

Towards a Reference Framework for Process Concepts

Reidar Conradi*, Christer Fernström†
Alfonso Fuggetta‡, Robert Snowdon§

Presented at 2nd European Workshop on Software Process Technology,
(EWSPT'92) 7–8 Sept. 1992, Trondheim, Norway.
Jean-Claude Derniame (ed.), Springer *LNCS 635*, p. 3–17.
EPOS TR 158

Abstract

This paper discusses the importance of process support for business activities. A reference framework for process concepts and technology support is sought. The general requirements and properties of the process domain are first discussed. Then, four process sub-models are presented to describe activities, products, tools and organisations, respectively. Five process model phases are also introduced, as well as meta-processes and related human roles to handle process models and their transformations. The process concepts are applied to a bank example.

Keywords: (software) process, process modeling, process improvement, meta-process, roles.

1 Introduction

Business¹ activities carried out in our society are becoming more and more complex and difficult to manage. The reasons for this are rooted in the rapid evolution of social and economic activities during the last decades, and the diffusion

*Norwegian Institute of Technology–NTH, Division of Computer Systems and Telematics, N-7034 Trondheim (Norway). Tel.: +47-7-593444, Fax: +47-7-594466, E-Mail: conradi@idt.unit.no.

†CAP Gemini Innovation, 7 Chemin du Vieux Chêne, F-38240 Meylan (France). Tel.: +33-76-764720, Fax: +33-76-764748, E-Mail: christer@capsogeti.fr.

‡CEFRIEL - Politecnico di Milano, Via Emanuelli 15, 20126 Milano (Italy). Tel.: +39-2-66100083, Fax: +39-2-66100448, E-Mail: alfonso@mailier.cefriel.it.

§Univ. of Manchester, Dept. of Computer Science, Oxford Road, Manchester M13 9PL (United Kingdom). Tel.: +44-61-2756176, Fax: +44-61-2756236, E-Mail: bobs@cs.man.ac.uk.

¹By business we mean any professional activity or enterprise, covering both technical and administrative aspects. There is no bias to the commercial side in our use of the term.

of information technology in many new areas. These two factors are strongly interrelated: advances in the technology enable creation of new products, services and activities, or modification of old ones. These will again change our life, and produce new needs, feedbacks, and requirements to technology providers.

Business activities with many interacting humans put hard demands on management. Examples of such activities can be found in a bank, where hundreds of employees cooperate to produce the bank's product, i.e. to provide its customers with flexible and efficient financial services. In most of these activities, humans rely on various computerised tools, which support (part of) their work. Typically, these tools automate a specific task, or manage large quantities of business-related information.

The insight that the quality of products and services intrinsically depends on the quality of the associated process has steered work on quality towards improvement of processes. Especially in these partly computerized businesses, the opportunities to support the process through software has led to increasing efforts in the industrial and academic communities to provide techniques, methodologies and tools to assist the process part of businesses. In particular, researchers are concentrating on the following topics:

- How can we describe a business, to understand it and to facilitate communication and teaching of the business's rules and procedures?
- How can we better manage, control and thus improve the business?
- How can we have humans and computerised tools cooperating in a coordinated and controlled way to support the business?

Software production represents an important business activity, where humans and computerised tools interact to deliver products to an end-user. We denote the activities, rules, procedures, techniques, and tools used within this business by *software process*. This term has recently gained popularity among researchers and practitioners.

Around *software process management*, several recent initiatives have been launched:

- *Industrial* initiatives to improve software processes have been adopted within software development organizations. Cooperations between academia and industry have also been established. A well-know initiative in this area is DARPA's Software Engineering Institute, created at Carnegie Mellon University in Pittsburgh in 1983.
- A new *research* area have grown up, and many scientific events have been established to facilitate the exchange and discussion of results, problems and early experiences gained during these years. See [FCA91] [IEE91a] [IEE91b].
- Outside the area of software production, similar efforts have emerged within the area of office automation, usually under the common heading of *workflow support* [HL91].

