Chapter

Overview of Software Process Models and Descriptive Criteria for their Analysis

In this chapter, we present an overview of the process models that will be detailed in the following chapters. Depending on their focus, these models have been divided into two groups: activity-oriented and people-oriented models. Also we describe the criteria used to analyse the models, which will serve to ascertain their advantages and disadvantages.

2.1. SOFTWARE PROCESS MODELS

The parameter used here to classify the process models is based on defining what process elements the model covers. Specifically, the models analysed have been classified as shown in Figure 2.1.

The *Activity-Oriented Models* focus on defining the functions, activities and information of the software process management, development and/or supporting processes. The *People-Oriented Models* focus on defining the people involved in the software process and their relationships.

These two model categories are further divided into two alternatives depending on the goal for which the model was developed:

- Descriptive Models
- Prescriptive Models.

Descriptive Models are mainly concerned with specifying a process now used within an organisation or with representing a proposed process to be able to predict some process features. Descriptive models answer the question, "How is software now developed?" Or how has software been developed? (Lonchamp, 1993). *Prescriptive Models* focus on defining how the process should be enacted. Prescriptive models answer the question "How should software be developed?" (Lonchamp, 1993).

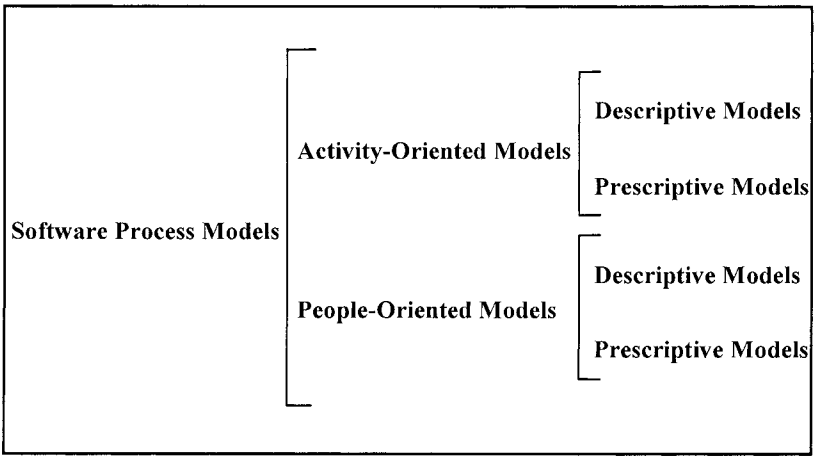

Figure 2.1. Classification used to analyse software process models

Table 2.1 lists the models belonging to each category. We have selected the most significant and representative models according to (Lonchamp, et al., 1990; Armenise et al., 1992; Lonchamp, 1994; McChesney, 1995; Ambriola et al., 1997; Fuggetta, 2000) for more detailed description in Chapters 3 and 4. They are listed in Table 2.1 under the surveyed models column. The selected models have served as a source of inspiration for most of the other models in this area (Finkelstein et al., 1994; Derniame et al., 1999; Acuña et al., 2001) or else somehow represent general features of other models (Beck et al., 2001).

In the following chapters, we will analyse the models that are representative of each class from the viewpoint of the software process dimensions described in the following section. It should be noted that this review includes both models that are designed to simply model the process and models that pursue other objectives, such as process evaluation, improvement or prediction. These models are also considered, because, to achieve their specific objectives, they model the process and are, therefore, of interest. The models surveyed in this part of the book are illustrated in Figure 2.2.

