Unifying Microprocess and Macroprocess Research

Leon J. Osterweil

University of Massachusetts, Department of Computer Science, Computer Science Building,
Amherst, MA 01003, USA
ljo@cs.umass.edu

Abstract. This paper proposes the unification of two complementary approaches to software process research. The two approaches can be characterized as macroprocess research, focused on phenomenological observations of external behaviors of processes, and microprocess research, focused on the study of the internal details and workings of processes. The paper suggests that it is time to bring these approaches together with the goal of using microprocess methods to provide definitive explanations of observed macroprocess behaviors. The paper suggests that this unification could lead to improved understandings leading to improvements in software development practice. The paper observes that such positive outcomes have resulted when the macro- and micro- approaches have been synthesized in domains such as Economics, Physics, and the Life Sciences.

1 Introduction

The recent years have been gratifying for the community of people who focus attention on the importance of process. Process has long been recognized by manufacturing industries as the key to achieving control over such core issues as costs and quality levels. In such industries there is considerable agreement that "quality products result from quality processes", for example. Recognition of the possibility that these same ideas hold for software development has been more recent. One key event in advancing the importance of process was the 1984 Software Process Workshop [1], held at Runnymede, England, although many will point out that there had already been a focus on process (e.g., in large companies such as IBM), prior to this event.

The past two decades have seen a steady growth in interest in the application of process to the solution of many problems and issues in software development. Process has been identified as being a vehicle for controlling development costs and achieving product quality (in strong analogy to the situation in industries that manufacture tangibles). Process has also been identified as a vehicle for supporting more effective training, for superior project coordination, for improved management visibility (leading to more

effective management), and for studying how to achieve the most effective deployment of resources.

With this diversity of applications of software process, it is no wonder that there is also a diversity of approaches to the development of process technology. This paper explores the possibility that two complementary types of software process technology research might be integrated so as to support each other and provide substantial benefits to the development community.

We propose the term *macroprocess research* to describe investigations that have emphasized the study of overall behaviors of process, and the term *microprocess research* to describe investigations that have emphasized the study of the internal workings of processes. There seems to be a natural complementarity between these two approaches, with the former investigating gross external effects, and the latter investigating the root causes of these observed effects. In this way, this dichotomy strongly mirrors similar dichotomies in such other disciplines as economics, physics, and the biological sciences. As in those other disciplines, it seems promising to consider the integration of these approaches.

This paper proposes that software process research increasingly focus on the systematic investigation of how external process behaviors are definitively explained (and potentially improved) by examination (and modification) of the internal structure and details of the processes themselves.

2 Macroprocess Research

We propose that macroprocess research be characterized as investigations whose focus is on the study of the external behaviors of processes. Such behaviors include the speed of execution of processes, the characteristics of the software products they produce, the way in which resource infusion affects product nature and process speed, and the effect of changes in production timetables upon products, and the processes themselves. Much of this research emphasizes the determination of appropriate measures of such characteristics as product quality, worker productivity, and process efficiency. It is not surprising, and indeed appropriate, that much of this research is empirical in nature.

Some of the most notable directions in this research are represented by such projects as the Software Engineering Institute's Capability Maturity Models (e.g., CMM and CMMI) [2, 3], the Experience Factory project [4, 5], and the many attempts to define measures of software quality and process characteristics [6]. It is undeniable that these efforts have had a noticeable positive effect on the way in which software is developed, and upon industrial understanding of many of the principal issues that need to be understood if software development is to be practiced as an order, disciplined, predictable activity. These is, for example, now a general understanding that, despite the obvious difficulties, effective measures must be defined and applied to software products in order to determine their size, cost, and quality. There is general agreement that processes themselves must be

studied in order to determine how to change them in order to achieve desired improvements.

