8-2004

Productivity Analysis and Use of Sequence-Based Specification in a Web-Development Environment

Carla Renee Sparks
University of Tennessee, Knoxville

To the Graduate Council:

I am submitting herewith a thesis written by Carla Renee Sparks entitled "Productivity Analysis and Use of Sequence-Based Specification in a Web-Development Environment." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Computer Science.

Tomas E. Potok, Major Professor

We have read this thesis and recommend its acceptance:

Jesse Poore, Stacy Prowell

Accepted for the Council:
Dixie L. Thompson

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

To the Graduate Council:

I am submitting herewith a thesis written by Carla Renee Sparks entitled "Productivity Analysis and Use of Sequence-Based Specification in a Web-Development Environment." I have examined the final paper copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Computer Science.

Tomas E. Potok, Major Professor

We have read this thesis and
recommend its acceptance:

Acceptance for the Council:

Vice Chancellor and Dean of
Graduate Studies

PRODUCTIVITY ANALYSIS AND USE OF SEQUENCE-BASED SPECIFICATION IN A WEB-DEVELOPMENT ENVIRONMENT

A Thesis
Presented for the
Master of Science Degree
The University of Tennessee, Knoxville

Carla Renee Sparks

August 2004

ACKNOWLEDGMENTS

I would like to thank my advisor Dr. Thomas Potok for his patience and guidance throughout the course of graduate studies. His enthusiasm for the practice of software engineering fostered my interest in the field and led me to search for better, more efficient ways of approaching software development in my own profession. I would also like to thank Dr. Jesse Poore and Dr. Stacy Prowell for their support and contributions to this study.

I owe much gratitude to my co-workers and friends at GoTrain Corp for their unending encouragement to complete this thesis and for their contributions to the projects studied herein. From GoTrain, I owe special recognition to Joel Townsend my supervisor and mentor who gave me the opportunity to perform this study and taught me to appreciate the attention to details in all things.

Finally, I would like to thank my husband and best friend Anthony Tingle for his support throughout the duration of my graduate studies. His dedication to science and research are inspiring, and this thesis would not have been possible without his patience, understanding and incredible cooking.

ABSTRACT

This study evaluates the productivity of a software team in a web-development company and assesses the effects of the sequence-based specification process on productivity and software accuracy in this environment. This study compares two software projects completed at GoTrain Corporation in 2001 and 2002. GoTrain is an application service provider and delivers environmental, safety and health (ES&H) training courses to a variety of clients through an Internet-based learning management system (LMS), called the Academy.

GoTrain was established in 1999 through the merger of two small companies – a training services organization and a web design group. Because neither of the parent companies specialized in software development, the new GoTrain programming team began creating the first Academy applications without the structure of a formal software process. This study evaluates the productivity of the GoTrain programming team at the time formal software processes were introduced into the development environment.

The first project evaluated was an upgrade of the GoTrain Academy to provide Spanish support for the end-user and was performed using the Microsoft Solutions Framework ™ (MSF) Process Model. The second project was an upgrade to the Academy software to support new employee training, enhanced reporting functionality, and improved administrative features. For this project, the MSF Process Model was again used but with sequence-based specification applied to selected Academy features during the design phase. Sequence-based specification is typically used in Cleanroom software engineering to create consistent and complete product requirements through enumeration of system inputs.

Focusing on active server pages (ASPs), productivity analyses were based on the total lines of code (LOC) generated during the project and the number of hours required to create the code. The count of errors discovered during testing and the hours required for rework after the Academy release were used to evaluate the accuracy and correctness of the software.

A productivity increase is seen between the first and second projects. The second project had higher LOC per man-hour than the first, which is likely a result of the software team becoming more experienced with the software process, developing cohesion among the team members, and improving communication among the project group. The files utilizing the sequence-based specification process in the second project had similar LOC and man-hour values as other files modified during this development effort. However, these files showed better accuracy and correctness in post-deployment use. Files utilizing sequence-based specification required no modification after the versioned release of the Academy, whereas 68% of the files, similar in LOC and man-hours, created using existing specification processes required modification and re-deployment following the initial release.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1. INTRODUCTION

This study evaluates the productivity of a software team in a web-development company and assesses the effects of the sequence-based specification process on productivity and software accuracy in this environment. A standard processes was used as the baseline for two software projects – the Microsoft Solutions Framework™ (MSF) Process Model (Microsoft, 2002). For the second project, sequence-based specification was introduced in the design phase for a set of the application features (Prowell and Poore, 2003). Sequence-based specification, often used in Cleanroom software engineering, is the process of systematically creating product specifications through enumeration of system inputs and abstraction of complex sequences (Prowell and Poore, 2003; Prowell et al., 1999).

The intent of this study was two-fold. First, the attempt was made to determine if a productivity gain could be seen between two software projects in a web-development environment where formal software processes were being introduced. Second, the study evaluates if the utilization of sequence-based specification in the design phase of a web-based application could result in improved efficiency and accuracy in the development effort. Metrics considered to indicate improved efficiency in the development process are:

- Increased lines of code (LOC) per man-hour
- Decreased man-hours per LOC
- Decreased number of errors per thousand LOC (KLOC)

Two projects similar in scope, including project team and development environment, were used in this study. Metrics for empirical analyses were tracked throughout both projects, including the following:

- Number of files edited
- Lines of code added, edited or removed
- Developers on the project team
- Man-hours to write the LOC
- Errors logged during testing.

Productivity comparisons were then made between the two projects based on LOC per man-hour (Potok et al., 1999), and errors per LOC.

1.1 Description of the Company

GoTrain.net was the branded product name given to the system created in 1998, in Knoxville, Tennessee by Tenera Energy and SoBran, Inc. Later merging and assuming the company name GoTrain Corp (GoTrain), the companies created a system for delivering and tracking compliance-based Environmental, Safety and Health (ES&H) training. GoTrain created ES&H courses for delivery over the Internet to help clients comply with Federal- and State-mandated training regulations. In order to deliver courses, track progress, register course completions, and allow a company's training administrators to assign required training, GoTrain created a Learning Management System called the Corporate Distance Learning Center. This product was later renamed the Online Training Academy© and hereafter referred to as "the Academy".

1.2 Description of the Academy

Like the ES&H courses, the Academy is accessed entirely over the Internet. End-users, referred to as "learners", use the Academy to launch courses and view their training histories. Training administrators have access to additional features in the Academy through a link to the Administrative Features. The Academy has features designed to help companies meet regulatory training requirements. The basic features include:

- Login where learners access the Academy;
- Course Menu where learners view and launch their training courses;
- Reports where learners can see their personal training history.

Additionally, the Academy has the following administrative features:

- Initial Setup and Demographics Interview where administrators control the set-up and data fields of their Academy;
- Curriculum, Training Groups and Training Requirements where courses can be assigned to groups of learners or individuals and parameters such as passing threshold, validity period, pre-test and test-out can be set;
- Learner Information where learner data can be modified or added;
- Course Catalog where course information can be edited and "other" types of training such as classroom, video, etc. can be defined;
- Assign Completions (AC) where completions for the "other" (non-web-based) types of training can be assigned to learners;

- Exemptions and Equivalencies (EX/EQ) where exemptions from or equivalencies for specified training requirements can be granted;
- Reports where administrators can view training information for groups of learners, learner data, and summaries of training requirements.

Ultimately, every parameter the administrator inputs or chooses in the features contributes to the final view of courses a learner sees on his or her Course Menu.

1.3 Description of the System

The GoTrain Academy is available to end-users and administrators over the Internet. Like many web applications, it is comprised of a tiered architecture. This type of architecture is comprised of layers that are independent components and often do not run on the same machine. The layers work together, but they are not combined into a single executable application. Generally, each layer has no specific knowledge of what other levels are doing. (Kolawa, et al., 2002)

The GoTrain Academy is a 3-tier architecture that includes the presentation layer, the application layer, and the database layer (Fraser, 2002). The presentation layer displays the application to the end user through files held on the web server, which are requested by the end-user's web browser, and then rendered on the user's workstation. The file types in the Academy presentation layer consist primarily of hypertext mark-up language (html) files and active server pages (ASPs). Code used in these files includes Visual Basic™ script (VBScript), html, dynamic html, and JavaScript™, which are executed at the end-user's web browser.

The application layer, or middle tier, is an executable file that resides on the web server and handles the business logic of the Academy. In this case, the executable was a dynamically linked library (dll) file created in Visual Basic™ (VB). The dll is registered on the web server and facilitates data exchange between the server-side pages and the underlying database. Code in the dll is executed at the web server.

Behind all of the code for the presentation and application layers resides a database holding information to support the basic functionality of the application. In this case, the back-end database was SQLServer™ 6.0, which held information on clients, end-users, assigned courses, course tracking, and course completions.

Web development in a multi-tier architecture such as this one is often considered difficult and challenging for developers (Fieldon, 2000; Strom, 2000). This environment poses unique challenges to the developer because the nature of the architecture creates multiple levels at which design and code must be addressed. Care must be taken to integrate all layers effectively and accurately. Programming languages and the associated skills required of the developers may vary drastically at each level of the application. As a result, layers may be developed independently of each other and perhaps even by different developers. Consequently, this environment creates opportunities for error at multiple levels.

1.4 Existing Problems in the Development Group

Because neither of the two founding companies of GoTrain had established software processes, the first developers of the Academy began creating software in an environment lacking structure and project control. Product specifications were informal, making it difficult to trace the final product back to original requirements. Code management was handled through Microsoft Visual Source Safe ™ (VSS), but releases of the Academy product were not formally versioned in the VSS database. Communication among team members was also informal, often leading to confusion between product designers and developers about product specification or design. Without regular project meetings, coding often occurred in isolation from both programming peers and product reviewers, resulting in long test-and-recode cycles at the end of a project. Likewise, product release dates were often pushed out from the original planned release, and errors were often discovered during normal product use after release.

CHAPTER 2. BACKGROUND

The difficulties experienced by the GoTrain development group were characteristic of a company operating without standard processes for creating and maintaining software (Carnegie Mellon, 1994). Numerous strategies exist for organizing and managing software products, such as the Initiating, Diagnosing, Establishing, Acting, Leveraging (IDEAL^{SM}) method described in (McFeely, 1996); the Cleanroom method (Dyer, 1992; Prowell et. al, 1999); the MSF (Microsoft Corp., 2003); and the Capabilities Maturity Model Integration^{SM} (SEI, 2002). This list is by no means comprehensive but is intended to demonstrate the number of possible approaches a company may take when attempting to improve its software development process.

Because the Academy 3-tier system was built entirely on Microsoft technology, GoTrain development procedures naturally fell in line with a process model created specifically for Microsoft development organizations. The MSF is a set of principles and guidelines established by Microsoft for businesses that design, create, deploy, and maintain software built on Microsoft technologies (Microsoft Corp., 2003). The MSF is intended to help companies manage software projects and produce consistent and reliable, high-quality products. In 2001, in an attempt to make development cycles more controlled, managed, and efficient, the GoTrain development group implemented the MSF Process Model, a feature of the MSF, to track and manage its projects.

While the implementation of the MSF Process Model greatly improved the effectiveness and structure of the GoTrain development process, specifications and design documents created during this process were not complete. As a result, use cases discovered during coding or testing often required rework. Project schedules were forced to allocate time at the end of each development effort to accommodate these test-rework cycles, and errors were still reported by client administrators after product deployment. In late 2001, GoTrain attempted to improve the specification step of the development process by including the sequence-based specification process in the design phase of certain Academy features. The intent was to evaluate if system requirements defined more completely on the front end of the development process could have a positive effect on the rework cycles at the back end.

2.1 Overview of the MSF Process Model

The MSF Process Model provides guidance on the complete software process to help a business manage and track a software development cycle from design to deployment. Phases of development are punctuated with milestones where project goals, risk, and progress are assessed (Townsend, 2000; Microsoft Corp., 2003), and as a result, project scope and schedule are more closely tracked and managed. With the implementation of this process, GoTrain intended to better control the design and development cycles of the Academy as well as deliver versions on a schedule consistent with client demands and company objectives (Townsend, 2004).

The MSF Process Model provides an orderly means of managing software projects. The model is characterized by a phase and milestone structure. It is an iterative model that takes an integrated approach to building and deploying software solutions (Microsoft Corp., 2003).

The MSF Process Model describes a means of organizing a project into five phases – envisioning, planning, developing, stabilizing, and deploying - each with accompanying milestones and deliverables. This model is particularly appropriate for large projects and progressive feature implementation, and combines aspects of the popular Waterfall and Spiral models (Microsoft Corp., 2003). Figure A.1[1] shows the phases and milestones of the MSF Process Model.

The process provides measurable results that can be examined at each milestone. Each milestone represents a project nodal point at which all teams should synchronize their efforts with project deliverables and customer expectations (Townsend 2000, Microsoft Corp., 2003). At any milestone in the process, the development effort can be evaluated and modifications to resources, features or schedule can be made as necessary, and risk can be assessed and a determination made to proceed or cease the development effort. Table A.1 lists the phases, milestones and deliverables, as they existed for the Academy development projects at GoTrain.

2.2 Cleanroom and Sequence-Based Specification

Cleanroom software engineering is an approach to software development intended to produce failure-free software. The Cleanroom process uses mathematically based methods for product specification, design, and correctness verification, and statistical methods for testing and software certification. The description of Cleanroom software engineering and sequence-based specification presented here can be found in (Linger and Trammell, 1996; Prowell 1996; Prowell et al, 1999; and Poore and Prowell 2003).

[1] All tables and figures are in Appendices.

The key features of Cleanroom are incremental development; function-based specification, design and verification; and statistical testing and correctness verification. Function-based specification, design and verification in Cleanroom software engineering are attained using box structures – black box, clear box and state box (Mills et al., 1986). Specification begins with the black box, an external view of the system, which maps input (stimulus) sequences to a set of anticipated outputs (responses). The response of the system is based not only on the current stimulus but also on the history of stimuli. The state box is derived from the black box and describes the required behavior of a system in terms of a transition from a current stimulus and state to a corresponding response and new state. The clear box implements procedures to carry out the state box mapping rules.

Sequence-based specification is often used in the Cleanroom software process to derive the black and state box definitions. This process has been shown to create complete, consistent, and traceably correct specifications by enumerating all possible sequences of system stimuli and mapping them to their correct responses. A stimulus is any valid input into the system, and a stimulus sequence is a series of such inputs. Sequences are identified as possible or impossible, intended or erroneous, and reducible or irreducible. By identifying the sequences using these descriptors, accurate and finite system specifications can be derived, and each element of the specification can be traced back to its originating requirement in the requirements document. For any sequence without a defined response in the requirements document, a response must be created to clarify the expected system behavior. These responses are called derived requirements, and they become part of the overall system specification.

There are four steps in the sequence-based process: tagged requirements, system boundary definition, sequence enumeration and canonical sequence analysis. Product requirements are extracted from the requirements document and tagged, or numbered, so that system responses can be traced back to defined behavior.

The system boundary determines which components are inside the system and which components are outside the system serving as sources of stimuli and destinations of responses. Sequence abstractions may be useful when defining the system boundary for a number of reasons, for instance, to hide well-understood details or to reduce growth of enumerations.

Sequence enumeration lists all possible sequences of the defined stimuli and evaluates each sequence to determine if the sequence is legal. For each enumeration, the correct response to the sequence must be documented and mapped to the requirement on which the response was based. When sequences identified as equivalent, meaning their responses to future stimuli are identical, only the shorter sequence is required to be extended in the enumerations. Enumerations continue until all sequences are identified as either illegal or equivalent.

Canonical sequences are legal sequences in the enumeration that are not equivalent to any previous sequence. These sequences are used to define the state of the system. State variables are created and their values defined for each of the canonical sequences. The state box definition is then derived from this canonical sequence analysis.

2.3 Productivity Analysis

This study compares productivity between two software projects to evaluate the effect of introducing formal software processes in a web-development environment. Likewise, productivity comparisons are done at the file level in the second project to assess the impact of incorporating sequence-based specification into the design process of certain application features.

While factors such as ability and training of team members and team behavior can factor into overall productivity (Potok et al., 1999; Potok and Vouk, 1999), typically, LOC is the only metric available for empirical analyses. In this study, an attempt was made to minimize the effects of team variation by evaluating two projects that shared four members of a six-person team. In both cases, these four developers were responsible for creating or editing at least 97% of the code.

Productivity comparisons were made between the two projects using the following:

Productivity = Project Size / Effort

(Potok et al., 1999)

In this case, project size was measured as LOC added, edited or removed during the project, and effort was measured in man-hours required to modify the LOC. Productivity was calculated for each file edited during the project.

This productivity equation implies a linear relationship between project size and effort. The Constructive Cost Model (COCOMO) developed by Barry Boehm in 1981 and subsequent derivative models show that average programmer productivity is a non-linear function that varies based on many factors, including the type and size of project (Boehm, 1981; Potok et al., 1999; Sodhi and Sodhi, 2001; Pandian 2004; Umbers 2004). However, the linear relationship can be used to compare productivity of a development team on multiple projects (Potok et al., 1999). Productivity analyses in this study uses LOC per man-hour for the files edited during each project, and comparisons are then made to typical productivity results as seen in (Potok et al, 1999; and Boehm, 1981).

CHAPTER 3. APPROACH

Each project in this study was similar in scope, development team and environment; and project tracking, documentation and communication were consistent between both development efforts. Both development efforts occurred between May 2001 and February 2002 and were upgrades of the existing GoTrain Academy code. The Spanish project utilized the MSF Process Model, and the Korova project used the MSF Process Model with sequence-based specification applied to certain Academy features during the design phase. Counts of LOC, man-hours and errors were compiled at the completion of each project.

3.1 Project 1 – Spanish Support

GoTrain's first use of the MSF Process Model was in Spring 2001 during the Spanish project. In this iteration of the Academy, the user interface (excluding administrative functions) and courses were converted to Spanish. Users would be allowed to select their language preference on both the login screen and their course menus, and the entire user interface and list of courses would be presented in the language chosen.

This project affected 31 ASPs, 22 html pages, and 11 VB class modules, and required overhaul of the entire end-user interface as well as exam features in the 60 GoTrain ES&H courses. The conversion of the course content for 60 courses is not included in this study because the translation effort was done by GoTrain graphic design staff without the contribution of the development team. However, the course player files, which were edited by the development staff, are included in data of this study.

3.1.1 Envisioning Phase

During the Envisioning Phase of the project, product team members created a list of requirements for the Academy modification. Client Academies could be enabled for Spanish support with a bit flag set in the database. Users in those Academies would then have the option to change the entire user interface, including courses, to Spanish by simply clicking a link. The Academy would maintain the user's language preference in subsequent logins. At any time, the interface and courses could be changed back to English by another click of the language link. Likewise, the training administrator would have the ability to select Spanish as the default language for particular a user.

The requirements document produced during the envisioning phase was formally called the Functional Specification. The document was finalized by the GoTrain product team and approved by GoTrain management, development, and product team members in June 2001. Because of the sensitive nature of product documentation, only excerpts of these documents are presented here. Portions of the *Support for Spanish Language Courses - Functional Specification* are shown in Appendix B.

3.1.2 Planning Phase

The next step in the process, the planning phase, allowed the development team to look at the project specifications and design the system additions and changes to support those requirements. This step required the participation of individuals who were familiar with both the existing system and the limitations of the technology used to create and deliver the product.

The deliverable for the planning phase was the Detailed Design document, which specified system changes for each tier in the application. Database schema changes were explicitly stated with definitions for table and field names, data types, default data values, triggers, and stored procedures. In the middle tier, functions and their parameters and variable names and types were specified for the passing of data to and from the database and the ASPs. The ASP additions and modifications were also detailed in this document. While the Spanish Detailed Design document described the functionality and data to be displayed on the ASPs, it did not include page layout and design. The final page layouts were created by the graphic design staff. Appendix B shows portions of this Detailed Design document.

3.1.3 Developing Phase

In the developing phase, GoTrain developers took the Detailed Design and implemented the specified code changes and additions. Project schedule and progress were tracked using Microsoft Project™ software. The initial project schedule was created on June 16, 2001 with a projected completion date of July 24, 2001. The schedule was updated based on information from the development team. Regular meetings allowed the development team to update project managers on progress and to discuss any project hurdles or coding contingencies with other members of the development team. These meetings proved extremely beneficial to both the management and development groups.

3.1.4 Stabilizing Phase

During the stabilizing phase, the development team worked with the testing team to identify, track and correct errors in the application. The MSF Process Model does not have explicit directives for creating and implementing test plans. However, to avoid the test team trying only haphazard use cases, the GoTrain project manager created a Test Plan with prescribed test cases for the administrator and multiple end users. The testing team followed the use case instructions in the Test Plan and extended the test procedures with an ad hoc testing style.

Issues were recorded by the test group and tracked by the project manager in a Microsoft Excel™ spreadsheet. The project management and development team held daily meetings to review the issues list, assign new issues to developers for correction, flag corrected issues for validation testing, and strike validated issues from the list. The testing process took approximately 1 month of the project schedule. One hundred thirty-four issues were tracked, and 96 were corrected. The remaining issues were flagged for a future release, listed as requested enhancements beyond the scope of the Spanish project, or found not to be an application error. The final testing round occurred on July 24, 2001, on target with the initial project completion date.

During this phase account management and sales staff prepared the client notification plan. Likewise, the instruction design staff completed the final product documentation for the end-users, in the form of Learner and Administrator Guides.

3.1.5 Deployment Phase

The Deployment Phase was completed on July 31, 2001, when the final Academy product including server files, the dll executable, and database schema, was deployed to the production servers at the company hosting facility. The Academy code was labeled in VSS as Academy version 2.4, in preparation for the next development effort and versioned release.

3.2 Project 2 – Korova

Beginning in August of 2001, immediately following the Spanish Academy project, and extending into early 2002, the GoTrain development team began another major overhaul of the Academy. Unlike the Spanish project, which focused primarily on learner features, this project involved almost all learner and administrative features in the Academy.

While the implementation of the MSF Process Model greatly improved the effectiveness and structure of the development process, the hours due to rework continued to be high. All project members knew what to expect in the final product based on the Functional Specifications document, and coders knew what to create based on the Detailed Design document. However, even the best Detailed Design document can rarely describe the system changes required to handle all use cases. Often the minute details of functionality had to be reconsidered as coding progressed. Decisions on functionality were left up to the programmer, or were overlooked altogether until considered by the testers. As exemplified by the multiple test-code-and-retest cycles of the Spanish project, the code created following the Detailed Design document was still less than perfect. Typical errors found during testing arose when a particular use case or sequence of user-created events had not been considered during the specification phase.

