting, but was able to implement the requirements quickly once the data was formatted and loaded correctly. In one extreme case, a participant working with OpenLayers struggled to implement any of the requirements due to issues with loading the dataset until finding a set of symbolization examples that included code for data loading, which in turn led to a period of rapid development of the representation requirements. However, this participant then was unable to extend these examples to implement the interaction requirements, leading to a plateau in development over the final third of the work period. Thus, proper formatting of geospatial data remains tantamount to web mapping, and should be processed not just to optimize the initial loading, but with the sequence of the workflow in mind to enable the representation and interaction design that follows.

Participant issues with data formatting and loading reflected broader issues with transitioning to the Open Web Platform. While the process focused on JavaScript web mapping libraries, the diaries revealed a multiplicity of competencies required to develop on the Open Web Platform and contributing to our emerging understanding of the scope of the workflow. Related to the above discussion on data processing, participants struggled to manipulate data objects through the DOM. In total, 61 (19.1%) entries noted problems with manipulating GeoJSON or TopoJSON, the geospatial variants of JavaScript Object Notation (JSON), a text-based data-interchange format. One participant working with OpenLayers stated “accessing feature properties of a JSON has proven to be difficult,” and a second participant working with Leaflet stated “I’m really understanding how to dig into the GeoJSON now, which I think is one of the

most important parts of this entire exercise.” In addition to issues related to JSON and the DOM, 19 (5.9%) of the entries noted problems with SVG, 15 (4.7%) noted problems with HTML, 7 (2.1%) noted problems with CSS, and 7 (2.1%) noted problems with XML. Thus, participants spent approximately one third of their time (109 of 320 entries; 34.1%) working on development tasks unrelated to the writing of JavaScript code.

Interestingly, new versions were released during the diary study for two of the four candidate technologies: D3 and Leaflet. As discussed in Chapter 3, maintenance/stability was identified as the most important practical consideration from the needs assessment survey. In both cases, the update was positively received by participants and aided development, rather than hindering it. One participant working with D3 stated “A new version of D3 was released this morning which allows a thresholded color ramp...it means I can use ColorBrewer to pick out colors, and then feed exact hex codes into D3 for my classification!” Similarly, a participant working with Leaflet stated “With the new version of Leaflet coming out, there's some new additions that make some of the work [simpler] so I decided to recode the map in the updated version...the language took me 3 lines to get my basemap and GeoJSON in and styled...super concise!” While only two examples, the ease participants had with integrating new releases of the D3 and Leaflet core libraries pointed to the broader trend of improved stability in open web mapping technology.

Finally, the diary entries suggested two limitations of the diary method design that could be improved in subsequent applications of the process. The most common complaint was about the rigid, hour-long structure imposed for each diary entry. Articulating this issue well, one participant working with D3 stated “[I] had a hard time with the format of this project, mostly because I'm not very good at staying focused but also because I like to work in small chunks of time,” and went on to say “If I have only half an hour to do some work, I feel like it's not worth it because, on average, I need about an hour and 15 minutes for each hour of this project.” While the 40-hour structure promoted consistency across the diary sessions—allowing for a more reliable comparison across technologies—it would be more practical in a

professional setting to instruct participants to log a diary only after making a significant breakthrough or running up against a difficult challenge. Several participants also felt the constraint of using only a single web mapping technology was counterproductive. One participant working with Leaflet stated, “I feel like if I want to bring in something like a graph or chart to complement this I would use D3...Leaflet doesn't support that and isn't intended to.” A separate participant working with OpenLayers noted both of the aforementioned limitations of the diary study, stating “Two of the constraints of the experiment that must be considered in terms of how they impact the practice of development: the un-interrupted forty hours and mutual exclusivity of technologies.” Because most firms are likely to combine web mapping technologies based on their relative affordances and limitations—rather than relying on only one ‘winner’ technology—allowing the mixture of technologies in the diary study would have better mimicked real-world development.

## **4.2. Exit survey**

Each of the research participants completed an in-depth exit survey directly following the 40-hour diary session. The exit survey aimed to solicit: (1) participant reflections about their experience using their assigned web mapping technology to attain the scenario requirements, (2) nuances and quirks about the the web mapping technology itself, (3) advice for future students who will be learning the given web mapping technology, and (4) reflections on the diary study activity itself. The survey consisted of both open-ended questions and Likert-scale questions (Appendix C). Overall, results from the exit survey validated findings from the diary study and clarified the emerging prototypical web mapping workflow.

Participants provided advice for teaching web mapping to future students and focused primarily on the technical considerations, rather than principles of conventional cartographic design. Overwhelmingly, participants reiterated their comments from the hourly diary entries that emphasized the importance of first learning the JavaScript programming language and the web standards of HTML and CSS in the context of in-browser development. They stressed that the necessary coding background went beyond



basic programming skills, and included thinking through effective data structure manipulation, pseudo-coding before development, and using the best practices of organizing computation into meaningful functions and methods. Although a basic concept of functional programming, one participant signaled this as a threshold concept when recommending future students work on “building all of your code into functions, rather than just one big function...it will make changing code and understanding how all of it works together more simple.” Though not programming languages per se, learning HTML and CSS was a challenge, in particular because these standards needed to be learned concurrently and used with JavaScript. One participant noted that “It’s easy to spend an hour trying to get a div to center for the first time.” Another participant wrote, “I think the hardest thing is to learn multiple languages at one time. Things get confusing [ ...], and it is hard to understand how everything interacts—how HTML utilizes CSS and JavaScript and how the mapping technology is incorporated.” This was independently affirmed by another participant who asserted, “The hardest thing for students to learn will be the concept of putting JavaScript, HTML & CSS, and then more JavaScript libraries all together to work with each other.” A few comments additionally noted confusion as to when they should seek a solution in functionality provided by the mapping library, or when they should rely on pure JavaScript instead. These comments point toward the challenge of learning how to integrate a given JavaScript mapping library—which itself may reference and make use of its own CSS style rules—with their own custom code or additional JavaScript libraries such as jQuery.<sup>24</sup>

As in the diary study, participants stressed the proper formatting and manipulation of data in the web mapping development workflow as a key threshold concept. Participants were challenged both technically and conceptually in terms of scripting the data loading process and then binding the loaded data to HTML and SVG elements for representation. One participant underscored this point:

The hardest thing for me was learning how to take stuff from a totally different data source (GeoJSON) and incorporate it into a language. It seemed like I was connecting

---

<sup>24</sup> The jQuery JavaScript library works as an abstraction library providing simple methods for DOM traversal and access, and event handling.

two things that weren't meant to be used together (which is ironic because *.json* is a JavaScript file type). What made the conceptualization even more difficult was how I got that data to be represented in the HTML of my page, and how it could possibly change.

Participants also were confused as to whether the data, which were provided to them in a CSV format, are best converted to and stored within another format such as GeoJSON before loading into the script, or whether it should be done at runtime.<sup>25</sup> This experience varied by web mapping technology, as Leaflet and OpenLayers provided native support for parsing GeoJSON, D3 provided support for loading both GeoJSON and CSV file formats, and Google Maps API converted the CSV file into its own web service Fusion Table, which the JavaScript would then access with an HTTP request. Whatever method used for data loading, however, all participants stated in the exit survey that they had spent significant time conceptualizing how these data were deserialized (i.e., translated) into either JavaScript objects or HTML/SVG elements accessible through the DOM. This was particularly the case with OpenLayers, in which case one participant wrote,

The absolute worst parts of this project were getting access to the data and displaying it in the map. While parts of the task list were frustrating at times, it was expected. I never thought, though, that it would take me so long to get a basemap in the first place.

All participants came to recognize proper data structuring and binding as an important place to start when teaching and learning web mapping in the future. One recommended providing specific scripts for parsing such data and explicit instruction to complement these scripts. Another participant said, “Figure out how your data can be accessed...then you will understand how you can use Leaflet’s capabilities to interact with it.” When asked about a recommended order for web map development, participants largely agreed that getting the data properly formatted and loaded into the map was a crucial first step, followed by representing those data on the map, even as simple point markers as a first step. This advice reflects the emerging understanding of the *data->representation->interaction* process of the prototypical web mapping workflow.

---

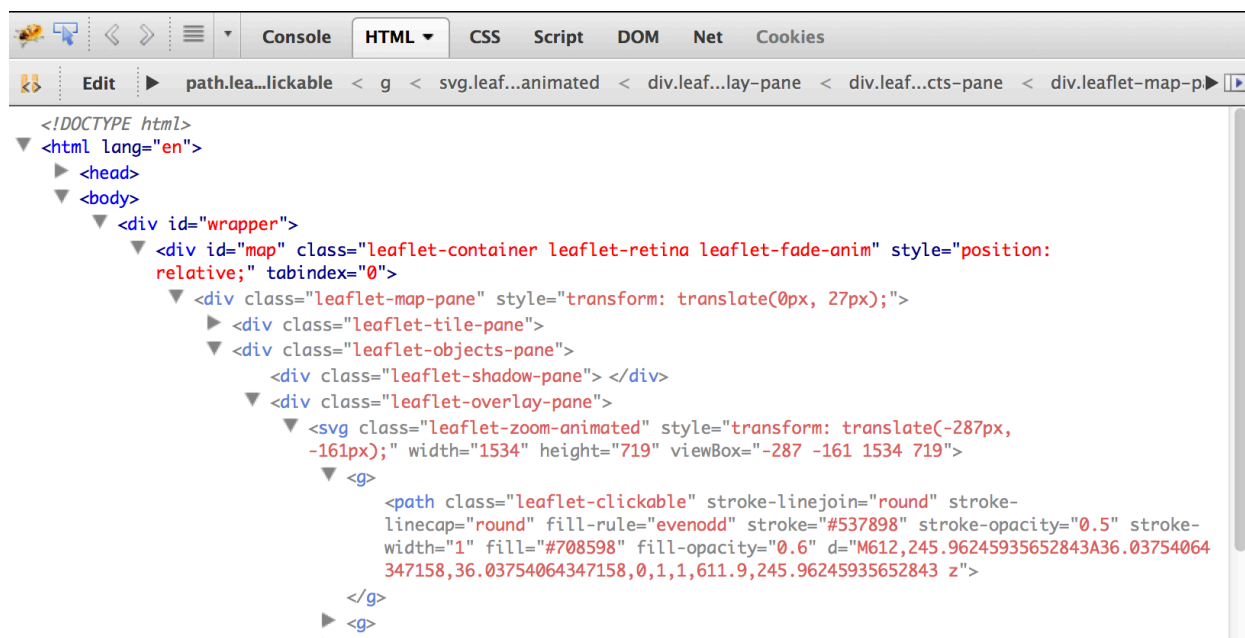
<sup>25</sup> Such confusion may have stemmed from misconceptions formed in the prior Flash-based model of hard-coding data sets into the scripts themselves.

While participants all accentuated that understanding JavaScript, HTML, CSS, and the JSON format is fundamental for web mapping, they also described the importance of learning how to implement them using plain-text code editors and render output within the browser as the development environment. While preceding web mapping processes often built a web map in some form of integrated development environment (IDE) separate than the delivery mechanism/medium, today web maps are in part authored in the same environment they are intended to be viewed: the web browser. While in a simple sense, the browser is used to view the results of coding and rendered data, the browser is also the tool used to debug programs, inspect the DOM, and even develop.<sup>26</sup> Although one participant utilized the IDE Dreamweaver, they later recanted the benefit of that approach, noting that, “First time users may want to stay with plain text files for simplicity.” One emergent finding in this vein was the importance of using in-browser debugging and development tools. These were important to the development process in a number of ways. First, participants noted that debugging JavaScript code overall was challenging. One participant in particular wrote that,

For me the hardest thing to learn (and I still struggle with it a lot) is debugging. There are many techniques, such as using Firebug, throwing errors in your code, or even checking Apache error logs, and it’s hard to determine the best method for any given situation. It’s also hard to figure out what to do differently when you are getting no errors.

---

<sup>26</sup> A typical development process may involve (1) editing and saving a script, and (2) refreshing the browser to see results. However, the feedback/response can be shortened by directly editing code in the browser and instantaneously view the result. Code solutions can then be transferred to the script file and saved.



**Figure 4.4** Screenshot of DOM hierarchy of a Leaflet map accessed through using the Firebug web development tool in Firefox.

Participants found the employment of Firebug—a web development plug-in tool installed within the web browser Firefox—useful for debugging JavaScript code errors.<sup>27</sup> In particular, students learned the importance of logging output statements and errors to the ‘console.’ When asked which tips or tricks they would recommend to other students, one participant humorously wrote *console.log()*—the method for logging output to the console—three times, emphasizing the importance of use this technique within JavaScript development.<sup>28</sup> However, the importance of using browser development tools extended beyond debugging. Such tools were critical for accessing visual representations of elements within the DOM hierarchy (Figure 4.4). This was important for examining the rendered output of the map being developed, but also provided a powerful way to dig into other examples to understand how the JavaScript code and complementing mapping library rendered output within the browser. Logging a data object to the console also proved useful for understanding the data structure itself, such as how to access key values using dot

<sup>27</sup> Recent versions of modern web browsers such as Chrome or Firefox are released with increasingly sophisticated web development tools built into the application itself.

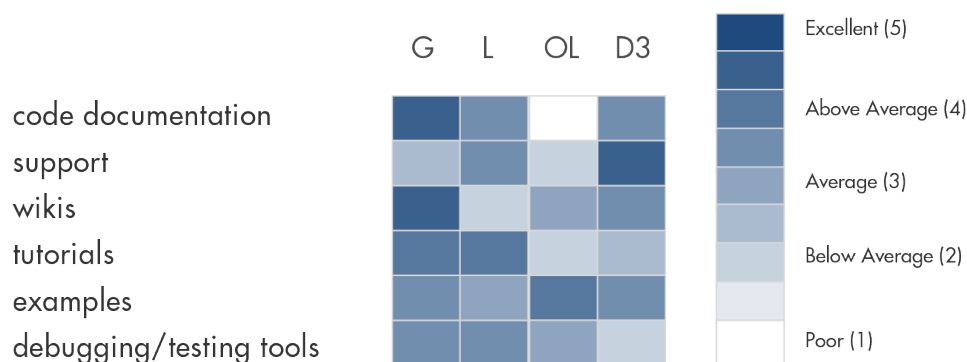
<sup>28</sup> While IDEs such as Flash provided built-in debugging and testing tools, development on the Open Web Platform departs from this and largely makes use of in-browser web development tools instead. However, additional libraries can be loaded for further testing and debugging. For example, OpenLayers provides such a script, creating a “Firebug Lite console”: <http://openlayers.org/dev/examples/debug.html>.

notation. While fairly obvious to more experienced developers, these findings emerged outside of the anticipated scope of representation and interaction design and also signaled their importance as threshold concepts to explicitly teach students.

Within the exit survey, participants also discussed the importance of utilizing online support resources and examples for finding answers to both general web development questions and solutions to web mapping-specific problems. In part because of the constraint of the experiment design itself (requiring independent work), participants turned to the web itself to learn the technologies and puzzle through various technical problems they encountered. Again, reiterating the findings from the diary study, they primarily signaled the importance of three forms of online help: (1) online forums, (2) social media (the open variation of staffed support), and (3) documentation. Participants found a wealth of information within the online discussion forum Google Groups and the online Q&A website Stack Overflow, particularly for Google Maps API and D3 (Table 4.3). Stack Overflow in particular was useful for the more general web development questions involving JavaScript, HTML, and CSS. One participant found it more useful than other online forums when advising,

Watch out for online forums. They can be the most useful in times of need, but are filled with people who enjoy talking about code that nobody knows [how] to show off. Try stackoverflow.com, which has a system built into it for best answer based on user votes. This will get you more concise answers than a random array of different responses.

While the web mapping community's presence appeared to be growing on these online forums, participants noted that the quality of these contributions was variable, as was the relevance to specific problems encountered within the development process. One participant noted spending too much time searching for a perfect solution within these resources, rather than tinkering through a solution. Examples initially offered the promise of a quick, easy solution, but participants found modifying them to suit their needs more difficult. Social media also served as a useful medium for attaining support.



**Table 4.3** The participants' rating of the quality of learning materials available by technology (n=5).

Finally, all participants stated in the exit survey that they utilized the help and documentation of the web mapping technology websites, including tutorials, examples, and API reference documentation. Importantly, the role of distributed version control code repositories (e.g., Github) that host mapping library code and are linked from product websites emerged as key resources and as an important component of our contemporary development environment, both in terms of accessing helpful material and eventually in terms of student engagement with their own code repositories. Altogether, these resources proved essential for participants, although as discussed above in the diary study, they rarely provided a 'silver bullet' solution for development of the more comprehensive implementation of scenario requirements. As one participant lamented, he or she spend two to three times more time puzzling through examples and learning the specifics of the web standards than on the actual technological implementation. Instead, this person suggested that immersion in the mapping project would have been a faster way to learn, rather than attempting to garner the missing background experience first.

#### **4.3. Conclusions: reflections on the research process and the establishment a prototypical web mapping workflow within the full stack to serve Web Cartography education**

*Research Question #2: What are the important characteristics of web maps that should inform the selection of web mapping technologies?*

Despite participants' emphasis on interactivity within the needs assessment survey, participants in the diary study accomplished nearly two more representation requirements than interaction requirements per diary session, suggesting the primacy of representation over interaction in the web mapping workflow when using these current web mapping technologies. Only the choropleth map and dynamic classification were implemented in all eight diary sessions, with the common interaction operators pan, zoom, and retrieve implemented in all but one diary session. The interaction operators calculate, filter, and search were not implementing in any diary session, further identifying gaps between practice and theory in Web Cartography. Finally, the diary study identified nuanced differences in supported functionality across the four candidate technologies, suggesting their relative affordances and limitations. Such insight can be used to derive preliminary recommendations for pairing web map requirements with potentially viable technology solutions, allowing for design to precede development (rather allowing technology to constrain design). Overall, the insight generated through the process reminded us that good web map design is more important than using novel tools—open or proprietary—and that university programs, government agencies, and cartography firms actively should combat path dependencies on one technology that lead to its use beyond its functional utility.

*Research Question #3: How should web mapping be taught in higher education?*

The diary study and exit survey yielded multiple insights that shed light on the emerging web mapping workflow and thereby inform the way web mapping should be taught in a university setting. Beyond identifying those technologies capable of supporting a wide array of representation and interaction needs, the collection of participant moods across the diary sessions highlighted the role of emotional experience when learning a new technology. The valence of moods overall was balanced nearly perfectly across all 320 diary entries, but varied considerably by technology. While participants accomplished slightly more scenario requirements with the closed Google Maps API, they had a slightly more positive emotional experience using the open Leaflet library, in part due to the larger amount of code refactoring necessary when working around a closed API. When choosing a web mapping technology, therefore, it is equally

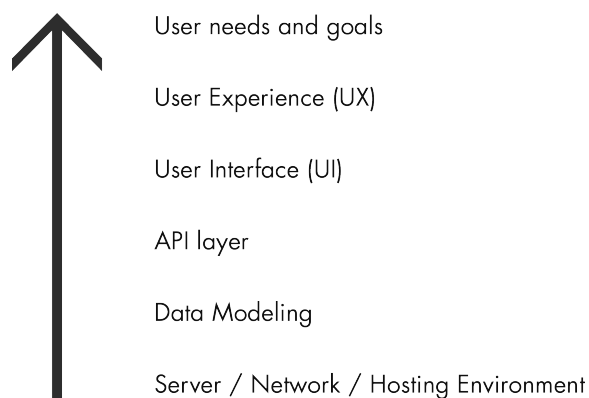
important to understand the emotional experience when applying the technology (i.e., how the student feels while doing it) as the functionality supported by technology (i.e., what the student can do with it).

Discussion in the diary entries suggested confusion about how best to get started initially, with six diary sessions beginning with manipulation of code examples and two with a multi-hour review of documentation. Participants in the needs assessment identified good tutorials/examples and code documentation as practical considerations of near equal importance. However, participants in the diary study who started with code examples ultimately came to question this decision due to their complexity, with comments suggesting that an initial, condensed overview of code documentation combined with simplified code examples would be a better approach to reducing the initial learning curve. Having simple ‘beginner’ exercises early in the learning process also would provide students with early ‘wins’, improving their emotional experience and promoting active learning.

The in-depth development of a web map using the scenario-based diary study allowed us to clarify the contours of a prototypical web mapping workflow. Due to the increased convergence of web mapping with Open Web Standards and web development in general, the range and sequence of our emerging workflow is also informed by the notion of the *full stack*, which refers to the suite of technologies and required skills needed to successfully transform data into a deployable product on the web (Figure 4.5). This stack often is conceived as a continuum from ‘back end’ server-side technologies to ‘front end’ client-side technologies through to the user experience itself designed to meet user goals. Working from the bottom to top of Figure 4.5, ‘back end’ concerns involve proper server configuration and the establishment of effective data models, either stored within relational and spatial databases, or perhaps more simply as ‘flat’, text-based files (GeoJSON, CSV) for smaller web mapping projects. The API layer requires strong programming skills and an ability to integrate the data model with the business logic of the application itself, as well as how code libraries and frameworks interface with one another. The User Interface layer may primarily make use of HTML, CSS, and JavaScript to implement good visual design

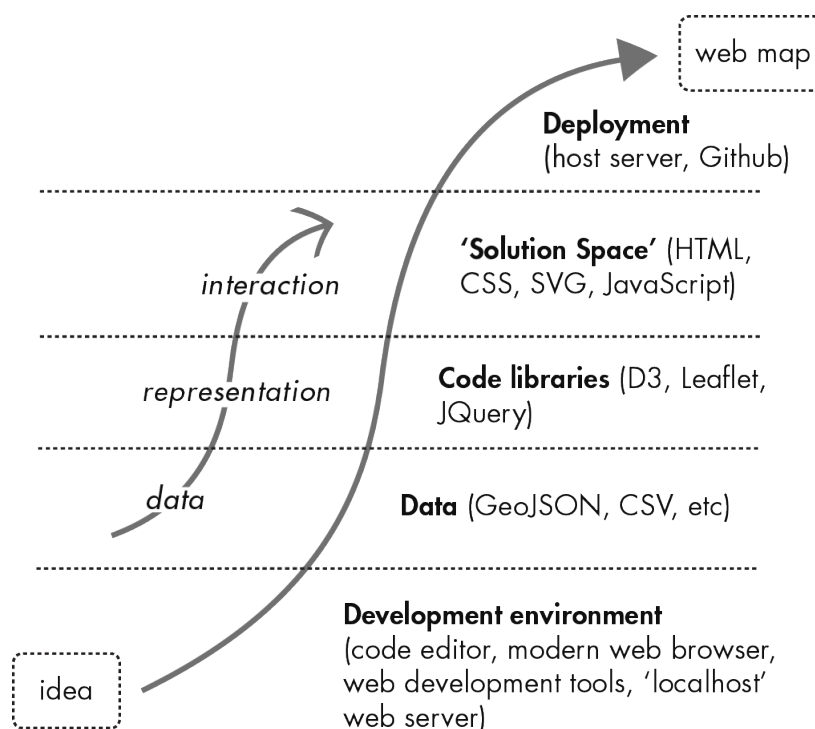


and effective interaction. The top layers of the stack attend to the goals of the user and a refinement of the experience of using the application.



**Figure 4.5** Typical conception of the layers of the ‘full stack’ development process.

Teaching toward a professional workflow in part means sequencing educational modules aiming to mimic this process. A web mapping workflow informed by both the full stack and our diary study helps identify the individual tools and technologies, how they relate, what skills and background knowledge are needed to use them, and the general sequence of steps taken within the process of making a web map (Figure 4.6). The diary studies revealed a complete set of competencies required for web mapping that fit within this web mapping workflow. Crucially, these are not isolated skills or tools, but rather they work both in the sequence of a development process and in concert with one another. Hence, this research revealed the importance of teaching across the *data->representation->interaction* workflow, with an emphasis on data formats and loading that enable future representation and interaction development. This involves teaching across the Open Web Platform, with a sequence of modules on manipulating JSON in the DOM and the HTML, CSS, and SVG specifications.



**Figure 4.6** The prototypical web mapping workflow.

Figure 4.6 offers the conceptual scaffolding for both the scope and sequence of web mapping today, as well as a mechanism for identifying key threshold concepts to emphasize in learning modules. A typical web mapping process or exercise will begin by ensuring students have a working understanding of their development environment. This involves proper structuring of directories and files within the computing system and the capacity to test rendered code output—written in a helpful code editor (e.g., Sublime Text)—within a browser supported by a server technology to facilitate AJAX requests. These are all important aspects of the scope of modern web map development unknown within the previous Flash-based development environment. Students then need to understand how to clean their data and encode within a useful format such as the GeoJSON specification. Our diary study revealed that common misconceptions involve data manipulation and loading into the script, so these aspects should be particularly emphasized within learning modules. With the help of a web mapping library (e.g., Leaflet, D3), students can then render data on a map. Our research process suggests that getting data into the map and visualizing it quickly is an important threshold concept that empowers students to continue with map

development (i.e., gains in meeting representation and interaction requirements can be quickly made once this critical step is successfully completed). Other key threshold concepts to emphasize at this point in the development process include using in-browser development tools for understanding the DOM structure, as well as debugging strategies (e.g., the use of `console.log()` to test code output). Further customization of the interface then ensues using a combination of custom code solutions as well as making use of a mapping library's additional functionality. Within this part of the learning scope, a thorough understanding of the relationship between the structural (e.g., HTML), stylistic (e.g., CSS), and behavioral (e.g., JavaScript) elements of the application is required.<sup>29</sup> Instruction should focus on identifying misconceptions within this 'solution space' and ensure that students understand each of these both individually and how they work together to create the modern web application experience. The visual design of the map is further improved through attendance to CSS style rules. Finally, maps are publicized on a remote server and code can be stored and shared via a distributed version control repository such as those hosted by Github.

As a result of this research, we developed lab tutorials for making web maps using Leaflet in Fall 2012 for our advanced course on Interactive Cartography and Geovisualization, and included one advanced lab introducing D3 after students learned JavaScript and Leaflet.<sup>30</sup> We later submitted these tutorials for publication within *Cartographic Perspectives*, the intent being to share our acquired knowledge with the broader community in an open format. The direction within the tutorials followed from the prototypical web mapping workflow and walked students through the sequence of (1) establishing their development environment, (2) formatting data, (3) using either Leaflet or D3 to first represent the data using an appropriate cartographic symbolization, and (4) adding additional interactivity with customized code

---

<sup>29</sup> A beginner's introduction to these core competencies may occur earlier in the learning sequence, perhaps before real data is acquired and loaded into a web application. This admission points to the need to revisit skills throughout the learning sequence.

