Using Context Distance Measurement to Analyze Results across Studies

Daniela Cruzes[1,2], Manoel Mendonça[1]
[1]NUPERC/UNIFACS, Salvador, BA- Brazil
[2]FEEC/UNICAMP Campinas, SP- Brazil
{daniela, mgmn}@unifacs.br

Victor Basili
Dept. of Computer Science, University of Maryland, College Park, MD, 20742, USA
basili@cs.umd.edu

Forrest Shull
Fraunhofer Center - Maryland, 4321 Hartwick Road, College Park, MD, 20740, USA
fshull@fc-md.umd.edu

Mario Jino
FEEC/UNICAMP Caixa Postal 6101 13083-970 Campinas (SP), Brazil
jino@dca.fee.unicamp.br

Abstract

Providing robust decision support for software engineering (SE) requires the collection of data across multiple contexts so that one can begin to elicit the context variables that can influence the results of applying a technology. However, the task of comparing contexts is complex due to the large number of variables involved. This works extends a previous one in which we proposed a practical and rigorous process for identifying evidence and context information from SE papers. The current work proposes a specific template to collect context information from SE papers and an interactive approach to compare context information about these studies. It uses visualization and clustering algorithms to help the exploration of similarities and differences among empirical studies. This paper presents this approach and a feasibility study in which the approach is applied to cluster a set of papers that were independently grouped by experts.

1. Introduction

Empirical studies have long been used to provide confidence in assertions about what is true and not true in the software engineering domain. By providing rigorous observation of the effects of a development technique under specific conditions, empirical studies allow for analyses of the conditions under which practices yield similar effects on a project's cost, quality, or schedule.

The ability to build up rigorous abstractions of information about practices not only provides confidence in individual assertions about specific techniques, but also is an important capability in providing an engineering basis for software development. This capability is an essential part of approaches like the Experience Factory [4] or the more recently suggested Evidence-Based Software Engineering [14].

Providing robust decision support for software development – i.e. making a statement about what development practices can help achieve goals related to cost, quality, or schedule for a given environment – requires the collection of data across multiple contexts so that one can begin to elicit these variables. However, the task of comparing contexts is a complex one. The set of potential context variables is quite large, including issues such as team size; team experience; lifecycle model; product size and complexity; automated support; organizational culture; application domain; among several others.

Due to the large number of possible variations from one development environment to another, we have argued [1] that this process of knowledge building about practices must therefore be based on families of related studies, designed so that a range of context variations can be explored. Although this approach is logically appropriate, it does pose some practical problems. First, it is not always clear a priori what the important context variables are, meaning that important sources of variation may go unmeasured. Second, because there are so many potential context variables, we often cannot design experiments or even identify environments which offer coverage of all the variables.

In other words, to design an effective family of studies, multiple experimenters, without having a clear concept of all the contributing factors, must agree a priori on a set of variables to collect and identify environments that cover a fairly complete set of variables, so that all studies are comparable.

An alternative approach is to abstract information across several previously run studies. One method for this is to perform a literature search, reviewing the

relevant literature in a rigorous way and constructing a textual summary of the evidence related to a given issue. If the sources do not agree then it is the reviewer's responsibility to construct a fair summary of the evidence on both sides of the issue [14]. A key issue in supporting these systematic reviews is therefore to have a robust approach to identify and compare the studies' contexts.

In this paper, we extend the work from an earlier paper [18], in which we proposed a practical and rigorous process for identifying possible hypotheses and context information from papers. In that work, we observed that to make this approach work, and build a suitably large and varied dataset, we had to be able to analyze information about many relevant variables and the effect of practices from several studies that were not a priori designed to fit together. In effect, this required the ability to simulate a family structure over independent studies that were not explicitly designed to build directly on one another. For this we defined a specific template to collect context information from the papers. Since then, we have improved this approach. We have better formalized the context information data collection process [6] and we have developed an interactive approach to compare context information across studies. This paper focuses on this last issue. It presents an approach that uses visualization and clustering algorithms to help the exploration of similarities and differences among context descriptions of empirical studies.

2. The Analysis Process

Our methodology has the goal of building a set of conclusions about contexts of experimental papers when analyzed together to get insights about software development practices contained in multiple studies, which need not have been designed specifically to produce related data.

As input, the methodology requires a focus of study, i.e. a (set of) software engineering phenomenon(a) about which information is needed. The process consists of three main steps (Figure 1), starting with a selection of papers of interest, the extraction of information from these papers and finally the analysis and interpretation of the contexts.

This process is iterative, in that the results of a given step may convince the researcher to go back to a previous step and redo the associated activities. For example, if the researcher is not satisfied with the information extracted from a set of papers, he or she may use these results to suggest new areas to search in order to select more papers for analysis.