In this project, the MSF Process was again utilized but with the addition of the sequence-based specification process, which was used to create system requirements for EX/EQ and

AC administrative features. GoTrain included the sequence-based specification process to evaluate its effectiveness at reducing ambiguity in the specification process and contributing to shorter test cycles and fewer code errors.

3.2.1 Product Requirements

This project consisted of an ensemble of requested changes in the administrative features of the Academy. The compilation of these requests resulted in an overall project designed to make the administrative features of the Academy more flexible while providing needed functionality for assigning and tracking regulatory-based training. Ultimately, the project affected almost every page of the administrative interface and most of the learner interface.

The Training Requirements feature was slated for modification to handle new employee training and annual retraining. The Reports feature was to be modified to provide administrators more report types and more flexible sorting and filtering capabilities. The EX/EQ and AC features were also scoped for revision.

The EX/EQ feature allowed administrators to grant a user an exemption from or an equivalency for a course. The AC feature was similar to EX/EQ, but it allowed an administrator to create a completion record for non-web-based courses tracked through the Academy. This feature provided administrators a way to track completions for such training types as on-the-job, classroom and video. The features were functional; but administrators could grant EX/EQ or AC for only one course per user at a time. Likewise, EX/EQ did not expire after they were assigned, which made them more permanent than was desired.

Requirements were defined for each of the sections of this project, and Functional Specification documents were created following the MSF Process Model. However, during the design phase, all features were combined into one project definition. Because the title "Retraining, New Learner, Reporting, Learner Remediation and Exemption/Equivalency" was a bit burdensome when referring to the project, the shorter name Korova was chosen by the project team, and will be the title used hereafter when referencing this project.

Given the extensive scope of the Korova project, the portions using sequence-based specification were limited. Because this project was driven by existing client demands as well as market needs identified by the sales group, GoTrain considered the time to delivery critical. It is well documented that a successful implementation of the full Cleanroom process requires management buy-in and support of the process, as well as project teams who are trained in the Cleanroom method (Dyer, 1992; Henderson, 1995; Becker et al., 1996; Linger and Trammell 1996; Prowell et al., 1999). Because GoTrain developers were not familiar with the Cleanroom process or sequence-based specification at the onset of the Korova project, management did not favor making necessary provisions in the schedule for the introduction of the full Cleanroom process to the development team. However, GoTrain management agreed that portions of the Korova project could incorporate sequence-based specification. Therefore, AC and EX/EQ were identified to use sequence-based specification. These features comprised 15.4% of the total Academy features slated for modification during the Korova project.

3.2.2 Tagged Requirements

The Functional Specification document GoTrain used to describe the required changes to these features can be found in Appendix C. The tagged requirements for the EX/EQ and AC specification processes were extracted from this document and are shown in Tables D.1 and D.2. Only the first three tagged requirements affect the functionality of the EX/EQ and AC pages. Because the various features of the Academy are tightly integrated, in each specification the effects must be detailed not only for the feature being modified but for the other affected Academy features, as well. The requirements for other Academy features are also indicated in these tables.

3.2.3 System Boundary

In defining the system boundary for EX/EQ and AC, stimuli could come only from user input. The stimuli and responses for the system are shown in Tables D.3 and D.4. The EX/EQ and AC systems were evaluated in two separate phases to cover both cases of adding or editing the features. Abstractions were used in the enumerations for adding EX/EQ, adding AC, and editing EX/EQ.

Because an administrator can add multiple EX/EQ or AC in one page instance, abstractions were used to cover multiple add cases. For example, an administrator may input an EX-1 for course #1 and an EX-2 for course #2; hence, the abstraction EX was used to cover both cases. This also applied to the EQ, Date and Justification stimuli on the EX/EQ page, as well as the Complete, Date, Score and Comments stimuli on the AC page. In the event information was entered for multiple courses, each course and its associated inputs could be considered separate stimulus sequences. Therefore, a stimulus history of "Ex-1, Date- 1, Justification-1" would take on the sequence "EX, Date, Justification". If the stimulus history were "Ex-1, Date- 2, Submit", the sequences would be interpreted as separate stimulus histories, "EX, Submit" and "Date, Submit", respectively.

Additionally, in the enumerations for editing EX/EQ, an abstraction was used for the stimuli of an existing EX or EQ. When either an EX or EQ is edited, the box checked as EX or EQ may be changed from one to the other. The system response would be the same in either case; therefore, an abstraction E-old was used to identify the field from which the check was removed, and E-new was used to represent the field in which a new check had been entered. This abstraction allowed enumerations to be done one time for the "Edit EX/EQ" case versus separate enumerations for both "Edit EX" and "Edit EQ".

3.2.4 Sequence Enumeration

As enumerations began, it quickly became apparent the requirements document was inadequate to cover the full functionality of adding or editing EX/EQ and AC. Early in the enumerations, the derived requirements shown in Tables D.5 and D.6 were created to fully define the two systems. The complete enumeration sequences for EX/EQ and AC can be found in Tables D.7 through D.10. The final listing of stimuli and responses and their requirements traces are shown in Tables D.11 through D.14.

Based on the complete set of requirements for EX/EQ and AC, the user interfaces shown Figures D.1 and D.2 were designed and served at the entry point for stimuli into the systems.

In the initial design of these pages, the learner selection box appeared on the page with the other EX/EQ and AC data entry options. This design would allow the administrator to navigate from one user to another without exiting this page. However, the Learner Remediation iteration of this project modified the method by which users were selected for both EX/EQ and AC. The Learner Remediation design specified a search feature allowing any learner functions to be applied to the selected user from a common page. The resultant Learner Information screen is shown in Figure D.3. While this change ultimately removed the "Learner" stimulus from the EX/EQ and AC pages, it remained a stimulus to both systems.

3.2.5 Canonical Sequence Analysis

Canonical sequences are those equivalent to no prior sequence in the enumerations. State variables are used to capture the conditions of the system for each sequence of stimuli and to represent the state data of the system. These variables retain the aspects of the stimulus history required to produce correct responses from future stimuli (Prowell et al., 1999). In the EX/EQ system, state variables were used to represent the page displayed, the learner training requirements, the date and justification fields, and the EX/EQ check box. In the AC system, state variables were used to represent the page displayed, the learner training requirements, the comments, score and date fields, and the completion check box. The canonical sequence analyses are shown in Tables D.15 through D.18, including the state variables and their values before and after the current stimulus.

The sequence-based specification process for EX/EQ and AC identified deficiencies in the original requirements document for this portion of the Korova project. As a result, further definition was provided to the development team for the coding of the EX/EQ and AC, which removed ambiguity in functionality that would not have been exposed until the testing phase of the project.

The development, stabilizing and deployment phases characteristic of the MSF Process Model were continued for the EX/EQ and AC features after the specification process described here. Likewise, the same testing methods were performed in the Korova project as in the Spanish.

CHAPTER 4. DATA COMPILATION

The data compilation procedures were similar for both the Spanish and Korova projects, with the few exceptions noted in the following sections. GoTrain did not have automated procedures for measuring LOC, recording hours per file, or tracking issues. Therefore, all data collected for this study were compiled manually using GoTrain's time-tracking system, source control software, and issues-tracking spreadsheets.

Data used in these comparisons were archived prior to starting each project and again at each project completion. In this way, post-Spanish files are synonymous with pre-Korova files. For simplicity, the term "pre-project" will be used here to describe files and code as they exited before the project began, and "post-project" will be used to describe files and code after project completion.

4.1 Code Base

The first step in collecting the project data was gathering the pre-project code base to which the post-project files could be compared. This code was archived from VSS before the project began. Visual Source Safe allowed files to be archived based on the modified date. This removed the possibility of counting lines of code in files edited after the previous Academy release but prior to the start of the next project. It was assumed that any edit to the files during the period of the project schedule were due to code changes specified by the project documentation or were otherwise in support of the current development effort. In both projects, however, there were files edited during the project period that ultimately had no associated hours. This was especially true when nearing the end of a project. These edits were most likely in preparation for other project or addressed an issue outside the current development effort. These files and associated LOC were not included in these analyses.

The files used for comparisons were ASPs, VB class modules, and html pages. Because database changes were not held in source control, they could not be compared between versioned releases. Therefore, database modifications are not included in these analyses.

4.2 Lines of Code

The directories of pre- and post-project files were compared using Microsoft Windiff ™, a utility that can compare two files or directories. In the case of a directory compare, it will show all files modified between the two directories and mark the newer file. When comparing files, it will also show lines added, modified or removed.

First, Windiff™ was used to compare the directories holding the pre- and post-project code base. All files identified as "newer" in the post-project directories were considered for further analysis. Files that were unchanged, missing from the post-project directories, or marked as older in the post-project folders were not considered in the data analyses.

The second step in calculating LOC was finding the number of lines per file that were added or edited during the development effort. The first attempt to count the LOC simply involved opening each file in a code editor and comparing total line counts. This method, however, did not take into consideration the lines of existing code that were edited or removed. Simple line counts in each of the files would not produce an accurate count.

The method chosen was a bit more labor-intensive but accounted for LOC added, edited and removed. In this approach, each file in the post-project code base was compared directly to its predecessor file in the pre-project code, again using Windiff™. Lines were then counted manually in the comparison window. For new files created during the current project, all lines in the file were counted.

One risk in manual count is human error counting the LOC. To validate that manual counting was producing accurate results, 25 of the 64 files edited in the Spanish project were counted using both automated and manual methods. Results showed that manual count was more accurate representing true level of effort for files with lines removed, and was in line with automated counts when lines were only added or edited. Complete details on the files considered for validation and the percent differences between automated and manual counts can be found in Appendix E.

Another risk in this method is the inflations of LOC counted when major edits were made to the ASPs that simply moved features from one location on the page to another. Because similar code the files was separated by approximately 200 LOC, manual counts in the Windiff™ window could result in inflated line counts. This scenario occurred only in the Korova project for the Add/Edit Training Requirements, AC, and add/edit Learner Information features. In these cases, the files driving these features were compared in hard copy where sections of code at the end of one file could be visibly matched to code at the top of another file. These counts ultimately reduced the total LOC for these files and more accurately reflected the true line count.

4.3 Man-hours

The next step in data compilation was determining the number of man-hours required to produce the LOC. First, it was necessary to determine the developer and day for each file edit. Again, VSS was used to examine file histories and extract both the modified dates and the developer who modified the file. Before a file can be edited by a developer, it must first be checked out of VSS. Visual Source Safe™ then records the date, time and network ID of the developer each time the file is "checked in" to the system. The potential error with this method is accounting for files checked out over multiple days. If the file was not actually checked into VSS during a day of work, the developer's hours for that day would not be mapped to the file. Only the date on which the file was checked in would be counted in the edit list.

The next step in calculating man-hours was to determine how many hours were worked by each developer for every day in the project schedule. This process varied slightly between the two projects. Because GoTrain did not have an electronic time-tracking system during the Spanish project, hours for the project were extracted manually from paper timesheets maintained by GoTrain administration. The Spanish project had a unique cost-tracking number that was entered by each developer on his or her timesheet. Any regular or over-time hours charged to that number were collected, along with the employee's name and dates the hours were accrued. One risk in this method is the possibility of human error when manually entering data into the project database. To minimize this issue, hours manually entered were checked after entry by comparing a report of hours in the project database back to hours on the timesheets.

At the time of the Korova project, GoTrain had implemented an electronic time-tracking system. Again, the Korova project had a unique cost-tracking number, and project team members entered this number along with their hours into the electronic system. At project completion, the time-tracking database was queried using the Korova tracking number, which provided a quick list of employees, dates and hours attributed to the project.

The final step in compiling man-hours was comparing the dates and developer names for file edits to the dates and hours worked by the developers. This created a mapping of developers, hours and dates to files edited during the project. To determine hours per file, a developer's hours for a given day were attributed to the files edited by him on that day. Given the method by which the file edits and hours were collected in this study, it was impossible to accurately determine the distribution of hours over the files on a given day. Therefore, the hours were normalized by simply dividing the developer's hours for that day by the number of files he edited. For example, if a developer worked 8 hours and edited 4 files, the result was 2 hours accrued for each of the 4 files.

It is important to note that GoTrain management requested all project hours be reported in the time-tracking systems. Because GoTrain was a young company and software development processes were being introduced for the first time, management preferred that all hours required to complete the project be recorded so future project costs and schedule could be better estimated. When compiling the man-hour data for this study, over-time hours were included, and it is assumed that all developers accurately entered all hours worked for the project. Thus, the time data presented here is assumed accurate and complete.

Ultimately, there were more project hours for developers than were attributed to file modifications. Some of these hours were time developers contributed to the initial project design and planning, as well as meetings throughout the project. Additional hours may be attributed to files checked out over multiple days, the edit of database tables or stored procedures, and replication of certain Academy files for client file folders on the web server.

4.4 Error Count

Counting project errors began by reviewing the project's master issues list. Because GoTrain did not have a formal issues-tracking system, errors detected during all phases of testing were maintained in a shared spreadsheet program. At project completion, the master error list was normalized, confirming each error record took only one row in the spreadsheet, and that duplicate entries were removed. Duplicates occurring in confirmation testing as result of an issue not being fixed or otherwise reappearing were not removed from the master list.

Reported errors were then mapped to the ASPs, html files, and VB class modules. One possibility for error using this method lies in the fact that errors in the underlying class modules or database stored procedures may only manifest themselves the user's ASP interface. It is often difficult to map errors to underlying code after project completion using simply the error description and knowledge of the code base. A better approach would have been to have developers track where the error occurred and how they corrected it.

CHAPTER 5. RESULTS

One approach to evaluating productivity for software projects is to compare the project LOC to the number of man-hours required to generate it (Boehm 1981; Potok et al., 1999). This type of comparison is typical of COCOMO models used to provide basic effort and schedule estimates for software development (Boehm, 1981). Likewise, evaluating errors per LOC can provide a perspective on the accuracy of the coding effort. Both comparisons are presented here for each of the projects in this study. The natural log (ln) of the LOC and hour values were used the graphs presented here to remove point scatter. In the case of reported errors, only those affecting functionality of the page are included in these analyses. Because the graphical interface, including page layout and text, was typically handled by GoTrain design staff and not developers, errors pertaining to these issues have been excluded here.

5.1 LOC vs. Man-hours

In the Spanish project, 3545 LOC were added or modified in 64 files. The breakdown of file types and corresponding LOC are shown in Table F.1. Six developers contributed to the coding of the final product and accumulated just over 358 hours. Figure F.1 shows the LOC vs. man-hours for the Spanish project.

In this project, several files had unusually high LOC for a small number of hours, and most noticeable are the points representing the html files. In most cases, these files were created by a non-developer using an html editor, which can rapidly produce hundreds of lines of html code. Most of the html edits were on pages presenting user help information or "frequently asked questions". In these cases, text translated from English to Spanish was simply copied from an electronic text document and pasted into the html editor. This process made it possible for hundreds of lines of code to be created in a matter of minutes. Likewise, files with just a few lines of code must go through the same process, requiring similar amounts of time. The result is the near flat trend that is visible along the -.25 and 1.5 ln(man-hour) lines. For this reason, the data associated with html file edits are not included in the data comparisons between the two projects.

Most LOC added or modified by developers during the Spanish project were in the ASPs. Therefore, to narrow the scope of this study, comparisons focused specifically on ASP files between the two projects. In the Spanish project, 2493 LOC were added or edited in 31 ASPs. For the Korova project, 7496 LOC were added or edited in 44 files as seen in Table F.2. The complete listing of files and LOC can be found in Table F.3.

When looking at man-hours vs. LOC for the remaining files – ASP and VB class modules – in the Spanish project, linear regression lines can provide a better picture of the trends in the data. Figure F.2 shows ln(Man-hours) vs. ln(LOC) for the ASP and VB files edited during the Spanish project, including regression lines for each data set. It can be seen from this graph that for the 22 VB class modules, the trend of the data is relatively flat compared to the trend of the ASP. The slope of the line for the VB modules is also atypical of the standard COCOMO model (Boehm, 1981).

For VB class modules, depending on the complexity of the processing, a developer may spend significant time unit testing the functions and properties in the module. This results in higher hours for less LOC relative to ASPs. The distribution implies that increased hours contributed to VB middle-tier coding does not necessarily result in proportionately increased LOC nor is LOC a good indicator of the hours required to produce the code in VB module. The distribution for ASP files in Figure F.2 appears more typical of data in a COCOMO model (Boehm, 1981).

The ASP files in the Korova project presented interestingly different results compared to those in the Spanish project. Again, six developers contributed to the final code base, accruing just over 512 hours. Figure F.3 shows the LOC vs. man-hours for the Korova files. Most noticeable are the three groupings of points on the graphs – the first with ln(LOC) less than 2.0; the second with ln(LOC) between 2.0 and 4.0; and the third with all points greater than 4.0 ln(LOC). Figure F.4 shows the ln LOC vs. ln man-hours for Korova with each data grouping highlighted. These will be referred to as data groups 1, 2 and 3 for discussion purposes.

In data group 1, there are 10 files represented. The relatively low LOC and hours are likely due the simple modifications in these files, which were typically text changes. Only 50% of the files in this grouping were actually called out in the Detailed Design document for modification during this project.

For the nine files in data group 2, modifications typically required code changes but only to minor procedures on the page. For example, these files were edited to check date formats, remove apostrophes from text and remove audio. Of the files in this group, 67% were actually slated for edit per the Detailed Design document.

The files in the group 3 underwent major changes of functionality and display. Not only were text and procedures edited within the pages, but the overall functionality of the pages was modified, as well. Twenty-four of the 25 files in data group 3, or 96%, were outlined for edit in the Detailed Design document. Files in this group represent the most complex feature set of the Academy and supported all features outlined for modification in the Korova project – Learner Information, New Learner Training, Reports, EX/EQ, AC, Training Requirements, and Course Menu.

When considering the entire set of Korova data, the distribution is typical of software projects represented in the COCOMO model (Boehm, 1981). As can be seen in Figure F.3, hours and LOC typically increase at a relatively uniform rate. However, when looking at the distinct groups separately, the data suggest that as the complexity of file functionality increases so do hours, regardless of the LOC. The most complex files in Korova, represented in Group 3, display a steeper trend when considered independently of Groups 1 and 2 (Figure F.5). This implies that as ASP files reach a certain complexity the basic COCOMO model cannot accurately predict the number of hours required to edit or create the LOC.

The portion of the Korova files utilizing sequence-based specification fell in line with the distribution of files in Group 3. The files edited for EX/EQ and AC actually had the two highest hour values per LOC of all the files with 54.9 and 44.2 hours, respectively. These files are indicated in Figure F.5 with a square symbols at points (6.18, 4.01) and (6.29, 3.79). The EX/EQ and AC files represented 4.6% of all files edited during Korova, 13.7% of the total LOC for the project, and 15.4% of distinct Academy features. A likely reason for the escalated hours EX/EQ and AC was that coding had already begun when the final enumerations were complete. As a result, the derived requirements actually required the programmer to revisit portions of the page. Likewise, the changes to these two features represented some of the most complex logic handled in the ASPs (Townsend, 2004).

5.2 Errors vs. LOC

In the Spanish project, there is an upward trend in errors as LOC increase for the ASP files, as can be seen in Figure F.6. There were 52 reported errors during testing for the 31 ASP files. The spike in the graph at point (152, 10) represents the file used to display exam questions in the course and pre-test. The logic in this page handles the presentation of both English and Spanish questions for three test types. Likewise, this page was coded by one of GoTrain's entry-level programmers. The complex logic in the page and the work of a beginning programmer likely contributed to the high error counted reported for this file.

High error count for ASPs may be attributed to the broad knowledge base required for ASP programming. The programmers must know multiple languages (VBScript, JavaScript, html) and be able to integrate them on the page. Errors in ASPs may also be attributed to the inherent complexity of data on the pages. Errors may originate in the code on the page or may be driven by the underlying VB code or data. While an effort was made to map errors reported on the ASP pages to the underlying VB modules or data, the documentation in testing did not include specific information regarding the source of the error. As a result, the procedure for mapping errors back to source files or data may have inflated error counts for ASPs.

In the Korova files, error distribution was less straightforward. There were 74 errors reported for 44 files. While there was an upward trend in errors as LOC increased, as seen in Figure F.7, a spike of errors occurred between line counts of 360 and 500. The three files comprising the spike in the graph are associated with the Academy features Add/Edit Training Requirements and the Course Menu. While these files did not have the highest LOC, the Training Requirements and Course Menu pages display data resulting from inputs of several Academy features, which requires extensive processing at both the database level and middle tier. For example, the data on the Course Menu results from the combination of inputs from Group Training Requirements, Individual Training Requirements, EX/EQ, and AC. The file with the largest error count is Edit Training Requirements, represented by point (328, 11). This page has more user inputs options than any other Academy page, including options for required or optional training, test out, and pre test, as well as selections for course title, passing threshold, validity period, due date, and new employee due date. Complex data processing in the database and middle tier, as well as numerous input fields on the ASP appear to increase the likelihood of errors on the pages.

Just as interesting is the point with high error counts and low LOC at point (11, 7). The file, learnerqueryresults.asp, represented by this point supports the reporting feature in the Academy where users can see a summary of their current and historical training. Much like the Course Menu, the data presented on this page is a compilation of almost all Academy features – Group Training Requirements, Individual Training Requirements, EX/EQ, and Assign Completions. Additionally, My Training Report is responsible for displaying all completions accrued for the regulatory courses throughout the user's training history. The most likely cause of errors on this page is malformed data compiled at either the database level or the middle-tier. However, as previously noted, tracing issues back to the middle tier or database is almost impossible without documentation by the developer at the time of correction. Therefore, all errors manifested on this page were attributed to the ASP.

The files supporting EX/EQ and AC had five reported errors. These errors are presented in Table F.4. Only one error found in each file had been addressed in the specification document and thus should have been avoided. Selecting a Learner should display a list of the learner's Training Requirements (Requirement Trace No. 1 in Tables D.1 and D.2). It is assumed this is an accurate list of requirements, and any requirement previously removed should not appear. The list of Training Requirements was neither driven nor controlled at the page level for EX/EQ or AC, and the error seen on these pages likely originated at a different level than the ASP. However, the errors were reported during EX/EQ and AC testing, and therefore, are shown here. The remaining errors for EX/EQ and AC related to data types passed between the ASP and middle tier. Error handling and data-type checking were not addressed in the Functional Specification, Detailed Design or sequence enumerations.

