Software Project Management Plan
for the
Semantic Web Crawler

13 March 2012

Prepared for:

National University

Prepared by:

Michael Bastos
Davendra Patel
Manuel Covarrubias Jr.

Approvals

Title	Printed Name	Signature	Date
Client Representative	Dr. Appel		
Project Manager	Manual Covarrubias Jr.		
Project Leader	Davendra Patel		
Project Engineer	Michael Bastos		

Revision Changes

The revision change record will contain the revision number, date of revision, engineering change order (ECO) number, description of what was modified, added or deleted, and the individual's name responsible for the change.

Revision Change Record

Revision Number	Revision Date	ECO Number	Description of Change	Author
0	15-Mar-2012		Original	Manager/Leaders

Table of Contents

Table of Tables

Table of Figures

1.0 Introduction

This document encompasses the Software Project Management Plan (SPMP) for development of the Semantic Web Crawler (SWC). The SPMP is the controlling document for the organization of a software project through the definition of the managerial process and the technical process essential to fulfill the project requirements.

1.1 Project Overview

This SPMP is centered on the Operational Concept Document (OCD) and describes the organization of the project development endeavor in preparation for the preparation of the ensuring documents of which the Software Development Plan (SDP) is next. The SPMP is intended for the Lead Project Engineer, the SWC Development Team, Senior Management, Investors, and the customer. The SPMP defines the organization of the product development effort, the managerial and technical process to be implemented and employed, the software packages to be developed in accordance with the stated schedule and budget, and the items to be delivered to the customer. The intended audience for this SPMP consists of:

- Customer (Professor J. Appel)
- The SWC Development Team
- Senior Management
- Investors

The organization of this is as follows:

Section 1.0 provides introductory information for this SPMP by identifying its relationship to the other documents of the project, the items that are to be delivered to the customer, and the date they are to be delivered. This section also explains the SPMP development procedure with respect to configuration and change control management. Finally, this section identifies related reference documents used and lists the definitions, abbreviations, and acronyms used.

Section 2.0 describes the project organization with a primary focus on the models used for the software development process. Following is an explanation of the software development organizational structure and its interrelationship with the many project entities which include the customer, parent organization, configuration management, quality control, and verification and validation efforts. The last part of this section assigns the project responsibilities.

Section 3.0 describes the managerial procedure to be executed in the SWC Product development. The managerial objectives and priorities, project assumptions, and dependencies and constraints affecting the work to be performed are described. Also discussed in this section is a key element on risk management. The risk management evaluates the risks involved in the project development effort, how the risks will be tracked and monitored, and contingency plans to alleviate the occurrence of the identified risk factors. Finally discussed in this section are the project monitoring, the control mechanisms, and the planned staffing to support the development effort.

Section 4.0 describes the technical procedure to be used on the program development. The section goes on to identify the tools, methods and techniques used in the program development, the development of the software documentation and when it is to be reviewed and approved. The last part of the section describes the project support functions, quality control, verification and validation, and configuration management.

Section 5.0 describes the software work packages, budget items, and schedules. The work packages describe the work efforts to be achieved, allocates the percentage of project effort to be assigned to each work package, and where in the project development schedule the effort is to be achieved in accordance with the chosen development plan. The development of dependency charts are used to illustrate the interrelationship of the work package. The last part of the section describes how the budget and resources are allocated to the various tasks and their achievement scheduled are shown in the development schedule chart.

1.1.1 Project Objectives

The objective of the SWC Program is to develop a software program that allows the customer to give it a web address and/or some specific variables to quickly and easily crawl hundreds of pages of a website in a matter of minutes, parsing, and saving only what data is valuable to the end user saving it in either a database or Comma Separated Values (CSV) and Excel spreadsheet format. With proper caching this process can be repeated hundreds of thousands of times with hundreds of websites in various ways allowing the user to better narrow down the type of data they wish to pull from the website without necessarily impacting the crawled site's overall performance or bandwidth.

1.1.2 Product

The SWC product will consist of two releases: (1) a prototype and (2) the full capability product. Each release consists of the software, the appropriate documentation, and a user manual. An enhanced version will be planned for in order to include the capabilities identified during development that are currently outside of the envisioned project scope.

1.1.3 Major Work Activities

The major work activities involved in the SWC Product development consists of research, new development, and integration efforts. The development models to be employed are a prototype development that allows the customer to better define the product and a full capability product. An evolutionary paradigm will be developed based on the customer's evaluation of the prototype and the full capability product. Every development will consist of a fully documented development effort.

1.1.4 Major Work Products

The software, documentation, user manuals, and customer delivery will be the major work products of the development effort.

1.1.5 Major Milestones

The major milestones of the SWC Product development effort consists of a prototype development and a full capability development. The milestones for the prototype are to develop a functional product during the first half of the development effort and the completion of the prototype for customer evaluation. Section 1.2 Product Deliverables provides the detailed milestones of the prototype effort.

1.1.6 Required Resources

The resources requirements for the SWC Product consist of the type of personnel, the number of personnel, office facilities, laboratory facilities, computer time, computer hardware, support software, maintenance requirements, and travel. Section 5.3 Resources Requirements provides a more detailed description of the resources required for the SWC Product development.

1.1.7 Master Schedule and Budget

The master schedule for the prototype development is stated in complete calendar time. Based on the results of the prototype, subsequent development master plans will be defined. The product development's available budget is for the prototype model and is strictly limited. The success of the prototype will determine the budget for the subsequent development.

1.2 Project Deliverables

Table 1.1 lists the products to be delivered to the customer, along with the approximate due dates.

TABLE 1.1 PROJECT DELIVERY SCHEDULE

Product	Due Date	Location	Quantity
Operational Concept Document	20 March 2012	Customer	1
Software Project Management Plan	29 March 2012	Customer	1
Software Requirements Specification (SRS)	12 April 2012	Customer	1
Software Design Spec (Top-level)	19 April 2012	Customer	1
Software Design Spec (Detailed)	26 April 2012	Customer	1
Software Quality Assurance Plan	10 May 2012	Customer	1
Software Test Plan	17 May 2012	Customer	1
User/Operator Manual	30 May 2012	Customer	1
Source Code	30 May 2012	Customer	1
Progress Reports	31 May 2012	Customer	1
Test Report	31 May 2012	Customer	1
CD/DVD	2 June 2012	Customer	1
Product Prototype	2 June 2012	Customer	1

1.3 Evolution of the SPMP

1.3.1 Issue

All of the team members will approve the first issue of the SPMP. This can be done either in person or via e-mail. A hard copy of all documentation and source code will be printed and delivered to the customer. The Project Manager will maintain electronic copies of the project.

1.3.2 Updates

The Senior Manager, Configuration Manager (CM), Quality Assurance (QA) Manager, Documentation Manager, and Project Manager will coordinate any and all changes to the SPMP, whether the change is scheduled or unscheduled. Before delivery to the client, a hard copy of the revised plan shall be printed and reviewed by all the team members. The Senior Manager (or Project Manager) will maintain the electronic copies along with all previous revisions.