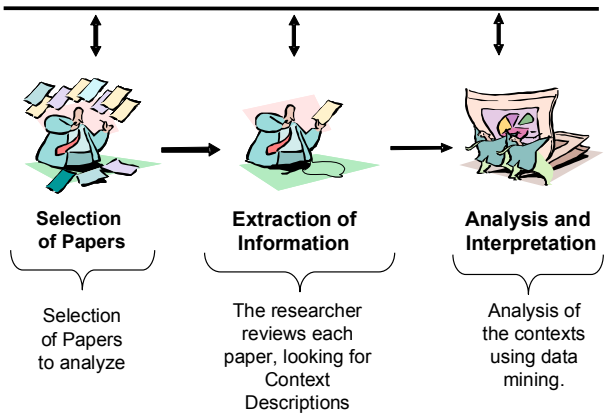

Figure 1 - High level analysis process

The output of the process is a set of conclusions and new knowledge that arises from the process. As a secondary output the process creates a structured and searchable repository of evidence and context information. The advantage of the creation of the Structured Base is that it can be reused. Other researchers can evolve and reuse it according to new research goals as they arise.

The first step of the methodology, selecting relevant papers, is performed much the same as it would be in any method, no matter how formal, and is thus not discussed here at length. Defining the problem of course depends largely on the interests of the researcher. The problem definition is also related to the amount of knowledge already accumulated in an area. For example, as more evidence is accumulated we can move from studying how failure-prone software products are to which types of failures are most common; to which types of products display common failure profiles; to which context variables make those failure types more likely to occur. This allows us to evolve our knowledge into more useful models over time. Selecting papers that can be searched for evidence in the focus area is also conducted largely in the same way regardless of the individual process being followed. To be suitable, a paper must provide some empirical information or experience-based hypotheses relating to the focus of study.

The remaining steps will be performed in a quite specific way in this methodology. Part of Step 2, specifically the extraction of evidence, was described in detail elsewhere [18]. In this paper we focus on the extraction of the context information. Then we present a context comparison approach for the reported studies that can be used as part of Step 3, Analysis and Interpretation.

3. Extracting Context Information from Papers

In the Information Extraction step, the researcher must review each paper, looking for experimental evidence on the subject of interest and potential context descriptions described in the text. While reviewing the papers selected in the previous step, the researcher should highlight the important information (so that there can be some traceability to the original source if questions arise later). After highlighting the information, it is important that key details are transferred to data entry forms to create the structured base for analysis. Up to now, we are using forms implemented in Excel; although in future work we intend to create a tool to support the activities of the process.

As said before, this paper will focus specifically on context descriptions. Context descriptions are the details concerning the environment from which the measures were drawn. The context descriptions are important for comprehensibility of measures and influencing factors for the focus of study.

Our approach proposes a template (Figure 2) to collect context information. At least one template is filled for each paper, possibly more, if the paper describes data that was collected from several studies. As different studies report different metrics of interest to them, not every paper will have all of the desired context information. However, the template should be filled out as completely as possible. Although there are not mandatory attributes, missing values will be accounted for during the analysis, as they limit the strength of the conclusions that can be drawn. Besides that the analyst can review the template to insert new fields, for example: threats to validity.

This list of attributes on the template was adapted partially from Sjøberg et al. [19]. The attributes of the context description template are:

1) **Paper Title:** The title of the paper from which the reader is extracting the information.
2) **Type of the Study:** This field classifies the study as an Experiment, a Case Study or a Survey [22].
3) **Topic**: This field uses the IEEE keywords in the Computer.org website to denote the topic of study.
4) **Goals:** These fields state the goals for the study described in the paper, using the GQM goal template [3].
5) **Variables:** These fields record all variables related to the study. A variable is a concept or construct that can vary or have more than one value. The researcher might then be interested in knowing how certain variables are related to each other. There are two basic kinds of variables: dependent and independent [22]. The following characteristics must be gathered for each dependent and independent variable in the study: Name, Type (independent, dependent or unclear), Possible Values, Data Collection Procedure (explains the method used to measure the variable, including for example what instrumentation and tool support were used).
6) **Subjects:** This field describes the subjects of the study by category (undergraduate students, graduate students, professionals, scientists, other or unknown) and number.
7) **Instrumentation:** These fields describe data gathering or data generation tools used in the experiment.
8) **Task:** These fields categorize the tasks done by the subjects, as reported in the paper; duration of the task(s); and work mode (team or individual).
9) **Work Products:** These fields describes the working products used in the tasks, including: Name, Type (Requirements; Architecture/design; Code; Change Reports; Error Reports; Other), Origin (Constructed, Commercial, Student Project, Open Source; Other, Unclear), Application Domain (E.g. Text Processing, Flight Simulation, etc), Size (using the metric specified by the author.), Representation Paradigm (E.g. Object Oriented, Imperative, Structured, etc.) and Language (E.g. plain English, Fortran, Pascal, C++, Java).
10) **Replication:** This field indicates whether this study is a replication of another one.
11) **Other:** This fields records any other information that is important for understanding the model, metric, techniques, or the empirical study itself (e.g., missing definitions, environmental characteristics, or information about process conformance).

