Toward Actionable, Broadly Accessible Contests in Software Engineering

Jane Cleland-Huang, Yonghee Shin,
Ed Keenan, Adam Czauderna, Greg Leach
DePaul University
Chicago, IL 60604
jhuang@cs.depaul.edu

Evan Moritz, Malcom Gethers,
Denys Poshyvanyk
College of William and Mary
Williamsburg, VA 23185
denys@cs.wm.edu

Jane Huffman Hayes, Wenbin Li
University of Kentucky
Lexington, KY 40506
hayes@cs.uky.edu

***Abstract*—Software Engineering challenges and contests are becoming increasingly popular for focusing researchers' efforts on particular problems. Such contests tend to follow either an exploratory model, in which the contest holders provide data and ask the contestants to discover "interesting things" they can do with it, or task-oriented contests in which contestants must perform a specific task on a provided dataset. Only occasionally do contests provide more rigorous evaluation mechanisms that precisely specify the task to be performed and the metrics that will be used to evaluate the results. In this paper, we propose actionable and crowd-sourced contests: actionable because the contest describes a precise task, datasets, and evaluation metrics, and also provides a downloadable operating environment for the contest; and crowd-sourced because providing these features creates accessibility to Information Technology hobbyists and students who are attracted by the challenge. Our proposed approach is illustrated using research challenges from the software traceability area as well as an experimental workbench named TraceLab.**

***Keywords*-Traceability, Contest, TraceLab, Actionable**

I. Introduction

The Information Technology (IT) world has a long history of using contests to focus researchers' attention on hard and thorny issues. Such contests can trigger a significant amount of research effort to solve or to improve a specific activity.

The *NetFlix* contest is perhaps the most famous IT contest because it carried a $US 1,000,000 award for the winner. Contestants were tasked with improving the accuracy for predicting how much a person would enjoy a new movie based on their past preferences [10]. Datasets were provided, rules were published, and evaluation metrics were specified. The competition ran for three years and was finally awarded in 2009 to a group of AT&T Research engineers. Over 50,000 teams participated in the contest. This created an excitement and buzz in the community, led to previously unforeseen collaborations, and culminated in a 10% improvement in movie prediction accuracy.

The Text REtrieval Conference (TREC) also stands out as an exemplar in this area. TREC was started in 1992 to help the information retrieval community build a shared infrastructure of large datasets in order to develop methodologies for performing standard comparisons of techniques, and ultimately to advance the state-of-the art in information retrieval [13]. TREC was very successful and is attributed with contributing to consistent improvements in information retrieval performance. In 2008, Google's Chief Economist, Hal Varian, stated that TREC had "revitalized research on information retrieval" [14].

In this paper, we first explore the way contests have been used to advance the state of the art in the Software Engineering community, and we then propose a new and innovative contest model and associated framework, applicable for highly focused contests which can be defined in terms of a specific, and measurable task. Furthermore, our framework goes beyond the current data-oriented or task-oriented approach, and creates actionable contests which enable participation from a far broader set of people including researchers from other fields, students, and even programmer hobbyists.

II. Software Engineering Contests

Building upon the success of other communities, several Software Engineering conferences and workshops have initiated contests in their own areas. We discuss three of them: the International Conference on Predictive Models in Software Engineering (PROMISE), Mining of Software Repositories (MSR), and the Traceability of Emerging Forms in Software Engineering (TEFSE) workshop challenges.

A. PROMISE Contests

The PROMISE conference was established in 2006 to explore "verifiable and repeatable models" that support the "implementation, evaluation, and management of software processes and projects" [2]. The organizers have done an excellent job of making a broad range of datasets publicly available. Each year the PROMISE call for papers challenges participants to use publicly available data to repeat, confirm, refute, or improve on previous results. While PROMISE provides a best paper award, it does not identify specific challenges to be addressed, but rather leaves the challenges open ended. PROMISE can therefore be categorized as a data-oriented experimental environment.

B. MSR Contests

The Mining Challenge has been hosted as a special track at the International Working Conference on Mining of Software Repositories (MSR) since 2006 [8]. Its primary goals are to foster relationships between industry and academia and to encourage MSR researchers to "show-off" their tools while applying them on common datasets. The MSR challenge is therefore an exploratory type of contest where participants discover "interesting things" on a shared data set. For example, this year's mining challenge evolves around the Android platform, an open source software stack for mobile devices, for which the organizers have provided the change and bug report data. The contributions (or paper reports) to the Mining Challenge are reviewed by the Program Committee and selected ones appear in the conference proceedings.

