Construcción de un módulo de registro para comunidades de agentes a través del concepto de robustez en redes ad hoc

Lina Johana Alfonso Silva

Universidad Nacional de Colombia
Facultad de Ingeniería
Bogotá, Colombia
Año 2018

Construcción de un módulo de registro para comunidades de agentes a través del concepto de robustez en redes ad hoc

Lina Johana Alfonso Silva

Trabajo de grado presentado como requisito parcial para optar al título de:
Magister en Ingeniería Telecomunicaciones

Director:
Ph.D. Jorge Eduardo Ortiz Triviño

Línea de Investigación:
Redes y Sistemas de Telecomunicaciones
Grupo de Investigación:
TLÖN

Universidad Nacional de Colombia
Facultad de Ingeniería
Bogotá, Colombia
Año 2018

-Soy un gran creyente en la suerte, y creo que cuanto más trabajo, más suerte tengo.-

Stephen Leacock.

Agradecimientos

En primer lugar quiero agradecer a mi familia y mi novio, quienes con su paciencia y consejos me dieron la motivación para seguir adelante y no desfallecer.

Al grupo TLÖN por darme la oportunidad de participar en el proyecto que se lleva a cabo y especialmente a mi asesor y al profesor Jorge Ortiz que me guiaron y apoyaron para alcanzar mis objetivos.

Resumen

Las redes ad hoc son sistemas descentralizados que carecen de infraestructura fija y exhiben comportamientos dinámicos y estocásticos haciendo difícil predecir su comportamiento en el tiempo. Esta condición permite estudiarlas como sistemas complejos a través del uso de propiedades como la auto-organización, la adaptación, anticipación y la robustez. Así, este trabajo tiene como propósito construir un módulo de registro para agentes de software combinando los conceptos de institución y robustez que esté en la capacidad de operar sobre una red ad hoc y soportar cambios constantes en el ambiente de operación. Para esto, se creo un módulo de software utilizando principios de diseño social inspirados y se implementaron tres escenarios de prueba donde se emulan las posibles fallas del sistema. Los resultados evidencian la robustez del módulo ya que cuenta con la capacidad de recuperar su estado de manera autónoma y permite que los servicios sigan operando de forma aceptable.

Palabras clave: Ad-hoc, Agente, Auto-organización, Comunidad, Institución, Registro, Robustez.

Abstract

Ad hoc networks are decentralized systems that lack fixed infrastructure and exhibit dynamic and stochastic behaviors making it difficult to predict their behavior over time. This condition allows studying them as complex systems through the use of properties such as self-organization, adaptation, anticipation and robustness. This work has the purpose of building a registration module for software agents combining the concepts of institution and robustness that it has the ability to operate on ad hoc network and support constant changes in the operating environment. For this, a software module was created using inspired social design principles and three test scenarios were implemented where possible system failures are emulated. The results show the robustness of the module since it has the ability to recover its status autonomously and allows services to continue operating acceptably.

Keywords: Ad-hoc, Agent, Self-organization, Community, Institution, Registration, Robustness

Contenido

Agradecimientos vii

Resumen ix

1 Introducción 2

2 Sistemas complejos e introducción al proyecto TLÖN 4
2.1 Red Ad-hoc . . . 4
2.2 Auto-organización . . . 6
2.3 Agentes Artificiales . . . 7
2.3.1 Agente como una Máquina de Estados Finitos . . . 9
2.4 Robustez . . . 9
2.5 Proyecto TLÖN . . . 12

3 Estructura del modelo social en la agrupación de agentes 14
3.1 Sistema Multiagente . . . 14
3.2 Relevancia de la Robustez . . . 15
3.3 Recomendaciones FIPA . . . 16
3.4 Modelo Social-inspirado . . . 17
3.5 Institución . . . 18
3.5.1 Definición . . . 18
3.5.2 Constitución de la Institución . . . 20
3.6 Funciones de la Registraduría . . . 20

4 Legislación de la institución y diseño del sistema robusto 22
4.1 Normatividad . . . 22
4.1.1 Normas Generales de la Institución de Registro . . . 22
4.1.2 Normas para el Registro de Agentes . . . 23
4.1.3 Normas Generales del Directorio de Servicios . . . 23
4.1.4 Normas del Agente Supervisor del Módulo . . . 24
4.2 Organigrama de la Registraduría . . . 24
4.3 Definición del Modelo de Software . . . 25
4.3.1 Almacenamiento de Datos . . . 25
4.3.2 Procedimiento para el Registro de Agentes . . . 26

4.3.3 Diagrama de Estado del Agente 28
4.3.4 Diagrama de Secuencias 30
4.3.5 Diagrama de Clases 31

5 Validación de Escenarios de Pruebas 33
5.1 Escenario 1 34
5.2 Escenario 2 37
5.3 Escenario 3 40

6 Conclusiones y Recomendaciones 44
6.1 Conclusiones 44
6.2 Recomendaciones 45

Bibliografía 46

1 Introducción

Las redes Ad-hoc son redes que carecen de una infraestructura fija y se crea por la interacción de dos o más nodos que se observan en efectos globales visibles como el transporte de un mensaje de un nodo origen a un nodo destino [11]. Esta interacción permite denominar a este tipo de redes como sistemas complejos ya que una de sus definiciones es que un sistema es complejo si este consiste en la interacción de varios elementos, dado eso, el comportamiento del sistema será difícil de deducir a partir del comportamiento de sus partes [16]. Sin embargo, el pensamiento complejo permite contemplar diferentes representaciones al mismo tiempo, con el fin de tener un entendimiento menos incompleto del sistema y de esta forma lograr trabajar con esta clase de sistemas, ¿cómo?, adaptando las acciones a la nueva situación, y de ser necesario reconfigurando el sistema sin tener que destruirlo [16].

Así mismo las redes Ad-hoc se relacionan con los mecanismos de auto-organización, que de acuerdo con [11] permite construir sistemas escalables que consisten en un gran número de subsistemas. El objetivo principal es la coordinación y colaboración sobre una meta global, dado esto surge la necesidad de cómo resolver los problemas más prominentes de la red como lo son la confiabilidad y disponibilidad.

Una forma para lograr lo anterior es hacer que un sistema sea *robusto* y con base en ello se va a centrar este trabajo, implementando un servicio, que gracias a su autonomía y manera de actuar frente a cualquier situación inesperada, permite que de manera automática se pueda aplicar esta característica de robustez a cualquier sistema multi-agente.

El trabajo se va a desarrollar al interior del grupo de investigación TLÖN y será un aporte al macro proyecto que se está llevando acabo al incluirlo en la capa de sistemas multi-agentes, más específicamente se elaborará un módulo de registro el cual será el responsable de ser el directorio de todos los agentes que se encuentren en el ambiente y al ser un servicio tan importante para la comunicación de todo el sistema, se debe garantizar que este sea robusto, es decir, frente a alguna falla el módulo debe ser capaz de recuperarse y mantener la información actualizada.

Teniendo en cuenta la revisión del estado del arte, no se encuentran muchos trabajos en donde un agente artificial sea quien, con la información que captura del ambiente, realice un proceso robusto a un servicio como lo es el registro y actualización del estado de una

comunidad de agentes el cual es fundamental para la interacción entre ellos. A partir de allí surgió la siguiente necesidad: ¿Cómo mantener actualizado el estado de comunidades de agentes en su ambiente bajo las condiciones dinámicas de una red ad hoc de forma que sea un sistema robusto?.

Para contestar esta pregunta se planteó el modulo de registro a través del concepto de institución el cual posee sus propias leyes y procedimientos para lograr alcanzar con éxito sus objetivos. Estas reglas se observan con más detalle en el desarrollo de este documento y son relevantes para el funcionamiento del programa. De igual forma se debe determinar la organización de los agentes a cargo del módulo de registro para que cumplan con la función de robustez e implementarlos sobre el sistema multiagente que se trabaja al interior del grupo.

