# Updating Inventories with Intelligent Agents

Mary Luz Mouronte López[1,2], Francisco Javier Ramos Gutierrez[2]
Department of Telematic Engineering
[1]Universidad Carlos III de Madrid
[2]Ericsson España
Madrid, Spain
mmouront@it.uc3m.es, mary.luz.mouronte.lopez@ericsson.com

*Abstract*—**This paper describes a procedure to automatically solve misalignments between inventories by means of expert agents. We apply this mechanism to maintain the repository of a Network Manager System (NMS), which allows executing different tasks over a transmission network in a Telecommunication Operator: fulfillment, supervision, etc. The consistency in the NMS inventory is maintained by means of connections to corporate systems (repository, planning, and assignment systems) and operations over Network Element Managers (NEMs) regarding to deployment of Network Equipments (NEs) or cards, setting up of circuits, paths and physical links. This repository contains data about managed elements, Synchronous Digital Hierarchy (SDH), Ethernet and Wavelength Division Multiplexing (WDM) components, and not managed elements: Radio, Plesiochronous Digital Hierarchy (PDH), fibers, etc. Sometimes misalignments occur between the corporate inventory and the repository in NMS. These misalignments have to be solved by the technicians. Our method incorporates the necessary intelligence to analyze failures and the mechanisms to solve them in expert agents, therefore preventing technicians from checking and solving main errors. It reduces Operating Expenditures (OPEX).**

***Keywords - Transmission Network, NMS, NEM, NE, Expert Agents, CLIPS***

## I. INTRODUCTION

In this paper we describe a method to solve misalignments between inventories by means of intelligent agents. This procedure is applied to a Transmission Network Management System (NMS) where the agents incorporate business information used by technicians when they do tasks manually. The solution both improves reliability and reduces human intervention (lower Operating Expenditures).

In a Telecommunication Operator, the NMS executes all business processes related to the transmission network [8] and its services across different vendors: fulfilment, supervision and performance monitoring in the network. It connects to corporate systems in order to automate routine tasks for fulfillment and to update the information about not managed elements. It also connects to vendor-specific **N**etwork **E**lement **M**anagers (NEMs).

This NMS manages **S**ynchronous **D**igital **H**ierarchy (SDH), **W**avelength **D**ivision **M**ultiplexing (WDM) and Ethernet over both SDH and **W**avelength **D**ivision **M**ultiplexing (WDM) networks. Other elements included in the transmission network but not related to management technologies (Plesiochronous Digital Hierarchy network elements, routers, fibres and radio elements) are also registered in the NMS to provide end-to-end vision.

The rest of the paper is organized as follows: Section II details relevant related works, Section III gives an overview about the NMS inventory and its current situation. Proposed technical solution is described in Section IV. Section V summarizes the results of applying this solution while Section VI describes main conclusions and areas for future works.

## II. RELATED WORKS

There are not previous works on applications using expert agents to solve inventory problems on the transmission network from a NMS. Nevertheless, similar methods have been developed in others areas such as:

- Plant asset management where various research approaches and systems in this area exist [3].
- Detecting and preventing SQL injection attacks using **G**ene **E**xpression **P**rogramming (GEP) with intelligent agents in Web applications [9].
- Supply chains where a community of autonomous, intelligent, and goal oriented units cooperate and coordinate their decisions to reach a global goal [7].
- Studies to identify key concepts in different expert agent frameworks [10].

## III. OVERVIEW

The main features of this NMS are:

- Network model based on standards like ITU-T G.803, ITU-T G.805, ITU-T G.709.
- It provides a vendor-neutral unified network management.
- It connects to the vendor **N**etwork **E**lement **M**anagers (NEMs). The interaction within the plant is carried out using the northbound interfaces offered by the NEMs which in turn deal with the **N**etwork **E**quipments (NEs).
- End to end control of the whole transmission network.
- Network inventory, which establishes relationships between the network vision in the network managers and the existing one in corporate systems. It also offers auditing and discovery mechanisms.

