Empirical Studies of Evolving Systems

K. BENNETT
Department of Computer Science, University of Durham, UK

E. BURD
Department of Computer Science, University of Durham, UK

C. KEMERER
Katz Graduate School of Business, University of Pittsburgh, USA

M. M. LEHMAN
Department of Computing, Imperial College, UK

M. LEE
Department of Informatics & Simulation, RMCS, Cranfield University, UK

R. MADACHY
Litton Guidance and Control Systems, USC Center for Software Engineering, USA

C. MAIR
Design, Engineering and Computing, University of Bournemouth, UK

D. SJOBERG
Department of Informatics, University of Oslo

S. SLAUGHTER
Graduate School of Industrial Administrations, Carnegie Melon University, USA

Abstract. This paper describes the results of the working group investigating the issues of empirical studies for evolving systems. The groups found that there were many issues that were central to successful evolution and this concluded that this is a very important area within software engineering. Finally nine main areas were selected for consideration. For each of these areas the central issues were identified as well as success factors. In some cases success stories were also described and the critical factors accounting for the success analysed. In some cases it was later found that a number of areas were so tightly coupled that it was important to discuss them together.

Introduction

The first step taken was to identify important issues. These were identified as follows:

1. Rules and tools for effective empiricism
2. Challenge of longitudinal (e.g. evolution over life to date) studies
3. Prediction techniques
4. Validation techniques
5. Forensic evidence; availability and use

6. Data Repository for results
7. Making an impact / raising the profile
8. Development of theory
9. Important Research topics

Each of the above was then considered in turn and the important problems and potential benefits surrounding them were discussed. Not all of the records of the discussions that follow are complete; but each addresses some of the important aspects. Furthermore, there are several instances of a high degree of overlap between the issues recorded.

1. Rules and Tools for Effective Empiricism

Collaborators should be in a position to provide real live systems and should be appreciative of the benefits of the study and expect to receive benefit from the studies. It is essential that there be a champion within each collaborator organisation, preferably someone in a management position, the more senior the better. There must be a high degree of trust between all collaborators. Other important points to consider are:

1. Careful Selection of Systems to Study

Typical criteria for assessment should include:

- Availability of historical data
- Operational systems—one where further changes are likely to be performed
- Significant size—not measured by lines of code, but, for example, a project where there are at least two levels of management control

2. Evolutionary Approach to Investigation

Data analysis should begin as soon as the data permits; part of the total data set should be used as a starting point for analysis. Accumulation of large amounts of data without analysis leads to stagnation and lack of direction. It may be best to start with small amounts of data and use the results of the analysis and the consequent increase in understanding of the phenomenon to guide the future research direction. Early results will also be used to guide the direction of investigation and the classes of further data collection. It also allows the early results to shape the future research directions, yielding an evolutionary research approach.

3. Appropriate Analysis Techniques

These issues include consideration on how to conduct the analysis and what types of analysis approach, methods, techniques, tools, to use. They are described in greater detail later in this paper.

2. Longitudinal Study Challenges

Software evolution studies can be categorised into two types, longitudinal and cross-sectional depending on the approach taken. Longitudinal studies analyse the evolution over the entire, or a significant part of the system lifetime. Cross sectional studies are those that compare software change across projects during some small interval of time.

A number of important challenges were identified. These include:

1. Availability of Historical Databases

Much of the necessary data for a study of evolution is never captured or retained. This lack of data makes the study of long term evolutionary trends difficult.

2. Working with Very Large Data Sets

Software applications are large, and changes are continuous and frequent. When all change data is retained the resultant volume is likely to be very large. The size of the resulting repository can make the analysis process hard—problems include where to begin, how to interpret, how to clean the data etc.

3. Working with Small Data Sets

Certain classes of evolution, or other, software process data are unlikely to comprise more than some 30 or so data points. Statistical analysis, and the determination of significance, poses a major challenge in searching for suitable techniques

4. Missing Data and Changed Definitions

As a consequence of the time that will have elapsed during the evolution of a system data collection methods will have changed, as will priority of collection, personnel, interpretation of definitions and so on. This inevitably poses major challenges to the investigator.

5. Technology Changes Over Time

This issue leads to problems in making comparisons, in interpretation of the results and in assessing their validity.

6. Interpretation and Relevance of Measures Across Projects and Across Organisations

This demands consistency by all working in the field to permit comparison of results and eventually to the generalisation of results. We need to work towards standardisation of terminology and definitions so that, the analysis of evolutionary trends across domains becomes feasible.

7. Lost Organisation and Process Knowledge

Not all information necessary for the study of the evolution process is captured, and even less is actively retained. Industry needs to be guided in which data to collect and retain and educated in the benefits of its retention.

8. Continuing Strengthened, Interdisciplinary, Evolution Research

The problem of understanding software evolution, and how its impact and control is difficult because of the many different variables involved. It is important to establish a common understanding within the computing industry, the software engineering community and the funding bodies about the importance and potential benefit of this work to ensure more consistent funding that is essential for real progress in longitudinal style studies.