Finalmente se va a validar el funcionamiento del módulo, creando tres escenarios de prueba en donde se generen fallas del sistema y corroborando la disponibilidad de la información en todo momento, con estas pruebas se espera obtener datos relevantes para demostrar cómo un servicio puede ser robusto aún sobre sistemas complejos social inspirados como lo son las redes Ad-hoc y el sistema TLÖN.

El documento se encuentra estructurado de la siguiente manera: En la primera parte se contextualizan los temas referentes a redes ad-hoc, sistemas auto-organizantes, robustez, ejemplos de donde se han implementado estos conceptos y la presentación del proyecto TLÖN. El tercer capítulo hace referencia a la conformación de los modelos sociales que se van a implementar en este trabajo.

En el cuarto capítulo se definen las reglas que se implementaron para el correcto funcionamiento de la institución y el diseño del módulo de software que cumpla con los objetivos planteados. En el quinto capítulo se muestran los resultados obtenidos en cada escenario de prueba y por último se hacen las conclusiones y recomendaciones acordes a estos resultados y a la investigación realizada a lo largo del trabajo.

2 Sistemas complejos e introducción al proyecto TLÖN

Parte de la motivación de este trabajo se debe al entendimiento tanto de las necesidades como de los requerimientos que se deben cumplir para lograr que un servicio sea robusto y por qué es tan importante esta cualidad en los sistemas que son dinámicos, de igual forma es importante conocer como se puede descentralizar la gestión de todos los componentes para no depender de una única instancia.

Es por ello que en este capítulo se van definir los conceptos más relevantes que se tuvieron en cuenta para el desarrollo del trabajo, los que sirvieron para justificarlo y cómo cada uno se relaciona con el otro para construir una solución al problema planteado.

Se presentará desde el concepto más elemental como lo es el de la red ad-hoc, al igual que el de auto-organización como un medio para tratar este tipo de sistemas, qué representa un agente y qué tipos de agentes existen, a qué hace referencia que un sistema sea robusto y se llegará al más más compuesto que es el Proyecto TLÖN ya que es quien va integrar todo el diseño y tecnología aquí implementadas.

2.1. Red Ad-hoc

Una red ad-hoc es una colección de nodos móviles inalámbricos que forman una red dinámica temporal sin el uso de una infraestructura existente o una administración centralizada. Los nodos son libres de moverse aleatoriamente y organizarse de forma arbitraria, así, la topología de la red inalámbrica puede cambiar rápida y de manera impredecible [6].

De igual forma los servicios deben trabajarse de manera distribuida en donde todos los nodos se coordinan para habilitar la comunicación entre ellos mismos [8]. Esto requiere que cada uno sea más inteligente para poder funcionar como host y router volviéndolas más complejas que las redes tradicionales.

Las redes ad hoc brindan características como fácil conexión a las redes de acceso, estructuras de red multi-hop dinámicos, y comunicación directa peer-to-peer. Así mismo los usuarios

se pueden beneficiar de estas redes al permitirles cambiar entre dispositivos, migrar sesiones, y aún así obtener los mismos servicios personalizados. En cuanto la movilidad de host permite a los dispositivos de los usuarios moverse por las redes y mantener la conectividad y accesibilidad. A continuación se muestra un esquema general de una red ad-hoc en la Figura **2-1**. [22]

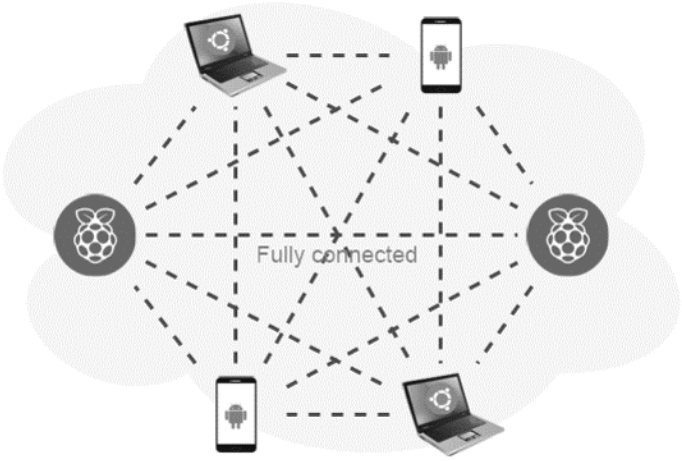

Figura 2-1: Red Ad-Hoc.

Adicional se deben tener en cuenta las siguientes características al momento de diseñar soluciones con este tipo de redes [23]:

- La redes manejan una operación distribuida porque cada nodo opera independiente y no hay un servidor centralizado para administrar la red.
- Cada nodo trabaja en cooperación con otro nodo para implementar funciones como de seguridad y enrutamiento.
- La capacidad de ancho de banda es menor en comparación con las redes cableadas. Se pueden experimentar problemas de tasa de error de bit porque la ruta de enlace end-to-end es usada por múltiples nodos en la red.
- Otro rasgo a tener en cuenta es que todos los dispositivos son móviles y obtienen su energía de baterías, el cual es un recurso limitado [13], por lo que debe ser usado de manera eficiente.
- La topología de la red está en constante cambio porque los nodos en la red ad hoc cambia su posición de forma aleatoria ya que son libres de moverse a cualquier parte. Por lo tanto los equipos deben ser capaz de soportar topologías dinámicas.

Atendiendo a estas características, se van a presentar formas de estudiar estos sistemas que podrían considerarse complejos dado su dinamismo.

2.2. Auto-organización

Äuto-organización es la forma de observar los sistemas, no una clase absoluta de sistema".[16]

Para hablar de auto-organización, primero se debe definir qué es un sistema complejo. En [7] describe un sistema complejo como las relaciones entre las partes que dan lugar a los comportamientos colectivos de un sistema, y cómo el sistema interactúa y establece relaciones con su entorno.

Cabe resaltar que el resultado de esas relaciones no siguen un patrón definido ni son fáciles de predecir en el futuro. Es por ello que deben surgir nuevos métodos de cómo tratar estos sistemas complejos para poder realizar un mejor análisis del comportamiento de estos sistemas.

Teniendo en cuenta lo anterior surge el concepto de auto-organización. Auto-organización es un proceso en el cual el patrón a nivel global de un sistema surge exclusivamente de numerosas interacciones entre los componentes de nivel inferior de ese mismo sistema. Además, las reglas que especifican las interacciones entre los componentes de los sistemas se ejecutan utilizando solamente información local, sin hacer referencia al patrón global y así la interacción de los componentes singulares finalmente definen el comportamiento del sistema global.

Aplicándolo a las redes ad hoc, la auto-organización puede ser vista como las interacciones entre los nodos de la red que conducen a efectos visibles globalmente, por ejemplo, el transporte de mensajes de un nodo de origen a un nodo receptor [11].

Un sistema de auto-organización exitoso utiliza combinaciones de los siguientes enfoques para mantener su integridad en un ambiente cambiante e inesperado [16]:

- **Adaptación:** El sistema cambia para hacer frente con el cambio. Le permite al sistema modificarse así mismo para .ªjustarse"mejor dentro del ambiente.
- **Anticipación:** El sistema predice un cambio al cual le debe hacer frente, y se ajusta en consecuencia. Esto prepara al sistema para los cambios antes de que ocurran, adaptando el sistema sin que este sea perturbado. El sistema no requiere experimentar una situación antes de responder a ella.
- **Robustez:** Un sistema es robusto si este continúa funcionando frente a una perturbación. Le permite al sistema resistir a perturbaciones sin perder su función o propósito, y dado eso le da la posibilidad de adaptarse.

De acuerdo con lo anterior, los dos primeros enfoques son activos, en el sentido en que ellos producen cambios en el sistema, mientras que el último es más bien pasivo.

Desde un punto de vista computacional en [37] definen auto-organización como el mecanismo o proceso habilitado en un sistema para cambiar su organización sin un comando externo explícito durante su tiempo de ejecución.

Así mismo ya se observan nuevos campos en donde se empiezan a implementar este concepto como lo es la computación en la nube. Tal como se observa en el estudio [9] se argumenta que la computación en la nube representa un sistema complejo y por tanto la auto-organización es la técnica apropiada para manejarlo.