CHAPTER 6. DISCUSSION

6.1 Summary

This study compared two similar software projects to evaluate productivity in a web-development environment and to determine if the use of the rigorous sequence-based enumeration process with the MSF Process Model would result in increased team software productivity and accuracy in a development cycle. Specifically, higher LOC per man-hour or fewer errors per LOC were considered indications of improved efficiency. In this case GoTrain's Korova project indeed saw higher LOC for similar number of hours indicating an increase in productivity. Likewise, the Korova project showed a reduced percentage of errors when compared to the Spanish project. Files utilizing sequence-based specification in the Korova project did not show increased productivity compared to similar files in the project, but showed a marked improvement in accuracy and correctness in use following deployment of the application.

The sequence-based specification process proved effective in identifying the requirements and directives overlooked by the project requirements document, and ultimately led to complete and accurate EX/EQ and AC features. Importantly, after the deployment of the Academy following the Korova development effort, several features, including the Course Menu, Training Requirements and Reports required rework to address errors found in the application after release to the GoTrain clients. Rework lasted two months following the initial release of the Academy, resulting in 84 additional hours accumulated for coding. Sixty-eight percent of the Group 3 files had to be edited and redeployed to address issues reported in customer feedback. However, neither AC nor EX/EQ required revision or redeployment during this time.

Additionally, Korova showed a reduction of errors as compared to Spanish. The percentage of files with reported errors in Korova was only slightly less than seen in Spanish with 50% of Korova files having at least one reported error as compared to 54.4% in Spanish. However, the average number of errors reported per file was dramatically reduced. The average error count for Korova was 9.9/KLOC, whereas Spanish had an average count of 20.8/KLOC.

The improved efficiency of the Korova project could be the result of several factors, including the team's increased familiarity with the software process during the second development cycle. Likewise, because the majority of team members worked on both projects, the shared project time could have improved team cohesiveness and efficiency. In addition, the benefit of regular team meetings became apparent during the Spanish project, and as result, project managers took more proactive roles during the Korova effort. Specifically, daily team meetings were held in which project schedule was reviewed and outlying issues were discussed. The improved communication among all team members may also have contributed to the overall project efficiency.

6.2 Recommendations for Future Study

One of the key components of the Cleanroom software process is the removal of developer unit testing and the incorporation of code review and verification (Prowell et al, 1999). It was suggested in this study that the differences in LOC per man-hour for VB modules versus ASPs might be attributed to the time required for unit testing. A recommendation for future study would be the removal of unit testing in both the middle and presentation tiers and utilize the Cleanroom process to verify the code.

One difficulty encountered in the Korova project was the integration of multiple features within one application. While features utilizing sequence-based specification demonstrated improved accuracy, integrating those with other features in the application resulted in errors during integration testing. Expanding the scope of the sequence-based specifications to include all features might ensure a better integration among the systems. Outputs or responses of one system may serve as inputs to another, and enumerating these in tandem may have an effect on accuracy of the entire application versus accuracy at the feature level.

Another finding in this study worthy of discussion is the variation of ASP effort measurements from standard COCOMO models. Results of other studies documented in (Umbers, 2002) have also noted that basic COCOMO does not work well for web-based applications. Other models have shown better results in estimating effort for web-development projects and might be valuable for productivity comparisons, specifically for ASP files, in future studies (Umbers, 2002).

CHAPTER 7. CONCLUSION

Productivity comparisons in a web-development environment can provide useful information on the state and efficiency of a project team. The metrics presented in this study show that introducing even a basic software process, such as the MSF Process Model, can provide the depth of data required to produce meaningful productivity analyses. Simply by tracking project requirements, code iterations, time and errors, it is possible to evaluate productivity in a web environment. Likewise, this study shows that increased productivity can be seen early in a web-development group after the initial introduction of software processes.

The use of sequence-based specification, a rigorous process for identifying product requirements, with the MSF Process Model can be a successful approach to creating accurate and correct product specifications in a web-development environment. This process creates a step otherwise missing in the MSF Process Model for ensuring the accuracy of product requirements. Likewise, it guarantees complete project documents specified by the MSF process - the Functional Specification and the Detailed Design. In this study, files edited using the combination of the MSF Process Model and sequence-based specification showed better accuracy in post-deployment use, including fewer errors and no required rework, than similar files created under the MSF Process Model only. While the sequence-based specification process may require additional hours on the front-end of a development effort, time can be saved at the completion of a project through reduced rework. Likewise, the product is deployed in a more reliable state than if specifications are left undefined.

The true cost of rework required in this project due to inadequate or incomplete product requirements cannot likely be measured in hours and LOC alone. The cost of customer support resulting from errors of the product and time to re-deploy the modified code are not captured in the LOC and hours-per-LOC. Likewise, it is difficult to quantify the loss due to tarnished company reputation when a faulty product is deployed. For these reasons, the incorporation of sequence-based specification likely contributes more to the overall quality and efficiency of a project than can be quantified in the metrics of this study.

REFERENCES

REFERENCES

Becker, Shirley A., Michael Deck, and Tove Janzon. "Cleanroom and Organizational Change". *Proc. 14th Pacific Northwest Software Quality Conference, Portland, OR, October 29-30, 1996.* 11 June 2004 <http://www.cleansoft.com/cleansoft_library.html#PNSQC96>

Boehm, Barry. *Software Engineering Economics*. New Jersey: Prentice-Hall, 1981.

Dyer, Michael. *The Cleanroom Approach to Quality Software Development*. New York: Wiley & Sons, 1992.

Fraser, Stephen R. G. Real-World ASP.NET – Building a Content Management System. Berkeley: Apress, 2002. *Books 24 x 7*. 14 Apr. 2004 <http://www.books24x7.com>.

Fielden, Tim. "Gone in 60 Seconds: Web App Dev Accelerated." *InfoWorld* 20 Oct. 2000. 5 Apr. 2004 <http://archive.infoworld.com/articles/mt/xml/00/10/23/001023mtwebappdev.xml>.

Henderson, Johnnie. "Why Isn't Cleanroom the Universal Software Development Methodology?" *Cross Talk - the Journal of Defensive Software Engineering* May 1995. 4 Apr. 2004 <http://stsc.hill.af.mil/crosstalk/1995/05/cleanroo.asp>

Kolawa, Adam, Wendell Hicken and Cynthia Dunlop. *Bulletproofing Web Applications*. New York: M&T-Hungry Minds, 2002. *Books 24 x 7*. 5 Apr. 2004 <http://www.books24x7.com>.

Linger, Richard C. and Carmen J. Trammell. *Cleanroom Software Engineering Reference Model Version 1.0.* CMU/SEI-96-TR-022. Pittsburgh: Software Engineering Institute, Carnegie Mellon U, Nov. 1996.

Linger, Richard C., Mark C. Paulk, and Carmen J. Trammell. *Cleanroom Software Engineering Implementation of the Capability Maturity Model (CMM^{sm}) for Software*. CMU/SEI-96-TR-023. Pittsburgh: Software Engineering Institute, Carnegie Mellon U, Dec. 1996

Mao, H. "The Box-Structure Development Method." Diss., U of Tennessee, 1993.

Microsoft Corporation. *Microsoft MCSD Self-Paced Training Kit – Analyzing Requirements and Defining Microsoft .NET Solutions Architectures*. Washington: Microsoft, 2003. *Books 24 x 7*. 5 Apr. 2004 <http://www.books24x7.com>.

---. "Microsoft Solutions Framework Version 3.0 Overview". *Microsoft Solutions Framework White Paper* June 2003. 13 June 2004 <http://msdn.microsoft.com/vstudio/teamsystem/msf/>.

--. "MSF Process Model v.3.1". *Microsoft Solutions Framework White Paper* 1 June 2002. 20 Apr. 2004 <http://msdn.microsoft.com/vstudio/teamsystem/msf/>.

Mills, Harlan D., Richard C. Linger, and A. R. Hevner. *Principles of Information Systems Analysis and Design*. Orlando, FL: Academic P, 1986.

Pandian, C. Ravindranath. Software Metrics – A Guide to Planning, Analysis, and Application. New York: Auerbach Publications, 2004. *Books 24 x 7*. 15 May 2004 <http://www.books24x7.com>.

Potok, Thomas E. and Mladen A. Vouk. "A Model of Correlated Team Behavior in a Software Development Environment." *Proceedings of the Workshop on Application Specific Software Engineering and Technology*, 1999.

Potok, Thomas E., Mladen Vouk, and Andy Rindos. "Productivity Analysis of Object-Oriented Software Developed in a Commercial Environment." *Software – Practice and Experience* 29 (1999): 833-847.

Prowell, Stacy J. "Sequence-based Software Specification." Diss. U of Tennessee, 1996.

Prowell, Stacy J. and Jesse H. Poore. "Sequence-based Software Specification of Deterministic Systems." *Software – Practice and Experience* 28 (1998): 329-344.

---. "Foundations of Sequence-Based Software Specification." *IEEE Transactions on Software Engineering* 29 (2003): 417 – 429.

Prowell, Stacy J., Carmen J. Trammell, Richard Linger, and Jess H. Poore. *Cleanroom Software Engineering: Technology and Process*. Reading, MA: Addison-Wesley, 1999.

Software Engineering Institute. *Capability Maturity Model - Integration (CMMI) for Systems Engineering (SE)/Software Engineering (SW)/Integrated Product and Process Development (IPPD), Version 1.1*. CMU/SEI-2002-TR-011. Pittsburgh: Carnegie Mellon U, Mar. 2002.

Sodhi, Jag and Prince Sodhi. *IT Project Management Handbook.* Vienna, VA: Management Concepts, 2001.

Strom, David. "Web Watch: Why is Web Development So Painful?" *SDTimes* 15 July 2000. 5 Apr. 2004 <http://www.sdtimes.com>.

Townsend, Joel. "Hampton Tilley Project Methods." White paper. Knoxville, TN: Hampton Tilley and Associates, 20 Mar 2000.

--. Personal interview. 11 June 2004.

Umbers, Paul. "Resource Estimation for Web Application Software Projects." 9 Apr. 2004. 15 May 2004 <http://internistic.typepad.com/public/ EstimatingWebApplications_Models.pdf.

APPENDICES

Appendix A. MSF Tables and Figures

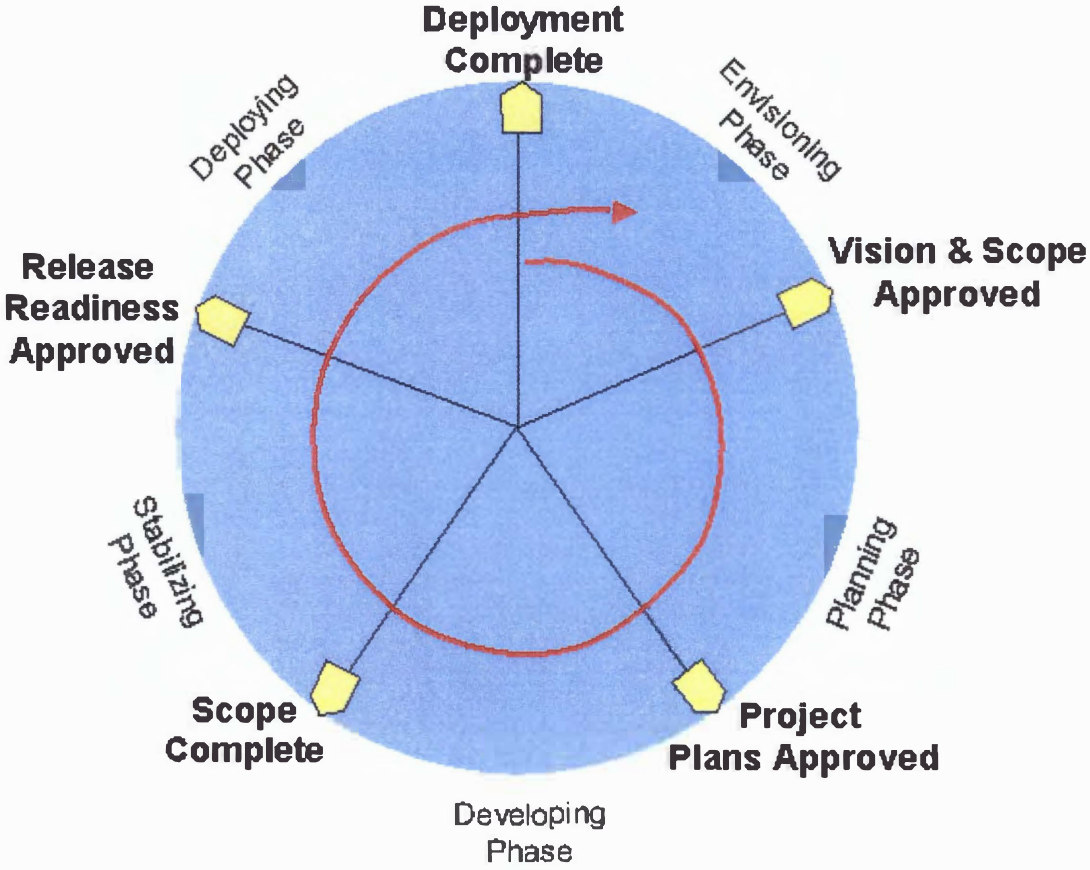
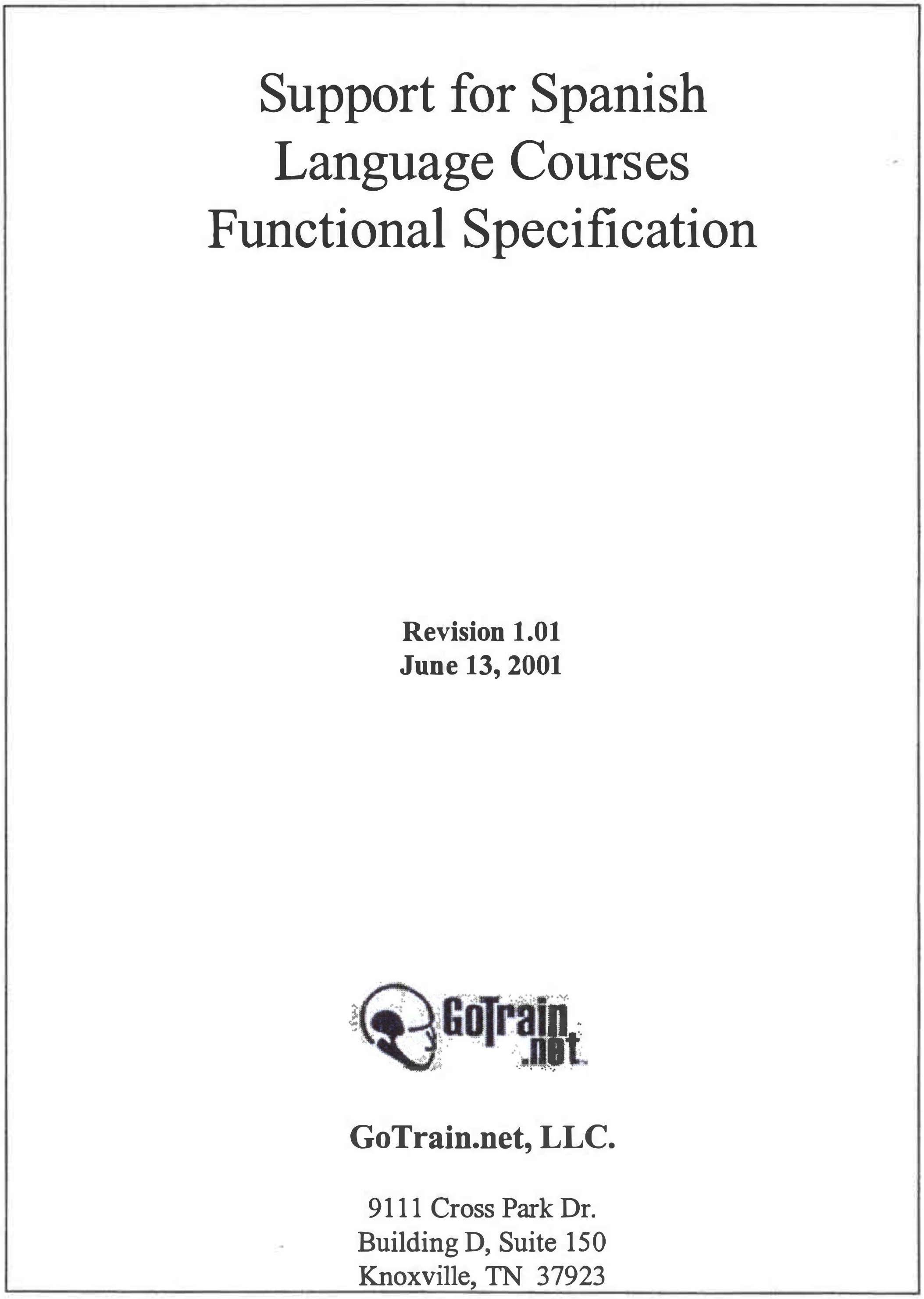


Figure A.1. The Phases and Milestones of the MSF Process Model.
(Adapted from Microsoft Corp., 2003.)

Table A.1. Process Model Phases, Milestones and Deliverables for GoTrain Projects.

Phase	Milestone	Deliverable
Envisioning Phase	Vision / Scope Approved	Functional Specification Document
Planning Phase	Project Plan Approved	Detailed Design document
Developing Phase	Scope Complete / First Use	Test Plan and Test Cases, Source code for deployment
Stabilizing Phase	Release Readiness Approved	Deployment file folder; Product documentation; Client notification plan
Deploying Phase	Deployment Complete	Files, application and online product documentation to the production web servers and database; Archive released code and files.

Appendix B. Spanish Project Documents



Support for Spanish Language Courses Functional Specification

Revision 1.01
June 13, 2001

GoTrain.net, LLC.

9111 Cross Park Dr.
Building D, Suite 150
Knoxville, TN 37923

Figure B.1. Spanish Functional Specification Title Page

THE PURPOSE OF A FUNCTIONAL SPECIFICATION

The Functional Specification document is intended to describe the specific features a proposed solution is required to include. Likewise, it is expected that the requirements be detailed an industry-standard manner. Content should include high-level statements of problems, goals and constraints. The incorporation of flow charts and UML diagrams are appropriate when they describe the overall flow of information or requirements of a system. Most importantly, the Functional Specification should produce a checklist of required and desirable features of a proposed project.

Although a Functional Specification can become very detailed in its description of a solution's requirements, it should be limited to describing the present state of the system, (if it pre-exists the project) and the final state of the system at the close of the project. Implementation details of how the system produces the desired features should be described in a formal Design Document.

The Functional Specification is one standard by which the success or failure of a project can be evaluated.

Once all stakeholders have signed off, no changes to the specification of a project are permitted without a formal change order.

PROJECT STATEMENTS

Business Problem

GoTrain.net has a market opportunity requiring support of Spanish language courses before it can be exploited. A number of courses have already been translated into Spanish, but the existing system design is expected to place undue burden on the training center administrators.

While it would be possible to simply offer the Spanish course in the current system, this would require a training center administrator to create additional training groups based on language preference. This added burden on the administrator is anticipated to be a hindrance in the sales and marketing efforts.

In order to gain wider market acceptance of Academy 2, the system will be modified to allow a training center administrator to assign only the English language version of a course. The comparable Spanish course (if available) will be presented to the learner on the course menu alongside the English; an equivalency record will be generated if the Spanish language course is taken, thus fulfilling the training assignment.

Project Vision Statement

The Academy 2 system will be modified to enhance the learning experience for bilingual learners by offering courses in Spanish, while minimizing the effort required by administrators to manage these course offerings.

Primary Project Requirements

The project should fulfill the following requirements:

- Allow the learner to choose his language of preference.
- Enable administrators to optionally pre-assign a language (default to English).
- Persist the language choice in the learner record (student table).
- Present the course menu in the selected language (if available).
- Automatically record an equivalency record upon completion of the comparable Spanish language version of the course.
- Enhance reports to show both the course taken, and the English equivalency for completions in the Spanish version.

Project Constraints

The Academy 2 application was not designed from its inception to support multiple languages. A conscious decision has been made to limit the scope of this feature, with full support envisioned for the Academy 3 release.

This interim release for Academy 2 will have the following limitations:

- Support for Spanish in the application interface will be limited; it will include:
    - The scripts necessary to support running a course, inclusive of the courses and knowledge feedbacks,
    - Supporting pages such as the welcome page, terms of use, course help, comments, and the ability to ask on-line questions of our SMEs,

    - The course menu will present the course titles in the preferred language (as available), and
    - The learner will be able to switch his language of preference on either the course menu or welcome page.
- The core system changes to support this initiative will be limited to US English and Spanish; this will not prohibit developing custom courses in any language, but the system-wide support will be limited.
- All administrative pages and the balance of the learner pages will continue to be available in US English only.
- System enhancements to support Spanish language courses have been limited in scope intentionally; the goal is to minimize schedule impact and make the Spanish courses available to customers in the near term.

Project Scope

The multiple language support is limited to a bilingual interface to support running the Spanish language courses that have been developed. The application interface will remain predominantly US English, with only the pages specifically associated with running a course being rendered in Spanish (dependent upon the language of the course selected by the learner).

Any further enhancements will be considered "out of scope".

EXISTING SYSTEM

Existing Code

There is currently no code within the system to explicitly support either Spanish courses or application interface.

Existing Data Structures

The existing data structures do not support the use of alternate languages within the Academy 2 application.

DESCRIPTION OF THE PROPOSED SYSTEM

Overview

Before a learner can access an Academy, the training center administrator must first create an associated student record. This record will contain the learner's language preference, which will default to US English.

Upon first logon to an Academy, the learner will see a welcome page in the language assigned by the training center administrator. The learner will be provided an option to change the language preference on this screen. If this option is exercised, the screen will refresh in the appropriate language.

After viewing the welcome page, the learner will proceed to the course menu. If the learner changes his preference at this time, the course menu will refresh and be displayed in the selected language. Course titles will be presented in the language of preference as available. However, if the learner's preference is Spanish and the corresponding course is not available in that language, the US English title will be displayed instead.

The user will proceed to run a course from the menu. The various screens identified under the Project Constraints topic will be presented in the language of choice whenever possible. When a course is available only in English, all elements of the course-related application pages, including any associated knowledge feedbacks will always be in English. All other screens will honor the learner's language preference.

After completing one or more courses the learner may choose "My Training Reports" from the application sidebar menu. The report will be presented in US English, with any Spanish language courses showing a completion code beside the Spanish title; a corresponding equivalency record for the English title will also appear.

