WORLD
INTELLECTUAL
PROPERTY
ORGANIZATION

IPAS JAVA

Functional and Technical Overview

Revision G

INFRASTRUCTURE MODERNIZATION DIVISION

Table of Contents

1 Overview

The Industrial Property Automation System (IPAS) of the World Intellectual Property Organization (WIPO) is a flexible, modular system that can be customized to individual industrial property (IP) offices to automate their IP business and administrative processes from application reception to registration, including post-registration actions such as amendments, assignment, renewal, annuities, etc.

IPAS and its new web-based version IPAS Java are developed by the Infrastructure Modernization Division of WIPO. IPAS is one of the key components of the global IP infrastructure that is used by WIPO, along with a set of modernization services, to offer a comprehensive sustainable automation solution to requesting IP offices from developing countries with diverse levels of development, resources, capacity, skills and infrastructure.

1.1 Objective

This document contains a functional and technical overview of the IPAS Java system developed by the Infrastructure Modernization Division of WIPO. The description of the system functionalities is presented from a conceptual point of view and its purpose is to give an overall idea of the capabilities without getting into its in-depth and complex technical aspects. In a similar way, the technical description is meant to give an insight of the technology used for developing of the IPAS Java.

Before getting into the functional and technical overview of IPAS, this document presents other aspects that the IP offices need to be aware of about the overall modernization process and what it takes to derive optimum benefit from it on a sustainable basis.

1.2 Modernization Process

Modernization is an ongoing process and it involves several components. It is important to understand that IPAS Java is only one component, though a critical one, in the suite of services provided by WIPO to assist IP offices in automation of their IP operations and enable them to provide improved services to their IP stakeholders. By itself, IPAS is not enough to achieve this. It is only a tool. There is a combination of other elements such as impacting factors, WIPO modernization services and expected contribution from recipient IP offices that affects its successful implementation and subsequent sustainability. Some of these aspects are listed below.

1.2.1 Impacting Factors

- National e-Strategy / e-Government initiatives
- Established and in force IP Legislation & Regulation
- IP Institution Structure & Organization
- Management Commitment & Involvement
- Strategic Plan & Approach
- Established business Processes as per Rules and Regulations
- Adequate Staff Profiles (skills & competencies)
- Requirements of IP Stakeholders
- Available, in-house IT skills and expertise
- Adequate IT infrastructure
- Procedures to minimize loss of knowledge when staff leaves the office

1.2.2 Modernization services offered by WIPO

- Technical consultancy, advise and guidance
- Needs assessment
- Streamlining of business process (business process re-engineering)
- Customization of IPAS to adapt to individual national requirements
- Establishment of IP databases (trademarks, patents, industrial designs)
- Upgrade of IT infrastructure (case-by-case basis, specially for LDCs)
- Phased deployment of IPAS
- Phased training of IP office staff and knowledge transfer to IT focal point(s)
- Technical support and problem resolution, online or on-site, as appropriate
- Post-implementation impact assessment with IP office
- Provision of new releases of IPAS as they are ready (over the Internet)

1.2.3 Expected contribution from the IP office

A WIPO assisted modernization project is essentially a joint IPO-WIPO partnership activity with well defined roles and responsibilities for both parties. On WIPO's part, its role and responsibilities are outlined in the set of Modernization Services listed above. They represent WIPO's contribution to the project. For the project to be successfully implemented and for it to achieve its desired objective, the recipient IP office is expected to contribute toward it with their part of the required elements. Some of these are listed below:

- Resources for capturing relatively large volume of IP data from paper records
- Maintain accurate, complete and validated data in IP databases
- Close collaboration during the customization and verification process
- Adequate network and Internet infrastructure to support automation
- IT staff with adequate skills to support automation
- Transition management from existing to new automated procedures
- Elimination/minimization of parallel manual procedures
- Resources for ongoing sustainability of automation systems and infrastructure
- Final ownership and self-reliance

1.3 IPAS Highlights

IPAS and its new web-based version IPAS Java (referred simply as IPAS) have been developed over the years by the Infrastructure Modernization Division (IMD) of WIPO. It is a result of IMD's collective knowledge and experience acquired over the last several years of providing modernization assistance to IP offices across all regions and with wide diverse profiles.

1.3.1 Frequently asked questions about IPAS

- **Ownership?** IPAS source code and copyright belongs to WIPO.
- **Flexibility?** Its modular design can be scaled up or down in terms of IP office size, IP legislation, functionality, workflow complexity, IP data volumes and any other national IP requirements.
- **What IP titles it can automate**? IPAS is an integrated IP administration system that can automate the processing of trademarks, patents and industrial designs. Utility models or petit patents are also covered.

- **What can be customized**? Extensive customization including: national IP legislation, workflow processes, actions & statuses, legal time-periods and deadlines, templates of official correspondences, certificates, gazettes, language, calendar support, data migration from legacy databases. No software programming is required in IPAS at customization!
- **Is it Open Source**? IPAS has rather an open architecture and is developed on the Java technology platform. IPAS itself is a proprietary product of WIPO.
- **Training and technical support**? WIPO provides training and knowledge transfer to IPO staff to increase their self-reliance. It also provides online or on-site support to resolve any system related problems, as required.
- **Future enhancements**? Ongoing enhancement of IPAS is an integral part of the IP Office Modernization Division. IPAS is continually improved in functionality, based on feedback from IP offices, and kept up-to-date to take advantage of new emerging technologies. New releases of IPAS are provided online via the Internet.
- **How much does it cost**? As it is owned by WIPO, IPAS is offered at no cost to requesting IP offices to develop and improve their capacity and efficiency of IP registration activities and to enable them to better serve their IP stakeholders. This includes training, technical support and new releases as appropriate.
- **How long does it take to implement IPAS**? IPAS functionality is ready to use but it needs three sets of information from an IP office before its deployment.
  1) Complete, accurate data of national trademarks, patents, designs as required through data capture of paper records or data migration from legacy systems.
  2) Knowledge of IP office's workflow processes as per national legislation, rules and regulations.
  3) Prepared Word templates of official forms, correspondences, notifications, certificates, publications, gazettes, etc.
  These tasks, specially for the database contents, are time consuming and sometimes require significant resources from the IP office. From experience, it can be anywhere from 6 months to 1 year for well prepared offices or longer for others.
- **Compliance with Standards**? IPAS complies with WIPO and industry standards. For example, International Classifications (Nice, Vienna, Locarno, IPC) and other Standards relating to trademarks, patents and industrial designs wherever applicable.
- **Interoperability**? IPAS can be interfaced with electronic document management systems for e-dossier functionality. IPAS can also export data that can be used by other non-IP systems of the office (e.g. back-office admin systems, accounting, etc.)
- **How many countries use IPAS**? As of May 2009, IPAS is operational in about 38 IP offices across all regions to automate their day-to-day IP processing. Though these IP offices have the same standard IPAS functionality their level of usage varies depending upon their own challenges and capacities.

2 IPAS Java functionality

IPAS Java is an information system designed to streamline the processing of trademark, patent and industrial design applications, and later modification of registers, covering most of the operations side of the work done by an IP office.

IPAS Java customization features enable its adaptation to different legislations and Industrial Property Offices administrative procedures and practices. Most of the official documents generated by the IP office can be automatically generated by the system as part of the workflow, e.g. requests from the examiner to correct the application, publication of the journal, certificate of registration, certificate of renewal, certificate of change of name, etc.

2.1 Register of trademarks

The data that can be recorded for trademark applications and registrations is basically:

- Filing number (as discusses below) and date.
- Registration number and date, as well as the validity period for the registered trademark.
- Trademark data: name, translation to local language if required.
- Trademark logo (as a TIF, JPG or BMP image) and its Vienna classes.
- Sound of sound trademarks, to be played back as a WAV file.
- Limitation data: disclaimer, series of trademarks, associations with other similar trademarks of the same owner so that assignment can only be accepted as a group, etc.
- List of Paris priorities and possible exhibition date. Each priority has an “accepted” flag, and the system keeps track or the earliest accepted Paris priority to compute the “novelty date” used in searches.
- List of each owner (name, nationality, legal nature, contact information, external identification numbers and identification as per some national register of companies or individuals) and its residence address (street, zone, city, state and country). Other features related to owners are as follows:
    - The external identification numbers can be customized, e.g. tax id number, passport number, identity document number, etc.
    - It is possible to also code the states and the cities, e.g. for the national country, so as to allow detailed statistics related to the geographical distribution of the IP office’s clients.
    - Some persons can be associated to “groups” so as to record the dependency existing in clusters of companies, or in state-owner companies, etc. This is only used for special statistics.
- Representative (similar data as the owner). The type of the representative can be configured also, indicating the possibly different types of empowerment vested, e.g. temporary representative still lacking formal power, individual representative, agent, etc. Additional features related to the representative are:
    - A register of agents is available, where details of the agent and the persons working for him are recorded.
    - A Power Of Attorney register is also available, where powers can be recorded in advance and then the registration number of such power can be indicated in subsequent filings, without need to supply all the formal paperwork required for each authorization.

- Relationships between the trademark and other predecessor or successor trademarks as per some existing relationship. The relationship types can be fully configured, indicating its name, to which application types/subtypes it applies, whether it is required and whether it is functional.

  E.g. if the country accepts that one trademark can be divided into many "divisional" applications, such a relationship type can be configured so that the system enforces that one "divisional" application must be associated with one and only one predecessor "divided" trademark, etc.
- Nice classes and its list of products. Other features related to Nice classes are as follows:
  - In cases where multiple Nice classes are used but no divisional applications are allowed, it is possible that some classes are registered and others are rejected, so the system supports a kind of "status" for each Nice class.
  - For search purposes, the definition of a Nice class can be mapped to other classes where related products or services can be found. This relationship, which is not necessary symmetrical, can be optionally used for performing searches not only over the protected classes but over an extended set of related classes where the associated products could belong.
  - In order to simplify data capture, the complete list of goods/products can be used to automatically initialize the class protection, and subsequently the data entry clerk can edit this list to make the required adjustments.
- If national classes are still used, they can also be recorded. Configuring the mapping from each national class to the corresponding Nice classes is supported, so that trademarks protecting "national" classes can be used in searches against trademarks protecting "international" classes.
- Contract terms regulating a collective trademark.
- Law regulating the processing of the trademark. The law for each trademark is defaulted based on the filing date and the configured validity period for the law, but can be updated to reflect decisions made by the applicant. E.g. upon a change of law, the applicant may decide that his old application shall be processed as per the new law, etc. The law has impact on the workflow of the trademark, limiting certain actions that can be taken (see workflow module).

Numbering of trademarks is very flexible, using a combination of:

- A basic file number.
- A series that can be either annual or fixed, e.g. some countries use annual series, others use fixed series that never change or that change to indicate major changes in the underlying legislation.
- A sequence which depends on both:
  - The application type, e.g. some countries use a separate sequence for marks, geographical denominations, etc, while others use a single sequence for all the application types.
  - The reception place of the application, e.g. some large offices use a separate sequence for each regional office where reception can take place, while others use a single sequence for all reception places.

2.1.1 Trademarks searching

Apart from the bibliographic searching, a specialized trademark similarity search module offers the examiner the possibility to detect prior similar trademarks, either as phonetic or logo similarities. Similarities found by any means can be selected by the examiner and formatted so as to be included for the production of office documents, e.g. the examiner's report.

2.1.1.1 Bibliographic search

Retrieval of trademarks can be done on the main bibliographic items:

- File and registration numbers and dates.
- Words contained within the trademark name, either complete words, prefixes, suffixes or other generic words. This search ignores the case and the possible accentuation of the trademark, e.g. *L'ÔREAL* will be retrieved if searching is simply done by *OREAL*.
- Words contained within the owner or representative name, with the same features described above.
- Nice classes.
- etc.

2.1.1.2 Phonetic search

Phonetic search uses an algorithm that computes an estimated similarity percentage between a new trademark and those already in the register. There are three variations:

- Interactive search: the trademark and list of Nice classes are captured on screen, and the results are shown on screen or printed. The search universe is the complete trademarks database at the search date.
- Preliminary search: a prospective trademark and list of Nice classes are captured following a preliminary search application, and the system produces a report with a list of potential similarities, which is delivered to the user so he can make the decision whether to file a new application or not. As in the previous case, the search universe is the complete database at the date of search.
- New applications search: all new applications are searched for at the time its processing gets to a certain status in the workflow, and the results can be printed or stored in the database as prior trademarks for later analysis. The universe for searching is all trademarks presented before or on the same date as the one examined.

The phonetic search is based on a phonetisation algorithm that can be configured without requiring changes in the program, by defining in an external XML file the desired set of rules to be applied. The rules are used for two purposes:

- To detect and separating common prefixes and suffixes in the words of trademark, e.g. in the trademark NETWORK the prefix NET can be isolated and therefore treated as if the trademark were *NET WORK*, which yields much better results.
- To perform the actual transformation of the trademark name into a phonetic equivalent. Currently two set of rules are available for this: rules following the Spanish-English phonetics, and rules following the Portuguese-English phonetics, each set composed of more than 110 rules.

As part of the configuration of the system, both set of rules (used for prefix/suffix analysis and for phonetisation itself) can be defined. Upon changing some of the rules, the system performs an automatic re-phonetisation of all trademarks.

These phonetisation rules are expressed using a syntax similar to the regular expressions supported by Perl or Java, and can take into account not only the letters to be replaced but neighbor letters that define how that specific case should be pronounced. The rules are stored in external XML files, and the phonetic module has tools to allow refinement of those rules to follow local needs.

2.1.1.3 Logo search

The other searching tool available in IPAS Java is the logo search using the Vienna classification codes, which shows results on screen and thus allows the examiner to visually detect similarities and select relevant trademarks.

This tool is interactive in the sense that the system selects trademarks having at least one Vienna code and Nice class in common to the search criteria, and also using the maximum filing date. Then the examiner must visualize the results and select those he considers appropriate.

The logo search can be triggered from a trademark view page, and the Vienna and Nice classes of such trademarks are used as search criteria. The selected logos can then be stored as similarities of the reference trademark.

2.2 Register of patents, utility models and designs

The data that can be recorded for patents applications and registrations is:

- Filing number and date (similar as for trademarks).
- Registration number and date, as well as the validity period for the registered patents.
- Title, abstract, up to 99 claims, and up to 20 drawings. If the national language is not English, an English translation of all the technical items is supported (title, abstract, claims, etc.).
- IPC classes (for patents and utility models).
- Local technical classes. Some countries require statistics based on local "technical classes", and so a "mapping" functionality is available to automatically transform the main IPC for the patent into one "technical class" based on the following:
    - Any number of "types of technical classes" can be configured.
    - For each type, any number of "technical classes" can be defined.
    - For each technical class, any number of IPC mapping can be defined, where each mapping means that certain sub-tree of the IPC must be mapped to that technical class.
    - To obtain the technical class for a patent, the main IPC of the patent is read, the IPC tree is traversed from the leaves upwards until a node is found where a mapping of such IPC sub-tree has been configured. In this way, whole branches of the IPC can be mapped to one technical class, while still allowing exceptions so that part of that branch can be mapped to other technical class.
- Locarno classes (for industrial designs).
- List of Paris priorities and possible exhibition date.
- PCT application and publication data.
- List of owners (similar as for trademarks).
- Representative (similar as for trademarks).
- List of inventors (similar as for owners).
- Relationships between the patent and other predecessor or successor patents as per some existing relationship (similar as for trademarks). Many more relationships normally exist for patents than for trademarks (e.g. division, enhancement, transformation of patent into utility model, etc.) and all these relationship types can be configured.
- The examination result produced by the examiner can be recorded also, detailing the search fields used, the search strategy used, the list of relevant documents as per WIPO standards, whether each of the legal requisites for granting the patent are fulfilled and the final recommendation.
- Support is provided for importing amino acid and other sequences according to WIPO standard ST.25. No search on sequences similarities is supported, just the display of the sequences loaded.