2.3. Agentes Artificiales

Según Wooldridge, un agente es un sistema computacional que es situado en un ambiente, y que es capaz de una acción autónoma en ese ambiente con el fin de cumplir sus objetivos de diseño [43]. A su vez el agente es un software diseñado con características que le permitan realizar las acciones para las cuales fue programado.

El efecto que tiene la interacción entre el ambiente y los agentes usualmente se pueden ver reflejados en los siguientes puntos [18]: La forma de existencia y comportamiento de los agentes está definida por el entorno, los estados del entorno influirán directamente en la capacidad de los comportamientos del agente y los agentes pueden modificar los estados del entorno llevando a cabo sus comportamientos.

Teniendo en cuenta lo anterior y analizando aspectos especiales que debe tener un agente para que sea considerado inteligente, en [29] se define los agentes inteligentes como un pieza de software que son:

- Situado, pues existe en un entorno específico;
- Autónomo, ya que debe ser independiente y no controlado externamente;
- Reactivo, debe responder de manera oportuna a los cambios de su ambiente;
- Proactivo, pues persigue sus objetivos de manera permanente;
- Flexible, tiene múltiples maneras de alcanzar sus objetivos;
- Robusto, se puede recuperar de los fallos;
- Social, ya que interactúa con otros agentes.

Adicional a esto, se espera que un agente sea racional, es decir, un agente no puede reservar un mismo recurso para dos acciones diferentes que posteriormente entrarán en conflicto o disputa por ese recurso. Para esto se han realizado estudios de cómo debe ser el diseño de un agente artificial [19] [20] [21] , los cuales proporcionan explicaciones ontológicas, mecánicas y operacionales.

Estos diseños pretenden ayudar a construir mejores agentes que apoyen la creación de agentes artificiales más usables y más accesibles para los desarrolladores. Así mismo, la ejecución de un agente sigue un ciclo de sentido-pensar-actuar para la toma de decisiones racionales [29] tal como se observa en la Figura **2-2**.

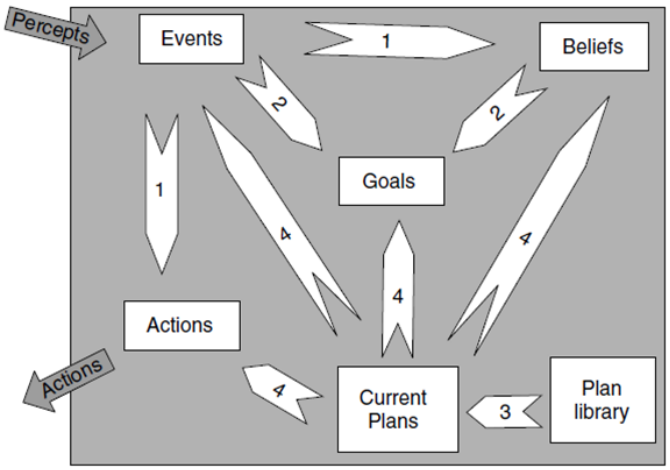

Figura 2-2: Ciclo de ejecución de un agente [29].

Donde cada paso corresponde a:

1. Los eventos se procesan para actualizar creencias y generar acciones inmediatas.

2. Los objetivos se actualizan: Nuevos objetivos son generados, alcanzados o aquellos que son imposibles abandonados, así mismo se determinan las prioridades de cada uno.

3. Los planes son seleccionados de la biblioteca de planes para alcanzar los objetivos o manejar los eventos.

4. Un paso del plan es ejecutado en el próximo plan, produciendo nuevos acontecimientos, sub-objetivos, cambios de creencias o acciones.

Por último y dado que anteriormente se describió que un agente debe ser ejecutado sobre un entorno, en [32] se proponen las siguientes clasificaciones para las propiedades de los ambientes:

- Accesible vs Inaccesible: Un ambiente accesible es cuando el agente puede obtener información completa, precisa y actualizada sobre el estado del medio ambiente. Internet no es accesible en este sentido.
- Determinístico vs No-determinístico: Determinístico es aquel en que cualquier acción tiene un solo efecto garantizado, no hay incertidumbre.
- Estático vs Dinámico: Un entorno estático es uno que puede ser asumida de permanecer sin cambios, excepto por el desempeño de las acciones por el agente. Dinámico es aquel que tiene otro procesos operando en él.
- Discreto vs Continuo: Un entorno es discreta si hay un número finito fijo de acciones y percepciones en el mismo.

2.3.1. Agente como una Máquina de Estados Finitos

Dado el concepto anterior, un agente siempre debe estar alerta a las acciones que ocurren en su entorno para actuar como según corresponda. En este orden de ideas se puede considerar un agente como una máquina de estados finitos descritos por [17] como una quintupla $M = (S, I, O, \nu, \omega)$, donde S es el conjunto de estados internos para M; I es la entrada del alfabeto para M; O es la salida del alfabeto para M; $\nu : SxI \rightarrow S$ es la función del estado siguiente; y $\omega : SxI \rightarrow O$ es la función de salida.

Las características principales de la máquina son:

- La máquina sólo puede estar en uno de los muchos estados finitos en un tiempo determinado. Estos estados son conocidos como los *estados internos* de la máquina.
- La máquina aceptará como entrada sólo un número finito de símbolos, los que en conjunto se conocen como el alfabeto de entrada I.
- La salida y el estado siguiente son determinados por la combinación de cada entrada y los estados internos. El conjunto finito de todas las posibles salidas constituye el alfabeto salida O de la máquina.

2.4. Robustez

Un concepto similar al que se ha venido trabajando hasta el momento lo encontramos en el análisis de robustez para sistemas embebidos que se hace en [5], donde definen que Robustez es la habilidad de tolerar perturbaciones que pueden afectar el cuerpo funcional del sistema y como tal un sistema robusto va a ser capaz de resistir una serie de perturbaciones al proporcionar una perdida insignificante en el rendimiento. Si la aplicación es lo suficientemente

robusta, se podrá garantizar la calidad del servicio durante un tiempo a pesar de la presencia de efectos de envejecimiento.

Así mismo allí hacen la claridad de que si la aplicación no es robusta, las perturbaciones que afectan los parámetros, por ejemplo, de un modelo no lineal, aunque pequeño, pueden introducir un cambio significativo en el comportamiento de la función, así como una caída drástica en el rendimiento.

Otro ejemplo son los estudios en los que se han entrenado un modelo de red neuronal con una máquina de alta precisión y luego se han portado a un hardware integrado menos preciso. Allí han experimentado la dramática pérdida de precisión asociada con esta operación a menos que la red se haya entrenado para ser robusta.

Acorde con la definición dada al interior de auto-organización, en [16] se proponen algunas técnicas para lograr que un sistema sea robusto, mediante las siguientes acciones:

- Modularidad: Los módulos pueden ser útiles para prevenir la propagación del daño en el sistema.
- Degeneración: Es la habilidad de que un elemento que es estructuralmente diferente pueda cumplir la misma función de otro.
- Robustez distribuida: Ocurre cuando la función del sistema es ejecutada por diferentes componentes y de esta forma pueden compensar los efectos de la perturbación.
- Redundancia: Se da cuando existen varias copias de un elemento, así si uno falla el otro puede asumir su rol.

Independiente de las acciones que se tome, la robustez es una de las características que debería tener un agente en su estado inicial o vegetativo, ya que estos últimos evolucionan acorde a su comportamiento y esta propiedad le permite continuar con su desarrollo a medida que va aumentando su complejidad y cognición, lo anterior se nota con más claridad en la Figura **2-3**.

A continuación se explica cada comportamiento de forma general:

1. Vegetativo: Están en el organismo por defecto.
2. Reflejo: Acción-respuesta basada en comportamientos
3. Reactivo: Depende de estímulos externos.
4. Motivado: Necesitan motivación interna y un estímulo externo.

