An engineering approach to Data Mining projects

Óscar Marbán[1], Gonzalo Mariscal[2], Ernestina Menasalvas[1], and Javier Segovia[1]

[1] Facultad de Informática, Universidad Politécnica de Madrid.
Campus de Montegancedo s/n, 28660 Boadilla del Monte (Madrid), Spain
{omarban,emenasalvas,fsegovia}@fi.upm.es
[2] Universidad Europea de Madrid
gonzalo.mariscal@uem.es

Abstract. Both the number and complexity of Data Mining projects has increased in late years. Unfortunately, nowadays there isn't a formal process model for this kind of projects, or existing approaches are not right or complete enough. In some sense, present situation is comparable to that in software that led to 'software crisis' in latest 60's. Software Engineering matured based on process models and methodologies. Data Mining's evolution is being parallel to that in Software Engineering. The research work described in this paper proposes a Process Model for Data Mining Projects based on the study of current Software Engineering Process Models (IEEE Std 1074 and ISO 12207) and the most used Data Mining Methodology CRISP-DM (considered as a "facto" standard) as basic references.

1 Introduction

In its early days, software development focused on creating programming languages and algorithms that were capable of solving almost any problem type. The evolution of hardware, continuous project planning delays, low productivity, heavy maintenance expenses, and failure to meet user expectations had led by 1968 to the *software crisis* [1].This crisis was caused by the fact that there were no formal methods and methodologies, support tools or proper development project management. The software community realized what the problem was and decided to borrow ideas from other fields of engineering. This was the origin of software engineering (SE). As of then process models and methodologies for developing software projects began to materialize.

Software development improved considerably as a result of the new methodologies. This solved some its earlier problems, and little by little software development grew to be a branch of engineering. This shift means that project management and quality assurance problems are being solved. Additionally, it is helping to increase productivity and improve software maintenance.

The history of knowledge discovery in databases (KDD), now known as Data Mining (DM), is not much different. In the early 90s, when the KDD processing term was first coined [2], there was a rush to develop DM algorithms that were capable of solving all problems of searching for knowledge in data. Apart from developing algorithms, tools were also developed to simplify the application of DM algorithms. From the viewpoint of DM process models, the year 2000 marked the most important milestone. CRISP-DM (*CRoss-Industry Standard Process for DM*)[3] was published.

While it is true that the number of applied projects in the DM area is expanding rapidly, neither all the project results are in use [4–6] nor do all projects end successfully [7, 8]. The failure rate is actually as high as 60% [9]. CRISP-DM is the most commonly used methodology for developing DM projects as a "facto" standard.

Are we at the same point as SE was in 1968? Certainly not, but we do not appear to be on a par yet either. Looking at the KDD process and how it has progressed, we find that there is some parallelism with the advancement of software. From this viewpoint, DM project development is defining development methodologies to be able to cope with the new project types, domains and applications that organizations have to come to terms with. Nowadays, SE pay special attention to organizational, management or other parallel activities not directly related to development, such as project completeness and quality assurance. CRISP-DM has not yet been sized for these tasks, as it is very much focused on pure development activities and tasks.

This paper is moved by the idea that DM problems are taking on the dimensions of an engineering problem. Hence, the processes to be applied should include all the activities and tasks required in an engineering process, tasks that CRISP-DM might not cover. The proposal is inspired by the work done in SE derived from other branches of engineering. It borrows ideas to establish a comprehensive process model for DM that improves and adds to CRISP-DM. Further research will be needed to define methodologies and life cycles, but the basis of a well-defined process model will be there.

2 Data Mining Process Models

There is some confusion about the terminology different authors use to refer to process and methodology.

A process model is defined as the set of tasks to be performed to develop a particular element, as well as the elements that are produced in each task (outputs) and the elements that are necessary to do a task (inputs) [10]. The goal of a process model is to make the process repeatable, manageable and measurable (to be able to get metrics).

Methodology can be defined as the instance of a process model that both lists tasks, inputs and outputs and specifies how to do the tasks [10]. Tasks are performed using techniques that stipulate how they should be done. After selecting a technique to do the specified tasks, tools can be used to improve task performance.

Finally, the life cycle determines the order in which each activity is to be done [11]. A life cycle model is the description of the different ways of developing a project.

