CERN-IT-2000-001
15 February 2000

Implementing Problem Resolution Models in Remedy

Miguel Marquina, Raúl Ramos, José Padilla
CERN, IT Division, User Support Group

This paper defines the concept of Problem Resolution Model (PRM) and describes the current implementation made by the User Support unit at CERN.

One of the main challenges of User Support services in any High Energy Physics institute/organization is to address solving of the computing-related problems faced by their researchers. The User Support group at CERN is the IT unit in charge of modeling the operations of the Help Desk and acts as a second level support to some of the support lines whose problems are receptioned at the Help Desk.

The motivation behind the use of a PRM is to provide well defined procedures and methods to react in an efficient way to a request for solving a problem, providing advice, information etc. A PRM is materialized on a workflow which has a set of defined states in which a problem can be. Problems move from one state to another according to actions as decided by the person who is handling them. A PRM can be implemented by a computer application, generally referred to as Problem Reporting Management System (PRMS). Through this application problems can be effectively guided through the states of the workflow by applying actions on them. This automatic handling improves problem resolution times and provides flexible incorporation of the problems in the workflow (either by email, the helpdesk operator etc.). It also provides registration and accounting of problems including the creation of a knowledge base, reporting, performance measurement, etc. For such implementation we have used Remedy Action Request System® (ARS), which is the current choice of the IT Division at CERN for a PRMS. Remedy ARS is a specialized development system to create PRM applications. We have developed a complete Remedy ARS application to implement the User Support PRM. Also, we have created complementary tools for reporting, statistics, backups, etc.

The aim of this paper is to explain all these concepts and the main issues behind their implementation.

1. Problem Resolution Models.

A Problem Resolution Model (PRM) is a defined strategy to handle the *action requests* arriving to the problem-solving section of an organization. Whithin IT's HelpDesks, **action requests** are problems that may arise regarding the usage of the computing facilities.

A PRM defines and states the ways in which an action request is going to be treated all along its lifecycle until its resolution, providing the means to facilitate the flow of information generated by the interventions required to address each action request. It also defines the different layers that are involved in this process (the different *escalating levels*), integration between layers, information required at each stage, integration with other systems etc. It may also include a quality control plan, such as *Service Level Agreements*, establishing the measures and criteria to audit the efficiency of the model. A complete proposal for a Problem Resolution Model is formulated in **[1]**.

A PRM is often specified using a flow diagram or *workflow* describing *states*, *transitions* and *actions* an action request may go through.

State. A particular condition (or status) of the problem inside a workflow. In every moment each problem is in one and only one state.
Transition. Constitutes a change of state of an action request.
Action. Event (or set of events) associated to a certain transition which occur when that transition is triggered.

To illustrate these concepts, figure 1 shows the current workflow used internally by the IT User Support unit at CERN.

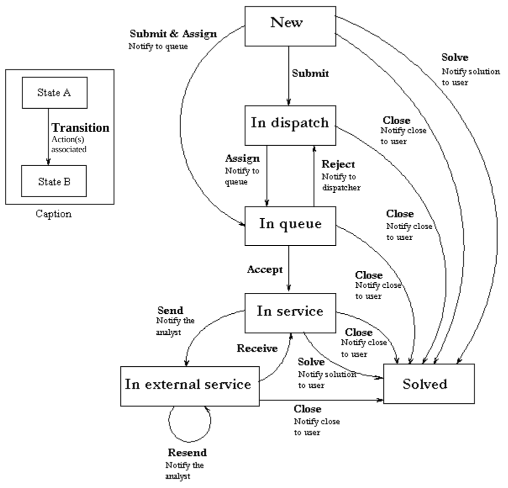


Figure 1. The current User Support PRM workflow.

For instance, the state **`In dispatch`** represents an action request which has been entered into the system but has not been assigned to any second level expert yet. The state **`In`**

queue represents an action request which has been assigned to a second level expert so that it becomes part of his list of action requests to address. The ***Assign*** transition between those two states represents the act of choosing a second level expert line and assigning (escalating) the action request to it. It also triggers certain actions which consist on notifying by email the experts now in charge of this action request.