As the macroprocess work progresses, however, it becomes increasingly apparent that there will be considerable value in complementing a strictly external, phenomenological view of software process by investigations of process internals in order to develop explanations of the causes of observed phenomena. Later in this paper, we support this view with some analogies. For now, however, we simply suggest that the desire to understand *why* something works is more than just natural human curiosity. Knowing the cause of a phenomenon provides an avenue for approaching its control. We may observe that there seems to be a linear relationship between cost decrease and infusion of additional resources. But does that linear relationship extrapolate indefinitely? Or can the relationship be expected to hold only over a restricted range of circumstances? If so, then how can that range of circumstances be characterized? Certainly additional empirical studies can provide additional insight, but a firm grip on the causes of a phenomenon can provide predictive power that should be expected to reduce the need for empirical studies, or eliminate them entirely.

The macroprocess research community has demonstrated a clear recognition of the need to determine the causes of the external behaviors that have been observed and codified. Often the search for causal relations has been based upon empirical studies and statistical investigations. Other approaches have been based upon process models. At this point, the macroprocess approach begins to take on the character of microprocess research.

3 Microprocess Research

We propose that microprocess research be characterized as investigations that focus on investigation of the precise specification of the details of software processes, for the purpose of inferring how those details effect the external behaviors of the processes. Much of this research has focused on the identification of languages and semantic features that are effective in supporting precise process definition. Other research has focused on the use of such linguistic features to describe and define specific processes (e.g., system evolution [7]). These process descriptions and definitions are then sometimes used as the basis for supporting process automation, or semi-automation, and for supporting reasoning about and analysis of these processes. An important goal of some of this work is to create a discipline of process engineering, in which process definitions can be crafted to deliver desired process behaviors, and product characteristics.

Some examples of work in this area include the Process Instance Evolution (PIE) Project [8], as well as process language efforts such as Adele [9, 10], Spade/Slang [11], Marvel/Oz [12, 13], and Little-JIL [14, 15]. It is worth noting that there are parallel efforts at identifying languages effective in supporting definitions of process in other domains. For example workflow research aims to support definition of business processes, and

enterprise framework modeling aims to support definition of internet-based processes for supporting innovative business models that use emerging network technologies [16]. A common theme in all of this research is the determination of how to characterize processes effectively, so that they can be carried out more satisfactorily, and so that their properties (both positive and negative) can be predicted accurately.

As such microprocess research can be seen both as providing something that macroprocess research needs, and needing something that macroprocess research provides. A key goal of microprocess research is to provide detailed, accurate, low-level definitions of processes, and reasoning capabilities that are able to predict and explain the high level phenomena discovered by macroprocess investigations. On the other hand, the phenomena that have been identified, and shown to be of greatest interest, by macroprocess research seem to be of central importance to microprocess research, in indicating the phenomena whose explanation seems to be of greatest interest and importance. Given that there are an infinite number and variety of process properties that might be studied, and a wide spectrum on analysis approaches that might be applied to process definitions, microprocess research needs a focus on the phenomena of greatest interest. Macroprocess research can provide just such a focus.

Here too, we note that microprocess research is just beginning to identify, and evaluate, process properties about which reasoning seems possible and useful. In this respect, microprocess research is beginning to address issues that might more accurately be characterized as macroprocess subject matter.

Thus, it seems that the two approaches, macroprocess and microprocess, are starting to reach tangency with each other, and have much to offer each other. The suggestion that an integration of these research directions therefore seems both logical and timely.

4 Some Analogies

The use of complementary approaches such as those suggested here is certainly not unique, and is indeed long predated by, similarly complementary approaches in such other disciplinary areas as Economics, Physics, and the Life Sciences. The success of such complementarity should be instructive and encouraging.

Economists have long distinguished among themselves using the complementary approaches of macroeconomics and microeconomics. The analogy to software process is particularly striking. Macroeconomics studies large-scale phenomena, and the gross behaviors of economies in response to large-scale forces. Such relations as how economic growth responds to interest rates, and fiscal policy are within the purview of macroeconomics. Microeconomics, on the other hand, emphasizes the creation of models of smaller scale behaviors and phenomena, seeking to use them to explain and predict the phenomena of macroeconomics. Thus, for example, microeconomists may employ systems of linear inequalities, and analytic approaches such as game theory, using complex mathematics, in order to explain the behaviors and properties of markets that

have been observed by macroeconomists. This dichotomy seems to provide a strong and close parallel to our suggestions about the complementarity between microprocess and macroprocess research. To go further with this analogy, it seems important to note that microeconomics can support a diversity of reasoning about its models, but the reasoning that is most important and most relevant is that which is directed towards explanation of behaviors whose importance has been established by macroeconomists.

