International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:9, No:8, 2015

Multi-Agent System Architecture Oriented Prometheus Methodology Design for Reverse Logistics

F. Lhafiane, A. Elbyed, M. Bouchoum

***Abstract*—**The design of Reverse logistics Network has attracted growing attention with the stringent pressures from both environmental awareness and business sustainability. Reverse logistical activities include return, remanufacture, disassemble and dispose of products can be quite complex to manage. In addition, demand can be difficult to predict, and decision making is one of the challenges task in such network. This complexity has amplified the need to develop an integrated architecture for product return as an enterprise system. The main purpose of this paper is to design Multi Agent System (MAS) architecture using the Prometheus methodology to efficiently manage reverse logistics processes. The proposed MAS architecture includes five types of agents: Gate keeping Agent, Collection Agent, Sorting Agent, Processing Agent and Disposal Agent which act respectively during the five steps of reverse logistics Network.

***Keywords*—**Reverse logistics, multi agent system, Prometheus methodology.

I. Introduction

REVERSE logistics, is the process of planning, implementing and controlling the efficient cost effective flow of raw material, in process inventory, finished goods and related information from the point of consumption to the point of origin for the purpose of recapturing value, or proper disposal [1]. The main goal of reverse logistics is to handle any type of returns from any customer with lower costs and greatest profits even if it is very complicated and quite difficult to manage.

The reverse logistics decisions are classified into strategic decisions, tactical decisions, operational decisions, and real time decisions. In particular, companies need to choose how to collect recoverable products from their former users, where to inspect collected products in order to separate recoverable resources from worthless, scrap, where to reprocess collected products to render them remarkable and how to distribute recovered products to future customers, as is noted in [2] that are the critical decisions facing producers. In some cases, making these decisions requires an intelligent modeling methodology to capture and support the reverse logistics tasks. Therefore to cope with decision making complexity, our proposed system considers seven important elements of the RL system and it's based on the integration of different types of intelligent agents and hybrid architecture. The Multi-Agent System (MAS) as a distributed problem-solving paradigm breaks complex problems into small and manageable sub-problems to be solved by individual agent co-operatively. However, designing five agents that can work together toward a common goal is one of the challenges in the research area of agent oriented software engineering.

In fact, to develop our multi agent decision support system for reverse logistics processes, we have chosen the Prometheus methodology because compared to other agent oriented methodology and according to [3] it covers (a) start to end development stages, (b) simple method, (c) easy to understand, (d) mature or have been described in sufficient detail to be of real use, (e) provide tool support and (f) comprehensive documentation for reference.

The remainder of this paper is organized as follows: Section II begins with a literature review about Reverse logistics. The Prometheus Methodology and tools are stated in Section III. Section IV presents the design of our proposed agent system. Finally, we conclude the paper with some remarks and perspectives.

II. Reverse Logistics Literature Review

Previous studies [1], [4], [5] have defined a reverse logistics with four main steps: Gate keeping (entry), collection, sorting and disposal. The first step is the recognition of product return, this is very critical to succeed in managing the system. Reference [1] defines it as deciding which products are allowed to enter the system. The second step is the collection permits the retrieval of products from internal or external customers; here the collection may be made in several ways. Detailed sorting (or the third step) decides the fate of each returned item. At that moment, the company may decide what to do with the product, be it subject to inspection, tests, or other manipulations. The last step involves the choice of disposal, the destination of the product. These reverse logistics activities need two other important elements to be integrated as is mentioned by [4] which are shown in Section IV.*A*: an integrated information system to keep track of what's happening, and coordinating system which is responsible of the overall performance and management of the RL system.

Since most of reverse logistics activities are triggered by customer, and are hard to predict accurately, the use of conventional analytic approaches, heuristics and artificial intelligence technique might help [6], indeed developing an agent based framework to automate the RL tasks could play a key role in the successful implementation of these steps.

None of the existing agent oriented methodologies, to our knowledge, have been demonstrated enough evidence to support reverse logistic activities , only recently some initial

Fatima Lhafiane, Abdeltif Elbyed, Mohammed Bouchoum are with the Department of Mathematics and Computer, Faculty of Sciences, Hassan II UniversityBP5366, Maarif, Casablanca 20100, Morocco (e-mail: lhafianefatima@gmail.com; a.elbyed@fsac.ac.ma).

steps have been taking towards this direction. Reference [7] has developed a framework using agent system to solve the problem of prediction in RL context, also [4] has initiated work that provides conceptual framework for decision making dedicate to reverse logistics process. However, the main contribution of this paper is the MAS architecture design shall be conceptualized using Prometheus Design Tool to identify the types of agent and the agent architecture.