Reports available to the training center administrators will likewise be presented in English. These reports will also show dual entries for any courses taken in Spanish. A completion code will appear beside the Spanish title, and a corresponding equivalency designator will appear by the English title.

Equivalencies for the English version of any Spanish courses taken will be displayed on the corresponding administrative page.

Required Data

To support the delivery of the Spanish courses under the scope defined in this document, it is necessary to track additional information:

- Language preference of the learner
- Alternate Spanish course number for English courses

REQUIRED INTERFACES

Learner Management

The learner maintenance page must be updated to allow the administrator of the training center to set the language preference.

Course Related Application Pages

Pages accessed by a learner that are directly related to running a course will be enhanced to support Spanish as an alternate language. The list of changes is:

- Scripts necessary to run a course
- Course menu
- Knowledge feedbacks (questions, learner feedback)
- Welcome page
- Login page
- Terms of use
- Course help
- Comments

- Ask on-line
- Nav Demo
- Conditions of Self-testing
- Academy sidebar (menu graphics, text)

This list does not explicitly include the courses; the translated courses are beyond the scope of this document.

Required Reports

By limiting the scope of the feature enhancements, it is possible to utilize the existing reports without changes.

FEATURE CHECKLIST

Required Features

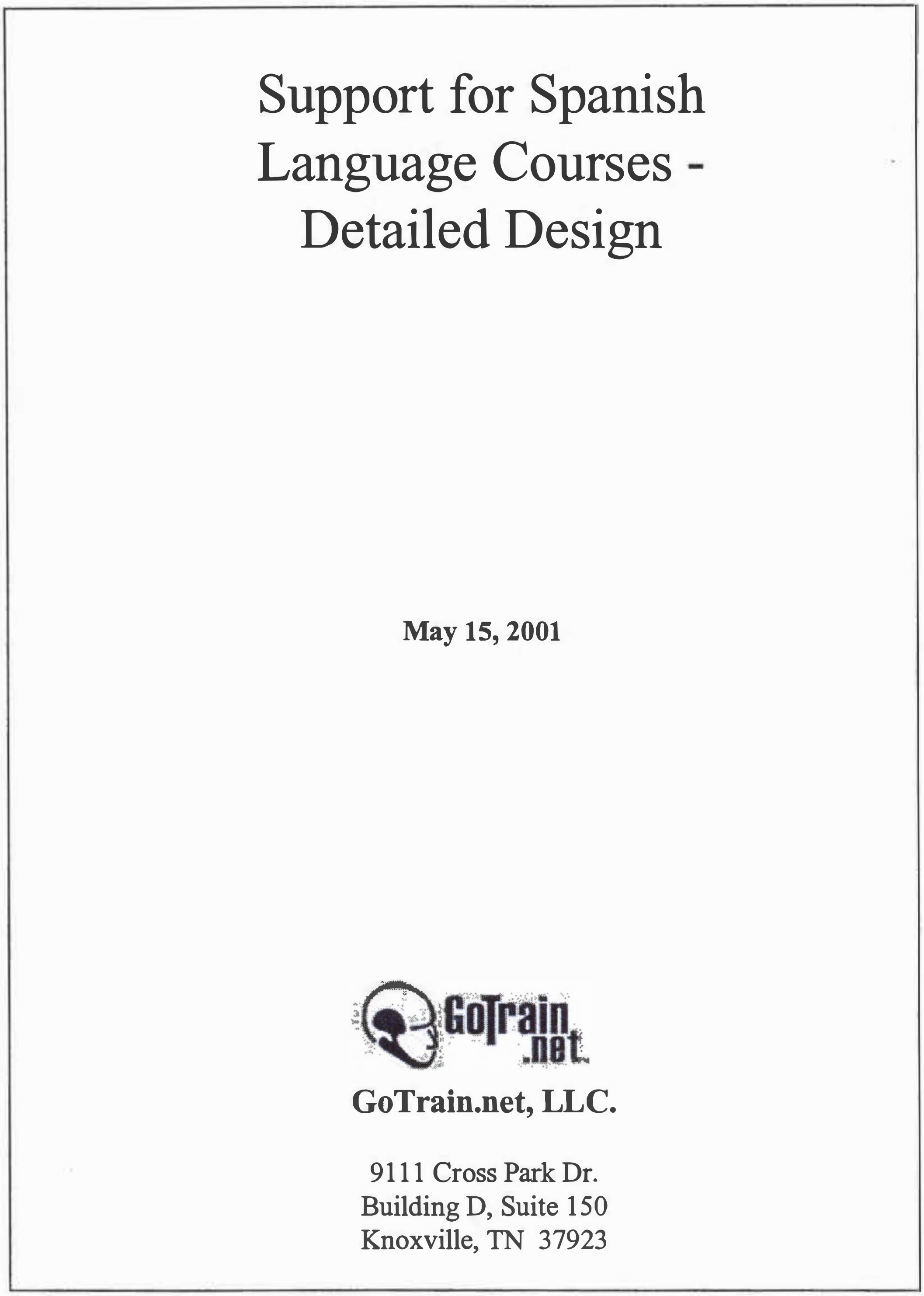
- □ The learner management page will allow the administrator to assign a language preference for the student.
- □ A learner will be able to choose his or her language preference from either the Course Menu or Login page.
- □ The language preference will be persisted in the learner record.
- □ The learner's Course Menu will display each course title in the learner's language of preference if the course is available in the preferred language. Otherwise, the course title will be displayed in English.
- □ Courses will run in the language of the associated course title as shown on the course menu.
- □ The following Academy pages will always display in the learner's language of preference:

    - Welcome page
    - Terms of Use (Academy version)
    - Navigation Demo
    - Login page
    - Course menu
    - Comments

- □ The following course-related pages will always display in the language of the displayed course
    - Course help
    - Terms of use (Course version)
    - Ask on-line
    - Knowledge feedbacks (questions, learner feedback)
    - Conditions of Self-testing
    - Academy sidebar (menu graphics, text)
- □ Completion of a Spanish course will generate an equivalency for the English version of the course.
- □ Reports will show the Spanish course completion plus the English equivalency.

Desirable Features

Features have been identified that are desirable but not required. However, these are not slated for inclusion in Academy 2 due to schedule constraints. Major feature enhancements like those identified have been reserved for Academy 3.



Support for Spanish Language Courses - Detailed Design

May 15, 2001

GoTrain.net, LLC.

9111 Cross Park Dr.
Building D, Suite 150
Knoxville, TN 37923

Figure B.2. Spanish Detailed Design Title Page

THE PURPOSE OF A DETAILED DESIGN

The Detailed Design document is the developer's blueprint. It provides precise instructions to application developers about how the user interface, business logic and data structures will be organized. The Detailed Design translates the project goals and feature set defined in the Functional Specification into an implementation plan.

Typically, the Detailed Design includes definitions of data structures, data flows and algorithms. The incorporation of images of UI prototypes, pseudo-code, code snippets and database diagrams is appropriate. Most importantly, the Detailed Design should produce a detailed specification of all features to be implemented.

The Detailed Design document is written before programming starts. It describes what functionality will be included and how the software will be structured. This document forms the basis for all future design and coding.

The designer's goal is to apply real-world technology constraints to the conceptual model and develop a model that achieves the functional requirements while operating within key constraints, such as performance goals, hardware, budget and schedule.

The ultimate goal is to create a design that is:

- Simple, while meeting the needs

- Easily understood
- Easily communicated
- Easily built
- Easily tested

Project Statements

Overview

The implementation of the Spanish Language support for Academy 2 will require participation by Web Designers, Web Developers, and a Translator. Each area of expertise will be assigned responsibility for completing the requirements associated with its specialty.

The Web Designers will be responsible for the cosmetics of the interface and incorporating the text provided by the Translator into the Spanish version of the affected pages. Web Developers will implement all application level changes, including ASP scripting, server side components, and changes to the database.

System Enhancements

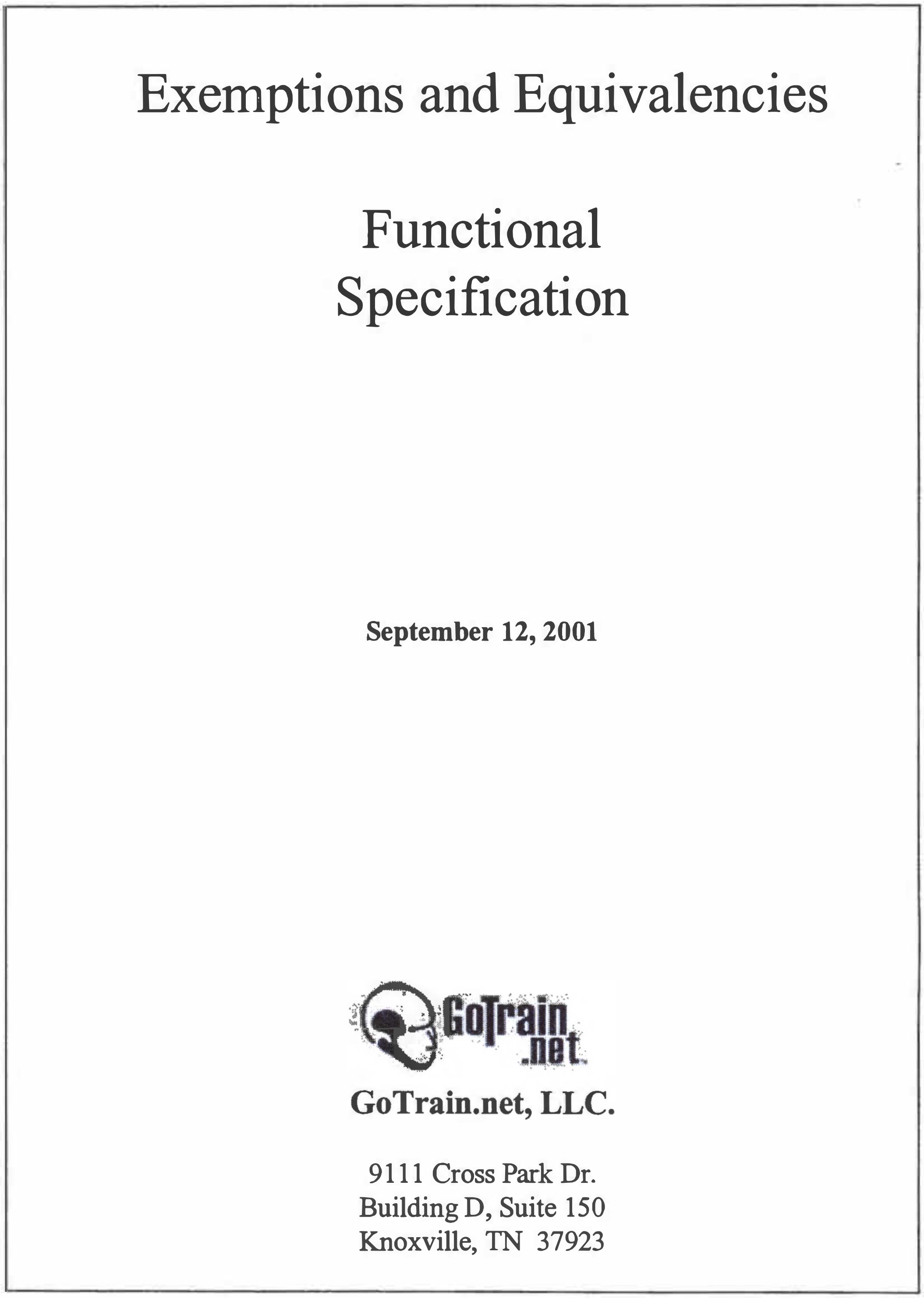
Introduction

The following topics will address specific Web pages that need to be modified to provide support for Spanish courses. Each page will be listed separately and will address the scope and responsibility of the required enhancements.

Application Enhancements

The following is a complete list of the pages that must be modified or created in order to provide support for Spanish language courses. Additionally, the changes necessary to the server side components and database are defined.

Appendix C. Functional Specification for EX/EQ and AC



Exemptions and Equivalencies

Functional
Specification

September 12, 2001

GoTrain.net, LLC.

9111 Cross Park Dr.
Building D, Suite 150
Knoxville, TN 37923

Figure C.1. EX/EQ Functional Specification Title Page

THE PURPOSE OF A FUNCTIONAL SPECIFICATION

The Functional Specification document is intended to describe the specific features, which a proposed solution is required to include. Likewise, it is expected that the requirements be detailed in an industry-standard manner. Content should include high-level statements of problems, goals and constraints. The incorporation of flow charts and UML diagrams are appropriate when they describe the overall flow of information or requirements of a system. Most importantly, the Functional Specification should produce a checklist of required and desirable features of a proposed project.

Although a Functional Specification can become very detailed in its description of a solution's requirements, it should be limited to describing the present state of the system, (if it pre-exists the project) and the final state of the system at the close of the project. Implementation details of how the system produces the desired features should be described in a formal Design Document.

The Functional Specification is one standard by which the success or failure of a project can be evaluated.

Once all stakeholders have signed off, no changes to the specification of a project are permitted without a formal change order.

PROJECT STATEMENTS

Business Problem

In our current system, equivalencies and exemptions assigned to a training requirement will never expire. Existing Equivalencies and Exemptions functionality does not support retraining and new employee training design (Korova project). The retraining functionality will use the "valid for" period to determine retraining due dates. To reduce the complexity associated with generating the Course Menu and training program performance reports, the training assignments will be de-normalized as a result of the new retraining design. The result will be a single table that includes an entry for each course requirement by learner. Because of this redesign as defined in Korova, Equivalencies and Exemptions functionality needs to be de-normalized as well to complement retraining design and meet the needs of a compliance-based LMS.

Project Vision Statement

Equivalencies and Exemptions will be learner-specific and training requirement-specific to correspond with Korova retraining functionality.

Primary Project Requirements

The project should fulfill the following requirements:

- Administrators will assign exemptions and equivalencies for training requirements currently assigned to the learner.
- The exemption and equivalency will expire upon the expiration date of the training requirement's assigned due date.

- The exempted training requirement will "reset" upon expiration of the due by date and will display on the Course Menu if a Training Requirement is still in effect.
- Exemptions and equivalencies will continue to display in reports (both current status and historical training reports.)

EXISTING SYSTEM

Existing Code

In our current system, equivalencies and exemptions assigned to a training requirement will never expire. Equivalencies and exemptions are not training requirement-specific. A learner can be exempted from a training requirement that has not been assigned.

DESCRIPTION OF THE PROPOSED SYSTEM

Equivalencies and Exemptions

When assigning exemptions or equivalencies, the administrator will select a learner name to view the learner's current required training requirements and assign an exemption or equivalency. Exemptions and equivalencies may be edited and/or deleted and will include the editable fields currently available (justification, date assigned).

When an exemption or equivalency is assigned, it will no longer appear on the learner's Course Menu but will display in reports (both learner and administrative). The exemption and equivalency will expire upon the expiration date of the training requirement's assigned due date. When an exemption/equivalency expires, the training requirement will be re-instated and will appear on the Course Menu with a due date of:

Due date=expiration date of EQ/EX + TR valid for period

At that time, the learner will be expected to take the requirement or be administratively exempted again.

If the exemption or equivalency is deleted, it will appear on the learner's Course Menu with the existing training group or individual training requirement due date.

Reports

Administrator will have the ability to run reports sorted by exemptions and equivalencies for current training as well as historical training.

Reports will state Exemption (EX) next to the training requirement and due date.

Reports will list Equivalencies (EQ) with a completion date (the date the administrator assigns to this field) in addition to the training requirement and due date.

Course Menu

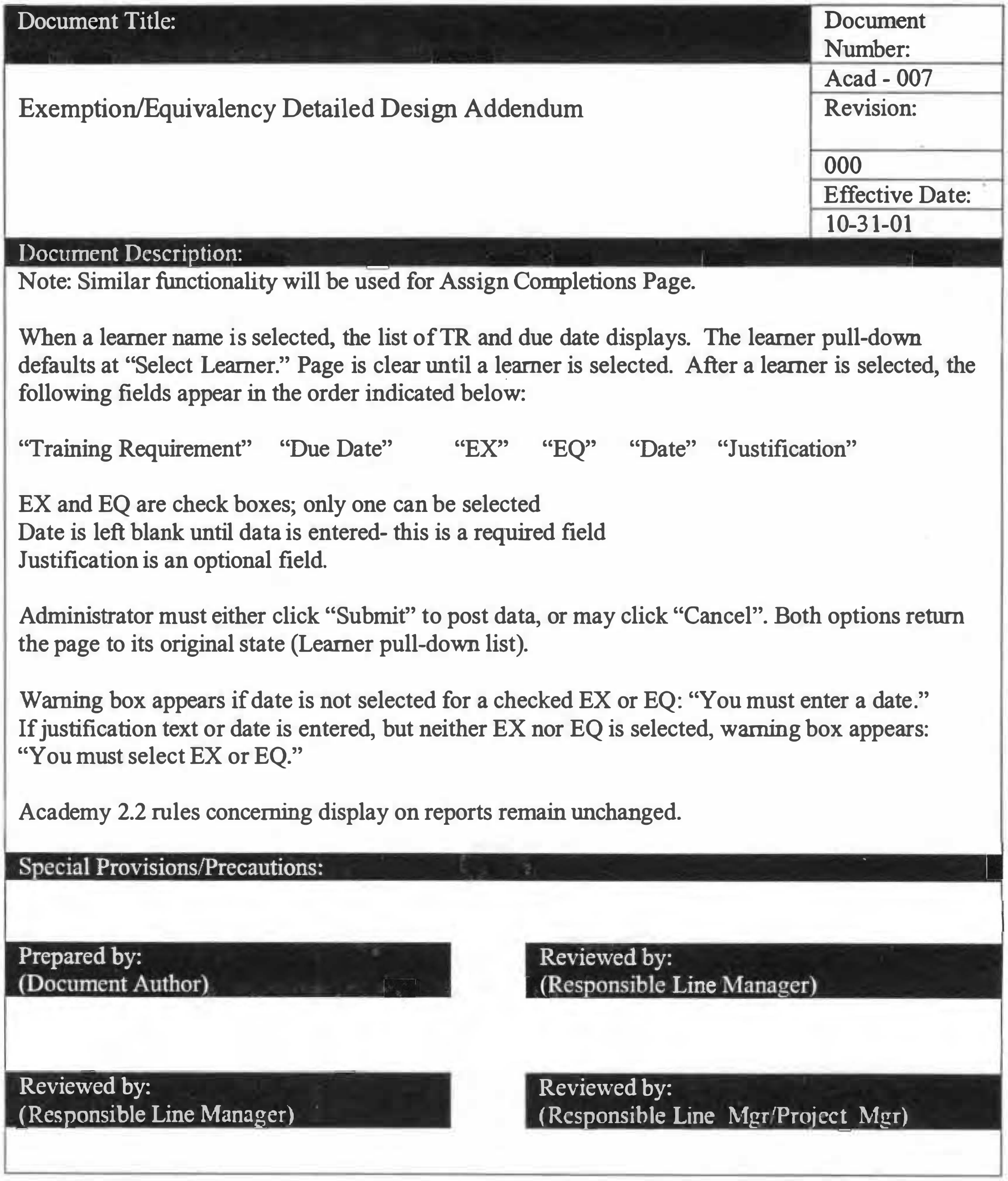
Training Requirements with an assigned exemption or equivalency will not appear on the Learner's Course Menu. It will appear in Learner Reports only.

When exemptions or equivalencies expire or are deleted, the training requirement will appear on the Course Menu's Current Training with a new due date:

Due date=expiration date of EQ/EX + TR valid for period

Email

Email reports will not include Exemption and Equivalency status. However, if an equivalency/exemption expires or is deleted, the training requirement will appear in e-mail notification, i.e., 30, 60, 90, overdue.



Document Title:	Document Number:
Exemption/Equivalency Detailed Design Addendum	Acad - 007
	Revision:
	000
	Effective Date:
	10-31-01

Document Description:

Note: Similar functionality will be used for Assign Completions Page.

When a learner name is selected, the list of TR and due date displays. The learner pull-down defaults at "Select Learner." Page is clear until a learner is selected. After a learner is selected, the following fields appear in the order indicated below:

"Training Requirement" "Due Date" "EX" "EQ" "Date" "Justification"

EX and EQ are check boxes; only one can be selected
Date is left blank until data is entered- this is a required field
Justification is an optional field.

Administrator must either click "Submit" to post data, or may click "Cancel". Both options return the page to its original state (Learner pull-down list).

Warning box appears if date is not selected for a checked EX or EQ: "You must enter a date."
If justification text or date is entered, but neither EX nor EQ is selected, warning box appears: "You must select EX or EQ."

Academy 2.2 rules concerning display on reports remain unchanged.