- Applicable law (similar as for trademarks).
- Annuities due are generated automatically when certain stage in the workflow is achieved, based on a set of flexible configuration parameters:
    - Number of validity years.
    - Due date of each annuity based on the yearly period covered: in some countries annuities are due at the beginning of the validity year, while in others this is not the case.
    - Number of annuities that should be ignored: some countries do not charge a fixed number of initial annuities, while others only charge those annuities that are not due at the time the registration of the patent takes place, etc.

  Through the reception of the annual payments, each one can be manually checked as "paid". Please note that the system does not automatically mark an annuity as paid, since the flexibility required to configure penalties and interests for late payment can be very complex in some countries.

  If unpaid annuities exist after their due date, the workflow module can automatically generate a process to analyze possible cancellation of the patent (see details in the description of the workflow).

Retrieval of patents can be done on the main bibliographic items, in a similar way as described for trademarks.

2.2.1 Patents searching

A technical search is available supporting the logical combination by AND/OR of complex conditions such as:

- By one or more words within the applicant name.
- By one or more words within the inventor name.
- By the country and the priority number ignoring special characters, e.g. "97/755" is equivalent to "97755".
- By one or more IPC codes at any level of detail combined with AND/OR.
- By one or more words within the title combined with AND/OR.
- By one or more words within the summary combined with AND/OR.

2.3 Register of other documents

Apart from the applications for new IP rights (e.g. trademarks and patents), other documents can be managed by the system, e.g.:

- Documents voluntarily filed by the applicant and aimed at altering some existing IP right, e.g. an assignment, a change of name, a renewal, a withdrawal, a license, etc.
- Documents filed by the applicant as a response to some request previously made by the office, e.g. the response to a letter of objections, etc.
- Documents filed by a third party who is against the applied IP right, e.g. an opposition, a request to cancel a trademark for non use, etc.
- Documents filed by a Court of Justice ordering the office to perform a certain action, e.g. to cancel a trademark, or to prevent possible changes in ownership in some trademark, etc.
- Requests of information by the public, e.g. report of trademarks similar to a prospective name.

A high degree of flexibility is offered to configure these documents, and some of the functionalities that IPAS Java uses to manage such wide range of possibilities are:

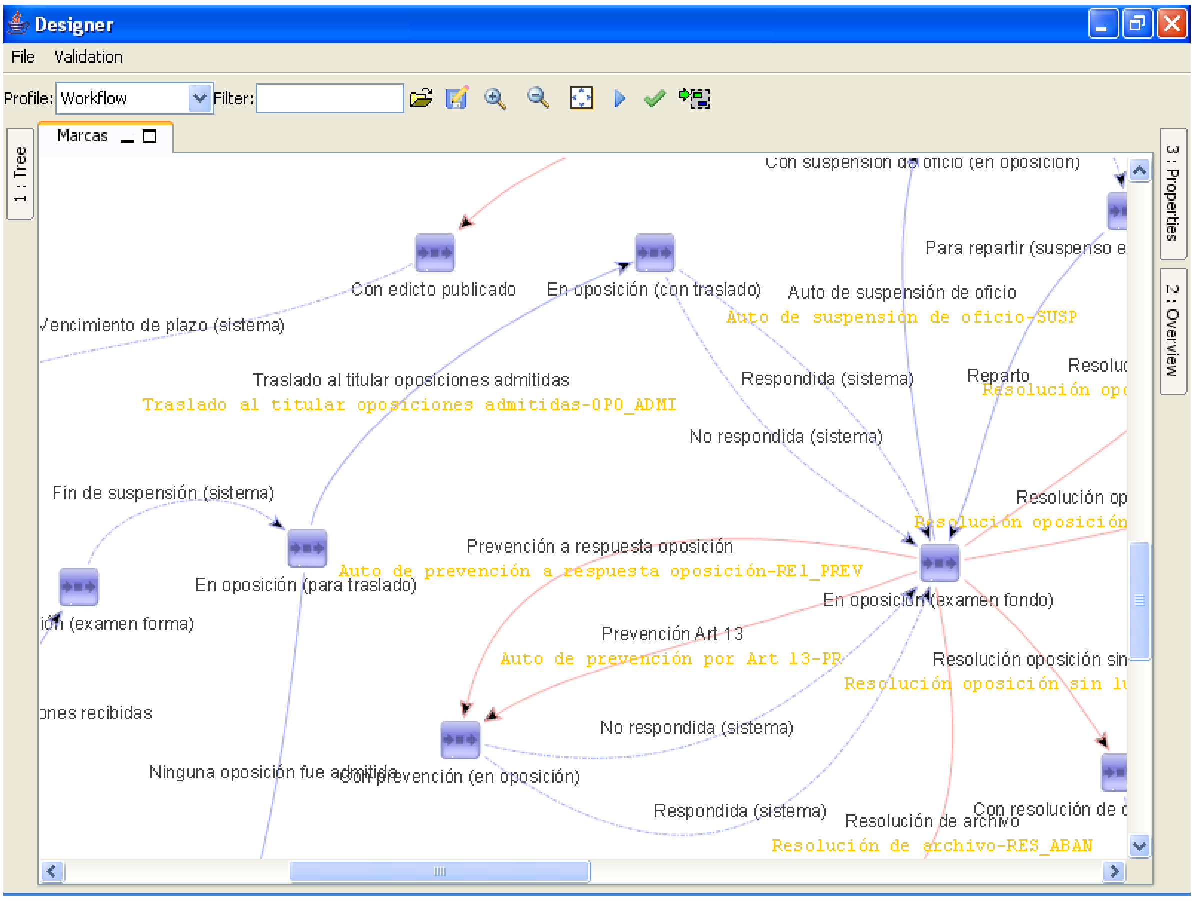
- All documents are identified by a correlative reception number within the year, and optionally an alternative numbering can be configured, assigning one or many document types to a "document sequence" which will generate numbers for those documents. E.g. all oppositions could be numbered as one sequence, all documents related to changes of name (assignments, change of the legal nature of the company, etc) could be assigned to a separate sequence, etc.
- Each document type can be configured so that it affects only one file (e.g. a request to withdraw a trademark), many files (e.g. normally the offices allow that documents changing the ownership can affect many files) or no files at all (e.g. a request by a Court asking for the trademarks registered in favor of a specific person).
- When the document type has been configured as being filed by a third party, details of such person can be entered.
- When the document type has been configured as being filed by a Court of Justice, details of such Court can be entered, as well as the case being dealt with by the Court, and the reference in such case to the request performed upon the IP office.
- Some document types can be configured so that they affect some "auxiliary register", e.g. an auxiliary register type called "licenses" could be configured along with the different document types aimed at:
    - Registering a new license in association with a trademark or patent. The identification of this document will be used to identify such license.
    - Renewing the validity period of an existing license.
    - Altering the license.
    - Canceling the license.

  Then the system will be able, upon approval of the respective document by the IP office, to automatically insert / modify / cancel the auxiliary register of "licenses".
- In case of documents aimed at altering the ownership or representatives, the system will allow to enter the new owners or representatives to be entered into the Register. Then, upon approval of the document, the system will be able to automatically update the ownership or representative information of the affected file.

  This feature is particularly valuable when many files are affected, e.g. the merger of companies A (owning 100 trademarks) and B (owning 80 trademarks) into new company C, would affect a change of ownership for 180 trademarks and the proposed approach would allow to enter the new ownership data only once, and then the system would automatically update the owner of each of the affected 180 trademarks (see details in "updating the Register from the workflow").
- Some document types can be configured as being a renewal for the affected file. In this case, its approval will trigger the extension of the validity period of the trademark or patent for a new period, according with the configured behavior. E.g. it can be configured that trademarks renew for period of time equal to the original validity and without any limitation in the number of renewals, while designs renew for a reduced period of time and only for 2 renewals as a maximum, while patents do not renew at all, etc.

2.4 Workflow

The IPAS Java workflow module allows the definition of the actions carried out by the office for each application and user document received. Each action produces a change in the status of the process, and the status transition diagram defining the structure of each workflow can be configured by using the interactive graphical tool called "IPAS Java Designer" (a sample printout follows):


Designer
File
Validation
Profile: Workflow
Filter:
1 : Tree
Marcas
3 : Properties
2 : Overview
Con suspension de oficio (en oposicion)
Para repartir (suspenso e
Con edicto publicado
En oposición (con traslado)
Auto de suspensión de oficio
Auto de suspension de oficio-SUSP
Vencimiento de plazo (sistema)
Traslado al titular oposiciones admitidas
Traslado al titular oposiciones admitidas-OPO_ADMI
Respondida (sistema)
Reparto
Resoluc
Resolución opo
No respondida (sistema)
Fin de suspensión (sistema)
Resolución op
Resolución oposición
Prevención a respuesta oposición
Auto de prevencion a respuesta oposición-RE1_PREV
En oposición (para traslado)
En oposición (examen fondo)
ión (examen forma)
Prevención Art 13
Auto de prevención por Art 13-PR
Resolución oposición sin
Resolución oposición sin l
ones recibidas
No respondida (sistema)
Ninguna oposición fue admitida
Con prevención (en oposición)
Respondida (sistema)
Resolución de archivo
Con resolución de c
Resolución de archivo-RES_ABAN

The workflow module is one of IPAS Java key features, and its customization features make it a flexible modeling tool for virtually any situation that could be found in an IP office. In this sense, IPAS Java is not tied to any specific legislation or procedure. Some of the key features are discussed below.

The main entities involved in the workflow are the following:

- The *process type* allows modeling the status transition diagram, composed by *statuses* which indicate one particular activity that needs to be carried out, and by *actions* connecting those statuses, which in turn indicate the possible outcomes of the activity carried out in such status. E.g. in the previous sample, the status “Controlling registration fee” indicates that the activity to be carried out is controlling whether the registration fee has been paid or not, and that the possible outcomes of this activity are:
    - Action “Registration fee paid”, indicating that the fee was already paid, and so the next status will be “To be registered”.
    - Action “Request for registration fee”, indicating that the fee has not been paid yet, and so an office document must be printed (called “Request for registration fee”) and the next status will be “Awaiting registration fee”.
- A *process* is the unity of workflow control (e.g. a trademark, a patent, etc.) and it is associated to one process type, which in turn defines its possible statuses along time and also which are the possible actions that can be recorded at each status.
- Once a process is identified, the system will display its current status, a history of all the actions that have led to such status, etc. It will also allow entering a new action, which must be one of the “outgoing arrows” in the diagram, connecting the current status to the target one.

- Any action can be configured so that the system generates and prints an *office document* with information automatically extracted from the action and the affected process (see detailed description below).
- A process is generated each time IPAS Java needs to perform a follow up of a certain activity. The events that generate processes are as follows:
  - When a new application is received, a process dedicated to controlling its registration is generated, using a process type which is configured from the application type. This means that different application types (like trademarks and patents) could use a different state transition diagram, while others (like patents and utility models) could use the same.
  - When a new document affecting an application is received, a process dedicated to controlling its approval is generated. This process is associated to both the affected file and the document, and uses a process type which is configured in the document type. This means that different document types (like oppositions and assignments) could use a different state transition diagram, while others (like assignments and changes of name) could use the same.
  - When a new document not affecting an application is received, a process dedicated to controlling its approval is generated, which it is associated just to the document, as uses a process type which is configured in the document type. E.g. a preliminary search request could generate a process with a specific workflow structure.
  - The recording of an action configured so that an office document should be printed will also generate a process to control the activities required for such office document. E.g. if the office document is just an internal report that just needs to be printed, its workflow structure will be very simple. But for other office documents, like requests that must be notified to the applicant and responded by him, a much more complex process type could be required, including all the steps required to perform the notification, etc.
  - The system can automatically generate processes to perform certain controls that can be done in parallel to the normal workflow, e.g.:
    - When unpaid annuities are detected by the system after their due date, IPAS Java can generate an auxiliary process using a configured process type, associated with the patent but working in parallel to its main process, and dedicated to analyze the possible need to cancel the patent for non-payment.
    - Upon entry to a certain status, IPAS Java can generate an auxiliary process of a configured type. E.g. when a patent enters the “formality examination” status, an auxiliary process for controlling the accordance of its main IPC classification (for publication purposes) could be triggered by the system; this activity could be processed in parallel to the main process of the patent, so that while the formality checking is being done, the substance examiners could be assigning the main IPC code that will be required at publication time.
- As can be seen, the association of a process with the controlled activity is very flexible. This allows that the same features be available for all kinds of processes, from the complex ones like those used to control the registration of a trademark, to the very simple ones like those used to control the processing of the payment of a fee.

Even though the same process type (e.g. the same workflow diagram) can be shared by many application types, it is possible to configure that certain actions can only be taken if the file has a specific law, or a specific application type or subtype. This enables that minor differences in the workflow could still be modeled in a single consolidated workflow with certain conditional parts, e.g. a certain area of the diagram can only be entered if the law controlling the file is a specific one, or that certain shortcuts could only be entered for designs and not for patents, etc. The possibility of using “conditional actions” must be balanced upon the configuration of separate workflows for each case, which is a cleaner approach but possibly requiring more configuration work.

Since each status denotes some activity to be done, it must be clear who the responsible user for the status is. In some cases, there are so many processes in that status that many users are required to deal with it (e.g. examination) and then the system can be configured so that a “responsible user” is required upon entry to such status. Each such user will be able to easily build a group with the processes that are assigned to him/her, which in effect is the pending work (see “process groups” below).

The responsibility for a process can also be assigned in an exclusive way, so that no other user can update the file.

Certain actions that represent formal decisions can be automatically numbered by the system, e.g. all actions indicating a decision taken by the Registrar. The numbering schema is very flexible, e.g. an annual series could be used, different action types could share the same sequence, etc. Even a separate record of such actions can be generated, so as to replace manual registers of the Registrar’s decisions normally kept.

Under certain circumstances, the process of a file could be affected by other related processed, in what is called “freezing” in IPAS Java jargon. The main effect of freezing is that when a new action is recorded in the “frozen” process while the “freezing” process has not yet arrived to a final status, then the system will produce a warning message reminding that such process is frozen by certain other pending process, and thus giving the examiner the chance to stop. For instance:

- If a request to add a disclaimer is so configured, while the document is still pending the system will produce a reminder warning message any time an action is recorded in the process of the affected trademark.
- In addition to what was described above, the “freezing” can be configured so that it even prevents the production of new office documents in the affected file. E.g. a “change of address” should be so configured so as to avoid the generation of any office documents while the change request has not been dealt with, since it doesn’t make much sense to send a letter to the “old address” (which the applicant has requested to changed) nor to the “new address” (until the office has officially recognized the new address as the valid one for the file).

In the above examples, the “freezing” is generated automatically by the system, but it is also possible to manually add a freezing. For example, when the examination of a trademark A yields a similar trademark B which is still pending, trademark A cannot be rejected since trademark B could eventually not get registered in the end, and a manually inserted freeze would do the job. The link between both processes is such that the “freezing” process B is able to “inform” the frozen process A that is has now ended, so that process A can continue via an action automatically generated by IPAS Java (this can be configured with the “end of freezing” automatic actions described later on).