- Complete functional support in the following areas: fulfillment (network deployment and provisioning), supervision and performance monitoring. The fulfillment function allows executing operations over the NEMs regarding NEs, cards, circuits, paths and physical links.
- Simple user interface, based on well-known web technologies, defining operating profiles adapted to each user and enforcing a strict security policy.
- Connection to corporate systems in order to automate most routine tasks for fulfillment and to update information about not managed plant.

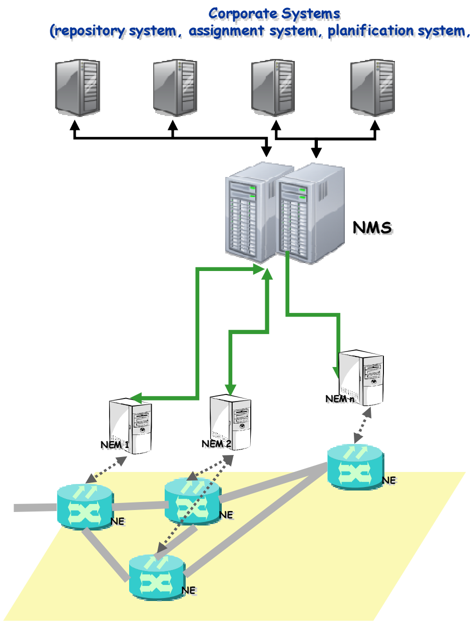
Corporate Systems
(repository system, assignment system, planification system, etc)
NMS
NEM 1
NEM 2
NEM n
NE
NE
NE
NE

Figure 1. Transmission NMS

### A. Inventory interface between the NMS and corporate system

The NMS is a system built upon a standards-based network model, backed by an ORACLE DBMS [2], with a business logic layer that allows interacting with the core applications through a CORBA bus.

The NMS receives information about transmission network elements, ports, paths, circuits, link connections, fibers and administrative data from a corporate repository, which contains all deployed plant, managed or not by NMS.

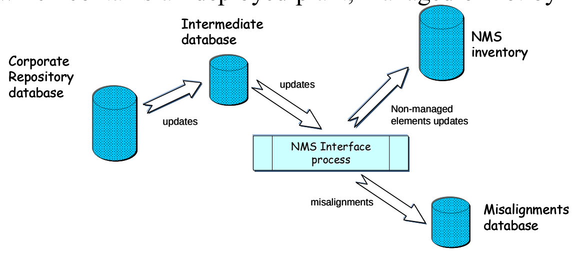

Figure 2. Inventory interface

This repository is a network inventory which uses proprietary operator names to label its elements.

In order to update the NMS inventory, a database-to-database interface has been implemented between the corporate repository and the NMS (fig. 2). This interface works the following way:

- The corporate repository captures each update generated in its database and inserts it into an intermediate database through ORACLE Net.
- A process in NMS reads the updates from this intermediate database and transforms it according to the NMS model.
- A NMS interface process establishes the operation to be performed. If the update affects to a not managed element, the NMS inventory is updated. Otherwise, it determines if received data is consistent with stored data in its inventory. If it is inconsistent, an error is generated and inserted into a misalignments database.
- NMS provides functions to search misalignments and find relationships among them. By means of this set of operations, misalignments are solved manually by technicians. There is a big delay until a misalignment is solved and further misalignments caused by it may appear. Moreover, some operations in the NMS are not executed (remain pending) until the misalignment is solved.

### B. Current situation in the NMS inventory

After analyzing misalignments in this interface between the NMS and the corporate repository during two months, 1300 errors were found. The percentage of each class error over the total failures is showed bellow (per entity basis).