<sup>30</sup> The lab instructions and source code resulting from this project are available for download on Github: <http://www.github.com/eroth/g575-2013>.

solutions. Through a linear narrative, the lab tutorials addressed both the scope of web mapping technologies and the proper sequence of steps students should follow.

Completion of the lab exercises were nonetheless challenging due to the range of concepts and skills to simultaneously learn and implement, particularly debugging techniques within the browser and developing greater student proficiency in incrementally building and testing code. As indicated within the diary study, a greater emphasis on working with smaller examples early on that focused on various aspects of the development workflow would have been helpful. Additionally, while students were able to successfully meet the minimal requirements for lab assignments, extending these applications to provide for greater interaction requirements was limited. Examples and linear tutorials were sufficient for guiding students to meet representational requirements and more basic interaction solutions (i.e., those natively supported by the web mapping library). However, another strategy is needed for the learning and teaching of more advanced, custom interaction solutions (e.g., calculate, filter, search). Chapter 5 proposes this strategy.

*Research Question #4: How can we better cope with continued evolution in web mapping technologies?*

The three-stage process described in Chapters 3 and 4 was successful in identifying viable web mapping technologies for the UW Cartography program. Importantly, the process aligned with the pilot study approach to exploring emerging technologies already completed by several of the participants in the needs assessment survey. The process proceeded in a discount, convergent manner, which should allay concern over technological experimentation due to the resource investment and ultimately act to combat path dependencies on one technology. The case study additionally revealed several ways to improve the diary stage of the process, including the composition of entries only following critical incidents—rather than every hour for 40 hours—and the ability to flexibly combine technologies using one as a base—rather than artificially restrict development to a single technology. We also put several structures in place to improve the scientific reliability of the generated insights, such as employing a pair of coders in the competitive analysis and having a fifth participant complete a diary session with all candidate

technologies; such structures may be removed when applying the process in a non-research context. The process suggested additional mechanisms for coping with the continued evolution in web mapping technology centered upon the active and growing open source web mapping community. From the academic world, this includes translating the development features of technologies into the *lingua franca* of cartographic design as well as the sharing of both conceptual and technical learning materials as open educational resources. From the development world, this includes crowd sourcing the organization and synthesis of web mapping technologies across the community to support young developers as they forge careers in Web Cartography.

## **Chapter Five: A Web Mapping Pattern Library**

### **Design Patterns for Web Cartography Education**

---

#### **Overview**

This chapter introduces the concept and practice of design patterns to Web Cartography education and practice. Software Engineering (SE) and Human-Computer Interaction (HCI) both have leveraged design patterns and pattern libraries for software development and interface design. Professional web designers and developers also have employed design patterns to aid novice designers, as well to bridge design and development. While Web Cartography successfully has applied principles and practices from these fields for web mapping, design patterns remain an unexplored solution to many of the challenges in Web Cartography education identified in Chapters 3 and 4. This chapter analyzes design patterns and offers recommendations for creating and maintaining design patterns within a pattern library to serve web mapping education and practice.

#### **5.1 Design patterns and pattern libraries: a solution for web mapping education?**

The cumulative results of Chapters 3 and 4 describe the set of competences and skills captured within the full stack workflow required for contemporary web map design. The analysis in Chapters 3 and 4 additionally identified elements of the scope and sequence of desirable learning, as well as the misconceptions that inhibit learning gains and the threshold concepts that break through such barriers. Together, this knowledge is useful for maintaining effective web mapping education in an era of rapid technological change. However, the initial instruction of the JavaScript-based Geography 575 course in Fall 2013, while overall successful, was not without substantial challenges. Here, we propose design patterns and pattern libraries as a potential solution for further improving web mapping education and practice. This chapter proceeds by critical analyzing existing approaches to design patterns and pattern libraries within a variety of computing fields against the established criteria of web mapping education established through the Chapter 3 and 4 studies.

**Design patterns** capture common solutions to recurring design problems, and were inspired by Christopher Alexander (1977, *et al.* 1979), who worked in the fields of architecture and urban planning.<sup>31</sup> Alexander often shared his solutions involving buildings and towns with others, and eventually he became convinced that recurring problems could be encoded systematically in a structured format to make them more accessible and useful. These design patterns are characterized by the following six tenets:

- (1) patterns emerge as *heuristics*, or ‘rules of thumb’, from known solutions established through practice; in other words, they emerge organically ‘bottom-up’ vs. ‘top-down’ or deduced from ‘first principles’;
- (2) patterns are stated in the form of a ‘rule’ that minimally addresses the relations among the broader context, the forces constituting the problem to be resolved, and the solution (or configuration) that resolves these forces;
- (3) patterns are presented in a specific format (see Table 5.1);
- (4) patterns are generative in that they allow users to develop new variations to a solution, rather than simply replicate a given solution exactly;
- (5) patterns empower novices or civilian users to design and build their own environments (i.e., are not intended solely for experts);
- (6) patterns are organized into a *pattern library* that relates individual patterns to one another, as well as makes explicit how lower-level patterns fit within higher-level ones.

For Alexander, the goal of sharing his patterns was to provide a way for inhabitants to re-engage with design processes and participate in building their own towns, neighborhoods, and buildings, something he believed to be increasingly threatened within contemporary society. The solutions presented within his

---

<sup>31</sup> Borchers (2001) attributes one of the first design patterns to the master builder Francesco d’Giorgio (1439–1501) who documented design solutions, which included a sketch and textual description of the solution.

patterns emerged not from the expertise or ‘first principles’ of his architect and urban planner colleagues, but rather from the processes of trial and error in which people made and remade their built environment to maximize utility and enjoyment.<sup>32</sup> For Alexander, through supporting patterns of events that happened frequently and involved relationships between spatial components, the ‘users’ of these environments eventually came to understand ‘successful’ design solutions. His design patterns sought to capture such tacit knowledge. In this sense, Alexander’s intent can be considered congruent with participatory and user-centered design.

Alexander emphasized that a design pattern principally addressed the relations between three components that form a ‘rule’: (1) a particular *context* that explains the larger set of relations in which the pattern is situated, (2) a system of *forces* that arise in context (thereby creating the *problem* to be solved), and (3) a *solution* (or *configuration*) that resolves or balances those forces. Once the solution is applied, a new context and set of conflicting forces emerge in need of a solution, and builders apply a succeeding (hence relational) pattern to that context (Vora 2009). This process repeats until an environment successfully is built into a livable space. The composite parts of a rule furthermore describe a solution, though “not in a narrow prescriptive way, but in such a way that you can use this solution a million times over, without ever doing it the same way twice” (Alexander *et al.* 1977: 182). Alexander’s design patterns therefore were intended to be *generative* of unique configurations, given specific contexts and forces.

Though not obvious solutions, design patterns are neither novel nor extraordinary. For example, Alexander *et al.* (1979) wrote a design pattern named ‘Beer Hall.’ The *context* of the pattern occurred in a neighborhood where there emerged a “special need for something larger and more raucous than a street café” (445). The *problem* was then a question of where people could go to sing, shout, and “let go of their sorrows” without disturbing the rest of the neighborhood. Among the *forces* in need of resolution were

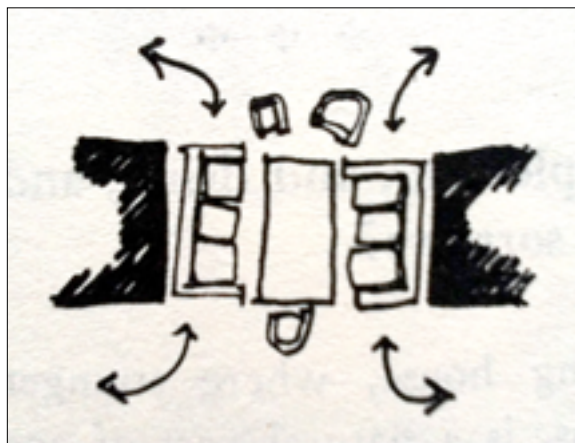
---

<sup>32</sup> The concept of a bottom-up, organic approach is a marked departure from the top-down, structuralist understanding of web mapping presented in Chapters 3 and 4. While generating great insight, the latter approach potentially creates blind spots that arise in the learning and practice of web mapping.



that bars often discourage social interaction and “become nothing more than anchors for the lonely.” Alexander stated the *solution* to resolve these forces in terms of the spatial layout of the English pub, where the entrance, bar, bathrooms, dance floor, fireplace, and dart games were all *configured* around the perimeter so they “generate a continual criss-crossing” of movement (446). Included in the proposed solution was the formation of seats as open alcoves (Figure 5.1) that helped to “sustain the life of the group and lets people come in and out freely,” as well as encouraging invitations to sit with strangers (466). Alexander prescribed a didactic structure with which design patterns were presented (Table 5.1).

Design patterns do not exist in isolation, however. Alexander *et al.* (1977) combined 253 patterns into what they called a “pattern language,” which described the linkages between patterns. A *pattern library* or (*pattern collection*) more commonly refers to a compilation of these design patterns organized to describe the relationships among all its elements (Gabriel 1996; Borchers 2001). More than merely a catalog of independent design patterns, a successful pattern library helps to relate parts to the whole, increases the accessibility of individual solutions, and “captures ordinary design wisdom in a practical and learnable way” (Tidwell 1999: 1).



**Figure 5.1** Alexander *et al.*'s (1979) diagram of “The open alcove—supports the fluidity of the scene” (446).

#	Pattern Component	Description	Example
1	the <b>name</b> of the pattern	conveys the idea of the pattern in one or a few words, to make it easy to remember and refer to when thinking about or discussing design solutions.	BEER HALL
2	a <b>ranking</b> of its validity	indicates the degree of confidence that the authors had in the pattern, ranging from a mere example to a truly timeless solution	low confidence
3	a picture as an <b>example</b> of its application	an image representing a good example of pattern solution	[ photograph image of beer hall ]
4	the <b>context</b> in which it is to be used	explains which larger-scale patterns this specific pattern helps to implement (e.g., a pattern at the street level may reference a higher level pattern at the neighborhood level)	neighborhood social space where people wish to celebrate
5	a short <b>problem statement</b>	summarizes the general situation that the pattern addresses	typical bars tend toward loneliness and inhibit socializing
6	a more detailed <b>problem description</b> with empirical background	gives empirical background information on the pattern and states problems in terms of the competing forces and discusses existing solutions	Discussion of social movement through the space. What should be the focus of attention? How will people mingle?
7	the central <b>solution</b> of the pattern	statement providing general solution	layout bar in style of English pub
8	a <b>diagram</b> illustrating the solution	visualization of the solution	[ diagram of room layout and seating alcove ]
9	<b>references</b> to similar patterns	relates the reader to related patterns	street café, promenade, park

**Table 5.1** Structural components with which Alexander *et al.* (1979) presented their design patterns

## 5.2 Design Patterns and Pattern Libraries Across Computing Fields

When applied to computing, design patterns explain a problem in relation to its context and prescribe an abstracted best practice for generating a solution. A pattern is “not necessarily a code solution ready for copy-and-paste but more of a best practice, a useful abstraction, and a template for solving categories of problems” (Stefanov 2010: 1). Alexander’s writing on design patterns influenced various fields involving computing and software interface design over the last twenty-five years. We consider the treatment of these in three broad categories: Software Engineering, HCI, and web design and development. **Software Engineering (SE)** is a field emerging from Engineering, is closely aligned with the discipline of

Computer Science (CS), and primarily addresses software engineering development processes, software design, and software tools and methodologies (Christensen and Thayer 2005). ***Human-Computer Interaction (HCI)***, closely related to the practically-oriented field of Usability Engineering (UE), is a multi-disciplinary approach to the interface between humans and computer systems that employs the method of User-Centered Design (UCD) to prioritize the end user within the design process to help insure user interface success (Robinson *et al.*, 2005; Roth and Harrower 2008; Haklay 2010). ***Web design and development***, while informed by SE and HCI, encompasses the technologies and practices involving the production and maintenance of websites, and is not a formal discipline *per se*, but more an amalgam of amateur and professional practice. What follows is an interrogation of design patterns and similar approaches within each to discern their respective approaches, differences, and relative strengths and weaknesses in the service of developing a pattern library for web mapping.

A comprehensive review of the treatment of design patterns and pattern libraries across computing fields reveals great variation in both what a design pattern even is (an ontological question) and how it is best discovered, constructed, presented, and employed (an epistemological question). It is therefore useful to consider each according to the criteria defined in Table 5.2. Not every criterion is applicable to each approach, but these criteria provide a useful means for comparison. This section proceeds by analyzing patterns according to these criteria, as well as relating to Alexander's tenets listed above when relevant.

#	Criteria	Definition
1	goal	What is the aim of the design patterns? What is the intent trying to achieve?
2	target audience	To whom are the patterns intended to be useful for?
3	rule	Does the pattern acknowledge the solution in terms of a context, forces, and resolution type of rule, or some alternative?
4	approach	How are the design solutions identified or derived? From heuristics and best practices? From established or theoretical principles?
5	presentation	What is the structure or what are the components that compose the design pattern presentation (e.g., name, example, diagram).
6	format	How are the design patterns encoded and presented (i.e., written text, diagrams, descriptions of examples, actual working examples, code snippets), including the architecture of web-based presentation formats?
7	organization	How are individual design patterns organized into higher-level categories?
8	stack focus	Are the design patterns intended to solve more 'back end' programming-oriented problems or more 'front end' interface problems?
9	relational	Are the design patterns organized in a way that they relate to both each other and to a larger composite whole?

**Table 5.2** Criteria for evaluating makeup of various approaches to pattern libraries.

### 5.2.1 Design Patterns in Software Engineering

Design patterns surfaced within SE as early as Beck and Cunningham's (1987) adaptation of Alexander's pattern language to object-oriented programming.<sup>33</sup> Consistent with Alexander's intent that patterns help occupants design their own homes and towns, Beck and Cunningham applied this to computer programming and posed design patterns as a mechanism allowing computer users to "write their own programs" (1). To demonstrate the feasibility of this claim, they describe a small pattern library for designing window-based user interfaces, coded in the object-oriented programming language Smalltalk. Beck and Cunningham asserted that given these patterns, a team of application specialists was able to build "very reasonable interfaces" with little knowledge of Smalltalk's internal mechanisms.

Influenced by Beck and Cunningham, Gamma *et al.* (1995) actuated broad interest in design patterns within SE and established an approach to their creation and use that greatly influenced their subsequent understanding and treatment. Departing from Alexander, as well as Beck and Cunningham, Gamma *et al.*'s employment of design patterns was focused more narrowly on providing well-established, object-oriented coding solutions *for programmers*, rather than for another domain specific or general audience.

<sup>33</sup> Their technical report was presented at the OOPSLA-87 workshop on the Specification and Design for Object-Oriented Programming. See <http://c2.com/doc/oopsla87.html>

Examples of the patterns they documented, familiar to experienced programmers, include: the Singleton pattern, which ensures a given class has only one instance and provides a global point of access to it; the Observer pattern, which defines a one-to-many dependency between objects that updates all dependencies automatically when one object changes state; and the Prototype pattern, which specifies the kinds of objects to create using a prototypical instance and creates new objects by copying this prototype. As these examples demonstrate, Gamma *et al.*'s design patterns were intended neither for novice coders nor the users of the software. Though the goal was to bestow best practice solutions for less experienced software developers, a good design pattern according to Gamma *et al.*'s criteria captured insights that could “inform even an experienced designer” (Dearden and Finlay 2006, 17). The goals of their patterns were to name, abstract, and identify common design solutions to make them more useful and reusable within object-oriented programming.

Gamma *et al.*'s work influenced experiments with design patterns among software engineering and prompted an annual Pattern Languages of Programming (PloP) conference in 1994 that continues to meet annually.<sup>34</sup> Such efforts within SE included Coplien and Schmidt (1995), Budinsky *et al.* (1996), Martin *et al.* (1998), and Harrison *et al.* (1999). Drawing upon Alexander's rule, each of their patterns had four essential elements: (1) a **pattern name**, which served as a handle for the design problem as well as contributed to the library's vocabulary, (2) a **problem**, which described the context and when to apply the pattern, (3) a **solution**, which offered a template-like description of how elements need to be modified or rearranged to solve the problem, and (4) the **consequences** (i.e., Alexander's 'configuration'), which were the results or trade-offs of applying the pattern (Borchers 2001). Gamma *et al.* also organized their patterns under three broad categories of *purpose*: (1) **creational** design patterns that dealt with the creation of new objects, (2) **structural** design patterns that were concerned with object composition and the relationships between different objects, and (3) **behavioral** design patterns that focused on

---

<sup>34</sup> See <http://www.hillside.net/plop/2014/>

communication between objects in a system.<sup>35</sup> In their approach to these organizing categories, Gamma *et al.* largely determined the path forward for design patterns within SE and eventually other fields in computing (Borchers 2001).

Nearly twenty years later, Osmani (2014) promoted pattern libraries for modern web development by focusing specifically on JavaScript coding solutions. Though intended more for web applications than traditional software development—as it was largely congruent with that of Gamma *et al.* (1995) in terms of target audience, approach, format, and organizing categories—I am including it within the SE category. Osmani’s patterns aimed to accelerate interactive response times within web applications and improve a seamless user experience within a modern web browser, rather than addressing code written in a pre-compiled object-oriented language intended for execution on a desktop or server processor. While Gamma *et al.*’s examples provided code examples in the object-oriented language of Smalltalk, Osmani’s pattern examples were written in JavaScript and demonstrated the application of the solutions using a limited amount of DOM elements (i.e., HTML elements). An indication of the staying power of Gamma *et al.*’s patterns, Osmani patterns are nearly identical in structure and organization, simply translated into the modern web environment. Osmani also importantly emphasized that patterns can be applied not just to vanilla JavaScript (i.e., standard JavaScript code), but also to abstracted libraries such as jQuery. Table 5.3 provides a comparison of Beck and Cunningham’s (1987), Gamma *et al.*’s (1995), and Osmani’s (2014) approaches to design patterns and pattern libraries within the discipline of Software Engineering.

#	Criteria	Beck & Cunningham (1987)	Gamma <i>et al.</i> (1995)	Osmani (2014)
1	goal	allow users to write own programs	share best programming solutions among software developers	improve their knowledge of design patterns and how they can be applied to the

<sup>35</sup> Gamma *et al.* further classified their patterns according to a second criterion of *scope* specifying whether the pattern applies primarily to classes or objects, but this nuance is not germane to the larger discussion within this chapter.

				JavaScript programming language
2	target audience	(novice) users	programmers	professional software developers
3	rule	n/a	discovered by reference to design solutions	n/a
4	approach	n/a	name, problem, solution, consequences	context, system of forces, configuration
5	presentation	n/a	name, classification, intent, aliases, motivation, applicability, structure, participants, collaborations, consequences, implementation, sample code, known uses, related patterns	name, description, context outline, problem statement, solution, design, implementation, illustrations, examples, co-requisites, relations, known usage, discussion
6	format	unclear	structured text, diagrams and source code	textual description and code solutions/examples
7	organization	unclear	(of purpose) creational, structural, behavioral	creational, structural, behavioral
8	stack focus	back-end programming	back-end programming	front-end programming
9	relational	n/a	Yes, explicitly	Yes

**Table 5.3** Comparison of SE approaches to design patterns and pattern libraries

## 5.2.2 Design Patterns in HCI

References to Alexander's design patterns within HCI first appear in Norman and Draper (1986). Norman (1988) additionally attributes influence to Alexander, writing that he found him "fascinating to skim, frustrating to read, and difficult to put into practice" (Norman and Draper 1986, 229). However, it is not until later in the 1990s that HCI approached design patterns in a more intentional and systematic way. Early engagement with design patterns in HCI include Riehle and Züllighoven (1995) and Rossi *et al.* (1997) and were modeled from the approaches in SE established within the work of Gamma *et al.* (1995). Martin *et al.* (1997) were among the first to break from code-specific solutions to encourage those in HCI to consider user interface patterns as distinct and worthy of their own attention within the PLoP conference (Dearden and Finlay 2006). For instance, the categorization of HCI patterns departed from the structural, behavioral, and creational categories predominant within SE and presented design patterns as visibly rendered graphical interfaces rather than pre-rendered code. The goal of promoting solutions spanning the programming to user interface divide and targeting an audience beyond strictly software developers then became a distinguishing characteristic of HCI design patterns (Borchers 2001). A range

of specific patterns emerged within HCI and reflected the goal of creating a successful user experience with the interface, rather than efficient computational routines.

Author	Guidelines/Standards	Design Patterns
De Souza and Bevan (1990)	difficult to integrate with experience	n/a
Chapanis and Budurka (1990)	too rigid and specific for application across different design scenarios	n/a
Bayle <i>et al.</i> (1998)	n/a	better for dealing with increasing complexity and diversity in HCI design
Mahemoff & Johnston (1998)	difficult to interpret	context and problem centered
Granlund (2001)	n/a	related to a context and are problem centered; convey knowledge about good design
Borchers (2001)	n/a	structured inclusion of examples; inclusion of solution within a context
Welie <i>et al.</i> (2000)	ignore context	makes context and problem explicit; solution is provided with rationale
Yong and Long (2009)	prescriptive but don't create new instances of a given solution	prescriptive and do create new instances of a given solution

**Table 5.4** Comparison between guidelines and design patterns, in favor of design patterns, by different HCI authors.

Some early examples of HCI engagement with design patterns manifested as design *guidelines* (sometimes referred to as design *standards*), which seek to provide recommendations for improving the visual look and feel of a product. While the aim of our research is not to resolve the ontological status of design patterns vis-à-vis alternative approaches such as guidelines, these comparisons allow insight into how HCI conceptualized design patterns in terms of their goals, composition, and utility for interface design. One example was Apple Computer's (1995) Macintosh Human Interface Guidelines, a 410-page document targeted for “people who design and develop products for use with Macintosh computers” (21).<sup>36</sup> These guidelines draw upon such HCI principles as *direct manipulation* (describes a situation where a user physically interacts with a computing system) and *consistency in the visual interface* (allows users to transfer gained knowledge of an interface across views or sections) to help designers make decisions. Usability engineering, a field that applies HCI principles to software engineering, also promoted design solutions in terms of guidelines (Mayhew 1992; Nielson 1992). For example, the

<sup>36</sup> An updated version can be found at: <https://developer.apple.com/library/mac/documentation/UserExperience/Conceptual/AppleHIGuidelines/Intro/Intro.html>



Nielson-Norman group organized their guidelines by topic, presented the guideline as a straightforward statement (e.g., “Do not create or direct users into pages that have no navigational options”), and provided good and bad examples, as well as related sources and resources (<http://guidelines.usability.gov/>). Though guidelines resemble design patterns in that they offer support for achieving a design solution, they have been distinguished—and often criticized—in a number of empirically supported ways (see Table 5.4 for a summary).<sup>37</sup> Overall, guidelines appear limited in comparison to the potential of design patterns to relate a problem to a specific context and produce a fitting solution. Additionally, guidelines are more narrowly prescriptive and hence are less apt to be generative of new solutions.

Tidwell (1999) offered one of the first widely accessible HCI pattern libraries ([http://www.mit.edu/~jtidwell/common\\_ground.html](http://www.mit.edu/~jtidwell/common_ground.html)), composed as a “set of interrelated patterns, which share similar assumptions, terminologies, and contexts” (1). The goal of her library was to offer novice designers a way to learn and use high-level principles of user interface design. She argued that such a language benefited individual designers at a specific project level, as well as improved the tools and paradigms used by the wider HCI design community. Tidwell’s approach closely followed Alexander’s tenets in terms of documenting the *context* of the problem, the *forces* in need of resolution (stated as a problem), the proposed *solution*, and the resulting context. She drew her collection of patterns straight from HCI best practices and organized them under categories that focused on the user’s actions and goals with respect to the interface. She presented her patterns in a web interface, which offered textual descriptions of the patterns (Figure 5.2a), diagrams of the solutions (Figure 5.2b), and hyperlinks to related patterns (Figure 5.3c). Tidwell updated her UI patterns in 2005, and again in 2011, accounting for changes in the web and thus added patterns relating to the integration of social media into a site or application, solutions specific to the mobile experience, and other miscellaneous new patterns such as

---

<sup>37</sup> Beyond comparison with guidelines, Dearden and Finlay (2006) also compare design patterns with *style guides* (particular to a specific design environment or product), *standards* (like guidelines but carry formal authority, such as that of the International Standards Organization), *claims* (encompassing theoretical justification and specific illustrations), and *heuristics* (broad statements of desirable characteristics).

displaying a password strength indicator and or an image carousel. She also removed patterns from the subsequent iterations that had become “blindingly obvious to everyone” (1). Since her 1999 pattern library, she also departed from a discussion of Alexander all altogether, including the use of his ‘rule’ and prescribed presentation format, instead using more approachable organizing categories of what, who, why, and how of user interaction.<sup>38</sup>

Welie *et al.* (2003) subsequently approached how good patterns are created in relation to one another and attempted to systematically address the relation of parts to whole. Drawing from Alexander’s hierarchy of scale (i.e., design patterns start at the city level, which comprise patterns at the neighborhood level, which in turn comprise patterns at the level of the home) they considered a top-down approach to organizing pieces of design within a larger design ‘puzzle.’ Rather than the spatial hierarchies as in Alexander’s case, they promoted a hierarchy of *problems*, moving from high to low-level. Welie *et al.* (2003) began by surveying more than 250 patterns emerging within HCI to establish a ‘network’ of patterns. They proposed four fundamental layers of this network: (1) *posture type patterns* which involve the genre or type of similar websites or applications (e.g., small commercial sites share common problems and solutions, while personal blog-style sites share another set); (2) *experience patterns*, which involve the user experience to support the primary user goals (e.g., ‘shopping’ or ‘playing’) as well as secondary goals (e.g., ‘information gathering’ or ‘community building’); (3) *task patterns* begin to approach concrete interactions for achieving higher-level experiences; and (4) *action patterns* which are the lowest level of building blocks (e.g., pushing a button to invoke a task pattern). Welie *et al.* proposed that the development of a pattern library would capture the hierarchical levels among patterns based on this categorization. However, the approach failed to yield a useful library and attempts to visualize relations between the patterns remained difficult to interpret (Figure 5.3).

