An Asset View on the Software Process

R. Mark Greenwood, Peter Kawalek, Ian Robertson, Brian C. Warboys

Informatics Process Group, University of Manchester, M13 9PL, UK.
http://www.cs.man.ac.uk/ipg/

Proceedings 10 th. International Software Process Workshop, Ventron, France June 1996.

Copyright 1996 IEEE.

Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

An Asset View on the Software Process

R. Mark Greenwood, Peter Kawalek, Ian Robertson, Brian C. Warboys

Informatics Process Group, University of Manchester, M13 9PL, UK.
http://www.cs.man.ac.uk/ipg/

Abstract

There are issues of concern to software engineers that cannot be addressed in the context of a single development project. This paper indicates some typical issues, and proposes an "asset" view of the software development process to enable them to be considered. The asset view places less emphasis on the temporal ordering of activities and more on the concepts of goals, interactions, learning and evolution. It is complementary to the more conventional "production" view. The asset view may be a better starting point for considerations of reuse and product lines.

1. Introduction

In [2] Feiler and Humphrey define the term process, in the software context, as "a set of partially ordered steps intended to reach a goal". This reflects a *production* view which is common to many software process modelling approaches. In this paper we propose a complementary *asset* view. While both views can be applied to individual software development projects, the asset view is particularly suited to scaling up to organization-wide issues such as software product lines and continual process evolution.

2. Typical Software Development?

In the majority of software process models the goal is the production of a software system which meets its requirements on time and within budget. The process normally starts with some expression of customer needs as its initial input and delivers the complete system as its final output. The most common motive for ordering activities within a process is an output of one activity being used as an input to another. This production view of software development seems to be particularly appropriate in situations where the software development organization's "modus operandi" is to temporarily reorganize part of itself to focus on a single project.

However, there is great variety among the organizations who are involved in developing software. While some develop specific products for clients, others produce a suite of products which they intend to market, and for others the main focus is to specialise commercial-off-the-shelf (COTS) components for their own or colleagues' use.

3. Asset view

All models are abstractions, so it is no surprise that it is possible to adopt a different view of software development and come up with a different model. One view which we have found appropriate, in all kinds of software development organizations, is the *asset* view. In the asset view an organization produces a stream of software systems in response to a stream of client requests; in doing so it continually uses and enhances its software development assets.

In the asset view the goal of producing new software systems is placed within the context of the organization's need to re-use and develop its software development assets. In order to provide a cost-effective response to clients' requests the organization must re-use some of its existing assets. In addition, it is in producing new software systems that the organization can enhance its assets; investment which it expects to yield benefits in future projects.

There is considerable evidence that organizations use an asset view in managing their software development, even if explicit models are rare. In most organizations many decisions about how software will be produced, such as whether to move to a client-server architecture, whether to move to an object-oriented language, or whether to buy a specific code library, are not taken in the context of a single project. The organization makes these decisions based on their impact over a number of projects. This is especially true when there are a number of closely related projects, as in software product lines. In addition, it is clear that both those people involved with an organization's longer term strategy and individual software engineers have horizons which extend beyond individual projects.

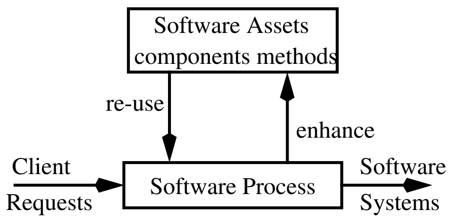

Figure 1. A High-level Asset View

4. Coordination

In the *production* view considerable emphasis is given to temporal ordering ("partially ordered steps [2]") within a process. An alternative is to decompose a process into distinct areas of responsibility and the interactions between them. Consider the abstract view of a software development organization in figure 1. It is not unusual for an organization to have a group involved in maintaining its assets which is distinct from the development projects. Obviously, this group will have to interact with the projects to provide assets to be re-used, to receive feedback on their value, and to acquire potential new assets for the future. It is far less clear to see when these interactions will take place. We would expect ongoing coordination between the groups throughout the project, rather than the project simply receiving "appropriate" assets at the beginning and providing "possible" assets at the end of the project. We have had some success building models for organizations which are based around identifying the different areas of responsibility ("roles") and the goals which cause them to interact. For example one group might have the goal of maintaining the organisation's code library while another might have the goal of reusing as much code as possible on a specific project. These related goals will lead to interaction between the groups.

5. Evolution and Learning

An organization has both product assets and process assets. The product assets: libraries, source code, designs, documentation, test plans etc., come from the software which the organization has produced. The process assets: review policies, configuration management policies, exemplar project templates, cost and risk models etc., come from the organization's experience of producing software. The management of these assets is a core part of how the organization learns and evolves.

An organization's models of its development processes are themselves part of its process assets. Reflexivity is vital. The effective deployment of an organization's process assets depends not just on identifying and applying the "correct" process, but also on the ability to adapt to circumstances. The need to find ways to manage process assets permeates our work on the meta-process [1,3].

6. Conclusions

The software development process can be modelled from a number of viewpoints. The customer, or project, focused *production* view, and the organization wide *asset* view appear to be complementary. Each emphasises important aspects of the process. The organizational perspective of the asset view suggests that it is a better starting point for considering multi-project models such as those dealing with software product lines and product and process re-use.

One area where asset based models differ from production based ones is their reduced emphasis on when things happen. We have been using a modelling approach which is based on areas of responsibility ("roles") and the goals which cause interaction between them. We see it as a high-level approach through which the model acts as a framework for the interrelationships of assets in the broadest sense.

Whenever we try to flesh out the details of an asset view model we come across the issue of reflexivity. If we want to provide support for assets models then our technology will have to be reflexive.

Finally, the close association of a production view model and an asset view model are not specific to software development. For example, an insurance company could model the resources employed in handling claims as a whole as well as modelling how an individual customer's claim was dealt with.

7. References

[1] J. Derniame and A. Fuggetta, editors. *Software Process: Principles, Methodology, Technology.* Meta Process chapter - In preparation.

[2] P.H. Feiler and W.S. Humphrey, "Software Process Development and Enactment: Concepts and Definitions", in *Proc. of the Second International Conference on the Software Process*, Berlin, Germany, February 1996. IEEE Computer Society Press

[3] B.C. Warboys, "The IPSE 2.5 Project: Process Modelling as the basis for a Support Environment", in *Proc. of the First International Conference on Software Development, Environments and Factories*, Berlin, Germany, 1989. Pitman Publishing.