A PROCESS META-MODEL IN A GRADUAL SOFTWARE PROCESS IMPLEMENTATION ENVIRONMENT
Process Meta-Model for a Software Process Definition and Improvement

Sandro Ronaldo Bezerra Oliveira

Centro de Ciências Exatas e Tecnologia – Universidade da Amazônia (UNAMA)
Av. Alcindo Cacela, 287, 66060-902 – Belém-PA-Brasil

Alexandre Marcos Lins de Vasconcelos, José Francisco Pereira, Igor Cavalcanti Ramos

Centro de Informática – Universidade Federal de Pernambuco (UFPE)
Caixa Postal 7851 – 50732-970 – Recife-PE-Brasil

Keywords: Software Process Definition and Improvement, Quality Model/Norm, Software Development Environment.

Abstract: PSEE - Process-centered Software Engineering Environment - PSEE has one of its intentions to provide that phases of the software process life cycle (definition, simulation, enacting and evaluation) can be automatized. This work presents the structure and the automation of a software process meta-model capable to group terminologies of processes based on quality models/norms and to help in the implementation and refinement these types of processes. This implementation must be made from characteristics and properties that define an organization or a specific domain of software project. The meta-model services were automatized trough a tool and the description of them can be found in this paper.

1 INTRODUCTION

The development of a quality software, with raised productivity, inside the established stated period and not needing more resources that those placed, became a challenge for the organizations (Machado, 2000). The growth of the size and the complexity of the software products make that these ones assume critical roles in the organization businesses.

The experience of the software industry samples that the main reason for failure of the projects is in the lack of a software process disciplined, or either, in the lack of a mechanism that qualifies the management and control the quality of product. Following the same trend, it already is widely accepted that the quality of a software product is determined by quality of process used during its development and maintenance strongly.

Knowing the processes means to know how the products are planned and produced. It fits to stand out that, from the process definition, it is possible to define measurements and to collect the execution data. It gives to visibility to managers and technician about the course of projects, making possible action to control the variations of the project and the processes for it used. In this direction, the software process definition is a basic requirement for the attainment of quality software products.

However, the software process definition is not a simple activity; it demands experience and it involves the knowledge of many aspects the software engineering. The difficulty in defining processes meets in the absence of a software process possible of being applied generically. The processes vary because the types of systems, the application domain, the teams, the organizations and the business restrictions are different, such as, schedule, cost, quality and trustworthiness (Machado, 2000).

It already was argued so much about the properties of this type of technology, however, it perceives during the process execution from these development environments that its implementation is always not the reality of organization and projects characteristics for this one. That's why the responsible users for the process definition do not make use of a guide contend its real needs of

execution and these ones indicate the best practices to be instantiated from a standard process. Some of these environments are PROSOFT (Reis, 2003), Adele-Tempo (Belkhatir, 1994), ProcessWeaver (Christie, 1997), Estação TABA (Rocha, 2001).

In this context Oliveira considered, in (Oliveira, 2005), the definition of an environment for software process implementation, called *ImPProS* (Gradual Software Process Implementation Environment), that goals to make possible: the specification of processes in accordance with the specific project domain and the organization characteristics; the instantiation of the software process in accordance with the properties of each project; its simulation from the configuration parameters (stated period, pressures, cost, resources, etc.); an execution (automation) similar the organizational process; and an evaluation from the collection of metric this execution.

This paper describes the specification and the automation of a meta-model capable to group all the components of a software process (processes, activities, tasks, resources, procedures, etc.), ally to the standards adopted for quality norms/models of software (product and process), in order to contemplate a repository with unified terminologies from mappings and compositions, and a software process definition and implementation taking referential the quality proposal by systems development industry.

Beyond this introductory section, the paper presents five sections. Section 2 approaches the detailing of the characteristics that compose the software process implementation environment considered by (Oliveira, 2005). In section 3 it is presented the standard process structure adopted for this environment. In section 4, we find the rules that had originated the specification and the automation of software processes meta-model. Finally, section 5 presents the final consideration.

2 *ImPProS*: A SOFTWARE PROCESS IMPLEMENTATION ENVIRONMENT

The *ImPProS* is a project which is being performed at the Center of Informatic of UFPE – Federal University of Pernambuco with the partnership of UNAMA - University of Amazônia, an financed by CNPq - National Agency for Scientific and Technological Development. The objective of *ImPProS* is the creation of an environment to support the implementation of a software process in an organization in a gradual way. The "gradual" term means that the implementation of the process is improved with the experiences learned in its definition, simulation, execution and evaluation.