---

<sup>38</sup> See Roth (2013) for a similar organizing structure of what, why, when, where, who, and how in terms of applying cartographic interaction.

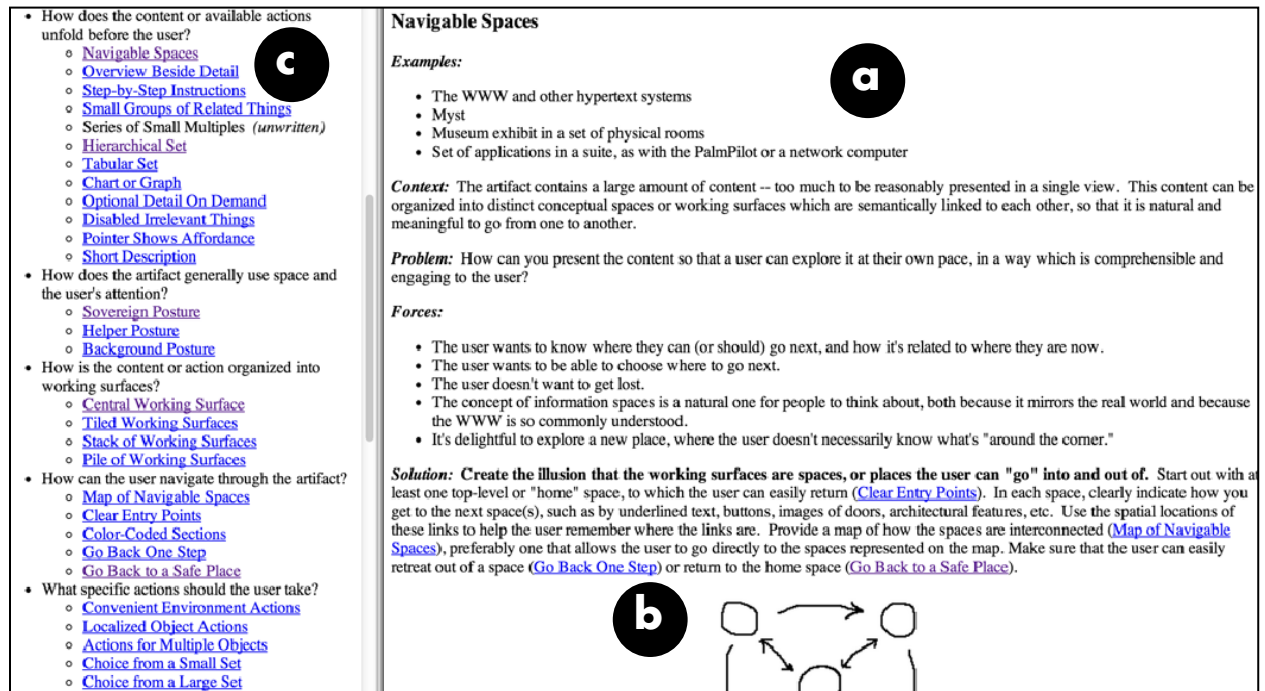
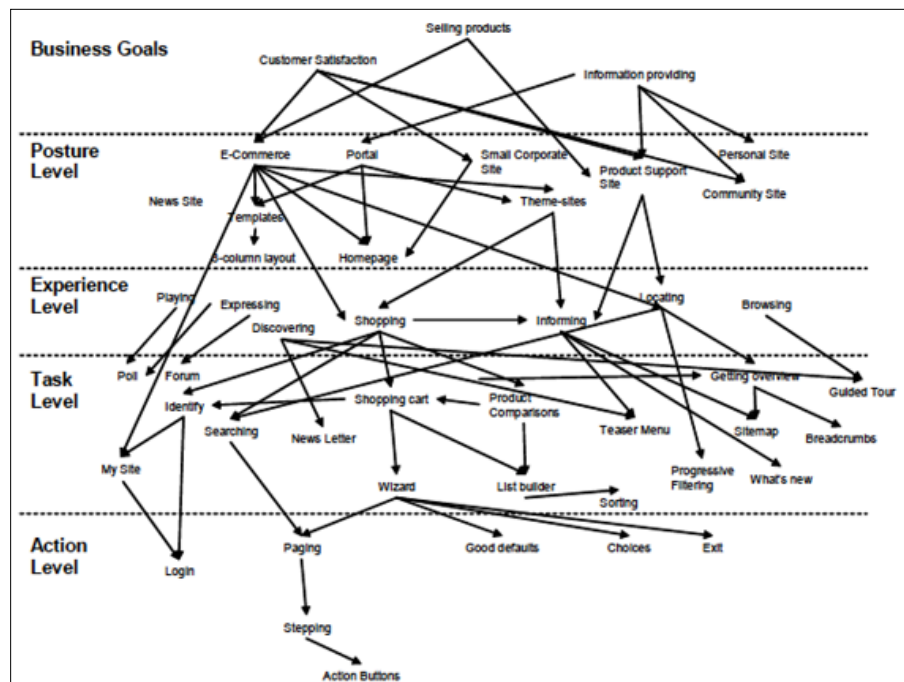


Figure 5.2 Tidwell's (1999) pattern 'Navigable Spaces.'

Figure 5.3 Welie *et al.*'s (2003) diagram indicating the hierarchical structure of related design patterns through different levels.

More useful was Welie's (2008) revised collection of UI patterns (<http://www.welie.com/patterns/>), this time organized less around a specific project's goals and instead, like Tidwell, according to HCI principles meaningfully grouped under such categories as 'Navigating Around' and 'Dealing with Data' (see Table 5.6). They also embedded their pattern library within a website allowing users to browse related patterns through hyperlinks. Welie's 2008 pattern library in many ways was merely an updated version of Tidwell's 1999 library in terms of the patterns described, examples, and format. Though—unlike Tidwell's updated patterns that were inscribed in book format—Welie's resource continued to be captured in a publically accessible web format (Figure 5.4). Table 5.4 provides a comparison of Apple's 1995 guidelines as well as the Welie (2003, 2008) and Tidwell (1999, 2010) approaches to design patterns within the discipline of Human-Computer Interaction.

### Navigating around

- [Accordion](#)
- [Headerless Menu](#)
- [Breadcrumbs](#)
- [Directory Navigation](#)
- [Doormat Navigation](#)
- [Double Tab Navigation](#)
- [Faceted Navigation](#)
- [Fly-out Menu](#)
- [Home Link](#)
- [Icon Menu](#)
- [Main Navigation](#)
- [Map Navigator](#)
- [Meta Navigation](#)
- [Minesweeping](#)
- [Panning Navigator](#)
- [Overlay Menu](#)
- [Repeated Menu](#)
- [Retractable Menu](#)
- [Scrolling Menu](#)
- [Shortcut Box](#)
- [Split Navigation](#)
- [Teaser Menu](#)
- [To-the-top Link](#)
- [Trail Menu](#)
- [Navigation Tree](#)

[< Pattern index](#)

## Accordion

### Problem

The user needs to find an item in the main navigation

### Solution

Stack panels vertically or horizontally and open up one panel at the time while collapsing the other panels

1. Accordion

2. AutoSize

It also supports three AutoSize modes so it can fit in a variety of layouts.

- **None** - The Accordion grows/shrinks without restriction. This can cause other elements on your page to move up and down with it.
- **Limit** - The Accordion never grows larger than the value specified by its Height property. This will cause the content to scroll if it is too large to be displayed.
- **Fill** - The Accordion always stays the exact same size as its Height property. This will cause the content to be expanded or shrunk if it isn't the right size.

3. Control or Extender

4. What is ASP.NET AJAX?

From [ASP net](#)

### Use when

Accordions are often used as part of [Main Navigation](#) or subnavigation. If used for navigation it is conceptually equivalent to [Tabs](#). Alternative to [Navigation Tree](#). Although accordions are often used as part of a [Wizard](#) I strongly recommend against it since it is

Figure 5.4 Welie (2008) 'Accordion' pattern, organized under a 'Navigating around' category.

<b>Criteria</b>		<b>Apple Guidelines</b>	<b>Tidwell (1999)</b>	<b>Welie et al. (2008)</b>	<b>Tidwell (2010)</b>
1	goal	promote Macintosh HIC guidelines	help inexperienced designers gain confidence and skills; help individual at specific project level; help community build better tools	provide best practice examples in interaction design (for novices)	capture common structure of familiar interface 'idioms' to facilitate flexibility and creativity
2	target audience	people who design and develop products for use with Macintosh computers <sup>1</sup>	novice interface designers	novice interface designers	designers of any user interface
3	rule	no	yes	no	no
4	approach	established styles	established best practices and solutions	established best practices and solutions	established best practices and solutions
5	presentation	textual descriptions, diagrams, screenshots	name, examples, context, problem, forces, solution, diagram (sometimes), resulting context, notes	name, problem, solution, use when, how, why, more examples, implementation, literature	what, use when, why, how, examples
6	format	textual descriptions, diagrams, screenshots	structured text and bullet lists, some diagrams	text, links, images, comments	text, screenshot examples, linked examples
7	organization	menus, windows, dialog boxes, controls, icons, color, behaviors, language	shape of the content, shape of user actions, how content or actions unfold, use of space, organization of content into working surfaces, user navigation, desired user actions, how user can modify the artifact, visual design	navigating around, basic interactions, searching, dealing with data, personalizing shopping, making choices, giving input, miscellaneous	what users do, organizing the content, getting around, organizing the page, lists, doing things, showing complex data, getting input from users, using social media, going mobile, making it look good
8	stack focus	front-end interaction	front-end interaction	front-end interaction	front-end interaction
9	relational	no	yes	yes	

**Table 5.4** Comparison of HCI approaches to design patterns and pattern libraries

### 5.2.3 Design Patterns in web design and development

As described in Chapter 2, web technologies and practices recently have shifted to harness the potential of an AJAX-enabled user experience that emulates the power of desktop software applications within the web browser while meeting the needs of a variety of browser-supported devices (including those that support touch interfaces). The web design community in turn has responded with a variety of

technological solutions to design and development within these technical enablements and constraints, such as full stack framework, open libraries, closed APIs, and web services. Outside of Web Cartography, developers bill such technical solutions as style guides, frameworks, boilerplates, front-end starter guides, front-end libraries, CSS modules, markup guides, pattern guides, pattern primers, and pattern libraries. They primarily are geared toward user interface designers to promote rapid prototyping and help less experienced developers implement web pages using fluid, flexible grid layouts using up-to-date coding standards.

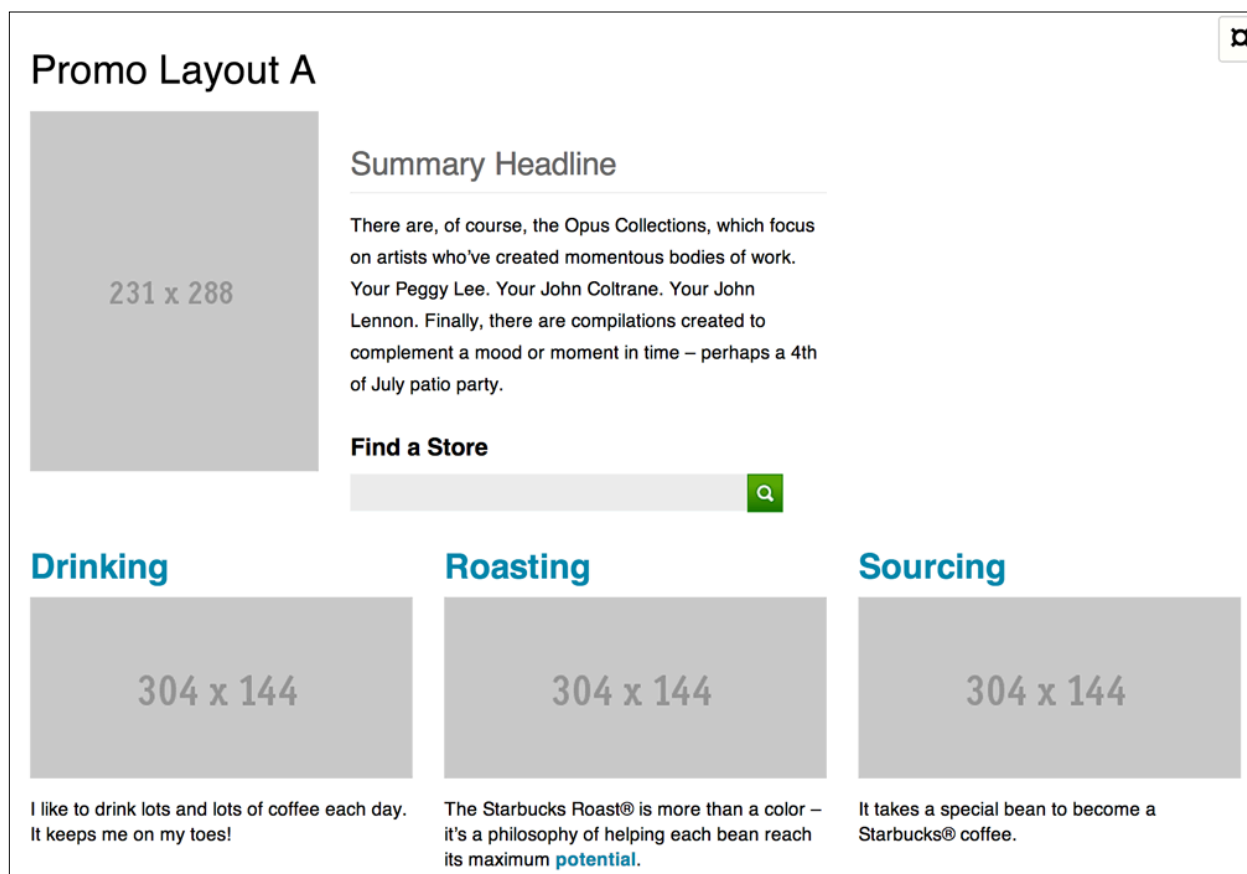
These solutions concurrently are growing in popularity as web designers and developers shift their efforts away from designing web pages and applications as single entities and toward designing for ‘systems’ in terms of modularity (Hay 2014). **Modularity** is an approach to web design and development seeking to break designs into basic fundamental building blocks. This ‘component architecture’ then allows user interface modules to be independently constructed, tested, reused and extended. Web pages and applications then are built up from these modules as a system of components. The potential of modularity specifically addresses the misconception identified in the Chapter 4 diary study regarding the usefulness of leveraging a comprehensive example to begin design and development. Breaking more complex web map examples into their constituent representation and interaction components—and relating such sub-examples to one another—is a potential means toward a threshold concept overcoming this misconception. This section discusses four approaches in Web Design and Development that increasingly approximate the concept of a pattern library as used in SE and HCI and a modular architecture: (1) style guides, (2) front-end frameworks, (3) templates and boilerplates, and (4) pattern variants.

Debenham (2013) discusses the use of **style guides**, which prescribe standards for a product. The style guide acts as a ‘living document’, growing organically with the development of a site to promote greater collaboration between designers and developers. Style guides document design solutions in a transparent and accessible way for designers, developers, and content strategists involved in the product development

process. However, they often tend to be project-specific, and serve more as a description of emerging design characteristics than tools for solving design or development problems. Though no hard lines exist between style guides and a pattern library in web design, Debenham suggests that style guides focus more on “how things are going to look” while a pattern library describes “how they work” (Debenham 213, 18). She distinguishes between three different types of style guides by their respective goals, namely those aiming to: (1) build branding consistency across a website, (2) provide content and editorial guides, and (3) establish coding standards in the process of website development. The following reviews three examples of style guides that illustrate these aims and notes specific attributes useful for a web mapping pattern library.

One of the first publicly released style guides to grab the attention of the web design community was the Starbucks style guide, which primarily served as a branding style guide but also demonstrated how individual components or modules of a design system can be broken down into smaller, maintainable elements (<http://www.starbucks.com/static/reference/styleguide/>). The presentation format of the styles was simply that of the rendered HTML and CSS (and in limited examples, JavaScript), which illustrated the aesthetic standards to which further designs consistent with the Starbucks brand would follow (Figure 5.5). Without explicitly showing any pre-rendered code, this specific approach is analogous to collections of map examples—identified by participants through the diary study—that either show no pre-rendered code or fail to explain how it works. Presumably any user seeking to make use of this style guide will have the expertise to inspect the markup and style sheets (e.g., using a web developer tool such as Firebug) to determine how to implement modifications or additions to the site.

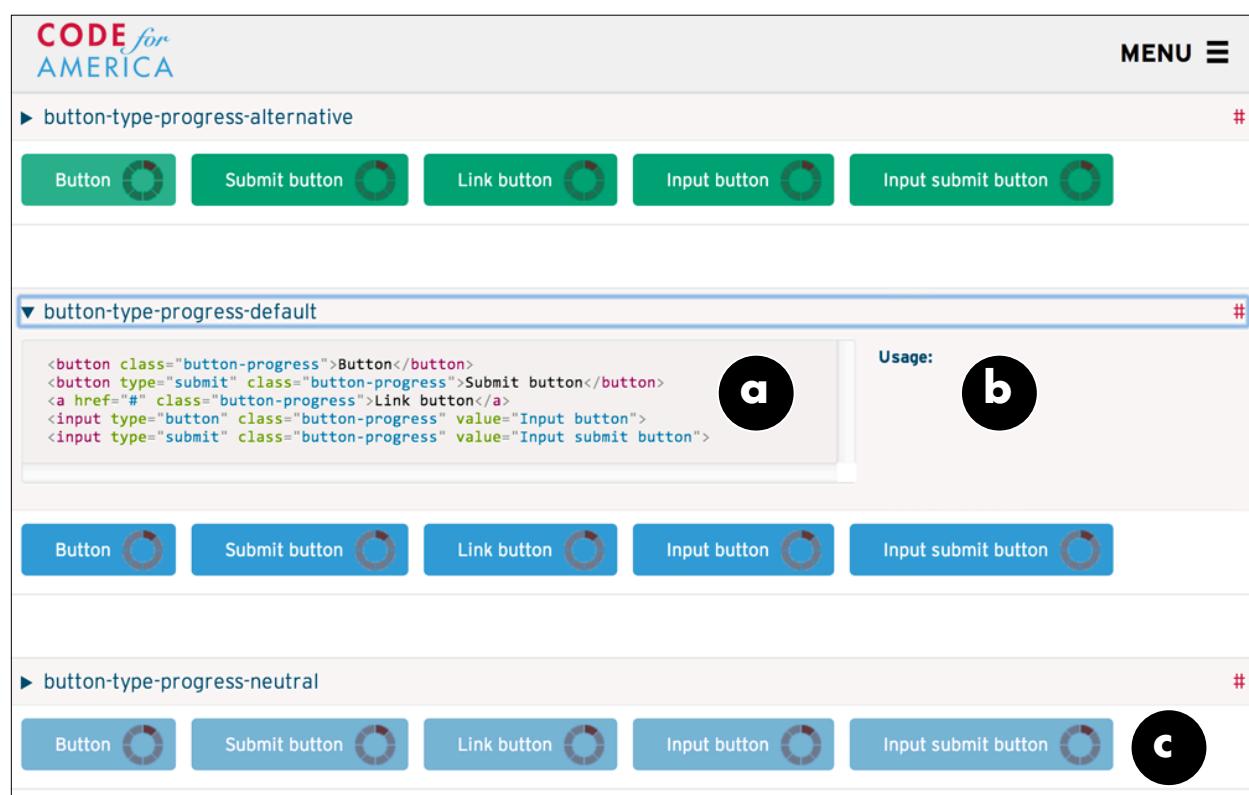




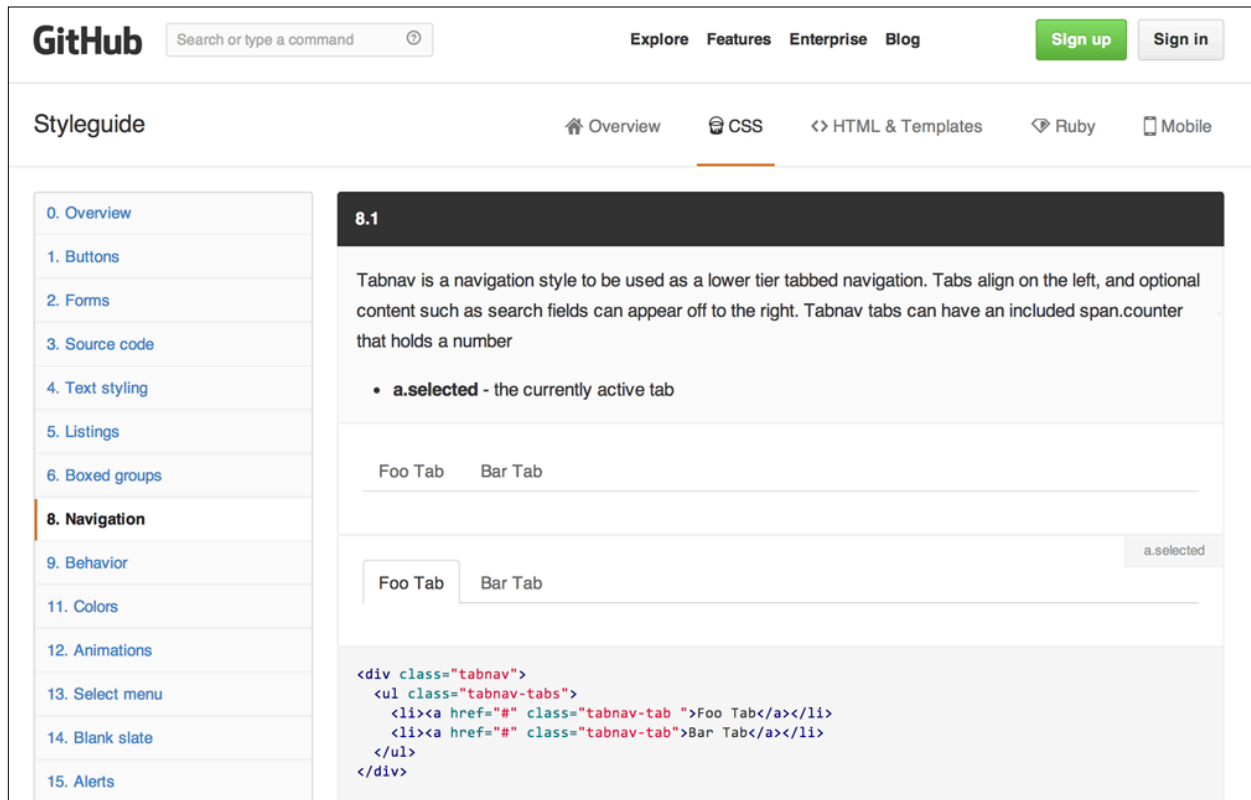
**Figure 5.5** Starbucks style guide for the 'Promo Layout A' demonstrating the constituent elements of a single page layout while emphasizing their branding.

While blending both branding identity aims and the aim of a content guide, Code for America publishes their style guide to express the value of 'openness' as a way of improving relationships between citizenry and government (<http://style.codeforamerica.org/>). This style guide, like Starbucks, accounts for the individual elements of the web page and illustrates the proper markup and styling. Additionally, the Code for America style guide includes pre-rendered code markup for all examples (Figure 5.6a), as well as recommendations for integration of various interface elements within structured semantic content (Figure 5.6b). The guide also provides a section for usage, although curiously there is limited or no documented information within these sections (Figure 5.6c).

GitHub, a version-controlled code repository web hosting service, also publically publishes their internal style guides and coding standards (<https://github.com/styleguide/css>). While informative for a general audience in terms of learning professional standards for writing good code, the intent of this style guide was to provide a resource that facilitated collaboration among several team members working on the same codebase. For example, it prescribed such specifics as the proper alignment of tabs (Figure 5.7). The goal of acting as a resource for collaboration is also of potential interest in terms of student group projects, though the individual work of the diary study limited our ability to address this potential directly (though participants did note in the exit survey the desire and importance of working collaboratively on future projects). One potential limitation of this approach is that the presentation format may make it unclear to beginning learning how to integrate specific components into a larger interface design.



**Figure 5.6** Code for America style guide, demonstrating pre-rendered HTML in relation to rendered output of elements.



**Figure 5.7** Github’s CSS style guide illustrating their recommended structure for creating a tabbed navigation.

Thus, the style guide is informative for designers and developers seeking to learn a ‘best practice’ for laying out a promotional set of interface elements or encoding class attribute values on HTML button elements. When considered in terms of web mapping, however, this approach is similar to many web map examples identified by the participants in the diary study. While the style guide breaks a more complex interface into its constituent parts, it fails to show explicitly how the HTML and CSS (and in limited examples JavaScript) work to produce the output.

**Front-end frameworks**—such as Twitter Bootstrap (<http://getbootstrap.com/>) or the Zurb Foundation’s front-end framework (<http://foundation.zurb.com/>)—provide an encompassing set of tools for building websites and applications.<sup>39</sup> Front-end frameworks are a valuable resource for designers and developers to

<sup>39</sup> Here the use of the term ‘framework’ differs from how we defined it within the two studies presented in Chapter 3. In the analysis of web mapping technologies, we used it to refer to technical solutions encompassing the ‘full stack.’ Within the web development community, a ‘framework’ commonly is used to refer to robust technical solutions for front-end development, and this is how the term will be employed within this chapter.

quickly create production-ready websites employing best practices using HTML and CSS, with optional JavaScript extensions. Additionally, the code base of these front-end frameworks can be used as a useful reference for learning professional best practices in web development. However, similar to how we found more encompassing full-stack web mapping frameworks overwhelming for achieving web mapping tasks (Chapter 3), these all-in-one solutions deliver more markup and style rules than are needed for a specific project and thereby present an excess of code and linked resources that is difficult for non-experts to navigate.<sup>40</sup> Furthermore, while the aesthetic design of websites or applications employing these front-end frameworks potentially can be modified to create truly custom user experiences, this requires further work on the part of a designer. The result is that multiple websites leveraging the same framework often end up having the same look and feel. Therefore, though front-end frameworks offer the potential to harness all three elements of the HTML/CSS/JavaScript solution space, their aim of providing near-complete products out of the box limits this potential.