1.4 References

Bastos, M., Covarrubias, M., Patel, D., Semantic Web Crawler Operational Concept Document March 2012

Parser definition. http://searchsoa.techtarget.com/definition/parser accessed 14 April 2012

Pressman, Roger S., Project Engineering: A Practitioner's Approach, Seventh Edition, McGraw Hill, 2010.

Used for definitions. www.webopedia.com accessed 17 March 2012

1.5 Glossary

This subsection will provide the definitions of all items, acronyms and abbreviations required to properly interpret the SPMP.

1.5.1 Definitions

Bandwidth Refers to the amount of data that can be sent through a network or modem connection

Bug A coding error in a computer program

C++ A general-purpose programming language that is statically typed, free-form, multi-paradigm, and compiled

Cache A location used to store data in order to have faster subsequent retrievals

Caching	To store data in a cache memory
Cloud Based Environment	Refers to applications and services offered over the Internet. Data Centers that offer services are referred to as the ‘cloud’
Core	Memory that consists of tiny doughnut shaped masses of magnetic material
Database	Collection of data/information that is organized so that a computer program can quickly retrieved desired data/information
Debugging	The process of locating and fixing or bypassing bugs (errors) in a computer program code or the engineering of a hardware device
End-user	Person that a software product or hardware product is designed for
Git	A distributed version control system that is free and open source. Designed to handle any sized projects with efficiency and speed
Github	A web-based hosting service for projects that use the Git revision control system
Google Docs	A free Web-based application in which documents and spreadsheets can be created, edited, and stored online
Hard Copy	Printed copy of information from a computer
Hash	A function that maps large sets of data into smaller sets of data
Java	An object-oriented programming language that is structured around classes instead of functions
Module	A self-contained hardware or software component that interacts with a larger system
MySQL Database	A Relational Database Management System (RDBMS) that runs as a server to provide multi-client access to a number of databases
Object Oriented	Type of programming that creates re-usable objects by combining data structures and functions
Paradigm	A typical example or pattern of something; a model

Parser A program, usually part of a compiler, that receives input in the form of sequential source program instructions, interactive online commands, markup tags, or some other defined interface and breaks them up into parts

Parse To analyze or separate components so they can be more easily processed

Source Code Text list of commands to be compiled or assembled into an executable program

Virtual Servers A server, usually a Web server, which shares computer resources with other virtual servers. In this context, the virtual part simply means that it is not a dedicated server, that is, the entire computer is not dedicated to running the server software

1.5.2 Abbreviations and Acronyms

API	Application Programming Interface
AWS	Amazon Web Services
CM	Configuration Manager
COTS	Commercial Off-The-Shelf
CSV	Comma Separated Values
ECO	Engineering Change Order
GB	Gigabytes
GHz	Gigahertz
GUI	Graphical User Interface
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol with Secure Sockets Layer
ID’s	Identifiers
IDE	Integrated Drive Electronics
I/O	Input/Output
Mbps	Megabits per second
OCD	Operational Concept Document
OS	Operating System
QA	Quality Assurance
RAM	Random Access Memory
regex	regular expression
RDBMS	Relational Database Management System
RDS	Relational Database Service
SCM	Software Configuration Management
SDD	Software Design Description
SDL	Software Development Library
SDP	Software Development Plan

SE	System Engineering
SPMP	Software Project Management Plan
SQA	Software Quality Assurance
SRS	Software Requirements Specification
SVGA	Super Visual Graphics Array
SVN	Subversion
SWC	Semantic Web Crawler
UI	User Interface
UX	User Experience
VGA	Visual Graphics Array

2.0 Project Organization

2.1 Process Model

A Spiral Software Development Model will be used to develop the Semantic Web Crawler Software Program. During this Spiral Model, the first phase will be the prototype. After the prototype has been completed, a version that contains full capability will be produced. The enhanced model will thus be stemmed from this full capability version.

Based upon the customer parameters, a reflection will be evident in the prototype. At this point, it will become necessary to further consult with the customer to address the direction of the prototype. The finished prototype will incorporate the objectives that are derived from the customer consultations.

The Spiral Model is the ideal production method for the Semantic Web Crawler as it affords the opportunity to determine the objectives and identify and solve the risks with prototyping before the next iteration and prototyping begins to produce an evolving software product.

The timing chart of the project functions and activities is defined in Table 2.1.

TABLE 2.1 PROJECT FUNCTIONS AND ACTIVITIES

Function/Activity	Purpose/Description	Date
Project Initialization	Initial review of the project and planning of specifications.	6 March 2012
OCD Final	Present Operational Concept Document.	20 March 2012
OCD Signoff	Management verifies OCD meets standards.	21 March 2012
SPMP Draft	Deliver draft version of Software Project Management Plan.	28 March 2012
SRS Draft	Draft preliminary version of Software Requirements Specification.	29 March 2012
SPMP Final	Present final version of SPMP to management.	29 March 2012
SPMP Signoff	Management verifies SPMP meets standards.	30 March 2012
Software Design Document (top-level)	Project Engineer begins functional design development for programmatic representation of data.	9 April 2012
SRS Final	Present final version of SRS to management.	12 April 2012
SRS Signoff	Management verifies SRS meets standards.	13 April 2012
Software Design Description Draft	Draft preliminary version of Software Design Description (SDD).	18 April 2012
Function Design Final	Project members review Function Design to familiarize themselves with its operation.	19 April 2012
Software Design Review	Review Software Design Description.	26 April 2012
SDD Final	Present Final version of SDD to management.	28 April 2012
SDD Signoff	Management verifies SDD meets requirements.	29 April 2012

TABLE 2.1 PROJECT FUNCTIONS AND ACTIVITIES Cont.

Function/Activity	Purpose/Description	Date
SDD Presentation	Short presentation to customer/management on Software Design Description.	1 May 2012
Question Responses	Deliver answers to management and customers questions.	1 May 2012
Software QA	Present software QA Plan to management.	10 May 2012
Create User Documentation	Start creating User Documentation (online help system).	18 May 2012
Test Document Draft	Draft preliminary version of Test Documentation.	18 May 2012
Test Plan Final	Present final version of Test Documentation to management.	29 May 2012
Test Plan Presentation	Management verifies Test Documentation meets standards.	30 May 2012
User Documentation	Release User Documentation to management.	30 May 2012
Test Report	Present test results to management and customer.	31 May 2012
Formal Project Presentation	Formally present the product to management and the customer.	2 June 2012
Product Prototype	Deliver Product Prototype to customer.	2 June 2012

2.2 Organizational Structures

Project Manager

Absolute responsibility for the project and total responsibility to the customer. Directs the entire project and development processes. Responsible for mediating development differences among individual Project Engineers.

Project Engineer

Analyzes system software requirements and coordinates software team efforts to develop the software system architecture, implementation of the software into working computer code language, integrates system hardware and software components, and tests the system to verify it complies with requirements. Responsible for working with the Configuration Manager to create a solid and efficient development environment.

Document Manager Manages & reviews all documentation for project development.

Quality Assurance Manager Conducts reviews and audits to verify software products comply with applicable procedures and standards.

Configuration Manager Identifies software configuration at specified points in project timeline and systematically controls configuration changes.

Testing Manager Plans and performs independent system testing of project system to determine whether the product satisfies requirements.

2.3 Organizational Boundaries and Interfaces

The organizational boundaries and interfaces section describes the interfaces between the core development team and all other organizations or teams involved in development of the final product.