C. TEFSE Contests

TEFSE contests have been run in 2009 and 2011 at each of the past two workshops [4]. In each case, the workshop organizers provided several datasets, and then encouraged contestants to identify their own traceability area of interest (such as trace recovery or trace visualization), formulate traceability questions, and then utilize their existing traceability tools to address these questions. These contests are supported by the *Grand Challenges of Traceability* [6], which represent a community effort to document traceability-related research challenges. Contestants are required to clearly specify which of the identified challenges their contest entry addresses. Unfortunately, given the broad range of traceability challenges, comparing solutions and identifying a "winner" can be rather arbitrary.

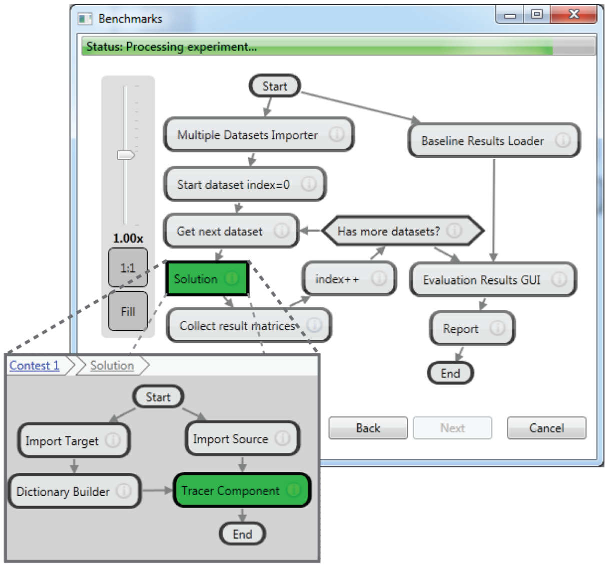
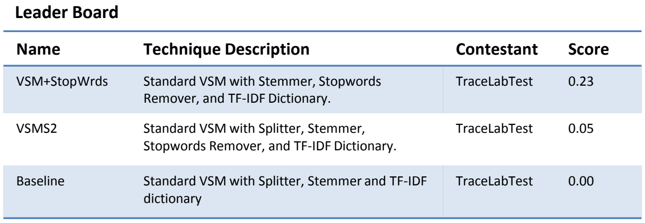


Figure 1. An Executable Experimental Framework in TraceLab



Leader Board

Name	Technique Description	Contestant	Score
VSM+StopWrds	Standard VSM with Stemmer, Stopwords Remover, and TF-IDF Dictionary.	TraceLabTest	0.23
VSMS2	Standard VSM with Splitter, Stemmer, Stopwords Remover, and TF-IDF Dictionary.	TraceLabTest	0.05
Baseline	Standard VSM with Splitter, Stemmer and TF-IDF dictionary	TraceLabTest	0.00

Figure 2. Leaderboard

III. Characteristics of an Actionable Contest

The current Software Engineering contest model, therefore, seems to be much looser than that of the information retrieval community. Most contests are focused around a set of datasets, and contestants are encouraged to use the datasets to showcase their research techniques. In contrast, the data mining community, while providing datasets for a contest or challenge, very clearly defines a specific task that is to be accomplished and associated metrics by which the success of the task will be evaluated.

While the PROMISE, MSR, and TEFSE challenges bring clear value to the Software Engineering community, in this paper we present a more structured form of contest defined in terms of (i) a clearly defined and arguably important task, (ii) realistic and publicly available datasets on which to perform the task, and (iii) clearly defined evaluation metrics for measuring the effectiveness of the task. We further present a novel experimental environment which packages up the datasets for a contest, provides a plug-and-play environment for contestants to try out their novel solutions, and evaluates results using the standard metrics associated with the contest.

Establishing contests in this way means that a specific challenge can live far beyond a local workshop event so that participation is not limited to workshop attendees. Furthermore, the standardized approach allows techniques to be comparatively evaluated over time. However, there are several issues that must be addressed in order to facilitate the contest and to ensure that all contestants utilize the same datasets, perform the same task, and evaluate their results in the same way. Furthermore, the start-up costs of entering the contests should be as low as possible so that researchers can tackle the real issues instead of spending months establishing the research environment. In the following section we introduce TraceLab as a viable option for addressing these challenges

IV. Executable Contest Environment

TraceLab is an experimental workbench constructed with Major Research Instrumentation grant from the National Science Foundation [3]. TraceLab allows researchers to compose and run experiments in a visual environment. In this paper we focus on one aspect of TraceLab: its ability to deliver a fully executable experimental environment for a specific contest. A more detailed explanation of TraceLab is provided in a separate paper [7].