Singular HTML or CSS development *templates* or *boilerplates* are a simplified alternative to the more robust frameworks that address the tendency for frameworks to look the same. Templates and boilerplates provide a gridded layout as a starting point for responsive web design and development, and offer useful code snippets of potential utility for a pattern library. Examples include the Zurb Foundation’s template (<http://foundation.zurb.com/templates.html>), which essentially strips the CSS rules and other resources from their framework and offers the bare HTML. The purpose of templates and boilerplates is to provide designers and developers with a properly structured layout of their choosing and a clear entry point to start the web design and development process (a problem identified in the diary study).<sup>41</sup> Additional styles and behavior are added later. Templates and boilerplates differ from style guides in this regard, but also in

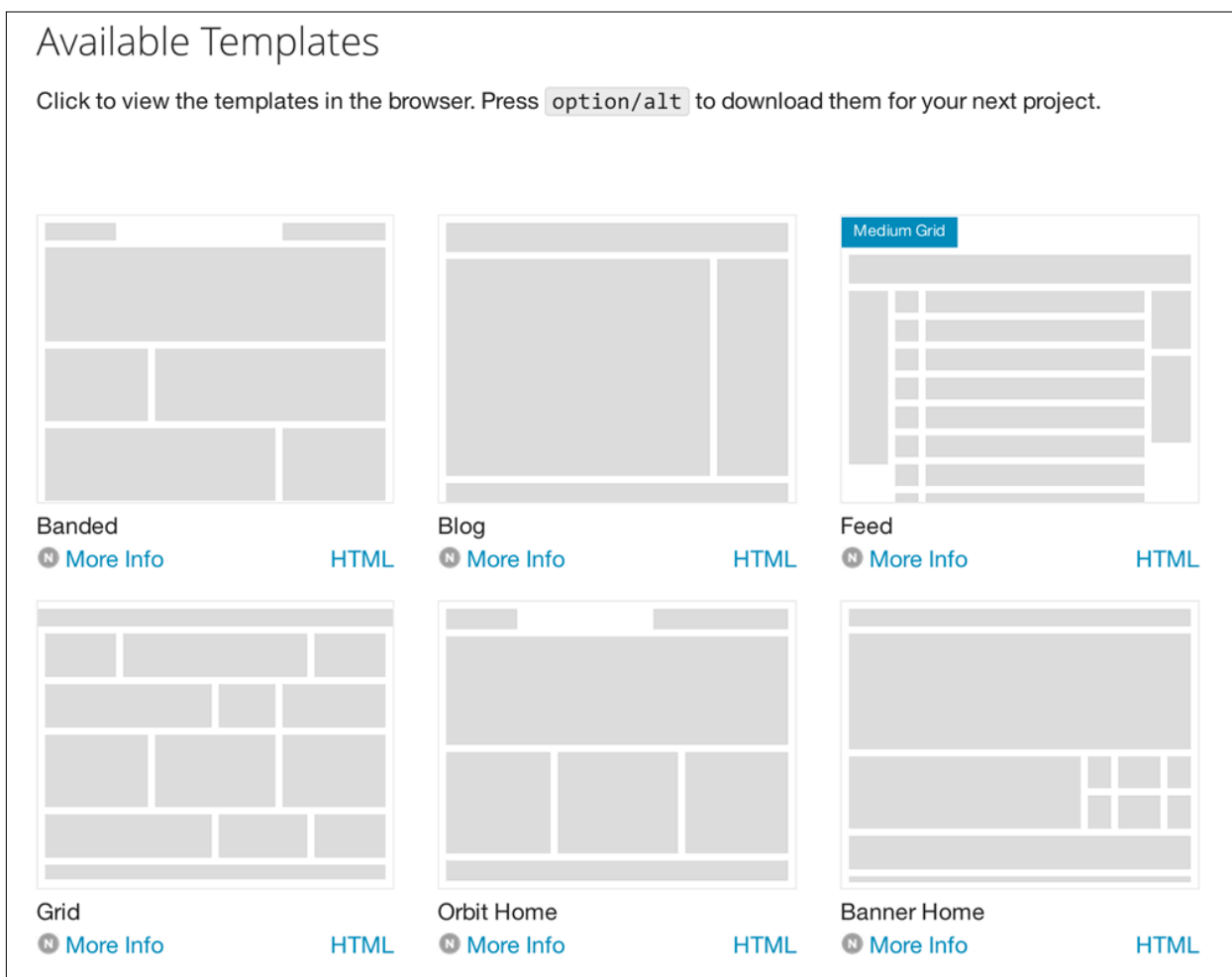
---

<sup>40</sup> This excessive code also leads to page bloat that hampers site and application performance, particularly for lower-bandwidth mobile devices.

<sup>41</sup> Note that CartoDB offers their own ‘publishing templates’ to help users embed a map within an appropriate layout, such as one offering a side panel and another layout out two maps side by side for comparison (<https://github.com/CartoDB/cartodb-publishing-templates>).

that their form is an entire layout, and not individual components or elements broken out of these templates (Figure 5.8).

Web designers also make use of a variety of products that bear the label of as ‘patterns’ including *Pattern Guides, Primers, Collections and Libraries* that serve similar web design and development needs as the other tools reviewed above. Table 5.6 provides a comparison of the aforementioned style guides, frameworks, and template/boilerplates to these pattern variants. The Yahoo Design Pattern Library was one of the first ‘pattern libraries’ emerging specifically within web design and development. Composed of fifty-nine pattern solutions for such common user interface elements as breadcrumbs (which allow a user to navigate up the site hierarchy while concurrently retaining their current location within the site) and image carousels (used when a user needs to browse among like products represented pictorially), the goal of the library was to catalogue and explain common user interface patterns with the web design and development community (<https://developer.yahoo.com/ypatterns/>). The patterns presented—as well as the presentation and format of the patterns—bear notable resemblance that the libraries of Tidwell (1999, 2010), Welie *et al.* (2003), and Welie (2008) derived from the discipline of HCI. The Yahoo patterns were organized into five higher-level categories dealing with page layout, navigation between pages, selection of elements (e.g., a color or date picker), ‘rich interaction’ (e.g., tooltips), and social media (Figure 5.9). Pattern solutions were largely descriptive, providing textual answers for such questions as ‘What problem does this solve?’, ‘When to use this pattern?’, ‘What’s the solution?’, and ‘Why use this pattern?’. Again, similar to the HCI pattern libraries, the Yahoo Pattern Library provides no working examples (examples are static image screenshots, though there are linked external examples), nor any coded solutions (pre-rendered or otherwise).



**Figure 5.8** The Zurb Foundation's template options.

#	Criteria	Style guides	Frameworks	Templates/ Boilerplates
1	goal	'living document' growing with project	all-in-one solution for building pages/site	limited 'starter' for HTML and/or CSS
2	target audience	web designers/dev	web designers/dev	web designers/dev
3	rule	no	no	no
4	approach	collect patterns with design/dev	established best practices, current trends on page layout, web standards	current design/layout trends, RWD/gridded layouts
5	presentation	common HTML element examples	full site layout with components and elements embedded within (often an accompanying README document and product site documentation)	limited documentation, README documents
6	format	single-web page of working examples	HTML5/CSS3 standard code in full directory structure with linked resources	HTML5/CSS code in single webpage
7	organization	HTML elements implemented within site	implicitly, as sections of web page, all constituent components	implicitly, as sections of a webpage
8	stack focus	front-end	front-end	front-end
9	relational	yes	potentially	potentially

**Table 5.6** Comparison of web design and development solutions approximating design patterns and pattern libraries.

Another example is that of the MailChimp Pattern Library, the aim of which was to provide a strictly defined collection of discrete UI elements that quickly could be combined to build an effective interface (<https://ux.mailchimp.com/patterns>). MailChimp's patterns for building such page elements as navigation tags or list elements show the pre-rendered HTML element structure explicitly (Figure 5.10a) and provide explanatory use case notes and examples (Figure 5.10b). While CSS style rules are applied to the examples within the library, these rules are shown not as pre-rendered code, and there are no interaction solutions using JavaScript. The library is further organized under common categories that are not directly applicable to meeting cartographic requirements. Of more value for web mapping is the presentation format itself, which allows a user to access a menu of pattern options (Figure 5.10a), see a working example (Figure 5.10b), pre-rendered code (Figure 5.10c), and detailed usage notes, all within a single view page with a clean, modern design.

Yahoo Design Pattern Library

Documentation Forum

## All Design Patterns

Here's an alphabetical listing of all 59 patterns in the library today.

As always, we welcome your feedback on these patterns as well as any stories you might like to share with us about how you've applied them to real-world design projects.

**BROWSE PATTERNS**

**Accordion**  
There are too many items to fit into a limited space without overwhelming the user.

**Add / Subscribe**  
A person wants to subscribe to the content of another person and read it in an environment of their own choosing.

**Alphanumeric Filter Links**  
The user needs the ability to look up information alphabetically within a large data set.

**Calendar**  
The designer needs to

**Architecture of a Review**  
A product or website needs to

**Auto Complete**  
The user needs to enter an item

**Yahoo Forum Discussions**

**What Design Patterns can I see in this diagram?**  
Tue, 26 Jun 2012

**Re: how to download**  
Wed, 16 May 2012

**Re: I can't seem to get patterns into Adobe Fireworks..**  
Fri, 10 Feb 2012

**Re: Starting a pattern library**  
Wed, 18 Jan 2012

**Re: How to Convert SWF to AVI on Mac OS X Lion**  
Wed, 16 Nov 2011

[More](#)

**MORE ABOUT PATTERNS**

Figure 5.9 The layout of the Yahoo Pattern Library, illustrating common patterns

**Pattern Library**

## Navigation

### Local Navigation

Example

Account settings ▾ **My profile** Billing ▾ Extras ▾ Rewards ▾

**a**

- Grid System
- Typography
- Form Elements
- Navigation**
- Tables
- Lists
- Slats
- Stats/Data

**b**

```

1 <p class="mobile-top-nav nomargin full-width fw"
2 >
3 <a href="#">Navigation</a>
4 <ul class="local-nav selfclear">
5 <li>
6 <ul class="hover-list">
7 <li>
8 <a href="#">Account settings</a>
9 </li>
10 <li>
11 <a href="/account/users">Users & a
12 ccount details</a>
13 </li>
14 <li>
15 <a href="/account/contact">Contact

```

**c**

Local navigation is used to jump between pages that categorically fall under the same (main) section. The local nav shown in the example above is from the Account section and it is present on all pages under /account.

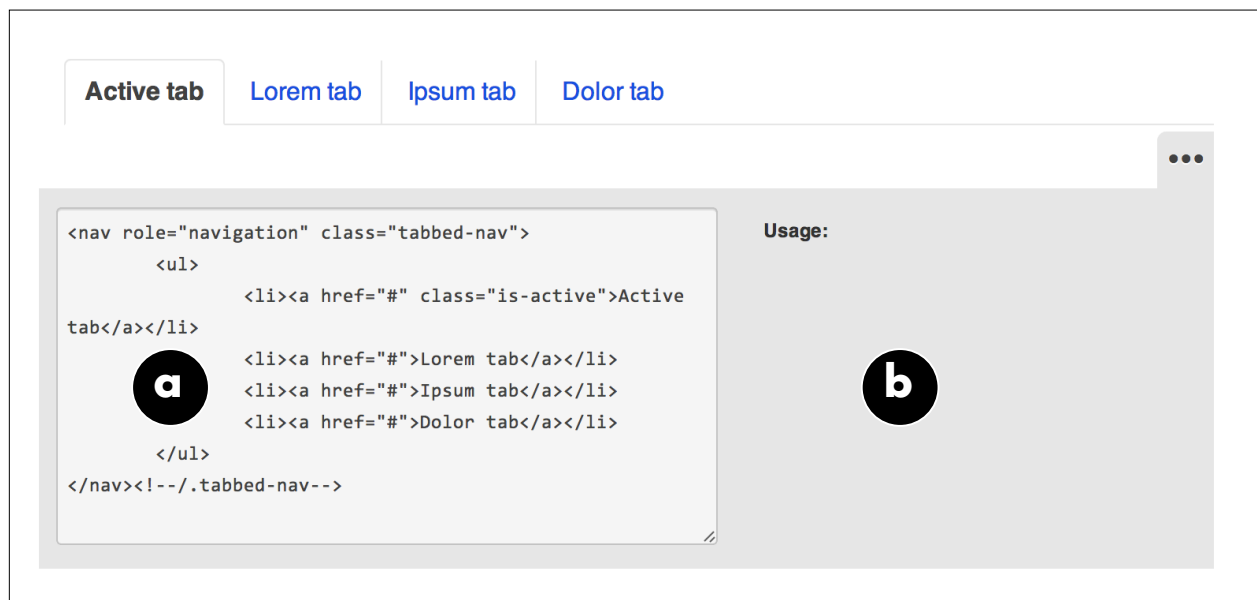
If there are secondary pages in a section then links to those pages are shown inside **hover-list** menus. A blue-colored link indicates the page/sub-section currently in view.

The local nav spans the full width of the page up to 640px at which point it collapses into a stacked menu.

Figure 5.10 MailChimp's Pattern Library.



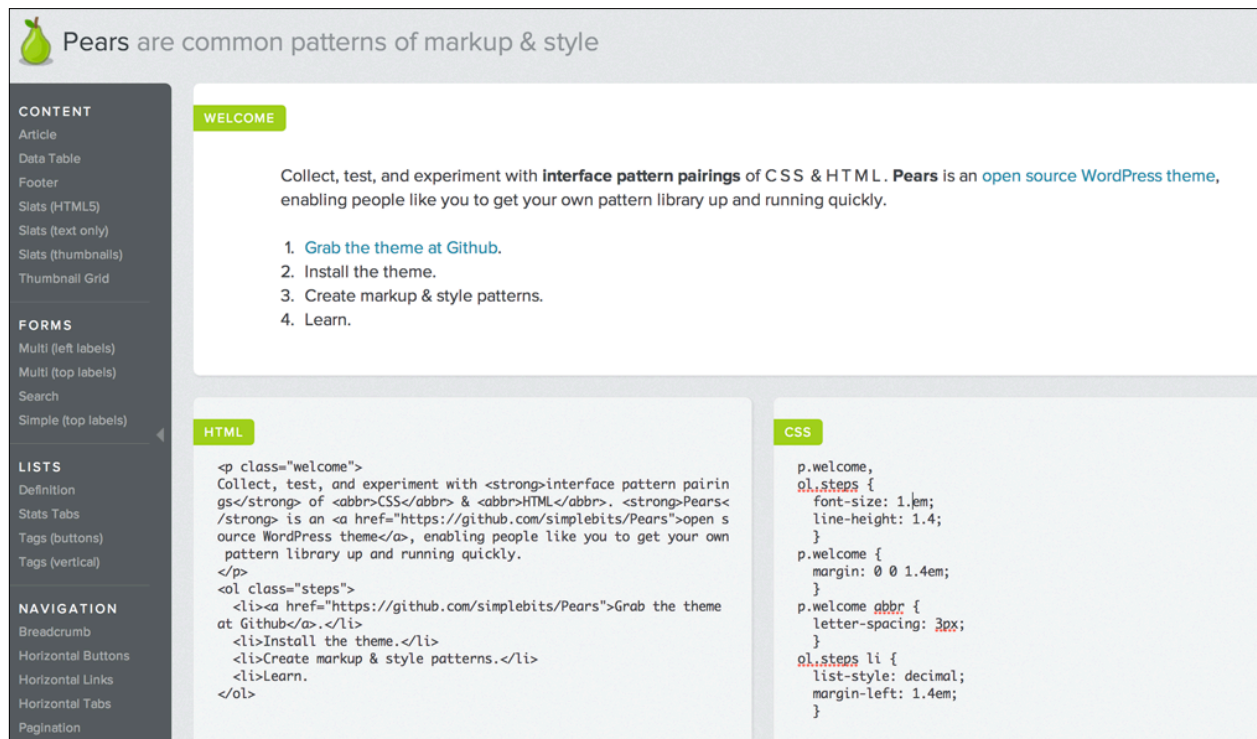
Barebones offers a directory setup guide, accompanying style guides and what is dubbed a ‘pattern primer.’ The goal of this approach departs from that of the Yahoo or HCI pattern libraries in that it does not explicitly catalog or prescribe solutions to interface problems, offering only a few examples of “common snippets of markup” (<http://barebones.paulrobertlloyd.com/>). The presentation format includes both explicit pre-rendered HTML (Figure 5.11a) and space to document relevant usage (Figure 5.11b). Rather than informing a way forward for a web mapping pattern library in terms of the approach to identifying scoped patterns, Barebones’ intent is to provide a technical starter for developing one’s own pattern library. Therefore, it is of potential utility for consideration in the architecture of presenting individual patterns, rather than offering any specific guidance in terms of organizing patterns or relating patterns to each other within the library.



**Figure 5.11** The Barebones Pattern ‘Primer.’

The Pears ‘Pattern Pairings’ library offers a noteworthy consideration in terms of its format (<http://pea.rs/>). Pears fulfills an often unmet need in many of the approaches analyzed by in explicitly demonstrating the relationship between HTML and CSS rules presented as pre-rendered code (Figure 5.12). Furthermore, it does so in an instantaneous, live-feedback format employing an AJAX solution to

update rendered examples as the user modifies the code. Any changes to either the markup or the CSS are immediately viewed within the application.<sup>42</sup> Currently, the Pears library is available as a Wordpress Theme, which offers a novel architectural option for technically storing individual design patterns within a database and rendering as Wordpress blog ‘custom posts’.



**Figure 5.12** The Pears ‘Pattern Pairings’ interface, rendered through the Wordpress content management system.

Given the importance of understanding the interplay between the structural, stylistic, and behavioral code of front-end web development—a threshold concept identified in the diary study—a pattern library such as that of Patternry is instructive for establishing a format for such presentation (<http://patternry.com/>). Patternry, though a pay-for-service pattern library, provided examples that allow users not only to see the pre-rendered HTML, CSS, and JavaScript (Figure 5.13a), but allowed users to modify the code directly in the library’s page to immediately see the resulting changes in the resulting output (Figure 5.13b), as with the Pears example.

<sup>42</sup> This principle of ‘instanaenous feedback’ is championed by Bret Victor (2012) in his discussion of ‘learnable programming’ (<http://worrydream.com/#!/LearnableProgramming>).

Frost promoted modularity in terms of building systems of interface components in web design and development with his novel approach of ‘atomic design’ (<http://bradfrostweb.com/blog/post/atomic-web-design/>). Following from component architecture, Frost used chemistry as a metaphor for breaking interfaces down into their fundamental building blocks, and then reassembling to solve particular interface problems (Figure 5.14). In this model, *atoms* are the basic building block of an Open Web Platform webpage or application and consist of such HTML elements as a simple *paragraph* element, an *img* tag for loading an external image resource, and *button* or *form* elements for user input. Atoms are combined to create *molecules*, or the functional elements of a user interface. For example, the atom-level *button* element may be combined with the atom-level *form* element to create a functional molecule-level interface element in which the user can key in a word and push the enter button to submit a query.<sup>43</sup> Continuing with the chemistry analogy, molecules then are combined to create entire *organisms*. Frost’s notion of *templates* (which consist of page-level elements) and *pages* (which are specific instances of templates that contain actual content and a polished visual design) break from the chemistry analogy, but describe the final realization of the design process.

---

<sup>43</sup> This example likely would require some additional behavioral code for the functional aspect of the molecule to work.

# Accordion HTML

Also known as: **Collapse**

Tags: [components](#), [content](#), [navigation](#)

An accordion is a grouped set of collapsible panels that the user can open and close by hovering on or clicking on the title of a specific panel. Accordion allows organizing the content into logical, titled sections in a space-saving way.

Accordions usually work best when the user can have only one section open at a time and each section is independent from each other. When the panels provide parallel context (for example search filters), allowing user to have several sections visible simultaneously might be a good idea.

**Note:** To make the example accordion work you need [Collapse Plugin](#) from [Twitter Bootstrap](#)

[Show preview in new window](#)

## Collapsible Group Red

Red is the color of blood, a ruby, and strawberries. It is next to orange at the end of the visible spectrum of light, and is commonly associated with danger, sacrifice, passion, fire, beauty, blood, anger, socialism and communism, and in China and many other cultures, with happiness.

## Collapsible Group Violet

## Collapsible Group Orange

HTML

CSS

Javascript

```

1 <div id="accordion2" class="accordion">
2   <div class="accordion-group">
3     <div class="accordion-heading red">
4       <a href="#collapseOne" data-parent="#accordion2" data-
toggle="collapse" class="accordion-toggle">
5         Collapsible Group Red
6       </a>
7     </div>
8     <div class="accordion-body collapse in" id="collapseOne">
9       <div class="accordion-inner">
10        Red is the color of blood, a ruby, and strawberries. It is next to
orange at the end of the visible spectrum of light, and is commonly associated with
danger, sacrifice, passion, fire, beauty, blood, anger, socialism and communism, and in
China and many other cultures, with happiness.
11      </div>
12    </div>
13  </div>
14  <div class="accordion-group">
15    <div class="accordion-heading violet">
16      <a href="#collapseTwo" data-parent="#accordion2" data-
toggle="collapse" class="accordion-toggle">
17        Collapsible Group Violet
18      </a>
19    </div>
20    <div class="accordion-body collapse" id="collapseTwo">
21      <div class="accordion-inner">
22        The color violet takes its name from the violet flower.[1] On the

```

Figure 5.13 Patternry's pattern output, coupled with pre-rendered HTML, CSS, and JavaScript.



Figure 5.14 Frost's 'atomic design' workflow.

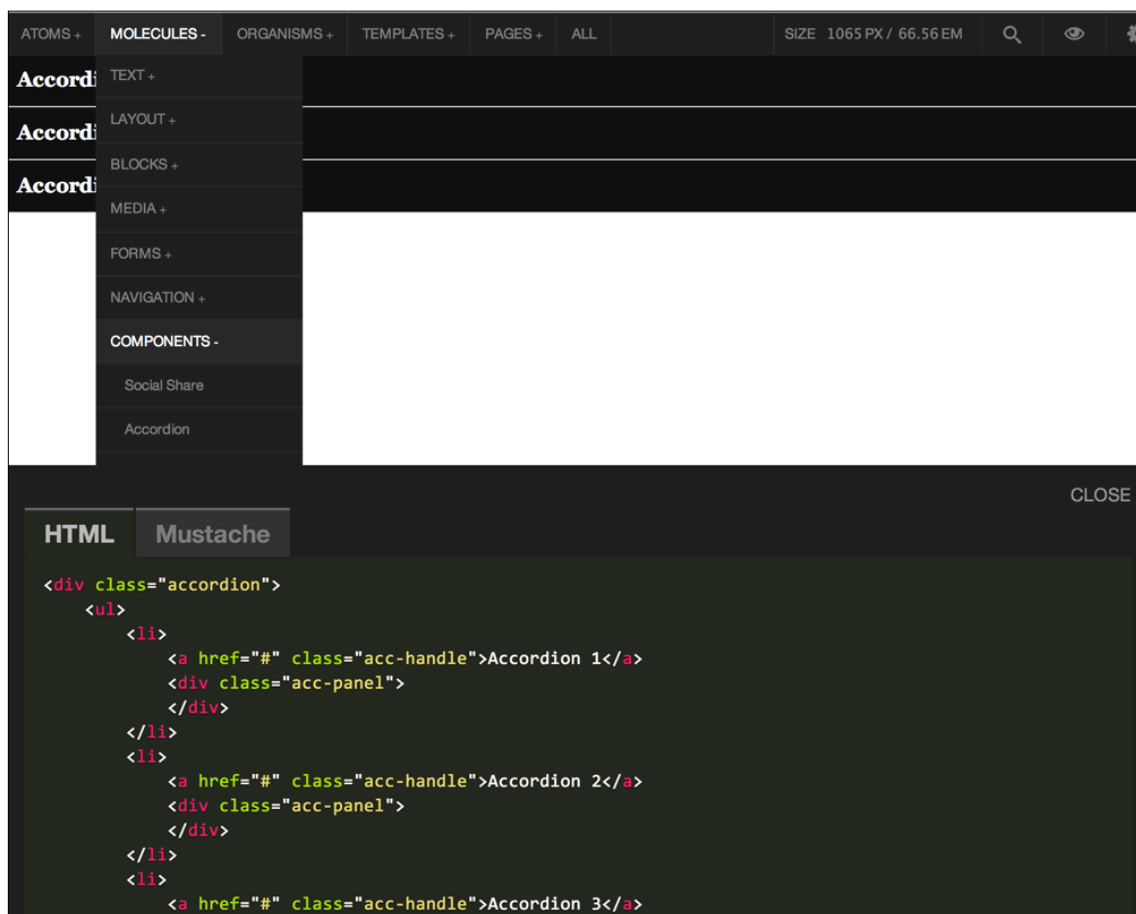


Figure 5.15 Frost's Pattern Lab interface, in this example showing an example for an 'Accordian' pattern, which comprises list elements. Note that the pre-rendered structural HTML is shown, but the CSS rules and accompanying JavaScript are not.

Frost advocates moving from the smallest, most abstract units of (i.e., atoms) toward more concrete, tangible interfaces composed of these various elements (Figure 5.14). He applied this design philosophy to a pattern library named ‘Pattern Lab,’ which provides a dynamic architectural format for building a web page or application using the modular approach (<http://patternlab.io/>). Like the Barebones example, although much more robust, the goal of the Pattern Lab is not to provide a catalog of existing representation or interaction solutions. Rather, it is a tool for designers and developers to build their own libraries in order to capture and share their own design solutions (i.e., patterns). The format itself provides a rendered example in the browser, an inspector to view the responsible HTML markup (or Mustache, a logic-less template language similar to HTML). Like many like other examples, CSS rules are not visible as pre-rendered code and remain hidden within a linked style sheet. The potential to integrate JavaScript solutions or examples within the Pattern Lab also remains opaque.