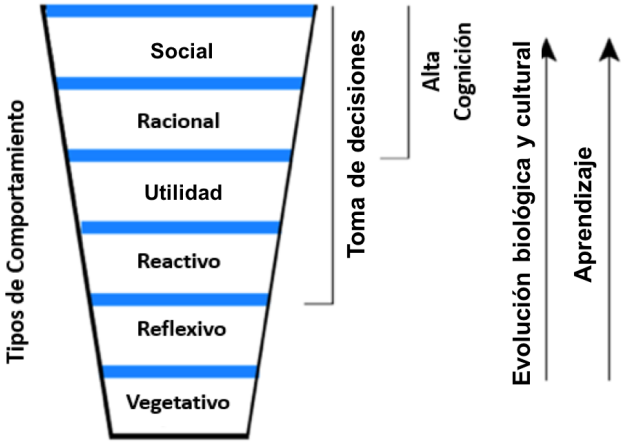

Figura 2-3: Organización para diferentes tipos de comportamientos [15].

5. Razonado: Requiere conceptos y la manipulación de estos para seleccionar el comportamiento.

6. Consciente: Comportamientos ejecutados mientras son consientes de ellos.

Revisando la literatura se observan experimentos en los cuales ya se han intentado generar robustez sobre una comunidad de agentes los cuales se observan con detalle en [36] y [26], en este último el objetivo fue comparar nueve agentes usando nueve paradigmas diferentes y aplicándolas en condiciones que diferían de aquellas para las que fue diseñado el agente originalmente. Así mismo buscaban responderse las siguientes preguntas: ¿Algunos paradigmas fomentan robustez?; ¿Hay un equilibrio entre la maximización del desempeño sobre un conjunto limitado de condiciones frente a un rendimiento robusto a través de una gama más amplia de condiciones?.

Para ello daban una noción inicial acerca de robustez y de cómo la adaptación es una de las formas para lograrlo, ya que es la propiedad en la que un agente puede cambiar su comportamiento, dependiendo del ambiente, mientras todavía mantiene los mismos objetivos y capacidades. Esta adaptación requiere de información mínima del ambiente la cual es obtenida a través de los sensores del agente y así puede determinar el paso a seguir frente a alguna anomalía.

Basados en el resultado de este experimento los autores en [26] aconsejan a los desarrolladores a ser conscientes de que obtener una alta implementación de robustez requiere ciertos compromisos que pueden afectar el rendimiento del agente en el caso óptimo.

2.5. Proyecto TLÖN

Al interior del Grupo de Investigación en Redes de Telecomunicaciones Dinámicas y Lenguajes de Programación Distribuidos - TLÖN, se está desarrollando un proyecto que propone un esquema de computación inspirado en modelos Sociales, inviables en la practica pero muy posibles en entornos artificiales controlados. Este sistema está basado en los conceptos de Justicia de Jhon Rawls, Inmanencia de Baruch de Spinoza, Paradigma Thomas Kuhn, Estado Thomas Hobbes y las concepciones de existencia y esencia de Jean Paul Sartre [3].

El objetivo de este sistema es contar con los modelos formales y computacionales que permitan a los miembros de una red compartir recursos y servicios de forma dinámica. En la figura **2-4** se pueden observar los componentes del modelo propuesto el cual además funcionará sobre una red Ad Hoc con todas sus condiciones dinámicas, estocásticas e inalámbricas.

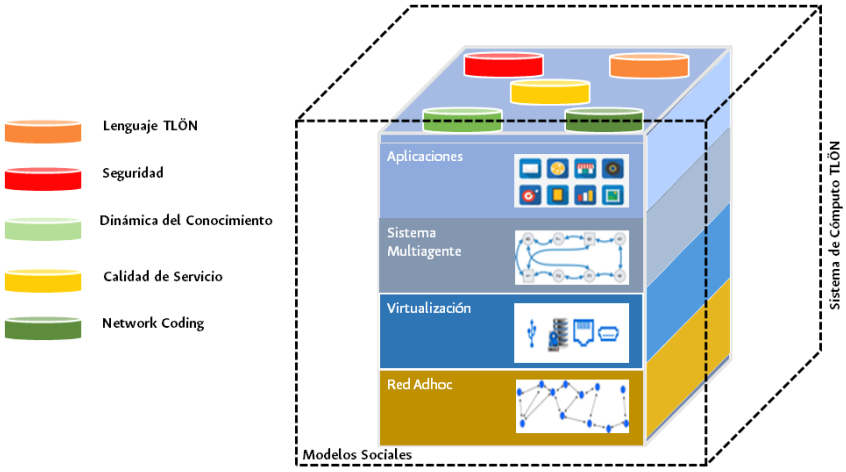

Figura 2-4: Componentes del Proyecto TLÖN [3].

Cada capa cumple con un objetivo fundamental que en conjunto componen el sistema TLÖN:

- Red Adhoc: Es una red que carece de infraestructura fija, los nodos entran y salen sin restricciones y tienen completa autonomía para configurar su participación en la red. Para el proyecto TLÖN es la parte física del sistema.
- Virtualización: Permite que con la suma de recursos individuales se pueda crear un súper recurso para toda la red de manera distribuida. Adicional proporciona el sistema operativo sobre el cual se va a ejecutar el sistema.
- Sistema Multiagente: Se encarga de la gestión y configuración de los servicios ofrecidos por el sistema. Se basan en inteligencia artificial y comunidades de agentes inteligentes auto-organizantes.

- Aplicaciones: Brindan solución a una necesidad mediante sistemas social-inspirados los cuales tienen utilidad para resolver problemas complejos.

El sistema está enmarcado con Modelos y Principios Sociales que se utiliza en la computación social-inspirada como forma de afrontar la dinámica de operación de las redes ad hoc. Para el caso del proyecto TLÖN el concepto de sociedad se ve reflejado en la dinámica del sistema multi-agente.

De igual forma y de manera transversal a todas las capas se cuentan con los siguiente componentes:

- Lenguaje: Lenguaje de programación de dominio específico que permite manipular el sistema en todas sus capas.
- Seguridad: Ya que en la red ad hoc se registran una gran variedad de nodos, se debe proteger el sistema contra ataques que puedan afectar el correcto funcionamiento del sistema.
- Dinámica del conocimiento: Hace referencia a los mecanismos de representación de conocimiento y revisión de creencias al interior del sistema TLÖN.
- Network Coding: Brinda seguridad y transferencia de datos de manera más eficiente al dividir y codificar las tramas de comunicación de manera óptima.

Acorde a la finalidad que se presentan del proyecto TLÖN, se va a iniciar el estudio de los modelos sociales en los cuales se inspira y aplican para este trabajo de forma que se vayan desarrollando paso a paso cada uno de los objetivos propuestos.

3 Estructura del modelo social en la agrupación de agentes

Como se vio anteriormente, un agente por sí solo no puede representar un sistema social como el que se está diseñando en el proyecto TLÖN, sin embargo la interacción de varios de ellos si pueden conformar un Estado y estos a su vez comunidades para el desarrollo de sus objetivos, de allí la importancia que se conozcan entre ellos y la forma de lograrlo es mediante el registro el cual habilita la posibilidad de generar acciones sociales como cooperación y coordinación.

En este capítulo se explora cómo se pueden conformar esos grupos de agentes y la forma de organización desde el punto de vista técnico con las recomendaciones FIPA hasta el punto de vista social con la conformación de una Institución.

De igual forma se resalta el por qué es importante hacer del registro un sistema robusto siendo este uno de los componentes fundamentales con el que debe contar todo sistema multiagente.

3.1. Sistema Multiagente

Entendiendo cómo funciona un agente y sus propiedades se puede llegar a definir un sistema multiagente. Un sistema multiagente está conformado por la interacción de múltiples agentes inteligentes o entidades computacionales que hasta cierto punto son autónomas y capaces de cooperar, competir, comunicarse [12], actuar con flexibilidad y ejercer control sobre su comportamiento dentro del marco de sus objetivos, así lo ratifica Gerhard Weiss en su libro [42].