From the viewpoint of the above definitions, what do we have in DM? Does DM have process models and/or methodologies? The KDD process [12] has a process model component because it establishes all the steps to be taken to develop a DM project, but it is not a methodology because its definition does not set out how to do each of the proposed tasks. It is also a life cycle. Like the KDD process, Two Crows [13] is a process model and waterfall life cycle. At no point does it set out how to do the established DM project development tasks. SEMMA [14] is the methodology that SAS proposed for developing DM products. Although it is a methodology, it is based on the technical part of the project only. Like the above approaches, SEMMA also sets out a waterfall life cycle, as the project is developed through to the end. 5 A's [15] is a process

model that proposes the tasks that should be performed to develop a DM project and was one of CRISP-DM's forerunners. Therefore, their philosophy is the same: it proposes the tasks but at no point suggests how they should be performed. The life cycle is similar to the one proposed in CRISP-DM. Data Mining Industrial Engineering [16] is a methodology because it specifies how to perform the tasks to develop a DM project in the field of industrial engineering. It is an instance of CRISP-DM, which makes it a methodology, and it shares CRISP-DM's associated life cycle. Finally, CRISP-DM [3] states which tasks have to be carried out to successfully complete a DM project, making it a process model. It is also a waterfall life cycle. CRISP-DM also has a methodological component, as it gives recommendations on how to do some tasks. However, it just proposes other tasks, giving no guidance about how to do them. Therefore, we class CRISP-DM as a process model.

3 Software Engineering Process Models

The SE panorama is quite a lot clearer, and there are two well-established process models: IEEE 1074 [17] and ISO 12207 [18] . In the following, we will analyze both processes in some detail and propose a generic joint process model. This joint model will then be used for comparison with and, if necessary, to expand the CRISP-DM.

3.1 IEEE STD 1074

The IEEE Std 1074 [17] specifies the processes for developing and maintaining software. IEEE Std 1074 neither defines nor prescribes a particular life cycle. Each organization using the standard should instantiate the activities specified in the standard within its own development process.Next, the key processes defined in this process model will be described. The *software life cycle selection process* identifies and selects a life cycle for the software under construction. The *project management processes* are the set of processes that establish the project structure, and coordinate and manage project resources throughout the software life cycle. *Development-oriented processes* start with the identification of a need for automation. With the support of the integral process activities and under the project management plan, the development processes produce software (code and documentation) from the statement of the need. Finally, the activities for installing, operating, supporting, maintaining and retiring the software product should be performed. *Integral processes* are necessary to successfully complete the software project activities. They are enacted at the same time as the software development-oriented activities and include activities that are not related to development. They are used to assure the completeness and quality of the project functions.

3.2 ISO 12207

ISO 12207 divides the activities that can be carried out during the software life cycle into primary processes, supporting processes and organizational processes.

The *primary life cycle processes* are a compendium of processes that serve the primary parties throughout the software life cycle. A primary party is the party that starts or enacts software development, operation or maintenance.

The *supporting life cycle processes* support other processes as an integral part with a distinct purpose and contribute to the success and quality of the software project. The supporting processes are divided into subprocesses, which can be used in other processes defined by ISO 12207. The supporting processes are used at several points of the life cycle and can be enacted by the organization that uses them. The organization that uses and enacts a supporting process manages that process at project level as per the management process, establishes an infrastructure for the process as per the infrastructure process and drives the process at the organizational level as per the improvement process.

The *organizational life cycle processes* are used by an organization to perform organizational functions, such as management, personnel training or process improvement. These processes help to establish, implement and improve software process, achieving a more effective organization. They tend to be enacted at the corporate level and are outside the scope of specific projects and contracts.

3.3 Unification of IEEE STD 1074 and ISO 12207

Having reviewed IEEE Std 1074 and ISO 12207, the goal is to build a joint process model that is as generic as possible to then try to use it as a basis for defining a process model against which to compare CRISP-DM.

If we compare both models, clearly most of the processes proposed in IEEE Std 1074 match up with ISO 12207 processes and vice versa. To get a joint process model we have merged IEEE Std 1074 and ISO 12207 processes. The process selection criterion was to select the most thoroughly defined IEEE Std 1074 and ISO 12207 processes and try not to merge processes from different groups in different process models. According to this criterion, we selected IEEE Std 1074 as a basis, as its processes are more detailed. Additionally, we added the ISO 12207 *acquisition* and *supply processes*, because IEEE Std 1074 states that ISO 12207 acquisition and supply processes should be used [17] if it is necessary to acquire or supply software.