III. Prometheus Methodology and Tools

A. Prometheus Methodology

The Prometheus methodology is an agent oriented software engineering methodology [8], it consists of three phases; system specification, architectural design and detailed design. The first phase, system specification focuses on identifying the basic functionalities of the system along with inputs (percept), outputs (actions) and any important shared data sources. The second phase, architecture design, where the agent types identified; and how they will interact. The third phase, detailed design, where the details of each agent's internals are developed and defined in terms of capabilities, data, event and plans; process diagrams are used as a stepping stone between interaction protocols and plans. The use of the JACK development environment (JDE) is strongly recommended because both Prometheus and JACK are oriented toward BDI agent. Fig. 1 indicates the main design artifacts arising from each of these phases as well as some of the intermediary items and relationships between items.

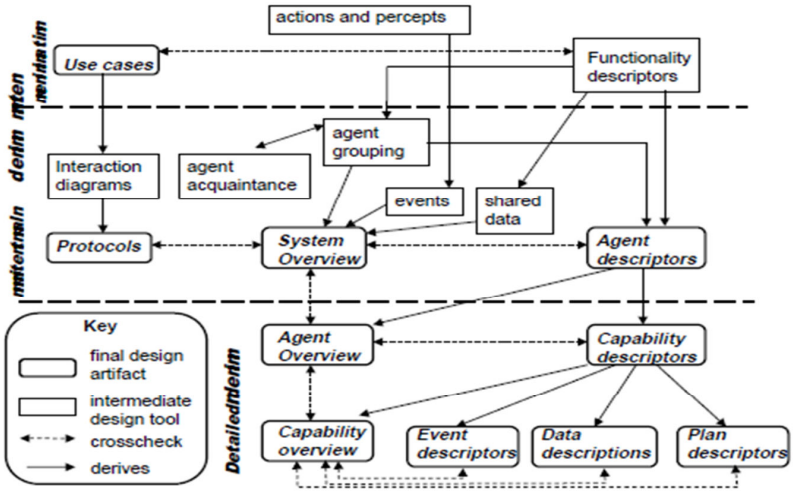

Fig. 1 Prometheus methodology phases [9]

1) System Specification Phase: The aim of this phase is to identify system's goals, develop use case scenarios illustrating the system's operation, identify the basic functionalities and specify the interface between the system and its environment in terms of actions and percepts.
2) Architectural Design Phase: The architecture design phase will use the system specification artifacts to build the system architecture; the system architecture will be developed in three main steps; first of all, the application agent types are identified; secondly, the system interaction are specified, in the third step, the system overviews are designed.
3) Detailed Design Phase: The detailed design uses the system architecture artifacts and focuses on defining capabilities, internal events, plans and detailed data structures.

B. Tools Support

1) Prometheus Design Tool (PDT) is a graphical editor which supports the design tasks specified within the Prometheus methodology for designing agent system. Similar to most modern software engineering methodologies, Prometheus is intended to be applied in an iterative manner, this leads to a need for consistency checking in order to ensure that the design remains consistent when changes are made. Manual consistency checking is tedious and error prone and tool support is therefore highly desirable [10]. The user interface of the PDT is shown in Fig. 2. The program is written in Java and its main features include structured textual descriptors, information propagation from one part of the design to another, consistency checking, hierarchical views and report generation. The output diagrams of the detailed design phase can be readily transformed into JACK agent code.
2) JACK Support for Prometheus: The Jack development environment (JDE) is a framework in Java for multi-agent system development, which provides a design tool that allows Prometheus-style overview diagrams to be drawn. The JDE can then generate skeleton code from these diagrams and ensures that changes made to the code are reflected in the design diagrams and vice versa, this facility has proven quite useful [11].

