Chapter 5
Mining Unstructured Software Repositories

Stephen W. Thomas, Ahmed E. Hassan and Dorothea Blostein

Summary. Mining software repositories, which is the process of analyzing the data related to software development practices, is an emerging field of research which aims to improve software evolutionary tasks. The data in many software repositories is unstructured (for example, the natural language text in bug reports), making it particularly difficult to mine and analyze. In this chapter, we survey tools and techniques for mining unstructured software repositories, with a focus on information retrieval models. In addition, we discuss several software engineering tasks that can be enhanced by leveraging unstructured data, including bug prediction, clone detection, bug triage, feature location, code search engines, traceability link recovery, evolution and trend analysis, bug localization, and more. Finally, we provide a hands-on tutorial for using an IR model on an unstructured repository to perform a software engineering task.
5.1 Introduction

Researchers in software engineering have attempted to improve software development by mining and analyzing software repositories, such as source code changes, email archives, bug databases, and execution logs [329, 371]. Research shows that interesting and practical results can be obtained from mining these repositories, allowing developers and managers to better understand their systems and ultimately increase the quality of their products in a cost effective manner [847]. Particular success has been experienced with structured repositories, such as source code, execution traces, and change logs.

Software repositories also contain unstructured data, such as the natural language text in bug reports, mailing list archives, requirements documents, and source code comments and identifier names. In fact, some researchers estimate that between 80% and 85% of the data in software repositories is unstructured [118, 351].

Unstructured data presents many challenges because the data is often unlabeled, vague, and noisy [371]. For example, the Eclipse bug database contains the following bug report titles:

- “NPE caused by no spashscreen handler service available” (#112600)
- “Provide unit tests for link creation constraints” (#118800)
- “jaxws unit tests fail in standalone build” (#300951)

This data is unlabeled and vague because it contains no explicit links to the source code entity to which it refers, or even to a topic or task from some pre-defined ontology. Phrases such as “link creation constraints,” with no additional information or pointers, are ambiguous at best. The data is noisy due to misspellings and typographical errors (“spashscreen”), unconventional acronyms (“NPE”), and multiple phrases used for the same concept (“unit tests”, “unit tests”). The sheer size of a typical unstructured repository (for example, Eclipse receives an average of 115 new bug reports per day), coupled with its lack of structure, makes manual analysis extremely challenging and in many cases impossible. Thus, there is a real need for automated or semi-automated support for analyzing unstructured data.

Over the last decade, researchers in software engineering have developed many tools and techniques for handling unstructured data, often borrowing from the natural language processing and information retrieval communities. In fact, this problem has led to the creation of many new academic workshops and conferences, including NaturaLiSE (International Workshop on Natural Language Analysis in Software Engineering), TEFSE (International Workshop on Traceability in Emerging Forms of Software Engineering), and MUD (Mining Unstructured Data). In addition, premier venues such as ICSE (International Conference on Software Engineering), FSE (Foundations of Software Engineering), ICSM (International Conference on Software Maintenance), and MSR (Working Conference on Mining Software Repositories), have shown increasing interest in techniques for mining unstructured software repositories.

In this chapter, we examine how to best use unstructured software repositories to improve software evolution. We first introduce and describe common unstructured