Figure 1 shows the joint process model developed after studying IEEE Std 1074 and ISO 12207 according to the above criteria. Figure 1 also shows the details of the major process groups, the activities they each involve according to the selected standard for that process group. In the next section we will analyse which of the above activities CRISP-DM includes and which it does not in order to try to build a process model for DM projects.

4 SE process model vs. CRISP-DM

This section presents a comparison between CRISP-DM and the joint process model discussed in section 3.3. This comparison should identify what SE model elements are applicable to DM projects and are not covered by CRISP-DM. This way we will be able to build a process model for DM projects based on fairly mature SE process models.

Note that the correspondence between CRISP-DM and SE process model elements is not exact. In some cases, the elements are equivalent but the techniques are different, whereas, in others, the elements have the same goal but are implemented completely differently.

PROCESS	ACTIVITY
Acquisition	
Supply	
Software life cycle selection	Identify available software life cycles
	Select software life cycle
Project management processes	
Initiation	Create software life cycle process
	Allocate project resources
	Perform estimations
	Define metrics
Project monitoring and control	Manage risks
	Manage the project
	Retain records
	Identify software life cycle process improvement needs
	Collect and analyze metric data
Project planning	Plan evaluations
	Plan configuration management
	Plan system transition
	Plan installation
	Plan documentation
	Plan training
	Plan project management
	Plan integration
Development-oriented processes	
Pre-development	
Concept exploration	Identify ideas or needs
	Formulate potential approaches
	Conduct feasibility studies
	Refine and finalize the idea or need
System allocation	Analyze functions
	Decompose system requirements
	Develop system architecture
Software importation	Identify imported software requirements
	Evaluate software import sources
	Define software import method
	Import software
Development	
Requirements	Define and develop software requirements
	Define interface requirements
	Priorizate and integrate software requirements

PROCESS	ACTIVITY
Design	Perform architectural design
	Design data base
	Design interface
	Perform detailed design
Implementation	Create executable code
	Create operating documentation
	Perform integration
Post-Development	
Installation	Distribute software
	Install software
	Accept software in operational environment
Operation and support	Operate the system
	Provide technical asistanse and consulting
	Maintain support request log
Maintenance	Identify software improvement needs
	Implement problem reporting method
	Maintenance support request log
Retirement	Notify user
	Conduct parallel operations
	Retire system
Integral processes	
Evaluation	Conduct reviews
	Create traceability matrix
	Conduct audits
	Develop test procedures
	Create test data
	Execute test
	Report evaluation results
Software configuration management	Develop configuration identification
	Perform configuration control
	Perform status accounting
Documentation development	Implement documentation
	Produce and distribute documentation
Training	Develop training materials
	Validate the training program
	Implement the training program

Fig. 1. Joint process model

4.1 Life cycle selection process

The purpose of the set of processes for selecting the life cycle (*Life cycle selection*) in projects is to *select a life cycle* for the project that is to be developed. Based on the type of product to be developed and the project requirements, life cycle models are identified and analysed and a model that provides proper support for the project is selected. This set of processes also extends to third party software *acquisition* and *supply*. These two processes cover all the tasks related to supply or acquisition management. CRISP-DM does not include any of the *acquisition* or *supply* processes at all. DM project development experience suggests that acquisition and supply processes may be considered necessary and third parties engaged to develop or create DM models for projects of some size or complexity. Developers undertaking a DM project also need to *select a life cycle*, and this depends on the type of project to be developed. Life cycle models are used for software development because not all projects are equal, neither do all developers and clients have the same needs. This also applies to DM projects, as a typical client segmentation, is quite a different kettle of fish from predicting aircraft faults. *Life cycle selection* is not an easy task, as you have to take into account the project type in terms of complexity, experience in the problem domain, knowledge of the data that are being analysed, variability, and data expiration. Therefore, the life cycle selection process is considered useful for DM projects. However, DM project life cycles will have to be defined, as no thorough studies on possible cycles for use or the variables or criteria that distinguish one life cycle from another have yet been conducted.

4.2 Project management processes

The set of processes defined here establish the project structure, and coordinate and manage project resources. The project *initiation* process defines the activities for creating and updating the project development or maintenance infrastructure . *Project planning* covers all the processes related to planning project management activities, including contingency planning. The *project monitoring and control process* analyses technical, economic, operational, and timetable risks in order to identify potential problems, and establish the steps for their management. Additionally, it also covers subprocesses related to project metric management.