Besides the context, we collect the following information for each paper:

1) The paper reports on a study done on pilot or production projects? The study involved one or several projects? This information helps to evaluate the applicability of the results;
2) How well the results were measured? This information helps on the assessment of the rigor on which the study was run.
3) How the experience was reported (journal papers, conferences, TRs)? It measures the acceptability of the results in a specific community;
4) The person who reported the evidence was directly involved with the study? It assesses the familiarity of the author with the results.

Paper Title		
Topic		
Type of Study		
Goals	*Analyze*	
	for the purpose of	
	with respect to	
	from the point of view of	
Variables	*Name:*	
	Type:	
	Scale Type:	
	Possible Values:	
	Data Collection Procedure:	
	Name:	
	Type:	
	Scale Type:	
	Possible Values:	
	Data Collection Procedure:	
Subjects	*Category*	
	Number	
Instrumentation	*Tool Name:*	
	Functions:	
	Tool Name:	
	Functions:	
Task	*Category*	
	Duration:	
	Work Mode:	
Work Products	*Name:*	
	Type	
	Origin	
	Application Domain:	
	Size	
	Representation Paradigm	
	Language:	
Replication	*Is replication?*	
	Reference:	
Other		

Figure 2 – Information Gathering Template

4. Analysis and Interpretation

The analysis and interpretation phase aims to analyze the study contexts in order to explore similarities and differences among them. For this, we propose the steps shown on Figure 3.

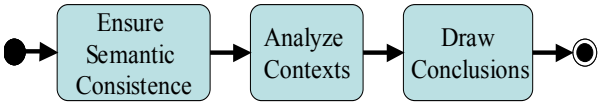

Figure 3 – Context Analysis Process

4.1. Ensuring Semantic Consistency

The first step aims to harmonize the context descriptions to ensure the consistence of the terms and concepts used among the forms. For that one needs to certify that conceptual mismatches problems are solved. There are three types of mismatches we foresee [20].

- Scope Mismatches: Occurs when there is a difference in the way a domain is interpreted (conceptualized), which results in different concepts or different relations between those concepts. In this type of mismatch, two results seem to represent the same concept, but do not have exactly the same meaning (although there may be some overlap). For example: Two studies may refer to the "cost" of a practice, although one may include only the cost of the effort to apply the practice, while the other may include the start-up costs as well (e.g. sending personnel for training).
- Model coverage and granularity: This type of mismatch describes problems that can arise in trying to combine results when it is unclear to what part of the domain those results are applicable. For example, a study may make claims about a large class of software development projects while only having evidence concerning one or two specific instances of such projects.
- Explication Mismatches: An explication mismatch is a difference in the way the conceptualization is specified. This can manifest itself in mismatches in definitions, mismatches in terms and combinations of both. There are three types:
  - Synonymous terms: Synonyms, in this context, are different terms that refer to the same concept. A trivial example is the use of the term "strength" in one study and the term "cohesion" in another, to refer to the same concept (that is, the amount of interaction within components of a system).
  - Homonym terms: This type of mismatch occurs when the meaning of a term is different in different contexts. For example, the term "interface defects" can have different interpretations, depending on the context: It can refer to a defect in the Human-Computer Interface or a defect in the interfaces between two software components.
  - Encoding: Values in the studies may be encoded in different formats. For example, a numbers of lines of code may be represented as "KLOC" or as "LOC" or "SLOC," etc.

4.2. Analyzing Contexts

Once the researcher solved the mismatches problems, the analysis can be conducted. The researcher should analyze the distance, proximity, affinity and confidence among the contexts on which the results from the papers were generated. There are many approaches that can be used to perform this activity; we propose an approach in which we use a hierarchical clustering algorithm to organize the study contexts.