1. Misalignments in circuits,
    - Link capacity is not registered in the NMS (30%): It does not exist in the path; has different structure or administrative properties.
    - Port is not assigned to the circuit in the NMS (30%): circuit is incomplete or has different administrative properties.
    - Link capacity is not assigned to the circuit in the NMS (22%): circuit is incomplete or has different administrative properties.
    - Port is not registered in the NMS (5%): card or port has different number, tasks have not been done in the NMS or port is in a card depending on other one.
    - Link capacity is not assigned to the circuit in the corporate system (4%): circuit is erroneous in the NMS.
    - Port is not assigned to the circuit in the corporate system (2%): circuit is wrong in the NMS.
    - Circuit does not exist in the corporate repository (0.5%): some operations were not done in the NMS.
    - Others (6.5 %)

2. Misalignments in paths,
   - Path does not exist in corporate repository (30%): some operations over the path were not done in the NMS.
   - Port is not assigned to the path in the NMS (18%): path is incomplete or has different administrative properties.
   - Port is not registered in the NMS (18%): card or port has different number, tasks have not been done in the NMS or port is in a card depending on other one.
   - Link capacity is not assigned to the path in the NMS (15%): path is incomplete or it has different administrative properties.
   - Link capacity is not assigned to the path in the corporate system (10%): path is erroneous in the NMS.
   - Port is not assigned to the path in the corporate system (5%): path is erroneous in the NMS.
   - Link capacity is not registered in the NMS (2%): Link capacity does not exist in the path; has different structure or administrative properties.
   - Other (2 %)
3. Misalignments in link connections,
   - Link connection does not exist in the NMS (42%): it has different structure.
   - Link connection does not exist in the corporate system (34%): it has unlike structure.
   - Link connection is occupied in the NMS and not in the corporate system (7%): occupancy is incoherent.
   - Link connection is occupied in the corporate system and not in the NMS (10%): occupancy is incoherent.
   - Different link connection type in the corporate system and in the NMS (6.5%): it has different structure.
   - Link connection is not in service in the NMS (0.5%): it is included in a pending operation.
4. Misalignments in cards,
   - Card does not exist in the NMS (60%): card number is different or some operations have not done in the NMS.
   - Card in service in corporate system (21%): it is include in a pending operation.
   - Card does not exist in corporate system (18%): some operations have not done in the NMS.
   - Different card number in the NMS and in corporate system (1%): some tasks have not done in NMS.
5. Misalignments in ports,
   - Card does not exist in the NMS (42%): card number is different or some operations have not done in the NMS.
   - Port does not exist in the NMS (31%): card has unlike number, some tasks have not done in the NMS or port is in a card which is depending on other card.
   - Port does not exist in corporate system (19%): tasks have not done in the NMS.
   - Different port position (6.5%): some operations have not done in the NMS.
   - Port in different card (1%): port is assigned to a depending card.
   - Port in service (0.5%): port is included in a pending operation.

Depending on the error, a technician needs to execute some tasks:
- Get additional data from the corporate repository, the NMS inventory and NEMs.
- Analyze obtained data.
- Decide which operations to perform in the NMS inventory.

The actions which can be executed in the NMS inventory are: modify (change its link connections, ports, structure, state) or remove the circuit or path, create or remove the card, change the number of the card, modify the state of the card, create or remove the port, change the number of the port and modify the state of the port.

Technicians have to spend much time analyzing error causes, comparing both inventories and solving misalignments. On a daily basis, the average spent time is about several minutes per failure depending on the error type.

TABLE I. TIME FOR ANALYZING AND SOLVING MISALIGNMENTS BY TECHNICIANS

| **Paths and Circuits** | |
|---|---|
| ***Misalignment type*** | ***Average time (minutes)*** |
| Link capacity does not exist or is not assigned in NMS | 13.70 |
| Port does not exist or is not assigned in NMS | 13.70 |
| Circuit or path does not exist in the corporate repository | 7.30 |
| Link capacity is not assigned to the circuit path in the corporate system | 11.30 |
| Port is not assigned to the circuit or path in the corporate system | 11.30 |
| **Cards** | |
| ***Misalignment type*** | ***Time (minutes)*** |
| Card does not exist in NMS | 15 |
| Card in service in corporate system | 3 |
| Card does not exist in corporate system | 1.10 |
| **Link Connections** | **Average Time (minutes)** |
| All types | 6.10 |

| Ports | Average Time (minutes) |
|---|---|
| All types | 4.90 |

## IV. SOLUTION

The proposed solution in this paper, allows supervising the errors in the interface between the NMS and a corporate repository.