Special Provisions/Precautions:

Prepared by:
(Document Author)

Reviewed by:
(Responsible Line Manager)

Reviewed by:
(Responsible Line Manager)

Reviewed by:
(Responsible Line Mgr/Project Mgr)

Figure C.2. EX/EQ Functional Specification Addendum

Appendix D. Sequence-Based Specification Tables and Figures

Table D.1 Tagged Requirements from the EX/EQ Specification

Tag No.	Requirement	Affected Academy Feature
1.	The administrator will select a learner name to view the learner's current required training requirements	EX/EQ
2.	Exemptions and equivalencies may be edited and/or deleted.	EX/EQ
3.	Exemptions and equivalencies will include the editable fields currently available (justification, date assigned).	EX/EQ
4.	The course for which the exemption or equivalency is applied will no longer appear on the learner's Course Menu but will display in reports	Course Menu, Reports
5.	The exemption and equivalency will expire upon the expiration date of the training requirement's assigned due date	Training Requirements
6.	When an exemption/equivalency expires, the training requirement will be re-instated and will appear on the Course Menu with a due date of: Due date=expiration date of EQ/EX + TR valid for period	Training Requirements
7.	If the exemption or equivalency is deleted, the training requirement will appear on the learner's Course Menu with the existing training group or individual training requirement due date	Course Menu, Training Requirements
8.	Administrator will have the ability to run reports sorted by exemptions and equivalencies for current training as well as historical training	Reports
9.	Reports will state Exemption (EX) next to the training requirement and due date.	Reports
10.	Reports will list Equivalencies (EQ) with a completion date (the date the administrator assigns to this field) in addition to the training requirement and due date.	Reports
11.	Training Requirements with an assigned exemption or equivalency will not appear on the Learner's Course Menu	Course Menu

Table D.1. Continued

Tag No.	Requirement	Affected Academy Feature
12.	When exemptions or equivalencies expire or are deleted, the training requirement will appear in the Course Menu under Current Training with a new due date. The new due date equals expiration date of the EQ/EX plus the training requirement "valid for" period.	Course Menu, Training Requirements
13.	Email reports will not include Exemption and Equivalency status.	Email Notification

Table D.2 Tagged Requirements from the AC Specification

Tag No.	Requirement	Affected Academy Feature
1.	The administrator will select a learner name to view the learner's current required "other" training requirements	Assign Completions
2.	Completions may be edited and/or deleted	Assign Completions
3.	Completions will include the editable fields currently available (date assigned, score, comment)	Assign Completions
4.	Completion will appear on learner's Course Menu and in reports	Course Menu, Reports
5.	The completion will expire upon the expiration on the completion date + the valid for period of the training requirement	Training Requirements
6.	When a completion expires, the training requirement will be re-instated and will appear on the Course Menu with a due date of: Due date=expiration date of EQ/EX + TR valid for period	Course Menu
7.	If the completion is deleted, the training requirement will appear on the learner's Course Menu with the existing training group or individual training requirement due date	Course Menu, Training Requirements
8.	Administrator will have the ability to run reports sorted by completions for current training as well as historical training	Reports
9.	Reports will list Completions (C) with a completion date (the date the administrator assigns to this field) in addition to the training	Reports
10.	When a Completion expires or is deleted, the training requirement will appear on the Course Menu's Current Training with a new due date: Due date=expiration date of Completion + TR	Course Menu
11.	Email reports will include non-WBT courses with completion status	Email Notification

Table D.3. Initial Specified Stimuli for the EX/EQ and AC Systems

Stimuli	System
Select Learner	EX/EQ, AC
Enter date	EX/EQ, AC
Add EX/EQ	EX/EQ
Enter Justification	EX/EQ
Add Completion	AC
Enter Score	AC
Enter Comments	AC

Table D.4. Initial Specified Responses for the EX/EQ and AC Systems

Responses	System
Show learner training requirements	EX/EQ, AC
Course no longer shows in learner's course menu	EX/EQ
Reports will show EX or EQ for the course	EX/EQ
Course Menu and Reports will show completion for the course	AC

Table D.5 Derived Requirments for the EX/EQ System

Tag No.	Requirement	Affected Academy Feature
D1.	Only one checkbox can be selected at a time (exemption or equivalency)	EX/EQ
D2.	Checkboxes will be used to create an exemption or equivalency	EX/EQ
D3.	Submit results in post to the database and any changes stored in database tables; page refreshes to Select Learner	EX/EQ
D4.	"Cancel" cancels all previous action and resets page	EX/EQ
D5.	Resetting justification or date field has no net affect	EX/EQ
D6.	Existing functionality does not require "Justification" to be completed	EX/EQ
D7.	A date and a check for either EX or EQ must exist for data to be submitted.	EX/EQ

Table D.6 Derived Requirements for the AC System

Tag No.	Requirement	Affected Academy Feature
D1.	When a learner name is selected, the list of "other" (non-web-based) courses and due dates are displayed.	AC
D2.	Checkboxes will be used to create a completion.	AC
D3.	Submit results in post to the database and any changes stored in database tables; page refreshes to select Learner.	AC
D4.	"Cancel" cancels all previous action and resets page.	AC
D5.	Resetting "Comments", "Date" or "Score" field has no net affect.	AC
D6.	Existing functionality does not require "Comments" or "Score" to be completed.	AC
D7.	A date and check for Completion must exist for data to be submitted. If no date is entered a warning box appears. If no check is entered, page is reset back to "Select Learner".	AC

Table D.7. Sequence Enumerations for Creating EX/EQ

No.	Sequence		Response	Equivalence	Req. Trace No
Length Zero					
	Empty		NULL		D1 - Page initially shows no training requirements
Length one					
A1	Learner		Show learner training requirements		1
	Date		Illegal		1
	Justification		Illegal		1
	EX		Illegal		1
	EQ		Illegal		1
	Submit		Illegal		1
	Cancel		Illegal		1
Length Two					
Extending Sequence No. A1					
	Learner	Learner	Show learner training requirements	Learner	1
B1	Learner	Date	Enter date		3
B2	Learner	Justification	Enter Text		3
B3	Learner	EX	Checkmark in EX		D2 - Checkboxes (on/off) will be used to create an exemption or equivalency
B4	Learner	EQ	Checkmark in EQ		D2

Table D.7. Continued

No.	Sequence			Response	Equivalence	Req. Trace No
B5	Learner	Submit		Data submitted to database; return to “Select Learner”		D3 - Submit results in post to the database and any changes are stored in the database tables; page refreshes to “Select Learner”.
B6	Learner	Cancel		Return to “Select Learner”		D4 - "Cancel" cancels all previous action and resets page to “Select Learner”.
Length Three						
Extending Sequence No. B1						
	Learner	Date	Learner	Show Learner Training Requirements	Learner	1
	Learner	Date	Date	NULL	Learner Date	D5 - Resetting justification or date field has no net affect.
	Learner	Date	Justification	Enter text	Learner Date	D6 - Existing functionality does not require "Justification" to be completed.
C1	Learner	Date	EX	Checkmark in EX		D2
C2	Learner	Date	EQ	Checkmark in EQ		D2

Table D.7. Continued

No.	Sequence			Response	Equivalence	Req. Trace No
C3	Learner	Date	Submit	Warning box, "Check for EX or EQ must be entered".		D7 - A date and a check for either EX or EQ must exist for data to be submitted.
	Learner	Date	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. B2						
	Learner	Justification	Learner	Show Learner Training Requirements	Learner	1
	Learner	Justification	Date	Enter date	Learner Date	D6
	Learner	Justification	Justification	Enter text	Learner Justification	D5
	Learner	Justification	EX	Checkmark in EX	Learner EX	D6
	Learner	Justification	EQ	Checkmark in EQ	Learner EQ	D6
	Learner	Justification	Submit	Warning box, "Check for EX or EQ must be entered".	Learner Date Submit	D3
	Learner	Justification	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. B3						
	Learner	EX	Learner	Show Learner Training Requirements	Learner	1
	Learner	EX	Date	Enter Date	Learner Date EX	3
	Learner	EX	Justification	Enter text	Learner EX	D6
	Learner	EX	EX	Checkmark removed from EX	Learner	D2

Table D.7. Continued

No.	Sequence			Response	Equivalence	Req. Trace No
	Learner	EX	EQ	Checkmark removed from EX; checkmark in EQ	Learner EQ	D2
	Learner	EX	Submit	Warning box, "Date must be entered".	Learner Date Submit	D7
	Learner	EX	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. B4						
	Learner	EQ	Learner	Show Learner Training Requirements	Learner	1
	Learner	EQ	Date	Enter date	Learner Date EQ	3
	Learner	EQ	Justification	Enter text	Learner EQ	D6
	Learner	EQ	EX	Checkmark removed from EQ; checkmark in EX.	Learner EX	D2
	Learner	EQ	EQ	Checkmark removed from EQ	Learner	1, D2
	Learner	EQ	Submit	Warning box, "Date must be entered"	Learner Date Submit	D7
	Learner	EQ	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. B5						
	Learner	Submit	Learner	Show Learner Training Requirements		1
	Learner	Submit	Date	Illegal		1
	Learner	Submit	Justification	Illegal		1
	Learner	Submit	EX	Illegal		1

Table D.7. Continued

No.	Sequence				Response	Equivalence	Req. Trace No
	Learner	Submit	EQ		Illegal		1
	Learner	Submit	Submit		Illegal		1
	Learner	Submit	Cancel		Illegal		1
Extending Sequence No. B6							
	Learner	Cancel	Learner		Show Learner Training Requirements	Learner	1
	Learner	Cancel	Date		Illegal		1
	Learner	Cancel	Justification		Illegal		1
	Learner	Cancel	EX		Illegal		1
	Learner	Cancel	EQ		Illegal		1
	Learner	Cancel	Submit		Illegal		1
	Learner	Cancel	Cancel		Illegal		1
Length Four							
Extending Sequence No. C1							
	Learner	Date	EX	Learner	Illegal		1
	Learner	Date	EX	Date	Enter date	Learner Date EX	3, D5
	Learner	Date	EX	Justification	Enter text	Learner Date EX	3, D6
	Learner	Date	EX	EX	Checkmark removed from EX.	Learner Date	D1, D2
	Learner	Date	EX	EQ	Checkmark removed from EX; checkmark in EQ.	Leaner Date EQ	D1, D2

Table D.7. Continued

No.	Sequence				Response	Equivalence	Req. Trace No
D1	Learner	Date	EX	Submit	Data posted and saved in database, Course is removed from Learner Course Menu; Learner and Group reports show EX for this course		4, D7
	Learner	Date	EX	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. C2							
	Learner	Date	EQ	Learner	Illegal		1
	Learner	Date	EQ	Date	Enter date	Learner Date EQ	3, D5
	Learner	Date	EQ	Justification	Enter text	Learner Date EQ	3, D6
	Learner	Date	EQ	EX	Checkmark removed from EQ; checkmark in EX	Learner Date EX	D1, D2
	Learner	Date	EQ	EQ	Checkmark removed from EQ	Learner Date	D1, D2
D2	Learner	Date	EQ	Submit	Data posted and saved in database, Course is removed from Learner Course Menu; Learner and Group reports show EQ for this course		4, D7
	Learner	Date	EQ	Cancel	Return to "Select Learner"	Learner Cancel	D4

Table D.7. Continued

No.	Sequence					Response	Equivalence	Req. Trace No
Extending Sequence No. C3								
	Learner	EX	Submit	Learner		Show Learner Training Requirements	Learner	1
	Learner	EX	Submit	Date		Enter date	Learner Date EQ	3, D5
	Learner	EX	Submit	Justification		Enter text	Learner Date EQ	3, D6
	Learner	EX	Submit	EX		Remove checkmark from EX	Learner Date EX	D1, D2
	Learner	EX	Submit	EQ		Checkmark removed from EX; checkmark in EQ.	Learner Date	D1, D2
	Learner	EX	Submit	Submit		Warning box, "Date must be entered"	Learner EX Submit	D9
	Learner	EX	Submit	Cancel		Return to "Select Learner"	Learner Cancel	D4
Length Five								
Extending Sequence No. D1								
	Learner	Date	EX	Submit	Learner	Show Learner Training Requirements	Learner	1
	Learner	Date	EX	Submit	Date	Illegal		1
	Learner	Date	EX	Submit	Justification	Illegal		1
	Learner	Date	EX	Submit	EX	Illegal		1
	Learner	Date	EX	Submit	EQ	Illegal		1
	Learner	Date	EX	Submit	Submit	Illegal		1
	Learner	Date	EX	Submit	Cancel	Illegal		1

Table D.7. Continued

No.	Sequence					Response	Equivalence	Req. Trace No
Extending Sequence No. D2								
	Learner	Date	EQ	Submit	Learner	Show Learner Training Requirements	Learner	1
	Learner	Date	EQ	Submit	Date	Illegal		1
	Learner	Date	EQ	Submit	Justification	Illegal		1
	Learner	Date	EQ	Submit	EX	Illegal		1
	Learner	Date	EQ	Submit	EQ	Illegal		1
	Learner	Date	EQ	Submit	Submit	Illegal		1
	Learner	Date	EQ	Submit	Cancel	Illegal		1

Table D.8. Sequence Enumerations for Editing EX/EQ

No.	Sequence		Response	Equivalence	Req. Trace No.
Length Zero					
	Empty		NULL		D1
Length one					
A1	Learner		Show learner training requirements		1
	Date		Illegal		1
	Justification		Illegal		1
	Eold		Illegal		1
	Enew		Illegal		1
	Submit		Illegal		1
	Cancel		Illegal		
Length Two					
Extending Sequence No. A1					
	Learner	Learner	Illegal		1
B1	Learner	Date	Enter new date		3
B2	Learner	Justification	Enter/modify Text		3
B3	Learner	Eold	Check mark removed from Eold		D2
B4	Learner	Enew	Check mark in Enew, removes checkmark in Eold		D1, D2
B5	Learner	Submit	Data submitted to database; return to "Select Learner"		D3
B6	Learner	Cancel	Return to "Select Learner"		D4

Table D.8. Continued

No	Sequence			Response	Equivalence	Req. Trace No.
Length Three						
Extending Sequence No. B1						
	Learner	Date	Learner	Illegal		1
	Learner	Date	Date	Enter Date	Learner Date	D5
	Learner	Date	Justification	Enter text	Learner Date	D6
	Learner	Date	Eold	Checkmark removed from Eold	Learner Eold	D2
	Learner	Date	Enew	Checkmark in Enew	Learner Enew	D2, D5
C1	Learner	Date	Submit	Posts new date data, no change in Course Menu, changes EX/EQ reports		2, 4
	Learner	Date	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. B2						
	Learner	Justification	Learner	Illegal		1
	Learner	Justification	Date	Enter date	Learner Date	D6
	Learner	Justification	Justification	Enter text	Learner Justification	D6
	Learner	Justification	Eold	Removes Eold checkmark	Learner Eold	D2
	Learner	Justification	Enew	Checkmark in Enew	Learner Enew	D2
C2	Learner	Justification	Submit	Posts new justification data, page refreshes, no change in Course Menu or Reports		
	Learner	Justification	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. B3						
	Learner	Eold	Learner	Illegal		1
	Learner	Eold	Date	Enter date	Learner Date Eold	D5
	Learner	Eold	Justification	Enter text	Learner Justification Eold	3
	Learner	Eold	Eold	Checkmark in Eold	Learner	D1, D2
	Learner	Eold	Enew	Checkmark in Enew	Learner Enew	D1, D2

Table D.8. Continued

No	Sequence			Response	Equivalence	Req. Trace No.
C3	Learner	Eold	Submit	Removes checkmark in Eold and posts data to the database; Training requirement appears on Course Menu, and Reports are updated.		D1, D2, D3
	Learner	Eold	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. B4						
	Learner	Enew	Learner	Illegal		1
	Learner	Enew	Date	Enter date	Learner Date Enew	D5, D7
	Learner	Enew	Justification	Enter text	Learner Justification Enew	3
	Learner	Enew	Eold	Checkmark in Eold	Learner	D1, D2
C4	Learner	Enew	Enew	Checkmark removed from Enew		
C5	Learner	Enew	Submit	Submits data to database. Changes TR from Eold to Enew, reports updated		
	Learner	Enew	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. B5						
	Learner	Submit	Learner	Show Learner Training Requirements	Learner	1
	Learner	Submit	Date	Illegal		1
	Learner	Submit	Justification	Illegal		1
	Learner	Submit	Eold	Illegal		1
	Learner	Submit	Enew	Illegal		1
	Learner	Submit	Submit	Illegal		1
	Learner	Submit	Cancel	Illegal		1
Extending Sequence No. B6						
	Learner	Cancel	Learner	Show Learner Training Requirements	Learner	1, D4
	Learner	Cancel	Date	Illegal		1

Table D.8. Continued

No	Sequence				Response	Equivalence	Req. Trace No.
	Learner	Cancel	Justification		Illegal		1
	Learner	Cancel	Eold		Illegal		1
	Learner	Cancel	Enew		Illegal		1
	Learner	Cancel	Submit		Illegal		1
	Learner	Cancel	Cancel		Illegal		1
Length Three							
Extending Sequence No. C1							
	Learner	Date	Submit	Learner	Show Learner Training Requirements	Learner	1
	Learner	Date	Submit	Date	Illegal		1
	Learner	Date	Submit	Justification	Illegal		1
	Learner	Date	Submit	Eold	Illegal		1
	Learner	Date	Submit	Enew	Illegal		1
	Learner	Date	Submit	Submit	Illegal		1
	Learner	Date	Submit	Cancel	Illegal		1
Extending Sequence No. C2							
	Learner	Justification	Submit	Learner	Show Learner Training Requirements	Learner	1
	Learner	Justification	Submit	Date	Illegal		1
	Learner	Justification	Submit	Justification	Illegal		1
	Learner	Justification	Submit	Eold	Illegal		1
	Learner	Justification	Submit	Enew	Illegal		1
	Learner	Justification	Submit	Submit	Illegal		1
	Learner	Justification	Submit	Cancel	Illegal		1
Extending Sequence No. C3							
	Learner	Eold	Submit	Learner	Show Learner Training Requirements	Learner	1
	Learner	Eold	Submit	Date	Illegal		1

Table D.8. Continued

No	Sequence				Response	Equivalence	Req. Trace No.
	Learner	Eold	Submit	Justification	Illegal		1
	Learner	Eold	Submit	Eold	Illegal		1
	Learner	Eold	Submit	Enew	Illegal		1
	Learner	Eold	Submit	Submit	Illegal		1
	Learner	Eold	Submit	Cancel	Illegal		1
Extending Sequence No. C4							
	Learner	Enew	Enew	Learner	Equivalent to Learner	Learner	1
	Learner	Enew	Enew	Date	Enter Date	Learner Enew Enew	D2, D5
	Learner	Enew	Enew	Justification	Enter test	Learner Enew Enew	D2, D5
	Learner	Enew	Enew	Eold	Checkmark in Eold	Learner Eold	D2
	Learner	Enew	Enew	Enew	Checkmark in Enew	Learner Enew	D2
	Learner	Enew	Enew	Submit	Submit data; return to select Learner	Learner Eold Submit	2, D3
	Learner	Enew	Enew	Cancel	Return to Select Learner	Learner Cancel	D4
Extending Sequence No. C5							
	Learner	Enew	Submit	Learner	Show Learner Training Requirements	Learner	1
	Learner	Enew	Submit	Date	Illegal		1
	Learner	Enew	Submit	Justification	Illegal		1
	Learner	Enew	Submit	Eold	Illegal		1
	Learner	Enew	Submit	Enew	Illegal		1
	Learner	Enew	Submit	Submit	Illegal		1
	Learner	Enew	Submit	Cancel	Illegal		1

Table D.9 Sequence Enumerations for Creating AC

No.	Sequence	Response	Equivalence	Req. Trace No.
Length Zero				
1	Empty	Null		1
Length One				
A1	Learner	Show Learner Training Requirements		D1 - When a learner name is selected, the list of "other" (non-web-based) courses and due dates are displayed
	Complete	Illegal		1
	Date	Illegal		1
	Score	Illegal		1
	Comments	Illegal		1
	Submit	Illegal		1
	Cancel	Illegal		1
Length Two				
Extending Sequence No. A1				
	Learner Learner	Illegal		1
B1	Learner Complete	Checkmark in Completed		D2 - Checkboxes will be used to create a completion
B2	Learner Date	Enter date		3
B3	Learner Score	Enter score		3
B4	Learner Comments	Enter text		3

Table D.9 Continued

No.	Sequence			Response	Equivalence	Req. Trace No.
B5	Learner	Submit		Data submitted to database; return to "Select Learner"		D3 - Submit results in post to the database and any changes are stored in the database tables.
B6	Learner	Cancel		Return to "Select Learner"		D4 - "Cancel" cancels all previsou action and returns to "Select Learner"
Length Three						
Extending Sequence No. B1						
	Learner	Complete	Learner	Illegal		1
	Learner	Complete	Complete	Removes checkmark from Complete	Learner	2, D2
C1	Learner	Complete	Date	Enter Date		3, D5 - Resetting "Comments", "Date" or "Score" field has no net effect.
	Learner	Complete	Score	Enter score	Learner Complete	3, D6 - Existing functionality does not require "Comments" or "Score" to be completed.
	Learner	Complete	Comments	Enter text	Learner Complete	3, D6

Table D.9 Continued

No.	Sequence			Response	Equivalence	Req. Trace No.
C2	Learner	Complete	Submit	Warning box: "You must enter a date."		D7 - A date and check for Completion must exist for data to be submitted. If no date is entered a warning box appears. If no check is entered, page is reset back to "Select Learner".
	Learner	Complete	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. B2						
	Learner	Date	Learner	Illegal		1
	Learner	Date	Complete	Checkmark in Completed	Learer Complete Date	3, D2
	Learner	Date	Date	Enter Date	Learner Date	3, D5
	Learner	Date	Score	Enter Score	Learner Date	3, D6
	Learner	Date	Comments	Enter Text	Learner Date	3, D6
	Learner	Date	Submit	Return to "Select Learner"	Learner Submit	D7
	Learner	Date	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. B3						
	Learner	Score	Learner	Illegal		
	Learner	Score	Complete	Checkmark in Completed	Learner Complete	D2, D6
	Learner	Score	Date	Enter Date	Learner Date	D6
	Learner	Score	Score	Enter Score	Learner Score	3, D5
	Learner	Score	Comments	Enter text	Learner	3, D6
	Learner	Score	Submit	Return to "Select Learner"	Learner Submit	D7

Table D.9 Continued

No.	Sequence			Response	Equivalence	Req. Trace No.
	Learner	Score	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. B4						
	Learner	Comments	Learner	Illegal		1
	Learner	Comments	Complete	Checkmark in Completed	Learner Complete	D2, D6
	Learner	Comments	Date	Enter Date	Learner Date	3, D6
	Learner	Comments	Score	Enter Score	Learner Score Comments	3
	Learner	Comments	Comments	Enter Text	Learner Comments	3, D5
	Learner	Comments	Submit	Return to "Select Learner"	Learner Submit	D7
	Learner	Comments	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. B5						
	Learner	Submit	Learner	Show Learner Training Requirements	Learner	1
	Learner	Submit	Complete	Illegal		1
	Learner	Submit	Date	Illegal		1
	Learner	Submit	Score	Illegal		1
	Learner	Submit	Comments	Illegal		1
	Learner	Submit	Submit	Illegal		1
	Learner	Submit	Cancel	Illegal		1
Extending Sequence no. B6						
	Learner	Cancel	Learner	Show Learner Training Requirements	Learner	1
	Learner	Cancel	Complete	Illegal		1
	Learner	Cancel	Date	Illegal		1
	Learner	Cancel	Score	Illegal		1
	Learner	Cancel	Comments	Illegal		1