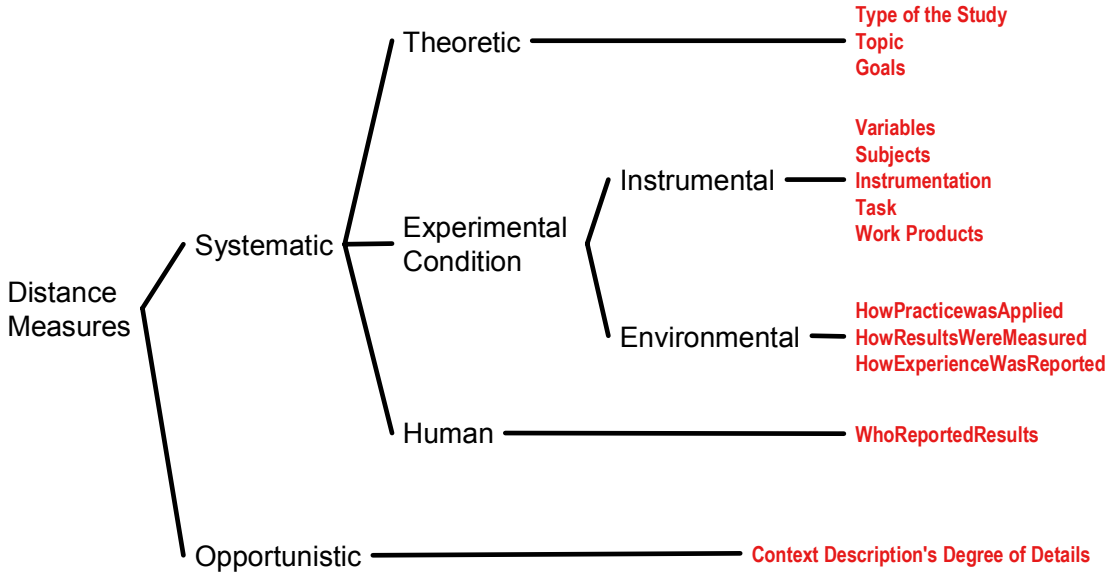

Figure 4 – Conceptual Distance Tree for Context Attributes

Before applying any automated approach one has to organize and weight the attributes of concern for the analysis of experimental contexts. As shown on Figure 4, we organize the attribute as follows:

1) Systematic: characteristics of the study that may have influence on the results and may be controlled by experimenters. They can be related to:
   a. Theory – concerns the theory used to design the experiment. We suggest the following attributes: type of study, topic and goals.
   b. Experimental Condition – differences that can come from instrumentation of environment conditions
   c. Human – differences on the results can come from the different skills of the experimenters while running the experiment, or of the person who is reporting the results.
2) Opportunistic: characteristics of the study that may have influence on the results but were not controlled by experimenters or not reported on the papers.

Based on these categories the researcher can weight the attributes collected using the information gathering template [10]. That can be done by: 1) directly raising the attribute importance for the clustering algorithm; 2) submitting to the algorithm only the attributes of interest; 3) creating new columns for each possible value of multi-valued attributes of interest.

These weighted attribute are used to define a composed distance measure to compare the studies experimental contexts. For that, our approach uses an interactive clustering approach in which hierarchical clustering is combined with visualization, to identify how different study contexts compare to each other. Section 5 will explain in detail how this is done.

4.3. Drawing Conclusions

The last step of the analysis process consists on drawing conclusions based on the patterns observed, create new hypotheses, or refute or confirm initial hypothesis or folklore. This is done by expert analysis of the evidence extracted from the papers and the analysis of the context in which this evidence was gathered [18]. In this manner, the clustering of studies by experimental context discussed in the previous section, and detailed in the next one, gives the analyst a systematic way to reason why study results agree with or contradict one another. This creates a robust basis for decision making support about a studied software engineering method, tool or technique.

5. Clustering Studies

Clustering is a division of data into groups of similar objects. A clustering algorithm attempts to find natural groups of components (or data) based on some similarity. To determine cluster membership, most algorithms evaluate the distance between a point and the cluster centroids. Cluster centroids represent the mathematical center of items in a cluster. The output from a clustering algorithm is basically a statistical

description of the cluster centroids with the number of components in each cluster.

Each cluster consists of objects that are similar among themselves and dissimilar to objects of other groups. The overall goal of the clustering is to allow many data objects to be represented by relatively few clusters, which means that the level of granularity is an important factor in the usefulness of the model. Representing data by fewer clusters necessarily loses certain fine details, but achieves simplification.

One of the requirements of good clustering is the ability to determine the number of natural clusters in the data set. In fact many clustering algorithms ask users to specify the number of clusters that they want to generate [10]. This is not an option in our scenario, in which the researchers want to interactively explore what is the best grouping for a set of studies, so as to produce clusters of studies run in meaningfully similar contexts. Unnecessary merges or splits need to be avoided, as they produce unnatural clusters. The solution to this problem is to use the hierarchical agglomerative clustering (HAC) algorithms [11] that allow users to control parameters to determine the proper number of clusters. HAC algorithms generate a hierarchical structure of clusters instead of sets of clusters. It has the disadvantage of requiring the calculation of the distance between all pairs of objects of the analyzed data set, O (N^2) of storage space. However, this is not a problem in our domain in which we have at most a few tens of studies to compare.

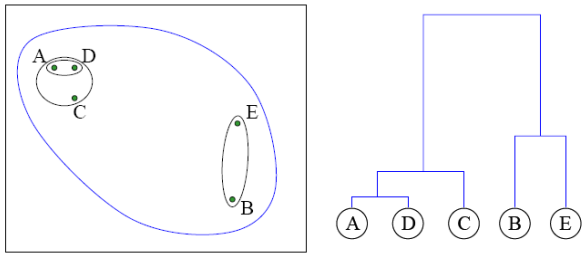

Figure 5 - Hierarchical clustering and a dendrogram [11]

Hierarchical clustering results are usually represented as dendrogram. A dendrogram is a binary tree, in which each data item corresponds to a terminal node of the binary tree and the distance from the root to a sub-tree indicates the similarity of the sub-tree – highly similar nodes or sub-trees have joining points that are farther from the root. Figure 5 shows the clustering of five data points (A, B, C, D, and E) on a 2D plane. The dendrogram (a binary tree) on the right side shows the clustering result by using Single-linkage and Euclidean distance [10]. The height of each sub-tree represents the distance between the two children. For example, the distance between A and D is the smallest among all possible pairs, they are merged together as a sub-tree and the height of the sub-tree is very short because they are similar in terms of the distance measure. On the other hand, B and E are not so close and the height of the corresponding sub-tree is taller because of this.