#	Criteria	Yahoo	Patternry	Barebones	Pears	Pattern Lab
1	goal	capture UI solutions	template-like resource for user defined solutions	template-like resource for user defined solutions	provide interface for pairing HTML and CSS interplay	static site generator, component library, pattern starter kit
2	target audience	web designers/devs	web designers/devs	web designers/devs	web designers/devs	web designers/devs
3	rule	described as usage case	potentially described as usage case	potentially described as usage case	potentially described as usage case	potentially described as usage case
4	approach	known, common solutions	open	open	open	small UI components to larger
5	presentation	problem, when to use, solution, recommendations, options, why use	notes, useful links	usage notes	none	usage notes
6	format	static web pages	dynamic web pages	static web pages	dynamic Wordpress pages	PHP rendered static web pages
7	organization	layout, navigation, selection, rich interaction, social	open	open	open (demo: content, forms, lists, navigation)	atoms, molecules, organisms, templates, pages
8	stack focus	front-end	front-end	front-end	front-end	front-end
9	relational	limited	potential if described in usage case	potential if described in usage case	potential if described in usage case	high, explicitly and hierarchical

**Table 5.7** Summary comparison of web design and development solutions approximating design patterns and pattern libraries.

A comparison of the pattern variants within web design and development reveals consistency primarily across their target audience and their lack of acknowledgement of Alexander (or application of his ‘rule’) in their format and organization (Table 5.7). The goals range from providing a general UI resource for the web development community, to documenting specific solutions for a given website, to acting more as starter kits for users to build up their own pattern libraries. This latter goal is of potential greater utility for a web mapping library, in particular the Pears and Frost Pattern Lab. The emphasis on component architecture of the Pattern Lab option also dovetails with the interest on approaching modularity practices

for the proposed web mapping pattern library solution. Additionally, the formats explicitly providing pre-rendered code—and in particular those like Patternry and Pears that show the inter-relatedness of web standard elements of HTML and CSS—speak toward unmet needs in educational resources and demonstrate the interplay of elements within the web mapping solution space. Overall, most of these resources demonstrate currently updated presentation formats using current web standards, particularly Frost’s Pattern Lab.

### 5.2.4 Application of design pattern approaches across disciplines

The three prior subsections characterize approaches to design patterns in the disciplines of Software Engineering, Human-Computer Interaction, and web design/development. Table 5.7 provides a synoptic comparison of design pattern approaches across these three computing fields. The treatment of design patterns in SE primarily is concerned with writing proficient coding solutions at an advanced level, and are therefore patterns more concerned with development than solving UI design problems. Even Osmani’s (2014) JavaScript design patterns—which include HTML elements in examples and are therefore more closely connected to rendered output within a browser than Gamma *et al.*’s (2005)—are focused on making computation more efficient. The uptake of design patterns within HCI, by contrast, is predominately more conceptual in nature and offers descriptive and diagrammatic solutions to user interface problems. While initially modeling their presentation on Alexander’s explicit discussion of the context, rule, etc., HCI pattern libraries eventually loosened this in favor of organizing solutions around questions of user-centered design. The examples emerging from web design and development fall on a range between those offering conceptually useful solutions for interface design problems similar to HCI, and a less useful model for a pattern library seeking to approach those specifying coded solutions using modern web standards, though rarely making use of all three of the structural (HTML/SVG), stylistic (CSS), and behavioral (JavaScript) constituents of a complete web map application. Notably, the web design and development solutions depart from Alexander’s ‘rule,’ at least stated in terms of stating a problem in terms of a context, forces, and solution. However, the supplement of ‘usage’ notes for patterns



effectively may achieve this. The utility of explicitly using Alexander's rule and presentation structure, at least as initially reified by Gamma *et al.* (2005), is an open question, and the shift in Tidwell's pattern library is worth further investigation in this vein.<sup>44</sup>

In more advanced web mapping interfaces—particularly those involving high numbers of rendered elements bound to large data sets—the SE patterns will prove edifying. However, for our target audience of beginner students, it is difficult to directly transition approaches in SE to learning simpler solutions for implementing cartographic representation and interaction requirements. In an educational context, this approach may find greater utility complementing lecture material that emphasizes HCI design principles over specific implementation solutions, as they rarely supply coded solutions or examples. In this sense, they are true to the goal of being concrete, yet are abstract enough to be applied within different languages or technologies (a need for the rapidly shifting technological terrain the overall research process aimed to address). The HCI format therefore not only is useful for demonstrating concepts to students, but facilitating a living document for students to potentially experience such an 'a-ha' threshold moment.

---

<sup>44</sup> It should be noted that Alexander *et al.* (1979) did not explicitly encode his patterns in the presentation format captured in Table 5.1, but rather implicitly using syntax.

#	Criteria	SE	HCI	Web design & dev
1	goal	present best-practice coding solutions	present known UI solutions	known UI and presentation solutions; provide template structures for building libraries
2	target audience	moderate to expert programmers	interface designers	web site designers and developers
3	rule	yes	yes	no
4	approach	capture established solutions	capture established solutions	capture solutions as they emerge
5	presentation	largely textual description with examples, follows Alexander's prescription	textual descriptions, static examples, Alexander's prescription and how, when, who, why explanations	rendered working examples, usage descriptions, some pre-rendered code
6	format	printed textbook	web-based interface	web-based interface
7	organization	creational, behavior, structural	user-centered tasks	range from user-centered tasks, to page elements, to nested components
8	stack focus	back-end	front-end	front-end
9	relational	high, explicit	high, explicit	variable, moderate to high

**Table 5.8** Synoptic comparison of design patterns and pattern libraries across the four application domains: (1) Software Engineering, (2) Human-Computer Interaction, (3) web design and development, (4) web mapping

The HCI workflow already is instructive for successful web map interface design. Norman's (1990) "seven principles," Norman & Nielsen's (1994) usability heuristics, or Schneiderman's (2010) "eight golden rules" for improving how users perceive and interact with systems all inform Web Cartography education at a conceptual level. To complement these approaches, students of web mapping need additional support in determining technical solutions for implementing specific design solutions. In this sense, the approaches provided by the HCI patterns above fail to provide specific development guidance. This is in part due to their desire to remain language and medium-agnostic; the principles informing Tidwell's interface solutions are intended as equally for a turnkey kiosk system as they are for building a Rich Internet Application. The HCI libraries offer an array of interface solutions, many of which may be useful for or overlap with those falling under the scope of a web mapping workflow or web map interface (e.g., an accordion-style interface may be useful for collapsing UI filter controls). But HCI patterns, while venerable in their intent to link to related patterns, do little to suggest a format helping address the sequencing of learning material. Furthermore, although such educational precepts as addressing misconceptions and enabling threshold concepts may apply to learning HCI principles—and that these pattern libraries may facilitate in those respects—these patterns do little to address the specific

development-focused concerns addressed within this dissertation. Of greater value may be their illustrated potential of applying an Alexandrian ‘rule’ to specific UI problems, which would fall on the interaction end of our emerging web mapping workflow. The HCI format for presenting design solutions within web interfaces is of potential value, as web mapping resources increasingly are integrated with a wider web community, though the available examples are dated in their own interface and visual design.

### **5.3 Approaching a pattern library for Web Cartography education**

This section returns to the question of Web Cartography education, focusing on the goal of providing positive user experiences with web maps. Our current model for instruction involves: (1) a lecture component, which focuses more upon the conceptual principles of useful representation and interaction techniques; and (2) a laboratory component, which applies these principles to practice within a design and development process culminating in the creation of a working web map. These Web Cartography principles informed the representation and interaction requirements employed within the diary study that effectually enable us to visualize complex geographic data as meaningful information (see Table 4.1). These requirements are therefore the specific, concrete bridge between principles and the end map product. This dissertation sought to address a gap—created by the inevitable technological change in hardware and software—in leveraging these cartographic requirements within the learning environment of the course laboratory component. The final section of this chapter then seeks to establish a conceptual module for how a web mapping design pattern library can potentially meet this goal.

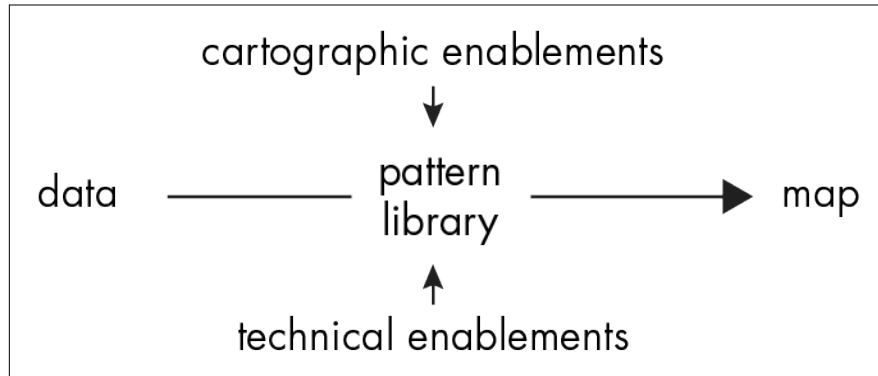
#### **5.3.1 Establishing a pedagogical model for web mapping instruction**

Chapter 4 identified a set of competencies involving specific technologies that encompass the full development stack workflow. Beyond the expected challenges of translating library-specific capabilities into solutions for meeting cartographic interaction and representation requirements, a host of additional

unexpected characteristics of the new mapping process surfaced requiring further attention. These included:

- understanding HTML, CSS, and JavaScript and how they work together to produce an interface;
- addressing limitations in the native support of web map libraries for such cartographic requirements as calculate, filter, and search;
- understanding data formats and structures (in particular the JSON/GeoJSON format) and how to integrate them into the DOM using JavaScript;
- using web development tools and the console for inspecting rendered DOM elements and debugging; and
- interpreting existing examples and modifying example code to apply to a different implementation solution.

Identifying these learning objectives is an important contribution akin to the content catalogued within UCGIS's (2006) *Body of Knowledge*. Yet a catalogue of skills and technologies does not in itself help us apply principles to practice. To this end, the diary study produced a recommended development process while identifying many specific challenges students face in approaching the learning curve and mastering web mapping. The problem remains how to successfully teach to such a broad range of demanding technical skills and apply them within this process to leverage the cartographic requirements and achieve the goal of making a web map (Figure 5.16).

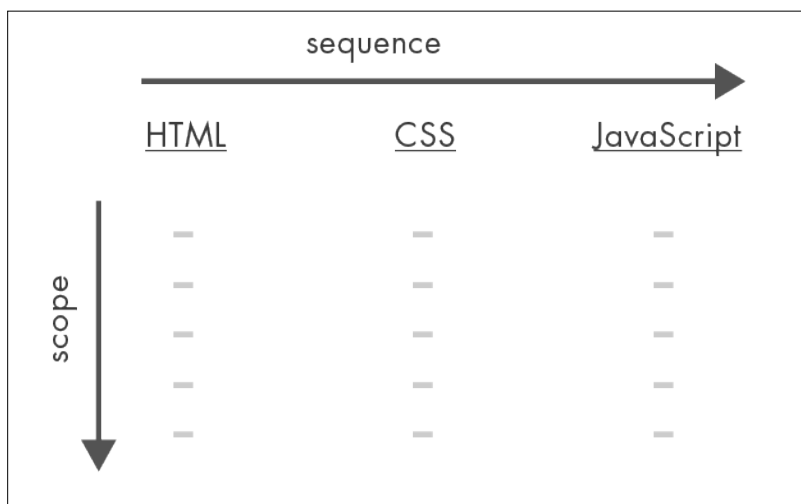


**Figure 5.16** Current educational challenge of integrating cartographic enablements with technical enablements to transform geographic data into a map.

Chapter 2 introduced and defined four concepts related to GIS&T education of potential utility in this regard: the associated notions of *scope* and *sequence* and the associated notions of *misconceptions* and *threshold concepts*. These concepts ostensibly were noted throughout the discussion of the three research studies in Chapters 3 and 4. What follows clarifies the utility of scope and sequence in particular by illustrating how a pattern library can be used to form a recommended instructional model to address the fundamental challenge diagrammed in Figure 5.16. The diary study revealed the primary importance of a solid foundation in HTML, CSS, and JavaScript programming, along with an understanding of the DOM, from which learning continued through the *data->representation->interaction* workflow. How then does the notion of a pattern library help educators approach these issues of scope and sequence to achieve the desired learning gains?

As introduced in Chapter 2, scope is best understood in terms of sequence with respect to web mapping education (Figure 5.17). How should the HTML and CSS web standards and JavaScript programming language be instructed? Our recommended sequence is first to learn the structural elements (HTML), followed by the style rules (CSS) that are applied to these structural elements, and finally the behavioral (JavaScript) logic that accesses and modifies the HTML elements and the style rules applied to them. A misguided educational approach first would exhaustively cover all 116 of the HTML elements (e.g.,

headers, lists, groupings), as well as how to apply 115 attributes (e.g., id, class, data) to these elements.<sup>45</sup> After acquiring a deep understanding of these essential skills, a student then could go on to learn all 432 CSS properties, pseudo-classes, units, and selectors before applying these rules to the HTML elements.<sup>46</sup> A third learning module then would hypothetically cover JavaScript programming (e.g., its syntax, variable declarations, built-in methods, etc.). Although clearly a straw man case, we pose it to illustrate that this would not be a good strategy. In addressing those competencies in such a manner, the scope of each technical enablement was too deep, with little attention to the cartographic enablements (Figure 5.17).

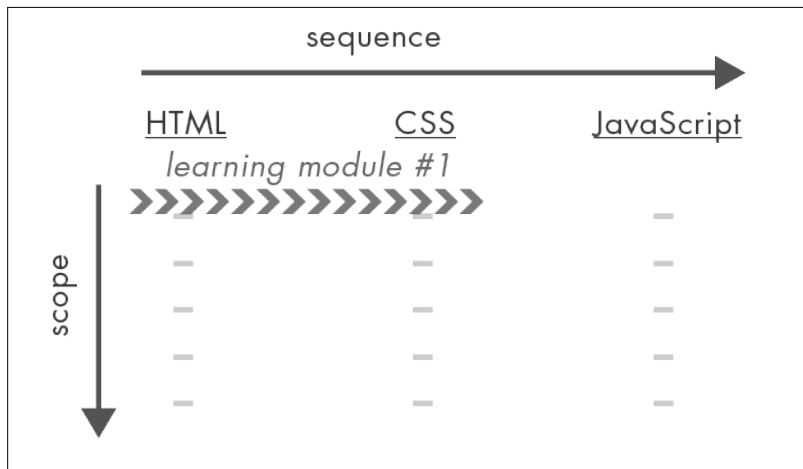


**Figure 5.17** Conceptual schema of how scope and sequence relate in web mapping education.

A better approach would be to begin with learning a few HTML elements and then introduce the CSS syntax to select these elements to apply style rules to them (Figure 5.18). Such a module would constitute a small, simple example of limited scope depth—as advocated by diary study participants—and offer an alternative entry point to a tutorial approach seeking to achieve a deeper scope through a single sequence of learning topics (though rightly replicating the *data->representation->interaction* workflow).

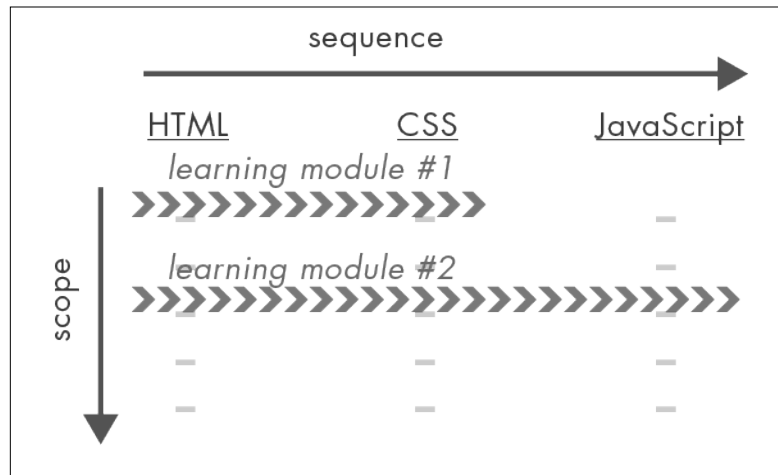
<sup>45</sup> See the Mozilla Developer Network's element reference (<https://developer.mozilla.org/en-US/docs/Web/HTML/Element>) and HTML attribute reference (<https://developer.mozilla.org/en-US/docs/Web/HTML/Attributes>)

<sup>46</sup> See the Mozilla Developer Network's CSS reference (<https://developer.mozilla.org/en-US/docs/Web/CSS/Reference>)



**Figure 5.18** Conceptual example of how a first learning module works through a sequence of topics at a shallow depth of scope.

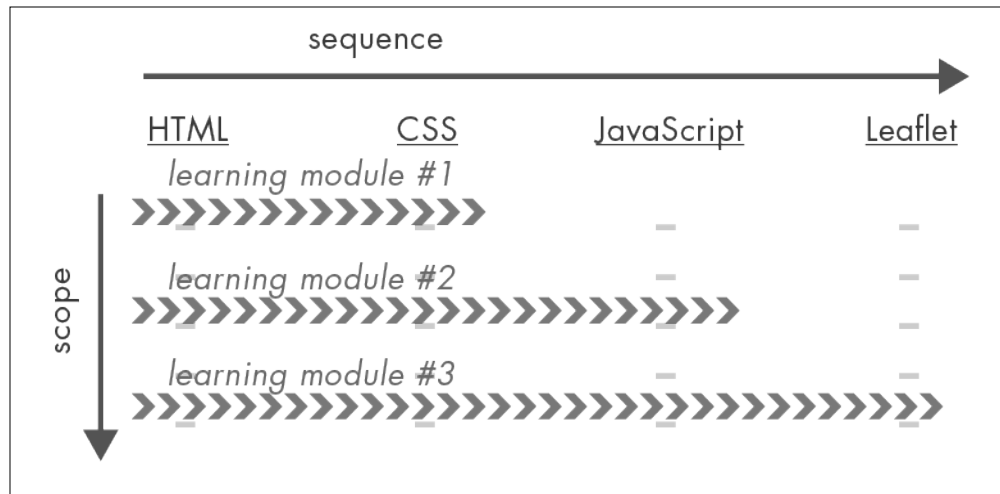
Rather than simply continuing on to the next sequential learning topic (e.g., JavaScript), an effective curriculum then may return to the first topic for a second run through those learning objectives, but this time venturing into a new topic while going into slightly more depth in those topics already covered (Figure 5.19). When considered together in this way, the employment of scope and sequence as an educational strategy is therefore one of successive runs through the material. Foote (2012), drawing from Bruner (1963), referred to this strategy as a *spiral curriculum* in which critical ideas and concepts are revisited as the instruction moves through progressively more challenging material or encompasses an ever-widening range of interconnected competencies. Foote argued that the approach also accounts for different learning speeds due to the inevitability of a student base with variable skills, experience, and abilities. Students needing more practice with fundamentals have more time to incrementally build their conceptual knowledge, while those students who have attained the desired competency quickly can move on to learn the new material.



**Figure 5.19** Conceptual example of a successive learning module, reinforcing the HTML and CSS topics at a deeper level of depth of scope while introducing JavaScript.

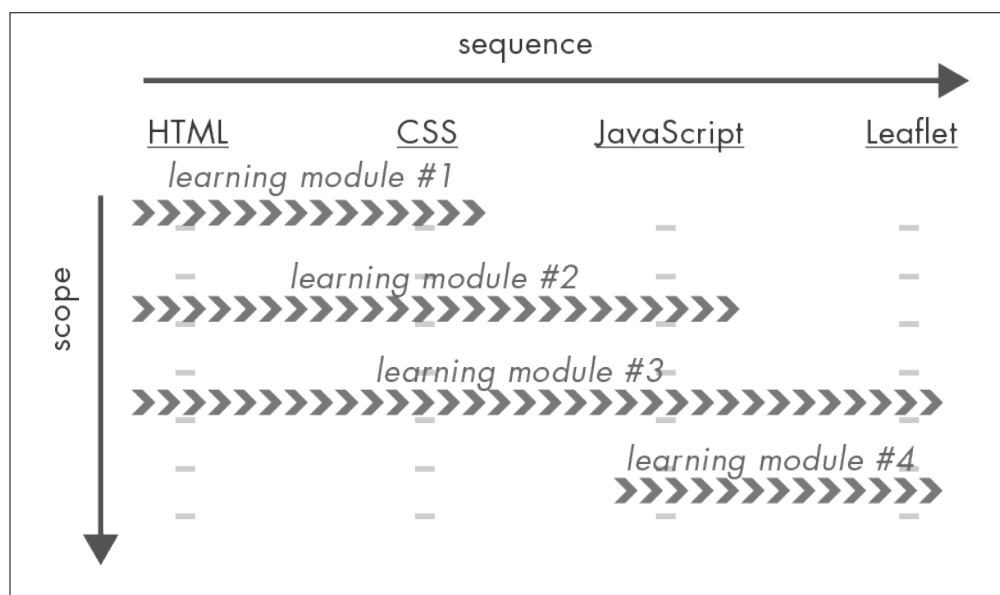
In this example, once the trio of HTML, CSS, and JavaScript have been covered sufficiently, the next iteration of a learning module may introduce a lesson on reading the web mapping library's Application Programming Interface (API) reference, which also was identified as an important competency in the diary study (Figure 5.20). As with the HTML and CSS, a misguided educational approach would be to exhaustively cover each method supported by the API and all its available options. Again this would be too deep of scope. Without situating a usage case in a wider scenario encompassing a sequence (e.g., (1) binding data to Leaflet SVG circles, (2) representing these with variable diameters to encode a data value, (3) adding interactivity to retrieve that value), such an educational strategy fails to approximate a threshold concept overcoming the conceptually confusing practice of employing a specific library's methods in a wider JavaScript program.





**Figure 5.20** Conceptual example of a third successive learning module, now integrating HTML, CSS, and JavaScript into learning a web mapping library's API.

Once foundational concepts are mastered, the reiteration through deeper scope of earlier topics in lab modules may not be necessary. Instead, a new focus upon deeper scope of subsequent topics is needed to move toward the goal of eventually implementing a solution to serve cartographic requirements and user-end goals (Figure 5.21).



**Figure 5.21** Conceptual example of a fourth learning module, departing from further depth of HTML and CSS scope to focus on interaction between JavaScript and the web mapping library's API.

### 5.3.2 Core heuristics guiding web mapping pattern library prototype

As a result of our overall process for keeping pace with emerging web mapping technologies—and the pedagogical model for web mapping described in 5.3.1 specifically—we recommend a set of heuristics for how to build a web mapping design pattern library and began development of a proof-of-concept prototype. *Heuristics* are holistic design goals built from experience. These are informed by consideration of how the various approaches to design patterns and pattern libraries reviewed in Section 5.2 inform our educational strategy and objectives (see Table 5.9). While these approaches all offer specific value for the schematic and technical considerations of the library itself, we underscore the distinctiveness of our need to build a design pattern library specifically as an educational tool, a goal not universally shared among the others (Table 5.7).<sup>47</sup>

#	Need	SE	HCI	Web design & dev
1	scope	too deep, assumes too much	adequate for conceptual understanding, but insufficient for implementation	modularity allows for scope to be approached at appropriate level
2	sequence	not sufficient as point of entry; assumes moderate to advanced knowledge	most likely n/a. interrelated patterns may suggest order, but difficult to apply to educational context	moderate potential when beginning with individual elements of a style guide and moving toward composite interfaces (a la Pattern Lab)
3	misconception	high potential to address misconceptions involving verbose or inefficient code solutions	offers guidance for potentially ineffective UI solutions	high potential to demonstrate proper coding techniques and separation of structure, styles, and behavior involving HTML/CSS/JS
4	threshold	high potential to elevate student coding skills from moderate to advanced	moderate potential for gaining or reinforcing conceptual understanding of UI solutions	moderate potential for understanding relatedness of solution space elements, particularly in instantaneous 'live feedback' context

**Table 5.9** Comparison of web design and development pattern solutions approximating educational concepts applied to web mapping.

The essential recommended heuristics for the development of a web mapping pattern library are as follows, diagrammed in a provisional walkthrough of the sequence and scope to achieve a prototypical web map (Figure 5.22):

<sup>47</sup> In other words, a web mapping design pattern library built strictly for practitioners would look different than the one proposed here.

(1) *Realize the Solution Space*. This research revealed that a fundamental starting place for web development and web mapping is a thorough grounding in HTML, CSS, and JavaScript (i.e., the ‘solution space’). These are fundamental skills that students need before applying the capabilities of a web mapping technology to server higher-order cartographic requirements (i.e., beyond simple ‘dots on a map’ push-pin maps achieve through simpler point-and-click solutions). Chapter 4 described these as the structural, stylistic, and behavior elements that together produce a desired user experience. The diary study covered in Chapter 4 revealed that students are challenged to understand when to edit or modify code in a particular element of these three areas. The design patterns therefore would make this explicit, in two ways. For one, it would show the pre-rendered code of all three aspects of the Open Web Platform. The example of Patternry, reviewed above, is illustrative of this potential. Also, the Pears Wordpress example shows the value of explicitly demonstrating the connection between HTML and CSS code in a single interface.

An example of a basic pattern is one that provides an HTML element for constraining the extent of a rendered web map within the webpage. This pattern is also the jumping off point for most web map development projects; a ‘boilerplate’ for development purposes. The HTML element provides the structure, and an id attribute provides the ‘hook’ for both the CSS and the JavaScript to access it within the DOM. The CSS will specify the width and height of the element’s presentation, as well the margin on the top and bottom and center the element using left and right margin values of ‘auto’. The JavaScript will in turn select the element and run additional routines to draw the map, as the example in Figure 5.22 designates the HTML div element with an id attribute of ‘map.’



**Figure 5.22** Pre-rendered code within the prototype web mapping pattern library showing the three elements of the solution space working together to produce the rendered map (at top).