The goals of *ImPProS* was adapted from the structure that compose the software meta-process described in (Reis, 2003), the characteristic proposals for the implementation of a software process (Balduino, 2002) and the life cycle for continuous process improvement defined by IDEAL Model (Mcfeeley, 1996). Thus, it is composed of a cooperative environment, formed by nine main tools:

- **ProDefiner:** it provides the definition of software process from the analysis of specific characteristics;
- **ProSimulator:** it makes possible the simulation of a software process instantiated from an execution plan of the process and thus allows to foresee problems;
- **ProEnacter:** it allows the automated execution and monitoring of a software process by a project team;
- **ProEvaluator:** it provides the evaluation of software process execution from the analyses of qualitative and quantitative criteria;
- **ProImprove:** it makes possible the systematic execution of activities regarding the software process improvement, based on the IDEAL model;
- **ProAnalyser:** it allows the analyses and decision taking concerning the evaluation items which compose the software process;
- **ProReuse:** it provides the software process reuse from the definition of project scope and its adaptation to the use context;
- **ProKnowledge:** it makes possible the collection, analyses and use of knowledge learned during the execution of a software process;
- **ProConverter:** it provides the conversion of software process components defined for a process (activities, artefacts, resources, etc.) from the structures of quality norms/models.

3 STANDARD SOFTWARE PROCESS STRUCTURE OF *ImPProS*

Because the *ImPProS* has the one of its characteristics the software process definition

through a software process model and its diagrammatic representation, it has a general structure of software process composition the model based on the definitions of ontologies the software process defined by Falbo (Falbo, 1998).

Processes are collections of related activities that have place during the product development. Basically, a process consists of a structuralized set of activities and, therefore, all the involved infrastructure in the accomplishment of these (devices, procedures and resources). **Activities** are the tasks or works to be carried through. An activity requires resources and can consume or produce devices. For its accomplishment, an activity can adopt a procedure. An activity can be decomposed in other activities. Moreover, activities, in any level, can depend on the finishing of other activities, called pre-activities. The activity concept is present in all the software process models and is considered primitive that generate devices from entrance devices assisted by resources. Activities can correspond to different levels, either a task or a stage of development process.

Describing the stages of process and the activities to be carried through in each stage, it appears the concept of **Life Cycle Model** that structure activities and defines boarding of how organize a project in phases. The life cycle is initiated when the software is conceived until it enters in disuse, or either, it contains a set of development, operation and maintenance activities. Ally to this concept we have the **Combination**, which defines the way how a set of phases a life cycle model must be carried through and specifies the type of ordinance that the structure can be: sequential or iterative.

The **Work Products** are software products produced or consumed by activities during its accomplishment. They are examples of devices: revision manuals, workflow diagrams, object diagrams, code source, etc. A device can be decomposed in other devices (composition of devices). It is the entrance or product of an activity, being able to be devices of code, documents or software components.

The **Procedures** are behaviours established and commanded for the accomplishment of activities. Some procedures can partially be automatized by software tools. They are used to assist the accomplishment of the activities, being able to be directed to a specific type of activity, having to be adjusted to a development technology and paradigm. Ally to this concept, we have the **Activities Patterns** that a procedure must suggest for the execution of an activity. It represents a behavior which decompositions of an activity have in common.

People, the software tools, the equipment, or any other necessary infrastructures to the execution of an activity, can be called **Resource**. A human resources, specifically, plays a role in the execution of the process activities. They are necessary elements for the accomplishment of an activity, such as human agents, hardware equipment and software tools. They support or acts in the accomplishment of the activity, but they can not be considered "raw material" for the activity, or either, they only assist the process, but they are not incorporated the software product being considered resources for the activity.

We can still find the concepts of Development Paradigm, that are principles and concepts that guide the development (for example: structuralized and objects-oriented), and the Development Technology that represents the technology to be used in the software development (for instance the conventional technologies of data processing and systems based on knowledge). Finally, to limit and/or to restrict the execution of the activities defined in the process, it has the Restrictions, that specify the rules of software process definition.