2. Practical Implementation of PRMs

Operations following a certain workflow may be performed with the help of a computer system. Typically computer systems implementing PRMs consist of a database containing action requests and a set of user interfaces through which anyone involved in the resolution of an action request can trigger transitions, manipulate information concerning action requests, etc. These systems range from simple database manipulation packages to complex architectures to account for different logistics such as security levels for different people solving action requests, integration with other systems, customizable workflows, etc. In a general way their aim is to systematize the handling of action requests, improving its efficiency, rationalizing it in an organized manner and offering high quality data regarding the operational functioning of the overall system. A non exhaustive list of desirable requirements for such a system would include:

- Overall Ease of Use.
- Integration with other knowledge bases and information sources (FAQs, user and organizational databases, etc.)
- Integration with other systems (such as viewing action requests through the www, automatic feeding of action requests through email messages, etc.)
- Customizable layouts providing different views over the same information.
- Customizable security on the basis of users of the system and action requests (i.e. rramos can only act on problems which are ***InDispatch***).
- Prioritization of problems, with automatic calculation of target resolution times and escalations, including customizable alarms, etc. triggered upon definable criteria.
- Customizable progress reporting and Service Level Agreement monitoring (to measure the efficiency of the different people involved in the resolution of an action request).

3. Introduction to Remedy ARS

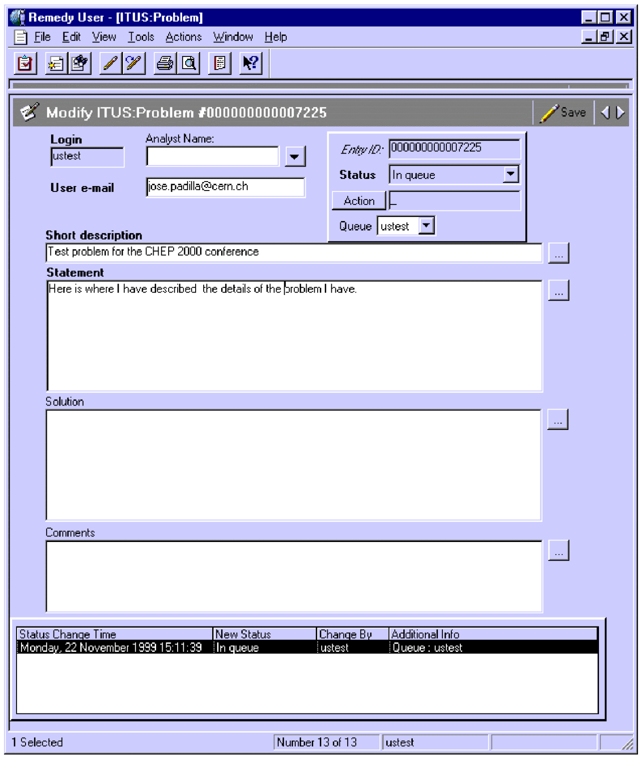
The current choice of CERN's IT Division for such a system is Remedy Corporation's *Action Request System® (ARS)*. This section succinctly describes Remedy ARS **[2]**.

Remedy ARS does not implement any particular workflow, but it is an specialized development system to build computer applications supporting operations on workflows defining the life cycle of action requests as has been described.

3.1 Implementing workflows in Remedy ARS.

Remedy ARS is a very particular development system. Since it is aimed at handling action requests in a generic way, it offers its own set of objects, which are used as building blocks when implementing any workflow. Thus, virtually any workflow can be implemented by combining accordingly these objects:

Forms are the main components users interact with. A form defines both the *fields* an action request is made of and the *graphical layout* of those fields. With the Remedy ARS we can design forms graphically and the underlying database structure is created automatically. Then, the form is used to handle actual action requests and the system takes care of interacting accordingly with the underlying database. Different forms define different kinds of action requests which may be handled differently. The same Remedy ARS may hold several forms through which different PRMs may be implemented. Fields within a form can be filled in manually or indirectly through definable callbacks to other forms or an SQL database.