Project management processes are evidently also necessary when we undertake a DM project. The tasks that are to be performed need to be planned, and there should be a contingency plan because of the high risk involved in DM projects. Also it is necessary to analyse project costs, benefits and ROI. Looking at the tasks covered by the CRISP-DM stages, however, only in the *business understanding (BU)* phase do we find tasks that are related to project management. The *identify major iterations* task is comparable to map activities for the selected life cycle, except that the DM project iterations are only roughly outlined as there are no defined DM life cycles. Additionally, the philosophy behind the *experience documentation* task is the same as the *identify software life cycle process improvement needs*. CRISP-DM's *inventory of resources* task accounts for resources allocation, although its tendency is to identify what resources are available rather than allocating resources throughout the project. CRISP-DM does not cover this issue.

The other tasks proposed by CRISP-DM directly match up with the SE process model tasks. And all the tasks that do not appear in CRISP-DM are considered necessary in a DM project. However, CRISP-DM's biggest snag in terms of project management is related to metrics (*Define metrics, retain records, collect and analyze metrics*). For the most part, this can be attributed to the field's immaturity. There is a need to define DM metrics in order to establish costs and deviations throughout project execution. The other major omission is the evaluation component (*Plan evaluations*). CRISP-DM does have a results evaluation stage, but what we are referring to here is process evaluation as a whole. Configuration management (*Plan configuration management*) aims to manage versions, changes and modifications of each project element. CRISP-DM does not cover DM project configuration management, but we believe that, because of the size of current projects and the teams of human resources working together on such projects, it should. Different people generate multiple versions of models, initial data sets, documents, etc., in a project. Therefore, if they are not well located and managed, it is very difficult to go back to earlier versions, should the current versions not be valid, and there is a risk of confusing models, data and documentation for different versions.

Additionally, any DM project should include tasks for managing the transfer and use of the results (*Plan system transition, plan installation*), tasks that CRISP-DM does not cover either. Finally, the other major oversight, fruit of process immaturity, is the documentation task (*Plan documentation*). Reports are generated in all stages, but there is no task aimed at planning what this documentation should be like to conform to thorough standards. This would improve documentation evaluation and review and facilitate work on process improvement.

4.3 Development-oriented processes

Software development-oriented processes start with the identification of a need to automate some tasks for performance using a computer (*Identify ideas or needs*). With the support of the *integral process* activities and subject to the project management plan (*Plan project management*), the *development processes* produce the software. Finally, activities for installing (*Installation*), operating (*Operation and support*), supporting (*Operation and support*), maintaining (*Maintenance*) and retiring (*Retirement*) the software product should be performed. They are subdivided into *pre-development*, *development* and *post-development* processes. DM projects start with the need to gather knowledge from an organization's data to help in business decision-making, knowledge that can be used directly or can be integrated into the organization's systems. This is the most mature set of processes at present, as all the existing "methodologies" for DM project development focus primarily on this part. As for SE, these processes can also be divided into *pre-development*, *development* and *post-development* stages.

Pre-development are related to everything that you have to do before you start to build the system, such as *concept exploration* or *system allocation requirements*. The *concept exploration* process includes identifying an idea or need (*Identify ideas or needs*) for the system to be developed, and the formulation (*Formulate potential approaches*), evaluation (*Conduct feasibility studies*) and refinement of potential solutions at system level (*Refine and finalize the idea or need*). Once the system limits have been established, a statement of need is generated for the system to be developed. This statement of need starts up the *system allocation* process and/or *requirements* process and feeds the *project management* processes. The statement of need is as necessary in DM projects as in any other project; it is a starting point for project development as it provides an understanding of the problem to be solved. Because of its importance, CRISP-DM already accounts for this process. However, it is spread across different stages and always in the *business understanding* stage at the start of the project. The *software importation* process is related to the reuse of existing software. In the case of software, this process provides the means required to identify what requirements imported software can satisfy and evaluate the software to be used. Software does, in principle, not need to be imported in a DM project, because a DM project gathers knowledge and does not develop software. Its equivalent in a DM project would be to import existing DM models that are useful for the current project. For example, one usual practice is to have a client clustering and use that clustering in the ongoing project to classify clients. Therefore, a process that manages the importation of DM models for use in the ongoing project is also required.