4 SOFTWARE PROCESS META-MODEL OF *ImPProS*

In the software process definition of *ImPProS*, adapted from model defined by (Rocha, 2001), defined in Figure 1, initially meets the software process meta-model, made up of components and the relationships between them that are deriving the mapping of some software quality process norms and models (CMMI (Chrissis, 2003), SPICE – ISO/IEC 15504 (ISO, 1998), ISO/IEC 12207 (ISO, 2000), etc.).

The goal of this meta-model is to determine a unique terminology for the software process definition in *ImPProS*. It is great to stand out that the structure of meta-model was defined to contemplate all the software process components (process, activities, devices, etc.), but not to restrict its composition for some software process norms/models (common in the works found in specialized literature), or either, depending on norm/model to be used, the user can make the mapping of the same one using as base the ISO/IEC 12207 terminology and define its processes from the use of this new meta-model.
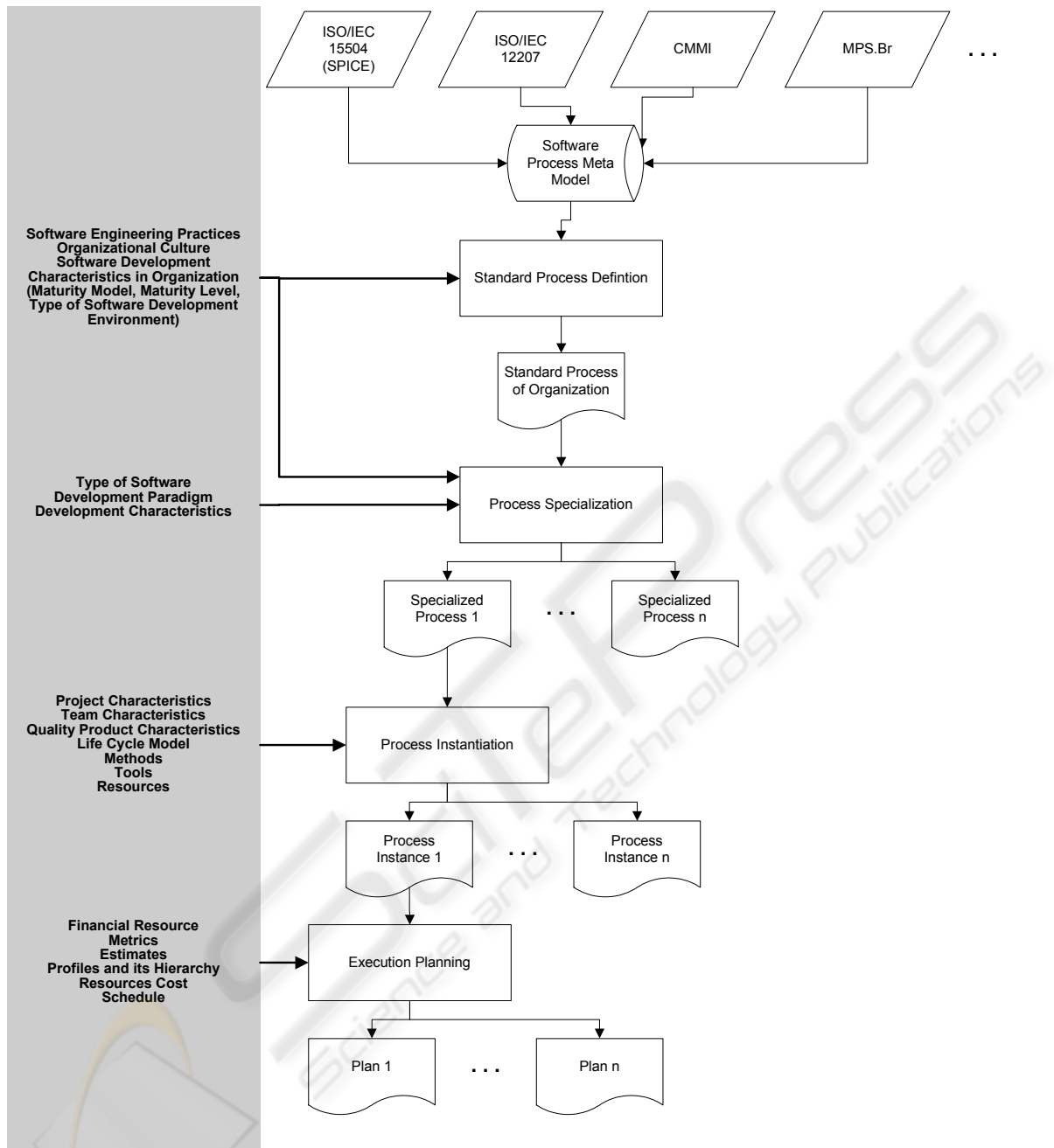
Figure 1: Software Process Definition Structure in *ImPProS* (Oliveira, 2005).