CLASS	SUBCLASS	SURVEYED MODELS	OTHER MODELS
Activity-Oriented Model	Descriptive	- TAME (Basili & Rombach, 1988) - FUNSOFT Nets-Based (Deiters & Gruhn, 1991) - STATEMATE (Kellner, 1991) - PRISM model of changes (Madhavji, 1992)	- Amadeus (Selby et al., 1991) - Wolf & Rosenblum (1993) - Conradi et al. (2000)
	Prescriptive	- TRIAD (Ramanathan & Sarkar, 1988) - Marvel (Kaiser, 1988a; Kaiser, 1988b) - IPSE 2.5 (Ould & Roberts, 1988; Warboys, 1989) - SPADE (Bandinelli et al., 1994) - ISO/IEC 12207-1995 (ISO/IEC, 1995; ISO/IEC, 2002) - IEEE 1074 (IEEE, 1997) - ISO 9001 (ISO-9001, 2000) - CMMI (SEI, 2002c; SEI, 2002d)	- GRAPPLE (Huff & Lesser, 1988) - HFSP (Katayama, 1989) - Appl/A (Sutton et al., 1990) - Articulator (Mi & Scacchi, 1990) - Minsky & Rozenshtein (1990) - Oikos (Ambriola & Jaccheri, 1991) - EPOS (Conradi et al., 1991b; Jaccheri et al., 1992; Conradi et al., 1994b) - Adele-Tempo (Belkhatir et al., 1993) - Hakoniwa (Iida et al., 1993) - LOTOS/SPD (Yasumoto et al., 1994) - E3 (Baldi et al., 1994) - Trillium (Bell, 1994) - DoD 2167A-1995 (DoD, 1995) - TickIT (BSI, 1995) - WADP (Weske et al., 1999)
People-Oriented Model	Descriptive	- Systems Dynamics Based (Abdel-Hamid & Madnick, 1989) - Process Cycle (Madhavji, 1991) - Agile Methods: eXtreme Programming (Beck, 1999) - WinWin (Boehm et al. 1998)	- Spiral Model (Boehm, 1988) - Cain & Coplien (1993) - Actor Dependency (Yu, 1993; Yu & Mylopoulos, 1994) - SCRUM (Schwaber, 1995) - DSDM (Stapleton, 1997) - Evo (Gilb, 2002)
	Prescriptive	- PMDB+ (Penedo & Shu, 1991) - ALF (Canals et al., 1994) - SOCCA (Engels & Groenewegen, 1994) - Unified Software Development Process or Unified Process (Jacobson et al., 1999) - People CMM (Curtis et al., 2001)	- CHAOS (De Cindio et al., 1988) - COSMOS (Yeh et al., 1991; Mittermeir & Schlemmer, 1992) - Conversation Builder (Kaplan et al., 1992) - MERLIN (Junkermann et al., 1994) - PADM (Bruynooghe et al., 1994) - PEACE (Arbaoui & Oquendo, 1994) - OPEN (Graham et al., 1997) - Personal Software Process (Humphrey, 1997) - ISO/IEC 15504 (ISO/IEC 15504, 1998) - Team Software Process (Humphrey, 1998a)

Table 2.1. Models of each class found in the literature

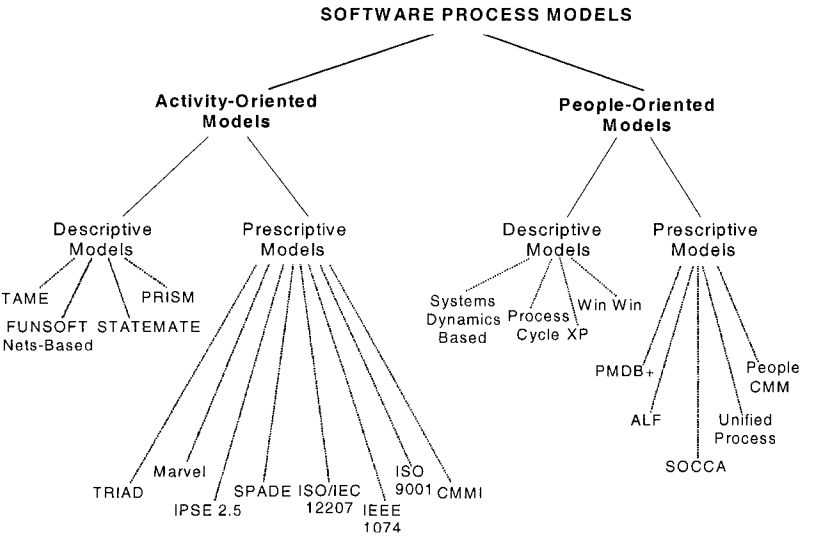

Figure 2.2. Surveyed software process models

Two dimensions are used to evaluate and compare the models shown in Figure 2.2. These are the modelling dimension and the representation dimension. The two dimensions (modelling and representation) have been established on the basis of the features of the existing models and the features that software process models should ideally have, with the aim of providing a good understanding, organisation and evaluation of the state of the art in software process modelling.

The criteria examined for each model are described below.

2.2. DESCRIPTIVE CRITERIA

2.2.1. Modelling-Related Criteria

The modelling dimension is divided into two top-level criteria. These criteria determine the process features that a model can model. Any individual model will explicitly model a given range of process features, which make up a range of descriptors associated with the model. The process features that can be explicitly modelled by existing approaches are:

- Process elements represented by the model. The possible values are one or more agents, activities, artefacts, roles, capabilities and events.