Our technical solution uses a knowledge base which contains rules to represent the actions executed by a technician when an error occurs. A method based on ORACLE queues is used to capture and notify asynchronous errors in NMS. In the NMS, CORBA operations are implemented to get data from NEMs and to perform tasks in the NMS inventory.

The advantages of the implemented solution are:

- Changes are not required in the interface between the NMS and the corporate repository. Errors are processed in background mode and there are not delays by the processed failures.
- Errors are processed as soon as they happen. Further misalignments are not generated by a previous one.

The architecture of the solution is shown in Fig 3.

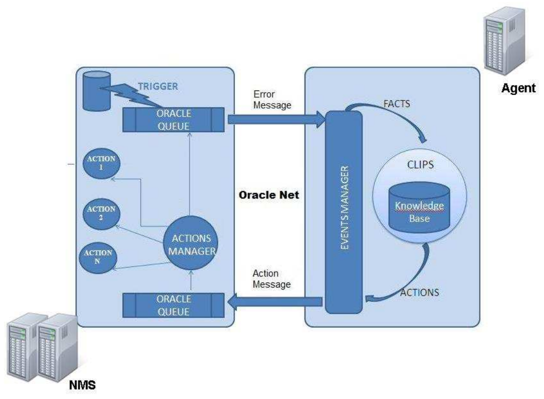

Figure 3. Architecture of the technical solution

**C L**anguage **I**ntegrated **P**roduction **S**ystem (CLIPS) is an open source expert system tool developed by NASA-Johnson Space Centre [1]. It is fast, efficient, free and updated and supported by the original author, Gary Riley.

The components in the architecture are: intelligent agents (*Event Manager* and *Action Manager*) and a *Knowledge Base* which contains different rules.

Intelligent Agent: *Event Manager*

In the NMS, a trigger is fired by an error in the interface and it writes an event in an ORACLE queue. The *Event Manager* (with CLIPS embedded) is connected to database through ORACLE Net. It executes the following tasks:

- Read an event from ORACLE queue. The format of the message in ORACLE queue is:
  message_type|entity_type|Id_entity|entity_name|text|element_type|Id_element|report_name
- Generate a fact and add it to the Fact Base which, is used by the Inference Engine. A fact has the following format:

```
(deftemplate misalignment
   (slot  entity_type (type STRING))
   (slot Id_entity (type STRING)))
   (slot name (type STRING))
   (slot text (type STRING))
   (slot elem_type (type STRING))
   (slot Id_element (type STRING))
   (slot report_name (type STRING))
)
```

- Run the Inference Engine which decides the actions to be performed based on facts contained in the Fact Base and the knowledge base. It contains different types of rules:
  - o Correlation rules: to establish relations between facts and to construct new ones.
  - o Decision rules about the actions: to establish the specific criteria to shoot actions.
  - o Metacontrol: to fix priorities in the rules according to their relevance.
- Send the action to an ORACLE queue. It will be processed by the *Action Manager*.

  The event format is:

  Action_type|entity_type|Id_entity

Intelligent Agent: *Action Manager*

This agent executes the actions generated by the Inference Engine and interacts with the NMS to perform the suitable operations. This intelligent agent read the actions from ORACLE queue and executes the specific tasks in NMS. There are different operations:

- Data acquisition operation: It is executed to get data from the corporate repository, the NMS inventory or NEMs. It can be a search in database or a complex CORBA operation.
- Final operation: It is a task to solve an error or to require a technician. It can be a transaction in a database or a complex CORBA operation.