Efforts have so far been restricted within their specific areas (software engineering, office automation). Two aspects therefore deserve further attention:

- Software and office processes are not the only production processes. We should aim at reusing the experience and research from other disciplines, such as Information Systems, or CIM systems for VLSI design and production.
- The recent intensive work on software processes has led to terminology and definitions for software processes which generally are confusing. Many different terms are used for the same or similar concepts, and vice versa.

It therefore seems worthwhile to establish a *reference framework* of process concepts. This can be used within the software process community as a common set of terms and concepts. It can also be used to exploit commonalities with related application disciplines or research areas. Other clarification efforts of PM concepts have been done by Mark Dowson et al. [DNR91] and by Watts Humphrey et al. at SEI [Hum88].

This paper is structured as follows: Section 2 describes in more details the systems we want to support. Sections 3 and 4 present the basic terminology and concepts we are introducing. Section 5 applies these concepts to a bank example.

2 Human oriented systems

We have previously used the terms “complex business activities” and “(software) processes” to denote a system where humans and computerised tools interact to achieve a common goal. In such systems, humans play a crucial and active role. Furthermore, proper management and coordination of the interactions among humans and between humans and computerised tools are critical and complex activities. We call such systems *human oriented systems*. Their requirements and properties can be summarised as follows:

- It is difficult to formalise human behaviour, since it is intrinsically non-deterministic. Thus, human activities are difficult to coordinate and control.
- Different humans or components of the systems may have conflicting goals.
- The (human) understanding of the actual application is often weak.
- Humans are intrinsically goal-driven and can rarely be forced to use a particular procedure or computerised tool, when other tools or manual procedures can be used to achieve the same goal.
- Humans are creative and continuously seek to enhance and optimize the way they carry out activities.

Typical examples of human oriented systems are a bank (as mentioned before), an engineering center, or a software factory. In the last case, the computerised tools are for example editors, compilers, CASE tools, and debuggers. The humans are the programmers and designers involved in software production.

To manage a human oriented system, we must effectively *model* and formally *describe* the complex relationships between humans and tools, inside and between products, and among humans. Within the software engineering community, this has led to several languages and notations to facilitate the above-mentioned requirements and properties.

To better understand how process-related technology can improve processes, we will present a set of basic concepts, which constitutes our initial reference framework of process concepts.

3 Basic concepts

A *business* is carried out by humans who use tools and follow specific rules and plans. Humans, procedures (manual or automatic), tools (i.e. external and automatic procedures), rules, policies (often high-level rules), and activities (tasks) are all components of the *process* to operate the business.

We distinguish between two main kinds of support technologies:

- **Application Support Technology** used within the process. This includes domain-specific and computerised tools used by human actors within the process.
- **Process Support Technology**, i.e. methods and guidelines, languages and formalisms, tools etc. to support the process itself (and the meta-process discussed below, see section 4). This includes tools to support and transform process models, and to execute the corresponding processes.

A *process model* and associated formalisms have at least five sub-models:

- An executable **activity model** (or task model), to express both simple and aggregate activities. The available activity *formalisms* fall into four main categories:
 - Descriptive or rule/trigger-based (MARVEL [KFP88] and ADELE2 [BEM91]).
 - Network-based (MELMAC [DG90], OPIUM [HFLB90], Extended Petri Nets [BFG91]),
 - Imperative or programmatic, usually interpreted (APPL/A [HSO90], IPSE 2.5 [War90]).
 - Hybrids (EPOS [C+90]).
- A **product model** to express (passive) data, being manipulated by activities. An object-oriented ER model is often used [BGMT88].

Note, that the product is evolved by activities, often driven by the product structure. The activities are themselves evolving and persistent artifacts, i.e. “products” being operated upon by meta-processes. Indeed, the model is itself a manipulatable product [BPR91].