- Process environments covered by the model. The possible values are organisational environment, social environment and technological environment.

The meaning of the different values is described in the following.

2.2.1.1. Elements

The basic process-related *Elements* are outlined below:

- *Agent* or *Actor*: an entity (person or system) that enacts a process. Actors can be divided into two groups as regards their relationship to the computer: a) human actors, who are the people who develop software or are involved in the software *process* and are possibly organised as teams; and b) system actors or system tools, which are the computer software or hardware components. An actor is characterised by the properties of its role and its availabilities. An actor can play several roles, which are composed of consistent sets of activities. A role can be played by several co-operative actors.

- *Role*: a description of a set of agent or group responsibilities, rights, skills and capabilities required to perform a specific software process activity. This is an association between agents and activities in terms of a defined set of responsibilities executed by agents.

- *Capability*: an underlying trait of a person, which is causally related to good or excellent performance in a particular role in a particular organisation. Personal capabilities determine behaviours, indicating the person's general predisposition to *behave* or react in a given way, for example, tenacity, self-control, stress tolerance, etc.

- *Activity*: the stage of a process that produces externally visible changes of state in the software product. An activity can have an input, an output and some intermediate results, commonly termed products (for example, requirements specification documents, database schemata, etc.). The activity includes and implements procedures, rules, policies and goals to generate and modify a set of given *artefacts*. An activity can be performed by a human agent or using a tool. The activities can be divided into more elementary activities; that is, there are elementary or compound activities. The activities can be organised in networks with both horizontal (chained) and vertical (decomposition) dimensions. The activities are associated with roles, other activities and artefacts.

- *Artefact* or *Product*: the output and input of an activity. The products can be created, accessed or modified during a process activity. An artefact produced by an activity can be used later as raw material for the same or another activity to produce other artefacts. An artefact can have a long lifetime, and there may be different versions of each artefact as the software process evolves.

- *Event*: a noteworthy occurrence happening at a specific moment in time. Event-based models provide a different view of the activities, thereby evidencing process dynamism.

2.2.1.2. Environments

The *Environment* criterion identifies the contexts that influence and are influenced by the software process. Software process models should consider three interrelated environments: the organisational environment, the social environment and the technological environment. These environments are described as follows:

- *Organisational environment*: this criterion indicates whether the models account for organisational issues, such as the culture, behaviour, design, development and evolution of the organisation.

- *Social environment*: this criterion indicates whether the models cover the people involved, the relationships between these people and with the organisation, that is, capabilities such as creativity, social interaction within a team and flexibility of the organisation with respect to the environment, etc.

- *Technological environment*: this criterion indicates whether the models cover the software techniques, tools, infrastructure, environments and methodologies for both the production of software and the management, improvement and control of the software process.

2.2.2. Representation-Related Criteria

This second dimension has to do with what guidance is provided for organising the information in a model, what properties the process modelling notation has and what quality of information the software process model representation affords. There are two types of representation-related criteria:

- Information perspectives

- Notational characteristics.

The information perspectives are related to the organisation of the information captured within a process model by applying different viewpoints: functional, behavioural, organisational and informational. These viewpoints can be combined to provide an integrated, consistent and complete process description. The notational characteristics are also presented from different viewpoints: a) information quality, which refers to the manner in which the software process model components are represented: informal, formal or automated, and b) formalised notation, which refers to the format: text or graphic.

The meaning of the possible values of the representation-related criteria is described below.

2.2.2.1. Information Perspectives

The information in a process model can be structured from different viewpoints. Curtis lists the following *Information Perspectives* commonly found in the literature (Curtis et al., 1992):

- *Functional perspective*: This represents what process elements are being executed and what information item flows are relevant for these elements.

- *Behavioural perspective*: This represents when (that is, order) and under which conditions the process elements are triggered.

- *Organisational perspective*: This represents where and by whom in the organisation the process elements are executed.

- *Informational perspective*: This represents the information items output or manipulated by a process, including their structure and relationships.

The models are built according to the languages, abstractions or formalisms created for representing the specific information about these information perspectives. The most commonly used language in practice is (structured) natural language due to its flexibility. Table 2.2 lists the representation abstractions (excluding natural language) organised according to the above-mentioned perspectives. These are the most popular abstractions used in software process research. None of these abstractions covers all the information perspectives. Therefore, most of the models found in the literature employ languages based on more than one of these abstractions. These models integrate multiple representation paradigms (that is, different process models), although this generally makes the model definition more complicated (de Vasconcelos & Werner, 1997).

