Chapter 4
Putting a TAG on Software
Purchaser-Centered Software Engineering

Mike Barker
Nara Institute of Science and Technology, Japan

Kenichi Matsumoto
Nara Institute of Science and Technology, Japan

Katsuro Inoue
Osaka University, Japan

ABSTRACT
This chapter describes the evolution of approaches to empirical software engineering from goal and data-driven to the latest purchaser-centered approach. The new Japanese Software Traceability and Accountability for Global software Engineering (StagE) project is developing this approach to ensure the transparency of software development processes and products for software purchasers by “tagging” software with empirical software development data. Global software development raises unprecedented difficulties for developers, including the international and intercorporate coordination of development and distribution, the change to composition as the primary development approach, the shift to software everywhere talking to everything, and continuing upgrades and interaction with released software. To work effectively in this environment, empirical data collection, analysis, and feedback must extend throughout the software lifecycle including both production and usage data.

INTRODUCTION
But when software doesn’t work right, all too often the development information is long gone. The developers are on other projects, and the development documents are not available. No one knows exactly how it was configured, or what the usage environment has been. So crashes are shrugged off or ignored, because there’s no way to look back. Or is there?

DOI: 10.4018/978-1-60566-731-7.ch004
This is the problem that the StagE project addresses. How can we collect data during the software lifecycle and then keep track of it so that we can trace root causes of errors?

This chapter discusses the issues that lead to the StagE project, which is currently investigating the use of software tags as a way to collect data, especially during development, which provides transparency, and then maintain the connection between the software products and the data during the software lifecycle, creating traceability. The chapter gives a brief background on empirical software engineering, then describes the purchaser-centered approach that software tagging supports. It then looks at how widespread use of such software tags might affect the software development process, especially in the global context. This chapter is a report on a work in progress, so while it outlines problems and the proposed approach of using software tags to provide purchasers with additional visibility into development processes and traceability across lifecycle and development boundaries, it does not report on the success of such use. You, the readers, will be the developers and purchasers who determine whether software tags are used, and how successful they are.

BACKGROUND

Top-Down, Bottom-Up, or Sideways: Getting the Data Right or Getting the Right Data?

One of the key questions in empirical software engineering is exactly what data do you want to collect. Empirical software engineering has used two basic approaches to decide what data to collect. The first, goal-driven metrics, typically starts with high-level goals or business directions and works down to specific data and metrics. One difficulty with this is that the data collection often is very specific to the environment and projects. The second approach has been to start with automated data collection and work upward to develop abstractions and analyses. Perhaps the leading example of this is the Hackystat project, which provides a wide array of data collection tools and a platform to tie them together. (Johnson, 2008) One of the difficulties with this approach has been linking the low-level data to business goals and abstractions.

Goal-driven metrics and data-driven approaches have shown the abilities of empirical software engineering to improve the software process. However, the managers and developers often have different interests from the purchasers of the software, suggesting that changing the stakeholders driving the selection and use of the data can provide a more effective process for selection and application of empirical measurements. But before we look at that new approach, let’s take a brief look at the older approaches.

The Conventional Approach to Empirical Software Engineering: Goal-Driven Metrics

Today, there are several national projects on empirical software engineering in Japan (EASE Project, 2007; SEC, 2008), Australia (NICTA, 2008), Germany (IESE, 2008), and the USA (CeBASE, 2004). We can find many research papers concerning empirical topics in major conferences in software engineering.

In most conventional projects and papers, only software developers use empirical data about software development to improve software quality and productivity. However, developers’ needs for software quality and productivity are often too abstract to relate to the data collected in software projects. Models and techniques that derive the “data to be collected and analyzed” from the “goal to be achieved” play an important role. The GQM (Goal/Question/Metric) approach proposed by Prof. Basili, and Measurement Information Model defined in ISO/IEC 15939 may be helpful. In addition, real-time data collection, analysis, and
Related Content

Japanese R&D Profitability: Industry Production Function Estimation Using Panel Data
[www.igi-global.com/article/japanese-rd-profitability/213952?camid=4v1a](www.igi-global.com/article/japanese-rd-profitability/213952?camid=4v1a)

An Introduction to Multiformalism Modeling
Marco Gribaudo and Mauro Iacono (2014). *Theory and Application of Multi-Formalism Modeling* (pp. 1-16).
[www.igi-global.com/chapter/an-introduction-to-multiformalism-modeling/91938?camid=4v1a](www.igi-global.com/chapter/an-introduction-to-multiformalism-modeling/91938?camid=4v1a)

EXTREME: EXecuTable Requirements Engineering, Management, and Evolution
[www.igi-global.com/chapter/extreme-executable-requirements-engineering-management/78209?camid=4v1a](www.igi-global.com/chapter/extreme-executable-requirements-engineering-management/78209?camid=4v1a)

VirtualCareGiver: Personalized Smart Elderly Care
[www.igi-global.com/article/virtualcaregiver/169916?camid=4v1a](www.igi-global.com/article/virtualcaregiver/169916?camid=4v1a)