In case of data acquisition operations, when the task is finished, the *Action Manager* sends a message to the *Event Manager* to inform about the result of the task. The message format is:

```
Message_type|entity_type|Id_entity|action|value
```

When the *Event Manager* receives this message, a new fact is generated and added to the Fact Base. The format of this fact is shown below:

```
(deftemplate condictions
   (slot entity_type (type STRING))
   (slot Id_entity (type STRING))
   (slot type (type STRING))
   (slot value (type STRING))
)
```

This process of adding facts to the Fact Base is repeated until a final action is obtained.

A particular case is shown below: misalignments due to a port not assigned to the path in the corporate system

1. *The Event Manager* receives the following event from the ORACLE queue:

   MISALIGNMENT|CIRCUIT|265792011|2/AB.LL ETH 7/2/AB.PA008ETH 1/ETH0/0001/NIGCM /|Port is not assigned to the circuit |020001003000089S000189|MISALIGNEMT IN CIRCUIT

2. *The Event Manager* adds a fact to the Fact Base:

   misalignment(entity_type "CIRCUIT") (Id_entity 265792011) (entity_name "2/AB.LL ETH 7/2/AB.PA008ETH 1/ETH0/ 0001/NIGCM /|") (text " Port is not assigned to the circuit " )(type_element "Port")(Id_element "020001003000089S000189")(report_name " MISALIGNEMT IN CIRCUIT")

3. The Inference Engine decides the required actions to be sent to the *Action Manager*

```
(defrule misalignemt_circuit
?h1 <- (misalignment  (entity_type
?ent_type) (Id_entity ?Id) (entity_name
?name) (text " Port is not assigned to the
circuit " )(type_element
"Port")(Id_element ?elem)(report_name
?report_name))
      (not(process_misalignment
(entity_type ?entity_type) (Id_entity
?Id_ent) (name ?previous_name) (type
?type) (pending_elem_number ?number)
(depending_entity_type
?depending_ent_type)
(depending_entity_Id ?depending_ent_Id)
))
=>
(retract ?h1)
(assert ( process_misalignment
(entity_type ?entity_type) (Id_entity ?Id)
(name ?name)( adaptation type)
(pending_elem_number 0)
(depending_entity_type "")
(depending_entity_Id "") ))
(send_action "there_is_request"
?entity_type ?Id)
```

4. The following message is read from the ORACLE queue by the *Action Manager* :
   there_is_request|CIRCUIT|265792011
5. *The Action Manager* executes the necessary actions in the NMS and sends a new message to the *Event Manager* through the ORACLE queue:
   CONDITION|CIRCUIT|265792011|there_is_request|NO
6. A fact is generated and added to the Fact Base:
   condition (entity_type "CIRCUIT") (Id_entity 265792011) (action_type "there_is_request") (value "NO")
7. The Inference Engine decides the required actions to be sent to the *Action Manager*

```
(defrule misalignment_no_request
?h1 <- (process_misalignment (entity_type
"CIRCUIT") (Id_entity ?Id_ent) (name
?name) (type ?type)
(pending_elem_number ?number)
(depending_entity_type
?depending_ent_type)
(depending_entity_Id ?depending_ent_Id))
?h2 <- (conditions (entity_type "CIRCUIT"
) (entity_Id ?id_ent) (type
"there_is_request") (values NO))
=>
(send_action "modify_circuit" ?ent_type
?Id)
(retract ?h1)
(retract ?h2)
```

8. The Inference Engine generates an action to modify the circuit and puts the following message in the ORACLE queue:

   modify_circuit|CIRCUIT|265792011

9. *The Action Manager* reads the message from the ORACLE queue and modifies the circuit.

