The Introduction and Evaluation of Object Orientation in a Company Developing Real-Time Embedded Systems

Colin J Theaker, Neil Blackwood, Robert Mason
Terrafix Limited, 23c Newfield Industrial Estate, High Street, Tunstall, Stoke-on-Trent, England, ST6 5PD
cjt@terrafix.co.uk

Abstract. This paper considers the practical experiences of a commercial company when undertaking the move to an object oriented paradigm, and the impact that the paradigm shift has entailed, both in terms of the product quality, and the process for software development. The context for the work is outlined, in particular identifying the demanding nature of the company's product development.

A significant aspect of the move to object orientation was the selection of appropriate technologies and tools to support the development, and the adaptation of the toolsets to suit the company context. A rigorous evaluation of the move was undertaken as part of an ESSI Process Improvement Experiment - PIOJAVA, and the initial experiences of collecting process and product metrics are described.

Introduction

Terrafix is a UK company that produces leading edge command and control systems, specifically involving vehicle location, data communication and control room management. The main application areas are for the emergency services and other organizations that require a command and control capability, including facilities for mapping and vehicle tracking. These systems are highly software dependent, with tight constraints that involve complex real-time, multi-tasking, distributed and communications intensive requirements spanning diverse platforms.

The customer base is also distinctive, and this impacts on the specific functionality of the systems, which are tailored to the individual user requirements, and the needs to support thousands of individual mobile/portable units. This has obvious implications on the maintainability of the system components, and particularly on the ability to upgrade as new technologies are introduced. It has also been noted that clients often request software related changes to the functionality at short notice.

To meet these market needs, the company must be responsive to this very specialized market, and consequently the ease of software production and change is of significant importance to the business. At the same time, high quality, reliability and performance in the most cost-effective way are all expected, particularly as many of the systems are safety-critical, and system failures in areas such as the ambulance services have a very high visibility [London Ambulance Service, in Flowers 1996]. This
imposes particular constraints on the quality of the delivered product, and the needs of the company to provide systems of auditable quality.

Historically, the software development process at Terrafix has been defined and managed by written procedures, which the software engineers are required to observe. Quality assurance is also very paper based and manual. It is obviously in the interests of the company to move towards more automated forms of quality management.

Most existing code is written in C and has been modified at frequent intervals for over a decade. This has resulted in highly functional but difficult to maintain and modify software modules. Portability across platforms is also a problem.

The company has recognized the need for a radical change in its software approach and to this end, the company decided to review its software engineering processes, and to adopt object oriented techniques within its development programmes. Support for this change has come from a number of sources, including the UK Department for Trade and Industry, and an assessment of the impact of the change is part of a Process Improvement Experiment, PIOJAVA, funded by the European Union.

Changes being introduced within the company

It was envisaged that a move to an object oriented paradigm for both design capture and development would have an impact in terms of both the development process (particularly with respect to module reuse and distribution) and to the products themselves. New design techniques and languages would have to be introduced, and this would entail staff training and also a learning curve as the software engineers gained experience in the techniques.

The choice of precise language and design notations was relatively straightforward. Java was seen as the most attractive language, particularly as many of the company products are based on mobile computing applications and platforms. The rationale was that Java is designed to be modular (due to its enforced object oriented structure), multi-tasking (due to its user controllable multi-threading capability), platform independent (due to a fixed strict binary interface and virtual machine approach) and tightly structured (being defined in such a way that some of the vagaries of C and C++ are not allowed). It can be compiled onto diverse platforms for speed advantages, and processors directly running Java byte-code are available and could be incorporated into the company products. It is becoming the de-facto standard for mobile and SMART card applications, and most importantly, provides the capability to integrate simply with communications networks. The potential for code reuse between the mobile systems and the workstation-based command and control stations was of particularly interest and is one of the aspects of the paradigm change that is being measured.

In terms of the design notations, the scope of the techniques in UML [Rumbaugh, Jacobson and Booch 1999] was attractive, particularly as it includes mechanisms such as State Transition Diagrams and Sequence Diagrams, which are of particular interest for real-time software development. Extensions to provide timing behaviour, as described in [Douglass 1998] were also positive points. The greatest weakness was seen as the development process itself, and many of the techniques appeared disjoint