The existence of a good interaction and clustering visualization interface is an important requirement of our approach. In order to better learn and understand how the studies compare to each other, the analyst should be able interpret how the studies are being grouped by their context information. In our work, we have used a tool that produces interactive visualization and exploration of hierarchical clustering, the Hierarchical Clustering Explorer (HCE) [11].

The HCE uses dynamic queries and coordination among multiple views to produce visualization of the hierarchical clusters. Users begin by performing a hierarchical clustering and build a dendrogram with a color mosaic display underneath (see Figure 6).

The color mosaic displays a graphical representation of the data set color-coding each value in the table according to a color mapping scheme. The records are transposed into the color map; they are shown as vertical lines color-coded in accordance to each of its attribute values. When researchers want to identify hot spots and understand the distribution of data, they can examine the color mosaic.

The dendrogram is displayed with a color mosaic at its leaves so that the analyst can better interpret the data. For this reason, the arrangement of rows and columns of the color mosaic display changes according to the clustering result. By default, in HCE, a high attribute value has a bright red color and a low value has bright green color. Middle values have a black color.

With a widget control, users can interactively adjust a minimum similarity parameter to find the most natural number of clusters. They can also see how the hierarchical clusters are presented in other familiar and easy-to-understand views such as 1-dimensional histograms and 2-dimensional scatter plots. The coordination between the overview color mosaic and those views is bi-directional, that is, users can select a group of items in a view and see where they fall in other views.

The HAC algorithm implemented on HCE [11] is summarized as follows. Let's assume that we want to cluster n data items, and we have n*(n-1)/2 similarity (or distance) values between every possible pair of n data items:

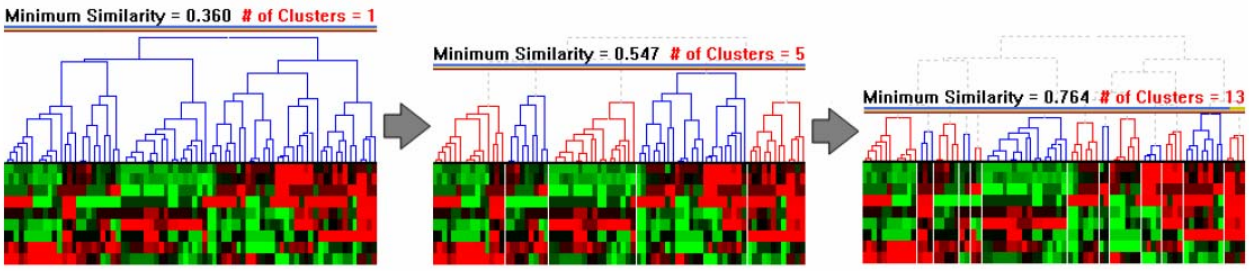

Figure 6 – Usage of the minimum similarity bar (MSB) [11]

1) Initially, each data item occupies a cluster by itself. So there are n clusters at the beginning.
2) Find one pair of clusters whose similarity value is the highest, and make the pair a new cluster.
3) Update the similarity values between the new cluster and the remaining clusters.
4) Steps 2 and 3 are applied n-1 times before there remains only one cluster of size n.

There are many possible choices in updating the similarity values in step 3. Among them, most common ones are complete-linkage, average-linkage, and single-linkage. Complete-linkage sets the similarity values between the new cluster and the remaining clusters to be the minimum of similarities between each member of the new cluster and the rest. Average-linkage uses average similarity value as a new similarity values. Single-linkage takes the maximum.

One of the key components in HCE is the minimum similarity bar. By dragging down the bar whose y-coordinate determines the minimum similarity threshold, users can filter out the less similar elements. Using this Minimum Similarity Bar, or MSB, users can easily find the clusters of elements that are tight enough to satisfy a given threshold. The algorithm used to calculate the MSB is explained in detail in reference [11].

Figure 6 shows the process of cluster discovery using the minimum similarity bar, from now on called MSB. The y coordinate of the bar determines the minimum similarity value. Users can drag down the bar to filter out items that are distant from a cluster. The minimum similarity values changed from 0.36 to 0.764 in this example to separate 1 large cluster into 13 small clusters.

To prevent users from losing global context during dynamic filtering, the entire dendrogram structure is shown on the background, and users can highlight the position of a cluster in the original data set by just clicking on the cluster.

Using the approach discussed in Section 4 and on the clustering procedures presented in this Section, the analyst has the tools to analyze the context from many papers and draw conclusions about the results found.

6. Applying Clustering on Testing Papers

We ran a feasibility study, in order to evaluate how the information is gathered and analyzed using our approach. Our strategy was to cluster some papers following our approach and to compare it to expert manual clustering done independently, the goal being to check if our approach led to the same grouping done independently by experts.