2.3.1 Parent Organization

The parent organization of the customer will interact with the software management team along with the customer staff. Issues that arise for the parent organization team will be directed to the customer staff who will engage with the management of the software project.

2.3.2 Customer Organization

The customer and the team that the customer works with will interact with the project core team via the project manager and also the customer manager. In addition, the customer may also interact with any member of the software project team through means of the management staff.

2.3.3 Subcontracted Organization

The parties and tools that are utilized for the software project will be contacted by the management team involved in the development of the software. These interactions will help facilitate the requirements and boundaries of the subcontracted parties.

2.3.4 Configuration Management

The Configuration Manager will be the point of contact and the liaison for the issues that pertain to configuration matters. The development team and team leaders will interact with the Configuration Manger to address obstacles with the software developing process. The

Configuration Manager will directly convey the issues to the Lead Project Manager for the decision process regarding the outcome with the resolution.

2.3.5 Quality Assurance

The Quality Assurance Manager will be the point of contact for areas of quality that are addressed to the internal core of the development team. The team members and managers will interact with the Quality Assurance Manager on a case by case basis to determine the ideal resolution for issues that arise.

2.4 Project Responsibilities

Table 2.2 defines task responsibilities according to organizational position.

TABLE 2.2 ORGANIZATIONAL PROJECT TASK RESPONSIBILITIES

Tasks	Project Manager	Project Engineer	Document Manager	Quality Assurance	Configuration Manager	Test Manager
Requirements Definition	X	X	X			
Requirements Analysis	X	X	X			
Process Management	X					
Scheduling	X	X	X	X	X	X
Data Collection	X	X	X	X	X	X
Configuration Management	X	X	X		X	
Hardware Setup		X			X	
Quality Assurance	X			X		
Testing	X					

3.0 Managerial Process

3.1 Management Objectives and Priorities

Objectives include producing software that exceeds the customer expectation, and producing a prototype that functions well within the requirements of the customer. This development of the functional prototype will facilitate the customer to grade the software and provide input to the management team to revise and manipulate the software to fit the updated customer requirements. Each interface will be evaluated for completion to the guidelines and requirements. Time is of the essence. The schedule of development is crucial for the tasks to stay on the correct course of development.

3.2 Assumptions, Dependencies, and Constraints

3.2.1 Assumptions

- It is assumed that the user will have an Internet connected web based browser to interface the software program to initiate the semantic web crawl.
- It is assumed that the user will have a limited experience with the techniques used to implement the software functions.
- It is assumed the functions used will be running in the background away from the user.
- It is assumed that the user will be able to utilize forms and fields to enter the semantic web based crawl criteria necessary for the program to initialize the instance.

3.2.2 Dependencies

The software project is dependent upon the access of web based servers and the source code that is used to implement the required functions for the semantic web crawl.

3.2.3 Constraints

Time is the largest restraint for the development of the software project. The goal is to provide a functional prototype within two weeks of final deliverance to allow for time for alterations and improvements that are derived from the customer feedback of the initial working prototype. The aggressive pace will leave little room for deviation from the initial requirement plan and is designed for a steadfast and rapid development process. As said before, time is of the essence and the wasting of time is forbidden during the project engagement. Therefore, a strict schedule of ongoing group meetings has been implemented and regimented into the developer’s timeline. The development environment has been obtained and the developers have configured their respective machines to engage with the environment. A collaborative document environment has been implemented to allow for the retrieval and updating of all documents necessary for the project.

3.3 Risk Management

The program development team will conduct risk analysis and management throughout the software development process. Areas of the software development project that involve potential technical, budget, or schedule risks will be identified, analyzed, and prioritized. We will include a contingency plan for each identified risk. Some of the risks that need to be taken into consideration include any other software in the market that provides the same function as the product being developed, development of interfaces between Commercial Off-The-Shelf (COTS) products, time, and debugging software errors. Below is a rough estimate of what categories of risk we attempted to consider:

3.3.1 Risk Analysis

3.3.1.1 Product Size

Having to essentially build a browser that can handle Hypertext Transfer Protocol with Secure Sockets Layer (HTTPS) as well as session data are risks associated with the overall size of the software to be built or modified and could cause us to run into technical barriers.

3.3.1.2 Business Impact

Since all three of the developers on this project work on other more pressing projects and issues for our perspective businesses, having to de-prioritize development time on the SWC are risks associated with constraints imposed by management or the marketplace.

3.3.1.3 Customer Characteristics

If our customer’s need changes throughout the product development then we could run the risk associated with the sophistication of the customer and the developer's ability to communicate with the customer in a timely manner.

3.3.1.4 Process Definition

Since this is our first documented software project for all three team members, falling behind on the documentation, and/or having to redo a document that we maybe didn’t first understand properly are risks associated with the degree to which the software process has been defined and is followed by the development organization.

3.3.1.5 Development Environment

We are building this project for our perspective needs but as time goes by and/or if we discover better solutions that take up less time to develop or implement can be a risk associated with the availability and quality of the tools to be used to build the product.

3.3.1.6 Technology to be Built

Finding a good template to base our crawler on, may create risks associated with the complexity of the system to be built and the ‘newness’ of the technology that is packaged by the system in designing a program which is generally built in a more of a scripting environment.

3.3.1.7 Staff Size and Experience

Not being able to finish our coding effort on the project schedule due to conflicting time constraints on the part of the three man project team can carry risks associated with the overall technical and project experience of the Project Engineers who will do the work and could potentially delay us as fixes are required along the way.

3.3.2 Probability, Impact, and Priority

All team members are responsible for monitoring the project’s risk factors. Each development team member shall have the programming software and database application installed and running on one or more local machines to support software development. Any issues which represent a risk to the successful completion of the project are to be reported to the Project Engineer and will be discussed with the entire team as soon as practical. See Table 3.1.

TABLE 3.1 RISKS: PROBABILITY, IMPACT, AND PRIORITY

Risk	Probability	Impact	Priority
Risk #1: 3.3.3.1 Staff Size and Experience	90%	5	4.5
Risk #2: 3.3.3.2 Process Definition	80%	4	3.2
Risk #3: 3.3.3.3 Product Size	70%	4	2.8
Risk #4: 3.3.3.4 Development Environment	40%	3	1.2
Risk #5: Customer Characteristics	30%	2	0.6
Risk #6: Technology to be Built	10%	4	0.4
Risk #7: Business Impact	20%	1	0.2

3.3.3 Risk Tracking/Monitoring

The Project Engineer is responsible for implementing contingency plans to mitigate any of the above risks. In general, the contingencies involved can be summarized as follows:

3.3.3.1 Risk #1: Staff Size and Experience

Aversion Not being able to finish our coding effort on the project schedule due to conflicting time constraints on the part of the three man project team can carry risks associated with the overall technical and project experience of the Project Engineers who will do the work and could potentially delay us as fixes are required along the way.

Monitoring The Project Engineer in conjunction with senior management will address risks related to the product or personnel management as well as handle all setup and training of team members in order to allow others to keep up in the development process.

Contingency We will allocate additional time at the end of the project specifically to doing any left over work that could not be accomplished due to lack of programming experience. We have leveled our prototype requirements to ensure that all necessary components are simple enough for the project team to tackle initially.