2.4.1 Printing office documents from the workflow

When entering an administrative action, the user can capture additional information through a set of up to five notes fields, and also through lists of previously coded options. For example, the system can be configured so that when entering an action of "formal examination request" type, the user can select one or several options that represent pre-defined reasons for which requests are made, and also that any additional non-predefined reasons are captured as an additional text field. All this data will be available for inclusion in the office document that will be printed.

Some additional features increase the flexibility of the data items associated to an action:

§ The predefined lists offer the possibility that the texts to be printed could contain some placeholders for additional information to be entered by the examiner, for instance it the texts reads *The Priority Documents for the priority <{priority}> should be filed*, the examiner will be prompted for the replacement text to be used for the placeholder *<{priority}>* and the entered value will be merged with the rest of the fixed text to produce the final printed result.

- § Some option list entries can be enabled only if the file has certain application type, or certain law. E.g. the same list could be shared for the formality defects of patents and utility models, and those options that would only apply to one application type could be so configured.
- § Each option list entry can be enabled only for certain action types. E.g. the “formality report” and the “substance report” actions could share the same list of possible objections, and some entries could be common to both action types while other entries will only apply to the formality check and others to the substantive check.
- § In any place where text can be entered, the right mouse button can be clicked to display a context menu from where data associated with the file can be selected and pasted automatically at the cursor position. This includes data from the priorities of the file, from the documents that been received in relation to the file, and from the office documents that been generated within the process of the file. E.g. following the above example, the list of priorities could be displayed and one priority could be selected at a time, getting a text describing the priority and thus minimizing the typing effort and the possibility of errors. The layout of these texts can also be configured (see the description of “generating output fields”).

Most of the paperwork generated by the IP office can be automatically generated by the system. In effect, each type of action can be configured so that the system generates an office document with information concerning the action and the affected file. The layout of the office document is configured by the administrator using Microsoft Word and mail merge fields extracted from the database. The features available for the configuration of office documents include:

- § The MailMerge fields available can be configured using the features discussed in the “generating output fields” section. Basically, any data item stored in the database can be accessed via the appropriate SQL statement, and made available as a MailMerge field.
- § The very flexible functionalities of Word’s MailMerge are even further extended via some additional functionalities:
    - o Inclusion of pictures in the Word document, e.g. the logo for trademarks and the drawings for patents. This enables production of complex documents such as the certificates of registration, etc.
    - o Formatting of fields containing HTML data. This allows for instance the inclusion of HTML tables as if they were MailMerge fields. One specific case where HTML tables are very useful is for including in the trademark examiner’s report the list of the selected similarities.

It is important to note that IPAS Java performs the MailMerge functions and generates the output Word documents without using Microsoft Word at all, but a commercial Java library called Aspose.Words. More details can be found in the technology discussions below.

The system generates a sub-process to keep track of the actions needed to be carried out concerning each office document, and shows this sub-process (and its current status) within the affected application's actions log. There are basically three different process structures normally used for office documents:

- Just printing the document. The action is automatic, so in this case no action needs to be captured by the user.
- Printing the document and notifying it to the applicant.
- Printing the document, notifying it to the applicant and waiting until a reply is received from the applicant (or expiration of the allowed period has elapsed). In this case, the affected application's process is “frozen” until the office document's process is finished, and then it resumes (please refer to the “freezing” discussion at the beginning of the workflow discussions).

Office documents can be re-printed on demand, e.g. to respond telephone queries received from the applicant. This lessens the need to have the physical file available.

2.4.2 Automatic actions

Some of the actions in the workflow can be configured as "automatic", in the sense that they are generated by the system upon certain triggering conditions instead of being manually captured by the user. The main automatic actions are discussed below:

- The simplest kind of automatic action is the "date is due" action, which is triggered when the due date of a process has been reached. E.g. when the "Awaiting opposition period" is over, this automatic action could lead to a new status "Analyzing if oppositions were filed".

  In order to set the due date for a process, an action must be configured to compute it by adding a fixed number of working days, calendar days or calendar months to a base date that can be:

  - The date of the action.
  - The date of filing of the application.
  - The date of the first priority date approved.
  - The date of enforcement of the corresponding law.
  - A manually entered value.
  - The current due date of the process.
  - The notification date of the action, i.e. the date on which the applicant has been officially notified of the decision corresponding to the action. This date can be either:
    - The date of notification of the office document associated to the action (if applicable). The office document is considered "notified" when the status of its process gets to some status so configured in the corresponding workflow.
    - The date on which the journal containing such action (if applicable) is made available to the public.

  In order to better support working days, the non-working days can be defined in a calendar where it is also possible to configure the new due date to be used in case that non-working day is the target of some due date function. Normally this new due date will just be the next working day, but some IP offices stop working at year end for a couple of weeks and so they specify that the due dates during that period should be postponed over a range of days, so as to avoid that the first working day after the period holiday is the due date for a lot of cases.

  The actual triggering of the "date is due" automatic action is in fact postponed until the due date has been duly processed in the reception module, to make sure that all documents received up to that date are already reflected in the system. In this way, the absence of some document in the process history will be evidence enough that no such document has been filed up to the due date, which is far more conclusive than analyzing the physical file since the document might take some time to make its way since the reception desk to the physical file.

- To assure that all conditions required for performing searches are met, an automatic action is available that is triggered when all the required conditions to perform the search are met, i.e. when:
  - The file has all the search classifications assigned, e.g. Vienna for trademarks having logos, IPC for patents and Locarno for designs (this is controlled by the reception module).
  - The priority period configured for that application type has lapsed (normally 12 months for patents and 6 months for trademarks), so as to give time for possible subsequent applications that might be claiming a priority date earlier than the filing date of the file in process. In fact the actual triggering of the action is postponed until the search classes have been assigned to the applications received at the end of the priority period, e.g. if a patent is filed on Jan 15, 2006 then this action will not be

triggered until Jan 15, 2007 has been marked in the reception module so that all search classes are assigned, in prevision that a subsequent patent filed up to Jan 15, 2007 could be claiming a priority that would make its novelty date earlier than Jan 15, 2006.

This kind of automatic action could then be used before entering the substantive examination status, as a delay to assure that the search can be conducted appropriately. Of course the IP office need not postpone the examination until the end of the possible priority period, but if this is desired then the tools to automate it will be available.

- To wait until the applicant files certain document, an automatic action can be configured so as to be triggered the day a specific document is filed, or if a specific document has been already filed in the past. E.g. when a registration fee must be paid but normally the agents already pay that fee before it has been required, this kind of action could be used as a warning that the fee was already paid and so no request for payment should be sent.
- In the register of agents, the preferred notification mode can be configured, i.e. personal notification at the IP office of by mail, and so a couple of automatic actions are available to be used in the workflow of the notification process of each office document.
- In order to assure a leveled workload on all patent examiners, an automatic action for assigning the responsible user upon entry to the examination status can be used, which implements a specific procedure required by some IP offices: the main IPC of the file is read, the IPC tree is traversed from the leaves upwards until a node is found where a section for the examinations of such IPC sub-tree has been configured, then all the examiners working in such section are considered as candidates for received the responsibility for the new file, and the one which the least files assigned is the one selected.
- Certain application types (e.g. patents) can be configured so that publication is not performed until a certain period of time has lapsed from the filing date, so an automatic action is available to indicate that such period has lapsed. Moreover, when the applicant is entitled to request a special publication date (either earlier or later than the standard period), such request can be entered in the file, and the action will use the special date requested and not the standard date. Separate actions are available for the cases where a special publication date has been requested and for the normal cases, since some countries publish separate journal sections with both applications types, something like "normal publications" and "publications with a special period requested".
- When the expiration of the validity period is lapsed, an automatic action is available for transferring the process to some "Expired" status, or "Awaiting renewal upon grace period", or whatever is required.
- For the processing of trademark renewals, both the months for "early renewals" and "late renewals" can be configured, and together with the expiration date these dates define four periods of time (before proper time, in time, in grace period and after grace period). Four automatic actions types are available to control this, based on the date of reception of the renewal request, so that the examiner has already an indication of what kind of renewal he has in hands.
- When the "freezing" was discussed above, it was mentioned that in some cases the "frozen" process should just wait until the "freezing" process has arrived to a final status, e.g.:
    - When a trademark must wait for a non-registered similarity to get either registered or not registered.
    - When a process an office document requiring a response must wait until such office document gets responded or it is verified that no response was filed.

  The "no longer frozen" automatic action is triggered in the formerly frozen process when all the freezing processes (there could be many, e.g. when a trademark has several pending similarities) have arrived at statuses which are either final, or that have been configured as "end of freeze". This action will then serve as an indicator that processing can continue.

  In fact, many of such actions could be configured, for each of the possible endings of the frozen process. E.g. if the freezing office document with a request was finally responded, a “request responded” action could be triggered in the main file, and if the request was not responded, a “request not responded” action could be triggered instead.

- Under normal circumstances documents can be received from the user no matter what the status of the affected file might be, but in certain cases some degree of control is required. E.g. an “opposition” should be received only if the status of the file is “awaiting end of opposition period”. If the type of the user document has been configured appropriately, the system can trigger automatic actions in its process to indicate if the status of the affected file was the correct one or not, at the time of the reception. This will provide a clue to the examiner, who will eventually take the final decision whether to accept the opposition or to discard it.

It is important to notice that even though IPAS Java offers a wide range of automatic actions, normally it is not a good configuration strategy that these automatic actions trigger decisions (such as whether the application should be registered or not), but they should be just used as “intelligent reminders” so that the examiner (or whoever responsible) can be readily informed of some specific situation. Then the final decision should be taken by a human, using the system just as a source of automatic reminders.

2.4.3 Updating the Register from the workflow

Since the main purpose of the workflow is to control the registration of applications and the approval of documents, in case this stage is reached the system offers the possibility of updating the Register accordingly. The available features are as follows.

In the case of new IP right applications, when the process gets to a status configured as “registration”, the system generates a new registration for the corresponding application following these guidelines:

- The duration of the period of validity is configured for each application type/subtype and each possible law (thus allowing changes in the criteria under different laws), and so is the starting date of the validity period which can be:
    - Since the national filing date.
    - Since the international filing date.
    - Since the date that the decision to register was taken.
    - Since the expiration of the renewed register (this only applies if the application is a renewal of another registration).
    - Since a manually entered date.
- The numbering of the new registration is also configured for each application type/subtype and each possible law, as follows:
    - The same file series and number used for the filing can be used for the registration also. This implies that the registration numbers sequence will not have the same order than the registration dates, and that there will be missing numbers (since not all applications end in registrations).
    - Use a separate registration sequence, with or without a series.

In the case of user documents configured as renewals of registrations, when the process gets to a status configured as “document approval”, the system extends the period of validity of the affected registration, according to what is also configured for each application type/subtype and each possible law.

In the case of documents configured as affecting the ownership of the affected file (e.g. a change of name or an assignment), when the process gets to a status configured as “document approval”, the system replaces the current owners with the new ones, which must have been previously recorded as part of the bibliographic data of the document.

This feature is specially useful is the case the document requesting the change is affecting many files, since the new owners are only captured once, and the system will be able to update the ownership of all affected files in a single operation (see the details in the description of “process groups”).

In the case of documents configured as affecting the representatives of the affected file (e.g. an appointment of new agent), when the process gets to a status configured as “document approval”, the system replaces the current representative with the new one, which must have been previously recorded as part of the bibliographic data of the document.

In the case of processes for office documents, there are also two special situations where the system updates other information (not directly related to the Register) upon entry to a specific status:

- When an action generates an office document, until such document has been signed the action is considered just a draft, and the change of status produced by such action is not taken. Only when the workflow of the office document gets to a status configured as “authorization of office document”, meaning that the document was actually signed, the action originating the office document is considered complete.
- When the office document has been configured that it must be notified to the applicant, when the workflow of the office document gets to a status configured as “notification”, meaning that the document was actually notified, the action originating the office document is considered notified. This in turn triggers calculation of the expiration date if it has been configured to be computed since the notification date. Please note that in most legislations the time frame to reply to official requests is counted since the date of notification of the official request, and hence this feature is very useful.

The system also allows that a specific SQL statement is configured in a status, being executed upon process entry to such status. This can be considered similar as a “database trigger” but activated by the workflow activity. This feature is nevertheless seldom used.

2.4.4 Generating output fields

IPAS Java allows a very flexible approach to data extraction from the underlying database in order to use it for different purposes (printing of documents, exporting Excel files, etc). In summary, any data item stored in the database can be extracted by executing a set of SQL statements which are not hard-coded into IPAS Java code but configured by the administrator instead, so an administrator with SQL knowledge could easily create new output fields after understanding the IPAS Java database structure, which is documented in detail in a separate document.

Other features include:

- Many rows can be concatenated to yield to final result, e.g. a “list of owners” output field could be built by providing the SQL statement needed to generate each of the owner names, and initial, final and separator texts can be configured so that IPAS Java will concatenate each owner and get to a final result that could look like *SMITH, John A.; CONNEL, Brian W. and LING, Sung Q.*
- The “seed variables” that IPAS Java can load are basically those identifying any major entity in the database:
    - A file (trademark or patent).
    - A document.
    - An action.
    - An office document.
- IPAS Java offers a testing tool that can be used to check the output generated for each field prior to actually using it.
- Each SQL statement can be linked to one or more “usage codes” that will define when it should be executed.

Output fields are used for the following functionalities with IPAS Java:

- When an output document needs to be printed, all SQL statements associated with a usage code configured as “MailMerge” are executed, using as “seed variables” the identification of the generated output document, the identification of the action which generated the output document, the associated trademark or patent, and the associated document. This implies that virtually any relevant data item can be extracted by the output fields.
- When a data export from a process group is executed, a usage code is selected by the user and so IPAS Java informs what are the available output fields so as to select which ones will be the “columns” of the generated Excel spreadsheet.

Of course, the expressive power of the generated output fields is only limited by the power of SQL, so extremely complex constructions can be built. E.g. in some countries, the special legal wording required in some documents (e.g. certificates of registration) has lead to special output fields used only in those documents, where SQL is used as a programming language for encapsulating complex conditional formatting. And of course, all these complex output fields are just local to that country, and configured by the local administrator.

2.4.5 Journal/Gazette generation

Official IP office journal publication can also be automatically performed. The system can be configured so that some administrative action types should be included in specific sections of the office journal (e.g. registrations, changes order by the Court, etc.). For each action type to be included in the journal, the name of the Word file containing the template for the corresponding journal section can be defined, and this layout can be defined using the same flexible features as those available for printing office documents.

Then journal production consists in just defining the cut off date for the actions to be included, optionally selecting what actions types are to be included, and producing an output Word file for each section. These Word files can then be used as “raw material” for building the actual journal.

Apart from the Word files for each section, the system can also produce Word files that act as an index by class (Nice class for trademarks, IPC for patents and Locarno for designs) and by owner. This would allow generating the data in each section ordered by file number, and then producing these “indexes”, e.g. the index by IPC would indicate, for each IPC, all the file numbers that were published containing such IPC.

2.5 Examination tools

Some additional tools are available to support the examination process.

2.5.1 Examination search support

One of the activities in trademark examination is the search, and the system offers a set of tools to help organizing the work:

- The phonetic search requests need not be manually captured, but a specific action type in the workflow can be flagged so that the system will generate a search request at this time. Normally this action type is the one which indicates that the formalities phase has ended and now the trademark is entering the substantive examination phase.
- Assignment of the Vienna codes can be done without the physical file, since the logos are part of the bibliographic data and the system can locate all trademarks with a logo but without Vienna codes yet.