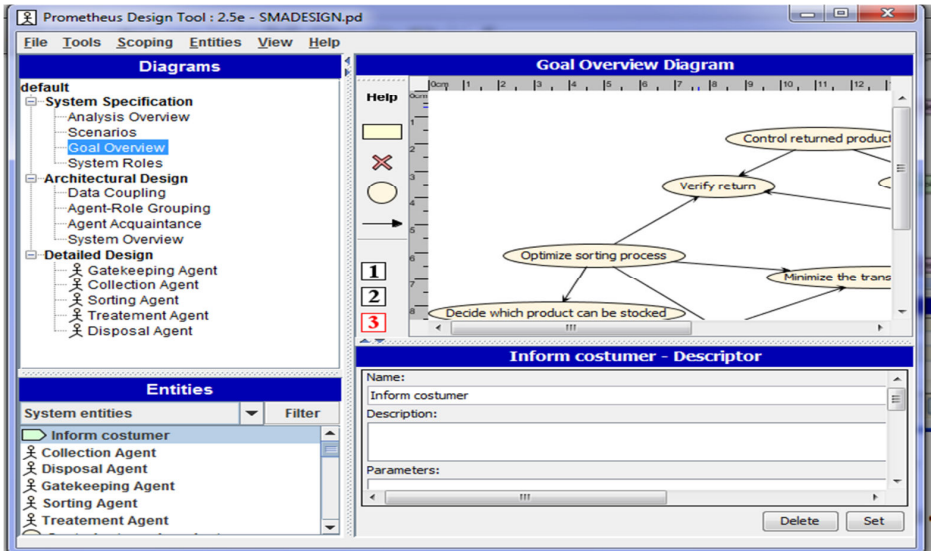

Fig. 2 Prometheus Design Tool user interface

IV. Design Based Prometheus Methodology

A. System Specification

The system specification is the initial phase of the Prometheus methodology. We start with a brief description of what our proposed system should be able to do a set of scenarios of the system are described in this section.

1) System Description

We would like to develop a distributed multi agent system which can perform the reverse logistical activities include gate keeping, collection, sorting, processing and disposal (See Fig. 3). The agent system should be able to optimize the reverse logistics processes; this optimization is performed by minimizing cost and time of each step. In addition to automatically integrate the RL system, the agents are able to communicate with the other stakeholders.

2) System Goals

The main goals of the agent system are briefly presented below and their further decomposition is depicted in Fig. 4.

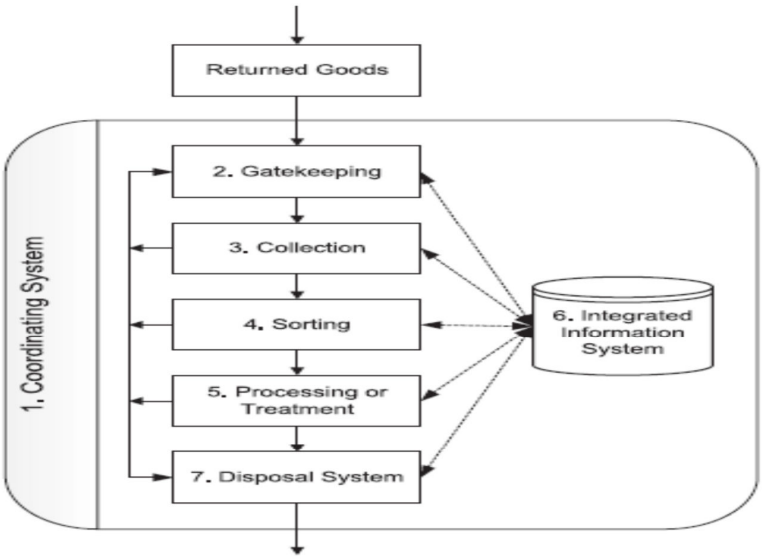

Fig. 3 The agent system's environment [4]

[G1]: Control returned product: The agent system, should be able to predict and handle the customer query, in such case the agent system's responsibility is to verify the quantity Q of returned product, then assign an authorization number and notify both the customer, distributor.

[G2]: Optimize the collection process: In order to optimize the collection process the product category, the

transport distance, the transit time and transit cost must be known. The agent system should be able to find the nearest service point and the appropriate transportation mode.

[G3]: Optimize sorting process: Here the agent system must examine the item in view of deciding how to treat it; here the system should consider the availability of refurbishing and disassembling site.

[G4]: Optimize the treatment process: The agent system should be able to choose the appropriate treatment option respecting certain criteria, in order to minimize the treatment cost and time. After that the agent system should inform the manufacturer.