Figure 1 presents the experimental canvas of TraceLab and depicts the four primary components of the executable experiment. The first component, labeled *Multiple Datasets Importer*, packages up the version controlled datasets used by the specific contest. It serves each dataset up in turn, until the experiment has been successfully executed on each dataset. The second major component, labeled *Solution*, represents a benchmarked solution. This solution may be entirely replaced, using TraceLab's plug-and-play technology, with a user-defined component developed by the researcher. Alternately, the experimenter may zoom into the solution space and examine the many sub-components that are contained in it. The experimenter may replace or modify any one of these components, or may alter or augment the workflow of the solution. The third element of the contest environment is the *Evaluation results GUI*, which evaluates the tracing results. The *Report* component provides the user with an option to automatically post results to a publicly visible leaderboard, as depicted in Figure 2.

Using TraceLab to conduct an experiment means that all contestants use exactly the same datasets, and compute metrics in exactly the same way. This is important because traceability researchers have traditionally used a wide variety of metrics and aggregation techniques, making it difficult to compare results [12]. In the proposed contest model, the results from each individual technique are compared using a metric defined by the contest owner, and the computed metric is then used to generate a ranked list of techniques. Additional comparisons are made using summary statistics such as median and mean metric values, various graphs including boxplots, and statistical tests such as the Wilcoxon signed rank test.

V. Traceability Contests

Two informal contests have already been launched to test and evaluate the contest infrastructure including its ability to support (i) the creation of a new contest, (ii) participation in a contest by a varied group of contestants, and finally (iii) comparative evaluation of results against an existing baseline, and then ranking multiple results and posting them onto the leaderboard. We describe both contests briefly below.

A. Contest 1: Trace Retrieval from Use Cases to Code

Goal To automatically retrieve traceability links from use cases to code without human intervention. **Data sets** The contest uses four datasets summarized in Table I. Each dataset includes a set of use cases, a set of classes (either Java code or a class description), and a trace matrix defined by the original developers of the system.

Table I
Datasets used in Use Case to Code Contest

Data set	Description	Reqs	Classes	Language	Traces
EasyClinic	Healthcare system	30	47	Class desc.	93
eTour	Tour guide system	58	116	Java	308
EAnci	Municipalities Mgmt.	140	55	Java	567
SMOS	Student Monitoring	67	100	Java	1044

Metrics The contest adopts a suite of metrics recommended by CoEST [1] to evaluate trace retrieval results. Each metric is first computed for each individual dataset and then aggregated across the four datasets. Here, we present only the results from *mean rank of average precision*. Average precision is computed as follows:

$$AveragePrecision = \frac{\sum_{r=1}^{N}(P(r) * relevant(r))}{|RelevantDocuments|} \quad (1)$$

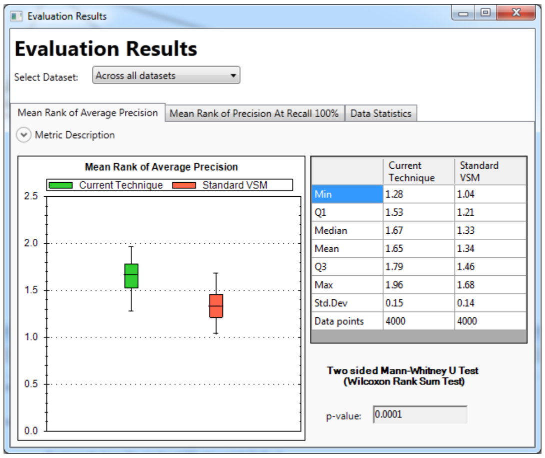
where r is the rank of the requirement in the ordered set of candidate trace links, N is the number of retrieved documents, $relevant()$ is a binary function assigned 1 if the rank is relevant and 0 otherwise, and $P(r)$ is the precision computed after truncating the list immediately below that ranked position. All targeted links are included in the computation, thereby computing Average precision at recall of 100%.

To compare two techniques across multiple data sets, the ranks of average precision values are computed between the techniques. The ranks are averaged over the four data sets with bootstrapping by randomly selecting a subset of queries, computing ranks, and repeating the whole process multiple times [9]. The details of the metric computation can be found on the CoEST web site.

Contest Results Figure 3 shows a screen shot of benchmarking results in TraceLab. In this benchmarking, the baseline is the Vector Space Model (VSM) [11] and the technique under evaluation is a Jensen-Shannon probabilistic model (JS) [5]. The result shows that the baseline VSM performed statistically significantly better than JS with a p-value of 0.0001.

B. Contest 2: Optimizing the value of Relevance Feedback

Goal To utilize relevance feedback to improve trace retrieval results. **Description** Various methods can be used to capture user feedback and utilize it to improve trace retrieval results. For example, the Rocchio technique captures relevance feedback and utilizes it to either increase or decrease term weightings in the underlying document representation [11]. Similarly, Direct Query Manipulation (DQM) allows a user to directly manipulate a trace query by filtering out terms and adding additional ones [11]. Various feedback methods and even different GUIs may be more or less effective. From a traceability perspective we are therefore interested in learning which user feedback mechanisms and user interfaces are most effective for improving traceability results. **Data sets** The contest uses the same data sets as used by Contest 1.