LANGUAGE BASE	INFORMATION PERSPECTIVES
Procedural programming language (Ramanathan & Sarkar, 1988)	Functional Behavioural Informational
Systems analysis and design, including data flow diagram (Frailey, 1991), structured analysis and design technique (SADT) and structure diagrams (McGowan & Bohner, 1993)	Functional Organisational Informational
Artificial intelligence languages and approaches, including rules and pre-/postconditions (Barghouti et al., 1995)	Functional Behavioural
Events and triggers // Control flow (Finkelstein et al., 1994)	Behavioural
State transition and Petri nets (Deiters & Gruhn, 1991; Bandinelli et al., 1995) // Statecharts (Kellner & Hansen, 1989; Kellner, 1991; Harel & Politi, 1998; Raffo & Kellner, 1999)	Functional Behavioural Organisational
Functional languages // Formal languages (Curtis et al., 1992; Huff, 1996)	Functional
Data modelling, including entity-relationship diagrams, relationship declarations and structured data (Penedo & Shu, 1991)	Informational
Object modelling, including class and instance types, hierarchy and inheritance (Engels & Groenewegen, 1994)	Organisational Informational
Quantitative modelling, including quantitative operational research and systems dynamics techniques (Abdel-Hamid & Madnick, 1991)	Behavioural
Precedence networks, including actor dependency modelling (Yu & Mylopoulos, 1994; Briand et al., 1995)	Behavioural Organisational

Table 2.2. Information perspectives of the process and the applicable language bases

2.2.2.2. Notational Characteristics

As mentioned earlier, this aspect is addressed from two viewpoints: information quality and formalised notation.

Information quality indicates whether the models provide informal, formal or automated specifications of the software process modelling elements. These values are:

- *Informal specifications*: manual, qualitative, subjective and informal representation of the process.

- *Formal specifications*: formal representation of the process, related to the formal description or prescription of the process.

- *Automated specifications*: computerised representation of the process. Automated models may include the features and principles of their manual counterparts, but they are automated to some extent and the recommended set of possible steps are restricted to the ones permitted by the model.

From the viewpoint of *Formalised notation*, some models adopt a *text*-based representation for the process model, whereas others take a *graphic*-based approach.

2.2.3. Criteria Employed by Other Authors

There are a number of reference frameworks for classifying and characterising software process models (Madhavji, 1991; Conradi et al., 1991a; Mi & Scacchi, 1991; Curtis et al., 1992; Benali & Derniame, 1992; Armenise et al., 1993; Lonchamp, 1993; Lonchamp, 1994; McChesney, 1995; Ambriola et al., 1997). Traditionally, classification schemas have focused on the process modelling language style (Madhavji, 1991; Kellner, 1991; Conradi et al., 1991a; Mi & Scacchi, 1991; Armenise et al., 1993). The selection of one or more styles depends principally on the information perspectives (Curtis et al., 1992) according to which the model elements are organised and on the notational characteristics used (Lonchamp et al., 1990). This book uses these two criteria.

Benali and Derniame (1992) presented an informal basis for assessing the scope of a software process model, highlighting the essential properties of the software process that a modelling formalism should represent. This basis takes into account three aspects: "what does the software process model?", "who are the process actors?" and "when do the actors intervene in the process?" Later, more systematic software process model classifications were developed. McChesney (1995) proposed a classification schema for comparing, characterising and applying different software process modelling approaches. This schema considers four dimensions on the basis of which these approaches can be characterised (McChesney, 1995): a) the objectives for which the model is used; b) the properties of the process modelling language; c) the process features that can be modelled; and d) the world view associated with the approach. This schema allows a more rigorous comparison of the existing process models. However, it does not include the ideal or required characteristics for comparing and evaluating models. To give a better understanding of the field of software process modelling, Lonchamp (1994) developed a four-part assessment grid: a) software process modelling approaches, b) process modelling language, c) metaprocesses, and d) representation engine. Ten European universities who were involved in

projects related to software process modelling and technology completed this grid.

However, the existing classification schemas do not systematically address the features of a model considered in the context of its relationships to other entities: organisations, social interaction activities and modelling procedures. The traditional classification criteria focus on the technological dimension and do not include the organisational dimension, whose presence, although its development within software modelling is in its early days, has grown in more recent research. This transition is taking place as process engineers become aware of the importance of the social and cultural environment on the initiation, development, application and evaluation of high-quality software processes.