- When the requests for search are processed, it is possible to indicate to the system not to print a physical report with the results but to store those results in the database, where they can be seen in a “similarities list” as a supplement of the trademark’s bibliographic data.
- Under more complex scenarios, in order to assure that all conditions required for performing searches are met, an automatic action is available that is triggered when the file has the Vienna codes assigned and also when the priority period configured for that application type has lapsed (normally 6 months for trademarks). This action is the one that should be configured to trigger insertion of the search request.
- When the logo search for an application is performed, a “reference trademark” can be entered, and this enables the functionality to store the selected logos as part of the “similarity list” of the reference trademark.
- Potential similarities can be browsed and flagged as “selected” by the examiner, thus replacing the physical highlighting of some of the lines in the similarity report.
- Some of the similarities flagged as “selected” can also be flagged as “discard” by the examiner, if the owner is the same, the status is not active, the goods and services do not collide, or any other circumstance detected by the examiner.
- The remaining similarities, i.e. those flagged as “selected” nit not as “discarded” are the relevant one, and a MailMerge field can be configured so as to generate such list for inclusion in the examination report.

2.5.2 Pre-coded lists of options

When the result of the examination is a letter informing several defects in the application, the system be configured so as to display a list of pre-coded options from which the examiner may choose the relevant defect(s), and also capture any additional non-predefined reasons as an additional text field. All this data will be available for inclusion in the defect letter that will be printed.

The options in these lists offer have the possibility that the texts to be printed could contain some placeholders for additional information to be entered by the examiner, for instance it the texts reads *The Priority Documents for the priority <{priority}> should be filed*, the examiner will be prompted for the replacement text to be used for the placeholder *<{priority}>* and the entered value will be merged with the rest of the fixed text to produce the final printed result.

These pre-coded lists are fully configurable and provide the following main benefits:

- Allow consistent results for the different examiners since the texts have been already approved by the management, enforcing a specific policy.
- Allow greater productivity.
- Enable the production of statistics of the different defect types more commonly found in the applications, which can be useful for the design of training policies, etc.

2.5.3 Suspension of processing

Under some circumstances, e.g. when a similar prior application was found, processing of an application must be suspended until another application has been dealt with, either by registering it or by rejecting it. The system offers the functionality of “suspending” processing of an application until another one has been registered or rejected, and at that time an automatic “en of suspension” action will be triggered in the suspended application.

These mechanisms are fully configurable and eliminate the need to keep track of suspended applications using manual systems.

2.5.4 Process groups

In many cases, batches of files in the same administrative status are processed together, e.g. trademarks for which a registration certificate is being printed, trademarks that are accepted for publication in the same day, etc. While it is always possible to enter the corresponding administrative actions individually for each trademark, the system offers the possibility of building “groups” of processes that are in a similar status, and then the same action is applied to the whole group. Internally, the system applies one separate action for each member of the group, as if the user had taken the job to browse through all the members and repeatedly perform the same action on each one.

Each user defines his or her groups. Files can be included in a group by criteria designed so as to support the “to-do list” for each user, so that each user can load a process group with his pending work in much the same way as the “in/out trays” contain the to-do list of physical files to be processed. The most relevant criteria for selecting the processes to be added to a group are as follows:

- Add all processes in a particular status, and optionally assigned to a particular user (normally the connected one) as responsible.
- Add all processes having received an action of a specific type during a certain period of time, or not having received such action.
- Add all processes in another process group, belonging to the same or different owner.
- Add all processes related to the files (or documents) in a delivery list.

Files can be easily transferred within groups (by selecting elements and redirecting them to a second group). It is even possible to transfer data from groups belonging to one user to groups belonging to a different user, so as to continue processing in the next stage, without the need to physically send the papers to request further processing.

Apart from being used as a tool for entering actions for many processes at a time, process groups offer also other tools, e.g. data from a process group can be exported in two ways:

- Printing a report, using the same features as those available for producing output documents.
- Generating an Excel file, using columns from configurable “output fields” described in “generating output fields”.

2.6 Physical tracking

Physical transfer of files between office members is made easier and tracked in detail, by building delivery lists identifying the sending user, the receiving department and the set of files involved. The list is then printed and used as a cover letter for the physical group of files to be delivered, and finally the system automatically generates a delivery record in the file tracking log of each affected file.

A sort of “electronic signature” is possible for the receiving user to acknowledge reception of the delivery list.

Delivery lists can be used for files (trademarks or patents), received documents (changes of name, etc) and office documents (letters generated by the office).

2.7 Communication with the applicants

Interface with the applicant is provided by three means:

- Reception of documents.
- Notification of letters and other documents produced by the IP office.
- Public query tool.

2.7.1 Reception of documents

Reception of documents filed by the applicant is controlled on a daily basis using reception programs designed for interacting with the applicant at a reception desk and offering the following functionalities:

- Reception of applications for new IP rights and of other documents affecting existing IP rights (either pending or already registered).
- Generation of the filing number for applications for new IP rights and identification numbers for other documents.
- Printing of acknowledgement receipts replacing electromechanical clocks. The layout can be customized using Java formatting tools, or the layout of the receipt can be configured using Word templates in a similar way as with office documents.
- Payments received can also be recorded for each application or user document. Different receipt types are supported, e.g. receipts issued by the IP office and receipts issued by a Bank, and also different currency types are supported, with the possibility of recording exchange rates so as to generate consolidated reports in the local currency. Payment information can be subsequently exported to Excel spreadsheets by receipt number (to check against the physical receipt issued) of by document (to check that the correct fees have been paid), and so further statistical analysis can be performed.

Please note that the system is not capable of generating the receipts for the monies received, since this functionality normally requires software modules approved and audited by some local controller office, e.g. the Ministry of Finances, and is normally linked to the IP office's accounting module.

If the IP office is already using a reception module, a simple COM interface to the IPAS Java Application Server is offered so that the current programs can automatically trigger the reception of new documents in IPAS Java. This approach is used in some countries where the reception of the monies and of the documents is performed in the same transaction.

After the reception, subsequent data capture process is controlled by the following features:

- Since the system already knows all the file and document numbers received on a particular day, controlling the completeness of the data capture is possible. The system validates that every application and user document received during the day has been duly captured into the system, thus offering a "no holes" guarantee.
- Once the capture of the bibliographic data has been done, a similar control can be used to assure the complete scanning of logos and the capture of the classifications required for the search process (Vienna for trademarks having logos, IPC for patents and Locarno for designs).
- Reception triggers a new scanning request to be sent to the e-doc interface.

2.7.2 Notification of documents

Notification of documents produced by the IP office is controlled on a daily basis using notification programs designed for interacting with the applicant at a notification desk and offering the following functionalities:

- The notification procedure can be configured using the same flexible tools used for configuring the trademarks or patents workflow, e.g. many notification stages can be configured, etc.
- Tools for locating all documents pending to be notified to a certain agent are available. In this way, a massive notification can be performed.
- Notification triggers a new scanning request to be sent to the e-doc interface.

2.7.3 Public searching tool

A web query tool similar to that used in the IP office is available for the public, either on the Internet or on public computers on the Intranet.

A “captcha” mechanism, which displays a distorted image and prompts the user to enter the hidden letters, is used to prevent robotic access to the query tools. Configuration of the number of requests available after solving each “captcha” challenge can be configured.

2.8 Data import and export

The following features allow data import and export to and from IPAS Java.

2.8.1 Madrid international trademarks

New international trademarks under the Madrid agreement or protocol, and their subsequent transactions, can be imported from the electronic notifications files generated by WIPO, as per the following details:

- The system establishes an Internet connection with an FTP server managed by WIPO International Bureau which contains the weekly electronic notifications for international trademarks. The weekly notification files for Madrid transaction affecting the country are downloaded from this FTP site, containing the bibliographic data and the logos for new Madrid trademarks and also the details of all subsequent transactions.
- To start up the module, massive XML files containing the whole historic Madrid transactions for the country are especially produced by the International Bureau and used to feed a massive initial loading of information.
- IPAS uses transactions for new Madrid trademarks (BIRTH transactions) to insert a new trademark application, in a similar way as if it were a paper-less application received via e-filing mechanisms. The Madrid notification date is used as the filing date, and a configurable numbering mechanism is used so as to avoid collision with the numbers used to identify national trademark applications.
- All of the bibliographic data contained in the electronic communication is automatically loaded into IPAS, including the logos and the details of the protected Nice classes. This minimizes the need for data capture.
- The IPAS workflow for a new Madrid trademark can use a specific structure or use the same workflow structure as regular national applications. In the latter case, special automatic actions can be configured in the IPAS workflow so as start processing from some specific stage (normally the formality checking phase and the control of fees paid will be skipped for Madrid trademarks).

- Subsequent transactions affecting international trademarks are also processed by IPAS. These transactions are used to generate user documents of a configured type, as if these transactions had been electronically filed in a paper-less scenario. The IPAS workflow structure for these transactions could be the same as for locally-received transactions of such kind, or a special workflow structure could be configured. In the first case, automatic actions are available to bypass certain initial stages like the formalities phase.
- For those transactions affecting the expiration date (i.e. PROLONG) or the ownership (i.e. NEWNAME), IPAS will automatically obtain the new expiration date or the new owners from the Madrid transaction and store this data as part of the transaction bibliographic data. At the time the transaction is approved by the IP office, this data will be used to automatically update the affected Madrid mark.
- All the electronic transactions are stored in the IPAS database. The historic transactions affecting one specific trademark can be viewed on demand so as to provide a complete perspective of the evolution of such trademark.
- In the case of Madrid applications, the IP office must be aware of the provisional refusal due date, since failure to communicate a provisional refusal to the International Bureau before that date will cause the application to be deemed registered in such country. IPAS workflow allows an alert mechanism for warning the IP office about this approaching due dates, so as to expedite processing of the pending application and therefore avoiding an unwilling registration.
- Whenever processing of a Madrid transaction fails, IPAS will inform the error and will keep track in its database of the failed transaction so as to automatically reprocess it until it is eventually processed OK. Failures can be caused by a variety of reasons such as:
    - Timing problems in the sequence of transactions published by the International Bureau, such as a NEWNAME transaction being published before the affected BIRTH transaction. In these cases, reprocessing of the failed NEWNAME will succeed as soon as the missing BIRTH transaction is eventually downloaded and processed.
    - New trademarks classified with new Vienna codes not yet configured into IPAS. In these cases, an action is required of behalf of the IP office to correct the problem, and then the automatic reprocessing of the failed transactions will succeed.

2.8.2 Patents data in ST.36

Patent applications can be exported to XML files following the recommendations of WIPO Standard ST.36. Full compliance is provided to the schemas *xx-patent-document-v1-0-20050220.dtd* and *package-data-v1-5.dtd*.

The data to be exported includes:

- The bibliographic data, e.g. title, classifications, priority claims, parties involved, PCT or regional filing and publication data, status dates, etc.
- The technical data, e.g. description, claims, abstract, drawings, ST.25 sequences, etc.
- The status of the application, e.g. filing, publication and final decisions like grant, refusal or withdrawal.

The purposes of this module could be for instance:

- Informing WIPO International Bureau about the PCT applications which have entered the national phase.
- Informing WIPO International Bureau or other offices about the national applications received and processed.

For historical reasons, patents data can also be exported in ST.32 format.

2.8.3 National emblems under Article 6ter of the Paris Convention

National emblems can also be imported from the 6-ter CDs published periodically by WIPO. The process is not so automatic as in the case of international trademarks, since it requires manually exporting data from the 6-ter CDs to text and then importing that data (and images) into IPAS Java. The trademarks are imported to the same tables as national trademarks, but using a special application type/subtype and numbering so as to avoid duplicates.

2.9 Migration from existing systems

In order to migrate from existing systems, a module is available to import data from a set of normalized views into IPAS Java. Those views include bibliographic data from trademarks and patents, and also the “administrative history” as a sequence of “actions” representing decisions taking by the IP office.

It is possible to configure “historic actions” in IPAS Java, which are special actions showing the history of the process under the existing system. In those cases where the exact status of the file under IPAS Java could be determined from the data in the existing system, the generated process can be assigned to that status so that IPAS Java workflow can start from that point onwards.

2.10 Access authorization

Access authorization to the different system functions is defined by means of roles. Each user is assigned one or several roles and for each role definitions are made for:

- Access to interface elements: each button or hyperlink can be disabled or even hidden, and each data item can be just displayed or also hidden.
- Permission for entering administrative actions of certain types. For example: some users would be allowed to enter "Registration" actions while the rest would not.

Authorization to interface elements can be either “granted” or “restricted”, and the level of authorization can be (from lower to upper):

- The data item, button or hyperlink itself.
- The page to which the item belongs.
- The use case to which the page belongs (e.g. the whole “trademark edit” use case).
- The whole system.

In this sense, even though the granularity of the authorization schema can go down to the field level if required, authorizations can be granted with a coarser granularity if such fine degree of control is not required.

The “restriction” of authorizations will take precedence over the “granting” of authorizations in order to limit the access. Another use of the restrictions could be to simply “hide” from all users, those interface elements which are not required for a specific country, like e.g. “trademark translation” if the law does not require this data item to be presented, etc.

The actual assignment of the authorizations is performed via an Excel spreadsheet exported by the system, and the administrator has all the flexible features of Excel, like dynamic selection, to help in this job.

Apart from the authorization schema, every change made to any database field is audited in a log containing the user responsible, the date and time, the information before the change and the information after the change. Every change made to any database field is audited in a log containing the user responsible, the date and time, the information before the change and the information after the change.

2.11 Interface with external electronic document module

IPAS Java does not currently offer an integrated module for managing the electronic documents, i.e. pages scanned from paper files or documents received already in electronic form. Nevertheless, an optional interface is offered between IPAS Java and any external electronic document module to allow communication and provide some benefits as described below:

- Since IPAS Java manages all the numbering of documents and the recording of bibliographic data, there is no need to perform additional indexing in the external electronic documents module, since all the indexing information can be extracted from the IPAS database. The only indexing required would be the minimum numbers in order to identify the document in IPAS Java, and all additional indexing elements could be extracted from IPAS, like mark name, owner, etc. IPAS Java has mechanisms for automatically updating these indexing codes when changes occur in the underlying file, e.g. when an assignment has changed the owner name.
- IPAS Java will generate an e-document for each input document received, each output document delivered and each internal document intercalated in the file. Such e-document will be considered as a new “scanning request” to the external electronic document module, and IPAS Java will be able to check if such e-document was already scanned.
- The daily log books will be used for this purpose, and IPAS Java can control that the electronic document module has already scanned all required documents produced in one specific day. In this way the “scanning” will be considered as an additional stage in the processing, just as the data capture.
- The interface can be customized so as to configure how the different “electronic folders” are built from the “electronic documents”, e.g. some document types could be just added to the folder which contains the main file, while other document types could be stored in folders of its own, etc.
- If the external electronic document module provides a web interface which can receive the parameters via the activation URL, then IPAS Java will be able to call this display functionality from button in its own interface, and in this way the user will not even have to work with the native interface of the external electronic document module.

Usage of these features would simplify the possible replacement of the external electronic documents module by another instance implementing the same interface, thus diminishing the dependency on any specific vendor.

As a valuable tool for countries which do not still have an existing electronic document solution, an IPAS Java implementation of such electronic document module is planned, based on open source components.

2.12 Tools for the manager

IP office management can exploit the information in their national IP databases for planning, projections, outreach, trends analysis, etc. In addition, the system generates valuable statistics:

§ Statistics related to the productivity of each area measured as quantity of actions performed during a certain period and totaled by action type / user / month / file type / user document type. This information is generated as Excel spreadsheets so that it can be analyzed by using standard pivot table functionality.