3. Predictive Techniques and 4. Validation Techniques

Due to the perceived closeness of these issues they were considered together. Some of the approaches for dealing with them were identified as:

1. Immediate Use of Available Data

Commencement of analysis should not wait until all the data has been collected. An immediate approach allows early formation of results so early feedback which is good for project motivation.

This approach raises, however, a number of issues that need to be resolved. These include:

- Partitioning—how should be results be split into appropriate portions for study
- Framing sets—design of a data set for mutual collaboration

- If analysis and early results are used to redirect an investigation without care they may direct it into difficult or undesired directions or block other, more desirable or beneficial investigations

2. Long Term Monitoring

The availability of live systems means that predictive techniques have the potential of being validated. Long term monitoring provides a means of assessing the validity of the results obtained. For instance, models and, hence, predictions can be made on the basis of existing data and then validated as new data is obtained.

3. Cross-Unit Validation

The applicability of the results obtained from evolution studies needs to be assessed. A number of ways in which this result validation could be achieved were identified. These include:

- Cross system validation—for instance using cross project studies.
- Organisation—comparing results obtained across different organisations
- Technologies—comparing results obtained across different technologies
- Application—comparing results obtained across different application areas

5. Forensic Evidence; Availability and Use and 6. Use of a Data Repository for Results

The availability of forensic evidence is essentially the ability to be able to capture all data relevant to the evolution study. This may include software changes and well as the justifications for the making of such changes. This issue is related to the availability of a data repository and so the two were considered together. Once available data has been identified and retrieved it should, subject to the agreement of the owner, be stored in a repository and made generally available. Collection and retention in a suitable form of such data, should it be available, are those issues encapsulated by the data repository issue.

A number of problems and benefits were identified to surround these issues. These include:

1. Benefits of the Sharing of Data

- Extending the benefit of historical data—historic data is hard to come by and is likely to contain more potential than can be exploited by a single group. Any data successfully extracted should therefore be made available as widely as possible
- Generally available data—more groups are likely to be involved leading to a number of different approaches and interpretations

  - Replication—verifications of results by others
  - Generalisation—analysis by different groups, analysis in and application of results to different domains and their verification may provide opportunities for their generalisation. This can make an important contribution to the development or extension of a theory or theories on software evolution.
  - Leverage of limited resources—the numbers of researchers working within the field of evolution is small so scarce resources need to be maximised.

2. Problems Associated with the Sharing of Data

- Corporate participation—industry must be convinced that neither their competitors nor their clients will gain advantage over them from the study, or rather that any advantage these may gain is less than the benefit to be expected from making the data available for analysis. Thus, important issues to consider include:
  - Confidentiality—there may be problems with publication of results
  - Overcoming such reservations—Potential benefit can be expected to greatly exceed any loss, so both the collaborating owners of the data and the research and software engineering communities should benefit
- Interpretation / standardisation—this applies to the interpretation of results. The ‘champion’ will play a prominent role in providing direct guidance or access to those who may be in a position to help.
- Cost and effort—data collection is performed at a cost this may put industry off its collection

3. Existing Successful Shared Data Storage Projects

The group acknowledged that it is often difficult to reach a consensus on the above issues but pointed to some successes in providing for data source sharing through the creation of appropriately administered repositories. These successes include:

  - UK Social Science Data Archive
  - COCOMO II

It is hoped such successes will help provide the motivation to ensure a shared repository for software metrics in general and evolution research data in particular.

7. Making an Impact / Raising the Profile

The ever-increasing dependence of world-wide-society demands wider exploration, understanding and dissemination of evolution issues. To achieve this requires, in turn, increased prominence in the computer science and software engineering communities as well as active, increased support, including funding, from research drivers. The following form a ‘wish-list’ of factors the group deemed necessary to achieve all this.

1. Recognition of Societal Dependence on Software and its Evolution

The important of software to society must be coupled with recognition of the importance of the adaptability of software and the ability of the software to adapt with the changes required by society. Y2K has highlighted the importance of the change process, but this awareness should be viewed only as a beginning, as the tip of an iceberg.

2. Scientific Content from Disciplined Empirical Approaches

The scientific approach to evolution research should be documented and tested so that a well-defined and respected scientific technology and process for evolution-based research, and for planning and control of software and computer application evolution, can be developed.

3. Computer Science as the Management of Change

All real world software applications that are actively used must evolve, through the evolution of the supporting systems and software. A large portion of the computer industry is involved with the resultant change process. This fundamental industrial activity must be recognised, acknowledged and taken account of in various ways.