Remedy User - [ITUS:Problem]
File Edit View Tools Actions Window Help
Save

Modify ITUS:Problem #000000000007225

Login	ustest
Analyst Name:	
User e-mail	jose.padilla@cern.ch
Entry ID:	000000000007225
Status	In queue
Action	
Queue	ustest
Short description	Test problem for the CHEP 2000 conference
Statement	Here is where I have described the details of the problem I have.
Solution	
Comments	

Status Change Time	New Status	Change By	Additional Info
Monday, 22 November 1999 15:11:39	In queue	ustest	Queue : ustest

1 Selected | Number 13 of 13 | ustest

Figure 2. The IT/User Support base form.

Figure 2 shows an example of a form. Notice that one field defines the state of an action request. This field is mandatory. The rest of the objects Remedy ARS provides are used to define the transitions and actions of the workflow. *Filters*, *active links* and *escalations* are defined by specifying a condition and a set of actions to perform whenever that condition is fulfilled.

Filters whenever an action request is modified through a form and the information is sent to the underlying database, Remedy checks the filters associated to that form and triggers the actions defined in those whose conditions are fulfilled. For instance, a filter having ***`state='In Queue'`*** as condition and ***`retrieve email address from DB; notify via email to assignee`*** as actions will send the correspondent email whenever an action request enters the In Queue state.

Active Links are similar to filters but they are checked and activated while during the interaction with a form. They are typically used to check or retrieve information as users interact with a form. For instance we can set an active link to display a warning if we try to solve an action request without providing a solution. In this case the condition would be ***`state=Solved AND solution=""`*** and the actions would be ***`display warning AND abort submission`***.

Escalations are similar to filters but instead of being fired whenever the information is sent to the underlying database, they are fired at regular time intervals, such as cronjobs or scheduled tasks.

The principle behind these three kinds of objects are the same: there is a condition checked at certain times (upon submission of information, interaction with a form or scheduled times) and if the condition is fulfilled then a set of actions is executed. Remedy ARS allows a wide range of actions to be programmed such as email messages, on-screen notification, SQL queries and updates, running processes, etc.

Using the above objects, Remedy ARS allows to take a "rapid prototyping and development" approach to the implementation of workflows. Developing a workflow in Remedy accounts to creating combining objects of the kinds described above. However, it is important to keep in mind that workflow implementations can become quite complex and Remedy does not offer appropriate tools to manage implementations with large collections of objects.

Next section explains the architecture of ARS. Besides that, Remedy also offers some ready to use implementations, e.g. Helpdesk 4.0, SLA, Asset Management, etc. These applications have also been designed with Remedy ARS (i.e. by combining the objects above) and they intend to cover the more common needs for business relating to their problem solving structure. It should be noted that if we use these "ready to use" applications we are implicitly also accepting the PRM that they implement, which may or may not adapt to your working strategy. These applications can be heavily configured and they stand as an intermediate solution between implementing a workflow and configuring an existing one. However, some of these implementations offered by Remedy are quite complex and their deployment also implies a significant amount of resources to maintain them. Adapting those systems or implementing your own becomes a critical issue to evaluate whenever adopting Remedy's ARS.

Because of the problems stated above plus the belief that first is the model, then the implementation of it (and not the opposite), we decided to design our own implementation of the workflow we use.

An evaluation of Remedy's Problem Report Management System can be found in **[3].**

3.2 The Remedy ARS Architecture.

The architecture of the complete system is a three-tier Client-Server model, as shown in figure 3. The client layer is provided through specific tools Remedy provides. The *Remedy User* client executes forms and active links so it is actually the tool through which users perform operations in action requests according to a certain workflow. The *Remedy Administrator* client **[4]** is the development tool used to create and manipulate the objects implementing a certain workflow.

All Remedy ARS objects and data are stored in a *Remedy ARS Server*. Client tools access the Remedy ARS Server through proprietary protocols. The Remedy ARS Server interacts with a backend *database server* which is takes care of finally storing the data about the defined objects and about the action requests.