§ Ageing of processes, i.e. length of time files have been in particular administrative statuses. Data can be analyzed for a specific user or for a specific status, and export to Excel is also possible in order to further process the generated statistics.

§ Official Statistics required by WIPO are generated automatically through the generation of a series of Excel spreadsheets with pre-defined formats in order to use COPY/PASTE commands in the spreadsheets supplied by WIPO in the statistics CD.

2.13 Multi languages support

All the interface elements (background texts, window names, name items, etc) can be translated into any language. The translation is stored into Java properties files which will be dynamically selected based on the configured language.

Translation of the interface is done via some Excel spreadsheets generated from the administrator module, which contain all the interface elements and the messages requiring translation. The administrator module allows the local administrator to perform all the translation tasks, please refer to the technical description of such module for further details about the interface translation process.

Localized translations are also available to support country-specific translations, e.g. a “neutral Spanish” translation is distributed as part of each new version, but a “local Spanish” translation of some specific terms can be created by the local administrator so as to further localize some specific terms.

Data is internally stored in the database as UTF8 data, so any language supported by Unicode is possible. In case where a codepage exists containing all the required data (e.g. in English or Spanish speaking countries) it is possible to use that codepage and not UTF8 for the storage of data.

In case of non-Latin languages (e.g. Cyrillic) a transliteration function is available to transform from the original text into a simplified Latin version, to be used in the name searches and in the phonetic search functions. E.g. a single Russian letter can sound as TCH, like in Tchaikowksy.

The transliteration rules can be totally configured by the administrator, by identifying the Unicode string to be searched for, and the replacing Unicode string (both strings having length one or more). Other transliteration rules can be used for Latin languages in order to remove accented letters, e.g. the following letters could be transliterated into A: a, á, Á, etc.

In fact, two sets of transliteration rules are defined in IPAS Java:

- The rules used to transliterate data before decomposing it into words for the purpose of the bibliographic words search used for trademark name, patent title, patent abstract, patent claims, etc.
- The rules used to transliterate the trademark name before the phonetisation according to the configured rules. E.g. in Portuguese, some accented vowels sound differently depending on the subsequent letters in the name, and the transliteration rules do not offer the required flexibility to manage those cases, but the phonetisation rules certainly do.

2.14 Singapore Treaty support

As a summary of IPAS Java functionalities, an analysis of the support of all the features included in the Singapore Treaty and its regulations and forms, indicates that most of them are supported by IPAS Java, with the only relevant exception of the e-filing.

The following features of the Treaty are fully supported by IPAS Java:

- Elements contained in application: Art. 3(1)
- Application with multiple Nice classes: Art. 3(2)

- Representative: Art. 4(2)(a). The address of the representative is considered the address for service: Rule 4(1)
- Address for service: Art. 4(2)(b) and Rule 4(2)
- Accordance of filing date: Art. 5(1)
- Registration with multiple Nice classes: Art. 6
- Division of application and registration: Art. 7(1) and (2)
- Recording of change of name or address of the holder or applicant, supporting a single request changing many registrations or applications: Art. 10(1) and (2)
- Recording of change of name or address of the representative or in the address for service, supporting a single request changing many applications or registrations: Art. 10(3)
- Change of ownership, supporting a single request changing many registrations or applications: Art. 11(1) and (2). Only some goods/services could be transferred, as stated in Form 6(3.2)
- Correction of a mistake, supporting a single request changing many registrations or applications: Art. 12(1) and (2)
- Request for renewal: Art. 13
- Request for the extension of a time limit: Art. 14(1)
- Support for Paris Convention: Art. 15
- Support for service marks: Art. 16
- Record of a license, supporting a single request affecting many registrations or applications: Art. 17
- Amendment or cancellation of a license: Art. 18
- Opportunity to make observations on intended refusal: Art. 21

The following minor features of the Treaty are currently partially supported by IPAS Java:

- The Office may require up to six different views of a three-dimensional trademark, while currently IPAS Java supports a single two-dimensional view.
- Hologram trademarks, motion trademarks, colour trademarks and position trademarks are not explicitly supported, even though specific subtypes can be easily configured for them. The only special trademark type supported is sound trademarks, through the storage of an audio recording file that can be played back for reproduction of the trademark.
- Transliteration of the trademark into plain Latin characters is performed automatically in order to perform the phonetic search, but an applicant-supplied transliteration is currently not supported.
- The Power of Attorney is supported as described in Art 4(3), but IPAS Java supports just a text description of the scope of the empowerment, while Form 2 contains more precise terms such as explicit checkboxes if the representative can withdrawn or surrender the trademark, etc. All these items related to the scope would need to be entered as plain text descriptions.

The following features of the Treaty are not currently supported by IPAS Java:

- Electronic filing of applications and communications.

From the procedural point of view, it was already mentioned that IPAS Java does not implement a fixed legislative framework but that its flexible workflow features allow configuring almost any procedure that an IP office might use. In particular, all the procedures described by the Singapore regulations as supported, such as the procedure in Case of Non-Compliance with Requirements, producing a letter inviting the applicant to comply with the non-complied requirements (Rule 5), etc.

3 Functionality summary

This section briefly summarizes the different functionalities of the IPAS Java system, which have been already described in detail in the previous section:

- **Register of trademarks**
    - Flexible numbering can be configured.
    - Full bibliographic data can be recorded, including logo.
    - Person identification numbers can be configured.
    - Register of Agents is available.
    - Register of Power-of-Attorney is available.
    - For search purposes Nice classes can be configured so that they are related to others where similar products or services can be found.
    - Mapping between national and Nice classes is supported.

- **Trademarks searching**
    - Bibliographic searching by all basis numbers and dates and also by words contained within the mark name or owner name (search is case-insensitive and accent-insensitive).
    - Phonetic search for new applications, preliminary search requests and ad-hoc interactive requests.
    - Logo search based on the Vienna codes.
    - Similarity list is kept for each trademark, where different search sources are consolidates (phonetic, logo, manual) and similarities can be flagged as “selected” and “discarded”.

- **Register of patents**
    - Flexible numbering can be configured.
    - Full bibliographic data can be recorded, including many drawings, abstract, many claims, etc.
    - Search classes are IPC for patents/utility models and Locarno for designs.
    - Local technical classes can be configured, which are automatically mapped from the main IPC.
    - Person identification numbers can be configured.
    - Register of Agents is available.
    - Register of Power-of-Attorney is available.
    - Relationships between the patent and other predecessor or successor patents as per some existing relationship like division, enhancement, transformation of patent into utility model, etc.
    - Annuities are generated automatically based on flexible criteria. Payment of annuities can be recorded.
    - Non-paid annuities are detected and a warning is generated in order to analyze the potential abandonment of the patent.

- **Patents searching**
    - Bibliographic searching by all basis numbers and dates and also by words contained within the title name or owner name (search is case-insensitive and accent-insensitive).
    - Technical search criteria available.
    - Examination report can be recorded also, detailing the search fields used, the search strategy used, the list of relevant documents as per WIPO standards, etc.

- **Register of other documents**
  - Flexibility in configuring the types of documents, for instance documents voluntarily filed by the applicant, documents filed by the applicant as a response to some previous request from the office, documents filed by a third party who is against the applied IP right, documents filed by a Court of Justice, requests of information by the public, etc.
  - Flexible numbering can be configured.
  - One document may affect many files (normally for ownership change).
  - Documents may alter the ownership or representatives, and the new owners or representatives can be entered into the Register. Then, upon approval of the document, the system will be able to automatically update the ownership or representative information of the affected file.
  - Documents may renew the affected file. In this case, its approval will trigger the extension of the validity period.

- **Workflow configuration**
  - Workflow is fully configured using a graphical tool to draw the state transition diagram showing all the statuses, the actions which can be recorded and the subsequent change in the status.
  - A “workflow unit” is generated by the system for each new file, received document or letter produced by the IP office.
  - Assignment of responsibility for a specific workflow unit is supported, for instance all the trademarks in the examination status can be broken down by the assigned examiner, etc.
  - Office documents can be printed for selected actions when the applicant needs to be notified. The layout of these office documents is configured using Word templates and MailMerge fields generated from the database. Images (like trademark logo and patent drawing) can also be included in the document.
  - MailMerge fields are fully configurable.
  - Each workflow unit can have a “due date” indicating the maximum time the underlying activity should be performed, e.g. when a letter was notified to the applicant and the IP office is waiting for a response. Calculation of due dates is fully configurable, e.g. non-working days can be used, etc.
  - Detection of due dates can be configured in the system so as to generate warnings.

- **Journal**
  - The moment in which an application or registration must be published in the journal is fully configurable.
  - Certain application types (e.g. patents) can be configured so that publication is not performed until a certain period of time has lapsed from the filing date, and also a special publication date request is supported.
  - Template of the journal section is configured using Word templates, and the fields to be included are MailMerge fields which are fully configurable.
  - The logo can be included as part of the data in the journal.

- **Physical tracking**
  - Physical deliveries of documents between office members can be tracked by printing “delivery lists” which electronically signed.
  - Applications, other documents received from the user and even letters generated by the IP office can be tracked.

- **Document reception and notification**
  - Documents can be received from the applicant, and a configurable reception acknowledgement receipt can be printed.

    - Letters and other documents produced by the IP office can be notified to the applicant. The notification procedure is fully configurable since it uses the same workflow engine.

- **Data import and export**
    - Electronic transactions generated in the Madrid system can be imported.
    - Patents can be exported as per the ST.36 standard.
    - National emblems protected under Article 6ter of the Paris Convention can be imported.

- **Authorizations**
    - Roles are supported for configuring the authorizations.
    - Access to all interface elements can be fully configured for each role, e.g. each button or hyperlink can be disabled or even hidden, and each data item can be protected or even hidden.
    - Permission for entering administrative actions of certain types is also configured for each role.

- **Internationalization**
    - The interface can be fully translated using Java resources.

4 IPAS Java architecture and technology

This section describes IPAS Java architecture and technology.

IPAS Java consists of the following components:

- A relational database management system for storing data.
- An application server offering a set of high-level interfaces to access IPAS Java functionalities.
- Several clients of the IPAS Java application server, namely:
    - A web application (named IpasWeb) offering daily functionality to the end users.
    - A web application (named IpasAdmin) offering special functionality to the supervisors and the administrator.
    - A web application (named IpasPublic) offering query functionality to the public.
    - A web application (named IpasImport) used for data import during the migration process.
    - A .NET application which imports electronic transactions from the Madrid system. This application is in the process of being converted to Java so as to integrate it within the IpasAdmin module.
- A standalone environment administration tool which creates and manages IPAS Java environments.
- A standalone domain administration tool which creates and manages GlassFish domains.
- A graphical designer tool which allows editing the IPAS Java configuration.

The relational database management system can be either ORACLE or Microsoft SQL Server. A single database instance is used, but many independent schemas can be created, one for each IPAS Java environment. The IPAS environment administration tool is used for creating and managing IPAS Java environments, including the creation of the corresponding database schemas. No database expertise is required for the administrator, since IPAS automatically creates all the required database objects like tables, views, indexes, etc.

IPAS Java uses the GlassFish v2 application server for the Java Enterprise Edition (Java EE) platform. GlassFish is an open source project led by Sun Microsystems and it is based on source code donated by Sun and Oracle. It uses a derivative of Apache Tomcat as the servlet container for serving Web content, with an added component called Grizzly which uses Java NIO for scalability and speed.

The IPAS Java application server is implemented as an enterprise application which is packaged as an EAR (Enterprise Archive) file named *IpasEar.ear*. This package is deployed inside the GlassFish application server and therefore it is shown in the "Enterprise Applications" list in the GlassFish console. This package contains the following sub components:

- ***Ipas.jar*** contains a set of EJB 3.0 (Enterprise Java Beans) which are stateless session beans encapsulating the back-end business logic of IPAS Java. Details on these EJBs will be discussed below.
- ***IpasMonitor.war*** is a WAR file (Web Application Archive) containing a web application used for performance monitoring within the application server. It is possible to monitor how much time each of the EJBs methods is taking, in average, minimum and maximum, with a breakdown of the time used by the different layers down to the database itself. A detail of all the SQL statements executed within each method is also available, which is a valuable tool for detecting performance issues related to the database layer.

The web application components of IPAS Java are packaged as WAR files which are also deployed inside the GlassFish application server and are shown in the “Web Applications” list. The following web applications are available:

- ***IpasWeb.war*** offers daily functionality to the end users.
- ***IpasAdmin.war*** offers functionality to the administrators and supervisors, like generation of statistics, preparing the journal, etc.
- ***IpasPublic.war*** offers query functionality to the public.
- ***IpasImport.war*** imports data during the migration process.

We will now analyze into some more detail each of the above components.

4.1 IPAS Java database

The IPAS Java database stores all the information in a relational database with the following features:

- The relational database management software can be Oracle or SQL Server, but internal IPAS Java code is one and only for all the database flavors. In order to support multiple databases, all IPAS Java SQL statements use an ad-hoc SQL dialect, christened X-SQL (for “eXtended SQL”), that is dynamically mapped to the target database SQL syntax.

  In this way, the same statement in X-SQL will be dynamically transformed into different SQL statements depending on the target database server. This is accomplished without extensive parsing by the following set of features:

  - X-SQL is as “neutral” as possible, avoiding syntax constructions which are specific to only one database manager, so most X-SQL statements require no transformation at all to be executed successfully in any database manager.
  - X-SQL is generally aligned with Oracle syntax (simply for historical reasons since Oracle was the first database supported). For some syntax constructions, the X-SQL statements are translated in order to produce syntax adequate for the target database, e.g. the “outer join” syntax used by X-SQL (using Oracle’s style “(+)” suffix after the column name) is transformed at runtime into the syntax used by SQL Server.
  - Most X-SQL functions are those of Oracle and during installation under SQL Server, IPAS Java creates equivalent functions in the target database so that those functions will work correctly. E.g. TO_DATE() X-SQL functions (similar to Oracle’s) are created as SQL Server scalar functions, etc.
  - In some complicate cases, syntax extensions allow the X-SQL statements to be mapped into different SQL statements. E.g. Oracle supports the conditions

    (A, B, C) IN (SELECT X, Y, Z FROM TABLE)

    while SQL Server does not, and so the statement

    (A + B + C) IN (SELECT X + Y + Z FROM TABLE)

    is generated for SQL Server.
  - A complete guide for X-SQL syntax is available for programmers wishing to alter IPAS Java code.

Multiple separated environments are supported, using a separate “database schema” for each environment. One of those environments is the one holding the production data, and the rest could be used for training, etc. There is no need to create separate databases, since all the environments are created and managed dynamically by IPAS Java as part of its environment administration module.

IPAS Java automatically controls structure of the underlying database (table names, column names, constraint names, etc) through analysis of the database catalogue, to automatically detect the software version. Based on this, IPAS Java is capable of updating the database structure from one version to the next one via execution of “delta scripts” containing all the SQL statements required for defining new tables or columns added in the new version. All the SQL statements of these delta-scripts are themselves in X-SQL, so a single script is used for updating all database brands.

Database type migration, e.g. from Oracle to SQL Server, is easily done through the export of all the data in XML format. In this way, a database-independent export file is generated, which can be restored in any IPAS Java installation using any database type. All the required activities are performed within IPAS Java, no additional software is required for such migration.

For backup purposes, even though this XML export file could be used, much better performance can be obtained using other IPAS Java features to generate an export of one environment, automatically calling the appropriate database export/import tool (this feature is only supported for Oracle). The export file can also be used to create a new environment, e.g. the test environment could be periodically updated with actual production data using the last production backup.
Alternatively, a DBA could take care of backups using native database tools.