Physics provides other analogies that seem interesting and relevant. Early physicists, such as Boyle, determined that air was "springy". When enclosed in an airtight container, pressure on the enclosed air was resisted, and when the pressure was released, the air sprang back. Compressing the air seemed to heat it up. Careful investigations of this, and other, external behavioral phenomena eventually resulted in the formulation of Boyle's Law, which related air pressure, volume, and temperature. Boyle's Law provided useful guidance for centuries before an explanation of this behavior was provided by statistical mechanics. Eventually it was explained that air consists of myriad molecules bouncing unceasingly off of each other and the sides of containers. Mathematics was used to demonstrate how this internal structure and behavior explained Boyle's Law. Conversely Boyle's Law's external verification of the predictions of statistical mechanics helped confirm the hypothesis that air is composed of molecules with elastic properties, leading to innumerable other scientific advances. Other examples from Physics are not hard to find. We note, for example, that various behaviors of electricity were observed, and even described by formulas, before physics determined the root causes of these behaviors. Light was bent and focused long before its wavelike nature was understood. But that understanding led to better engineering of light.

Indeed, Physics provides a long list of illustrations that the observation of phenomena generally precedes their explanation, but that the explanations generally lead to improved management and engineering of the phenomena. The Life Sciences provide more examples of this.

Physicians, and primitive healers before them, have had growing success in treating human ailments simply by observing external phenomenology. Even the earliest healers understood that bleeding and high fevers were dangerous and problematic. They focused attention on controlling them long before there was an understanding of the role of blood (and hence the danger posed by its loss), or the specific impact of increased body temperature upon the functioning of the body's organs and systems. The importance of cleanliness, both in public health, and in treatment therapies, was observed long before there was a recognition of the existence, and dangers posed by, microorganisms. Once the nature of microorganisms was understood, and the details of the processes by which they caused disease understood, superior therapies (e.g., antiseptics and antibiotics) could be deployed to greater effect.

Currently we are seeing that basic understandings of DNA, molecular biology, and cell biology are leading to clearer understandings of viruses, and are leading to the more effective treatment of maladies ranging from the common cold to cancer. It is noteworthy that these advances are being derived from micro-level understandings of their causative

factors, while phenomenological observation of these diseases has been relatively ineffective in advancing their treatment.

Thus, there seem to be numerous examples of the complementarity of macro- and micro- level research in other disciplines, and considerable reason to expect that parallels with software process research are valid. This seems to us to add more credence to the suggestion that this complementarity should be studied and pursued.

5 Future Directions

As macroprocess research and microprocess research continue to widen their scopes, their intersections can be expected to continue to increase. What seems needed now is some consensus about one or more projects that have the potential to cause each to gain a better understanding of the other, and to find in the other the sort of complementarity that will lead to added value for both. While there would seem to be many such projects, one will be suggested here, more in the spirit of being specific than in an attempt to be prescriptive or exhaustive.

A principal problem in software project management is the need to determine what mix of skills and other resources are needed at different phases of a software project in order to maximize productivity. Numerous empirical studies have suggested various behaviors that seem useful and applicable. Thus, for example, there have been numerous examinations of software design. These studies have suggested, for example, that experienced designers tend to be more productive and effective than novices, especially on larger, more unprecedented, projects. But highly experienced designers may be overkill for some less ambitious projects. Similarly, adding more designers seems to be effective in larger projects. But it is all too possible to put too many designers to work on a project. Empirical studies have come up with statistics and numerical measures that suggest how to quantify these qualitative observations. Still, however, the quantifications are generally interpolations and extrapolations from relatively small sets of observations. And the possibility that some of these observed behaviors may be affected by yet-undetermined factors still exists.