The definition of a standard process establishes a common structure to be used by organization in its software projects and constitutes the base for definition of all its processes. This way, a basic process is established that will serve as starting point for the posterior definition of the adequate software processes to the different characteristics of each project, allowing economy of time and effort in the definition of new processes.

In view that different types of software have distinct characteristics and require different development boarding, the standard software process of organization will have to be adapted (specialized) considering the characteristics related to the type of software (for example, information systems) and to the development paradigm used (for example, object-oriented). Thus, during the stage of standard process specialization, activities could be

added or be modified, in accordance with the context for which it is carrying through the specialization.

The instantiation for specific projects consists the adaptation of a specialized process to a project, considering its peculiarities. In this stage, the life cycle model, the methods and the tools that will be used in the project, the human resources and its responsibilities during the process and the consumed and generated devices (products) are defined. The ISO/IEC 9126 norm is used to identify the requirements of quality product. Such characteristics will influence the process from activities, methods and techniques. The activities of specialized process will have to be adapted to the life cycle model chosen for the project and new activities could be inserted in a determined life cycle. Thus, as result of this phase an instance of process is generated contends the components necessary for the software process in order to represent the specific project to be developed, taking care of all characteristics of this development.

The *ImPProS* considered three contribution points that, from analyses made in process definitions and what specialized literature considers as use, had perfected the specification of software process in the three definite levels and make possible that the definite components to the process could be simulated in order to foresee problems in the execution by project team:

- Inclusion a set of new organizational, software projects and products characteristics;
- Suggestion of software process components from definitions processes have done previously and knowledge learned during these definitions;
- Planning Level of Instantiated Process so that it can serve as base for the simulation of this process for a specific project.

The definition of execution plan an instantiated process consists a specification of some characteristics the process planning that make possible its previous execution. This way, the user can shape one or more plans to verify how the instantiated process holds from the characteristics of execution a specific project: financial resource; metric; estimates; profiles and its hierarchy; cost; and schedule. These attributes will be parameterized to *ProSimulator* tool.

4.1 Specification of Meta-Model Structure of *ImPProS*

From the previous agreement of the real importance of a software process meta-model in a gradual software process definition in automatized way, on the basis of structure adapted in Figure 2, it searched to project a repository that could add the concepts defined in section 3 with the considered ones by quality norms/models of software processes. As defined by Falbo (Falbo, 1998) the definition of ontology about software process composed of all the standard content this type of process, not taking in consideration terms associates to the quality norms/models of software process.

Taking care of the needs proposals for the *ImPProS* about its software process definition from quality norms/models, it initially analyzed two types of quality patterns standards for this goal: the ones that we call in this work **Maturity Norms/Models**, which describes guiding for the definition and implantation of processes through specified practices (activities, tasks, products generated, etc.) that they are used directly in the software process, for example the CMMI, ISO/IEC 15504, etc.; and **Reference Models**, similar to the maturity norms/models however they do clearly not indicate practices for processes composition but intentions (objective to be reached), waited results (produced device, a significant change of state and the attendance of specifications) and additional information (references that can help in the process definition and implementation) that they are described through a relationship with maturity norms/model, we can call the MPS.Br (Softex, 2005).

Beyond of software process, for the *ImPProS* meta-model was used the **ISO/IEC 12207 norm** for establishing a common architecture for the life cycle of software processes with a terminology well defined, contends processes, activities and tasks to be applied during the software products supply, development, operation and maintenance. It allows that the processes are specified with a unified terminology and this norm serves as base to promote the relationship between the maturity norms/models from the mapping of processes and practices specified by these norms/models to the processes, activities and tasks in ISO/IEC 12207 norm.

In addition to this, the **ISO/IEC 9126 norm** (ISO, 2002) was also used to identify the quality requirements of product. Such characteristics will influence the process about activities, methods and techniques. The activities of the specialized process

will have to be adapted to the life cycle model chosen for the project and new activities could be inserted in a life cycle. The influence of activities to the process from the characteristics desired to the product is proceeding the relationship between activities and tasks of ISO/IEC 12207 norm and the relevance degree of the quality characteristics presented in ISO/IEC 9126 norm proposal for ODDO et. al. (Oddo, 2003).