However, note that not all patterns make use of all three elements of the solution space. For example, an initial AJAX request to load data into the DOM would not need corresponding HTML and CSS code. These can simply be marked as ‘N/A’ in the pattern example:

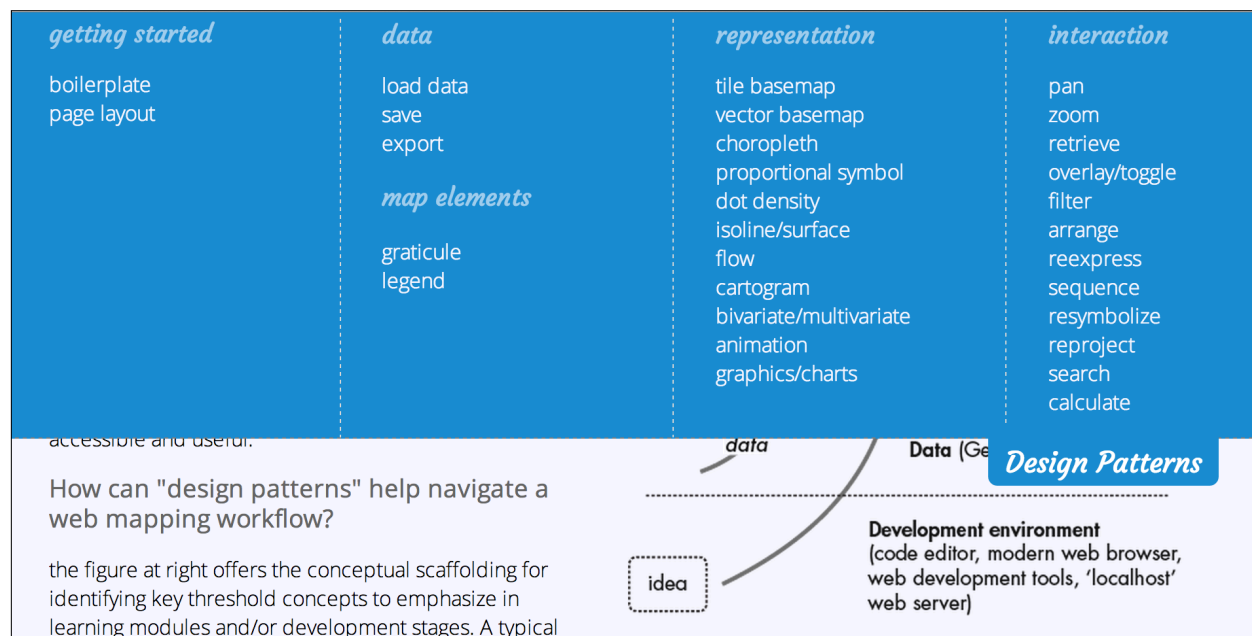


**Figure 5.23** Pre-rendered code within the prototype web mapping pattern library showing the solitary JavaScript code loading in an external data resource with no complimenting CSS or HTML.

(2) *Enablement toward Cartographic Requirements.* Patterns within the web mapping library should fundamentally address the specific needs of Web Cartography. Though some patterns necessarily may demonstrate more general web development solutions, the aim of the library is to approach the cartographic interaction and representation requirements (see Table 4.1). Patterns therefore will need to address two situations: (1) implementing methods and routines that draw upon the mapping library's native (i.e., intended) support for our established cartographic requirements, and (2) implementing custom solutions that either extend the web mapping library's functionality or use novel native JavaScript code to do so. Examples of the former include those we identify in Chapter 3.1 as now constituting the 'prototypical web map,' such as the additions of retrieve and overlay to the more familiar pan and zoom of the slippy map. Examples of the latter include the need to filter data using an UI element or re-expressing the represented data as a different map type. The HCI shift in emphasis away from the SE-

specific approaches to UI-centered solutions employing user-centered design exemplify this (see Table 5.4, #7).

The web mapping pattern library follows directly from both established Web Cartography principles and research findings disclosed through the diary study and exit survey described in Chapter 4. Figure 5.24 illustrates the presentation of design patterns organized as navigational menu items under the principle stages of web map development (i.e., *data->representation->interaction*). Additionally, the library includes supplementary patterns such as ‘getting started’ and ‘map elements’ that are neither representational nor constituting an interaction operator.



**Figure 5.24** The navigation menu of the prototype web mapping pattern library, organizing available design patterns according to ‘getting started,’ ‘data,’ ‘representation,’ ‘interaction,’ and ‘map elements.’

(3) *Edification of Web Cartography Principles.* Though the primary thrust of this dissertation has been to address deficiencies in learning and teaching the development side of a design-development process, the library has the potential to reinforce learning beyond purely technical challenges. Therefore, the library should also make use of the presentation format to describe and explain how specific patterns are used within a more conceptually-informed context of cartographic representation and interaction. This could simply be achieved textually such as in usage notes, modeled after MailChimp’s format (Figure 5.10c), or

with a more explicit structuring similar to SE's encoding of Alexander's rule situating the problem within a context-forces-solution formula.

The example in Figure 5.25 illustrates information pertaining to the retrieve operator as captured within a design pattern, and three variants of solutions are offered that include dynamic labeling, an information window, and an information panel. Not only would this pattern provide example code and demonstrate a solution for achieving these UI requirements, but a user would also garner (or perhaps be reminded of) the relevant conceptual information. In this example, a retrieve operator solution is provided that addresses: (1) the problematic of uncertainty produced through cartographic abstraction, and (2) an information seeking mantra of 'overview first, zoom and filter, then details-on-demand.' Furthermore, the presentation format of the library's individual patterns provide for recommendations and additional advice derived from Web Cartography (i.e., the solution shifts from a dynamic label to an information window as the information being retrieved increases in complexity). Finally, the interrelatedness of patterns can be encoded within each pattern. In this example, a user may link from the retrieve to a compare or sequence pattern when exploring ways support more sophisticated objectives.

*web mapping design pattern library*
Design Patterns

## *retrieve*

**definition:** request details about a map feature of interest

**context**

The **retrieve operator** is a primary way in which users interactively overcome the cartographic problematic (in the context of Interactive Cartography) and complete Shneiderman's information seeking mantra of "overview first, zoom and filter, then details-on-demand" (in the context of Geographic Visualization).

**problem statement**

(1) The map is an abstraction of reality that introduces uncertainty to make a cartographic representation understandable and useful. Greater details are required to mitigate this uncertainty.

(2) Shneiderman's information seeking mantra is "overview first, zoom and filter, then details-on-demand".

**solution**

Information is either bound to an object within the script or HTML element (as when D3.js binds data to a selection), or information is stored in a separate data structure within the script. A user-driven event is required to retrieve information corresponding to a target. The information must then be displayed to the user.

**dynamic label:** retrieve activates and populates a place name label

**information window:** retrieve activates and populates an information container directly atop that map

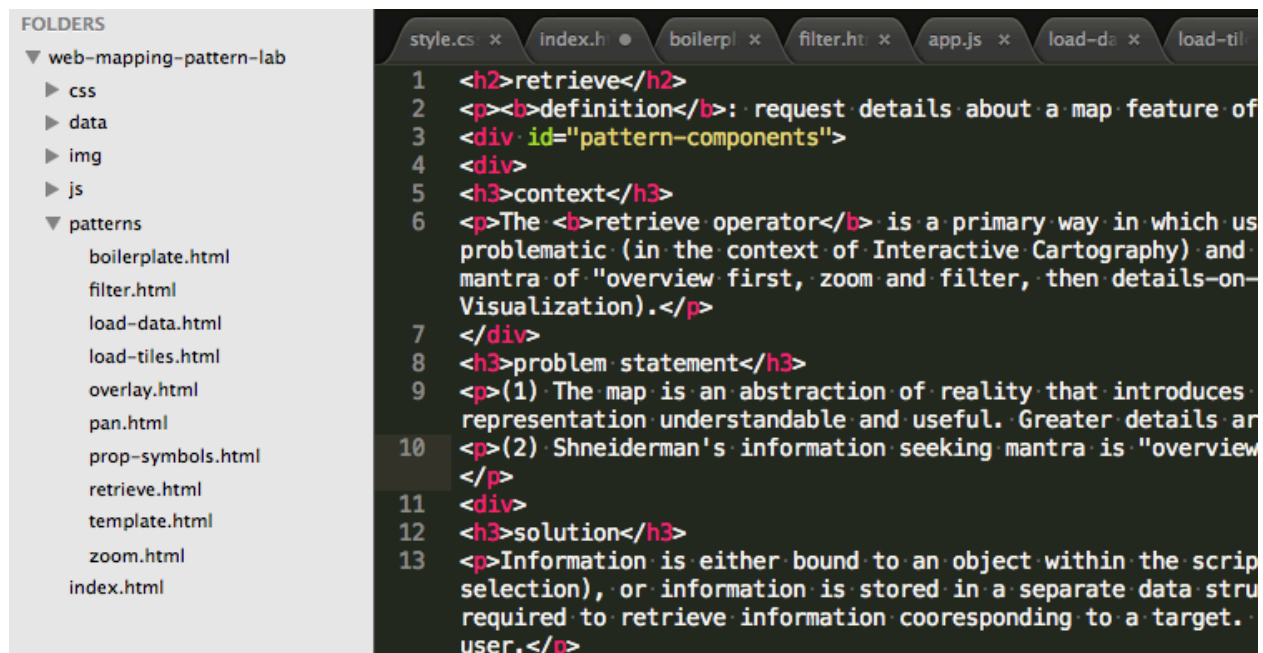
**information panel:** retrieve activates and populates an information container anchored outside of the map

**recommendation:** as the amount and complexity of information provided upon retrieve increases, move from a dynamic label to an information window to an information panel

**recommendation:** while retrieve primarily supports identifying features, consider how you can support more sophisticated objectives such as comparing and sequencing

**Figure 5.25** The pattern components of a design pattern for the retrieve operator. The context, problem statement, and solution provide conceptual knowledge that links specific development solutions with higher-order Web Cartography learning objectives.

(4) *Navigational*. The presentation format of the library should be designed in a way to allow patterns to be accessible through a menu-style navigation interface (Figure 5.24). The GitHub style guide (Figure 5.7) and MailChimp pattern library (Figure 5.10a), for instance, provide an easily accessible menu within a traditional sidebar menu layout. The navigation menu itself also may be usefully constructed as nested under meaningful categories, similar to Tidwell's (1999) web-based library (Figure 5.2c). The example prototype makes use of modern design aesthetics (e.g., clean, minimalist design and modern web fonts to provide attractive typography). The layout making use of entire screen space and employs currently available adaptive and responsive web design techniques. Code highlighting also is used to display pre-rendered code and allow for easy copy/paste procedures by the user. Finally, the architecture of the library itself stores individual patterns as external files and loads them into the library at runtime using AJAX requests (Figure 5.26).



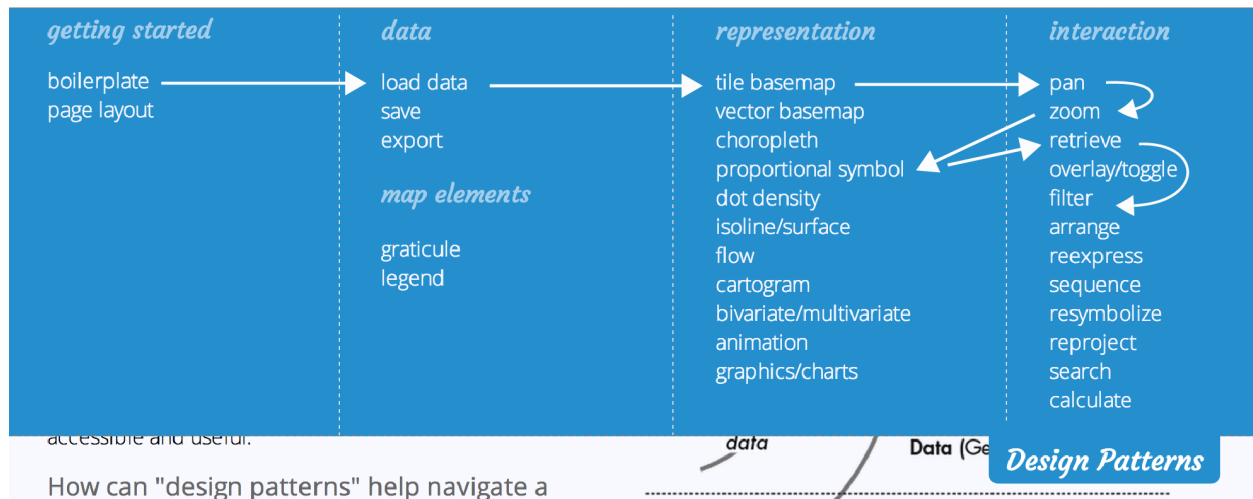
**Figure 5.26** Screenshot of the architectural structure of the prototype web mapping pattern library. Individual patterns are stored as independent HTML files and loaded into the library.

(5) *Relational*. Closely connected with the modularity principle (below) is the heuristic that patterns relate to one another, and for the library to help facilitate this understanding. This is achieved in two senses: (1)



hierarchical relations between patterns and (2) sequential ordering of patterns to achieve a specific web mapping learning objective. Patterns related by hierarchy help students see how specific UI elements such as a slider element are constructed from smaller units of coded solutions, yet fit into a large pattern that associate user interaction with that UI slider to changes in map representation and associated output in coordinated information graphics (e.g., ‘coordinated visualization’). Such hierarchical relatedness is best exemplified best by Frost’s ‘atomic’ approach and his Pattern Lab, but is also diagrammed within the web mapping library in a similar way as Welie *et al.*’s (2003) hierarchical network of design problems (Figure 5.3).

The prototype web mapping pattern library relates pattern to each other in a number of ways. As indicated above under the third heuristic, individual patterns can easily link to and relate with other patterns through the textual ‘usage’ notes. This may also be useful for achieving a hierarchical relationship among patterns (e.g., a graticule pattern solution may fit within a higher-level vector basemap pattern). A novel feature of the prototype pattern, however, is the addition of sequential suggestions for moving through the patterns (Figure 5.27). Again, the navigation menu acts as an organizing visual aid. Within a given mapping scenario, a student can refer to arrows guiding them through various patterns (which may also compliment a linear tutorial).



**Figure 5.27** Conceptual model of a web mapping workflow to create a web map with a tiled ‘slippy’ basemap with proportional symbols drawn atop that support a dynamic label and filter functionality.

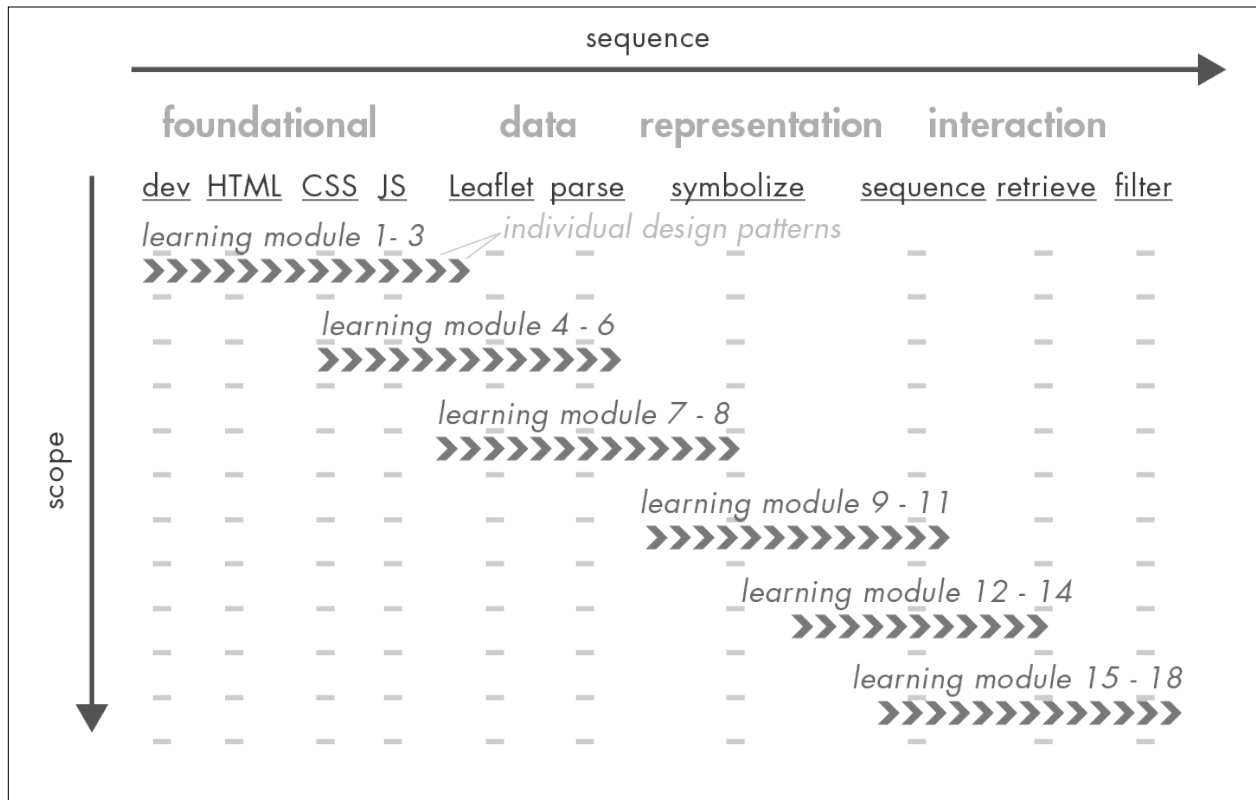
### 5.3.4. Conclusion

This chapter introduced design patterns and pattern libraries as common solutions to recurring design problems. First identified by architect and urban planner Christopher Alexander (1977), design patterns informed computing fields of Software Engineering, HCI, and web design/development. While Alexander proscribed a specific structure for encoding patterns within a library, the design patterns have manifested in various forms and structures over the ensuing 35 years. Within Software Engineering, the work of Gamma *et al.* (1995) targeted professional computer programmers and closely followed Alexander’s structure and use of the tripartite relations between context, the forces constituting the problem to be resolved, and the solution that resolves these forces. HCI departed from Gamma *et al.*’s approach to prescribe interaction solutions that focused on front-end design problems, eventually eschewing the context-forces-solution requirement and presenting solutions in terms of when and how the patterns are best applied. HCI also largely shifted the focus away from the computational routines of desktop software applications to web user interface solutions. The more general field of web design and development have utilized a wide variety of approaches that resemble design patterns to various degrees. These include style guides, front-end frameworks, templates, boilerplates, and pattern collections. While some of these approaches yield the benefit of directly addressing UI problems, like HCI, web design and development

solutions extend the merely visual and conceptual solutions posited by HCI to directly include development solutions making use of the Open Web Platform itself. The proposed web mapping pattern library draws from Alexander and these three fields to offer an approach uniquely suited to the design and pedagogical needs of Web Cartography, while continuing to answer the research questions guiding this dissertation research.

*(3) How should web mapping be taught in higher education?*

A web mapping design pattern library can improve upon Web Cartography education in a number of ways. The heuristics defined and demonstrated above emphasize that design patterns can help create more explicit connections between more conceptual lecture material and the application of those principles to specific web mapping exercises, tutorials, and final projects. Figure 5.28 illustrates a hypothetical curriculum using the web mapping design pattern library approach. Each of the dark arrows moving forward through the modules represents a single design pattern. Initial exercises in modules 1–3 use patterns demonstrating the establishment of the development environment (noted in Figure 5.28 as ‘dev’, which would constitute the directory structure and files, web browser with accompanying web development and console tools, the DOM, and a localhost server). As students gain competency in the fundamentals of HTML, CSS, and JavaScript, these skills are reinforced through use of the Leaflet library, as well as AJAX requests to load external data and appropriately parse within the script (modules 4–6). Modules 7–8 begin to approach the cartographic requirements. These patterns help guide students to choose an appropriate representation given the data (e.g., a proportional symbol for discrete, abrupt phenomena). The remaining modules help the student apply UI elements for sequencing through the data and adding functionality for retrieving specific data values.



**Figure 5.28** Conceptual walkthrough of a sequence of learning modules approximating a laboratory assignment in an advanced-level web mapping course.

*(4) How can we better cope with continued evolution in web mapping technologies?*

The web mapping design pattern library also contributes to adapting to technological change in the service of Web Cartography and the ongoing learning required to remain abreast of changes in the web more generally. Through continued maintenance, the example code within the library for particular solutions will need to be updated to reflect advancements in particular web mapping libraries or APIs. As patterns are refined over time—and solutions are vetted collaboratively through their sharing both within the Web Cartography classes and the broader public web mapping community—the library acts as a mechanism for further monitoring cutting-edge mapping technologies and integrating them into an existing curriculum emphasizing good cartographic design. In a similar way as Debenham (2013) described a style guide as a ‘living document’ that grows alongside a particular web product, a web

mapping pattern library will continue to grow as designers and developers experiment with new ways to create beautiful and technologically sophisticated maps and geographic visualizations.

## Chapter Six: Conclusion

### Executive Summary and Future Research Directions

---

#### 6.1 Overview of findings to date

This dissertation bridged the conceptual knowledge of Web Cartography with the technical practice of building a web map. Effective instruction within our advanced-level cartography courses required a new assessment of the opportunities and challenges rising with a recent shift in web technologies from stand-alone, proprietary-based solutions to those harnessing the Open Web Platform. Beyond gaining a thorough understanding of the available options, we also needed to reexamine the practice of web map development itself. Therein, we exposed misconceptions and troublesome knowledge inherent in our own assumptions and experience. Our educational focus on established principles for effective cartographic representation and interaction—coupled with the pragmatic need to encode instructional material in tutorials that can persist through the turnover of graduate student laboratory instructors—had created an over-reliance on one technology. At the time, we were not equipped to respond to the sudden transformation in mapping technology, nor the likely continual change across the web itself. To ameliorate this, we employed a four-stage process aiming to answer four guiding questions.

#### *#1. What technologies currently are available for web mapping and how do they vary?*

We gained knowledge of available web mapping options and how they varied through a competitive analysis of 35 web mapping technologies. While some of these options focused primarily on a particular cartographic need, such as rendering tiles, others attempt to be catch-all ‘full-stack’ solutions. Interpretation of these results disclosed the potential of these options to meet specific instructional needs within our advanced-level courses, as well as insight into the changing nature of web maps as constituted by the emerging technologies themselves. The surveyed technologies predominately provided native support for tiled basemaps and vector overlays, and choropleth and proportional symbol maps to a slightly lesser extent. However, the majority was ill equipped to produce a broad range of symbolized map types representing the full spectrum of mapped geographic phenomena. Additionally, we identified

widely supported interaction functionality—those operators constituting a ‘prototypical web map’—that extended beyond the now archetypal pan/zoom functionality of the ‘slippy’ map to include retrieve and overlay operators. However, these mapping technologies offered little native support for such cartographic interaction operators as arrange, reexpress, sequence, and resymbolize. Therefore, we recognized the cartographic requirements that we value as academic cartographers that are either unknown or simply unrealized within these products. However, one advantage of these open-source alternatives over proprietary software was the potential to extend the code libraries to support a given representation technique, such as to produce an isoline or dot density map. In a similar way, a library could be extended or hacked to support interaction operators such as filter or calculate. This meant that it was still possible to achieve non-natively supported functionality, but such solutions required greater technical expertise. The majority of the surveyed technologies also catered to an ethic of open source development and harnessed open web standards supported by modern web browsers. This meant that a thriving community of users and developers offering support and continued improvement upon these options was of potential utility for further application of these solutions into as of yet unmet cartographic requirements.

## *#2 What are the important characteristics of web maps that should inform the selection of web mapping technologies?*

A second study—administered as an online survey—sought to learn about the processes and solutions used by the local digital mapping community comprised of twenty-six University of Wisconsin campuses, as well as how these practices related to the technologies we analyzed within the competitive analysis study. Overwhelmingly, participants had not used the majority of technologies collected in the competitive analysis study, nor were they even aware of most of them. Rather, participants were familiar with and used the ESRI products ArcServer, Adobe Flash, Google Maps API, and OpenLayers. Despite their familiarity with the emerging options however, survey participants acknowledged the growing importance of JavaScript-based mapping solutions working in concert with the Open Web Platform. Survey results also garnered information as to what participants felt the important qualities of web maps

to be, organized by design characteristics, technical considerations, and practical considerations. While interaction and aesthetic design choices were important to participants, the survey corroborated a diminishing interest in animation techniques in web mapping. The capacity for a map to update in response to real time data was also of limited priority for these practitioners, despite the web's 'big data' explosion accessible through APIs. Platform dependence was the most important technical consideration, followed closely by browser compatibility. Additionally, maintenance and stability of the technology was rated as the most important practical consideration when selecting a new web mapping technology. Finally, when asked about their approaches for keeping pace with web mapping technological change, survey participants emphasized the value of experimenting with potential options through a pilot study or proof-of-concept prototype. The triangulated analysis between the competitive analysis study and the online survey helped us determine four viable candidate mapping technologies—OpenLayers, Leaflet, D3, and the Google Maps API—from which to follow with a pilot study of our own.

### *#3 How should web mapping be taught in higher education?*

We continued the process with two more studies aiming to more directly address the question how we would teach web mapping within higher education: (1) an in-depth diary case study of the four selected candidate technologies, and (2) an exit survey. The scenario-based diary study invited student participants to test each candidate web mapping technology through 40 hours of development. Participants documented their experiences and captured salient challenges, breakthroughs, emotional states, and useful information pertaining to the specific technology and the wider process of web map development on the Open Web Platform. The diary study constraints, while at times frustrating for participants, helped enable these novice web mappers to approach not only a new web mapping technology, but to clarify the contours of an unfamiliar development environment and workflow. Upon completion of the 40 hours, each participant also completed a comprehensive exit survey seeking to elicit feedback on the individual web mapping technology, the wider development process, and the utility of the diary study itself. From the analysis of the diary study and the exit survey, we were able to not only narrow the four candidate technologies to two—the Leaflet and D3 libraries—but we were also able to



identify the full range of specific technologies and technical competencies needed to successfully make a modern web map. The requisite skills extended beyond gaining familiarity with an individual web mapping library's or API's methods to encompass the wider development practices shared with professional web designers. Yet importantly, just as the end product of the web map differs from that of other web sites or applications, so too does the process of developing a web map. The diary study and exit survey helped establish a development sequence of *data->representation->interaction* that in turn informs how students should proceed through a tutorial or learning exercise. Additionally, the studies underscored the importance of the first stage of data preparation for establishing useful data structures within an executed script. Reaching the goal of extending these mapping libraries beyond their native capacities to further support additional cartographic requirements requires both mastery of the full scope of this development workflow and deep understanding of manipulating the solution space comprised of HTML/SVG, CSS, and JavaScript. The studies additionally accentuated the importance of debugging strategies and using the online community of product users, designers, and developers as a resource for gaining help and support. The final two studies of the process helped solidify a foundation from which lab exercises and tutorial assignments informed a highly successful Spring 2013 course in interactive mapping and Geovisualization.