También adiciona que estos sistemas son la tecnología habilitadora para una amplia gama de aplicaciones avanzadas que dependen del procesamiento distribuido y paralelo de datos, así como la información y conocimiento relevante en una gran variedad de dominios del conocimiento.

Al interior de un sistema multiagente, los agentes quienes posean intereses y objetivos comunes pueden conformar una gran variedad de comunidades de agentes y a su vez un grupo de

comunidades puede conformar una comunidad mayor, lo que permite tener una estructura jerárquica de la comunidad [40] [41]. Esto se observa con mayor detalle en la Figura **3-1**.

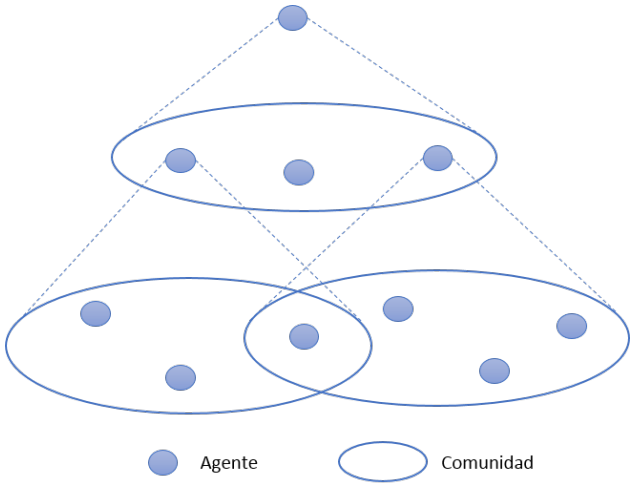

Figura 3-1: Estructura jerárquica de la comunidad [40].

3.2. Relevancia de la Robustez

Varias investigaciones evidencian que esta la robustez se vuelve aún más relevante en la implementación de agentes artificiales, así lo menciona Russel en su artículo [31] donde señala que a medida que los sistemas autónomos se vuelven más prevalentes en la sociedad, se vuelve cada vez más importante que se comporten de manera robusta.

El desarrollo de vehículos autónomos, sistemas de comercio autónomos, armas autónomas, etc., ha despertado el interés en sistemas de alta seguridad en los que se pueden garantizar sólidas garantías de robustez. Así mismo menciona las diferentes áreas de investigación de robustez en las cuales un sistema de inteligencia artificial puede fallar [31]:

- Verificación: cómo probar que un sistema satisface ciertas propiedades formales deseadas.
- Validez: cómo garantizar que un sistema que cumpla con sus requisitos formales no tenga comportamientos y consecuencias no deseados.
- Seguridad: cómo prevenir la manipulación intencional por parte de personas no autorizadas.
- Control: cómo habilitar un control humano significativo sobre un sistema de inteligencia artificial después de que comience a funcionar.

3.3. Recomendaciones FIPA

El desarrollo de software para comunidades de agentes ya ha sido estudiado y se han generado especificaciones para el correcto despliegue de estas. Uno de los modelos al que nos acogimos es el presentado por FIPA ya que presenta un estándar acorde a las necesidades del proyecto.

Foundation for Intelligent Physical Agents (FIPA) es una organización internacional que está dedicada a promover la industria de agentes inteligentes mediante el desarrollo abierto de especificaciones que respalden la interoperabilidad entre agentes y aplicaciones basadas en agentes [14].

En el modelo de referencia que propone FIPA está la gestión de agentes que proporciona el marco normativo dentro del cual los agentes de FIPA existen y operan. Adicional establece el modelo de referencia lógica para la creación, registro, ubicación, comunicación, migración y retiro de agentes.

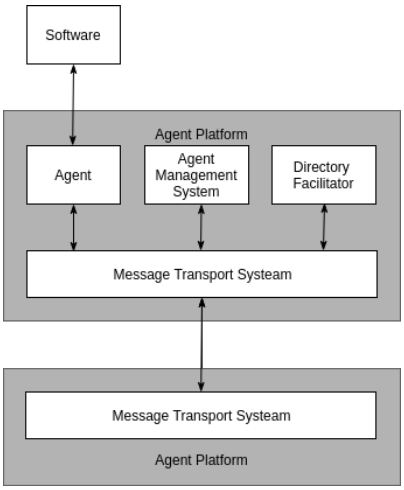

Figura 3-2: Modelo de referencia de gestión de agentes [14].

Tal como se observa en la figura **3-2** el modelo consiste en los siguientes componentes lógicos, cada uno representando un conjunto de capacidades:

- **Agent**: Es el proceso computacional que implementa la funcionalidad de comunicación autónoma de una aplicación. Un agente es el actor fundamental en un AP que combina una o más capacidades de servicio, según se publica en una descripción del servicio, en un modelo de ejecución unificado e integrado. Debe poseer un identificador único que etiqueta un agente (AID) para que se pueda distinguir inequívocamente dentro del Universo de los Agentes y *puede estar registrado en varias direcciones de transporte en las que se puede contactar.*

- **Directory Facilitato (DF)**: Es un componente opcional del AP. Provee las páginas amarillas de los servicios a otros agentes, de igual forma pueden existir múltiples DF dentro de un mismo AP.

- **Agent Management System (AMS)**: Es un componte obligatorio del AP. El sistema de gestión de agentes ejerce un control de supervisor sobre el acceso y el uso del AP. Solo un AMS existirá en un solo AP. El AMS mantiene un directorio de AID que contienen direcciones de transporte (entre otras cosas) para agentes registrados con el AP. El AMS ofrece servicios de páginas blancas a otros agentes. Cada agente debe registrarse en un AMS para obtener un AID válido.

- **Message Transport Service (MTS)**: Es el método de comunicación por defecto entre agentes de diferentes APs.

- **Agent Platform (AP)**: Provee la infraestructura física en la cual los agentes pueden ser desplegados. El AP consiste en la(s) máquina(s), el sistema operativo, el software de soporte del agente, los componentes de administración del agente FIPA (DF, AMS y MTS) y los agentes.

Para fines prácticos, algunos de los componentes anteriormente mencionados se conocerán a lo largo del desarrollo del trabajo de la siguiente forma, teniendo en cuenta que son social-inspirados:

- AMS $\rightarrow$ Institución de Registro: Teniendo en cuenta que es única y quien es la que administra todo lo referente al registro de los agentes.

- AP $\rightarrow$ Ambiente: Es el país o lugar de origen donde nacen y conviven los agentes. Adicional de soportar todas las instituciones que se constituyan.

3.4. Modelo Social-inspirado

Dado que el proyecto está enmarcado por el sistema TLÖN, se pretende construir un modulo de registro para comunidades de agentes a través del concepto de institución. Como se argumenta en [35] la conciencia de los agentes se convierte en un ciudadano de primera clase en las aplicaciones multiagentes contemporáneas. Dado esto, el módulo será análogo a la Registraduría Nacional y de ahí su relevancia pues todos los agentes deberán registrarse para que puedan ser reconocidos dentro del ambiente y por tanto puedan pertenecer a una comunidad y socializar con otros agentes.

El modelo de institución se usó ya que se ha demostrado que es un mecanismo poderoso para hacer las interacciones de los agentes más efectivas, estructuradas, coordinadas y eficientes [10] creando un orden social.

Sin embargo se debe considerar un factor muy importante que son las *normas*, estas permiten controlar el ambiente haciéndolo más estable y predecible. Las normas son un conjunto de convenciones a las que los agentes están dispuestos a ajustarse, pueden ser estrictamente obligatorias o un poco más permisivas permitiendo violarlas al prescribir sanciones apropiadas [24].

Es por ello que para lograr lo anterior primero se van a establecer las reglas y los procedimientos asociados a ellas para el registro de la comunidad de agentes. Con ello se pretende establecer una organización para realizar este proceso y definir unas bases para constituir una institución.

Posterior a esto se determinará la organización de los agentes a cargo del módulo de registro, quienes serán los funcionarios de la institución y uno de sus principales objetivos será garantizar la robustez del sistema.