- A **tool model** to describe tools and their architecture. This can partly be expressed by the activity model, embedding a tool as an activity “envelope” [BCN90].
- An **organisational model** to structure and control activities, and their executable resources. Common artifacts are humans (e.g. modeled by roles), machine resources, overall project constraints, team coordination, and work delegation.
- A **user model** to describe how various process actors benefit from the assistance provided through process support technology. A useful model is the *information logistics*, introduced in [FO91], to describe the availability and flow of information between process actors.

Thus, a process model may cover an entire application, not only its “active parts”.

A process model and its formalism must consider the requirements and properties of the real process (Sec. 2). Some relevant characteristics are:

- *Modularisation*, e.g. by hierarchical submodels or reusable model fragments.
- *Abstraction*, and possibilities for gradual *Specialisation* (cf. PMi phases below).
- *Evolution* and *Customisation*:
A useful model must evolve to reflect changes in the business and to absorb improvements, and it must be customisable [BL79].
- *Formalisation*, and thus support for automated analysis and assessment.
- *Monitoring* and *feedback* mechanisms, to assist the above assessment and evolution.
- *Clarity* and *Orthogonality*, so that a small set of well-defined concepts can be freely combined.
- *Understandability* by humans, e.g. through an external graphical notation.

Initially, only an informal process model, or its requirement or high-level design, may exist. This process requirement/design can be developed into a more formal, albeit *generic process model*. The generic and abstract model can gradually be refined and customised into a more *specific process model*. This can be instantiated into a concrete process, ready for execution (“enaction”).

We thus have five stages through which process models go:

1. **PM1. Process Model Requirements & Overall Design:** a rather informal process model, resulting from elicitation and formulation of external process models. It will also contain feedbacks received by an executing process, and the associated requirements on how to evolve the process.
2. **PM2. Generic Process Model or Schema:** a high-level and possibly incomplete description of the general sequence of activities of the process, and of the products involved. It can be used in many similar projects and organisations, sharing common properties and characteristics.

Such a Schema suggests a very high-level process architecture, according to general principles and requirements. Within software production these schemas are called software life-cycles, such as the Waterfall [Roy70] and the Spiral models [Boe88].

3. **PM3. Specific / Customised Process Model:** a more detailed and rigorous description of each type of task in a process, possibly automatically derived from the generic Schema in PM2. Requirements from management will be considered to refine all the sub-models above, e.g. by subtyping or by binding project policies, domain-specific tools and roles etc. We can mention a specific process model for the delivery of software modules to the integration test group (in a software factory), or for the emission of travel checks (in a bank).

At both PM2 and PM3 we need tools for analysis, assessment, simulation etc. of process models.

4. **PM4. Instantiated Process Model:** A specific process model must be further instantiated to accommodate the constraints and information of a given project. An instantiated process model is thus an executable process description, which links instantiated activities (i.e. task types which have been instantiated to tasks) with concrete products and project resources, and which obeys scheduling deadlines.

Once the process model has been instantiated for a corresponding project, the model can be used to support, control, and evolve the project and its activities.

5. **PM5. Executing/Active Process Model:** the manual or automatic execution (rule-based or program-driven) of the instantiated model, leading to changes in all the above PM1–PM4 phases through feedbacks. The execution of a model is usually referred to as model *enactment*.

The borderlines between these model phases can be hard to draw, and may vanish in systems with late / dynamic binding.

Feiler and Huphrey have introduced a somewhat different taxonomy [FH92]. They speak of a Process Architecture, being the conceptual framework to express process models. Our process model requirements & overall design corresponds to their process design, our specific process model corresponds to their

process definition, and our scheduling constraints (present in inputs to an instantiated model) correspond to their process plan. Further, we speak mostly about process *models*, making the distinction between the real-world processes and their descriptions (which we call “models”), while Feiler and Humphrey mainly refer to processes at different stages of abstraction, refinement and binding. Lastly, we put more emphasis on the meta-process and human roles.