Figure 2 presents, from the use of SPEM notations - Software Process Engineering Meta-model (OMG, 2005), the relationship existing between the types of quality norms/models used in *ImPProS*, the ISO/IEC 12207 norm and the ISO/IEC 9126 norm. It is important to perceive the types of relationships existing between them, which will produce the results for the composition of the terminologies the software process meta-model of *ImPProS*.
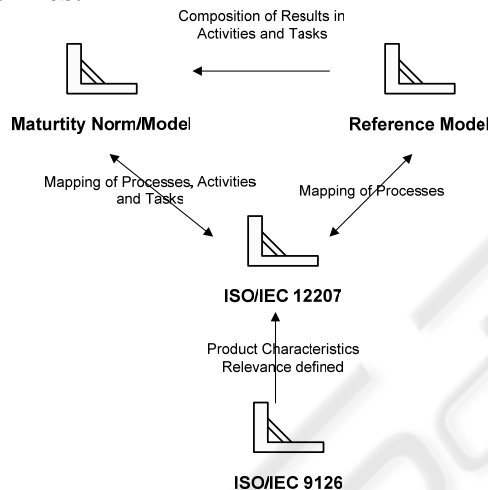

Figure 2: Relationship between quality norms/models adopted by software process meta-model of *ImPProS*.

From the relationships considered in Figure 2, Figure 3 presents, from the SPEM notations, an adaptation of the concepts considered by Falbo, presented in the session 3, and the patterns adopted for quality norms/models used for the process specification of *ImPProS*. This adaptation defines all the elements that compose the software processes meta-model of *ImPProS*.

It can be noticed, in Figure 3, the representation of types of mappings between the quality norms/models and ISO/IEC 12207 norm, which will produce a great base for the software processes definition and implementation in *ImPProS*. It is observed that all are formed by software processes and these have activities, if its origins will be maturity norms/models, and waited results, if they

are resultant of reference models. In turn, the waited results are mapped in activities in order to describe the software process. These activities have: an **origin**, which does not restrict that it is exclusively deriving of a maturity norm/model or ISO/IEC 12207 norm, being able to originate from a type of organization, a type of software project or simply generic; a **granularity**, that defines the composition of activities, or either, makes possible to describe if the activity can be decomposed in others ones or if its value is atomic (elementary); and a **type**, that it classifies in accordance with its implementation goal (Construction, Management or Quality Assurance).

In the following section, we have the description that detail the automation of some functionalities provided for software process meta-model of *ImPProS*.

4.2 Automation of *ImPProS* Meta-Model

The software process meta-model was developed and integrated in the *ImPProS* environment to assist the maintenance and management of terminologies this repository described in previous section.

It is composed of some management services that specify the maturity norms/models or reference models that will serve as base for software processes definition and implementation.

After that, the processes that compose them must be configured in the *ImPProS*. The software process meta-model makes possible to keep the information about origin of the software process. From this information, the activities used to specify the processes can be registered through its detailing about origin, granularity, type, restriction, composition, chaining and others ones.

The human, software and hardware resources have a specific area to be registered and associated to an activity. It is important to emphasize that a categorization of software resource is presented in this automation in order to detail the tools available in *ImPProS*.

All types of procedures can be visualized in *ImPProS*, and its use depends on the development paradigm and technology used for the software project. Some procedures can have: activities patterns to help in detailing of activities; tools associated to automatize its use.

We can find the mapping between the activities and processes constant in the maturity norms/models with the activities of ISO/IEC 12207 norm, from the analysis of tack between these components about its use. It is possible to refer the waited results found in

reference model with activities of maturity norms/models to specify how these ones can be executed inside a software process.

Finally, the automation of software process meta-model has some particularities to specify and choose the best life cycle model in guiding a software process, through its structure and characterization.

The automation of the software process meta-model by itself does not represent an advantage for the process definition and implementation. The *ImPProS* structure contemplates the automation of tools for standard process definition, process specialization and instantiation that to provide the customization with the terminologies constant in this meta-model. This way, the importance of *ImPProS* meta-model is perceived through the conception of a great repository of components that to provide the specification and the refinement the software process near the reality of a specific organization and their software projects.

5 FINAL CONSIDERATION AND FUTURE WORK

Any product resultant of an engineering activity, it is waited that the software products have intrinsic characteristics of quality perceivable by user. The software quality can be seen a set of characteristics that must be reached in a determined degree so that the product takes care of the needs of its users.