4. Inclusion in Computer Science Education

Inclusion in computer science education is one way to acknowledge that software change is a major responsibility and challenge, absorbs large amounts of effort, expertise and other scarce resources in the software industry. Failure to execute the work adequately and on time will increasingly pose a threat to society at large and in the small. This indicates that software evolution and related topics must be included in computer science and software engineering educational programmes. Such inclusion is rare at the moment. Note that it is important to define the core intellectual issues required for the development of a theory early on so that studies can be geared to its formation.

5. Leveraging / Citing Other (Especially Non-Local) Research Work

In addition to its inclusion in the appropriate educational courses, awareness of the importance and potential of evolution research must be acknowledged by others both within and outside the professional research community. Thus, successful research in the area must be widely publicised. One way to achieve this is through referencing widely the variety of research that it being conducted within the area. This will permit others to be introduced to the research area with minimum effort.

6. Software Evolution as the Fruit-Fly of Business Evolution

Software evolution is an instance of more general business and organisational process evolution. Software change is different in that the rate of change with which its evolution process must cope greatly exceeds that of other such processes. Nevertheless, there is good reason to believe that some findings from software evolution research will be relevant and of value in the wider area of business process improvement, now an area of worldwide interest.

7. Applicability of Research

Because of the greater rate of change of software, study of its evolution is likely to bring quicker and richer benefit than evolution studies in other areas. There should, therefore, be more general support for this area of research.

8. Development of Theory

It is well understood that all applied technologies require a theoretical base and framework, a theory, to support and advance them. One of the most important, although long term, outcome of software evolution research is the development of such a theory. Issues surrounding such development include:

1. Does there exist or is there potential for developing a solid conceptual framework for future technological advances including, for example:
   - General principles
   - Laws
   - Relationships and dependencies
2. Is it a ‘hard’ or ‘soft’ theory?—for instance, does it have a mathematical or sociological basis?
3. A strong position including solid evidence is required to back up the case. Such evidence should, preferably, be applicable across domains.
4. The interdisciplinary nature of any theory of software evolution, its relationship to and dependence on such areas as computer science, control theory, statistics, organisation theory, management, cognition, psychology, marketing and so on must be recognised and the appropriate links incorporated.

9. Important Research Topics

Many of the above issues identify some of the problems that are associated with research in the area of software evolution. The group turned it attention to the future and decided to define long term and short term strategies for research. Some of the approaches proposed were:

1. Long Term Strategies

Strategies are needed to:

- Demonstrate convincingly that evolution is an intrinsic property of computer applications.
- Understand the nature of evolution, including:
  - Laws
  - Patterns
  - Impact
  - etc.
- Gain an understanding of how to control evolution, including, for example, the study of the feedback mechanisms that drive evolution and help determine its direction.
- Overcome the laws—Since the laws known today are primarily a reflection of the behaviour of people and organisations, understanding of the control mechanisms within evolution may ultimately make it possible to overcome them or lessen any undesirable implications, for example through the provision of automated support.
- Meta-analysis—concerning the philosophical basis of the way in which we propose, hypothesise, test and gain evidence.
- Evolution support—Through developing a theoretical framework, as exemplified by the current laws of software evolution and through gaining deeper understanding of the laws and the phenomenology that underlies them. The studies of evolution can lead to a long term benefit for the technology of software systems change and to the security and benefit of computerisation.

2. More Immediately Applicable Strategies

Strategies are needed to:

- Provide an interpretation for characterising and controlling evolution, including:
  - Measurement
  - Assembly

    - Analysis
    - Application or exploitation of results

- Perform, document and publish case studies and their characterisations to provide data for shared repository
- Include analysis of decaying systems—to provide possible evidence of the consequences of poorly controlled evolution
- Plan for a public data repository.

3. Techniques for Conducting Research

The proposed investigations of evolution can make use of a variety of techniques, methods and tools: These include:

- Black box
- System dynamics
- Statistical techniques—especially as applicable to small and very small data sets
- Time series analysis for longitudinal studies
- Techniques for studying change e.g. from social sciences where studies are made of event histories
- Data mining—this is an archaeological style of approach to data interpretation
- Fuzzy dynamics
- Control theory

Note that the above list includes only "technical" approaches to analysis. Other approaches such as those used in areas such as those listed in section 8, subsection 4 must also be considered in conjunction with experts. The determination, customisation and application of all such approaches should be shared within the research community to ensure maximum effectiveness, benefit and, above all, consistency of results.

Conclusions

The issues described within this paper from the discussion of the working group on empirical studies of software evolution are each critical to successful maintenance of software applications. The group identified a number of issues that need to be resolved in order that continued research within the field of software evolution is to be successful. One critical issue is increased collaboration of researchers within the field, where results are shared possibly through a shared and united repository. The difficulties of such collaboration were debated but overall it was decided that the benefits of such shared resources were worth the effort.

Overall if was the group's resolve that effort should be directed towards gaining a better understanding of the process of software evolution and that effort should be concentrated on deriving laws and principles of software evolution. From this improved understanding better predictions regarding the changes to software will be available and in general more control of the process of software evolution will then be possible.