Evaluation Results

Select Dataset: Across all datasets

	Current Technique	Standard VSM
Min	1.28	1.04
Q1	1.53	1.21
Median	1.67	1.33
Mean	1.65	1.34
Q3	1.79	1.46
Max	1.96	1.68
Std.Dev	0.15	0.14
Data points	4000	4000

Two sided Mann-Whitney U Test (Wilcoxon Rank Sum Test)

p-value: 0.0001

Figure 3. Mean Rank of Average Precision: VSM (baseline) vs. JS

Metrics Relevance feedback can be measured in terms of both accuracy and human effort. Although both of these metrics must be taken into consideration, the contest focuses on accuracy results based on the previously described *Mean Ranking of Average Precision*.

Operating Environment Although not depicted in this paper, the contest provides a fully modifiable and executable experimental environment in which an experimenter can change or replace different kinds of components. For example, a human computer interaction (HCI) researcher could replace the GUI component used for collecting feedback, while another researcher might modify or replace the component for computing changes in term weightings, and still another researcher might remove the solution and replace it entirely.

VI. Conclusions and Next Steps

In this paper, we have advocated the use of software engineering contests to focus attention on specific research challenges. By making such challenges publicly known and accessible, we anticipate attracting more people to address traceability challenges. Our approach facilitates sharing of data, sharing of baseline results, and easy startup of experiments. It also reduces the initial investment to setup the experimental environment. It clearly is beneficial for a research community to have a way to evaluate results and, where possible, to measure improvements in a quantitative way. Furthermore, the history of contests in the Information Retrieval community has shown that contests foster healthy competition and collaboration, and have in fact resulted in significant advancements in many different areas.

Looking to the future, the Center of Excellence for Software Traceability (CoEST) is planning a series of public traceability challenges that will launch in the Summer of 2012. Each contest will address a previously identified traceability challenge [6], several of which are already sponsored by industrial organizations with vested interest in advancing the state of the art in traceability. Our proposed contest model therefore has the potential to encourage greater collaboration between industry and academia.

Our future work we will investigate the generalizability of our contest model to other areas of research, by modeling and executing typical experiments from those domains.

VII. Acknowledgments

The work described in this paper was funded by National Science Foundation grant # CNS 0959924.

References

[1] CoEST: Center of excellence for software traceability, http://www.CoEST.org.

[2] Predictive models in software engineering (promise), http://promisedata.org.

[3] Grand Challenges, Benchmarks, and TraceLab: Developing Infrastructure for the Software Traceability Research Community. *International Workshop on Traceability in Emerging Forms of Software Engineering (TEFSE)*, 6, 2011.

[4] TEFSE (Traceability in Emerging Forms of Software Engineering) 2011 Traceability Challenge, May 2011.

[5] M. Gethers, R. Oliveto, D. Poshyvanyk, and A. DeLucia. On integrating orthogonal information retrieval methods to improve traceability link recovery. In *Int'nl Conf. on Software Maintenance (ICSM'11)*, pages 133–142, 2011.

[6] O. Gotel, J. Cleland-Huang, J. Huffman Hayes, and A. Zisman. Grand Challenges of Traceability, Software and Systems Traceabilty, eds. Jane Cleland-Huang, Olly Gotel, and Andrea Zisman, springer verlag. 2012.

[7] E. Keenan, A. Czauderna, G. Leach, J. Cleland-Huang, Y. Shin, E. Moritz, M. Gethers, D. Poshyvanyk, J. Maletic, J. Huffman Hayes, A. Dekhtyar, D. Manukian, S. Hossein, and D. Hearn. Tracelab: An experimental workbench for equipping researchers to innovate, synthesize, and comparatively evaluate traceability solutions. In *Tool Demo, 34th International Conf. on Software Engineering (ICSE)*, 2012.

[8] Mining Software Repositories. http://2012.msrconf.org /challenge.php.

[9] C. Mooney and R. Duval. Bootstrapping: A nonparametric approach to statistical inference. 1993.

[10] NetFlix Challenge. http://www.netflixprize.com/.

[11] Y. Shin and J. Cleland-Huang. A comparative evaluation of two user feedback techniques for requirements trace retrieval. In *27th Symposium on Applied Computing (SAC)*, 2012.

[12] Y. Shin, J. Huffman Hayes, and J. Cleland-Huang. A framework for evaluating traceability benchmark metrics. In *Technical report, DePaul University, School of Computing*, pages TR:12–001, 2012.

[13] G. Tassey. *Economic Impact Assessment of NIST's Text REtrieval Conference (TREC) Program*. National Institute of Standards and Technology, Gaithersburg, Maryland, 2010.

[14] H. Varian. Why data matters, http://googleblog.blogspot.com/2008/03/why-data-matters.html.