We choose the work by Juristo, Moreno and Vegas that performs a review of 25 years of Testing Technique Experiments [12] as our basis of comparison. This paper, which we refer to as the TTE paper, analyzes the maturity level of the knowledge about testing techniques by examining existing empirical studies about these techniques. The work analyzed 24 studies, and produced a testing technique knowledge classification.

In our study, we selected 11 of the 24 papers from the TTE paper. The criterion for choosing them was to choose the papers related to functional, control-flow, data-flow and mutation testing techniques, following the classification scheme of Juristo et al. This criterion was used for the reason that our previous knowledge on the domain of these techniques could help on the analysis of the papers. Table 1 lists the chosen papers and their original grouping in the TTE paper.

Table 1 Selected papers and their original grouping

Group	Studies
Group 1	Weyuker [21]
	Bieman & Schultz [5]
Group 2	Frankl & Weiss [7]
	Hutchins [9]
	Frankl & Iakounenko [8]
Group 3	**Myers [15]**
	Basili & Selby [2]
	Kamsties & Lott [13]
	Wood et al. [23]
Group 4	Offut & Lee [16]
	Offut et al [17]

After selecting the papers we executed the following steps:

1) Using the proposed template, we extracted the context information from each one of the selected papers.
2) We interactively applied the clustering approach on the gathered data.
3) We compared our results to the original grouping.

As defined in our approach, we used the following attributes to derive a data file describing the contexts of the studies reported in the 11 papers analyzed:

- Type of the Study;
- Description of the Topic;
- Object of Study;
- Subjects Category (students, professionals, etc);
- Subjects Work Mode (individual, team, etc);
- Task Category (create, analyze, plan, etc)
- Work Products (code, requirements, design, etc);
- Instrument Origin;
- The name of each dependent and independent variable after semantic consistency check;
- The testing technique studied on the paper (Functional, Code Reading, Structural, etc).

In the list above, we must notice two particularities. The first is that each study involves several dependent and independent variables. The second is that the number and type of variables vary from study to study. In order to weight this in our clustering approach, we created a record for each variable of each study on our data file. This way, a study that involved five variables, for example, yielded five records in the data file.

Another important issue is the weighting of the testing technique attribute. This attribute defines the study treatments, so one has to consider it as the most important attribute to group the studies. The assumption was confirmed by interviews that we conducted with the TTE paper authors last year. To factor this into the data file, we created a column for each technique involved in the studies.

In both cases discussed above, we are effectively strengthening the weighting given to the context attributes categorized as "instrumental" on the conceptual distance tree shown in Figure 4. It is important to remark that one could also have asked the algorithm to weight other attributes more strongly on the clustering process based on the Theoretical, Experimental or Human context attributes.

The input file was opened in the HCE tool and we ran the algorithm of Hierarchical Clustering using the Pearson Correlation Coefficient as the distance measure. The minimum similarity distance starts with 50%. This, in our case, yielded two main clusters. One included Group 1 and the other included Groups 2, 3 and 4 of the TTE paper (see Table 1). In order to get more groups, we moved the MSB, raising the internal similarity measure within the clusters. We obtained four groups for a MSB between 56% and 60%. Figure 7 shows how the HCE tool uses the dendrogram colors to highlight them.

Looking at Figure 7, one can see that there is only one paper that was classified in a different way than the TTE paper: the paper by Myers was categorized by us as in Group 4 while the TTE paper placed it in Group 3, as shown in Table 1. We discussed this anomaly with the authors of the TTE paper, who said that this paper was later excluded from their analysis because of a lack of some details of the context information, especially on the details of the studied techniques. This is a good result as the other 10 papers were classified correctly.

Some interesting results were obtained at different MSB values. For an MSB greater than 60.5%, for example, we obtained 5 clusters. Group 1, with papers by Bieman & Schultz and Weyuker, was split into two (see Table 1). Looking at the data, we realized that both papers have a technique in common, but Weyuker's study included three other techniques that were not covered by Bieman & Schultz.

Next, we investigated other distance measures and obtained similar results. We believe that the grouping of studies will vary little with the distance metrics used (Manhattan, Euclidean, and Pearson).

Another important point to mention is that the color mosaic is very useful to visualize the similarities and differences among contexts. For example, Figure 7 shows that this set of studies is very uniform in many of the context attributes.

The feasibility study illustrated the usefulness of using our approach to quickly understand how a sizeable amount of studies compare to each other. We want to emphasize the importance of using a good interactive visualization tool for this task. As an example, consider a sample of the data file (reduced both in number of lines and columns) shown in Table 2. Looking at it, one can realize how difficult it is to cluster the studies manually and see a relation among the studies even when only a few attributes are used.