The above discussions show that the need for a DBA role is minimized, for instance:

- Initial installation only requires that the database server software be installed and that a user with DBA privilege is created. Then IPAS Java does all the rest of the process (creating tables, indexes, etc).
- Export and import are also handled by IPAS Java, with no need to use DBA tools (only under Oracle).
- Index rebuilding is automatically triggered by IPAS Java based on a period configured by the user.

IPAS Java was designed to be able to work in an autonomous way, without any local expertise in database management. Even if the server crashes, the recovery requires very little technical knowledge, for instance the steps for a disaster recovery would just be:

- Database software must be installed again.
- IPAS Java must be installed again.
- The last backup is then used to create a new environment.

The IPAS Java database is documented in a document which describes each table in terms of:

- A brief description about its contents.
- A list of its columns, with a description of its meaning and data type.
- A list of the Foreign Keys (FKs) indicating the source columns, target table and the meaning of the association. The name of the target table is a hyperlink that allows easy access to its details.

The full conceptual description of the tables and their relationships is contained in *Manual Tecnico Base Datos - Modelo Logico.doc* (only is Spanish, an English translation will be available) which is a 110 pages description of the semantics of each entity underlying the IPAS Java tables and its relationships with other entities.

Table names use prefixes indicating:

- ***IP_*** : tables related to the normal daily operation of the IP office: trademarks, names, files, etc.
- ***SE_*** : tables related to the search module, which is independent from the *IP_**** tables used for daily operations.
- ***CF_*** : tables related with the configuration of the system, i.e. information that does not change with the daily operation but reflects the structure of the work being done: process types, application types, etc.
- ***DO_*** : tables related with the e-doc interface.
- ***WK_*** : work tables used as auxiliary means in certain processes: work data for searching, etc.

4.2 IPAS Java application server

As already mentioned, the *IPAS Java application server* is in fact a set of EJBs deployed inside the GlassFish J2EE application server. The services provided are aligned to the business needs and not to the database design. More than 100 methods are available, structured in 11 interfaces.

All application server methods are logically grouped in interfaces. Interfaces start with *I*, then the method name starts with *m*, e.g. *IMark.mCount* is a methods which allows counting how many trademarks fulfill certain criteria.

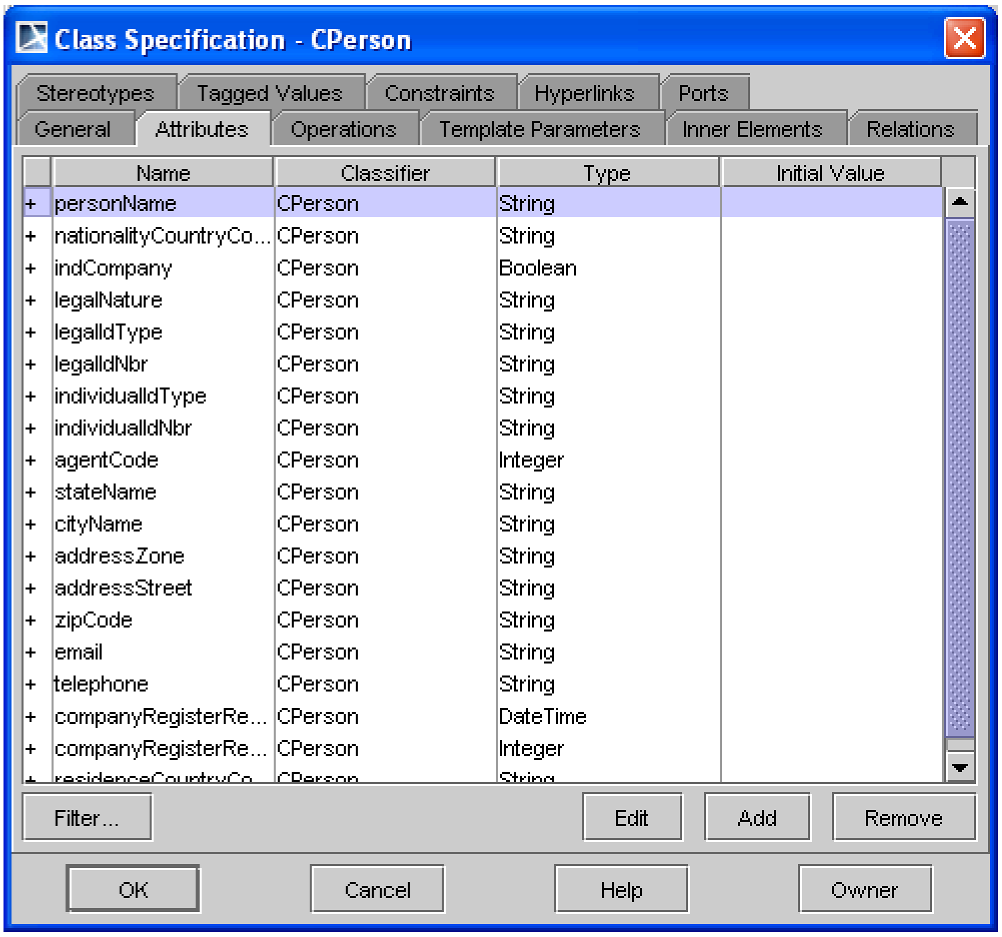
Some main interfaces and their methods are as follows:

- **Interface ICommonServices (6 methods):** manages configuration parameters, connection to the Application Server and Madrid register.
- **Interface IDelivery (17 methods):** manages physical deliveries.
- **Interface IFile (6 methods):** manages “files” as a common super-type of both trademarks and patents.
- **Interface IJournal (12 methods):** manages the journal.
- **Interface IDailyLog (16 methods):** manages the daily logs.
- **Interface IMark (13 methods):** manages trademarks.
- **Interface IOfficedoc (5 methods):** manages office documents.
- **Interface IOutputField (10 methods):** manages output fields, which are used for the production of output documents.
- **Interface IPatent (8 methods):** manages patents.
- **Interface IProcess (25 methods):** manages processes, actions and the workflow.
- **Interface IProcessGroup (10 methods):** manages process groups.
- **Interface IUserdoc (11 methods):** manages user documents.

This totals some 140 methods exposed by the Application Server, which allow access to the IPAS Java kernel functionality. Access to this functionality can be done directly via native method invocation of the EJBs, or via a factory-type API which returns Java objects implementing the methods of each EJB.

It is not the objective of this document to fully describe all the available methods in the application server, nor the Java classes involved (there are more than 900 Java classes), since this detail is only required for the developer, but just to give a broad idea of the high-level IPAS Java application server interface to the clients.

It must be noted that most of those Java classes and interfaces are generated using the MDA (Model-Driven Architecture) approach from an underlying UML (Unified Modeling Language) model. The model itself is an XML file following the UML structure (the DTD is uml14xmi12.dtd) and a UML tool is used for editing this model. The tool used is MagicDraw 9.5, but any other UML tool could be used. The UML model offers a high-level representation of the classes and interfaces, for instance here we include a screenshot of the contents of the *CPerson* class containing the details of a person:



	Name	Classifier	Type	Initial Value
+	personName	CPerson	String	
+	nationalityCountryCo...	CPerson	String	
+	indCompany	CPerson	Boolean	
+	legalNature	CPerson	String	
+	legalIdType	CPerson	String	
+	legalIdNbr	CPerson	String	
+	individualIdType	CPerson	String	
+	individualIdNbr	CPerson	String	
+	agentCode	CPerson	Integer	
+	stateName	CPerson	String	
+	cityName	CPerson	String	
+	addressZone	CPerson	String	
+	addressStreet	CPerson	String	
+	zipCode	CPerson	String	
+	email	CPerson	String	
+	telephone	CPerson	String	
+	companyRegisterRe...	CPerson	DateTime	
+	companyRegisterRe...	CPerson	Integer	
+	residenceCountryCo...	CPerson	String	

From this UML model, an MDA (Model-Driven Architecture) tool is used to generate the Java classes and interfaces. The MDA tool used is an open-source tool called AndroMDA, which allows configuring the generation mechanisms (called cartridges) so as to produce the desired target Java code. All the generated Java classes are POJOs, with the required setter and getters, which are then used for building the EJBs (Enterprise Java Beans) required by the J2EE architecture.

The Java code generated from the UML model includes the API itself and all the stubs for the actual implementation of the application server methods. Whenever a change in the API is required, the underlying UML model is updated, the API and the stubs are regenerated, and the application server implementation is updated so as to provide the additional services.

4.2.1 MailMerge

In the functionality described above it was mentioned that IPAS Java generates the output Word documents without using Microsoft Word itself. In fact, IPAS Java uses a pure Java library called *Aspose.Words (Java version)*.

The Aspose.Words library does not require Microsoft Office to be installed on the server computer in order to work. The most relevant features of this approach are the following:

- Aspose.Words allows developers to build or modify documents and formatting with ease using a document object model consisting of over 100 classes. Developers can programmatically create, modify, extract and replace all document elements including sections, headers, footers, paragraphs, lists, tables, text, fields, hyperlinks, bookmarks and images. Developers can specify detailed formatting for any document element.
- The above features are used by IPAS Java to allow not only simple text MailMerge fields, but more sophisticated processing like inclusion of images and tables in the output documents. These features are especially useful for the inclusion of trademark logos and patent drawings in different documents, like journal entries, registration certificates, etc.
- The data for the MailMerge comes directly from Java structures, so no “data sources” are used.

The generated Word documents can be either:

- Stored locally in the file system of the server, e.g. when a journal is generated then the Word files with each section of the journal can be stored in the file system and then copied to some workstation for further processing. Obviously Microsoft Word must be installed in such workstation in order to access those files.
- Downloaded via the Internet Browser to the client computer. It is also not required to have Microsoft Word installed in the client computers, since all that is required is a free Word viewer (see http://www.microsoft.com/DOWNLOADS/details.aspx?familyid=95E24C87-8732-48D5-8689-AB826E7B8FDF&displaylang=en). With this (or any other) viewer, the client computer can view and print the generated Word document, but not edit or save the document. If editing is desired, Microsoft Word must be installed in the user computer.

4.3 IPAS Java web client for the end user

The daily functionality of IPAS Java is supported by a web application called *IpasWeb*. The main features are as follows:

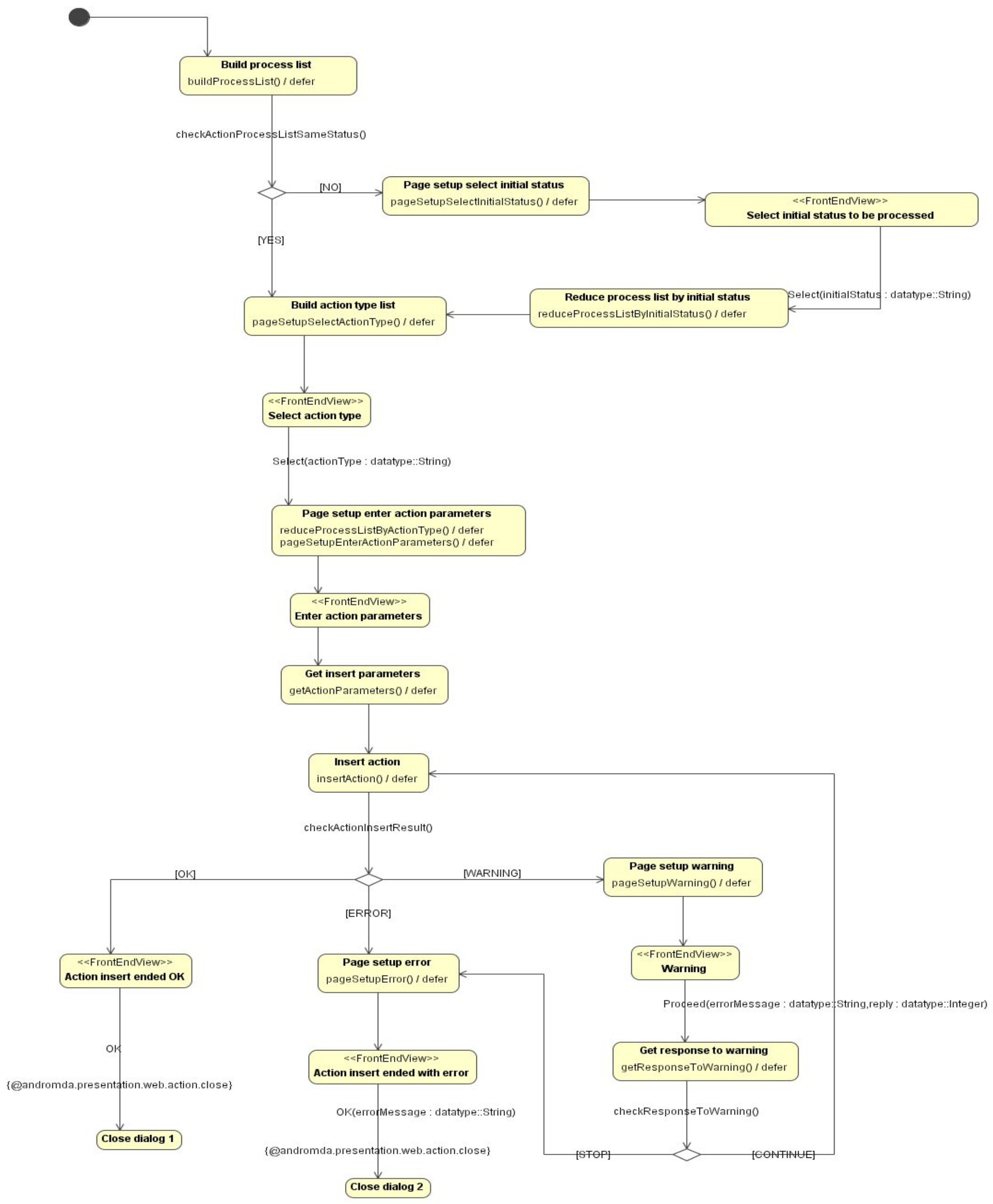
- Supports all the functionality described above under the “Functionality” section, except some administration functions which are covered by the IPAS Java Administrator module.
- Totally developed in Java, with English meaningful function names, variable names and comments.
- Does not access IPAS Java database directly, but uses the “IPAS Java Application Server” through the interfaces already discussed.
- Based on open source standards (Java Server Pages and Apache Struts).
- Extensive use of Java resource files allows easy customization of the interface.
- Unicode native support.

The IPAS Java web client is fully written in Java, using the Apache Struts framework. The web module is divided into “use cases”, and each of them is an interlinked set of web pages (what the user sees) and server processes (what the system does). A UML model was used to design those use cases, which represent all the possible courses of action in the interaction between the user and the system.

This approach allows for a high degree of independence between the business logic layer (the processes performed by the system) and the presentation layer (the web pages viewed by the user). This decoupling is a key element in the Model-View-Controller architecture.

In order to clarify the above concepts, the next image is a printout from the UML editing tool showing the diagram which represents one of those use cases diagram, in particular the one responsible for the recoding of a new action in a process. The main elements in the diagram are the following:

- Rectangular nodes with the <<FrontEndView>> tag (e.g. *Select action type*) represent client-side operations, i.e. JSPs (Java Server Pages) where information is shown to the user, and data is captured from the user.
- Lines getting into those nodes contain the information displayed on the pages, and lines coming out of these nodes contain the information captured by the user.
- The other rectangular nodes (e.g. *Build process list*) represent server-side operations used to prepare the information to be displayed on the web pages, or the process the information captured in the web pages. These server functions are developed in Java and they interact with the IPAS application server.
- Diamond nodes represent decisions taken on the server that affect the normal flow, e.g. when an error situation is found the flow of control can be diverted.