Table D.9 Continued

No.	Sequence				Response	Equivalence	Req. Trace No.
	Learner	Cancel	Submit		Illegal		1
	Learner	Cancel	Cancel		Illegal		1
Length Four							
Extending Sequence No.C1							
	Learner	Complete	Date	Learner	Illegal		1
	Learner	Complete	Date	Complete	Checkmark removed from Completed	Learner Date	2, D2
	Learner	Complete	Date	Date	Enter Date	Learner Complete Date	3, D5
	Learner	Complete	Date	Score	Enter Score	Learner Complete Date	3, D6
	Learner	Complete	Date	Comments	Enter text	Learner Complete Date	3, D6
D1	Learner	Complete	Date	Submit	Data submitted to database; return to "Select Learner"		2, D3
	Learner	Complete	Date	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. C2							
	Learner	Complete	Submit	Learner	Illegal		1
	Learner	Complete	Submit	Complete	Checkmark removed from Completed	Learner	D2
	Learner	Complete	Submit	Date	Enter Date	Learner Complete Date	3, D6
	Learner	Complete	Submit	Score	Enter Score	Learner Complete Score	3, D5
	Learner	Complete	Submit	Comments	Enter Text	Learner Complete Comments	3, D5
	Learner	Complete	Submit	Submit	Warning box: "You must enter a date."	Learner Complete Submit	D7
	Learner	Complete	Submit	Cancel	Return to "Select Learner"	Learner Cancel	D4

Table D.9 Continued

No.	Sequence	Response	Equivalence	Req. Trace No.
Length Five				
Extending Sequence No. C3				
	Learner Complete Date Submit Learner	Show Learner Training Requirements	Learner	1
	Learner Complete Date Submit Complete	Illegal		1
	Learner Complete Date Submit Date	Illegal		1
	Learner Complete Date Submit Score	Illegal		1
	Learner Complete Date Submit Comments	Illegal		1
	Learner Complete Date Submit Submit	Illegal		1
	Learner Complete Date Submit Cancel	Illegal		1

Table D.10 Sequence Enumerations for Editing AC

No.	Sequence	Response	Equivalence	Req. Trace No.
Length Zero				
1	Empty		Null	
Length One				
A1	Learner	Show Learner Training Requirements		1, D1
	Complete	Illegal		1
	Date	Illegal		1
	Score	Illegal		1
	Comments	Illegal		1
	Submit	Illegal		1
	Cancel	Illegal		1
Length Two				
Extending Sequence No. A1				
	Learner Learner	Illegal		1
B1	Learner Complete	Remove checkmark from complete		D2
B2	Learner Date	Enter date		2, 3
B3	Learner Score	Enter score		2, 3
B4	Learner Comments	Enter text		2, 3
B5	Learner Submit	Data submitted to database; return to "Select Learner"		D3
B6	Learner Cancel	Return to "Select Learner"		D4
Length Three				
Extending Sequence No. B1				
	Learner Complete Learner	Illegal		1

Table D.10 Continued

No.	Sequence			Response	Equivalence	Req. Trace No.
	Learner	Complete	Complete	Checkmark in completed	Learner complete	2, D2
	Learner	Complete	Date	Enter date	Learner complete	2, D5
	Learner	Complete	Score	Enter score	Learner complete	2, D5
	Learner	Complete	Comments	Enter text	Learner complete	2, D5
C1	Learner	Complete	Submit	Data submitted to database to remove completion; return to "Select Learner"		2, D2, D3
	Learner	Complete	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. B2						
	Learner	Date	Learner	Illegal		1
	Learner	Date	Complete	Remove checkmark from complete	Learner Complete Date	2, D2, D5
	Learner	Date	Date	Enter Date	Learner Date	D5
	Learner	Date	Score	Enter Score	Learner Score	D5
	Learner	Date	Comments	Enter Text	Learner Comments	D5
	Learner	Date	Submit	Data submitted to database; return to "Select Learner"	Learner Submit	2, D3, D5
	Learner	Date	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. B3						
	Learner	Test Score	Learner	Illegal		1
	Learner	Test Score	Complete	Remove checkmark from complete	Learner Complete Score	2, D2, D5
	Learner	Test Score	Date	Enter Date	Learner Date	D5
	Learner	Test Score	Score	Enter Score	Learner Score	D5
	Learner	Test Score	Comments	Enter Text	Learner Comments	D5

Table D.10 Continued

No.	Sequence			Response	Equivalence	Req. Trace No.
	Learner	Test Score	Submit	Data submitted to database; return to "Select Learner"	Learner Submit	2, D3, D5
	Learner	Test Score	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. B4						
	Learner	Comments	Learner	Illegal		1
	Learner	Comments	Complete	Remove checkmark from complete	Learner Complete Comments	2, D2, D5
	Learner	Comments	Date	Enter Date	Learner Date	D5
	Learner	Comments	Test Score	Enter Score	Learner Score	D5
	Learner	Comments	Comments	Enter Text	Learner Comments	D5
	Learner	Comments	Submit	Data submitted to database; return to "Select Learner"	Learner Submit	2, D3, D5
	Learner	Comments	Cancel	Return to "Select Learner"	Learner Cancel	D4
Extending Sequence No. B5						
	Learner	Submit	Learner	Show Learner Training Requirments	Learner	1, D1
	Learner	Submit	Complete	Illegal		1
	Learner	Submit	Date	Illegal		1
	Learner	Submit	Test Score	Illegal		1
	Learner	Submit	Comments	Illegal		1
	Learner	Submit	Submit	Illegal		1
	Learner	Submit	Cancel	Illegal		1

Table D.10 Continued

No.	Sequence				Response	Equivalence	Req. Trace No.
Length Four							
Extending Sequence C1							
	Learner	Complete	Submit	Learner	Show Learner Training Requirmenets	Learner	1, D1
	Learner	Complete	Submit	Complete	Illegal		1
	Learner	Complete	Submit	Date	Illegal		1
	Learner	Complete	Submit	Test Score	Illegal		1
	Learner	Complete	Submit	Comments	Illegal		1
	Learner	Complete	Submit	Submit	Illegal		1
	Learner	Complete	Submit	Cancel	Illegal		1

Table D.11. Descriptions of Stimuli for the EX/EQ System

Stimulus	Description	Requirement No.
Learner	Displays Training Requirements for Learner to whom the EX/EQ is being assigned.	1, D-1
Date	The date the EX/EQ was granted. This is a required field	3, D-4
Justification	Text explanation of the reason for the EX/EQ. This is an optional field.	3, D-5
Exemption (EX)	Check box indicating an exemption.	2, D-3
Equivalency (EQ)	Check box indicating an equivalency.	2, D-3
Submit	Submits the data on the page to the middle-tier for processing and storage into the database.	D-6
Cancel	Clears any data entry on the page and returns the administrator to the learner selection page.	D-6

Table D.12. Description of Responses in the EX/EQ System

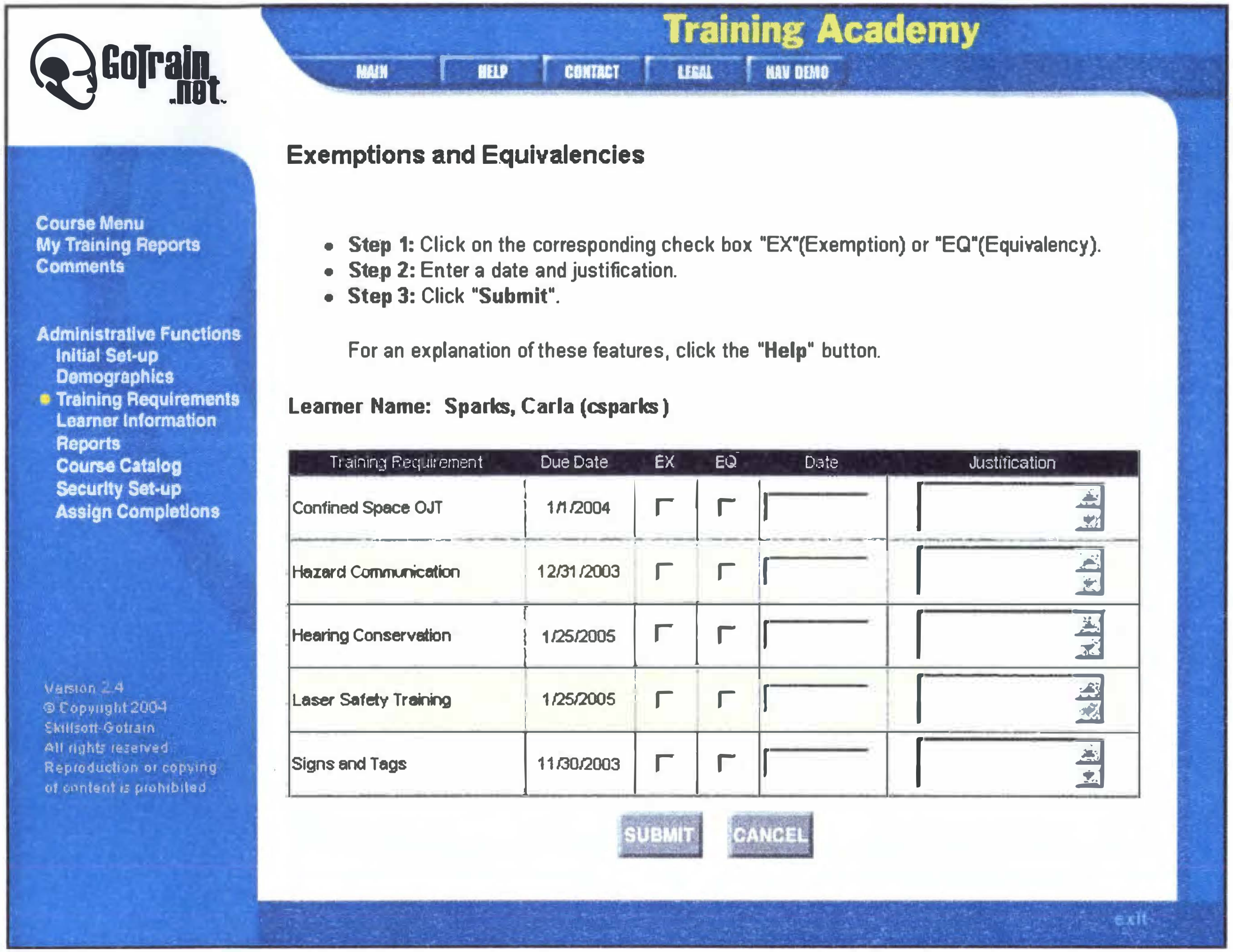
Response	Description	Requirement No.
Show Learner Training Requirements	Display courses assigned to the learner for whom the EX/EQ is to be assigned.	1
Checkmark in EX or EQ; Remove checkmark from EX or EQ	Select either EX or EQ. This check box control works like a radio button; only one box (EQ or EX) can be checked at a time.	D-3
Enter text or Enter date	Display information entered by the administrator.	2, 3, D-4, D-5
Return to "Select Learner"	Return to the "Select Learner" page where the sequence began. No data is submitted to the database.	D-6
Data submitted to database; Return to "Select Learner"	Data is submitted to the middle-tier for processing although no net changes were made. The administrator is returned to the "Select Learner" page.	D-6
Warning Box "[Text]"	Warning "popup" box is displayed with the administrator attempts to submit page without the required data.	D-7
Data posted and saved in database. Course is removed from Course Menu; Learner and Group reports show [EX or EQ] for this course. Return to "Select Learner".	Exemption or Equivalency successfully added to the database. Descriptions of the effects on other Academy features are included here for reference.	4, 7, 9, 10, 11, 13, D-6

Table D.13. Description of Stimuli for the AC System

Stimulus	Description	Requirement No.
Learner	Displays non-web-based Training Requirements for Learner to whom the AC is being assigned.	1, D-1
Date	The date the AC was granted. This is a required field.	3, D-5
Comments	Text explanation of the reason for the AC. This is an optional field.	3, D-5, D-6
Complete	Check box indicating a completion.	2, D-2
Score	Number field indicating score of the exam/quiz for the completion, if applicable. This is an optional field.	3, D-5, D-6
Submit	Submits the data on the page to the middle-tier for processing and storage into the database.	D-3
Cancel	Clears any data entry on the page and returns the administrator to the learner selection page.	D-4

Table D.14. Description of Responses in the AC System

Response	Description	Requirement No.
Show Learner Training Requirements	Display non-web-based courses assigned to the learner for whom the AC is being assigned.	1, D-1
Checkmark in AC; Remove checkmark from AC	Select AC. This check box control in either off or on.	D-2
Enter text, score or date	Display information entered by the administrator.	2, 3, D-5, D6
Return to "Select Learner"	Return to the "Select Learner" page where the sequence began. No data is submitted to the database.	D-4
Data submitted to database; Return to "Select Learner"	Data is submitted to the middle-tier for processing. A completion is either assigned, edited or removed. The administrator is returned to the "Select Learner" page.	3, D-6
Warning Box "[Text]"	Warning "popup" box is displayed with the administrator attempts to submit page with a completion but without the required date.	D-7



Exemptions and Equivalencies

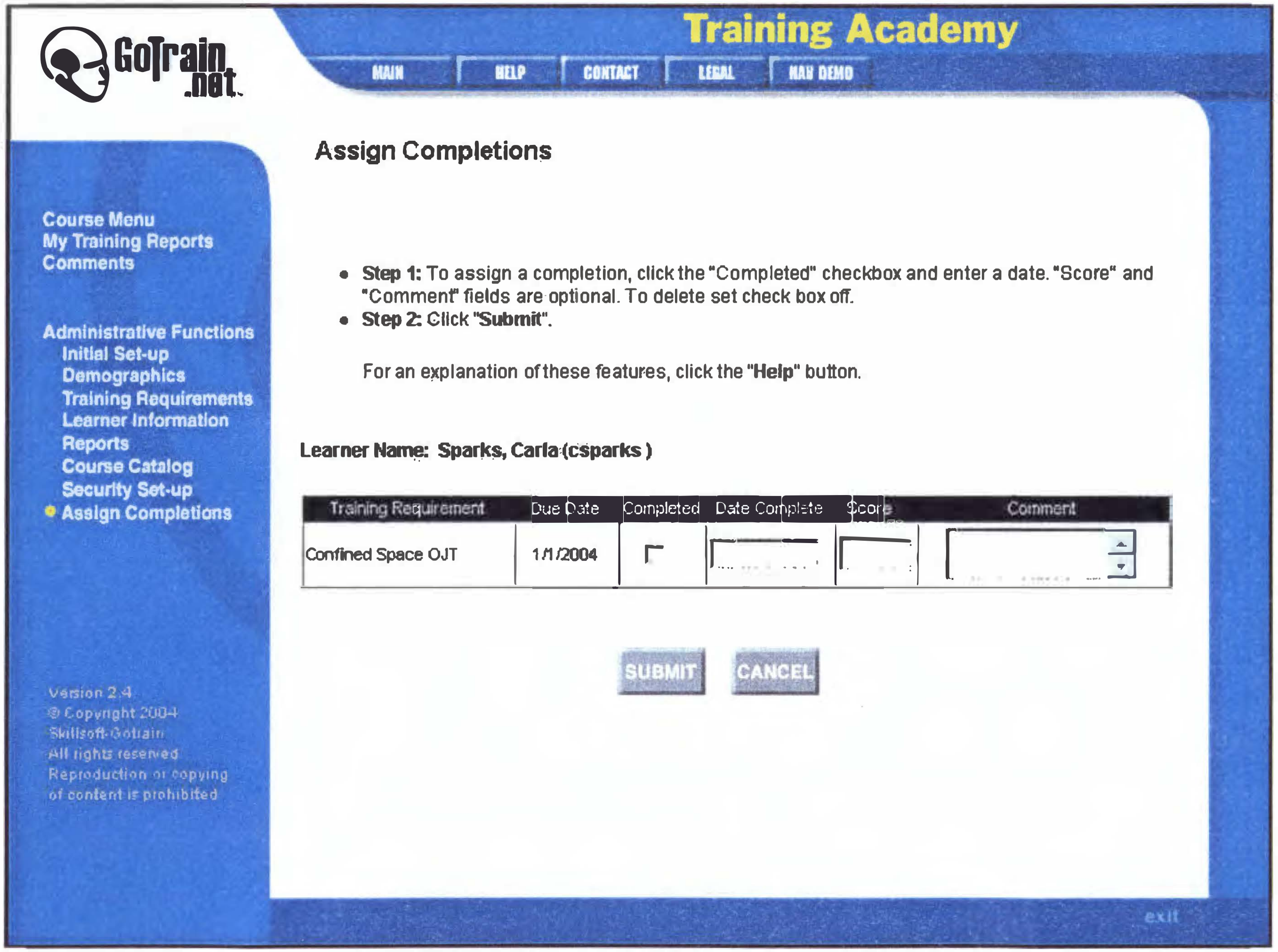
- **Step 1:** Click on the corresponding check box "EX"(Exemption) or "EQ"(Equivalency).
- **Step 2:** Enter a date and justification.
- **Step 3:** Click **"Submit"**.

For an explanation of these features, click the **"Help"** button.

Learner Name: Sparks, Carla (csparks)

Training Requirement	Due Date	EX	EQ	Date	Justification
Confined Space OJT	1/1/2004	☐	☐		
Hazard Communication	12/31/2003	☐	☐		
Hearing Conservation	1/25/2005	☐	☐		
Laser Safety Training	1/25/2005	☐	☐		
Signs and Tags	11/30/2003	☐	☐		

Figure D.1. EX/EQ Page in the GoTrain Academy



Assign Completions

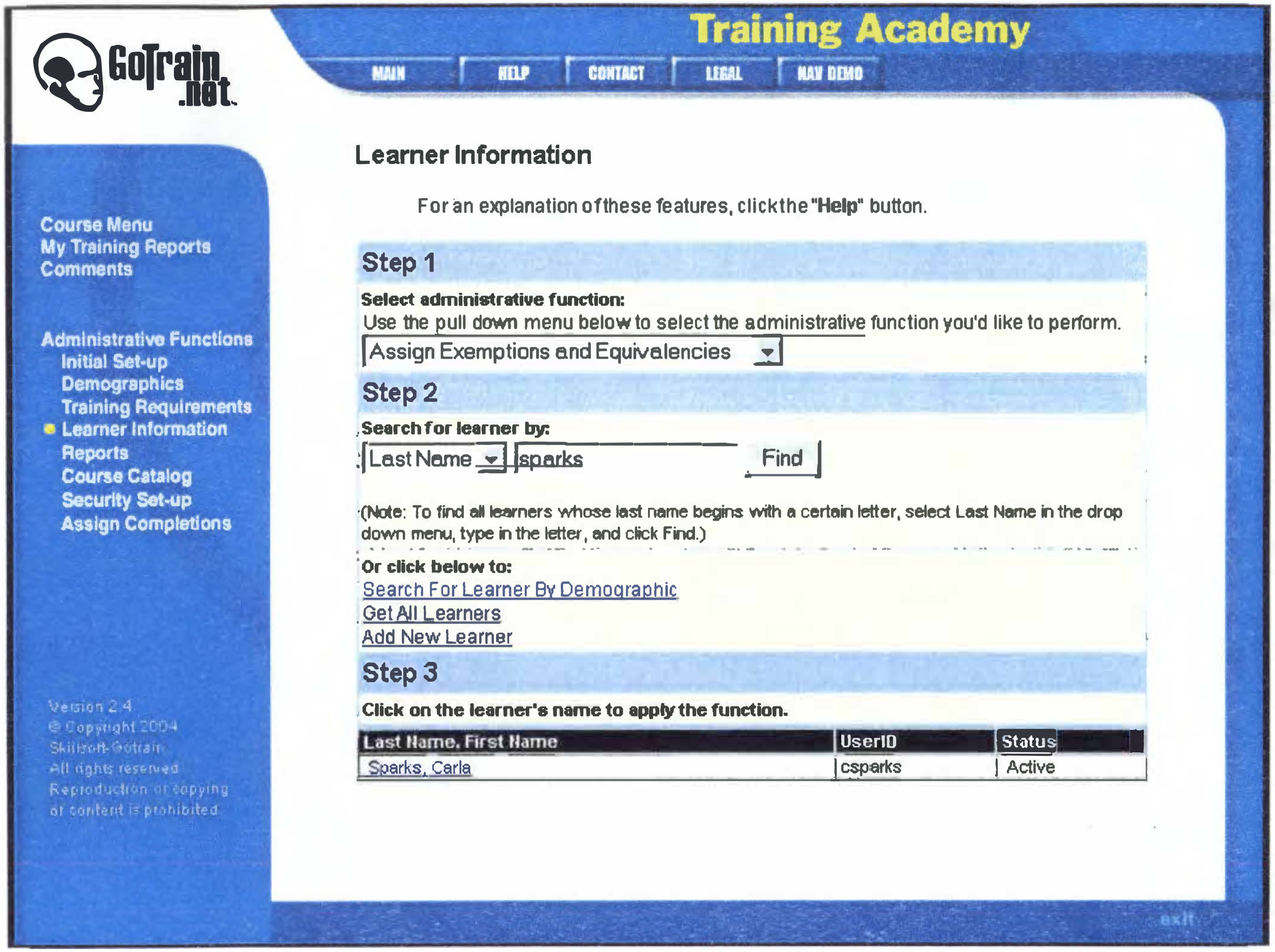
- **Step 1:** To assign a completion, click the "Completed" checkbox and enter a date. "Score" and "Comment" fields are optional. To delete set check box off.
- **Step 2:** Click "**Submit**".

For an explanation of these features, click the "**Help**" button.

Learner Name: Sparks, Carla (csparks)

Training Requirement	Due Date	Completed	Date Complete	Score	Comment
Confined Space OJT	1/1/2004	□			

Figure D.2. AC Page in the GoTrain Academy



Learner Information

For an explanation of these features, click the "**Help**" button.

Step 1

Select administrative function:

Use the pull down menu below to select the administrative function you'd like to perform.

Assign Exemptions and Equivalencies

Step 2

Search for learner by:

Last Name | sparks | Find

(Note: To find all learners whose last name begins with a certain letter, select Last Name in the drop down menu, type in the letter, and click Find.)

Or click below to:

Search For Learner By Demographic
Get All Learners
Add New Learner

Step 3

Click on the learner's name to apply the function.