Despite these wins, the question of teaching the development side of web mapping in the service of good cartographic design remains challenging. The array of skills needed to navigate the web mapping development workflow is daunting enough, and a highly varied range of student experience and skill levels makes effective instruction difficult to predict. Foundational exercises and detailed, explicit tutorials can be complimented with a web mapping design pattern library. Design patterns and pattern libraries, which capture and make accessible common solutions to recurring design problems, have served a variety of purposes across various computing fields. A systematic review and analysis of their treatment, when coupled with our Web Cartography education objectives, offered a set of heuristics for continued development of a library prototype. Importantly, the proposed pattern library offered design

patterns that clarify both the preferred sequence of learning objectives and an adequate scope for redressing gaps within laboratory instruction and students' varied abilities. Such a strategy helps make single out-of-context examples more relevant, captures cumulative knowledge gained through student and teacher learning, and when hosted on a distributed version control platform such as GitHub, shares the excellence of the UW Cartography program with the wider web community.

#### *#4 How can we better cope with continued evolution in web mapping technologies?*

This dissertation began with a challenge of coping with change in web technologies and respond to an abundance of emerging web mapping solutions. To answer that challenge, we completed four studies that compose a process aiming to meet both short and long-term goals. The preceding chapters detail the short term goals of (1) identifying and evaluating viable web mapping technologies according to our curricular needs, (2) comparing these analyses against professional practice, and (3) experimenting with a subset of these technologies to further determine their potential within our web mapping workflow. Alternatively put, these studies approached the challenge of web map development, clarified how development serves good cartographic design, and provided a way toward exceptional teaching. The long term goal, however, was to establish a repeatable process to periodically update our curriculum and that may also find purchase outside of our specific educational context. Minimally, meeting our short-term goals extended our own understanding and vocabulary for approaching current changes in the web. While the level of detail and extensiveness of our studies may appear to be too much to easily repeat, it is crucial to acknowledge that these intensities are variable. Our three stage process—over-simplified as (1) identify and weigh options, (2) gather information from people familiar with these options or like options, and (3) test and further evaluate a subset of options—can be applied to different degrees. This consideration of the proposed process warrants articulation of an ongoing research agenda to both improve the process and extend the scholarship in new directions.

## **6.2 Future Directions**

One component of a future research agenda involves a continuation of and improvement over the existing process described within this dissertation. The competitive analysis study was a single snapshot in time

that, while greatly informing our process through this iteration, unfortunately was dated the moment it was completed. Minimally, it should be updated within the next two years, ideally repeating the analysis with multiple coders. Future research would introduce a web-based survey mechanism to invite wider participation (‘i.e., crowd source’) in both suggesting new web mapping technologies, as well as updating already catalogued options as their feature support improves. As new mapping solutions are introduced, or current ones extended, participants could update the data the resultant matrix visualization. Given the insight produced in terms of the supported capabilities of available options, and the broader ontological answers to what a web map is, this component is of high priority for continuing the process established within the research. Coupling with the survey of participant practitioners, as in our established survey, would strengthen the analysis.

The diary study was the most valuable and informative part of the process. This was largely due to our own internal shift from the Flash-based IDE and plug-in produced workflow to the new technical medium. For the time being, the web continues to evolve but is static in making use of the Open Web Platform. Presumably, a second diary study initiated today would inform participants of what we’ve learned so far, and even incorporate the pattern library as a learning tool. Less time would be spent conceptualizing and formatting to the JSON data structure, discovering web development tools for debugging, and learning fundamental HTML, CSS, and JavaScript. These now can be anticipated and instructed prior to the initiation of this stage of the process. A second diary study therefore would focus more upon the capabilities of existing web mapping technologies to support those representational and interaction solutions not natively supported. Additionally, the further development of the pattern library would be enrolled within a diary study as well. While the further development of the pattern library—as well as empirically testing it out in learning environments—is no small task that will likely consume a two year timeline, its integration into a modified diary study is a likely direction for a five year timeline (i.e., the scope of a tenure timeline).

A final component for future research is to engage with scholarship emerging from Curriculum and Instruction (C&I), and in particular consider ways in which the lessons we've learned in classroom teaching can be transferred into an online education program. While GIS&T instructors have taken great care in reflecting upon pedagogical and curricular—most notably through the *Body of Knowledge* and the notions of misconceptions, threshold concepts, scope, and sequence from Foote and Bampton operationalized within this dissertation—additional insight into effective teaching strategies and how students learn can greatly inform the continued growth of web mapping education. A quick win will be to contribute toward an updated version of the *BoK*. Yet as educators we need to work against a 'deficit model' in instruction that looks toward the student and his or her lack of ability or mastery of route knowledge as the defining feature of a pedagogical strategy (Harry, Beth, and Janette Klingner, 2007). Instead, as stated by Vigotsky (1980), future research should explore how students will learn best when pushed to their limit, yet through careful attentiveness to the 'zone of proximal development,' or the difference between student learning gains when working without help and with help. C&I research suggests that providing good instructional 'scaffolding' can provide such help. The application of the web mapping pattern library to teaching is an explored research area worthy of more attention. C&I educators emphasize the importance of 'bracketing' the unintended learning objectives while pushing students to focus on a specific skill for understanding. Again, the reflection upon scope and sequence discussed within the dissertation has already made an important step forward in this regard. The pattern library can be explored as a practical application of such C&I conceptual contributions.

## Appendix A: Protocol for Online Needs Assessment Survey

### Biographic Information

1. Please provide 4-6 keywords (multiple word phrases are OK) that describe your current work responsibilities.

<free response>

2. Please mark the category that best describes your affiliation with the UW System (choose one):

- Undergraduate Student
- Graduate Student
- Staff
- Faculty
- Not Affiliated
- Other: <describe>

3. How frequently do you assemble and/or manipulate geographic information in your own daily work (choose one)?

- daily
- weekly
- monthly
- yearly
- never
- I supervise this activity, but do not regularly complete it myself

How frequently do you design print maps in your daily work (choose one)?

- daily
- weekly
- monthly
- yearly
- never
- I supervise this activity, but do not regularly complete it myself

How frequently do you design/develop web maps in your daily work (choose one)?

- daily
- weekly
- monthly
- yearly

- never
- I supervise this activity, but do not regularly complete it myself

### Use of Existing Web Mapping Technologies

1. Please rate your design/development team's (i.e., you or the project members you supervise) engagement with the following web mapping technologies; please do not search for these technologies in your browsers while completing the survey:

Web Mapping Technology	I have <u>not</u> heard of this technology.	I <u>am</u> aware of this technology, but have <u>not</u> used it for any projects yet.	I have used this technology in the past, but it was <u>more than</u> one year ago.	I have used this technology <u>within</u> the past year.
ArcServer				
Adobe Flash				
Adobe Flex				
Bing Maps API				
CartoDB				
CartoWeb				
Cloudmade Map Style Editor				
d3				
deCarta				
GeoEXT				
GeoMoose				
Google Maps API				
Jump				
Ka-Map				
Kartograph				
Leaflet				
MapBender				
MapBuilder				
Mapnik				
MapQuery				
MapQuest API				
MapServer				
Mapstraction				
Modest Maps				
NearMap				
OpenLayers				
OpenScales				

			R=4-ab8e 54908	
Ovi Maps API (formerly Yahoo!)				
Polymaps				
Processing				
Processing.js				
Raphaël				
ReadyMap				
Tiledrawer				
TileMill				
Tilestache				
TimeMap				
ViaMichelin				
WorldKit				
Wax				
Other? (optional)				
Other? (optional)				
Other? (optional)				

2. Among the above technologies used by your design/development team, which are most commonly leveraged in your projects? What aspects of these technologies make them particularly useful in your team's work?

<free response>

3. Among the above technologies used by your design/development team, which have been abandoned completely? What aspects of these technologies led your team to abandon them?

<free response>

## Qualities of Web Mapping Technologies

1. Rate the following characteristics of web maps, as they relate to your team's design/development priorities:

Characteristics	not important	somewhat important	important	very important	essential
<b>adaptability:</b> change in the map to respond to the use and user context,					

such as user location or user profile					
<b>animation</b> : dynamic use of display time to represent real-world time					
<b>interactivity</b> : change in the map display to respond to user requests/manipulations					
<b>multiscale</b> : display of integrated map designs of varying abstractions/resolutions when change map scale (i.e., zooming)					
<b>real-time</b> : dynamically load information to represent current conditions					
<b>scalability</b> : load, represent, and interact with large datasets without system response delays					
<b>cartographic design aesthetics</b> : ability to customize the look and feel of the map itself					
<b>interface design aesthetics</b> : ability to customize the look and feel of the user interface to the map					

**2. Rate the following technical considerations of web maps, as they relate to your team's design/development priorities:**

Technical Consideration	not important	somewhat important	important	very important	essential
<b>browser compatibility</b> : the technology works across browsers					
<b>location aware</b> : the technology can leverage the user's location					
<b>mobile support</b> : the technology works on mobile devices					
<b>platform dependency</b> : the technology works across operating systems					
<b>reliance on a plug-in</b> : the technology requires a browser plug-in or installation of an executable					



**3. Rate the practical considerations of web maps, as they relate to your team's design/development priorities:**

Practical Considerations	not important	somewhat important	important	very important	essential
<b>access/cost:</b> open source versus commercial technologies, and the cost of the technology if commercial					
<b>code documentation:</b> a complete description of functionality provided by the technology					
<b>maintenance:</b> long-term stability of the technology (e.g., regularity of updates, significance of updates, handling of deprecation)					
<b>support:</b> contact support from staffed individuals or a user community (e.g., email inquiries, FAQs, forums)					
<b>tutorials/examples:</b> descriptions or demonstrations of how to implement the technology					

**4. Are there any additional technical or practical considerations of web maps not listed above that are important in your team's design/development priorities?**

## Outlook on Web Mapping Technologies

1. How does your design/development team experiment with new web mapping technologies before selecting them for a project?

<free response>

2. What information do you need to know about a newly released web mapping technology before selecting it for a project?

<free response>

3. What do you find to be the critical barriers or constraints to learning a new web mapping technology?

<free response>

4. What design or development skill sets regarding web mapping technologies would you look for in a new hire (e.g., a recently graduate student) joining your design/development team?

<free response>

5. Do you have additional thoughts or ideas related to web mapping technologies that you would like to share with us?

<free response>

## Appendix B: Protocol for Diary Study

---

### About and Consent

You are invited to participate in a research study about emergent web-based mapping technologies.

**Web mapping** is a broad term used to describe the diverse suite of APIs, code libraries, and coding frameworks that facilitate design and development of dynamic maps served online. Your input will be used *specifically* to inform the UW-Madison Cartography curriculum and UW-Madison Cartography Lab web mapping projects as well as *broadly* to generate guidelines for the design and development of web maps.

The primary goal of the research in which you are participating is to test a process by which a web mapping design/development team can keep pace with evolving web mapping technologies. Design of the process was constrained to match the anticipated resources available to a small-sized design/development team during an average work week. In other words, it is hoped that this process can be applied efficiently and effectively to identify an appropriate technology for a new web mapping project.

The process itself draws from social science methods adapted for the field of Usability Engineering and includes four integrated stages of empirical input and feedback: (1) an initial competitive analysis study of contemporary web mapping technologies to define and organize the contemporary web mapping solution space, (2) an internal survey of past experiences with and opinions of these technologies, (3) a diary study charting the implementation of a subset of viable technologies, identified from the first two stages, and (4) a closing focus group study reflecting on successes and frustrations during implementation of these candidate technologies. The first two stages have been completed and will be summarized in the following section. Today, you will be introduced to the procedure of the diary study, an adaptation to participant observation that requires participants to “self-observe” by composing diary entries about their experiences. By participating in the diary study, you are required to also participate in the closing focus group study.

## Procedure

### Technology Designation

The first two stages of input/feedback (the competitive analysis and internal survey) identified a subset of viable technologies that could meet the web mapping needs of the UW-Madison Cartography curriculum and UW-Madison Cartography Lab projects [*summary of first two stages provided here*]. Each of you will be assigned a single candidate technology for use in the diary study; to adjust for individual skill levels, one participant will complete the study using all of the candidate technologies.

It is important that you constrain your design to the technology listed. You are encouraged to leverage online documentation and examples, but **do not implement examples that require use of multiple technologies**. Instead, list such interoperability/integration opportunities in your diary entry (see below), describing how the use of the additional technology would have improved the effectiveness of your web map or your efficiency in developing it. Contact the principle investigator about any ambiguities about overlapping technologies.

### Scenario & Requirements

A scenario is provided in Section 3 of the protocol that represents a prototypical mapping challenge. The scenario is accompanied with a detailed requirements document outlining the features that must be included in the final web map. You have **40 hours in total** to work through as many of the requirements as possible. While you should attempt to accomplish all requirements, one or several requirements may not be supported by your assigned web mapping technology or may not be feasible given the time constraint of 40 hours. Therefore, one goal of the diary study is to capture your successes, failures, breakthroughs, frustrations, etc., as you work through the requirements, particularly providing an explanation as to why some requirements were difficult to accomplish or could not be accomplished at all. Thus, try to meet all requirements, but start with the ones perceived to be easiest and work towards those perceived to be most difficult. In doing so, however, be sure to consider the scenario itself and to design for it. Also, it is recommended to reserve the final 2-3 hours (or more if time allows) for debugging (i.e., stabilizing your code) and skinning (i.e., improving the design aesthetics).

### Initial Training

All candidate technologies use the JavaScript programming language. To ensure that all participants begin with a baseline understanding of JavaScript, you **must** complete the online *JavaScript Essential Training* Lynda tutorial series (available at: <http://www.lynda.com/JavaScript-tutorials/Essential-Training-2011/81266-2.html>). Please complete these tutorials prior to investigating your assigned technology. You also will be asked to complete a background survey prior to the start of the diary study to assess your knowledge about JavaScript development and web mapping.

You are welcomed to ask the principle investigator questions regarding the research design at any time; you may find logical errors in the information set provided for the scenario, have questions about Web Cartography or JavaScript generally, or have clarifications in the procedure itself. However, **do not ask questions regarding how a listed requirement can be implemented in your assigned technology**; one

goal of the diary study is to assess the quality of the learning support structure for each candidate technology (i.e., how useful are the available code documentation, tutorials, examples, etc.).

You may work on your own computer and use the development environment with which you are most comfortable. We will provide a computer and/or recommend a development environment if necessary. Following a rapid prototyping model, **begin development as soon as possible**, rather than first vetting a series of static mockups.

### Diary Entries

You are required to submit a diary entry after each hour of design/development, resulting in 40 entries in total. Please **plan to work in one-hour increments** to allow for homogenous and immediate diary entries. An analog timer will be provided to assist with unitizing your work time. You may split the 40 hours of work across the study period in the manner that best fits your schedule. **It is critical that you “stay on task” during each of these hours of work.**

#### The following tasks count towards your 40 hour limit:

- review of learning materials (code documentation, examples, tutorials) related to your assigned web mapping technology
- information processing (you may find that your technology requires or benefits from a specific format)
- web map design and development
- debugging, troubleshooting, etc.

#### The following tasks do not count towards your 40 hour limit:

- the opening Lynda tutorials
- the background and closing surveys
- questions/discussion regarding the research design (issues with provided information, questions about Web Cartography or JavaScript generally, clarifications in the procedure)
- configuring the development environment (e.g., download of library stacks, acquisition/installation of software)
- composition and submission of diary entries
- breaks between hour-long sessions

Diary entries will be entered through an online Google spreadsheet shared with the principle investigator. A set of closed- and open-ended questions (Section 4) are included as columns in the Google spreadsheet to structure each diary entry and ultimately speed their composition; please familiarize yourself with these questions prior to beginning design/development. **Be as comprehensive as possible in your diary responses**, as these entries will be used as a primary source in subsequent analysis and reporting; a summary of your diaries will be provided to inform the follow-up focus group study. Please save a local copy of your diary at the end of each work day to avoid loss of your entries. You also are required to capture a screenshot of your map and save a duplicate copy of your source code with every entry.

You will be asked to complete a closing survey at the end of your 40 hours of design/development.

## Web Map Scenario & Requirements Document

### Web Map Scenario

You have been contracted by National Public Radio (NPR) to create a web map in support of a *Talk of the Nation: Science Friday* (<http://sciencefriday.com/>) discussion on renewable energy (<http://sciencefriday.com/topics/energy.html>). This particular installment will include input from a variety of experts on renewable energy science and technology, with a focus of reviewing past trends in renewables and potential future opportunities. In particular, the discussants will compare the use of renewables in the United States—which has remained consistently low over the past 40 years—to the strategies and trends exhibited by other nations. The web map therefore will support two goals: (1) reveal new insights into the patterns, trends, anomalies, etc., of international reliance on renewables and (2) provide a web supplement for listeners to explore while listening to the installment online. As a result, the map should include a large number of interactions to allow for free exploration, but remain focused on a single information set (renewables by percentage of energy consumed).

### Information Sets

You are limited to the following information sets:

- Thematic Information:
  - Organization for Economic Co-Operation and Development (OECD) Factbook 2011-2012: Economic, Environment, and Social Statistics ([http://www.oecd-ilibrary.org/economics/oecd-factbook-2011-2012\\_factbook-2011-en](http://www.oecd-ilibrary.org/economics/oecd-factbook-2011-2012_factbook-2011-en))
  - The specific variable you will be mapping is described as “Renewable Energy as a Percentage of Energy Consumed” and includes yearly information points from 1971-2010 (40 years)
  - Includes a total of 40 nations:
    - 34 participating OECD nations (considered “developed”)
    - 6 non-participating or “developing” nations (Brazil, China, India, Indonesia, Russia, and South Africa)
    - 3 former-Soviet nations have information available starting in 1990 only (Russia, Estonia, Slovenia)
    - Many nations have yet to report for 2010
- Basemap Information:
  - Natural Earth (<http://www.naturalearthdata.com>)

## Requirements

### Representation:

- map types*: support each of the following map types; follow all associated conventions
  - ☐ classed choropleth (visual variable = color hue+value)
  - ☐ graduated symbol (visual variable = size)
- ☐ *animation*: animate the map over the included time periods
- ☐ *typography*: label map features following typographic conventions; labels may be suppressed at the global zoom level
- ☐ *classification*: use an equal interval classification scheme for both the choropleth and graduated symbol map
- ☐ *legend*: dynamically (re)draw the map legend to match the displayed map type
- ☐ *highlighting*: include a highlighted variant of each map feature to indicate selection
- ☐ *information graphic*: include a line graph showing the signature of a selected country in comparison to the United States and the median value for the year
- ☐ *visual hierarchy*: style the basemap to produce a strong visual hierarchy
- ☐ *storytelling*: provide a title and supplementary text to introduce the subject/purpose
- ☐ *cartographic design aesthetics*: customize the look and feel of the map itself to fit the scenario

### Interaction:

- ☐ *reexpress*: change the displayed map type between choropleth and graduated symbol
- ☐ *sequence*: include standard VCR controls (play, stop, step, back) to control the animation
- ☐ *resymbolize*: change the number of classes used for the choropleth or graduated symbol map; allow the user to range from 3 to 20
- ☐ *overlay*: toggle between a vector map and aerial image / shaded relief basemap
- ☐ *reproject*: set the map projection to an equal area
- ☐ *pan*: change the geographic center of the map
- ☐ *zoom*: change the scale and resolution (of labels) of the map
- ☐ *filter*: filter the map according to the attribute range using a two-thumb slider; matching map features should become highlighted
- ☐ *search*: allow the user to search for a specific country; matching map features should become highlighted
- ☐ *retrieve*: highlight a probed map feature and activate an associated information window with details about the feature
- ☐ *calculate*: dynamically calculate deviation of a map feature from median (i.e., percentile) and include in the information window upon *retrieve*
- ☐ *link*: coordinate *retrieve* with the line graph to show the selected map feature in context with the United States and the median
- ☐ *interface design aesthetics*: customize the look and feel of the interface to the map to fit the scenario

## Diary Questions

1. What is your current hour of work (respond 1-40)?
2. What is the current date and time?
3. In your estimation, approximately how complete is your web mapping application in its current form (reply with an estimated percentage)?
4. In the last hour, on which features from the requirements document were you working (copy and paste from the requirements document)?
5. In the last hour, which features from the requirements document did you accomplish, if any (copy and paste from the requirements document)?
6. Describe any important breakthroughs or other useful insights regarding your assigned web mapping technology that were generated during the last hour of work.
7. Describe any major frustrations or key problems/concerns regarding your assigned web mapping technology that arose during the last hour of work.
8. Which of the following term or terms best describes your current mood with regards to your web mapping application (bold the most applicable):

Accepted	Chipper	Drunk
Accomplished	Cold	Ecstatic
Aggravated	Complacent	Energetic
Alone	Confused	Enraged
Amused	Content	Enthralled
Angry	Cranky	Envious
Annoyed	Crappy	Exanimate
Anxious	Crazy	Excited
Apathetic	Crushed	Exhausted
Ashamed	Curious	Flirty
Awake	Cynical	Frustrated
Bewildered	Dark	Full
Bitchy	Depressed	Geeky
Bittersweet	Determined	Giddy
Blah	Devious	Giggly
Blank	Dirty	Gloomy
Blissful	Disappointed	Good
Bored	Discontent	Grateful
Bouncy	Ditzy	Groggy
Calm	Dorky	Grumpy
Cheerful	Drained	Guilty



Happy  
High  
Hopeful  
Hot  
Hungry  
Hyper  
Impressed  
Indescribable  
Indifferent  
Infuriated  
Irate  
Irritated  
Jealous  
Jubilant  
Lazy  
Lethargic  
Listless  
Lonely  
Loved  
Mad  
Melancholy  
Mellow  
Mischievous  
Moody  
Morose  
Naughty  
Nerdy  
Not Specified  
Numb  
Okay  
Optimistic  
Peaceful  
Pessimistic  
Pissed off  
Pleased  
Predatory  
Quixotic  
Recumbent  
Refreshed  
Rejected  
Rejuvenated  
Relaxed  
Relieved  
Restless  
Rushed  
Sad  
Satisfied  
Shocked

Sick  
Silly  
Sleepy  
Smart  
Stressed  
Surprised  
Sympathetic  
Thankful  
Tired  
Touched  
Uncomfortable  
Weird

9. Do you have additional thoughts or ideas related your web mapping application or your assigned web mapping technology that you would like to note?

10. Please copy and paste a screenshot of the current state of your application and save a duplicate copy of your source code dated according to the current hour.

## Appendix C: Protocol for Exit Survey to the Diary Study

### Instructions

In the following survey, you are asked to respond to a series of primarily open-ended questions regarding your experiences with the UW-Madison Cart Lab web mapping activity. Roughly half of the questions ask about your experiences using your assigned web mapping technology to implement the requirements of the web mapping scenario, while the other half ask you to reflect more deeply on the nature of web mapping, best practices for both learning and doing web mapping, and the diary activity itself as a method for articulating the learning and development process.

A visual summary of your diary experience is presented in an attachment for reference while responding to the exit survey questions. **Please also review your complete diary prior to beginning the exit survey.** Wherever possible, please indicate the diary entry or entries (**identified by hour**) during which you experienced a particular breakthrough, frustration, etc. Please alert the Principle Investigator if you find any mistakes in your original diary entries or in the supplied visual summary of your entries.

The survey will require at least 45 minutes to complete; you are encouraged to be as comprehensive as possible in your responses. Please compose your responses directly following each question, leaving your responses unbolded. You are encouraged to write beyond the space provided if appropriate, allowing the survey length to expand accordingly.

### Reflections on Attaining the Scenario Requirements

The following questions ask you about the requirements of the web mapping scenario as they relate to your final web map. In your answers, please include both characteristics regarding your assigned technology as well as general/conceptual concerns. Please include any anecdotes, opinions, frustrations, recommendations, etc., derived from the diary activity to support your responses.

1. Indicate which features from the requirements document that you feel you accomplished adequately (not necessarily perfectly) by checking the associated box:

#### Representation:

*map types*: support each of the following map types; follow all associated conventions

- ☐ classed choropleth (visual variable = color hue+value)
- ☐ graduated symbol (visual variable = size)
- ☐ *animation*: animate the map over the included time periods
- ☐ *typography*: label map features following typographic conventions; labels may be suppressed at the global zoom level
- ☐ *classification*: use an equal interval classification scheme for both the choropleth and graduated symbol map
- ☐ *legend*: dynamically (re)draw the map legend to match the displayed map type
- ☐ *highlighting*: include a highlighted variant of each map feature to indicate selection
- ☐ *information graphic*: include a line graph showing the signature of a selected country in comparison to the United States and the median value for the year
- ☐ *visual hierarchy*: style the basemap to produce a strong visual hierarchy
- ☐ *storytelling*: provide a title and supplementary text to introduce the subject/purpose

- ☐ *cartographic design aesthetics*: customize the look and feel of the map itself to fit the scenario

**Interaction:**

- ☐ *reexpress*: change the displayed map type between choropleth and graduated symbol
  - ☐ *sequence*: include standard VCR controls (play, stop, step, back) to control the animation
  - ☐ *resymbolize*: change the number of classes used for the choropleth or graduated symbol map; allow the user to range from 3 to 20
  - ☐ *overlay*: toggle between a vector map and aerial image / shaded relief basemap
  - ☐ *reproject*: set the map projection to an equal area
  - ☐ *pan*: change the geographic center of the map
  - ☐ *zoom*: change the scale and resolution (of labels) of the map
  - ☐ *filter*: filter the map according to the attribute range using a two-thumb slider; matching map features should become highlighted
  - ☐ *search*: allow the user to search for a specific country; matching map features should become highlighted
  - ☐ *retrieve*: highlight a probed map feature and activate an associated information window with details about the feature
  - ☐ *calculate*: dynamically calculate deviation of a map feature from median (i.e., percentile) and include in the information window upon *retrieve*
  - ☐ *link*: coordinate *retrieve* with the line graph to show the selected map feature in context with the United States and the median
  - ☐ *interface design aesthetics*: customize the look and feel of the interface to the map to fit the scenario
2. Going into the diary process, which features from the requirements document did you perceive as being particularly easy to implement? Why did you think these would be straightforward?
  3. Going into the diary process, which features from the requirements document did you perceive as being particularly difficult to implement? Why did you think these features would be challenging?
  4. Looking back on your completed map, which features from the requirements document were easier than expected to implement? What made these features so straightforward?
  5. Looking back on your completed map, which features from the requirements document were more difficult than expected to implement? What made these features so challenging?
  6. Based on your experience, can you recommend an order in which a developer should attempt to tackle the features listed in the requirements document? If not a complete order, does it make sense to implement several of these features before the others (or vice versa)?
  7. As part of the scenario, you were given a dataset in a basic CSV format. Did you have to convert the dataset into a different format? If yes, which format and how easy was the conversion process; how could the conversion process be improved?
  8. What do you like about your final web map?
  9. What do you dislike about your final web map?