Development is responsible for building the software or gathering knowledge in the case of DM projects. There is no exact match between the development processes in DM projects and SE projects, as the ends are completely different. DM projects aim to gather knowledge, whereas SE projects target software construction. Even so, they share the same phases: *requirements* definition, solution *design* and solution development (*Implementation*). The requirements stage bears most resemblance, as its aim is to gather the client needs and describe these needs in practical terms for the designers and/or implementers (*Assess situation and Determine DM goals*). As for software, DM's design stage has to design the software support for data, since the available data

will ultimately be analysed on the software support. However, the key SE design task, which is ‘‘*perform architectural design”*, has no direct equivalent in DM. As already mentioned, the goal of SE design is to translate specifications and requirements into a preliminary design of the solution (i.e, object-oriented design). Therefore, perhaps the best thing would be to equate this task to the early decision made on what DM paradigms (clustering, classification, etc.), are to be explored to achieve the *DM Goals*. This would fit in with the later *implementation* phase, where the modelling technique will be selected (*Select modeling technique*) for each goal. There is no direct mapping between the implementation stages, as the goal they pursue is different. This is the best researched stage of DM, on which all the proposed “methodologies” focus. The implementation stage would be equivalent to gathering and analysing the data available for the project, the creation of new data from what are already available, tailoring for DM algorithms and the creation of DM models, all of which are covered by CRISP-DM.

Post-development *processes* are the processes that are enacted after the software has been built. The *installation* process implies the transportation and installation of a system from the development environment to the operating environment. The *operation and support* process involves system operation by the user. Support includes technical assistance, user queries and support request entry in the support request log. This process can start up the *maintenance* process that provides feedback information to the software life cycle and leads to changes. Finally, the retirement process is the *retirement* of an existing system by withdrawing it from operation. The knowledge gathered in DM projects should be passed on to the user and installed either as pure knowledge or integrated into the client organization’s software system for use. The *operation and support* process is necessary to validate the results and how they are interpreted by the client in a real environment in the same way as the *maintenance* process is required to update models obtained or to discover which of the gathered knowledge is erroneous or invalid when new data are entered. This can lead to backtracking in the global process in order to select new attributes or techniques not considered before. As regards *retirement*, DM models also have a period of validity, as if the data profiles change, the models will also change and will no longer be valid. CRISP-DM neither satisfactorily nor completely covers any of the above processes, despite their importance.

4.4 Integral processes

Integral processes are necessary to successfully complete the project. They are enacted simultaneously to development processes and include activities that are unrelated to development. They are used to assure the completeness and quality of the project functions. The *evaluation* processes are used to discover defects in the product or in the process used to develop the project. This process covers the performance of all the verification tasks to assure that all the requirements are satisfied. The *configuration management* process identifies the structure of a system at a given time in the life cycle (called system configuration). Its goal is to control system changes and maintain system coherence and traceability. On the other hand, the *documentation development* process is the set of activities that produce, distribute and maintain the documents developers and users require. Finally, the *training* process includes the development of training

programs for staff and clients and the preparation of proper training materials. The *documentation development* process for DM projects will be almost the same as for SE, but changes should be made to how the *evaluation* process is done. The *configuration management* process is an especially important CRISP-DM omission, as mentioned earlier. This process is considered absolutely necessary, because one or more people developing a DM project generate a great many versions of input data, models and documents, etc. If these versions are not properly organized by means of configuration management, it is very difficult to return to previous models if it is necessary. We believe that any new DM process model should account for the *training* process, making a distinction between data miner training and user training. To be able to repeat the process enacted in the project or properly interpret the results when new data become available, users sometimes need to be trained in DM.

5 A process model for Data Mining engineering

Having compared CRISP-DM with a SE process model, we find that many of the processes defined in SE and that are very important for developing any type of DM engineering project are missing from CRISP-DM. What we propose is to take the tasks and processes that CRISP-DM includes and organize them by processes similar to those covered in SE and add what we consider to be key development activities. The activities missing from CRISP-DM are primarily *project management* processes, *integral* processes and *organizational* processes. Figure 2 shows an overview of the proposed process model[3], including subprocesses. *KDD process* is the core of *development*.