The Server layer is composed by the Remedy ARS Server and the Database Server. All the information flow is handled by the Remedy ARS Server which control the flow of data to/from the client(s), checks access permissions, fires filters and escalations and finally interacts with the backend database server. Clients only interact with the Remedy ARS Server and all database transactions are performed by the Remedy ARS server, which hides all of the database complexities from the developer and lets him/her concentrate on the process to be automated. The Remedy ARS server is able to interact with several database backends such as Oracle, Informix, Sybase, etc.

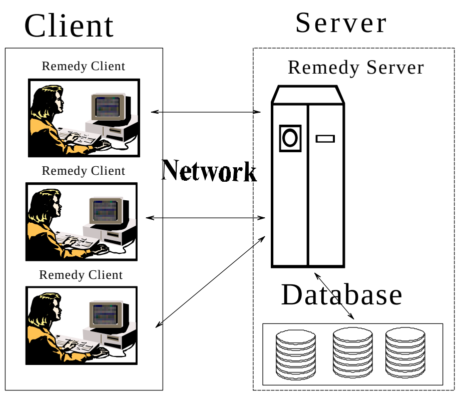


Figure 3. The Client-Server model.

The system is relatively scalable since two or more Remedy ARS Servers can interact across the network using the Distributed Server Option.

As stated previously, the Remedy ARS Server has a security control system for keeping the information accessible only to the users that should handle it. For that there are three types of access with different permissions:

Administrator.
User with privileges to act on any Remedy ARS object in the Server. He has also access to two administrative forms named User and Group from which he can manage all the permissions and access control for the users and groups in the server.

Subadministrator.
User with privileges to act as administrator only in the objects that belong to the group(s) he administers. A subadministrator is usually a developer of Remedy ARS objects. The access to the administrative forms is not allowed.

Normal User.
User who only has access privileges when the group to which he belongs is given these privileges. The access permissions have to be set for every component of every object to which they need access.

4. The User Support role in the PRMS at CERN.

The IT division at CERN is in charge of handling the problems that the HEP scientists may have regarding the usage of the computing facilities it offers. In order to address this, there is a defined strategy **[5]** which consists in different layers (or lines) of support as figure 4 shows. The entry point is the Help Desk, which may be accessed by email, phone or personal visit. If the problem can be solved by the Helpdesk it will be handled there and won't go on to the next line of support. If the helpdesk cannot solve the problem, at least it will be classified and sent to the correct branch in the second line of support specialized in his problem type, which will get in touch with the user directly (no more helpdesk intervention from this point on). Eventually this support line will need help of more skilled persons and will escalate the problems to other lines of support.

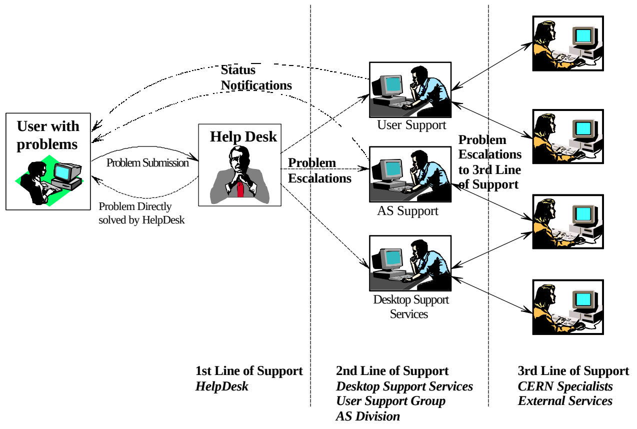


Figure 4. Overview of the computing PRMS at CERN

There is a way to submit a problem directly to the second support line, bypassing the Help Desk entry point. This happens when the user "self-classifies" his problem and sends it to the corresponding support line. This is done by sending the problem by e–mail to service.support@cern.ch where *service* can be any of the second line of support categories or sub-categories (ie. www.support, mail.support etc.)

In this context is where the User Support group at CERN is, as a part of the second line of support. The workflow is adapted to the role the group plays in the PRMS, and the interrelations among the group, its users and its external analysts (third line of support) have been defined.