Build process list
buildProcessList() / defer
checkActionProcessListSameStatus()
[NO]
Page setup select initial status
pageSetupSelectInitialStatus() / defer
<<FrontEndView>>
Select initial status to be processed
[YES]
Select(initialStatus : datatype::String)
Reduce process list by initial status
reduceProcessListByInitialStatus() / defer
Build action type list
pageSetupSelectActionType() / defer
<<FrontEndView>>
Select action type
Select(actionType : datatype::String)
Page setup enter action parameters
reduceProcessListByActionType() / defer
pageSetupEnterActionParameters() / defer
<<FrontEndView>>
Enter action parameters
Get insert parameters
getActionParameters() / defer
Insert action
insertAction() / defer
checkActionInsertResult()
[OK]
[WARNING]
Page setup warning
pageSetupWarning() / defer
[ERROR]
<<FrontEndView>>
Action insert ended OK
Page setup error
pageSetupError() / defer
<<FrontEndView>>
Warning
Proceed(errorMessage : datatype::String,reply : datatype::Integer)
OK
{@andromda.presentation.web.action.close}
<<FrontEndView>>
Action insert ended with error
Get response to warning
getResponseToWarning() / defer
OK(errorMessage : datatype::String)
checkResponseToWarning()
Close dialog 1
{@andromda.presentation.web.action.close}
[STOP]
[CONTINUE]
Close dialog 2

In order to increase the decoupling, the server functions do not interact directly with the pages, but interact with three elements:

- With a “Struts form” that includes the required Java representation of each interface element in the web pages which are directly or indirectly related to such server function. E.g. the server function (the “controller” in MVC jargon) uses Java getters and setters in the form to access certain data items, but does not know whether those data items will be displayed in the web page as a data field, as a radio button, or as a combo, etc.
- With session variables that store data at the session level.
- With the IPAS Java application server.

Generation of the JSP code has been customized in order to allow a richer interface than the rather simple one offered by plain JSP. The following features were added using user-defined UML tags in the model in order to support the following rich-interface elements:

- Any button triggering an action can be rendered in a toolbar, with an option icon graphically describing the action (like a disk icon for “save”, and “X” icon for “cancel”, etc).
- When many data items exist in a single form, they can be grouped in tabs so as to hide/show them as required. E.g. all the data items for a trademark are not shown in one long form that will need scrolling, but in several tabbed frames that allow a better distribution in the page.
- Some use cases can be executed in parallel to the main use case, in independent sections of the page using Ajax features. E.g. the main bibliographic data for a trademark is managed by a use case which is displayed in one or many tabs, and each of the “repeating” data items, like list of owners, list of Nice classes, etc, are displayed in “Ajaxized” use cases which are displayed in their own tabs. All these “satellite” use cases communicate via session variables, and a single “save” button allows saving all the data to the database.
- Some pop-up use cases are executed in a separate browser window, and processing in the main use case is stopped until execution of the pop-up use case ends. This “modal dialog” feature is not offered by JSP, and required some JavaScript code to be embedded into the JSPs.
- Recursive use cases are supported, e.g. from a use case editing a trademark it is possible to activate another instance of the same use case, e.g. with the purpose of selecting a referenced trademark.
- Buttons, hyperlinks and data fields can be conditionally rendered or disabled based on JSP EL (Expression Language) constructs. The authorizations mechanisms described in the functionality sections above use these features to selectively hide or protect specific interface elements.

All these features allow a much richer interface design than plain JSP would offer, and since the code is generated from the UML model, it would be relatively simple to replace one implementation by another.

In summary, the MDA approach allows a very high level of decoupling between the server functions and the interface, and makes the IPAS Java web client fairly independent from the underlying technology used.

4.4 IPAS Java web client for the administrator

The administrator functionality of IPAS Java is supported by a separate web application called *IpasAdmin*. From a technological point of view, this web application is total similar to the *IpasWeb* application described above, so all the concepts discussed there will also apply.

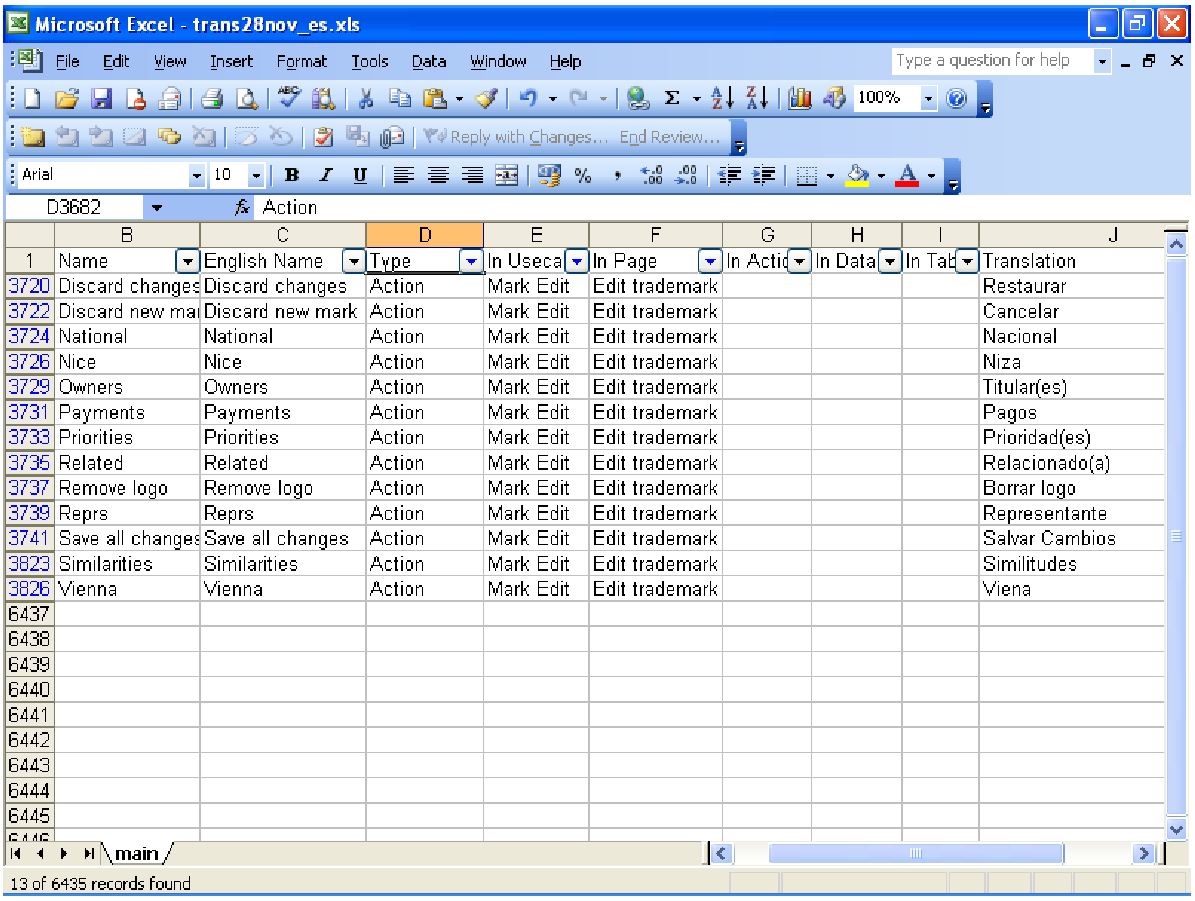
4.5 IPAS Java environment administration module

The administration functions required to deal with the environments are implemented through this stand-alone Java administration module. It accesses the IPAS Java database directly via JDBC, i.e. not through the application server, and the covered functionality is basically:

- Initial creation of the IPAS database when the tool is connected to an empty database.
- Export and import of configuration data to XML files. These XML files will be updated off-line by the graphical configuration tool (Designer) descried below.
- Export and import of translation data to XLS files. These XLS files will be updated off-line by the administrator using Excel.
- Export and import of authorization data to XLS files. These XLS files will be updated off-line by the administrator using Excel.
- Environment management, i.e. creation, deletion, backup (for Oracle), import from a backup (for Oracle), etc.
- Tuning (rebuilding of the indexes).

4.5.1 Interface translation

Interface translation is one of the most important tasks of the administrator. The image below shows an example from an actual generated Excel file showing all the buttons in the *Edit trademark* page within the *Mark Edit* use case, and the translated texts into Spanish:


Microsoft Excel - trans28nov_es.xls

D3682 | Action

	B	C	D	E	F	G	H	I	J
1	Name	English Name	Type	In Useca	In Page	In Actio	In Data	In Tab	Translation
3720	Discard changes	Discard changes	Action	Mark Edit	Edit trademark				Restaurar
3722	Discard new mai	Discard new mark	Action	Mark Edit	Edit trademark				Cancelar
3724	National	National	Action	Mark Edit	Edit trademark				Nacional
3726	Nice	Nice	Action	Mark Edit	Edit trademark				Niza
3729	Owners	Owners	Action	Mark Edit	Edit trademark				Titular(es)
3731	Payments	Payments	Action	Mark Edit	Edit trademark				Pagos
3733	Priorities	Priorities	Action	Mark Edit	Edit trademark				Prioridad(es)
3735	Related	Related	Action	Mark Edit	Edit trademark				Relacionado(a)
3737	Remove logo	Remove logo	Action	Mark Edit	Edit trademark				Borrar logo
3739	Reprs	Reprs	Action	Mark Edit	Edit trademark				Representante
3741	Save all changes	Save all changes	Action	Mark Edit	Edit trademark				Salvar Cambios
3823	Similarities	Similarities	Action	Mark Edit	Edit trademark				Similitudes
3826	Vienna	Vienna	Action	Mark Edit	Edit trademark				Viena
6437									
6438									
6439									
6440									
6441									
6442									
6443									
6444									
6445									

main

13 of 6435 records found

The columns of the Excel spreadsheet are:

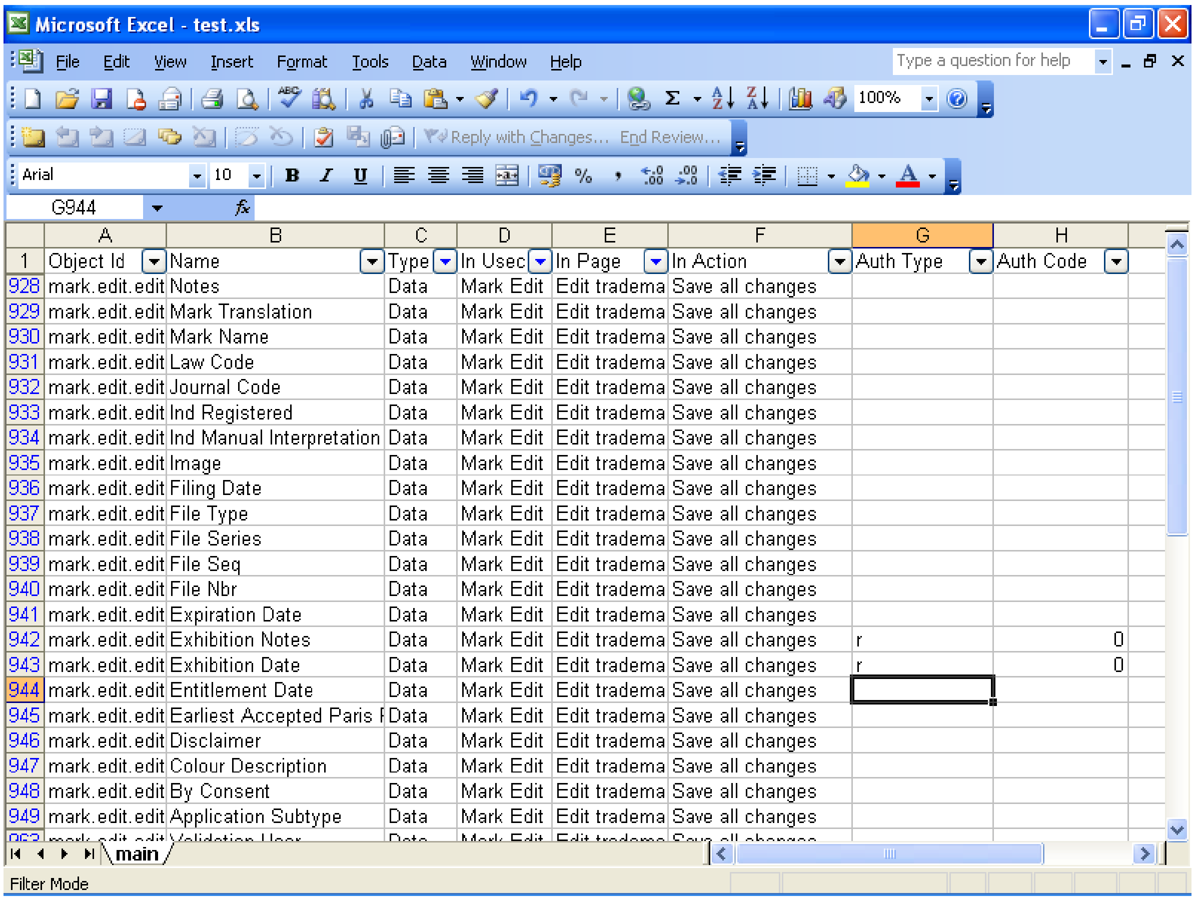
- Object Id: internal id of the object in the Java properties file. This is the internal key of the object, but can be ignored b the person performing the translation.
- Name: original name of the object.
- English Name: English translation of such name (normally it is the same as the name).
- Type: type of object:
  - Action: buttons performing actions.
  - Action help: help for those actions.
  - Column: columns within a table.
  - Data: data fields.
  - Data help: help for the data fields.
  - Message: messages.
  - Page: titles of the pages.
  - Page help: help for the pages.
  - Radio option: options for radio buttons.
- In Usecase: name of the use case where the object appears.
- In Page: name of the page where the object appears.
- In Action: name of the action where the object appears,
- In Data Field: name of the data field where the object appears.
- In Table: name of the table where the object appears.
- Translation: translated text.
- Status: indication whether the object is new and therefore requires translation.

Excel “autofilter” features can be used in this spreadsheet, for instance in the shown example the filters “Type=Action”, “In Usecase=Mark Edit” and “In Page=Edit trademark” were used for selecting just the desired objects.

When each new IPAS Java version is distributed, both an English and a “basic Spanish” version are delivered. These translations are contained in the Java properties files named *application-resources_en.properties* and *application-resources_es.properties*. The “basic Spanish” terms include a translation into Spanish terms but trying to use a “neutral” sort of Spanish. If the IP office wishes to alter some of those terms, a specific translation into its own variation of the language can be created using the above Excel spreadsheet and the tools in the administrator module. For instance, if Costa Rica want to alter some 50 terms because of their local customs, all that is required is to build a new translation table for the language *es_CR* (i.e. Spanish from Costa Rica), and if the web browsers indicate this value in the regional configuration, then the updated 50 terms will be used in the interface, and the rest of the “neutral Spanish” terms will be used for the remaining terms.