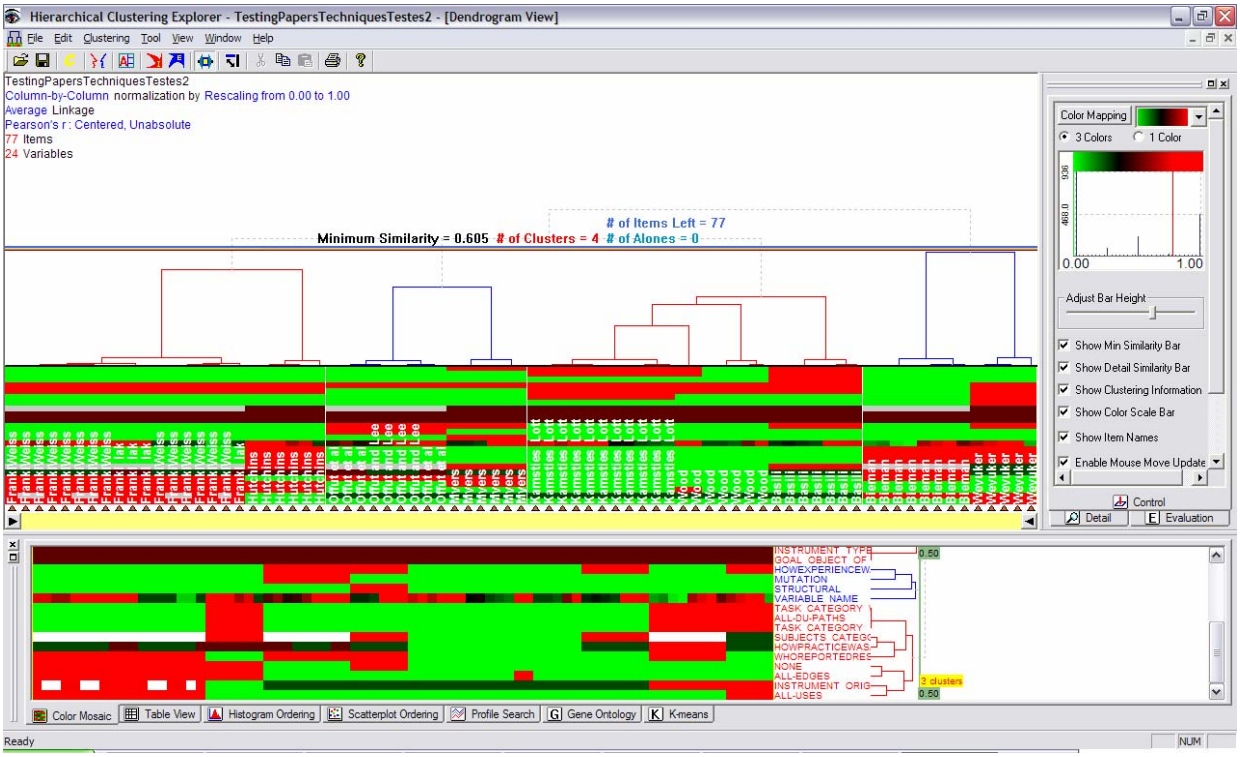

Figure 7 - HCE Tool - Clustering Context Based on Various Attributes

Table 2 Techniques studied on each paper

Study	Functional	Structural	Code Reading	None	All-c-uses	All-p-uses	All-uses	All-du-paths	All-Edges	Sentence Coverage	Mutation
Basili	X		X							X	
Weyuker					X	X	X	X			
Bieman								X			
FrankIWeiss				X			X		X		
Hutchins				X				X	X		
Frankl_lak				X			X		X		
Myers	X	X		X							
Kamsties_Lott	X		X		X						
Wood	X		X						X		
Offut and Lee											X
Offut et al											X

7. Conclusions

In order to provide useful and accurate decision support about software development practices and their effects on projects, one usually needs to analyze results from several empirical studies that cover different development environments. Due to the wide range of influencing factors, and the fact that one cannot yet confidently specify them all ahead of time, it is desirable to have an approach from which observations about experimental results and influencing factors can be built bottom up. Building such a dataset would be infeasible if we cannot make use of existing data, even data that was never designed to contribute to a larger empirical base.

We have been working on a practical process to gather and combine empirical evidence from papers. Having looked at a collection of datasets and abstracted up conclusions on several specific topics, we have shown elsewhere that our methodology can produce useful and feasible results, especially when it is compared to the results output from the more manual, expert-based approach [18]. However, a problem we frequently find is how to compare studies that report conflicting results among themselves.

This paper proposes an approach to group studies according to their context information, so that conflicting and corroborating evidence can be better understood according to the context in which they were obtained. The paper presents evidence that the use of a systematic approach to gather context information combined with clustering techniques can group studies

in the same way as an expert would. This opens up several interesting possibilities such as using interactive clustering to evaluate the generality of evidences across studies, and to use cluster centroids to identify the typical context for a set of conflicting evidences.

It is important to point out that the approach presented here can be used together with any other methods of combining results from studies. It can be helpful to analyze data collected for systematic reviews for instance.

As future work, we intend to use our approach in bigger contexts, analyzing studies on the context of large systematic reviews. Other than this, we will investigate the cost-effectiveness of the use of the algorithm in early stages of systematic reviews.

