ABSTRACT
This paper presents our experiences with experimentation in the last two years, using frameworks both in the development and in the reengineering of software systems, as well as the perspectives of applying experimentation in the near future. We also discuss insights and lessons learned from our experience.

Categories and Subject Descriptors
D.2.7 [Software Engineering]: Distribution, Maintenance, and Enhancement - restructuring, reverse engineering, and reengineering.

General Terms
Experimentation.

Keywords
Development, reengineering, software process, framework.

1. INTRODUCTION
There is a growing interest both by industry and academia to use empirical studies to evaluate processes and products developed. However, as observed in [2], few works can be found in the literature concerning experimentation in reverse engineering and reengineering areas. This has motivated us to plan and conduct case studies to analyze and evaluate reengineering and development processes supported by object-oriented frameworks. Such experiences are presented in this paper, as well as some new perspectives regarding experimentation to be conducted in the near future.

2. OUR EXPERIENCE ON THE LAST TWO YEARS
We have developed an agile process for framework-based reengineering, called PARFAIT [4], which aims at migrating legacy systems to the object-oriented paradigm supported by frameworks, in particular those frameworks whose construction was based on analysis pattern languages [7].

Initially, two reengineering case studies have been conducted, on a non-systematic way, to observe the adequacy of some PARFAIT activities previously defined. Each case study has used a different small-scale legacy system, both developed using Clipper. During these case studies, several inconsistencies among the process activities were found and solved: a) the activities description and steps have been better defined to ease the process usage and some activities names have been changed to more meaningful ones; b) some activities have been reallocated to other phases, other than the ones initially assigned; and c) the goals that were initially defined for each process phase have been refined.

Next, a third case study has been planned and documented according to the structure proposed by Wholin et al. [9], with the purpose of evaluating the usage of the PARFAIT process in a reengineering project, as well as to observe its effectiveness in the reengineering of procedural systems to the OO paradigm. This case study has been performed by an undergraduate student that was taking the fourth grade of the Computer Science Program. The case study has been conducted in two steps: the first consisted in training and studying the techniques needed for the beginning of the case study and the second consisted of using PARFAIT during the reengineering of a library control system developed in Clipper, with about 6 KLOC. Several data have been collected during the case study conduction. From that, it was observed that a) about 86% of all the reengineering time has been spent mainly to prepare the artifacts related to VV&T; b) a tool for framework instantiation used at the reengineering (i.e. the GREN framework [1]) does not support iterative and incremental reengineering, i.e., it is not possible to instantiate the framework again for a certain system without loosing all the adaptations manually done in the source code of the previous versions.

The results of this third case study motivated the creation of a test reuse approach, named ARTe [3], which associates text requirements to the pattern language patterns used in the framework construction and this is used by PARFAIT to support legacy system understanding and documentation; and also the construction of a tool that controls versions of the systems created from the GREN framework instantiation [5]. We emphasize that, with ARTe, it is possible to reuse not only the pattern solutions, but also the test requirements associated to them.

From that, a fourth case study has been planned to evaluate test requirements reuse, created with the support of the ARTe approach in the reengineering of systems, in order to observe if there is time reduction. This case study has been conducted by an undergraduate student that was taking the second semester of the Informatics Program. In that case study, also conducted in two steps (training and reengineering conduction), guidelines of the ARTe test reuse, and the same legacy system of the previous case study reported, have been used.
As results of this fourth case study, we can mention: a) about 81% of
the test cases created have been based in the reuse of test
requirements available in the patterns, cooperating to reduce the time
spent with VV&T activities in systems reengineering; b) about 57%
of time reduction in relation to results of the third case study, in the
creation of artifacts related to VV&T with reuse of test requirements
associated to patterns and with the application of guidelines of the
reuse ARTe approach; and c) meaningful reduction (52%) of the
total reengineering time.

Another point observed in the fourth case study has been that the
total reengineering time percentage spent with VV&T activities has
been suffered small modification (77%) in relation to the third case
study (86%). This can be justified because in the other PARFAIT
activities not related to VV&T, the time spent was similar in the third
and in the fourth case studies. Another information obtained in the
fourth case study has been a meaningful increase in the number of
test cases (of about de 96%). A reason for that is that the third case
study test cases have been created from the legacy system
functionality, whereas in the fourth case study the test cases have
been created from the reuse of the domain test resources.

Besides the four case studies conducted, an experimentation package
has been defined in the reengineering context, which has as its main
objectives to analyze an agile reengineering process based in
frameworks (PARFAIT in this case). The purpose of this package is
to evaluate the applicability of the framework-based process with
respect to its effectiveness in the reengineering of procedural systems
to OO, in relation to ”ad hoc” reengineering processes. The main
objective of that package construction has been to motivate that
controlled experiments may be repeated both in academic and
industrial environments. That package planning and description were
partially validated with the support of a case study conduction in an
academic environment [6].

The hypothesis formulation has been based on the reuse level of
analysis, design and implementation during the reengineering, on the
quality of the resulting product, and on the effort and time spent in
both processes.

With relation to the experiment design, it can be done individually or
in group. This last one is used when there is interest in analyzing the
reengineering cooperation. For the selection of each group
components, we suggest that one participant experience completes
that of the others, in order to make equivalent the group knowledge
level. The experiment design is in blocks, and there should be
exchange of the reengineering processes and also of the legacy
systems among the individuals or groups, if that is the case.

Initially, a reengineering is conducted using the ”ad hoc” process,
with all the participants (or groups) - Step 1. Then, training is
supplied on PARFAIT and on the techniques involved for all the
participants. Finally, reengineering is conducted using the PARFAIT
process, exchanging systems among participants (or groups) – Step
2. The experiment conduction must follow that sequence to avoid
training influence in the ”ad hoc” reengineering execution.

3. PERSPECTIVES

From the advantages offered by PARFAIT, an agile development
process based in framework has been abstracted from its activities,
named PARFAIT/EA [8]. That has been possible as the
reengineering is composed by the reverse engineering and forward
engineering steps. To observe this process applicability in software
development, a case study is being planned. The hypothesis
formulation will be based under different aspects: a) if the
documentation produced by the process, both for the framework
instantiation and for the adaptation of the version generated by
the framework, is enough to support software development; b)
whether or not the major part of the business rules are identified
during the requirements list definition; and c) if all the agile
practices defined in each process activity are used in all the
iterations.

4. REFERENCES