## V. RESULTS

The solution was implemented in the production environment over 600 misalignments obtaining the following results:

TABLE II. TIME FOR ANALYZING AND SOLVING MISALIGNMENTS BY EXPERT AGENTS

| **Paths and Circuits** | |
|---|---|
| ***Misalignment type*** | ***Average time (seconds)*** |
| Link capacity does not exist or is not assigned in NMS | 169 |
| Port does not exist or is not assigned in NMS | 169 |
| Circuit or path does not exist in the corporate repository | 90 |
| Link capacity is not assigned to the path in the corporate system | 139 |
| Port is not assigned to the path in the corporate system | 139 |
| **Cards** | |
| ***Misalignment type*** | ***Time (seconds)*** |
| Card does not exist in NMS | 185 |
| Card in service in corporate system | 36 |
| Card does not exist in corporate system | 10 |
| **Link Connections** | **Average Time (seconds)** |
| All types | 42 |
| **Ports** | **Average Time (seconds)** |
| All types | 63 |

The solution reduces the time needed for analyzing and resolving problems. Therefore, technicians are required only when the solution depends on pending task in the NMS or produces a loss of quality of service, as shown in Table III.

TABLE III. STATISTICS

| ***Solution*** | ***Percentage*** |
|---|---|
| Misaligments solved by agents | 78% |
| Misalignments where technicians are required in final stages. Technicians decide when the tasks have to be done in network equipments in order to maintain the quality of service. | 13% |
| Misalignments where technicians are required to perform pending tasks in NMS (routing, programmed jobs, etc.) | 9% |

When we apply this method to resolve all misalignments, it can be necessary to add new rules and actions in the expert agents.

## VI. CONCLUSIONS AND FUTURE WORKS

This paper described a good solution to solve misalignments between inventories. In particular, we applied the method to solve the problems between a corporate repository and an inventory of a NMS in a Telecommunication Operator.

This method reduces operating expenditures because it is not necessary to hire technicians to resolve main errors and a solution is provided as soon as a problem is detected. The few unsolved problems can be addressed by those technicians in charge of fulfilment operations in the NMS.

The procedure can be applied to other areas:

- Adaptive graphical user interface [4], [5].
- Diagnosis of problems in different networks [6], [7].

## REFERENCES

[1] "CLIPS. A Tool for Building Expert Systems", http://clipsrules.sourceforge.net/index.html, April 2011.

[2] "Oracle9i Net Services Administrator's Guide", http://download.oracle.com/docs/cd/B10500_01/network.920/a96580/toc.htm, April 2011.

[3] "Using Multi-Agent Systems for Intelligent Plant Maintenance Functionality", http://www.tik.ee.ethz.ch/~naedele/WCICA04.pdf, April 2011.

[4] C. Mourlas and P. Germanakos, "Intelligent user interfaces: Adaptation and personalization systems and technologies", Information Science Reference, September 2009.

[5] F. Nasoz and C. L. Lisetti, "Affective user modeling for adaptive intelligent user interfaces", Human Computer Interaction. HCI Intelligent Multimodal Interaction Environments, pp. 421-430, August 2007

[6] L. Bunch et al., "Software agents for process monitoring and notification", Proc. ACM Symp. Appl. Comput., Vol. 4, pp. 94-99, 2004.

[7] J. García, P. Arozarena, S. García, A. Carrera, and R. Toribio. "A Lightweight Approach to Distributed Network Diagnosis under Uncertainty". In "Intelligent Networking, Collaborative Systems and Applications", Springer, pp. 74-80, 2010.

[8] M. L. Mouronte, R. M. Benito, and J. P. Cárdenas, "Complexity in Spanish optical fiber and SDH transport networks", Computer Physics Communications, Vol. 180, No. 4., pp. 523-526, 2009.

[9] S. Kadirvelu and K. Arputharaj. "Intelligent Agent Based Prevention System for Web Applications from SQL Injection Attacks Using Gene Expression Programming", European Journal of Scientific Research, Vol.49, No.2, pp. 286-292, 2011.

[10] W. C. Regli et al. "Development and specification of a reference model for agentbased systems.", IEEE Trans. On Systems, Man, and Cybernetics, Part C, Vol. 39, No.5, pp. 572–596, 2009.