However, it was observed that the software product quality is on the software development process. In other words, it is not possible to add quality to software after soon. Thus, in 90's, it had a great concern about the quality of process the
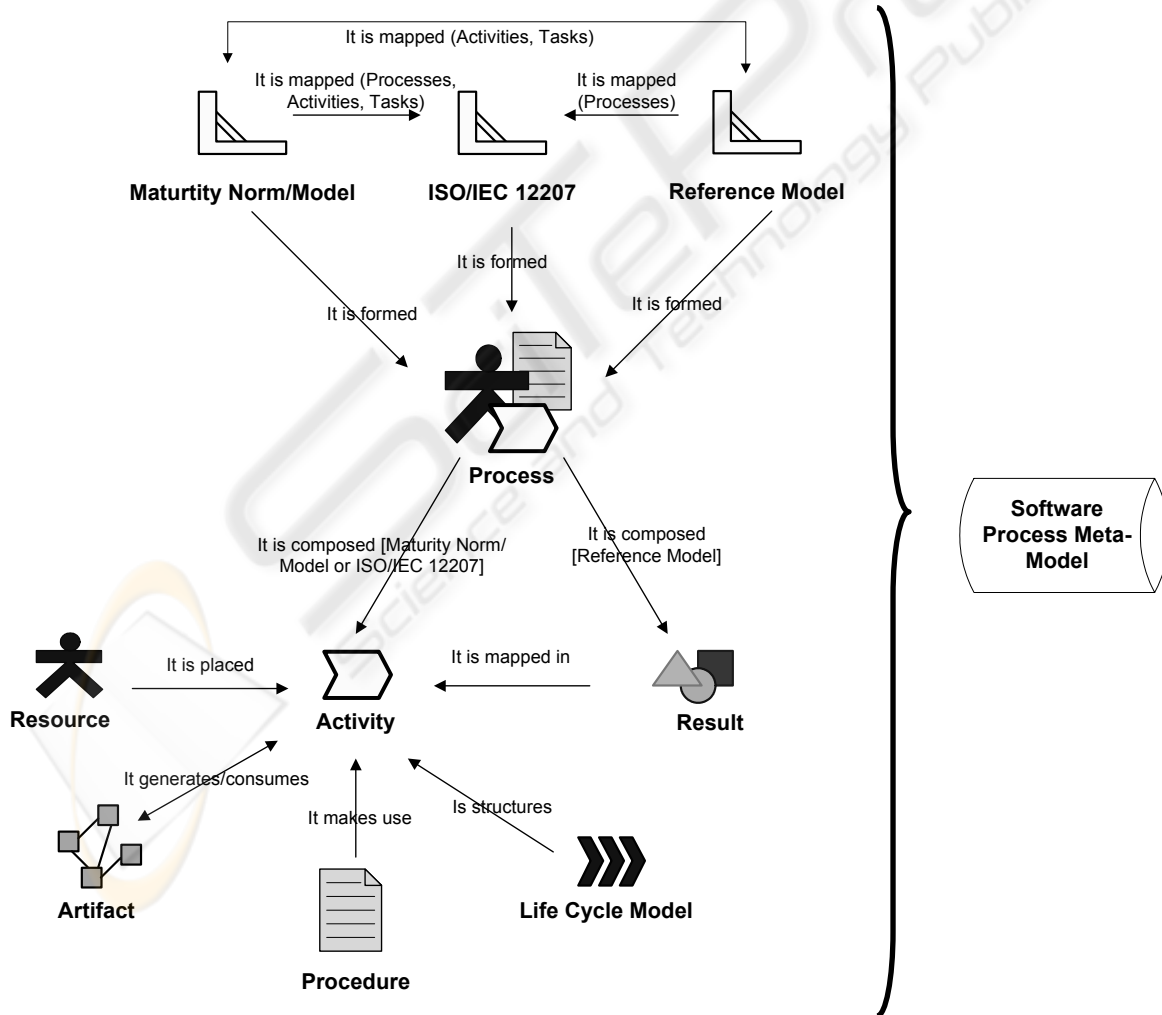
Figure 3: Composition Elements from Software Process Meta-Model of *ImPProS*.

software development organizations. This concern resulted in models to evaluate and to improve the software processes, whose goal is to give an indication of the maturity of a software process and to define action for evolutes it.

