Mechanisms for Cooperation (Chair: Christer Fernström)

Current Issues on Integration

Vincenzo Ambriola*   Giovanni A. Cignoni*   Christer Femström**

*Dipartimento di Informatica, Università di Pisa, Italy   **Cap Gemini Innovation, France

Introduction

Much has been said about the need for powerful integration mechanisms in process-centred (software engineering) environments. In many cases, however, issues that have been raised stem more from a general wish for “total integrability” (where environment builders freely combine tools to work on any data produced by any tool) than from a careful analysis of what is really needed in terms of integration. In addition, many approaches build on (or try to cope with the deficiencies of) fairly low-level mechanisms, such as those available in standard UNIX systems. In order to achieve rapid progress in the deployment of software process technology, we suggest that researchers and practitioners more carefully analyse the minimal needs for integration and seriously consider how the emerging integration technology for personal computers can be applied in the context of process-centred environments.

Process-Centred Environments Needs

Software process technology helps supporting the software process by covering primary issues such as interaction with people involved in the process, the production and management of project documents, and the execution of development tools. In this perspective the ideal process enactment support is logically made of two components upon which the environment is built:

- the process program interpreter that supports the execution of the process activities as described with a process definition language;
- the environment integration layer that supports the management of the objects involved in the process (documents, products, development tools), their integration, and their integration with underlying networking systems.

Generally speaking, integration concerns interoperation between tools and documents. From the point of view of the software process, we are usually mainly interested in the aspects of coarse-grained integration (between tools and the process control mechanism), where data produced and updated at certain stages of the process is consistently made available to other parts of the process, while fine-grained integration (integration among tools) is much more an issue for tool set builders.

The design and the implementation of a language for process control is a well-defined problem. The knowledge of requirements and the available technology make this problem

---

1 By tool set we mean a tightly integrated set of tools, typically used in combination, often by a single user, during an activity in the software process. Typical tool sets are design tool sets (design tool + design rule checkers) or programming tool sets (editor + compiler + debugger).
autonomous, guaranteeing a large degree of freedom. To date, many process modelling languages have been invented, proposed, realised, and even used.

The realisation of the environment integration layer, however, is generally a harder problem. In addition to the general concerns relevant to systems integration, such as concurrency control, networking, security, etc., the following issues (often underestimated) are constraints to the freedom of the environment implementation. In a more pessimistic view, they constitute serious impediments in the attempt of building an effective solution:

- The environment builder cannot choose the tools to be integrated, as they are imposed by the specific processes, and thus need to be introduced by the user organisation. Hence, the environment needs to be able to accommodate new tools and the integration cost must be small, in order to avoid that the set-up of the process environment be too heavy a burden on the process itself.
- A too loosely coupled integration can leave back-doors to escape the process; editors that allow the user to open documents independently from the process, tools that give access to other tools or to system shells.
- Interchange formats for documents are imposed by the adopted tools: this complicates the integration and the control of the flow of documents inside the process.
- The process is often based on heterogeneous platforms: for instance, in the same process the documentation could be maintained on personal computers while the development is performed on a network of workstations (with different UNIX flavours) or on a mainframe.

The majority of current research on process technology is aimed at producing demonstrators or experimental prototypes and, when analysed under the previous argumentation, show their weaknesses. Effective solutions for integration among tools, documents, process control and network resources in an environment suitable for process enactment do not seem to be available.

New Perspectives

We fully recognise that there are technical obstacles that slow the realisation of process-centred environments able to fully satisfy the integration requirements posed. However, the current impossibility to practically exploit the results of the research in software process technology is balanced by the existence (often ignored) of new, powerful, albeit less generic, mechanisms for integration that are the results of the continuous evolution of personal computing environments. A considerable de-facto standardisation has here taken place and the success of effective proposals for integration is encouraged by a wide community of users gradually educated to electronic desk: document processing (but without a process!), which is nowadays the most important activity performed on personal computer systems.

The idea of communication-based software buses started in the software engineering environments area and has been applied for service-oriented integration in different contexts (ESF, Field, SoftBench,...), and standardisation efforts have been undertaken to define standard services to be exported by different classes of software engineering tools (e.g. by Case Communiqué and Case Interoperability Alliance). Interestingly, the PC mass market has moved in a similar direction with the definition of DDE (Dynamic Data Exchange) by Microsoft and Apple Events by Apple: due to the fertile environment, these mechanisms are now widely accepted, exploited by application builders to offer new functions and available to end user programming via simple script languages.

Our view on the needs for integration from the perspective of process management is that of coarse-grained integration. From this perspective there is a need to provide different users with role-oriented pictures of information: a designer needs to provide design reviewers with