3.3.3.2 Risk #2: Process Definition

Aversion Since this is our first documented software project for all three team members, falling behind on the documentation and/or having to redo a document that we maybe didn’t first understand properly are risks associated with the degree to which the software process has been defined and is followed by the development organization.

Monitoring We have essentially given different members of the team specific responsibilities that are self managed and have established accountability points for when a team member does not complete a specific assigned task or job.

Contingency The Project Engineer and the Customer Coordinator will meet with the customer to determine if there is any way in which the product can be altered to meet customer needs. We have different program modules readily available in the event that the project cannot be completed by the team for a specific part of the project.

3.3.3.3 Risk #3: Product Size

Aversion Risks related to programming will be mitigated through implementation of sound programming and design practices from the project’s inception. All code will be well documented and built to conform to the customer’s requirements as laid out in the SRS and SDD.

Monitoring Peer review will be emphasized as part of the overall testing program. All team members will unit-test their code and the QA organization will report any programming errors it uncovers to the Project Engineer, who will coordinate with the other programmers to find and correct them.

Contingency We will use standard templates that are already created in other languages for how a web crawler should be built, we will analyze the structure and pinpoint what and where our compiled code should differ from existing systems and where it should be exactly the same.

3.3.3.4 Risk #4: Development Environment

Aversion Risks related to the software that underlies the system will be mitigated by either changing the program interfaces to the software or by changing the software systems used. The team has established a git repository on github that allows us to share our code and spans our software and development risks across all of us. Risks related to the hardware on which the system runs should be mitigated by the upgrading and/or changing of the hardware involved.

Monitoring If something needs to be fixed it can quickly be worked on by anyone on the team from any location and allows us to share the work load virtually. The team has setup virtual hardware on Amazon Web Services (AWS) Platform specifically so that we can resolve some of our hardware related risks. Everything is virtually located in the cloud and thus we do not need physical access to physical hardware.

Contingency We also use Google Docs for all of our documentation so if anyone in the team can't complete a specific project area then others can jump in on their working copy and make corrections as needed. Each team member will use their own personal laptop and our development environment can be quickly setup and running on any computer with a browser whether it's running a Windows, Mac, or Linux Desktop.

3.4 Monitoring and Controlling Mechanisms

Project monitoring will adhere to the provisions of the SRS. Project monitoring will be done in all meetings and approved by the Project Engineer. The major monitoring mechanism is the testing phase of the project. Suggestions with regard to documents shall be made by the client/user and approved by the Project Engineer.

3.5 Staffing Plan

The number of personnel is limited to three. The number of team members will remain the same until the prototype version of the product is produced. Eventually, the software will require additional personnel assets for full development maintenance and implementation. The skill level of the personnel will increase over the duration of the project. Training is not a factor for the prototype version. After the prototype is complete our goal is to hire on more personnel with the following qualifications.

- Virtual Server Architect - Responsible in setting up and maintaining AWS Virtual Servers - Potential Cost $65,000 a year.
- MySQL Database Designer - Responsible for handling growth in the data being collected - Potential Cost $80,000 a year.
- Lead Software Engineer - Responsible for maintaining and making modifications to core - Projected Cost $120,000 a year.
- Lead User Experience (UX) Designer - Responsible for building and maintaining the Graphical User Interface (GUI) for creating crawls - Projected Cost $100,000 a year.

4.0 Technical Process

4.1 Methods, Tools, and Techniques

The methods, tools, and techniques section details the technical aspects of the core development process. Programming methods, specific tools and languages, and other techniques employed by the SWC Program development team are discussed.

4.1.1 Programming Method

The development process will follow an object-oriented method, in which the output of the requirements analysis phase produces a set of classes that describe the behavior of the system. Discrete areas of functionality in the final system will be provided by single modules (both executable and object code), which may be re-used as their functionality is needed in other areas of the system. This also eases the debugging process, as bugs that are corrected in one area will automatically--by virtue of module re-use--be corrected in all areas that use that module.

4.1.2 Tools

The program is comprised of six discrete software components:

- Hypertext Transfer Protocol (HTTP)/HTTPS crawler.
- Input/Output (I/O) Operations.
- Content Storage.
- Hypertext Markup Language (HTML) & Data Parser.
- Database Management.
- User Interface (UI).

Development of the system requires tools appropriate for working with each of these elements.

4.1.2.1 HTTP/HTTPS Crawler

A semantic web crawler functions much like a simple web browser, it will be able to download individual web pages and accept session identifiers (ID's) as well as other necessary minor requirements in order to properly view the page's contents.

4.1.2.2 I/O Operations

The client will send a command to crawl a website in which case a cloud instance will begin and start the crawl using the core software. Commands will be sent through to the parser on which lines of a downloaded page will be read and properly analyzed as well as told what type of data to pull out. Once the crawl has ended and the data has been queued into the Relational Database Service (RDS) database system, then the instance will be immediately turned off allowing for proper resource allocation and to save the client money on hosting and or server costs.

4.1.2.3 Content Storage

The first time the program downloads a website, before it enters the parser, it will be uploaded to a static storage medium with a website address, time, and date stamp. This prevents sites from getting crawled repeated times because something went wrong during the parsing process. It's essentially like copying a website locally to view it at a later time.

4.1.2.4 HTML Parser

The secondary part of the Core Software will be the Parser itself; in this case there will be multiple parser internal pieces that will function with different forms of filtering through the data itself. All functions within the Parser will be templated and recursive so that it may be used again and again throughout the parsing process. All regular expression (regex) and/or parsing comments to include pagination will be passed through as well as allowing the Core Software to be as nimble as necessary in order for it to perform its function as efficiently as possible.

4.1.2.5 Database Management

All possible forms of collected data will be placed into a hash and sent to the third part of the Core for database integration and/or implementation. A Cloud managed MySQL database through the RDS Amazon Services will be utilized to design and store the database, due to the modularity of the system, this is a benefit to the client as the client may choose to port the entire system to a local machine if they no longer wish to utilize AWS or if the service gets discontinued at some point.

4.1.2.6 User Interface

The last but most important piece of the puzzle is the UI, it will really need to be able to allow the client to select certain elements of a specific page which will be crawled, the UI will then send an Application Programming Interface (API) command to the Core to do the actual crawling but the secret is how the API in the Core is formatted to accommodate the possible desires of the client. It'll also allow for future upgrades of the UI without the need to necessarily modify the core.

4.2 Software Documentation

This section discusses the documentation plan for the SWC Program.

4.2.1 Deliverables

TABLE 4.1 DOCUMENT SIGN-OFF/REVIEW DATES

Document	Date	Sign-off/Review
Status Report	Weekly	N/A
OCD	22 March 2012	21 March 2012
SPMP	30 March 2012	30 March 2012
SRS	13 April 2012	13 April 2012
SDD	29 April 2012	29 April 2012
User Documentation	30 May 2012	30 May 2012
Test Report	30 May 2012	30 May 2012
Source Code	31 May 2012	31 May 2012
Working Application (Prototype)	2 June 2012	2 June 2012

4.3 Project Support Functions

This section discusses the project support functions and development approach for the SWC Program.