It would seem reasonable to propose that microprocess approaches might help here. Microprocess research [17] has resulted in the development of definitions of processes that seem to help novices produce better designs more rapidly. These processes seem to rely importantly upon the ability to apply design constraints over specified scopes, and upon the ability to accurately describe, and thus more effectively support, the elusive notion of “rework”. This seems to be a good example of how microprocess research, focused on developing appropriate process definition language constructs, and using them to develop more precise and accurate process definitions, can develop understandings of the nature of a key software process like design. We suggest that understandings such as these, drawn from microprocess research be used to create definitions of specific design processes, indicating precisely how various designers, of various skill levels, and other

resources, could be used to develop designs. We suggest it might then be possible to then apply analyzers and reasoning approaches to explore how different numbers and mixes of resources and design expertise levels might affect the progress of a design. Many different design processes have been defined, and more could be defined. But macroprocess research would, in this case, be used to suggest specific characteristics and desiderata in a design process that would make it particularly worth studying, and would then focus the process definition and analysis.

Other such integrative projects would seem to be relatively easy to identify. Joint pursuit of them would seem to offer important benefits for both macroprocess and microprocess research, and for the overall discipline of software engineering.

6 Acknowledgments

This material is based upon work supported by the US National Science Foundation under Award Nos. CCF-0427071 and CCR-0204321. The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of The National Science Foundation, or the U.S. Government.

References

1. *Software Process Workshop*, in *Software Process Workshop*. 1984. Runnymede, England.
2. Paulk, M.C., et al., *Capability Maturity Model Version 1.1*. IEEE Software, 1993: p. 18-27.
3. *Capability Maturity Model® Integration*. http://www.sei.cmu.edu/cmmi/
4. Basili, V.R. *Software Development: A Paradigm for the Future*, in *COMPSAC'89*. 1989. Orlando, FL.
5. Basili, V.R., *The Experience Factory and its Relationship to Other Quality Approaches*, in *Advances in Computers*. 1995, Academic Press, Inc.
6. Boehm, B.W., *Software Engineering Economics*. 1981, Englewood Cliffs, NJ: Prentice-Hall.
7. Valetto, G. and G. Kaiser. *Using Process Technology to Control and Coordinate Software Adaptation*, in *Twenty-fifth International Conference on Software Engineering*. 2003. Portland, OR.
8. Cunin, P.Y., et al. *The PIE Methodology - Concept and Application*, in *EWSPT-8*. 2001. Witten, Germany: Springer-Verlag.

9. Estublier, J. *A Configuration Manager: The Adele Data Base of Programs*, in *Workshop on Software Engineering Environments for Programming-in-the-Large*. 1985. Harwichport, MA.
10. Estublier, J., et al. *An Approach and Framework for Extensible Process Support System*, in *9th European Workshop on Software Process Technology (EWSPT 2003)*. 2003. Helsinki, Finland.
11. Bandinelli, S., A. Fuggetta, and S. Grigolli. *Process Modeling in-the-large with SLANG*, in *Second International Conference on the Software Process*. 1993. Berlin, Germany: IEEE Computer Society Press.
12. Kaiser, G.E., P.H. Feiler, and S.S. Popovich, *Intelligent Assistance for Software Development and Maintenance*. IEEE Software, 1988. **5**(3): p. 40-49.
13. Ben-Shaul, I.Z. and G. Kaiser. *A Paradigm for Decentralized Process Modeling and its Realization in the Oz Environment*, in *16th International Conference on Software Engineering*. 1994.
14. Wise, A., *Little-JIL 1.0 Language Report*. 1998, Department of Computer Science, University of Massachusetts: Amherst: Amherst, MA.
15. Wise, A., et al. *Using Little-JIL to Coordinate Agents in Software Engineering*, in *Automated Software Engineering Conference*. 2000. Grenoble, France.
16. Estublier, J. and S. Sanlaville. *Business Processes and Workflow Coordination of Web Services*, in *IEEE International Conference on e-Technology, e-Commerce and e-Service*. 2005. Hong Kong.
17. Cass, A.G., S.M. Sutton, and L.J. Osterweil. *Formalizing Rework in Software Processes*, in *Ninth European Workshop on Software Process Technology*. 2003. Helsinki, Finland: Springer-Verlag.