4.1 Implementing the User Support PRM in Remedy ARS. The problem solving staff view.

Once a problem is inside our workflow, the problem solving staff who will be in charge of it receive an email notifying the event, as well as the user, who is given the reference ID number of his problem, which univocally references the problem inside Remedy ARS. These notification actions are sent as it is specified in the correspondent filters in our implementation. When the problem solving staff get a notification, looks for the problem opening the form shown in figure 2 by using the Remedy User client. Using this graphical user interface the problem solving staff have the access to the information that is available about the problem, which is stored in a database and is retrieved and showed on the screen. The Remedy User takes care of interacting accordingly with the Remedy ARS Server following the definitions of all our objects. This way the problem solving staff can effectively handle the action request being transparent to them the complexity of the underlying database and the maintenance processes that help them guiding the problem through the workflow. The meaning of the field is self-descriptive. As we stated previously, each action request has an identifier which uniquely identifies it inside the Form, which is the *Entry- ID*.

Through the Remedy Client the problem solving staff may guide the problems through the workflow. The program knows what transitions can be performed in every moment, as stated in the workflow, and only allows the to apply the suitable transitions depending on the current state of the problem. A complete description of the details of the implementation is stated in **[6]**

4.2 Additional / complementary tools.

Also, we have designed other auxiliary tools to fill in gaps in the functionality that Remedy ARS provides.

Examples of those unsatisfactory features are the notification and reporting systems. The default notification mechanism that Remedy supplies is based on a small proprietary program (*Remedy Notifier*) that recipients of notifications must be running at the time notifications are issued. This is not very satisfactory since notifications may be lost if the Remedy Notifier is not open. Alternatively, Remedy ARS can also send notifications through regular email messages but offer limited formatting capabilities. To overcome this limitation we implemented a simple mail gateway mechanism, which reformats notifications according to user definable templates.

Also, the reporting capabilities of Remedy ARS are quite limited both in formatting and functionality. The built-in reporting option is very poor and, although it allows interaction with Crystal Reports for pretty reporting we are still missing an automatic (schedulable) reporting capability. To fulfill this need we also created a simple script-based system **[7]** which queries directly the underlying database and formats the results according to user definable templates. With this we can schedule the script to run at predefined times and we have timely automatically generated reports.

Finally, we also had to address the generation of backups and version control. Again, Remedy ARS does not provide any batch facility to automatically create back ups on a per-workflow basis. Only the full backup of the underlying database is available, but restoring data concerning only one workflow in a given server implies affecting any workflow living in that server. Again, with a simple scripting-based systems we were able to create and restore backups on an automatic (as scheduled tasks and autonomous way). Regarding version control, Remedy ARS does not offer any way to keep accounting of the modifications performed on objects, or even on an implementation as a whole. In order to keep developments under control we had to set forth mechanisms to **(1)** extract objects' definitions automatically from the Remedy Server and insert them into a CVS repository and **(2)** insert objects' definitions extracted from the CVS repository into the Remedy Server.

5 Conclussions.

In this paper we have defined the concept of Problem Resolution Model (PRM) and described the characteristics a PRM should attain.

Also, we have explained the current implementation of a PRM made by the User Support unit at CERN. This implementation is based on Remedy Action Request System, whose main characteristics we have shown.

Finally we have explained other auxiliary tools we designed to fill in gaps in the functionality that Remedy ARS provides.

6 References.

[1] M. Marquina, "A Problem Resolution Model ", Technical Report CERN-IT-US-1997-004. March 1997.

[2] Remedy Corporation, "Action Request System 4.0 Concepts Guide". December 1998

[3] A. Ballaminut and R. Ramos, "Evaluation of Remedy's Problem Report Management System", Technical Report CERN-IT-US-1998-005. July 1998.

[4] Remedy Corporation, "Action Request System 4.0 Administrator's Guide Volume 1, Using Remedy Administrator". December 1998.

[5] A. Lovell, "The IT Problem Report Management System" http://www.cern.ch/wdc/meeting/130499/alan_remedy.ppt

[6] A. Ballaminut, J. Padilla and R. Ramos, "IT US Help Desk implementation with Remedy AR System" Technical Report CERN-IT-US-1999-046. November 1999.

[7] J. Padilla and R. Ramos, "Automatic Remedy Reports" Technical Report CERN-IT-US-1999-028. April 1999.