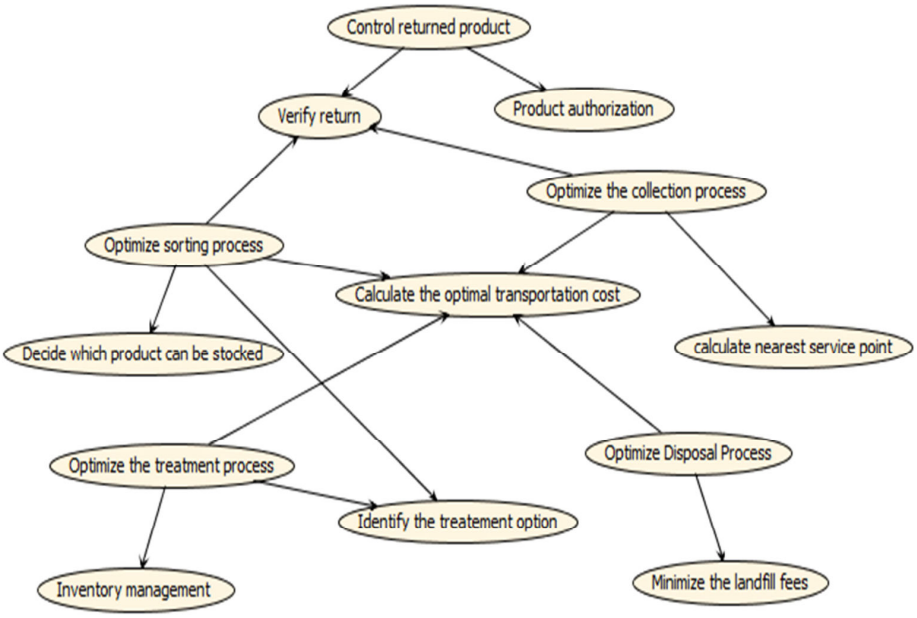

Fig. 4 Goal Overview Diagram

3) Scenarios

Here we describe certain scenarios that take place in the system together with their appurtenant steps. The scenarios illustrate the normal running of the system. "OR" denotes situations where there are two alternative sequences. Fig. 5 illustrates the scenarios and the connections between them:

[S1]: Control returned Product: Trigger: the return control required once the customer declares the need to return a product back, the agent system filters which products are allowed to enter the RL system and then it sends the authorization to the customer. Percept: product information. Goal: verify retuned product. Action: assign the number authorization, notify the customer and inform the distributor. Scenario: collection process (S2), Or Scenario: disposal process (S5).

[S2]: Collection Process: Trigger: when the collection is initiated, the nearest service point, and the transportation mode must be specified. Percept: product information, transport distance, transit time. Goal: calculate nearest service point. Goal: calculate the optimal transportation cost. Scenario: sorting process (S3), Or Scenario: disposal process (S5).

[S3]: Sorting Process: Trigger: after analyzing the returned products Quality, during the sorting process the capacity of refurbishing and disassembling site must be calculated. Goal: identify the treatment option. Goal: decide which product can be stocked. Action: inform the manufacturer. Scenario: treatment process (S4), Or Scenario: disposal process (S5).

[S4]: Treatment Process: Trigger: optimization of disposal requested. Percept: stock permitted limit. Goal: choose the treatment option. Scenario: inventory management (S6) or disposal process (S5).

[S5]: Disposal Process: Trigger: optimization of disposal requested. Percept: stock permitted limit

Goal: minimize landfill fees. Goal: calculate the optimal transportation cost.

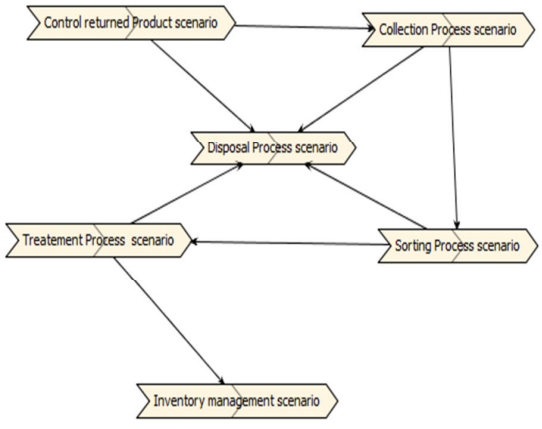

Fig. 5 System Scenarios Diagram

B. System Development

The purpose of this section is to define what agents are to be parts of the system, how they perform their tasks and how they interact with each other to meet the overall required functionality.