Vale la pena tener presente que existen otras clases de instituciones dinámicas [27] que emergen con un objetivo claro por la necesidad del ambiente y por la asociación de varios agentes con finalidades similares pero que una vez se ha culminado ese objetivo se disuelven y la institución desaparece ya que no genera ninguna ganancia. Para este caso no aplica ya que la institución de registro será permanente a lo largo del tiempo y bajo los términos que se ven en la siguiente sección.

3.5. Institución

Acá se va a puntualizar el concepto de institución que se tuvo en cuenta para desarrollo de este trabajo y todos los factores correspondientes para su constitución.

3.5.1. Definición

Se puede definir Institución como los conjuntos de reglas dentro de un contexto social que se utilizan para determinar [28]:

- Quién es elegible para tomar decisiones en algún ámbito.
- Qué acciones están permitidas o restringidas.

- Qué acciones están permitidas o limitadas.
- Qué reglas de agregación usarán.
- Qué procedimientos se deben seguir.
- Qué información debe o no se debe proporcionar.
- Qué pagos se asignarán a las personas que dependen de sus acciones.

Existen tres niveles de reglas que afectan acumulativamente las acciones tomadas y los resultados obtenidos tal como se muestran en la Figura **3-3**.

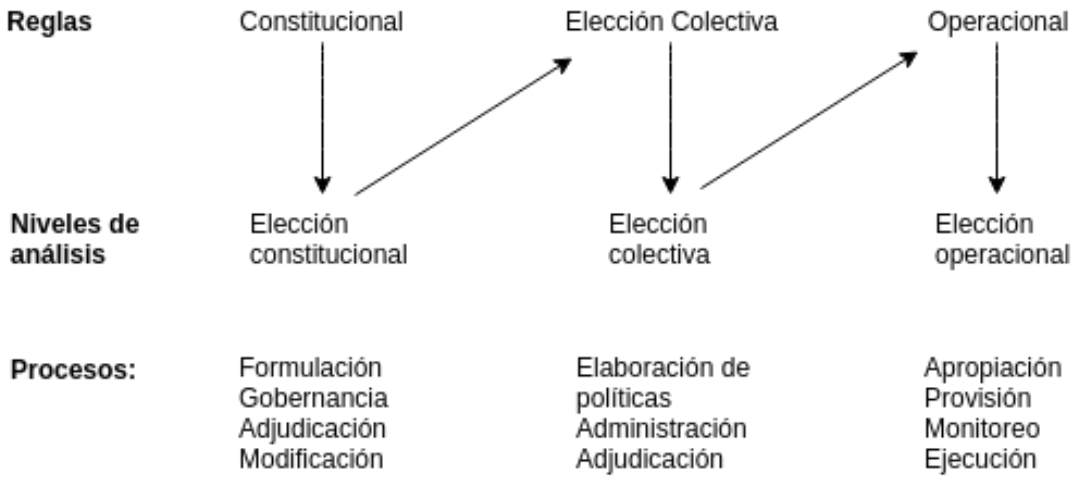

Figura 3-3: Vínculos entre reglas y niveles de análisis [28].

Donde las *Reglas Operacionales* afectan las decisiones cotidianas de los usuarios sobre cuándo, dónde y cómo retirar los recursos, quién debe monitorear las acciones de los otros y cómo, qué información debe intercambiarse o retenerse y qué recompensas o sanciones se otorgarán a la combinación de acciones y resultados.

Las *Reglas de Elección Colectivas* afectan indirectamente las elecciones operacionales. Son las reglas que usan sus funcionarios o las autoridades externas para hacer política (las reglas operacionales) sobre cómo deberían ser administrados los recursos.

Por último las *Reglas de Elección Constitucional* afecta las actividades operacionales y los resultados a través de sus efectos para determinar quién es elegible y determinar las reglas que usarán para elaborar las reglas colectivas que a su vez afectan las operacionales.

Este trabajo se enfoca en las reglas operacionales ya que es el primer escalón para conformar una institución. De este modo se definen las normas de uso diario que deben cumplir todos los agentes del ambiente que interactúen con el módulo de registro.

3.5.2. Constitución de la Institución

Entendiendo por qué se conforma la institución y la naturaleza de la misma, se definen el objeto y la misión sobre la cual se constituye.

Objeto

Es objeto de la Institución de Registro, registrar el ciclo de vida e identificar a los agentes del sistema y organizar los procesos electorales y los mecanismos de participación, en orden a apoyar la administración de justicia y el fortalecimiento democrático del sistema[1].

Naturaleza

El módulo de Registro es una Institución de creación Constitucional, que forma parte integrante del sistema, el cual contribuye, conjuntamente con las demás Instituciones, al modelo social-inspirado del proyecto TLÖN así como lo relativo a la identidad de los agentes sobre el ambiente para permitir la interacción con las demás partes del sistema.

Misión

Es misión de la Institución de Registro, garantizar la identidad de todos los agentes del sistema, el registro del estado actual de los agentes, que se disponga de su información a quien deba solicitarla, se certifique mediante los instrumentos idóneos y se garantice su confiabilidad y disponibilidad plenas.

3.6. Funciones de la Registraduría

Las funciones que aplican para la Institución son[2].

1. Proponer las iniciativas sobre proyectos de ley y presentarlos a consideración por conducto del Registrador Nacional, así como los decretos y demás normas relacionadas con la función de registro civil.

2. Adoptar las políticas del Registro de Agentes y atender lo relacionado con la adopción, ejecución y control de los planes y programas propios del registro con miras a garantizar su óptimo funcionamiento.

[1] Basado en el Decreto 1010 de 2000 [1]
[2] Basado en las funciones de la Registraduría Nacional del Estado Civil [2]

3. Garantizar en el ambiente y permitir desde ambientes externos, la inscripción confiable y efectiva de los hechos, actos y providencias sujetos a registro, definir las autoridades habilitadas para que concurran en el cumplimiento de dicha función, y conocer mediante los actos administrativos pertinentes de todo lo relativo a cancelaciones, reconstrucciones, anulaciones, modelos de expedición y demás actos sobre el registro.

4. Brindar información sobre los registros guardados de los agentes de conformidad con las normas establecidas.

5. Atender el manejo, clasificación, archivo y recuperación de la información relacionada con el registro.

6. Difundir las normas y procedimientos a seguir dentro del proceso de registro.

7. Asignar el Número Único de Identificación, al momento de hacer la inscripción de nacimiento en el Registro de los agentes y ejercer los controles físico, lógico y técnico, para que dicho número sea exclusivo a cada ciudadano y exista un único documento de identificación.

8. Atender todo lo relativo al manejo de la información, las bases de datos, el Archivo de Identificación y los documentos necesarios para el proceso técnico de la identificación de los agentes.

9. Adelantar inspección y vigilancia de los servicios de registro del estado de los agentes.

10. Dirigir y organizar el proceso electoral y demás mecanismos de participación y elaborar los respectivos calendarios electorales.

11. Llevar el Censo Electoral.

12. Coordinar con los organismos y autoridades competentes del Estado las acciones orientadas al desarrollo óptimo de los eventos electorales y de participación.

4 Legislación de la institución y diseño del sistema robusto

En este capítulo se explicará el modelo que se diseñó e implementó para el desarrollo del proyecto, bajo qué conceptos y bases se conformó la Institución de Registro, las funciones y reglas que aplican tanto para el módulo como para sus funcionarios y las que deben cumplir todos los agentes que se registren en esta institución con el fin de mantener la integridad del módulo y del ambiente.

Con la ejecución de este trabajo se constituye la primera institución social-inspirada del proyecto TLÖN y brinda las bases par que se puedan conformar más de ellas de acuerdo a la evolución que va teniendo el sistema multiagente y la auto-organización del mismo.