The development and evolution of a process model is complex and critical. This activity is called a *(meta-)process*, and it can be described as any other process. We thus make the following distinction:

Process: Set of partly ordered activities – and related products, tools and organisations – to produce the requested outputs.

Meta-process: Set of (meta-)activities to model, analyse, and support a process.

This means that process models and processes are the objects upon which meta-processes operate.

In the following, we will use simplified SADT diagrams [Ros77] as a process formalism for processes and meta-processes.

4 The meta-process

As mentioned, every process needs to be made explicit, generalised and then specialised/customised, instantiated, executed, and then improved and modified. We therefore need meta-process and its meta-subprocesses to transform a PM_i into its successor, and to continuously adapt and evolve them. The relationship between a process and its meta-process can be described as in Figure 1:

- Processes and meta-processes are operated by humans who can play different roles, e.g. Process Owner, Process Manager etc. See later in this section.
- One of the outputs of a process is the feedback from its operating people on the procedures and tools used. This feedback drives the meta-process activity, thus impacting the changes of the process itself.
- The meta-process produces and modifies the process model of the phases PM1–PM5. To do this, the meta-process uses a description (a model) of the relevant Application Support Technology, and relies on *meta-process tools*. Tools which are part of Process Support Technology, thus fall into either of two categories: process tools, which support the Process Actors (in PM5) and meta-process tools, which support the modeling activity.
- One of the outputs of the meta-process is comments and requirements back to the producer of the Application Support Technology on the efficiency and effectiveness of the domain-specific tools.

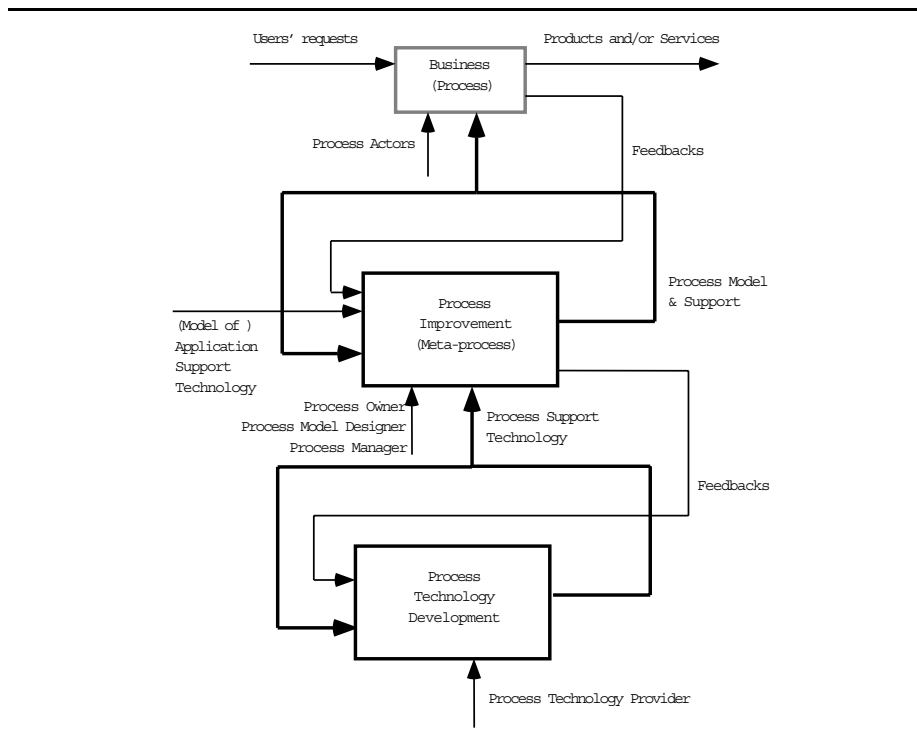


Figure 1: Process and meta-process.