8. References

[1] Basili V.R., Shull, F. and Lanubile, F. Building knowledge through families of experiments, IEEE Trans. on Software Engineering, 25 (4), 456–473, 1999.

[2] Basili, V. and Selby, R. Comparing the Effectiveness of Software Testing Strategies, IEEE Trans. on Software Engineering, 13 (12), pp 1278-1296, December, 1987.

[3] Basili, V.R., Caldiera G., and Rombach H.D. Goal Question Metric Approach, Encyclopedia of Software Engineering, pp. 528-532, John Wiley & Sons, Inc., 1994.

[4] Basili, V.R., Caldiera G., and Rombach H.D. The experience factory, Encyclopedia of Software Engineering, 2, pp. 469–476, 1994.

[5] Bieman, J. M., and Schultz, J. L. An empirical evaluation (and specification) of the all-du-paths testing criterion. IEE/BCS Software Engineering Journal, 7(1):43-51, Jan. 1992.

[6] Cruzes D., Mendonca, M., Basili, V., Shull, F. and Jino, M.; Extracting Information from Experimental Software Engineering Papers, Technical Report 2007-2, Nuperc, Salvador-University – Unifacs, 2007.

[7] Frankl, P. G., and Weiss, S. N. An experimental comparison of the effectiveness of branch testing and data flow testing. *IEEE Transactions on Software Engineering*, vol.19, no.8, pp. 774-787, Aug., 1993.

[8] Frankl, P., and Iakounenko, O. Further empirical studies of test effectiveness. SIGSOFT-FSE:1998. Lake Buena Vista, Florida, USA, 153–162, 1998.

[9] Hutchins, M., Foster, H., Goradia, T., and Ostrand, T. Experiments on the effectiveness of data-flow and control-flow-based test adequacy criteria. Proceedings of the 16th ICSE. Sorrento, Italy, 191–200, 1994.

[10] Jain, A. K., Murty, M. N., and Flynn, P. J. Data clustering: a review. ACM Computing Surveys, 31(3):264–323, 1999.

[11] Jinwook S. Information Visualization Design for Multidimensional Data: Integrating the Rank-By-Feature Framework with Hierarchical Clustering. Ph.D. Dissertation, Dept. of Computer Science, Univ. of Maryland, Dec. 2005.

[12] Juristo N., Moreno A. M., Vegas S. Reviewing 25 years of Testing Technique Experiments. Empirical Software Engineering Journal, v.9, pp 7-44, 2004.

[13] Kamsties, E., and Lott, C. M. An empirical evaluation of three defect-detection techniques. Proceedings of the Fifth ESEC. Sitges, Spain, 1995.

[14] Kitchenham, B., Dyba, T., Jørgensen, M.; Evidence-based software engineering, Proceedings of the 26th ICSE 2004, Edinburgh, Scotland, pp. 273–281, 2004.

[15] Myers, G. J. A controlled experiment in program testing and code walkthroughs/inspections. Communications of the ACM. 21(9): 760–768, 1978.

[16] Offut, A. J., and Lee, S. D. An empirical evaluation of weak mutation. IEEE TSE 20(5): 337–344, 1994.

[17] Offut, A. J., Lee, A., Rothermel, G., Untch, R. H., and Zapf, C. 1996. An experimental determination of sufficient mutant operators. ACM Transactions on Software Engineering and Methodology 5(2): 99–118.

[18] Shull, F., Cruzes, D., Basili, V. and Mendonca, M.; "Simulating Families of Studies to Build Confidence in Defect Hypotheses," Journal of Information and Software Technology, vol. 47(15): 1019-1032, December, 2005.

[19] Sjøberg, D.I.K., Hannay, J.E., Hansen, O., Kampenes, V.B., Karahasanović, A., Liborg N.K. and Rekdal, A.C. A survey of controlled experiments in software engineering. IEEE TSE, 31 (9), pp 733-753, Sept. 2005.

[20] Visser, R.S. Pepijn, Jones, M. Dean, T.J.M. Bench-Capon, M.J. R. Shave, An analysis of ontological mismatches: heterogeneity versus interoperability, AAAI 1997 Spring Symposium on Ontological Engineering, Stanford, USA 1997.

[21] Weyuker, E. J. 1990. The cost of data flow testing: An empirical study. *IEEE Trans. on Software Engineering* 16 (2), pp 121-128, 1990.

[22] Wohlin, C.; Runeson, P.; Höst, M.; Ohlsson, M. C.; Regnell, B.; Wesslén, A. Experimentation in Software Engineering: An Introduction. The Kluwer, International Series in Software Engineering, 2000.

[23] Wood, M., Roper, M., Brooks, A., and Miller, J. Comparing and combining software defect detection techniques: A replicated empirical study. Proceedings of the 6th ESEC. Zurich, Switzerland, 1997.

9. Acknowledgements

The work presented has been partly funded by CNPq (the Brazilian National Research Council).

Special thanks go to Professors Natalia Juristo and Sira Vegas for many valuable insights on the work.