4.3.1 Configuration Management

4.3.1.1 Responsibilities

The Project Engineer will work with the CM to design, create, and maintain the Software Configuration Management (SCM) environment and the git repository. They will also be responsible for software acquisition and build environment policies. Both managers will work together to do the following:

- Identify the files to be placed under version control and the build approach for the system software.

- Create a logically partitioned build environment for the Software Development Library (SDL).
- Identify the policies and steps for developers to check out and check in files.
- Setup and Maintain the git repository for storing and managing changes to the source code.

Specific CM tasks include verifying that the product installs correctly onto a given machine and ensures that the product is always built from a consistent source code base. All changes to the files which comprise the code base or client-distributed code must be coordinated through the CM. For the SWC Program this means managing:

- C++ crawler source code
- HTML and Ajax files for front end GUI
- Backup Copies of the MySQL RDS Database

4.3.1.2 Resource Management

The team configuration management manager will accomplish the CM. All changes to any of the above resources must be sent to that team member for review and approval prior to being integrated into the baseline distribution. The baseline will be maintained on the project's website and git project for easy access by all developers.

4.3.1.2.1 Configuration Identification

Several references are made in this SPMP as to which files are to be placed under version control and their eventual locations. The following entities will be placed under configuration control:

- All newly created code
- All COTS source code and libraries

All of the following and remaining documentation will be maintained in our Google Docs Folder Repository:

- Approved System Engineering (SE) documents (e.g., SRS, SDD)
- Initial approved project schedule

4.3.1.2.2 Configuration Control

The CM will establish and implement procedures designating the levels of control through which each identified entity must pass. The CM will be responsible for the following activities for establishing and maintaining configuration control:

- Create version branches for the SWC code.
- Create permissions for the development team to check in and checkout of source code, yet preventing them from deleting software elements for the version repository.

- Create the git file system structure in conjunction with the Project Engineer.
- Identify the procedures to be followed by developers for merge conflicts during code check-in.
- Identify the chain of command to follow to request modifications to the git repository.
- Identify the steps for users to follow when reporting a git problem.

4.3.1.2.3 Configuration Status Accounting

The inherent accounting, log and history features of SWC will be used to maintain records of the configuration status of all entities that have been placed under project-level or higher configuration control.

4.3.1.2.4 Configuration Audits

The acquirer (Customer) will not be able to conduct audits on the configuration environment.

4.3.1.2.5 Packaging, Storage, Handling, and Delivery

Once the System Qualification Tests have been completed without error, the CM will execute a final package build of the software. The package will be housed in a specific file in the library. The filename will indicate the version information and its permissions will be set to read-only for selected users, groups, and others.

4.3.1.3 Budget

No additional budget is required for this support function as this project prototype is completely client-funded.

4.3.1.4 Schedule

The CM will commence with the achievement of a proof-of-concept program, and will continue until the final product is released to the customer.

4.3.2 Software Quality Assurance

4.3.2.1 Responsibilities

The process of Software Quality Assurance (SQA) involves verifying that the final software product is free of observable defects. This means testing of the following components:

The Web Crawler:

- Is the response time of each crawl acceptable?
- Does it crash under system load in the middle of a crawl?

The RDS database:

- Is the data in the database consistent?
- Does adding, retrieving, and deleting data actions function as they should?

The user interface:

- Is it intuitive and easy to understand?
- Is navigation consistent and clear?

The HTML Parser:

- Does it produce any internal server errors?
- Does it produce proper data, or occasional incorrect or corrupted data?

4.3.2.2 Resource Requirements

All SQA functions will be accomplished internally. Internal testing will take place whenever programmers make changes to any system components and will need no additional resources.

4.3.2.3 Budget

No additional budget is required for this support function as this prototype project is completely client-funded.

4.3.2.4 Schedule

Internal testing will begin when active development begins on the system, which will occur after the achievement of a proof-of-concept C++ sample crawler. Integration testing is expected to occur during the last week in May 2012.

4.3.3 Verification and Validation

4.3.3.1 Responsibilities

Verification and validation involves assuring that each delivered item meets the agreed-upon specifications as represented in the appropriate documents. This involves both internal and external review of all products released to management or customers, and also overlaps into the area of SQA, as it is crucial to validate that the system performs exactly as specified.

4.3.3.2 Resource Requirements

No additional resources are needed to provide verification and validation services to the project. These services will be provided by a strict internal review of all products before they are released to the management or the customer; by management and customer review of products after their release; and by incorporating the results of these reviews into future releases.

4.3.3.3 Budget

No additional budget is required for this support function as this prototype project is completely client-funded.

4.3.3.4 Schedule

Verification and validation services shall begin and end at the same time as SQA services.

5.0 Work Packages, Schedule, and Budget

5.1 Work Packages

Upon completion, the work packages will comprise of a successful completion of the SWC Product Prototype. See Table 5.1. Work packages for subsequent phases will be developed based on the outcomes of the prototype development.

