Effect of Product Lines on Current Process Technology (Session Summary)

Karen E. Huff
GTE Laboratories Incorporated
Waltham, MA

Abstract

In this discussion of the implications of product lines on current process technology, the key characteristic of product lines was identified as the development and reuse of assets of various kinds. Two primary themes emerged: 1) mechanisms for representing generic processes or families of processes, and 2) the implications of viewing process technology very broadly as any technology that helps or enables the enactment of a process. Addressing the requirements on process technology for product lines—whether those requirements are unique or not—offered a way to discuss potential advances in the field.

Product Lines

The central theme of product lines is the development and reuse of assets of various kinds, including requirements, components, architecture, generators, and processes [Balzer]. These assets arise from analysis and identification of the commonalities within the product line. Reuse of assets leads to ecomonies in both cost and time-to-market for product line development as compared with independently developed products.

Product line definitions necessarily draw boundaries. What is left out is often the source of future problems, as pressure mounts over time to widen the coverage and reduce bottlenecks and irritants. The processes for product lines will be under the same pressure [Lehman]. Is there an opportunity to get leverage over this kind of evolution?

It was also noted that other aspects of product line development than processes have yet to be specialized. For example, project management is not explicitly specialized for the case of product lines, although specialized knowledge may be used implicitly by managers experienced in a particular product line. There are some obvious candidates for specialization of management, including estimation and progress tracking [Boehm].

Mechanisms for Generic Processes

One view is that a *generic* process corresponds to a product line, and that the generic process represents a class of processes for building a class of products [Balzer]. Each product in the line is built by a process *type* that belongs to this process *class*. A similar view is that a product line process represents a *family* of processes, just as a product line represents a family of products. The variations in the processes within a family reflect the variations in the products [Sutton and Osterweil]. (While the concept of a process family is more commonly based on lifecycle notions, this was not pursued as being outside the scope of product lines.)

Factorization of commonalities and variability is key to the notion of a generic process. The more there is separation of concerns, the more reusability is achieved. How can process aspects be orthogonalized? What role can abstraction layers play? What process mechanisms will provide greater coverage for a perceived space of processes?

Parameterization is a basic, but limited, technique for capturing process variations. The use of dynamic or delayed bindings adds more power. Decisions can be made at project instantiation, for example when the tools, methods, and other implementation details are set. Decisions can also be furthered delayed to process execution time, and made responsive to the then-current goals and constraints. These delayed bindings are termed stratification and primitivation [Perry].

Another choice is to extend parameterization by placing a full complement of constraints in a (generic) process, then arming/disarming the constraints as needed [Osterweil]. For example, preconditions could be ignored in certain cases. This approach could accomodate both proactive and reactive control in a process family. Simple constraint enforcement is another way of achieving variability.

Many process formalisms have one type of relationship between process fragments, but there are many such relationships and constraints that need to be identified and represented. One approach is to have a process schema, and to provide operations on that schema to change the organization of the fragments [Derniame]. This kind of reflexivity can also address the fact that product lines evolve over time, and their processes will do the same. When customization and tailoring are present, there is the problem of maintaining the relationships between a generic process and its instantiations, as well as the relationships between the instantiations themselves.

Another approach to product line processes is to deploy cooperating processes that can work together to achieve a variety of goals [Heineman]. Autonomous and distributed processes can interoperate—the challenge is to modularize the work and place responsibility properly so that control and coordination is not overly difficult. However, total control may not be necessary—viewing the whole as a single huge process may be detrimental. In any case, guidelines would be useful as to how, and how much, distribution can be managed.

The alternative to generating multiple processes from a generic process was covered in discussions of work on generalizing multiple processes to create a generic process [Madhavji]. Models are organized by views (dataflow, control flow), and generalized along the same views. The generalizer identifies commonalities, but this may not account for all the relationships between the subject processes.

Additional discussion was centered on how to make the characteristics of the products evident in the processes to be carried out, and how to reflect product line constraints in the specification of the process [Boehm]. Examples included representing process objectives (even simple ones like 1 system or many systems) and process interdependencies (different markets, same time-to-market, etc.) One goal would be to make inferences about process aspects such as reliability.

One other technology/tool impact follows from the fact that processes (and perhaps more especially product line processes) are complex systems. In such systems, the whole cannot be understood by analyzing the parts [Lehman]. An analysis of (static) structure does not give insight into the behavior as a whole. System models must encompass the parts, their interconnections, and the dynamic behavior. In product lines, it may actually be cost effective to engage in process modeling (and simulation); an example offered was modeling the effects of software inspections, to understand when they should be used. Since the leverage is greater in product lines, the benefits of modeling may well exceed the costs [Boehm].

All the mechanisms discussed above have tooling implications. The success of reuse in particular, whether in product lines or not, depends upon the ability of tools to facilitate all the mechanisms discussed here [Perry].

A Broader View of Process Technology

In this part of the session, the scope of the discussion was broadened beyond process modeling languages and the tools that support them. The leverage in product lines, which comes from explicitly identifying product commonalities, takes two forms: eliminating work and automating work. Reuse of a component or architecture eliminates work; building and using a generator automates work.

Any mechanism that helps to fold product line properties into an automated tool can be thought of as process technology [Balzer]. An example of such a tool is the CM tool for product lines [Schaefer]. The tool provides for the incorporation of product line information in the form of constraints, dependencies, responsibilities, propagation of changes, and so on.

The challenge is to understand: what is the space for tool support/automation, and what are the general mechanisms for incorporating product line properties. The more tools that incorporate such product line knowlege, the better. On the other hand, tooling up for product lines can be expensive, as the Shuttle software experience has shown.

Even more generally, incorporating into tools knowledge about any product, individually or as part of a product line, offers significant leverage. Incorporation of such knowledge also impacts infrastructure components that percolate into process technology, such as object management. For example, software product representation can be “server-ized” to support requests for product information and state. However, this “serverization” may need to be specialized based on characteristics of specific products, for example to support optimization of queries or constraint enforcement [Sutton].

Any technology that helps or enables the enactment of a process is a kind of process technology [Balzer]. Examples include agenda managers, workflow managers, animators, and help systems. Other examples are not traditionally thought of as process technology; consider generators, or model management tools (that detect inconsistency and incompleteness between separate views of a system).

Any use of a tool is an example of automating a part of a process. Enactment support is not an all-or-nothing proposition. It is possible to support process enactment with a whole variety of tools, and never get to “complete” automation. Conversely, “complete” automation may be a more achievable goal in the case of product lines than in the general case [Bandinelli]. However, the problem of interoperation of tools is significant [S. Wolf]; on the one hand, integrating independent tools is difficult; on the other hand, an integrated toolset without a process is useless.

Conclusion

Throughout the discussion, the question lingered as to whether there are process technology implications of product lines. There was the view that new mechanisms (in process modeling languages) were not required, although product lines stress certain existing ones in ways that independent product development does not [Cugola]. There was also a view that the new kinds of mechanisms and tool support discussed are needed *anyway*, independent of product lines [Tully]. In the end, addressing the requirements on process technology for product lines—whether those requirements are unique or not—offered a way to discuss potential advances in the field of process technology.