The artifacts produced during the system specification are used as a basis for developing the high-level design of the agent system. The Subsection *B*.1 presents the architecture of our agent system. It starts with an illustration of how our agents are distributed in the simulated environment and continues with a detailed description of the agent types and the overall interaction between them. Subsection *B*.2 provides the system overview diagram of our MAS architecture.

1) System Architecture

The agent System used in our architecture is composed of 5 agent types who act respectively during the 5 steps of reverse process: each agent has its own knowledge base that contains knowledge about the system environment (as is shown in Fig. 6). Furthermore, each of those is characterized by probabilistic learning capability to improve its own behavior.

These agents communicate with internal resources (local databases) and external (partners of the supply chain).

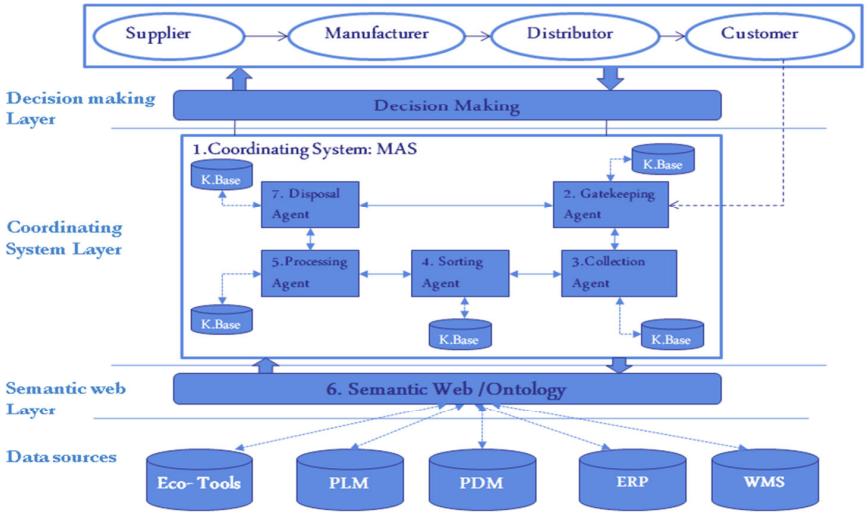

Fig. 6 Agents in our simulated environment [12], [13]

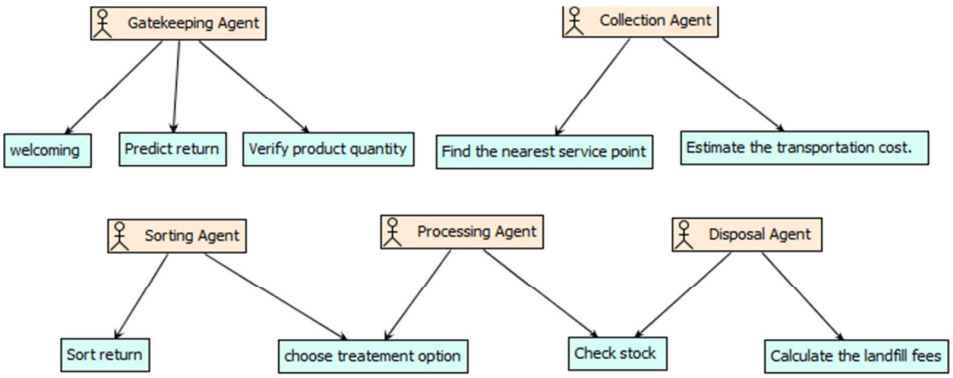

Fig. 7 Agent types and the roles they fill

The interaction between agents capture the dynamic aspects of the system, Fig. 8 shows an acquaintance diagram that illustrates how the agents are connected. The gate keeping agent sends messages to the collection agent. The collection agent communicates in the same way with the sorting agent. This agent and the processing agent communicate between themselves. All of these agents and at any stage of reverse logistics could communicate with the disposal agent.