The meta-process of Figure 1 is further refined in Figure 2, showing two important meta-activities:

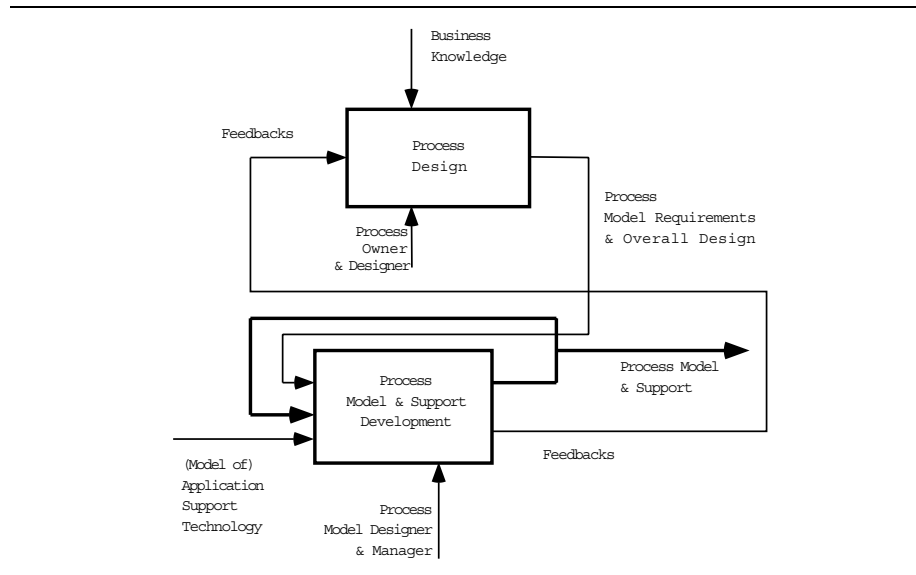


Figure 2: Process Improvement (the meta-process).

- **Development of a Process Model Requirements & Overall Design (PM1).**
- **Development of Process Model & Support Environment:** i.e. to produce PM2–PM5 and to carry out Process Technology Development. The inputs are PM1 and given domain- or project-specific requirements. The PM2–PM5 production can be refined as in Figure 3.

There are two important considerations to the previous discussion:

Incremental definition: It must be possible to generate, instantiate and execute a process model incrementally. Typically, parts of the process and its model are left unspecified until execution, by *late* (sometimes called *dynamic*) *binding* as indicated above. This gives flexibility and helps to avoid strait-jacket effects.

Feedback loops: Process models must be changeable also when the process is active, i.e. after the model is instantiated and executed. This is because the business which the model refers to, as well as the technology supporting it, can and will change. These changes must be properly and promptly reacted upon to implement an effective and efficient business management.

As mentioned in the beginning of this section, the meta-activities to produce PM1–PM5 are operated by humans who can play different roles:

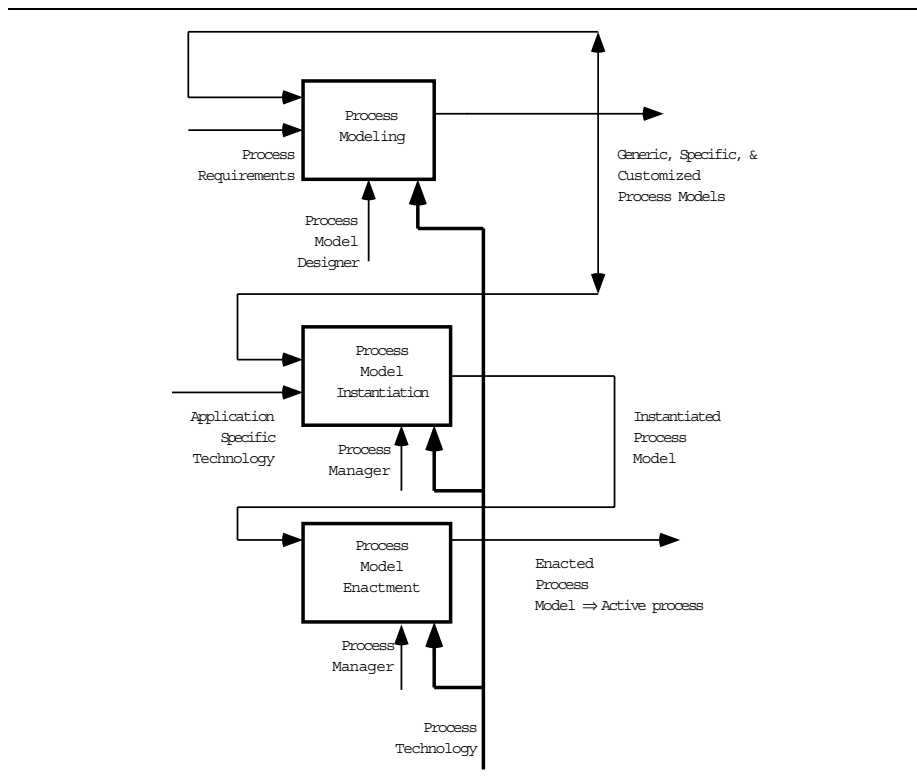


Figure 3: Process Model & Support Development.

Process Owner: gives inputs to PM1. He/she is responsible for managing the process to be modeled. He provides the process description and evaluates process performance, thus producing additional feedbacks and requirements for process improvement.

Process Model Designer: responsible for PM1, PM2 and partly for PM3. His/her goal is to understand the nature of the process, and to elicitate or create the initial, partly informal PM1. This PM1 requirements & overall design must later be converted into a formal and generic PM2, e.g. by using a library of existing and reusable models. The specialisation of a Process Schema (output of PM2) to a particular project or situation may or may not be under the responsibility of the Process Model Designer.

Process Manager: responsible partly for PM3 and for PM4. His/her goal is to provide information to specialise/customise and instantiate the process model, and to activate and monitor the execution of this instantiated model.

Process Actor: responsible for PM5 execution and for feedbacks to the different PMi's. He/she is a human who operates within the business process. He uses the set of domain/application-specific rules, tools and technologies, all of which are controlled, assisted or enforced by the process support.

Process Technology Provider: supports all PMi's. He/she is the agent of the Process Technology Development activity.

5 An example: a bank and its environment

This section will apply the definitions and concepts from the previous section to a business example, consisting of the different processes involving IT (Information Technology) operating within a bank. The goal is to structure the bank processes according to the *process/metaprocess* introduced before. The different process sub-models will not be elaborated.

We will consider processes and meta-processes of two different organisational entities (see Figure 4: the bank offices (and the process involving the bank's customers), and its EDP department (in charge of developing the software applications used in the bank offices)).

The Bank Process and Meta-process

The business of a bank is to deliver financial services to its customers. The process in place to achieve this business goal is carried out in bank offices, where the Process Actors are bank employees, and is referred to as the *Bank Process*.