10. If you were to keep working on your web map, which requirements would you like to add or revise? How much more time do you think you would need to make these changes?

11. How much more time do you think is needed to complete all features in the requirements document. Please provide your best estimate and explain your reasoning.

12. Do you have any further comments about your final web map not shared above?

## Reflections on Your Assigned Technology

The following questions ask you about the ease-of-learning and overall power of your assigned web mapping technology, as you have come to understand it following the diary activity. Please include any anecdotes, opinions, frustrations, recommendations, etc., derived from the diary activity to support your responses.

1. What did you find to be the strengths of your assigned technology? You may bullet your answers, but please be as comprehensive as possible in your listing.

2. For what kinds of web mapping contexts would you recommend using your assigned technology over others? In other words, make an argument for using your assigned technology (i.e., “I would use this if...”).

3. What did you find to be the weaknesses of your assigned technology? You may bullet your answers, but please be as comprehensive as possible in your listing.

4. For what kinds of web mapping contexts would you not recommend using your assigned technology over others? In other words, make an argument against using your assigned technology (i.e., “I would not use this if...”).

5. The experimental design of the diary activity required that you only use your assigned technology; did this restrict your development? If yes, which additional web mapping technologies would you liked to have used in combination with your assigned technology and how would you liked to have used them.

6. Rate the quality of the ‘learning materials’ currently available for your assigned technology by checking the associated box; please leave comments justifying each of your answers, allowing the table to expand accordingly.

Learning Material	Poor: This kind of learning material was not available or was indecipherable	Below Average	Average	Above Average	Excellent: This kind of learning was available and extremely helpful	Comments
<b>code documentation:</b> clear and comprehensive information						

describing the complete set of classes, functions, variables, etc., included in your technology						
<b>support:</b> readily available contact support from individuals directly associated with the project						
<b>wikis:</b> readily available and comprehensive information and feedback from a broader user community of the technology						
<b>tutorials:</b> clear and comprehensive step-by-step instructions for installing and working with the technology						
<b>examples:</b> concise and comprehensive code providing example implementations of functionality included in the technology						
<b>debugging/testing tools:</b> tools and services that help you identify errors in your logic and syntax						

7. Rate the ability of your assigned technology to support the following characteristics of web maps; please leave comments justifying each of your answers, allowing the table to expand accordingly.

Characteristics	Poor: The technology does not support this characteristic	Below Average	Average	Above Average	Excellent: The technology includes both established and novel solutions for supporting this characteristic.	Comments
<b>adaptability:</b> change in the map to respond to the use and user context, such as user location or user profile						
<b>animation:</b> dynamic use of display time to represent real-world time						
<b>interactivity:</b> change in the map display to respond to user requests/manipulations						
<b>multiscale:</b> display of integrated map designs of varying abstractions/resolutions when change map scale (i.e., zooming)						
<b>real-time:</b> dynamically load information to represent current conditions						
<b>scalability:</b> load, represent, and interact with large datasets without system response delays						
<b>cartographic design aesthetics:</b> ability to customize the look and feel of the map itself						
<b>interface design aesthetics:</b> ability to customize the look and feel of the user interface to the map						

8. Do you have any further comments about your assigned web mapping technology not shared above?



## Advice for Future Students

The following questions solicit advice from you to help train future students in the UW-Madison Cartography curriculum, with an emphasis on how web mapping should be taught from a technical perspective. Please include any anecdotes, opinions, frustrations, recommendations, etc., derived from the diary activity or from your past web mapping experience to support your responses.

1. What should students be taught before starting design or development on a web map? In other words, what are you glad you knew—or wish you would have known—prior to starting your web map? This may include both conceptual/theoretical knowledge and technical knowledge that is not specific to a given web mapping technology.
2. What do you think will be the hardest things for students to learn about web mapping when starting with no experience? What were the hardest things for you to learn when you first started web mapping? The two do not need to be the same.
3. Based on your experience, where should students ‘start’ when learning a new web mapping technology?
4. Based on your experience, where should students “start” when implementing a new web map with an already known technology?
5. Can you recommend any ‘best practices’, ‘general guidelines’, or ‘guiding principles’ in web mapping that would help students? Please describe each fully.
6. What tricks, tips, or short cuts would you like to communicate to future students taking on web mapping for the first time? Please describe each fully.
7. What key problems, common mistakes, or time sinks would you like to warn future students about when taking on web mapping for the first time? Please describe each fully.
8. Do you have any further advice regarding web mapping for future students in the UW Cartography curriculum?

## Reflections on the Diary Activity

The following questions solicit feedback on the diary activity as a possible method for understanding how students learn new web mapping technologies as well as for identifying an appropriate technology for a given web mapping context. Please include any anecdotes, opinions, frustrations, recommendations, etc., derived from the diary activity or your past web mapping experience to support your responses.

1. What aspects of the diary activity do you believe helped you to articulate your learning and development progress? In other words, what aspects of the diary activity did you find acceptable, keeping in mind the research-orientation of the project?
2. What aspects of the diary activity do you believe directly hindered your learning and development progress? In other words, what aspects of the diary activity did you find problematic, even given the research-orientation of the project?
3. Rate how well each of the diary questions helped you articulate your learning and development process as you worked on your web map; please leave comments justifying each of your answers, allowing the table to expand accordingly.

Characteristics	Poor: This question should be removed in future applications of the diary activity	Below Average	Average	Above Average	Excellent: I felt most at ease or empowered articulating my progress with this question	Comments
#3. In your estimation, approximately how complete is your web mapping application in its current form (reply with an estimated percentage)?						
#4. In the last hour, on which features from the requirements document were you working (copy and paste from the requirements document)?						
#5. In the last hour, which features from the requirements document did you accomplish, if any						

<i>(copy and paste from the requirements document)?</i>						
<i>6. Describe any important breakthroughs or other useful insights regarding your assigned web mapping technology that were generated during the last hour of work.</i>						
<i>7. Describe any major frustrations or key problems/concerns regarding your assigned web mapping technology that arose during the last hour of work.</i>						
<i>8. Which of the following term or terms best describes your current mood with regards to your web mapping application (bold the most applicable): [see sheet]</i>						
<i>9. Do you have additional thoughts or ideas related your web mapping application or your assigned web mapping technology that you would like to note?</i>						
<i>10. Please copy and paste a screenshot of the current state of your application and save a duplicate copy of your source code dated according to the current hour.</i>						

4. What aspects or topics about your learning and development process did the above set of questions not cover, if any? Could you have articulated these topics better if ask to record diary entries in an unstructured or open-ended format (i.e., without specific questions)?

5. The diary activity spanned 40 hours, with one entry each hour. Do you recommend changing either the total number of hours or the regularity of diary entries? Please justify your answer, emphasizing what would be gained and lost with the change.

6. Do you think the diary activity would be an effective and/or efficient process for identifying an appropriate technology in the private and/or government sectors? If no, can you recommend ways in which the diary activity could be adjusted to increase its effectiveness or efficiency for such 'real-world' applications?

7. Is there anything else you would like to say about the diary activity?

## References

- Alexander, C. (1979). *The timeless way of building* (Vol. 1). Oxford University Press.
- Alexander, C., Ishikawa, S., & Silverstein, M. (1977). *A Pattern Language: Towns, Buildings, Construction* (Center for Environmental Structure Series).
- Bampton, Matthew. (2011). "Addressing Misconceptions, Threshold Concepts, and Troublesome Knowledge in GIScience Education." In *Teaching Geographic Information Science and Technology in Higher Education*, edited by David J. Unwin, Kenneth E. Foote, Nicholas J. Tate, and David DiBiase, 117–32. John Wiley & Sons, Ltd.
- Beck, Kent, and Ward Cunningham. (1987). *Using Pattern Languages for Object-Oriented Programs*.
- Bloom, H. (1956). *Taxonomy of educational objectives* (Vol. 1). New York: David McKay.
- Borchers, J. O. (2001). A pattern approach to interaction design. *AI & SOCIETY*, 15(4), 359-376.
- Bostock, M., & Davies, J. (2013). Code as cartography. *The Cartographic Journal*, 50(2), 129-135.
- Budinsky, F. J., Finnie, M. A., Vlissides, J. M., & Yu, P. S. (1996). Automatic code generation from design patterns. *IBM systems Journal*, 35(2), 151-171.
- Bruner, Jerome S. (1963). *The Process of Education*. Vintage edition. New York: Vintage Books.
- Buttenfield, B. 1999. Usability evaluation of digital libraries. *Science & Technology Libraries*, 17, 39-59.
- Cartwright, William. 2003. "Maps on the Web." In *Maps and the Internet*. Elsevier.
- Cartwright, W. (2012). Neocartography: opportunities, issues and prospects. *South African journal of geomatics*, 1(1), 14-31.
- Cartwright, W. (2008). Delivering geospatial information with Web 2.0. In *International Perspectives on Maps and the Internet* (pp. 11-30). Springer Berlin Heidelberg.
- Christensen, M. J., & Thayer, R. H. (Eds.). (2005). *Software engineering* (3rd ed.). Los Alamitos, Calif: IEEE Computer Society Press. Retrieved from <http://library.books24x7.com/library.asp?^B&bookid=12379>
- Cartwright, W. (2003). Maps on the Web. *Maps and the Internet*. Amsterdam: Elsevier, 35-56.
- Coplien, J. O., & Schmidt, D. C. (Eds.). (1995). *Pattern languages of program design* (Vol. 58). Reading: Addison-Wesley.
- Crampton, J. W. (2009). *Mapping: a critical introduction to cartography and GIS*. John Wiley & Sons.
- Crampton, Jeremy W. 1999. "Online Mapping: Theoretical Context and Practical Applications." In *Multimedia Cartography*, edited by Dr William Cartwright, Dr Michael P. Peterson, and Dr Georg Gartner, 291–304. Springer Berlin Heidelberg.
- . 2008. "Cartography: Maps 2.0." *Progress in Human Geography*, September.
- De Souza, F., & Bevan, N. (1990, August). The use of guidelines in menu interface design: Evaluation of a draft standard. In *INTERACT* (pp. 435-440).
- Dearden, Andy, and Janet Finlay. (2006). "Pattern Languages in HCI: A Critical Review." *Human-Computer Interaction* 21 (1): 49–102.
- Debenham, A. (2013). *A Pocket Guide to Front-End Style Guides*. Five Simple Steps.
- Delikostidis, Ioannis, and Corné P. J. M. van Elzakker. (2009). "Geo-Identification and Pedestrian Navigation with Geo-Mobile Applications: How Do Users Proceed?" In *Location Based Services and TeleCartography II*, edited by Georg Gartner and Karl Rehl, 185–206. Lecture Notes in Geoinformation and Cartography. Springer Berlin Heidelberg.
- Elwood, Sarah. (2010). "Geographic Information Science: Emerging Research on the Societal Implications of the Geospatial Web." *Progress in Human Geography* 34 (3): 349–57. doi:10.1177/0309132509340711.
- . 2011. "Geographic Information Science: Visualization, Visual Methods, and the Geoweb." *Progress in Human Geography* 35 (3): 401–8. doi:10.1177/0309132510374250.
- Foote, K. E. (2011a). "Issues in Curriculum and Course Design: Discussion and Prospect." In *Teaching Geographic Information Science and Technology in Higher Education*, edited by David J. Unwin, Kenneth E. Foote, Nicholas J. Tate, and David DiBiase, 159–64. John Wiley & Sons, Ltd.

- Foote, K.E. (2011b). "Scope and Sequence in GIS&T Education: Learning Theory, Learning Cycles and Spiral Curricula." In *Teaching Geographic Information Science and Technology in Higher Education*, edited by David J. Unwin, Kenneth E. Foote, Nicholas J. Tate, and David DiBiase, 81–95. John Wiley & Sons, Ltd.
- Foote, Kenneth E., David J. Unwin, Nicholas J. Tate, and David Dibiase. (2011). "GIS&T in Higher Education: Challenges for Educators, Opportunities for Education." In *Teaching Geographic Information Science and Technology in Higher Education*, edited by David J. Unwin, Kenneth E. Foote, Nicholas J. Tate, and David DiBiase, 1–15. John Wiley & Sons, Ltd.
- Gale, G. (2013). Push Pins, Dots, Customisation, Brands and Services: The Three Waves of Making Digital Maps. *The Cartographic Journal*, 50(2), 155-160.
- Goodchild, M. F., & Li, L. (2012). Assuring the quality of volunteered geographic information. *Spatial statistics*, 1, 110-120.
- Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). *Design patterns: elements of reusable object-oriented software*. Pearson Education.
- Garrett, Jesse James. (2003). *The Elements of User Experience: User-centered Design for the Web*. Indianapolis, Ind. : London: New Riders.
- Garrett, Jesse James. (2005). Ajax: A new approach to web applications.
- Gibson, Rich. (2006). *Google Maps Hacks*. 1st ed. Beijing ; Sebastopol, CA: O'Reilly.
- Goodchild, M. F. (2014). Twenty years of progress: GIScience in 2010. *Journal of Spatial Information Science*, (1), 3-20.
- Graham, Ian. (2003). *A Pattern Language for Web Usability*. London ; Boston: Addison-Wesley.
- Haklay, Mordechai. (2010). *Interacting with Geospatial Technologies*. Chichester, West Sussex, UK ; Hoboken, NJ: John Wiley. <http://www.WISC.ebib.com/patron/FullRecord.aspx?p=514453>.
- Haklay, Muki, Alex Singleton, and Chris Parker. (2008). "Web Mapping 2.0: The Neogeography of the GeoWeb." *Geography Compass* 2 (6): 2011–39.
- Haklay, M., & Zafiri, A. (2008). Usability engineering for GIS: learning from a screenshot. *The Cartographic Journal*, 45(2), 87-97.
- Harris, L., & Hazen, H. D. (2006). Power of maps:(Counter) mapping for conservation. *ACME: An International E-Journal for Critical Geographies*, 4(1), 99-130.
- Harrower, Mark. (2004). "A Look at the History and Future of Animated Maps." *Cartographica* 39 (3): 33–42.
- Hay, Stephen. 2013. *Responsive Design Workflow*. 1 edition. San Francisco, CA: New Riders.
- Hopfer, S., and A. M. MacEachren. (2007). "Leveraging the Potential of Geospatial Annotations for Collaboration: a Communication Theory Perspective." *International Journal of Geographical Information Science* 21 (8): 921–34.
- Hu, S. (2008). Advancement of Web Standards and Techniques for Developing Hypermedia Maps on the Internet. In *International Perspectives on Maps and the Internet* (pp. 115-124). Springer Berlin Heidelberg.
- Huang, Haosheng, and Georg Gartner. (2009). "Using Activity Theory to Identify Relevant Context Parameters." In *Location Based Services and TeleCartography II*, edited by Georg Gartner and Karl Rehr, 35–45. Lecture Notes in Geoinformation and Cartography. Springer Berlin Heidelberg.
- Hudson-Smith, Andrew, Andrew Crooks, Maurizio Gibin, Richard Milton, and Michael Batty. (2009). "NeoGeography and Web 2.0: Concepts, Tools and Applications." *Journal of Location Based Services* 3 (2): 118–45.
- Jenny, Bernhard, Helen Jenny, and Stefan Räber. (2008). "Map Design for the Internet." In *International Perspectives on Maps and the Internet*, edited by Prof Michael P. Peterson, 31–48. Lecture Notes in Geoinformation and Cartography. Springer Berlin Heidelberg.  
[http://cartography.oregonstate.edu/pdf/2008\\_Jenny\\_etal\\_MapDesignForTheInternet.pdf](http://cartography.oregonstate.edu/pdf/2008_Jenny_etal_MapDesignForTheInternet.pdf).
- Jobs, S. (2013). *Thoughts on Flash*. [online] Apple, Inc. Available:  
<http://www.apple.com/hotnews/thoughts-on-flash/>.

- Jobst, M., & Döllner, J. (2009). Neo-Cartographic Influence on Map Communication in LBS. In *Location Based Services and TeleCartography II* (pp. 207-219). Springer Berlin Heidelberg.
- Kraak, M. J., & Ormeling, F. (2011). *Cartography: visualization of spatial data*. Guilford Press.
- Kraak, J. M., & Brown, A. (Eds.). (2003). *Web cartography*. CRC Press.
- Lienert, Christophe, Bernhard Jenny, Olaf Schnabel, and Lorenz Hurni. (2012). "Current Trends in Vector-Based Internet Mapping: A Technical Review." In *Online Maps with APIs and WebServices*, edited by Michael P. Peterson, 23–36. Lecture Notes in Geoinformation and Cartography. Springer Berlin Heidelberg.
- Liu, S. B., and L. Palen. (2010). "The New Cartographers: Crisis Map Mashups and the Emergence of Neogeographic Practice." *Cartography and Geographic Information Science* 37 (1): 69–90.
- Marsh, S., & Haklay, M. M. (2010). Evaluation and deployment. *Interacting with Geospatial Technologies*, 199-221.
- Mayhew, Deborah J. (1992). *Principles and Guidelines in Software User Interface Design*. Englewood Cliffs, N.J: Prentice Hall.
- Meng, Liqiu, and Tumasch Reichenbacher. (2005). "Map-based Mobile Services." In *Map-based Mobile Services*, edited by Prof Dr L. Meng, Dr T. Reichenbacher, and Prof Dr A. Zipf, 1–10. Springer Berlin Heidelberg. [http://link.springer.com/chapter/10.1007/3-540-26982-7\\_1](http://link.springer.com/chapter/10.1007/3-540-26982-7_1).
- Meyer, J., & Land, R. (2003). *Threshold concepts and troublesome knowledge: linkages to ways of thinking and practising within the disciplines*. UK: University of Edinburgh.
- Miller, Christopher C. (2006). "A Beast in the Field: The Google Maps Mashup as GIS/2." *Cartographica: The International Journal for Geographic Information and Geovisualization* 41 (3): 187–99. doi:10.3138/J0L0-5301-2262-N779.
- Monmonier, Mark S. (1985). *Technological Transition in Cartography*. Madison, Wis: University of Wisconsin Press.
- Muehlenhaus, Ian. (2014). *Web Cartography: Map Design for Interactive and Mobile Devices*. Boca Raton, FL: CRC Press.
- Neumann, Andreas. (2012). "Web Mapping and Web Cartography." In *Springer Handbook of Geographic Information*, edited by Wolfgang Kresse and David M. Danko, 273–87. Springer Berlin Heidelberg.
- Nielsen, Jakob. (1994). *Usability Engineering*. Elsevier.
- Nielson, J. (1992) The usability engineering life cycle. *Computer*, 25, 12-22.
- Norman, D. A. (2002). *The design of everyday things*. Basic books.
- Norman, Donald A., and Stephen W. Draper. (1986). *User Centered System Design: New Perspectives on Human-computer Interaction*. Hillsdale, N.J: L. Erlbaum Associates.
- O'Reilly, B. (2007) What Is Web 2.0: Design Patterns and Business Models for the Next Generation of Software. *Communications & Strategies*.
- Osmani, Addy. (2012). *Learning JavaScript Design Patterns*. 1 edition. Sebastopol, CA: O'Reilly Media.
- Peng, Z. R., & Tsou, M. H. (2003). *Internet GIS: distributed geographic information services for the internet and wireless networks*. John Wiley & Sons.
- Peterson, Michael P. (2008). "International Perspectives on Maps and the Internet: An Introduction." In *International Perspectives on Maps and the Internet*, edited by Prof Michael P. Peterson, 3–10. Lecture Notes in Geoinformation and Cartography. Springer Berlin Heidelberg.
- Plewe, Brandon. (2007). "Web Cartography in the United States." *Cartography and Geographic Information Science* 34 (2): 133–36. doi:10.1559/152304007781002235.
- Prager, Steven D. (2011). "Using the GIS&T Body of Knowledge for Curriculum Design: Different Design for Different Contexts." In *Teaching Geographic Information Science and Technology in Higher Education*, edited by David J. Unwin, Kenneth E. Foote, Nicholas J. Tate, and David DiBiase, 61–80. John Wiley & Sons, Ltd. <http://onlinelibrary.wiley.com/doi/10.1002/9781119950592.ch5/summary>.
- Pulsifer, Peter L., Amos Hayes, Jean-Pierre Fiset, and D. R. Fraser Taylor. (2008). "An Open Source Development Framework in Support of Cartographic Integration." In *International Perspectives on*

- Maps and the Internet*, edited by Prof Michael P. Peterson, 165–85. Lecture Notes in Geoinformation and Cartography. Springer Berlin Heidelberg.
- Riehle, D., & Züllighoven, H. (1996). Understanding and using patterns in software development. *TAPOS*, 2(1), 3-13.
- Robinson, A., Pezanowski, S., Troedson, S., Bianchetti, R., Blandford, J., Stevens, J., Guidero, E., Roth, R.E., & Macheachren, A. (2013). SymbolStore.org: A web-based platform for sharing map symbols. *Cartography and Geographic Information Science*, 40, 415-426.
- Robinson, A., Roth, R.E., Blandford, J., Pezanowski, S. & Macheachren, A.M. (2012). Developing map symbol standards through an iterative collaboration process. *Environment and Planning B*, 39, 1034-1048.
- Rossi, G., Schwabe, D., & Garrido, A. (1997, April). Design reuse in hypermedia applications development. In *Proceedings of the eighth ACM conference on Hypertext* (pp. 57-66). ACM.
- Roth, Robert E. (2013). “Interactive Maps: What We Know and What We Need to Know.” *Journal of Spatial Information Science* 0 (6): 59–115.
- Roth, Robert Emmett. (2011). *Interacting with Maps: The Science and Practice of Cartographic Interaction*. University Park, Pa.: Pennsylvania State University, College of Earth and Mineral Sciences.
- Roth, R. E., & Harrower, M. (2008). Addressing map interface usability: learning from the Lakeshore Nature Preserve interactive map. *Cartographic Perspectives*, (60), 46-66.
- Rouse, L. Jesse, Susan J. Bergeron, and Trevor M. Harris. (2007). “Participating in the Geospatial Web: Collaborative Mapping, Social Networks and Participatory GIS.” In *The Geospatial Web*, edited by Prof Arno Scharl and Prof Klaus Tochtermann, 153–58. Advanced Information and Knowledge Processing. Springer London.
- Scharl, A., & Tochtermann, K. (2009). *The geospatial web: how geobrowsers, social software and the Web 2.0 are shaping the network society*. Springer.
- Shneiderman, Ben. 2010. *Designing the User Interface: Strategies for Effective Human-computer Interaction*. 5th ed. Boston: Addison-Wesley.
- Skarlatidou, Artemis. 2010. “Web-Mapping Applications and HCI Considerations for Their Design.” In *Interacting with Geospatial Technologies*, edited by Mordechai (Muki) Haklay Senior Lecturer, 245–64. John Wiley & Sons, Ltd.
- Slocum, T.A., McMaster, R.B., Kessler, F.C. & Howard, H.H. (2009). *Thematic cartography and geographic visualization*, Upper Saddle River, NJ, USA, Pearson Prentice Hall.
- Spek, Stefan van der. 2009. “Mapping Pedestrian Movement: Using Tracking Technologies in Koblenz.” In *Location Based Services and TeleCartography II*, edited by Georg Gartner and Karl Rehrl, 95–118. Lecture Notes in Geoinformation and Cartography. Springer Berlin Heidelberg.
- Stefanov, Stoyan. 2010. *JavaScript Patterns*. Sebastopol, CA: O’Reilly.
- Suchan, T. A., & Brewer, C. A. (2000). Qualitative methods for research on mapmaking and map use. *The Professional Geographer*, 52(1), 145-154.
- Taylor, D. R. (2005). The theory and practice of cybercartography: An introduction. *Modern Cartography Series*, 4, 1-13.
- Tidwell, Jenifer. (2011). *Designing Interfaces*. 2nd ed. Sebastopol, CA: O’Reilly.  
<http://proquestcombo.safaribooksonline.com/9781449379711>.
- Todd, E. G., E. A. Kemp, and C. P. Phillips. (2010). “Guiding the Development of UI Pattern Models in an Educational Context.” In *Proceedings of the 11th International Conference of the NZ Chapter of the ACM Special Interest Group on Human-Computer Interaction*, 57–64. CHINZ ’10. New York, NY, USA: ACM.
- Tsou, Ming-Hsiang. (2011). “Revisiting Web Cartography in the United States: The Rise of User-Centered Design.” *Cartography and Geographic Information Science* 38 (3): 250–57.  
doi:10.1559/15230406382250.
- Turner, Andrew. (2006). *Introduction to Neogeography*. Sebastopol, Calif: O’Reilly.  
<http://proquest.safaribooksonline.com/0596529953>.



- University Consortium for Geographic Information Science, Model Curricula Task Force, & Body of Knowledge Advisory Board (UCGIS). (2006). *Geographic Information Science and Technology: Body of Knowledge* (1st ed.). Washington, D.C: Association of American Geographers.
- Vora, Pawan. (2009). *Web Application Design Patterns*. Amsterdam ; Boston: Elsevier/Morgan Kaufmann Publishers. <http://library.books24x7.com/library.asp?^B&bookid=32163>.
- Welie, Martijn van, Gerrit C. van der Veer, and Anton Eliëns. (2001). "Patterns as Tools for User Interface Design." In *Tools for Working with Guidelines*, edited by Jean Vanderdonckt and Christelle Farenc, 313–24. Springer London. [http://link.springer.com/chapter/10.1007/978-1-4471-0279-3\\_30](http://link.springer.com/chapter/10.1007/978-1-4471-0279-3_30).
- Welie, M. van. (2003) "Patterns in interaction design."
- Welie, M. (2008) "Design patterns for Web, GUI, and mobile interfaces."
- Wiggins, L. L., & French, S. P. (1991). *GIS: Assessing your needs and choosing a system* (No. PAS Rept No. 433).
- Woodruff, A. (2011). Introducing "On the Horizon". *Cartographic Perspectives*, 68, 83-86.
- Zook, M. A., & Graham, M. (2007). The creative reconstruction of the Internet: Google and the privatization of cyberspace and DigiPlace. *Geoforum*, 38(6), 1322-1343.