The agent types of this architecture are defined by considering the roles and scenario of the system specification. The agents should be evaluated against criteria of coupling and cohesion, and determining the data needed by the different roles is of key importance here. Some agent type is described in more detail together with the overall interaction between them as follow and their further decomposition is depicted in Fig. 7:

[T1]: Gate keeping agent: Cardinality: one agent for gate keeping process. Lifetime: lives as long as the controlling

product process is in operation.

Role: [R1] Welcoming, [R2] Verify return, [R3] Predict return. Description: the gate keeping's main responsibility is to verify return, in order to accept or refuse the returned product. Then it should communicate the authorization number, and notify both collection agent and the customer.

Initialization: The agent needs to know the return data.

[T2]: Collection agent: Cardinality: one agent for collection process. Lifetime: lives as long as the collection processing of the returned Product is in operation. Role: [R3] find the nearest service point, [R4] estimate the Transportation cost. Description: the collector's main responsibility is to collect the returned product, In order to do this, it must estimate the nearest company service point and the transportation cost, the goal of this estimation is to Identify and optimize the collection process from the customer to the collection Site, then just after defining optimized solution the Agent notifies the sorting agent. Initialization: The agent needs to know the names of the external services it should communicate with during the collection process.

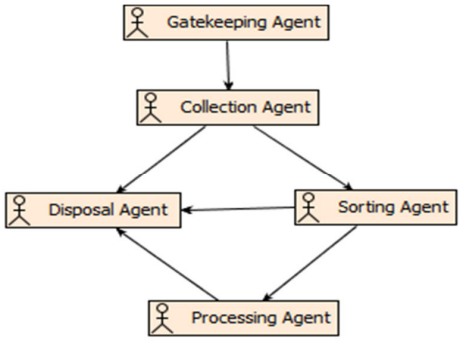

Fig. 8 The agent acquaintance diagram

2) Detailed Design

In the section above we introduced the Fig. 6 which illustrated how the agents of our system fit into our environment. We have now described the roles and tasks of these agents, and the overall interaction between them. Fig. 9 provides an overview of the architecture with the main entities involved.

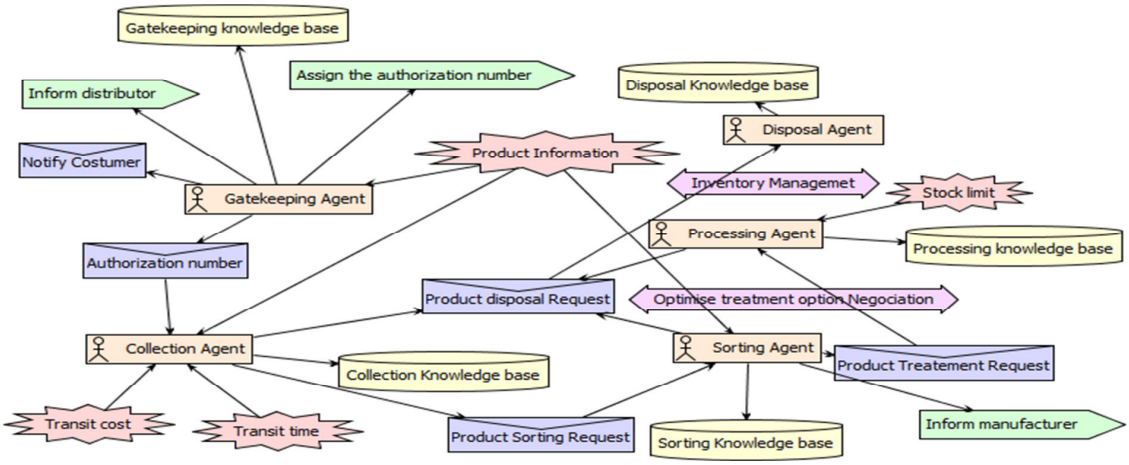

Fig. 9 System Overview

The agent overview diagram developed as an example using PDT detailed design process. PDT has the ability to validate each design entity dynamically while the development process is running.

V. Conclusion

In this paper we have briefly described the design of our proposed approach; the system is based on multi-agent system architecture, consisting of five agents: Gate keeping Agent, Collection Agent, Sorting Agent, Processing Agent and Disposal Agent where each agent execute different tasks in each step in reverse logistics network. The agent is designed using Prometheus Methodology and we've used PDT (Prometheus Design tools).

The next step is to integrate a model of Bayesian decision network to improve the agents' behavior, and to develop an example of this system.