Last Name, First Name	UserID	Status
Sparks, Carla	csparks	Active

Figure D.3. Learner Information Page in the GoTrain Academy

Table D.15. Canonical Sequence Analysis for Creating EX/EQ

No.	Canonical Sequence	Description	State Variables	Value before Current Stimulus	Value after Current Stimulus
1	Empty	-	-	-	-
A1	Learner	Administrator has selected a Learner and the Learner's Training Requirements are displayed	Page Training Requirements	Select Learner None	EX/EQ Displayed
B1	Learner Date	Training Requirements are displayed, and a date has been entered for an EX or EQ on the screen	Page Training Requirements Date	EX/EQ Displayed Empty	EX/EQ Displayed Entered
B2	Learner Justification	Training Requirements are displayed, and the justification text for EX or EQ has been entered on the screen	Page Training Requirements Justification	EX/EQ Displayed Empty	EX/EQ Displayed Entered
B3	Learner EX	Training Requirements are displayed, and the exemption box has been checked	Page Training Requirements Check Box	EX/EQ Displayed Empty	EX/EQ Displayed EX

Table D.15. Continued

No.	Canonical Sequence	Description	State Variables	Value before Current Stimulus	Value after Current Stimulus
B4	Learner EQ	Training Requirements are displayed, and the equivalency box has been checked	Page Training Requirements Check Box	EX/EQ Displayed Empty	EX/EQ Displayed EQ
B5	Learner Submit	Data with no changes has been submitted to the middle tier, and "Select Learner" page has been displayed.	Page Training Requirements	EX/EQ Displayed	Select Learner None
B6	Learner Cancel	No data has been submitted to the database, and the "Select Learner" page has been displayed	Page Training Requirements	EX/EQ Displayed	Select Learner None
C1	Learner Date EX	The learner's training requirements have been displayed, a date has been entered, and the EX box checked	Page Training Requirements Date Check Box	EX/EQ Displayed Entered Empty	EX/EQ Displayed Entered EX

Table D.15. Continued

No.	Canonical Sequence	Description	State Variables	Value before Current Stimulus	Value after Current Stimulus
C2	Learner Date EQ	The learner's training requirements have been displayed, a date has been entered, and the EQ box checked	Page Training Requirements Date Check Box	EX/EQ Displayed Entered Empty	EX/EQ Displayed Entered EQ
C3	Learner Date Submit	The learner's training requirements have been displayed, a date has been entered, and the Submit button pressed resulting in a warning box.	Page Training Requirements Date Warning Box	EX/EQ Displayed Entered None	EX/EQ Displayed Entered Displayed
D1	Learner Date EX Submit	A successful exemption has been entered, data has been submitted to the middle tier, and the "Select Learner" page has been displayed	Page Training Requirements	EX/EQ Displayed	Select Learner None
D2	Learner Date EQ Submit	A successful equivalency has been entered, data has been submitted to the middle tier, and the "Select Learner" page has been displayed	Page Training Requirements	EX/EQ Displayed	Select Learner None

Table D.16. Canonical Sequence Analysis for Editing EX/EQ

No.	Canonical Sequence	Description	State Variables	Value before Current Stimulus	Value after Current Stimulus
1	Empty	-	-	-	-
A1	Learner	Administrator has selected a Learner and the Learner's Training Requirements are displayed	Page Training Requirements	Select Learner None	EX/EQ Displayed
B1	Learner Date	Training Requirements are displayed, and a date has been entered for an EX or EQ on the screen	Page Training Requirements Date	EX/EQ Displayed Displayed	EX/EQ Displayed Entered
B2	Learner Justification	Training Requirements are displayed, and the justification text for EX or EQ has been entered on the screen	Page Training Requirements Justification	EX/EQ Displayed Displayed	EX/EQ Displayed Entered
B3	Learner Eold	Training Requirements are displayed, and the Eold box has been unchecked	Page Training Requirements Check Box	EX/EQ Displayed Checked	EX/EQ Displayed Empty
B4	Learner Enew	Training Requirements are displayed, and the box for Enew has been checked	Page Training Requirements Check Box	EX/EQ Displayed Eold	EX/EQ Displayed Enew

Table D.16. Continued

No.	Canonical Sequence	Description	State Variables	Value before Current Stimulus	Value after Current Stimulus
B5	Learner Submit	Data with no changes has been submitted to the middle tier, and "Select Learner" page has been displayed.	Page Training Requirements	EX/EQ Displayed	Select Learner None
B6	Learner Cancel	No data has been submitted to the database, and the "Select Learner" page has been displayed	Page Training Requirements	EX/EQ Displayed	Select Learner None
C1	Learner Date Submit	The learner's training requirements have been displayed, a date has been modified, and the new date and data submitted	Page Training Requirements Date	EX/EQ Displayed Entered	Select Learner None None
C2	Learner Justification Submit	The learner's training requirements have been displayed, a justification has been entered or modified, and the new date and justification text submitted	Page Training Requirements Justification	EX/EQ Displayed Entered	Select Learner None None

Table D.16. Continued

No.	Canonical Sequence	Description	State Variables	Value before Current Stimulus	Value after Current Stimulus
C3	Learner Eold Submit	The learner's training requirements have been displayed, Eold had been unchecked, and the data submitted removing the Eold.	Page Training Requirements Check Box	EX/EQ Displayed Unchecked	Select Learner None None
C4	Learner Enew Enew	The learner's training requirements have been displayed, Enew had been checked and unchecked, and the data submitted removing the Eold.	Page Training Requirements Check Box	EX/EQ Displayed Unchecked	Select Learner None None
C5	Learner Enew Submit	The learner's training requirements have been displayed, Enew had been checked, and the data submitted creating Enew.	Page Training Requirements Check Box	EX/EQ Displayed Checked	Select Learner None None

Table D.17. Canonical Sequence Analysis for Creating AC

No.	Canonical Sequence	Description	State Variables	Value before Current Stimulus	Value after current Stimulus
1	Empty	-	-	-	-
A1	Learner	Administrator has selected a Learner and the Learner's Training Requirements are displayed	Page Training Requirements	Select Learner None	AC Displayed
B1	Learner Complete	Training requirements are displayed and a checkmark has been entered for a completion	Page Training Requirements Check box	AC Displayed Empty	AC Displayed Complete
B2	Learner Date	Training requirements are displayed and a date has been entered for the completion on the screen.	Page Training Requirements Date	AC Displayed Empty	AC Displayed Entered

Table D.17. Continued

No.	Canonical Sequence	Description	State Variables	Value before Current Stimulus	Value after current Stimulus
B3	Learner Score	Training requirements are displayed and a score has been entered for the completion on the screen.	Page Training Requirements Score	AC Displayed Empty	AC Displayed Entered
B4	Learner Comments	Training requirements are displayed and comments have been entered for the completion on the screen.	Page Training Requirements Comments	AC Displayed Empty	AC Displayed Entered
B5	Learner Submit	Data with no changes has been submitted to database, and "Select Learner" has been displayed	Page Training Requirments	AC Displayed	Select Learner None

Table D.17. Continued

No.	Canonical Sequence	Description	State Variables	Value before Current Stimulus	Value after current Stimulus
B6	Learner Cancel	No data has been submitted to the database, and the Select Learner page has been displayed.	Page Training Requirments	AC Displayed	Select Learner None
C1	Learner Complete Date	Training requirements are displayed, a checkmark and a date for the completion have been entered	Page Training Requirements Check box Date	AC Displayed Complete Empty	AC Displayed Complete Entered
C2	Learner Complete Submit	Training Requirements have been displayed, a checkmark has been entered for the completion, and a successful completion has been submitted to the database.	Page Training Requirements Check Box	AC Displayed Complete	Select Learner None None

Table D.17. Continued

No.	Canonical Sequence	Description	State Variables	Value before Current Stimulus	Value after current Stimulus
C3	Learner Score Comments	Training Requirements have been displayed, and a score and comments have been entered for the completion	Page Training Requirements Score Comments	AC Displayed Entered Empty	AC Displayed Entered Entered

Table D.18. Canonical Sequence Analysis for Editing AC

No.	Canonical Sequence	Description	State Variables	Value before Current Stimulus	Value after Current Stimulus
1	Empty	-	-	-	-
A1	Learner	Administrator has selected a Learner and the Learner's Training Requirements are displayed	Page Training Requirements	Select Learner None	AC Displayed
B1	Learner Complete	Training requirements are displayed and a checkmark has been removed for the completion	Page Training Requirements Check box	AC Displayed Complete	AC Displayed Empty
B2	Learner Date	Training requirements are displayed and a new date has been entered for the completion on the screen.	Page Training Requirements Date	AC Displayed Displayed	AC Displayed Entered
B3	Learner Score	Training requirements are displayed and a new score has been entered for the completion on the screen.	Page Training Requirements Score	AC Displayed Displayed	AC Displayed Entered

Table D.18. Continued

No.	Canonical Sequence	Description	State Variables	Value before Current Stimulus	Value after current Stimulus
B4	Learner Comments	Training requirements are displayed and comments have been entered for the completion on the screen.	Page Training Requirements Comments	AC Displayed Displayed	AC Displayed Entered
B5	Learner Submit	Data with no changes has been submitted to database, and "Select Learner" has been displayed	Page Training Requirments	AC Displayed	Select Learner None
B6	Learner Cancel	No data has been submitted to the database, and the Select Learner page has been displayed.	Page Training Requirments	AC Displayed	Select Learner None
C1	Learner Complete Submit	Training Requirements have been displayed, a checkmark has been removed for the completion, and the data has been submitted to the database.	Page Training Requirements Check Box	AC Displayed Complete	Select Learner None None

Appendix E. Validation of Manual Line Counts

To validate that manual counting was producing accurate results, 25 of the 72 files edited in the Spanish project were counted using both automated and manual methods. Considering the 14 of the 25 files, with greater than 10 lines edited and no lines removed, the average percent difference between automated and manual count was 8.63%. The manual counts produced higher LOC in four cases, less LOC in 3 cases, and the same LOC in seven cases.

For files with lines removed, automated counts varied from manual counts by as much as 200%, with an average difference of 105%. In cases where lines were removed, manual counts more accurately reflected the total LOC modified during the project. For example, the file gotrain_academy_sidebar.asp had a net LOC of –38 when counted by comparing only number of lines in the files between pre- and post-Spanish files. When manually counted, it was found that 40 lines were added or edited and 74 lines were removed for 114 LOC modified, or 200% difference. The files compared, their automated and manual LOC, lines removed and percent differences are shown in Table E-1.

Table E.1. Comparison of Automated and Manual LOC Counts

		LOC						
		Automated Count	Hand Count			Percent Difference		Average Percent Difference
No.	File		Add/Edit	Removed	Total		Description	
1.	coursecatalog.asp		2		2	200.0%	No more than 10 lines	2.1%
2.	printpage.html	10	10		10	0.0%	No more than 10 lines	
3.	systemrequirements.htm	6	6		6	0.0%	No more than 10 lines	
4.	gotrain_hfs.inc	9	10		10	10.5%	No more than 10 lines	
5.	contact_text.htm	-4	0		0	-200.0%	No more than 10 lines	
6.	gotrain_academy_sidebar.asp	-38	40	74	114	200.0%	Removed Lines	105.0%
7.	c_coursehelp_kf.asp	-5	26	21	47	200.0%	Removed Lines	
8.	login.asp	69	99	16	115	50.0%	Removed Lines	
9.	askonline.asp	89	117	21	138	43.2%	Removed Lines	
10.	kfexam.asp	76	131	21	152	66.7%	Removed Lines	
11.	n_kfexam.asp	76	134	24	158	70.1%	Removed Lines	
12.	comments.asp	115	113		113	1.8%		8.6%
13.	conditions_of_self.asp	81	97		97	18.0%		
14.	course_menu.asp	100	131		131	26.8%		
15.	coursedescription.asp	119	119		119	0.0%		
16.	groupqueryresults.asp	51	51		51	0.0%		
17.	learnerqueryresults.asp	71	70		70	1.4%		
18.	test.asp	186	186		186	0.0%		
19.	testout.asp	16	16		16	0.0%		
20.	useradmin.asp	41	59		59	36.0%		
21.	contact_us.asp	11	11		11	0.0%		
22.	gotrain_academy_footer.asp	67	52		52	25.2%		
23.	gotrain_academy_header.asp	65	73		73	11.6%		
24.	contentserver.inc	16	16		16	0.0%		
25.	apputils.bas	37	37		37	0.0%		

Appendix F. Results Tables and Figures

Table F.1. Summary of Files Edited and LOC for the Spanish Project

File Type	No. of Files	LOC
ASP	31	2493
Html	22	737
VB class module	11	315
Totals:	**64**	**3454**

Table F.2. Summary of Files Edited and LOC for the Korova Project

File Type	No. of Files	LOC
ASP	44	7496
Html	32	N/A
VB Class Module	22	N/A
Totals:	**98**	**7496**

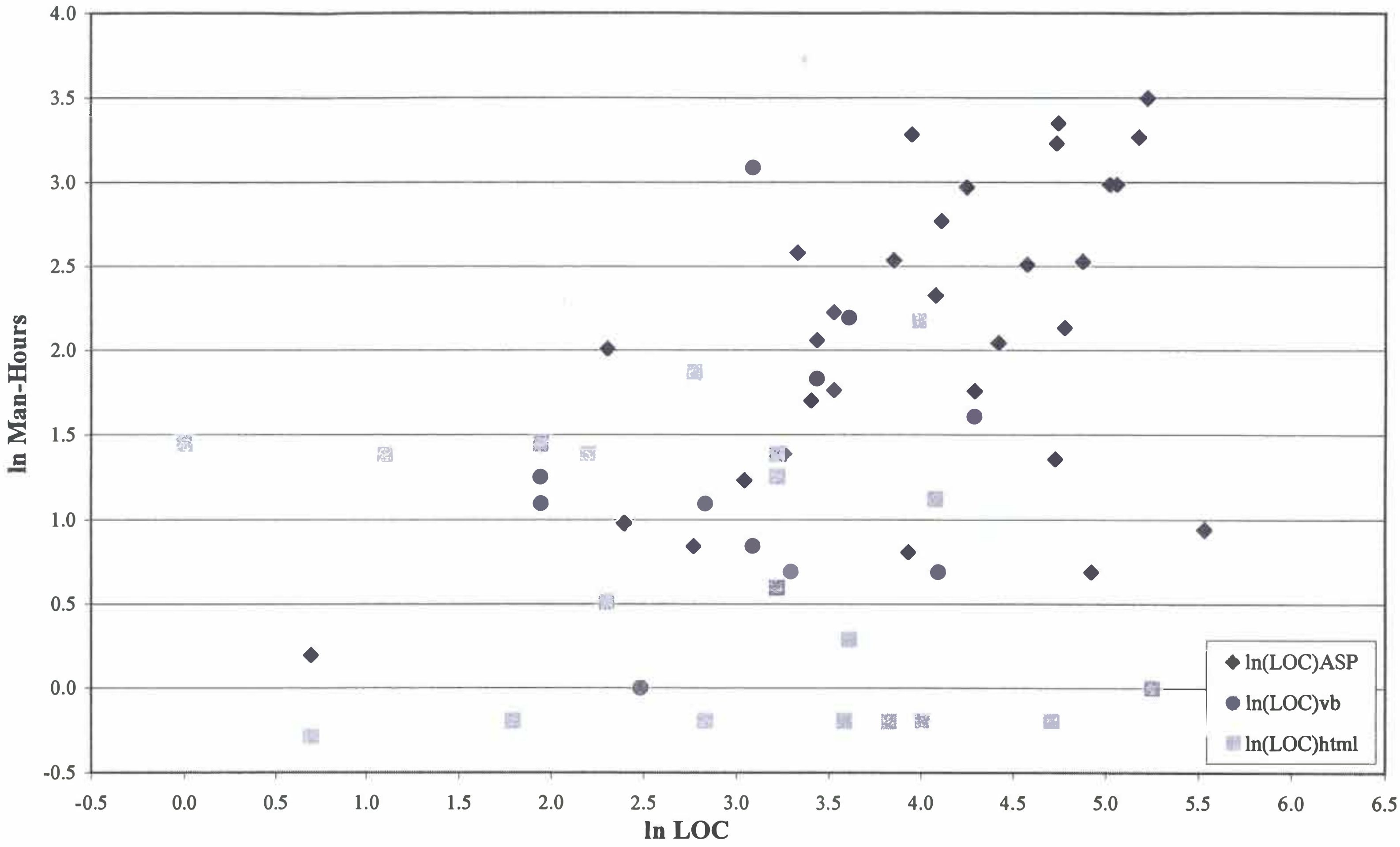


Figure F.1. ln Man-Hours vs. ln LOC for All Files in the Spanish Project

Table F.3. Files for Korova

FileID	Filename	File Type	LOC	ln LOC	Error Count	Hours	ln Hours	Group No.	Edits after Release	LOC After Release	Hours after Release	Project LOC
76	coursedescription.asp	asp	2	0.69		2.27	0.82	1				
62	gotrain_academy_footer.asp	asp	2	0.69		1.6	0.47	1				
63	gotrain_academy_header.asp	asp	2	0.69		1.6	0.47	1				
64	gotrain_academy_sidebar.asp	asp	2	0.69		1.6	0.47	1				
29	setup_interview_intro.asp	asp	2	0.69		0.8	-0.22	1				
24	email_1.asp	asp	3	1.10		1.78	0.58	1				
25	email_2.asp	asp	3	1.10		1.78	0.58	1				
23	access.asp	asp	4	1.39		1.78	0.58	1				
26	email_3.asp	asp	4	1.39		1.78	0.58	1				
71	checklogin.inc	asp	5	1.61		1.6	0.47	1				
86	groupqueryresults.asp	asp	7	1.95	1	3.01	1.10	2				
88	learnerqueryresults.asp	asp	10	2.30	7	5.61	1.72	2		1	4	
30	setup_interview_sidebar.asp	asp	14	2.64		2.58	0.95	2				
80	edit_curric_group.asp	asp	19	2.94	1	2.8	1.03	2				
59	furtherdefine_demographics.asp	asp	18	2.89		2.89	1.06	2		2	4	
31	summary.asp	asp	20	3.00		1.69	0.52	2				
75	coursecatalog.asp	asp	21	3.04		2	0.69	2				
57	define_demographics.asp	asp	19	2.94	1	5.89	1.77	2		2	4	
77	create_curric_group.asp	asp	30	3.40		2.8	1.03	2				
68	assign_train_req_ind.asp	asp	240	5.48	2	23.71	3.17	3	2	38	6.67	
69	assign_train_require.asp	asp	270	5.60	4	34.13	3.53	3	2	37	6.67	
73	course_menu.asp	asp	471	6.15	8	18.44	2.91	3	0			
78	create_train_group.asp	asp	81	4.39	2	10.07	2.31	3	0			
81	edit_train_req_ind.asp	asp	268	5.59	8	14.48	2.67	3	4	95	16.84	
82	edit_train_require.asp	asp	328	5.79	11	10.48	2.35	3	4	95	16.84	

Table F.3. Continued

FileID	Filename	File Type	LOC	ln LOC	Error Count	Hours	ln Hours	Group No.	Edits after Release	LOC After Release	Hours after Release	Project LOC
84	exemptions.asp	asp	484	6.18	3	54.9	4.01	3				
87	groupqueryresultsco.asp	asp	530	6.27		19.14	2.95	3	1	3	2.67	
89	learnerqueryresultsco.asp	asp	437	6.08	3	7.41	2.00	3	1	2	4	
27	learnerstatus.asp	asp	134	4.90		1.69	0.52	3	0			
91	reportbatch.asp	asp	121	4.80		6.16	1.82	3	1	31	0.5	
92	reportgroupquery.asp	asp	239	5.48		4.02	1.39	3	3	33	4.5	
93	reportgroupqueryco.asp	asp	222	5.40		3.42	1.23	3	2	22	0.5	
94	reportlearndemoquery.asp	asp	168	5.12	2	2	0.69	3	1	8	0.5	
95	reportlearndemoresults.asp	asp	241	5.48	3	6.74	1.91	3	0			
96	reportlearnerquery.asp	asp	190	5.25	4	4.02	1.39	3	2	29	4.5	
97	reportlearnerqueryco.asp	asp	173	5.15		4.09	1.41	3	1	18	0.5	
98	reports_query.asp	asp	299	5.70	1	22.77	3.13	3	1	33	0.5	
99	reporttrquery.asp	asp	153	5.03		11.69	2.46	3	1	8	0.5	
100	reporttrresults.asp	asp	234	5.46	3	11.41	2.43	3	1	7	4	
103	test.asp	asp	146	4.98		5.34	1.68	3	0			
104	train_completions.asp	asp	541	6.29	4	44.15	3.79	3	0			
105	train_require.asp	asp	168	5.12	2	23.75	3.17	3	0			
106	useradmin.asp	asp	79	4.37	2	9.82	2.28	3	0			
107	useradminedit.asp	asp	567	6.34	4	20.61	3.03	3	2	60	2.67	
Totals	**45**		**6971**		**76**	**420.3**			**25**	**524**	**84.36**	**7495**
Group 3 Only			**6784**		**66**	**374.44**			**29**	**519**	**72.36**	