4.1. Normatividad

4.1.1. Normas Generales de la Institución de Registro

A continuación se presentan las normas que cobijan a la institución de registro[1]:

1. Sólo puede existir una única institución de registro de agentes y servicios por ambiente.
2. La institución debe contar como mínimo con un agente con funciones administrativas, quien será el encargado de corroborar los datos almacenados.
3. La institución debe proveer la información de los agentes, análogo a unas páginas blancas, a los demás miembros del sistema.
4. La institución tiene la autoridad para aceptar o denegar el registro de un agente dentro del sistema si este determina que no cumple con todos los requisitos para el proceso o que es un registro duplicado.
5. Las funciones de la institución le permiten ejecutar las siguientes acciones sobre sí mismo: registrar, des-registrar, modificar y buscar.
6. La información almacenada en el Archivo de registro debe permanecer a lo largo de la historia, no se permitirá la eliminación de registros bajo ningún criterio.

[1]Basado en el estándar FIPA [14]

4.1.2. Normas para el Registro de Agentes

Con el fin de manejar un estándar en el módulo de registro para todos los agentes que quieran ingresar al ambiente, se definirán las siguientes normas que se deben cumplir para que un agente sea reconocido en el sistema:

1. Todo agente que desee hacer parte del sistema multiagente debe tener como mínimo la siguiente información: Nombre, Comunidad, Función, Estado y Ubicación (parámetros descritos anteriormente). Una vez sea validada la información, el módulo retornará el número UID con el cual se podrá identificar el agente.

2. En condiciones normales, cada vez que el agente modifique o actualice los datos previamente almacenados, este debe informar al registro el parámetro a actualizar. Esto puede ser ejecutado en cualquier momento por cualquier razón pero debe contar con la aprobación del registro.

3. El agente que administrará el módulo de registro, en este caso el registrador, debe ejecutar validaciones periódicas de los datos que se encuentran almacenados con el fin de corroborar su veracidad. En caso de que encuentre alguna diferencia se notificará con el agente la información y de no recibir respuesta se actualizará con los datos capturados.

4. En condiciones normales, el periodo de vida de un agente termina con el des-registro del mismo en el módulo, proceso en el cual se actualizará el estado correspondiente al UID asignado previamente.

5. Todo agente que se encuentre registrado debe contestar a las solicitudes que le haga el representante de la institución con la información de su estado actual.

4.1.3. Normas Generales del Directorio de Servicios

Una vez definido el registro a nivel de agentes, se definirá la normatividad que aplica para el registro de servicios, qué aspectos se deben tener en cuenta y las reglas generales a considerar. Esta parte será considerada como el directorio de páginas amarillas del ambiente[2]:

1. El directorio será el responsable de publicar la existencia de todos los servicios que los agentes quieran hacer conocer.

2. No es mandatario para los agentes el registro de todos los servicios que cada uno ofrece.

3. El directorio no garantiza la funcionalidad de los servicios que allí se encuentren, sin embargo si debe mantener los datos actualizados.

[2]Basado en el estándar FIPA [14]

4. En cualquier momento, y por cualquier motivo, el agente puede solicitar que se modifique uno de los parámetros de su servicio.

5. Las funciones del directorio le permiten ejecutar las siguientes acciones sobre sí mismo: registrar, des-registrar, modificar y buscar.

4.1.4. Normas del Agente Supervisor del Módulo

A continuación se presentan la normas que debe seguir el agente a cargo del módulo de registro de acuerdo al punto 2 de la sección 4.1.1 y el cuál será responsable de su correcto funcionamiento, dado que es un sistema social-inspirado, alguna de sus funciones se basaron en [2]:

1. Todo agente que esté en función del módulo de registro deberá estar correctamente inscrito en el directorio como cualquier agente del sistema.

2. Atender a las solicitudes de su coordinador o superior para mantener el cumplimiento de las funciones.

3. Fijar los procesos, planes, programas y estrategias para la disponibilidad de los datos que se encuentran registrados.

4. Vigilar que la identificación de los agentes y el registro de los mismos contenga información válida y certera.

5. Garantizar la constante actualización de los datos de la institución de registro si corresponde.

6. Generar archivos de respaldo con la información del directorio una vez ha sido validad por los medios correspondientes.

4.2. Organigrama de la Registraduría

En la Figura **4-1** se observan los funcionarios de la institución de registro y su estructura jerárquica, de esta forma cada uno tendrá unas tareas específicas con el objetivo de velar por el correcto cumplimiento de las normas y funciones de la institución.

Figura 4-1: Organigrama de la Institución de Registro.

4.3. Definición del Modelo de Software

Teniendo las normas que se van aplicar para el proceso de registro de agentes, se va a especificar la información que debe enviar cada uno para que esta regulación se cumpla a cabalidad.

Como la mayor parte de los componentes del proyecto TLÖN se desarrollan en Python3, entre ellos el Sistema Multiagente (SMA), la programación de los agentes y los escenarios de prueba se realizaron bajo este mismo lenguaje. En las siguientes secciones se detalla la construcción del mismo.

4.3.1. Almacenamiento de Datos

Los datos en el módulo de registro serán almacenados en un diccionario de datos, ya que viene compilado en el lenguaje [30], no tiene cargas altas sobre el sistema, son descentralizados y el arreglo de llaves y valores permite manejar los datos correspondientes de cada agente de manera sencilla y eficaz.

La información que contendrá cada registro en el diccionario está basada en las recomendaciones del estándar FIPA y se compone de lo siguiente:

- **UID**: Unique Identifier. Identidad propia de cada agente proporcionada por un identificador, en este caso la institución de registro. El UID será la llave de cada registro y su valor será dado por la función uuid4 que crea un valor aleatorio de UUID (Universally Unique Identifier). Este parámetro se escogió ya que el número de posibles UUID es de 16^{32} [25] lo que significa que es muy poco probable de que se repita en el ambiente del sistema que se está construyendo.

- **Name**: Nombre con el cual se va a conocer el agente en el sistema.
- **Community**: Comunidad a la cual va a pertenecer cada agente dentro del sistema multiagente.
- **Function**: Descripción del trabajo que va a desarrollar cada agente para cumplir con la finalidad por la cual fue creado.
- **State**: Estado en el cual se encuentra el agente. Este debe permanecer actualizado en línea. Los diferentes estados que se pueden presentar son Activo, Inactivo, Disperso.
- **Location**: Cada agente debe ser capaz de ser ubicado en el sistema por lo tanto en este parámetro se almacenará y se debe mantener actualizado constantemente.

Teniendo en cuenta lo anterior, todos los datos a almacenar excepto por el UID deben ser suministrados por el agente al momento de ingresar por primera vez al sistema con el fin de ir alimentando el diccionario de datos.

La actualización de estos se desarrollará en trabajo conjunto entre los funcionarios del módulo de registro y los agentes que se encuentren en el ambiente.

4.3.2. Procedimiento para el Registro de Agentes

A continuación se muestra la interacción entre los diferentes actores del sistema y su relación con el módulo que se está desarrollando para el proceso de registro.

Los actores principales en este paso son: los agentes, quienes se van a registrar; la institución, quien contiene todo la lógica para la inscripción y/o actualización de un usuario; y la mensajería instantánea, quien brinda la comunicación entre agentes pero también necesita un proceso de registro para habilitarla.