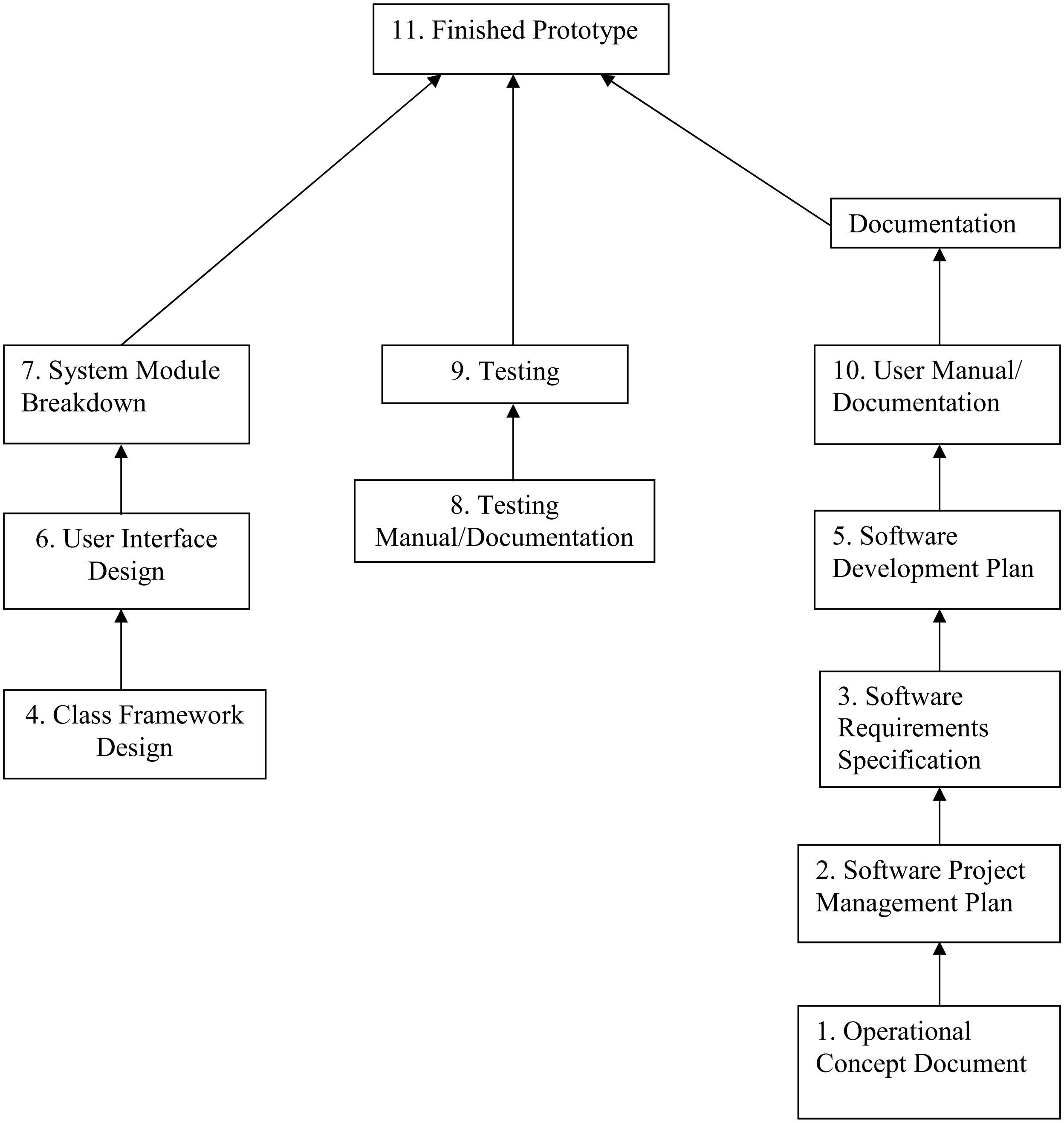
TABLE 5.1 WORK PACKAGES

Task Number	Title/Description
1	Operational Concept Document
2	Software Project Management Plan
3	Software Requirements Specification
4	Class Framework Design
5	Software Development Plan
6	User Interface Design
7	System Module Breakdown
8	Testing Manual/Documentation
9	Testing
10	User Manual/Documentation
11	Finished Prototype

5.2 Dependencies

Figure 5.1 illustrates the dependencies between the work packages.

FIGURE 5.1 WORK PACKAGE DEPENDENCIES


11. Finished Prototype
Documentation
7. System Module Breakdown
9. Testing
10. User Manual/ Documentation
6. User Interface Design
8. Testing Manual/Documentation
5. Software Development Plan
4. Class Framework Design
3. Software Requirements Specification
2. Software Project Management Plan
1. Operational Concept Document

5.3 Resource Requirements

The resources required for the successful completion of this project are discussed in this section. Section 5.3 covers the hardware, software, and personnel resources that will permit the SWC Project Development Team to deliver the system according to management and client specifications.

5.3.1 Hardware

The hardware requirements are specified for each member on the Development Team. They include:

- An Intel-compatible personal computer with server capability and speed of 1 Gigahertz (GHz) or higher (or equivalent).
- An Internet connection for access to Web based resources of at least 1 Megabits per second (Mbps) or higher.
- Mouse or compatible pointing device and keyboard.
- Monitor - Visual Graphics Array (VGA) or higher resolution monitor (Super Visual Graphics Array (SVGA) recommended).
- 2 Gigabytes (GB) of Random Access Memory (RAM) for the operating system.

5.3.2 Software

The software requirements are also specified for each member on the Development Team. They include:

- An Operating System (OS) capable of generating an Internet connection.
- Internet browser.
- Eclipse Integrated Drive Electronics (IDE) for C++ development.
- Ubuntu Server.
- SWC Core Software.
- Git or Subversion (SVN) Server Repository for shared development and historical data recovery.

5.3.3 Personnel

The SWC Project requires a minimum of 3 team members. Each team member fulfills multiple roles. See Table 5.2.

TABLE 5.2 ORGANIZATIONAL ASSIGNMENTS

Organizational Position	Team Member
Project Manager	Manuel Covarrubias Jr.
Lead Project Engineer	Michael Bastos
Project Engineers	Davendra Patel, Manuel Covarrubias Jr.
Documentation Manager	Manuel Covarrubias Jr.
Customer Coordinator	Davendra Patel
Quality Assurance Manager	Manuel Covarrubias Jr.
Configuration Manager	Michael Bastos
Testing Manager	Davendra Patel

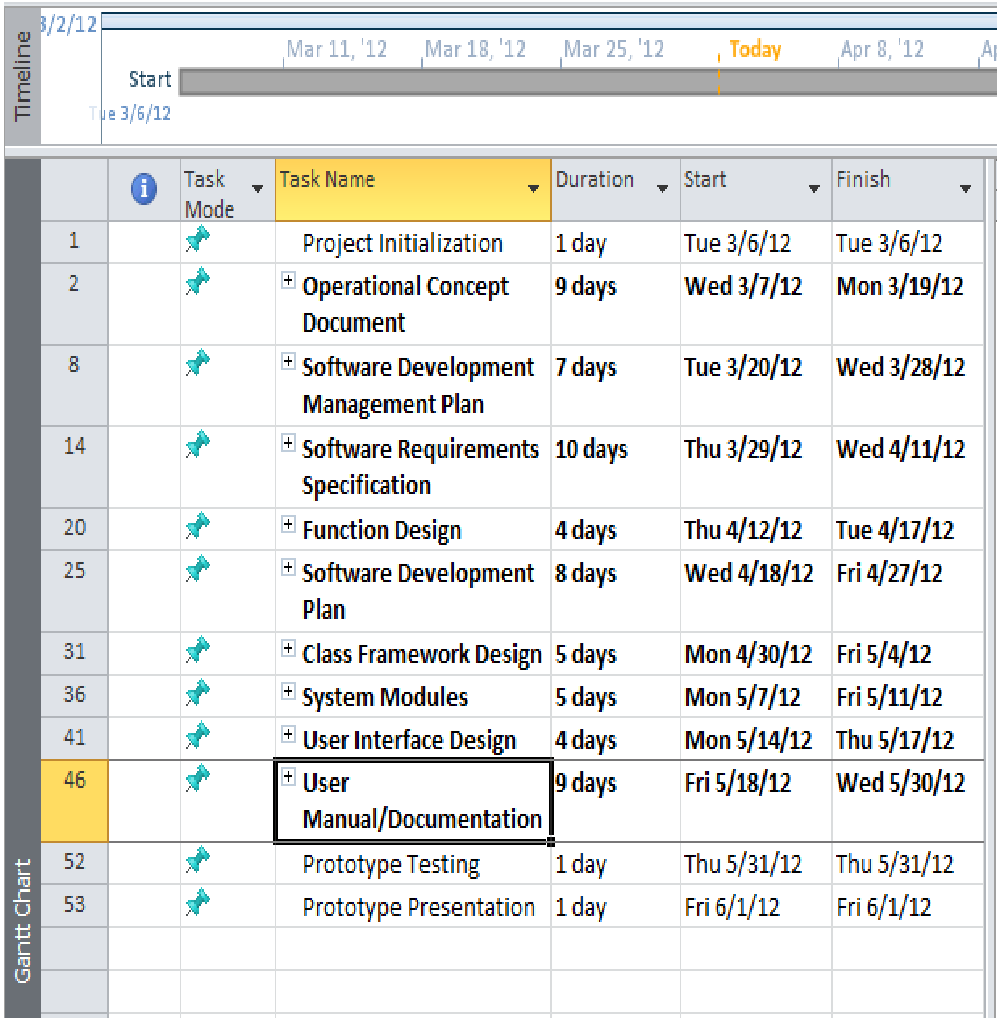
5.4 Budget and Resource Allocation

The SWC Prototype is completely funded by the client and no additional budget is required. Regarding the resource allocation, the reader is to refer to the individual sections of this SPMP document that discuss the resource allocation.