References

[1] D. S. Rogers, R. S.Tibben-Lembke, "Going backwards: Reverse logistics trends and practices". Reverse logistics executive council, Reno, NV, USA (1998).

[2] M. Fleischman, J.M. Bloemhof-Ruward, R. Dekker, E. van der Laan J.A.E.E. van Nunen, and L.N. van Wassenhove, "Quantitative models for reverse logistics: a review," European Journal of Operational Research, vol. 103, no. 1, pp. 1–17, 1997.

[3] A.M.Talib, R.Atan, R. Abdullah, and M.A.A. Murad, "Multi Agent System Architecture Oriented Prometheus Methodology Design to Facilitate Security of Cloud Data Storage", Journal of Software Engineering, No 5, pp. 78-90. (2010).

[4] S.Lambert, D.Riopel and W.Abdulkader, "A reverse logistics decisions conceptual framework", Computer and Industrial Engineering 61(2011), pp.561-581.

[5] S. Lambert, D. Riopel, « Logistique inverse: review of literature », Les Cahiers du GERAD, Quebec (2003).

[6] M.P. Papazoglou:"Agent-Oriented Technology in Support of E-Business". Communications of the ACM. 44(4) (2001) 71-77.

[7] Yang, H.-L., and Wang, C.-S. "Integrated framework for reverse logistics," In New Trends in Applied Artificial Intelligence, Berlin, Heidelberg: Springer, pp. 501–510, 2007.

[8] L. Padgham, M. Winikoff, 2002. "Prometheus: A methodology for developing intelligent agents", In Proceeding of the 1st International Joint Conference on Autonomous Agents and Multi agent Systems: Part1, July 15-19, 2002, Bologna, Italy, pp: 37-38.
[9] Padgham, L. &Winikoff, M. : "Prometheus: A pragmatic methodology for engineering intelligent agents", In J. Debenham, B.Henderson-Sellers, N. Jennings & J. Odell (Eds.), Proceedings of the OOPSLA 2002 Workshop onAgent-OrientedMethodologies, Seattle,Washington. pp.97-108, (2002).
[10] J. Thangarajah, L. Padgham, and M. Winikoff: "Prometheus design tool". In Proceedings of the fourth International Joint Conference on Autonomous Agents and Multi agent Systems, New York, NY, USA, pages 127–128, (2005). ACM Press. 3.2, 3.2
[11] L. Padgham, M. Winikoff, 2002. "Prometheus: A Practical Agent-Oriented Methodology". In: Henderson-Sellers, B., Giorgini, P (Eds). Agent-Oriented Methodologies, pp. 107-135. IDEA Group Publishing (2005).
[12] F. Lhafiane, A. Elbyed, M. Bouchoum, "Improving Reverse Logistics Process Using Multi-Agents System and Semantic Web," Journal of Traffic and Logistics Engineering, Vol. 2, No. 3, pp. 206-210, September 2014.
[13] F. Lhafiane, A. Elbyed, M. Bouchoum, "Reverse Logistics Information Management Using Ontological Approach", World Academy of Science, Engineering and Technology, International Science Index 98, International Journal of Computer, Control, Quantum and Information Engineering, 9(2), 396 - 401.

F. Lhafiane was born in 1988. She earned her master's degree in computer science at Hassan II University, faculty of sciences Casablanca, Morocco. In 2013, she joined the computer lab and decision support at Hassan II University. Her actual main research interests concern Decision making in the context of Reverse Logistics.

A. Elbyed was born in Casablanca- MOROCCO on April 17, 1978 He is a professor in department of Mathematics and Computer science, at Hassan II University, faculty of sciences Ain Chock, Casablanca - Morocco. He obtained a PhD in the National Institute of Telecommunications (INT) of Evry, France, where he participated in European projects, such as SAPPHIRE and POPS. His research focuses on semantic web and the automatic detection of contextual similarity (Ontology Mapping) using artificial intelligence. He is the author or co- author of several articles published in journals and international conferences (IEEE).

M. Bouchoum holds a PhD in Algebraic Number Theory from Laval University in Canada. Since 1986 he has been professor in Department of Mathematics and computer at Hassan II University, faculty of sciences Ain Chock - Casablanca, during the last years he has been working on security of system using cryptography and has authored and co-authored several publications in journals and international conferences (IEEE).