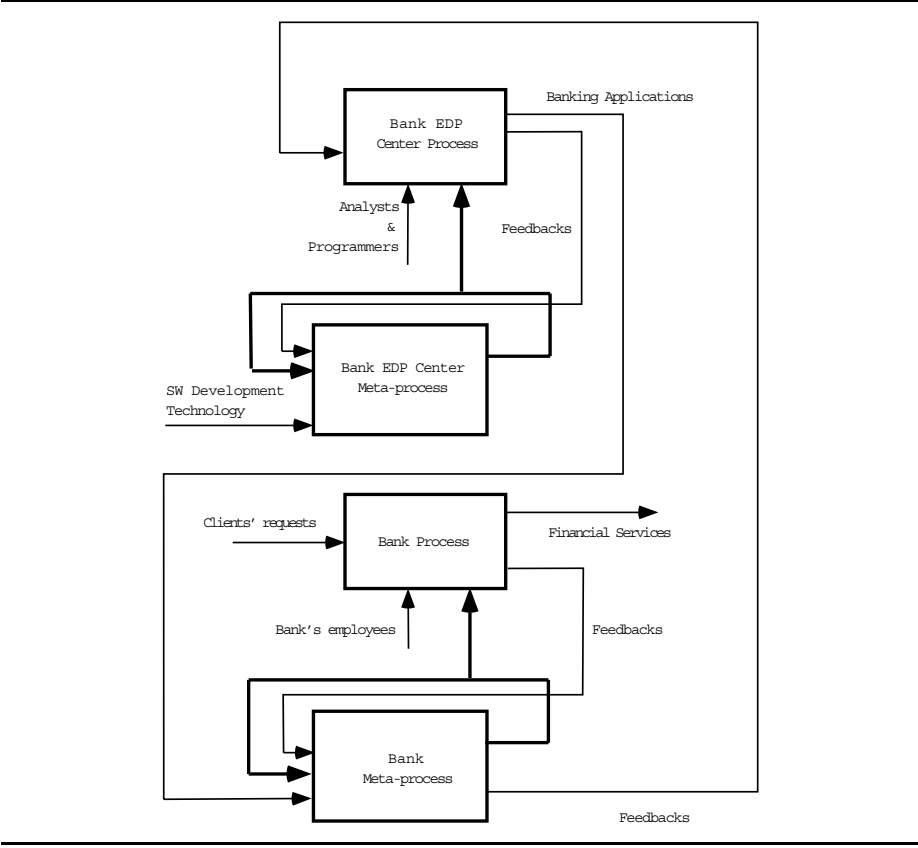


Figure 4: Bank operating environment.

In carrying out their work, the Actors use several technologies (computers, networks, etc.) within the framework of procedures and rules that apply in the bank. The Bank Process is supported and improved by means of the process support developed by the *Bank Meta-process*. This is the joint activity of the Process Owner, e.g. the bank's steering group, and the Process Designer, for example the bank's methods department.

The Bank EDP Department Process and Meta-process

(Part of) the applications used by the bank's employees are developed by the bank's EDP department. In the case of the EDP department, the actors are the analysts and programmers in charge of producing these software applications. The EDP department also involves a meta-process, which is operated by the group of persons in charge of improving the EDP department's procedures to develop software. This meta-process uses as input the application tools (compilers, CASE tools, debuggers, ...) denoted with Software Development Technology in the figure, that will be assembled to produce the EDP department process model and support. The EDP department process is an example of what we called "software processes".

6 Conclusions

The paper has presented a reference framework for (software) process concepts. Specially the meta-process apparatus should be followed up with tool support, i.e. CASE tools for life-cycle process management. However, many process aspects are not covered, e.g. monitoring, which SEI have treated in more detail.

The framework will be evaluated in an upstarting ESPRIT Basic Research Action, PROMOTER, where it will be applied to different scenarios and process management systems.

Acknowledgement

Thanks go to our colleagues, whose research and comments have made this paper possible.

References

- [BCN90] Hugh R. Beyer, Kathy Chapman, and Chris Nolan. The ATIS reference model. Technical Report ZK02-3N30, Digital Equipment Corp., 110 Spirit Brook Rd., Nashua, NH 03062, April 1990.
- [BGMT88] Gerard Boudier, Ferdinando Gallo, Regis Minot, and Ian Thomas. An overview of PCTE and PCTE+. In *Proceedings of the ACM SIG-*

SOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments, Boston, Massachusetts, November 28-30 1988.