5.5 Schedule

The SWC Prototype development schedule is illustrated in Microsoft Project screen shots in Figures 5.2 and 5.3.

FIGURE 5.2 SWC PROTOTYPE DEVELOPMENT SCHEDULE SCREEN SHOTS



	ℹ	Task Mode	Task Name	Duration	Start	Finish
1		📌	Project Initialization	1 day	Tue 3/6/12	Tue 3/6/12
2		📌	+ **Operational Concept Document**	**9 days**	**Wed 3/7/12**	**Mon 3/19/12**
8		📌	+ **Software Development Management Plan**	**7 days**	**Tue 3/20/12**	**Wed 3/28/12**
14		📌	+ **Software Requirements Specification**	**10 days**	**Thu 3/29/12**	**Wed 4/11/12**
20		📌	+ **Function Design**	**4 days**	**Thu 4/12/12**	**Tue 4/17/12**
25		📌	+ **Software Development Plan**	**8 days**	**Wed 4/18/12**	**Fri 4/27/12**
31		📌	+ **Class Framework Design**	**5 days**	**Mon 4/30/12**	**Fri 5/4/12**
36		📌	+ **System Modules**	**5 days**	**Mon 5/7/12**	**Fri 5/11/12**
41		📌	+ **User Interface Design**	**4 days**	**Mon 5/14/12**	**Thu 5/17/12**
46		📌	+ **User Manual/Documentation**	**9 days**	**Fri 5/18/12**	**Wed 5/30/12**
52		📌	Prototype Testing	1 day	Thu 5/31/12	Thu 5/31/12
53		📌	Prototype Presentation	1 day	Fri 6/1/12	Fri 6/1/12

FIGURE 5.3 MICROSOFT PROJECT GANTT CHART SCREEN SHOT

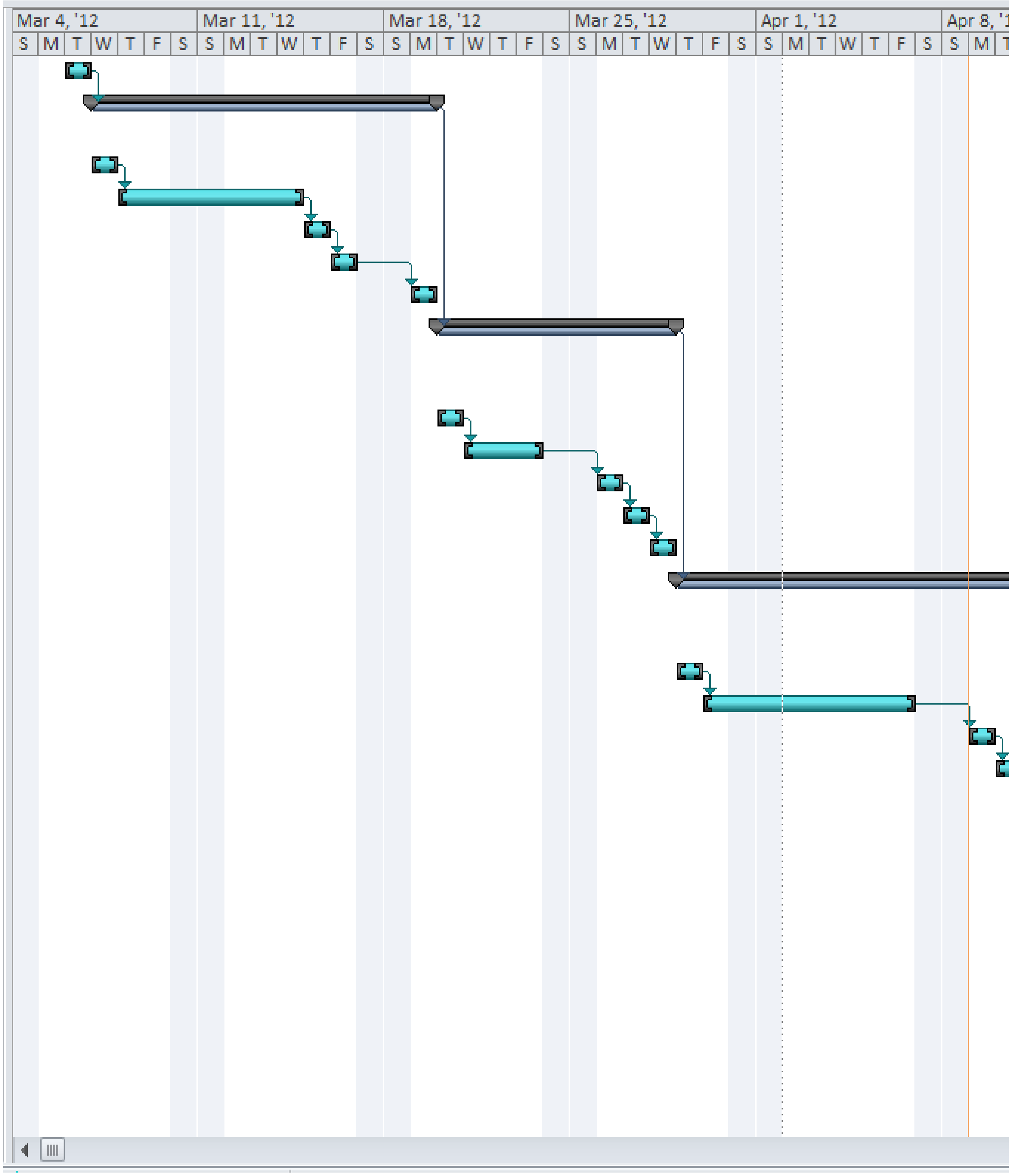

Mar 4, '12
Mar 11, '12
Mar 18, '12
Mar 25, '12
Apr 1, '12
Apr 8, '1
S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T

FIGURE 5.3 cont.