It expects that during the years the software development organizations adjust its software processes for the development of quality products inside of trustworthy stated periods. Moreover, these organizations will constantly be pressured to optimize its processes of development and maintenance, to produce products the lesser costs and with increasing quality.

Thus, an environment capable to provide the gradual software processes implementation from the definition, simulation, execution and evaluation of this process has a basic importance so that the scene of process improvement the software development organizations is more brightened up with the automation of the activities, represented through flows.

As we saw, a software process meta-model, which adds the components of this type of process associated the quality patterns inferred by systems development industry, brings an enormous advantage during the processes definition and implementation, as well as its refinement for the attendance of adequate organization and software project characteristics. The goal of this work was to present how the quality models/norms of software process and product hold together to the components of a software process and the importance to have this automatized meta-model.

Currently the software process meta-model meets total automatized and serving as base for the tools of standard process definition, software process specialization and instantiation in *ImPProS*.

REFERENCES

Balduino, R., 2002. Implementação de um processo de desenvolvimento de software: uma abordagem passo-a-passo, Rational Software White Paper.

Barbosa, I. M., 2005, Análise de Características de Projetos de Software para a Definição de Processo de Software, Trabalho de Graduação apresentado ao CIn/UFPE, orientador Prof. Alexandre Vasconcelos, Recife-PE.

Belkhatir, N., Estublier, J., Melo, W. ADELE-TEMPO: an Environment to Support Process Modelling and Enaction, In: FINKELSTEIN, A. et al. (Ed.). Software Process Modelling and Technology. Taunton: Research Studies Press, 1994.

Chrissis. M. B., Konrad, M. and Shrum, S., CMMI Guidelines for Process Integration and Product Improvement, Addison-Wesley, 2003.

Christie, A. Software Process Automation: The Technology and its adoption, Berlin: Springer Verlag, 1997.

Cunha, M. B. F. L., 2005. Análise das Características Organizacionais para a Definição de Processo de Software, Trabalho de Graduação apresentado ao CIn/UFPE, orientador Prof. Alexandre Vasconcelos, Recife-PE.

Falbo, R A., 1998. Integração de Conhecimento em um Ambiente de Desenvolvimento de Software, Orientadora: Ana Regina Cavalcanti da Rocha. Tese de Doutorado, COPPE/UFRJ.

ISO/IEC TR 15504, Parts 1-9, 1998. Information Technology – Software Process Assessment, International Organization for Standardization.

ISO/IEC TR 12207, 2000. Amendment: Information Technology – Amendment to ISO/IEC 12207, PDAM 3 version.

ISO/IEC TR 9126, 2002. Software engineering – product quality, International Organization for Standardization.

Machado, L. F., 2000. Modelo para Definição de Processos de Software na Estação Taba, Orientadora Ana Regina Cavalcanti Rocha, Tese de Mestrado, COPPE/UFRJ.

Mcfeeley, B., 1996. IDEALSM: A User's Guide for Software Process Improvement, Software Engineering Institute Handbook. Carnegie Mellon University. CMU/SEI-96-HB-001.

Oddo, M., Rocha, A. R., 2003. Relating Process Activities to Software Quality Characteristics, trabalho submetido à revista Software Quality Journal.

Oliveira, S. R.B., Vasconcelos, A. M. L., Rouiller, A. C., 2005. Uma Proposta de um Ambiente de Implementação de Processo de Software, Revista InfoComp – Revista de Ciência da Computação da UFLA – vol. 4, n. 1, Lavras-MG.

Oliveira, S. R. B., Vasconcelos, A. M. L., 2006. Modelo Comportamental de um Ambiente de Implementação de Processo de Software, InfoComp - Revista de Ciência da Computação - vol.5, n.1, março, Lavras/MG.

OMG – Object Management Group, 2005. SPEM – Software Process Enginering Metamodel Specification, version 1.1, formal/05-01-06.

Reis, C. A. L., 2003. Uma Abordagem Flexível para Execução de Processos de Software Evolutivos, Tese de Doutorado, Instituto de Informática, Universidade Federal do Rio Grande do Sul.

Rocha, A. R. C., Maldonado, J. C. and Weber, K. C., Qualidade de software: teoria e prática, São Paulo: Prentice-Hall, 2001.

Softex - Sociedade para Promoção da Excelência do Software Brasileiro, 2005. MPS.BR - Melhoria de Processo do Software Brasileiro, Guia Geral, versão 1.0.