- [BL79] L.A. Belady and M. M. Lehman. Characteristics of large systems. In Peter Wegner, editor, *Research directions in Software Technology*. MIT Press, 1979.
- [BEM91] N. Belkhatir and J. Estublier and W. L. Melo. Adele2: A Support to Large Software Development Process. In Proc. 1st Conference on Software Process (ICSP1), Redondo Beach, CA, October 1991, pages 159–170.
- [Boe88] Barry W. Boehm. A spiral model of software development and enhancement. *IEEE Computer*, pages 61–72, May 1988.
- [BPR91] R.F. Bruynooghe, J.M. Parker, and J.S. Rowles. PSS: A system for process enactment. In *Proceedings of the First International Conference on the Software Process, Los Angeles, USA*, October 1991.
- [C⁺90] Reidar Conradi et al. Design of the Kernel EPOS Software Engineering Environment, 17 pages. In *Proceedings of the First International Conference on System Development Environments and Factories*. Pitman Publishing, 1990.
- [DG90] Wolfgang Deiters and Volker Gruhn. Managing Software Processes in the Environment MELMAC. In *Proc. of the 4th ACM SIGSOFT Symposium on Software Development Environments, Irvine, California. In ACM SIGPLAN Notices, Dec. 1990*, pages 193–205, December 1990.
- [DNR91] Mark Dowson, Brian Nejme, and William Riddle. Fundamental Software Process Concepts. In *[FCA91]*, pages 15–37, 1991.
- [FH92] Peter H. Feiler and Watts Humphrey. Software Process Development and Enactment: Concepts and Definitions, January 1992, 12 pages. (Second version).
- [FCA91] Alfonso Fuggetta, Reidar Conradi, and Vincenzo Ambriola, editors. *Proceedings of the First European Workshop on Process Modeling (EWPM'91)*, CEFRIEL, Milano, Italy, 30–31 May 1991, 1991. Italian Society of Computer Science (AICA) Press.
- [BFG91] Sergio Bandinelli, Alfonso Fuggetta, and Carlo Ghezzi. Software Process as Real-time Systems: A Case Study Using High Level Petri Nets. In *[FCA91]*, 1991.
- [FO91] Christer Fernström and Lennart Ohlsson. Integration Needs in Process Enacted Environments. In *Proceedings of the First International Conference on the Software Process, Los Angeles, USA*, October 1991.

- [HFLB90] Laurence Hubert, Frederic Fournier, and Maryse Bourdon Le-Brasseur. Eureka Software Factory: OPIUM, an environment for software process modeling integrated with a project management tool. In *Proceedings of the 6th International Software Process Workshop, Hakodate, Japan*, October 1990.
- [HL91] Keith Hales and Mandy Lavery. *Workflow Management Software: the Business Opportunity*. OVUM Ltd., 1991.
- [HSO90] Dennis Heimbigner, Stanley M. Sutton, and Leon Osterweil. Managing Change in Process-centered Environments. In *Proceedings of 4th ACM/SIGSOFT Symposium on Software Development Environments*, December 1990. In ACM SIGPLAN Notices.
- [Hum88] Watts S. Humphrey. Characterizing the Software Process: A Maturity Framework. *IEEE Software*, pages 73–79, March 1988.
- [IEE91a] IEEE Computer Society. *Proceedings of the First International Conference on Software Process*, 1991.
- [IEE91b] IEEE Computer Society. *Proceedings of the Seventh International Workshop on Software Process*, 1991.
- [KFP88] Gail E. Kaiser, Peter H. Feiler, and Steven S. Popovich. Intelligent Assistance for Software Development and Maintenance. *IEEE Software*, pages 40–49, May 1988.
- [Ros77] D. T. Ross and K. E. Schuman. Structured Analysis for Requirements Definition. *IEEE Trans. on Software Engineering*, SE-3(1), pages 16–34. January 1977.
- [Roy70] W. W. Royce. Managing the Development of Large Software Systems: Concept and Techniques. In *Proceedings of WesCon*, August 1970.
- [War90] Brian Warboys. The IPSE 2.5 Project: Process Modeling as the basis for a Support Environment. In *Proceedings of the First International Conference on System Development Environments and Factories*. Pitman Publishing, 1990.