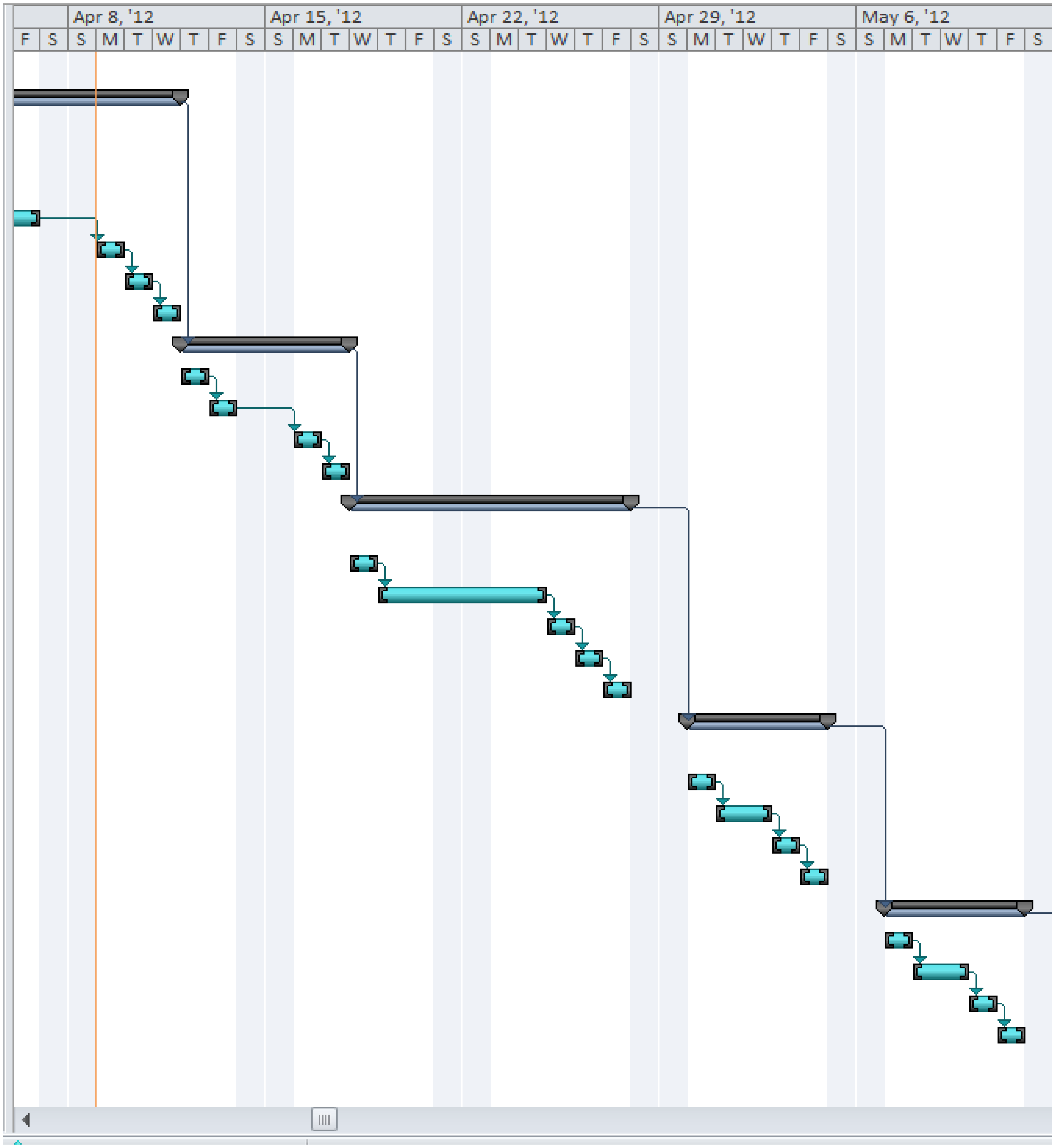
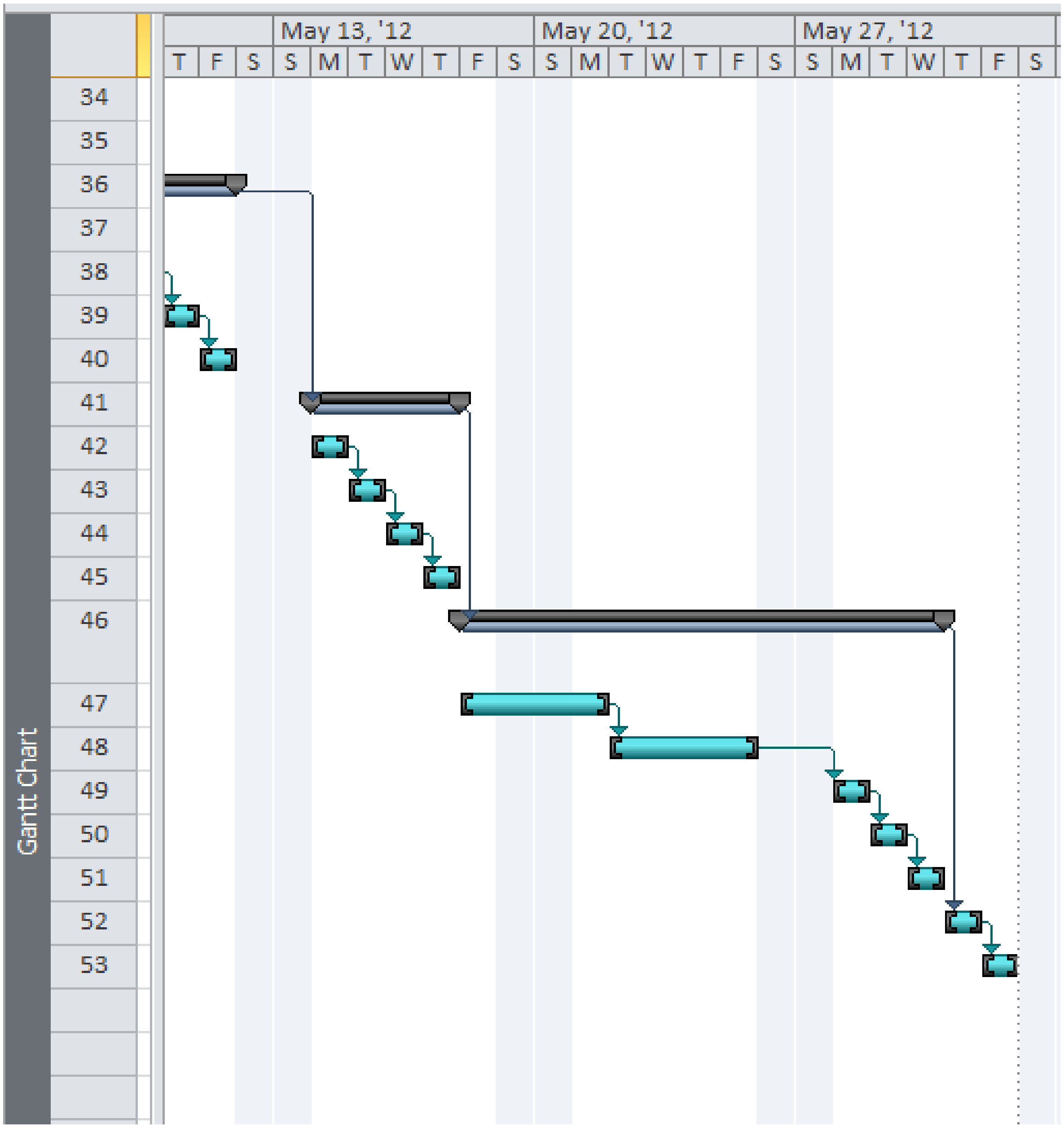

Apr 8, '12
Apr 15, '12
Apr 22, '12
Apr 29, '12
May 6, '12
F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S

FIGURE 5.3 cont.


Gantt Chart
May 13, '12
May 20, '12
May 27, '12
T F S S M T W T F S S M T W T F S S M T W T F S
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53