4.5.2 Interface authorization

The Excel spreadsheet used for editing the authorizations works in a similar way, for instance the image below shows a sample:


Microsoft Excel - test.xls
File Edit View Insert Format Tools Data Window Help
Type a question for help
Reply with Changes... End Review...
Arial 10
100%
G944
main
Filter Mode

1	Object Id	Name	Type	In Usec	In Page	In Action	Auth Type	Auth Code
928	mark.edit.edit	Notes	Data	Mark Edit	Edit tradema	Save all changes		
929	mark.edit.edit	Mark Translation	Data	Mark Edit	Edit tradema	Save all changes		
930	mark.edit.edit	Mark Name	Data	Mark Edit	Edit tradema	Save all changes		
931	mark.edit.edit	Law Code	Data	Mark Edit	Edit tradema	Save all changes		
932	mark.edit.edit	Journal Code	Data	Mark Edit	Edit tradema	Save all changes		
933	mark.edit.edit	Ind Registered	Data	Mark Edit	Edit tradema	Save all changes		
934	mark.edit.edit	Ind Manual Interpretation	Data	Mark Edit	Edit tradema	Save all changes		
935	mark.edit.edit	Image	Data	Mark Edit	Edit tradema	Save all changes		
936	mark.edit.edit	Filing Date	Data	Mark Edit	Edit tradema	Save all changes		
937	mark.edit.edit	File Type	Data	Mark Edit	Edit tradema	Save all changes		
938	mark.edit.edit	File Series	Data	Mark Edit	Edit tradema	Save all changes		
939	mark.edit.edit	File Seq	Data	Mark Edit	Edit tradema	Save all changes		
940	mark.edit.edit	File Nbr	Data	Mark Edit	Edit tradema	Save all changes		
941	mark.edit.edit	Expiration Date	Data	Mark Edit	Edit tradema	Save all changes		
942	mark.edit.edit	Exhibition Notes	Data	Mark Edit	Edit tradema	Save all changes	r	0
943	mark.edit.edit	Exhibition Date	Data	Mark Edit	Edit tradema	Save all changes	r	0
944	mark.edit.edit	Entitlement Date	Data	Mark Edit	Edit tradema	Save all changes		
945	mark.edit.edit	Earliest Accepted Paris I	Data	Mark Edit	Edit tradema	Save all changes		
946	mark.edit.edit	Disclaimer	Data	Mark Edit	Edit tradema	Save all changes		
947	mark.edit.edit	Colour Description	Data	Mark Edit	Edit tradema	Save all changes		
948	mark.edit.edit	By Consent	Data	Mark Edit	Edit tradema	Save all changes		
949	mark.edit.edit	Application Subtype	Data	Mark Edit	Edit tradema	Save all changes		

The columns identifying the objects are similar as what was already described, and so the possibilities exist to use “auto filter” so select the desired elements. But the relevant columns here are the following:

- Auth Type: indicates whether we are granting an authorization (using the code “G”) or whether we are restricting an authorization (using the code “R”).

- Auth Code: indicates the authorization level granted or restricted:
  - 0: completely hide the interface element.
  - 1: just display the interface element but do not allow edit.
  - 2: allow edit of the interface element.

In the above case, we are “hiding” two interface elements which are not relevant for the specific IP office.

As discussed in the functionality section, authorizations can be recorded at any level, and they are “cascaded” to all lower levels, unless a more specific authorization is given at a lower level. This allows great flexibility without having to record many individual authorizations.

4.6 Graphical Designer module

The general configuration procedure consists in the following steps:

- Exporting the IPAS configuration tables to an XML file.
- Editing that XML file with a specialized graphical designer module.
- Importing the updated XML file back into IPAS.

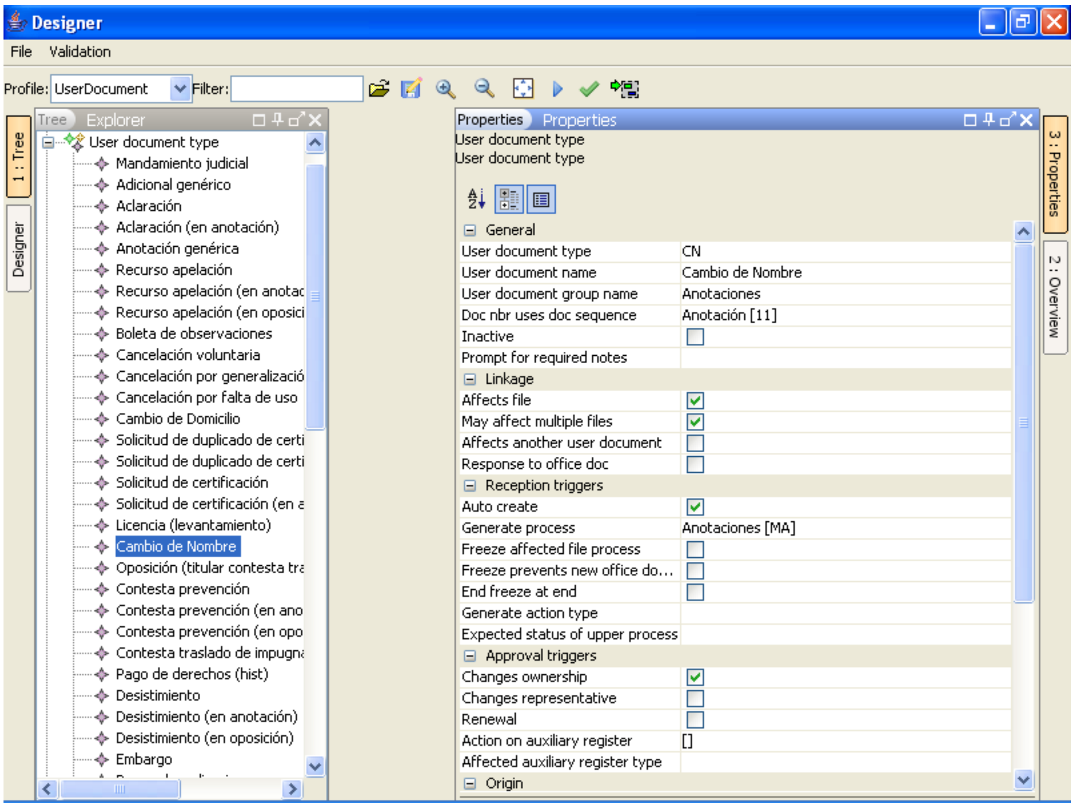
Export and import are performed by the environment administration module already mentioned, and the editing of the exported XML configuration file is done using a graphical designer tool, which allows editing the data in a hierarchical approach using properties panes. Some configuration information, like the workflow, is also displayed graphically in the form of a state transition diagram.

The entities to be configured are shown in groups related to the different configuration areas, and these groups are the following:

- ***Workflow*:** modeling of the status transition diagrams which are the key to the IPAS workflow. A graphical tool is offered to show this diagram.
- ***Authorisation*:** users, roles and the relationship between them. Also the authorizations for a role to record certain action types. The authorizations for each role to interact with certain interface elements can also be configured here, but the system offers a specialized Excel-style interface to perform this task.
- ***OutputField*:** indicates the output fields to be generated by IPAS via configured SQL statements, what are the conditions that trigger generation of those fields, and the details about the required SQL statements to be executed.
- ***Reception*:** origins where documents can be received, and what document types can be received in each one.
- ***Application*:** configured the different types and subtypes of applications, how is each subtype configured under each possible law, which types can be received in each document origin and what are the types of relationships that link the different applications (e.g. trademark division, etc).
- ***User documents*:** types of received user documents and how they should be treated.
- ***Numbering*:** mechanisms for assigning numbers to applications, user documents and actions.
- ***ClassNice*:** defines the Nice classes, how they are related for the purposes of the search, the national classes and the reclassification between national and Nice classes.
- ***ClassLocarno*:** list of Locarno classes.
- ***ClassIPC*:** IPC classes, technical classes and the mapping between them.
- ***ClassVienna*:** Vienna classes.

- ***E-doc***: details the sequences to be used for numbering e-docs and e-folders, the types of the e-docs, the sections types contained in the e-docs, which sections are valid for indexing each e-doc type, and what is the mapping between the different IPAS documents (applications, user documents and office documents) to the e-docs and e-folders.
- ***DocumentSection***: types of document sections and which ones are valid for each application and user document type.
- ***Coding***: various codes like countries, person identification criteria, application types, etc.
- ***ConfigParam***: configuration parameters controlling some IPAS features.

The following image is a printout from the Designer module, showing the configured list of user document types for one specific country, and the properties which define how IPAS will treat documents of each type:

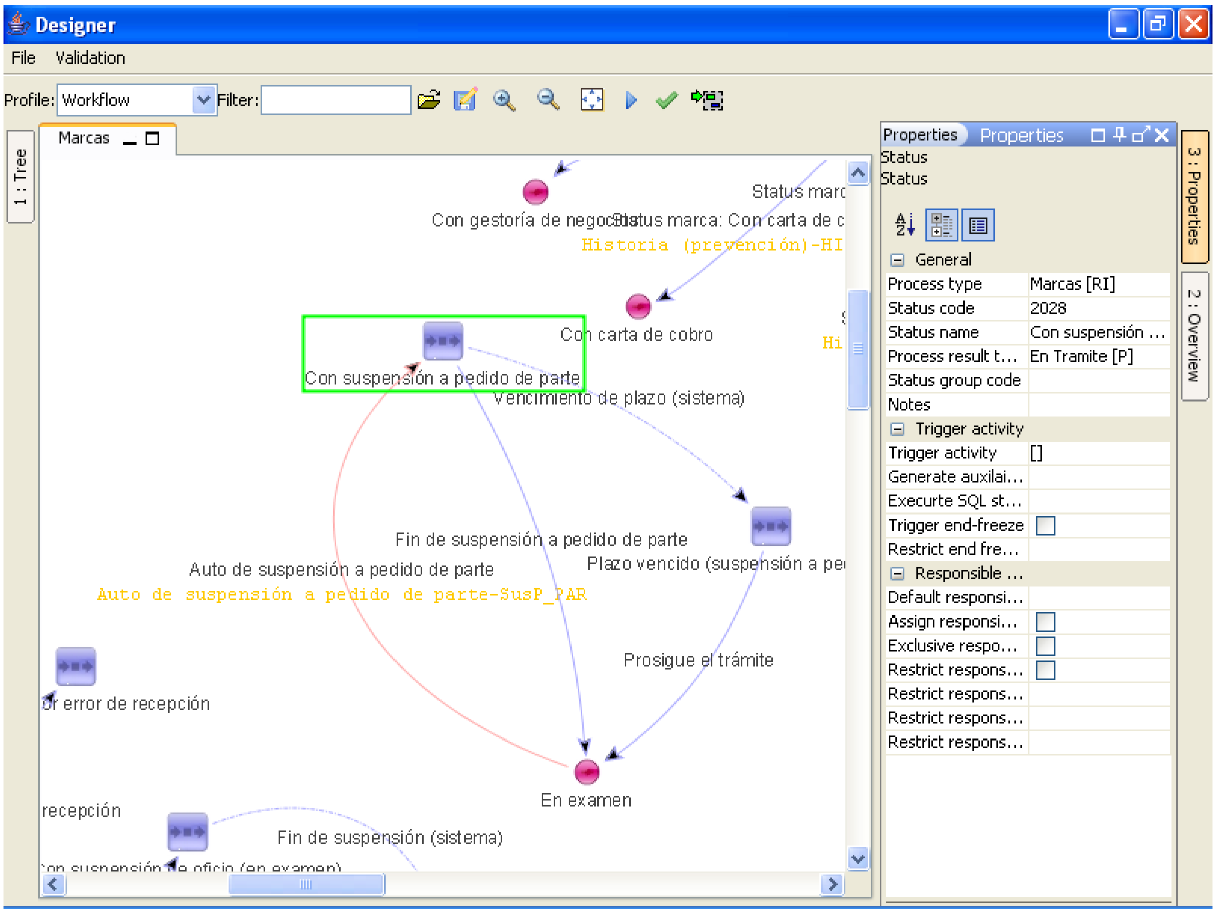


User document type
- Mandamiento judicial
- Adicional genérico
- Aclaración
- Aclaración (en anotación)
- Anotación genérica
- Recurso apelación
- Recurso apelación (en anotac
- Recurso apelación (en oposici
- Boleta de observaciones
- Cancelación voluntaria
- Cancelación por generalizació
- Cancelación por falta de uso
- Cambio de Domicilio
- Solicitud de duplicado de certi
- Solicitud de duplicado de certi
- Solicitud de certificación
- Solicitud de certificación (en a
- Licencia (levantamiento)
- Cambio de Nombre
- Oposición (titular contesta tra
- Contesta prevención
- Contesta prevención (en ano
- Contesta prevención (en opo
- Contesta traslado de impugna
- Pago de derechos (hist)
- Desistimiento
- Desistimiento (en anotación)
- Desistimiento (en oposición)
- Embargo

User document type
User document type

General	
User document type	CN
User document name	Cambio de Nombre
User document group name	Anotaciones
Doc nbr uses doc sequence	Anotación [11]
Inactive	☐
Prompt for required notes	
Linkage	
Affects file	✓
May affect multiple files	✓
Affects another user document	☐
Response to office doc	☐
Reception triggers	
Auto create	✓
Generate process	Anotaciones [MA]
Freeze affected file process	☐
Freeze prevents new office do...	☐
End freeze at end	☐
Generate action type	
Expected status of upper process	
Approval triggers	
Changes ownership	✓
Changes representative	☐
Renewal	☐
Action on auxiliary register	[]
Affected auxiliary register type	
Origin	

The following image is a printout from the Designer module, showing the workflow diagram and the properties which configure how IPAS behavior for a specific status:


Designer
File
Validation
Profile:
Filter:
1 : Tree
Marcas
Properties
3 : Properties
2 : Overview
Con gestoría de negocios
Status marca: Con carta de c
Status marc
Historia (prevención)-HI
Con carta de cobro
Con suspensión a pedido de parte
Vencimiento de plazo (sistema)
Fin de suspensión a pedido de parte
Auto de suspensión a pedido de parte
Auto de suspensión a pedido de parte-SusP_PAR
Plazo vencido (suspensión a pe
Prosigue el trámite
or error de recepción
En examen
recepción
Fin de suspensión (sistema)
con suspensión de oficio (en examen)

Profile: Workflow

Properties
Status
Status

General	
Process type	Marcas [RI]
Status code	2028
Status name	Con suspensión ...
Process result t...	En Tramite [P]
Status group code	
Notes	
Trigger activity	
Trigger activity	[]
Generate auxilai...	
Execurte SQL st...	
Trigger end-freeze	☐
Restrict end fre...	
Responsible ...	
Default responsi...	
Assign responsi...	☐
Exclusive respo...	☐
Restrict respons...	☐
Restrict respons...	
Restrict respons...	
Restrict respons...	

This graphical tool plays a key part in the flexibility of IPAS Java, since allows to design a model of the workflow, indicating all the statuses, the transitions between them produced as a consequence of the recording of actions, the office documents that will be generated at each stage, etc. In order to give an idea of the flexibility of the IPAS customization tools, it should be noted that the IPAS Java database uses some 100 tables for storing the actual data of the trademarks, patents, user documents and so on, but also devotes another some 100 tables to configuration data.

4.7 Madrid Import Module

The current Madrid import module is a .NET program developed in C#, which uses the IPAS Java application server for accessing and updating IPAS Java information. This module is in the process of being converted to Java so as to integrate it into the *IpasAdmin* module.

The Madrid module automatically downloads the electronic notifications which are made available by the International Bureau from an FTP site, and uses the information contained in those electronic notifications to insert the new trademark applications to be processed by the Industrial Property office.

All Madrid transactions are supported, and the system treats them as electronic filing requests as if they had been filed at the reception desk on the date of notification of the corresponding electronic gazette.