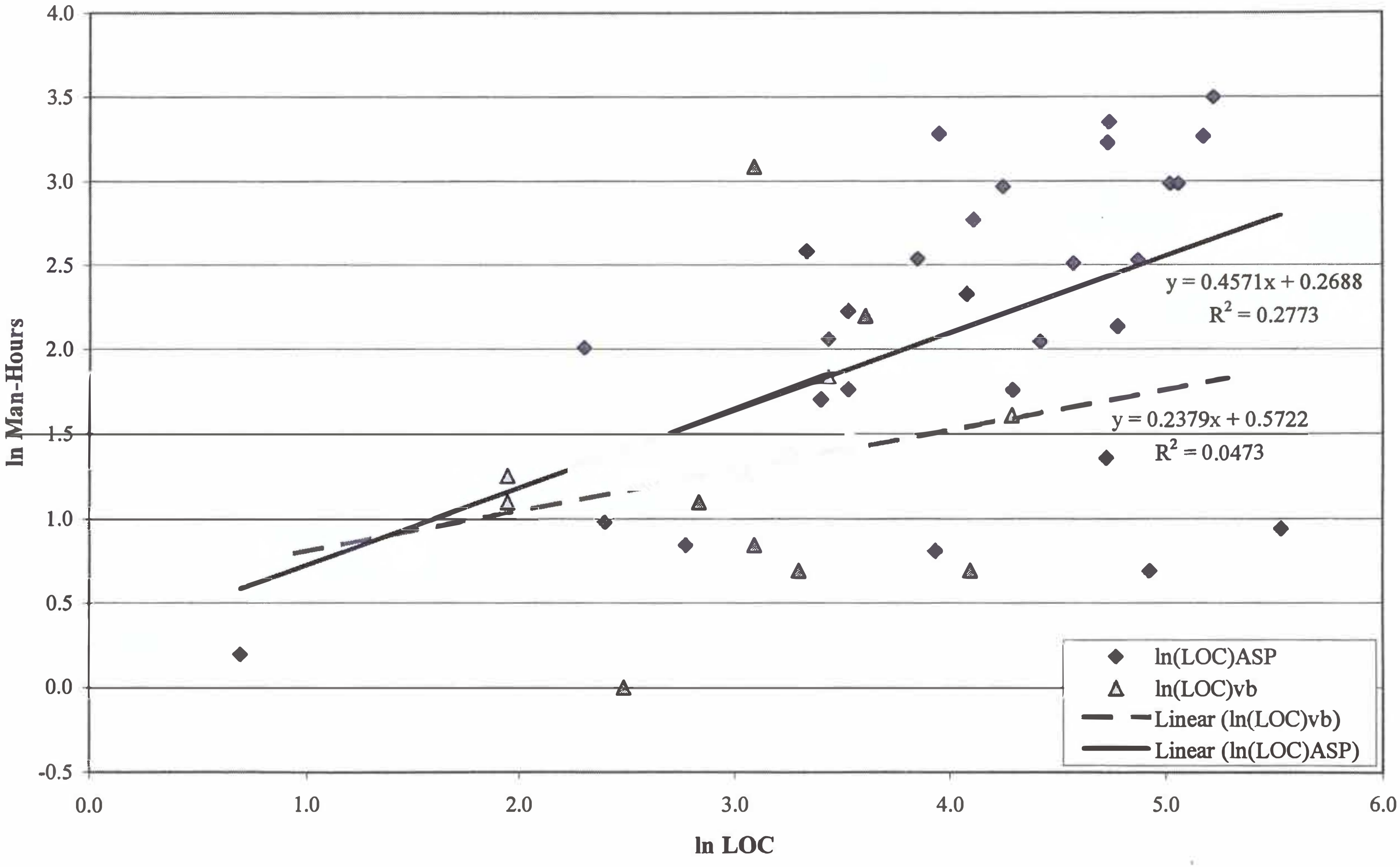


Figure F.2. ln Man-Hours vs. ln LOC for ASP and VB Files in the Spanish Project

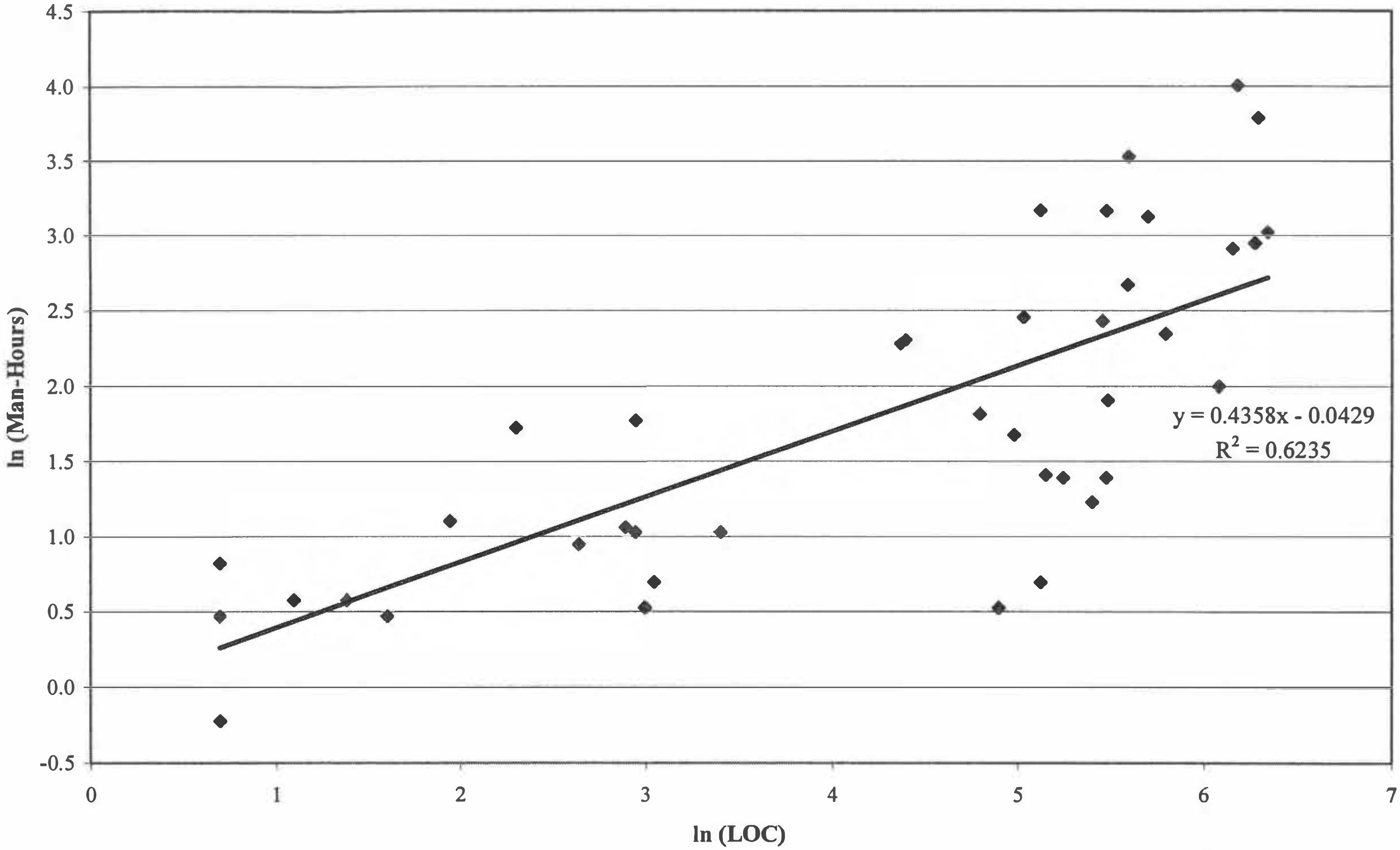


Figure F.3. ln Man-Hours vs. ln LOC for VB and ASP Files in the Korova Project

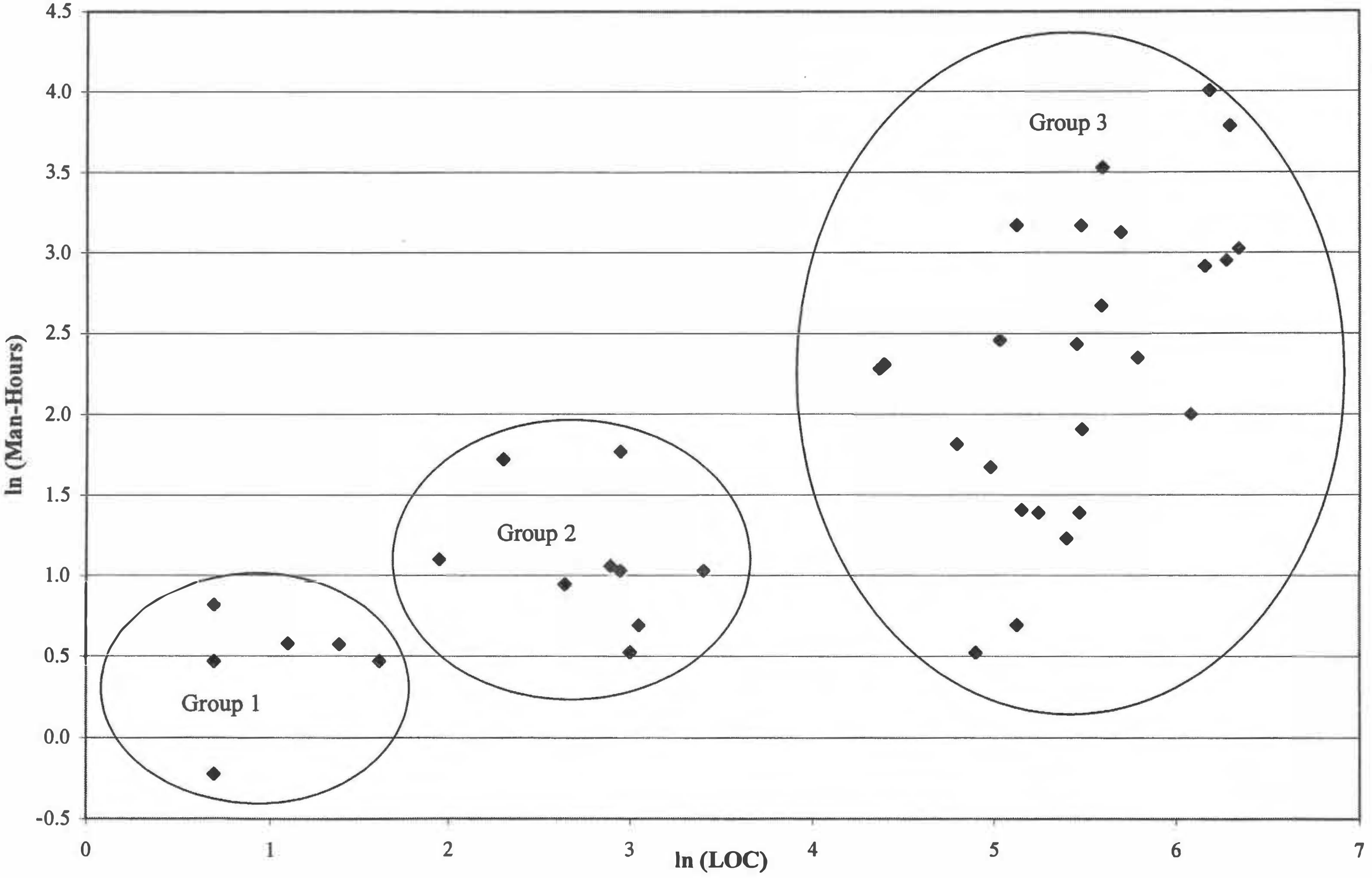


Figure F.4. Point Groupings in ASP Distributions for the Korova Project

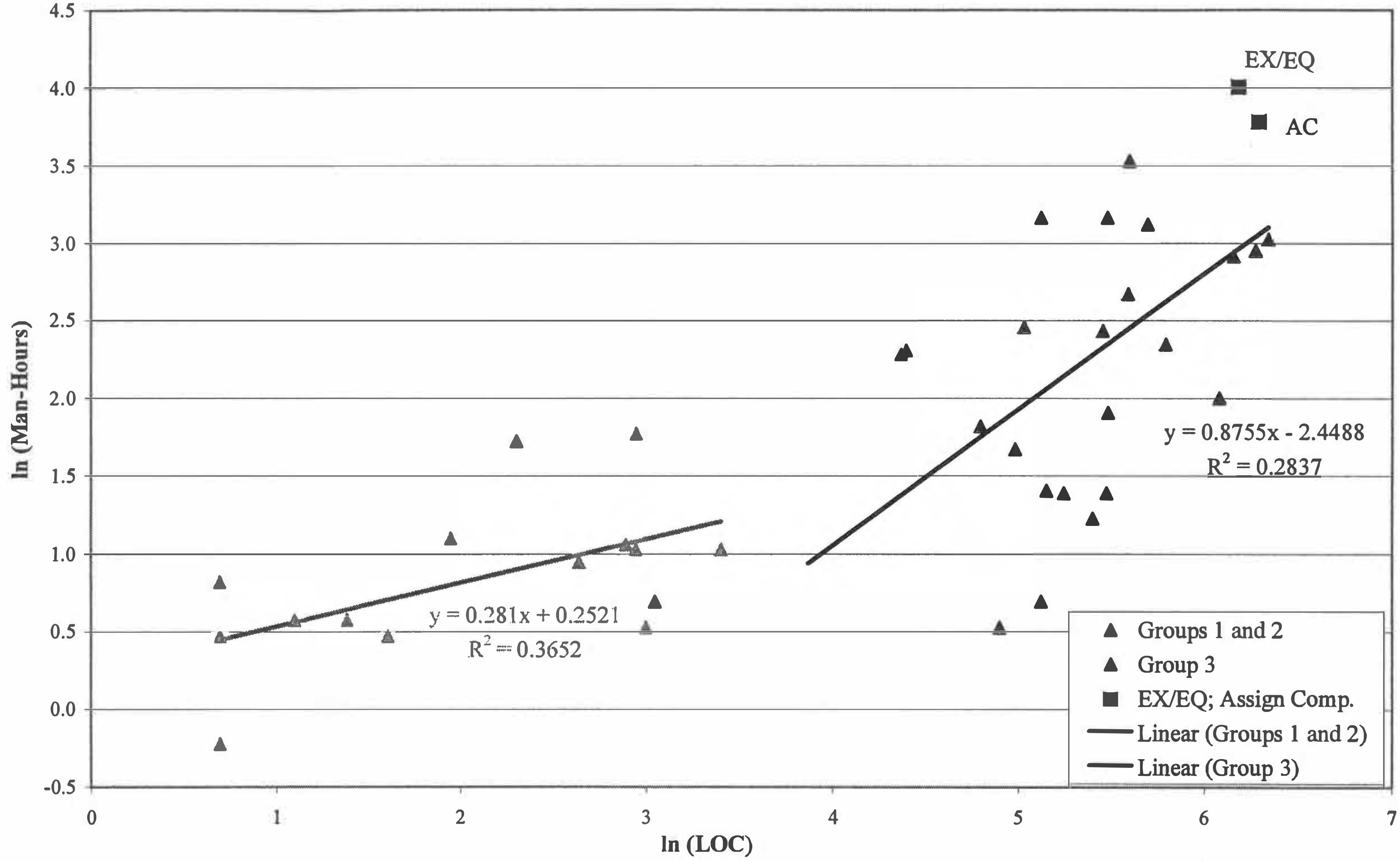


Figure F.5 Group 3 ASP Files in the Korova Project

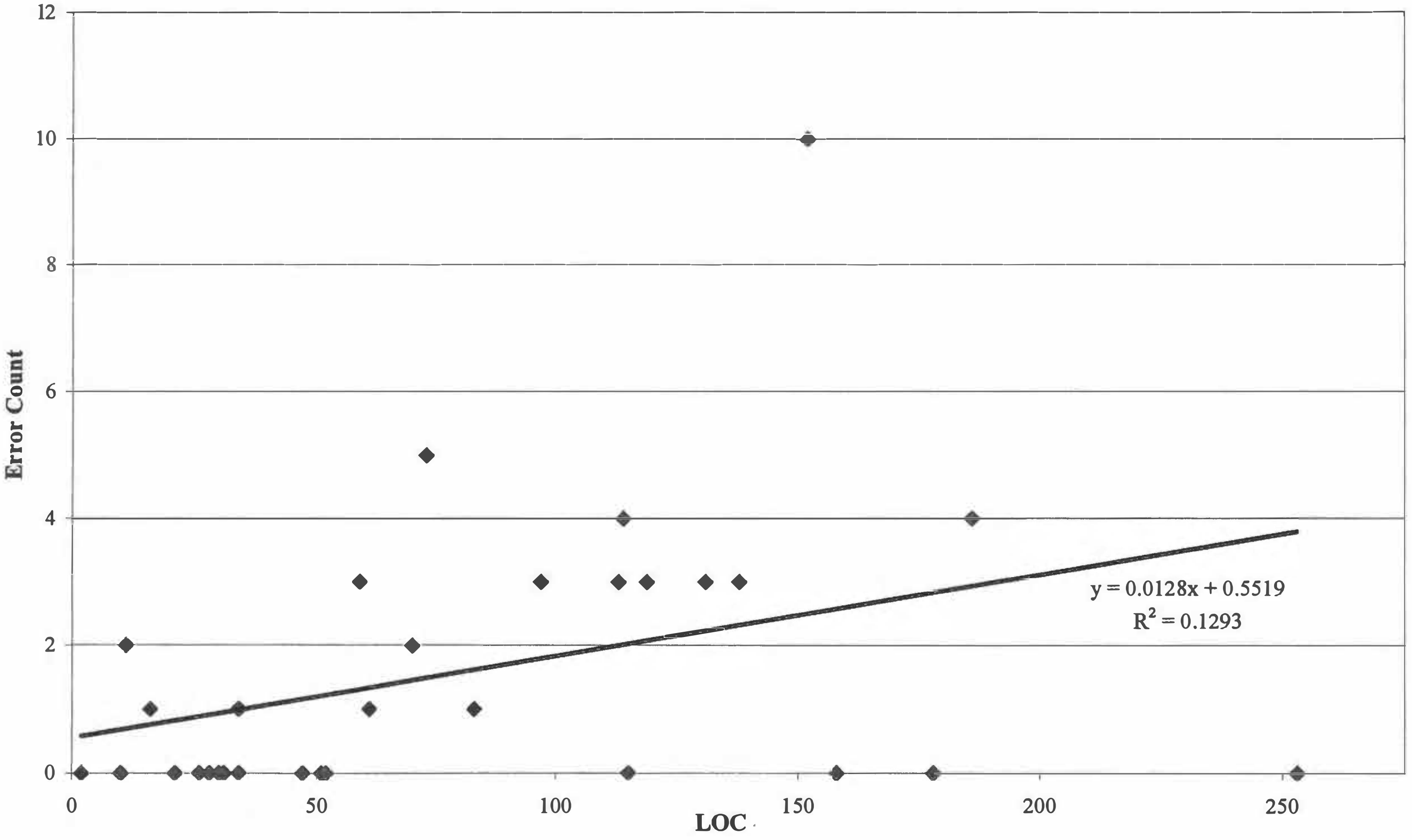


Figure F.6. Errors vs. LOC for ASP Files in the Spanish Project

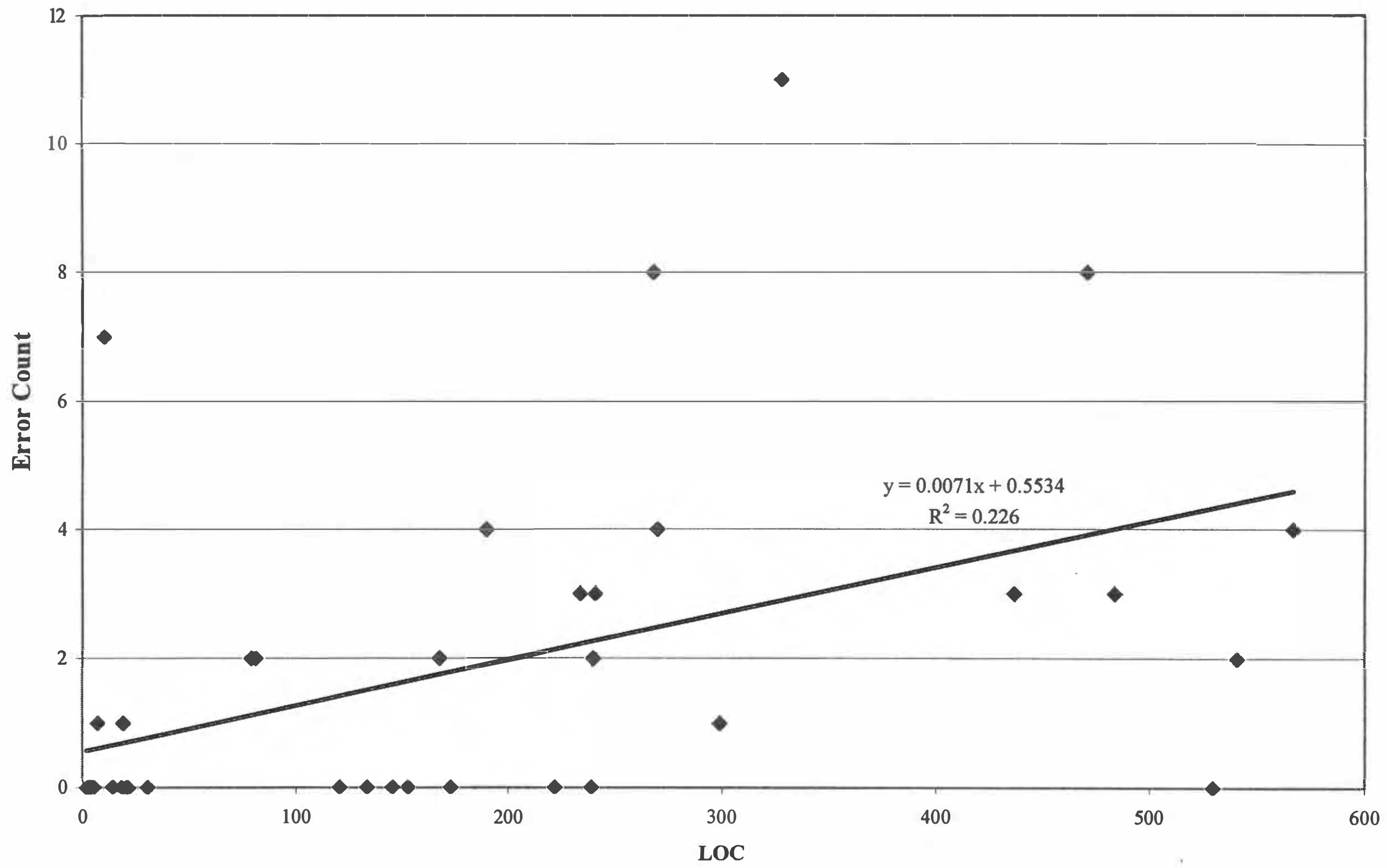


Figure F.7. Errors vs. LOC for ASP Files in the Korova Project

Table F.4. Description of Errors Related to EX/EQ and AC

Date	Feature	Error	Notes
1/8/2002	EX/EQ	Courses that are not allowed to be EX or EQ as defined in course catalog should not appear in Learner's TR list on EX/EQ	Not in Functional Specification or Detailed Design
1/8/2002	EX/EQ	Error if enter date incorrectly Error Type: clsExemptEquiv.updatedb (0x80004005) Microsoft][ODBC SQL Server Driver] Datetime field overflow /academy/exemptions.asp, line 70	Issue of data type mismatch between the presentation tier and middle tier. Not in the Function Specification or Detailed Design.
1/22/2002	EX/EQ	If TR deleted needs to be removed from EX/EQ learner TR list	
1/15/2002	AC	Error if enter an "A" in test score. Should be alphanumeric. Error Type: Microsoft VBScript runtime (0x800A000D) Type mismatch: 'clng' /academy/train_completions.asp, line 98	Issue of data type mismatch between the presentation tier and middle tier. Not in the Function Specification or Detailed Design.
1/22/2002	AC	If TR deleted needs to be removed from EX/EQ learner TR list	

VITA

Carla Sparks Tingle was born in Copperhill, TN on January 3, 1973. She grew up in Turtletown, TN and graduated Copper Basin High School in Copperhill, TN in 1991. Carla attended The University of Tennessee Knoxville and graduated with a Bachelor of Science degree in Geology in 1996.

Carla is currently working as Implementation Manager at GoTrain Corp. She has six years experience in software development and software engineering. Prior to her work in the software industry, she spent two years working as an environmental geologist. She currently resides in Oak Ridge, Tennessee.