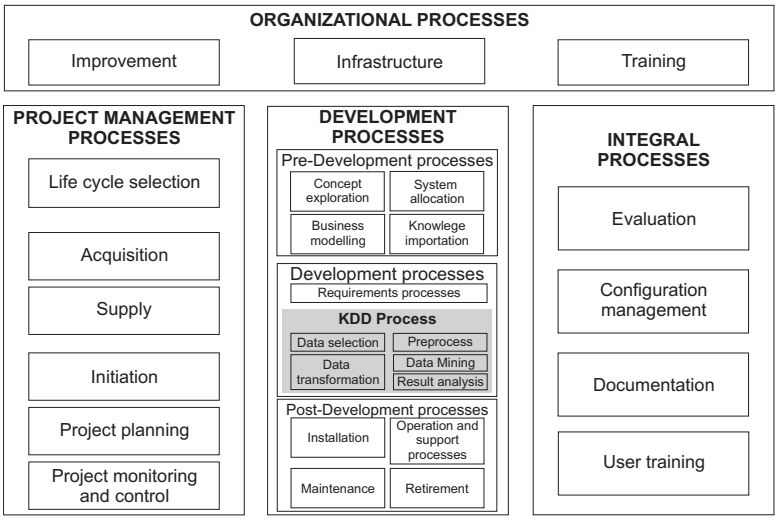

Fig. 2. Data Mining engineering process model

[3] This work was conducted as part of the CYCIT-funded project no. TIN2004-05873.

6 Conclusions

After analysing SE standards, we developed a joint model that we used to compare SE and DM procedures process by process and activity by activity. This comparison highlighted that CRISP-DM either fails to address many tasks related to management, organization and project quality at all or, at least, in enough detail to be able to deal with the complexity of projects now under development. These projects tend to involve not only the study of large volumes of data but also the management and organization of large interdisciplinary human teams. As a result, we proposed a process model for DM engineering that covers those aspects, making a distinction between what is a process model from what is a methodology and life cycle. The proposed process model includes all the activities covered in CRISP-DM, but spread across process groups according to more comprehensive and advanced standards of a better established branch of engineering with over 40 years of experience: SE. The model is not complete, as this paper merely states the need for the subprocesses and especially the activities set out in IEEE Std 1074 or ISO 12207 but missing in CRISP-DM.

References

1. P. Naur and B. Randell. Software engineering: Report on NATO conference. 1969.
2. G. Piatetsky-Shaphiro and W. Frawley. *Knowledge Discovery in Databases*. AAAI/MIT Press, MA, 1991.
3. P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, and R. Wirth. CRISP-DM 1.0 step-by-step data mining guide. Technical report, CRISP-DM, 2000.
4. B. Eisenfeld, E. Kolsky, and T. Topolinski. 42 percent of CRM software goes unused. www.gartner.com, February 2003.
5. B. Eisenfeld, E. Kolsky, T. Topolinski, D. Hagemeyer, and J. Grigg. Unused CRM software increases TCO and decreases ROI. www.gartner.com, Febrero 2003.
6. A. Zornes. The top 5 global 3000 data mining trends for 2003/04. *META Group Research-Delta Summary*, 2061, March 2003.
7. H. A. Edelstein and H. C. Edelstein. *Building, Using, and Managing the Data Warehouse*. Data Warehousing Institute. Prentice Hall PTR, 1st edition, 1997.
8. M. Strand. *The Business Value of Data Warehouses - Opportunities, Pitfalls and Future Directions*. PhD thesis, University of Skövde, December 2000.
9. J.E. Gondar. *Metodología Del Data Mining*. Data Mining Institute, S.L., 2005.
10. R. Pressman. *Software Engineering: A Practitioner's Approach*. McGraw-Hill, 2005.
11. J. Moore. *Software Engineering Standards: A User´s Road Map*. IEEE, CA, 1998.
12. U. Fayyad, G. Piatetsky-Shapiro, P. Smith, and R. Uthurusamy. *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, MA, 1996.
13. Two Crows Corp. *Introduction to Data Mining and Knowledge Discovery*. 3rd edition, 1999.
14. SAS Institute. SEMMA data mining methodology. http://www.sas.com, 2005.
15. F.J. Martínez de Pisón. *Optimización Mediante Técnicas de Minería de Datos Del Ciclo de Recocido de Una Línea de Galvanizado*. PhD thesis, Universidad de La Rioja, 2003.
16. J. Solarte. A proposed data mining methodoloy and its aplication to industrial engineering. Master's thesis, University of Tennessee, Knoxville, 2002.
17. IEEE. *Standard for Developing Software Life Cycle Processes. IEEE Std. 1074-1997*. IEEE Computer Society, Nueva York (EE.UU.), 1991.
18. ISO. *ISO/IEC Standard 12207:1995. Software Life Cycle Processes*. International Organization for Standarization, Ginebra (Suiza), 1995.