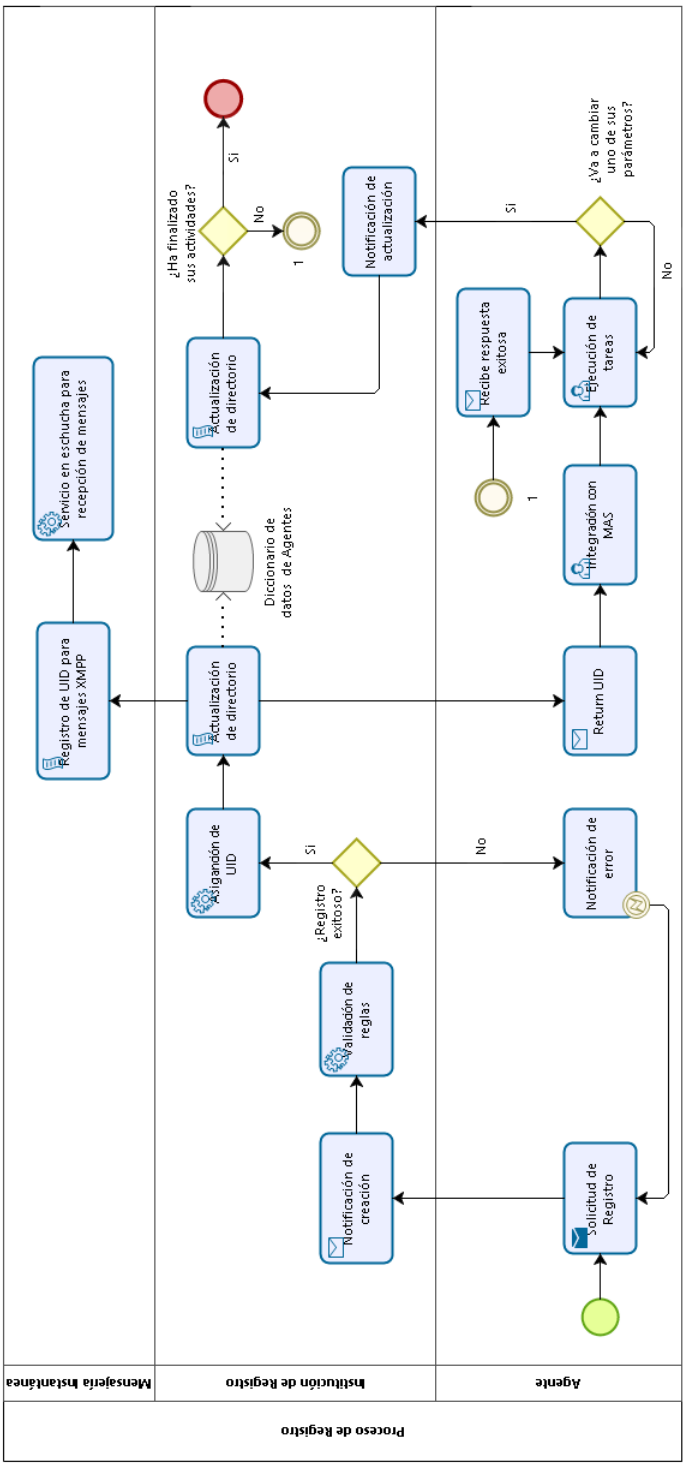

Figura 4-2: Proceso de Registro de Agentes

4.3.3. Diagrama de Estado del Agente

Ahora se van a mostrar los agentes que hacen parte de la institución en calidad de funcionarios de acuerdo a lo establecido previamente.

Agente Supervisor

Este agente es el encargado de construir el mensaje y enviárselo directamente a los demás agentes de su sistema para corroborar que la información que han registrado sea la correcta y esté actualizada.

En el diagrama de la Figura **4-3** se observan los estados en el que se va a encontrar el agente supervisor del módulo con el fin de cumplir sus objetivos.

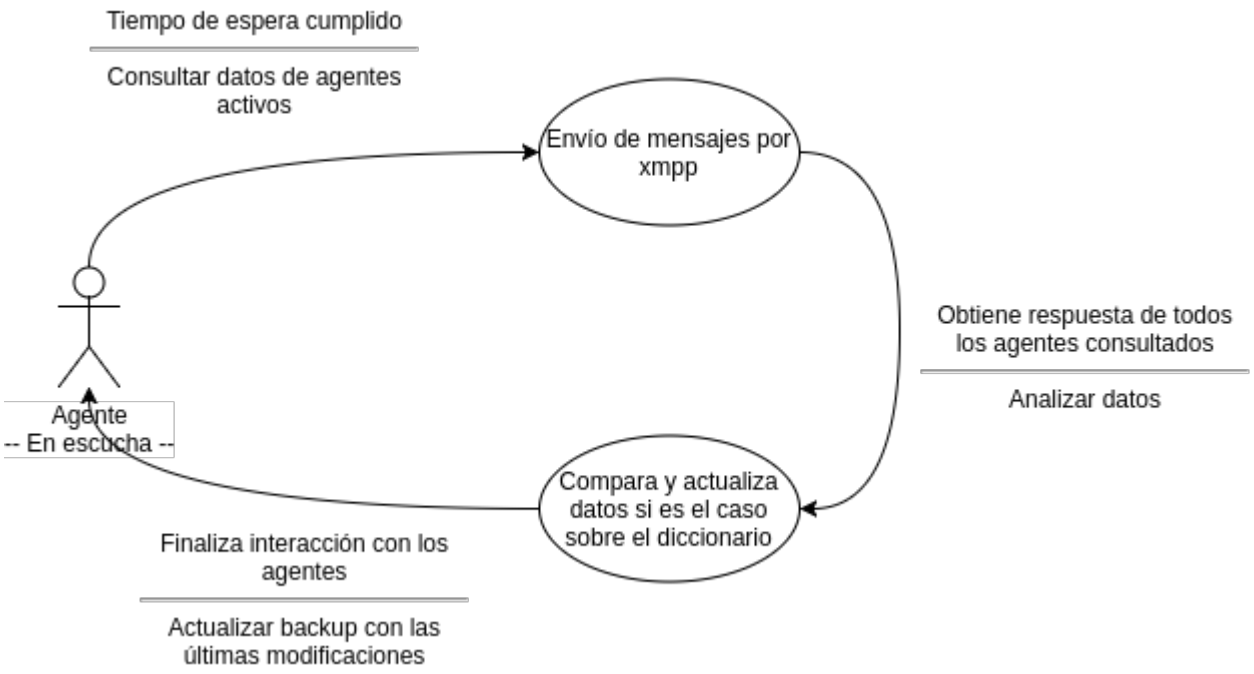

Figura 4-3: Diagrama de Estado del Agente Supervisor.

Agente Coordinador

Este agente verificará que el supervisor se esté ejecutando 7x24, ya que mientras el sistema esté activo debe haber un supervisor que garantice la calidad de los datos para que el módulo sea robusto.

En el diagrama de la Figura **4-4** están los estados del Coordinador.

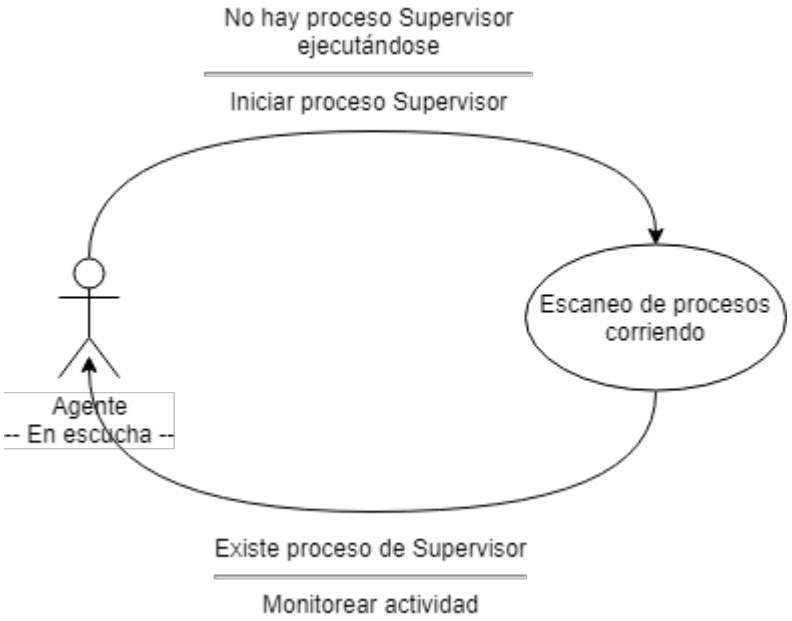

Figura 4-4: Diagrama de Estado del Agente Coordinador.

Agente de Soporte

Este agente es el encargado de velar por el diccionario para que la información sea respaldada y en caso de exista una alteración del mismo se restauren los datos, minimizando el daño que se ocasiona a la institución.

En el diagrama de la Figura **4-5** están los estados del agente de soporte.

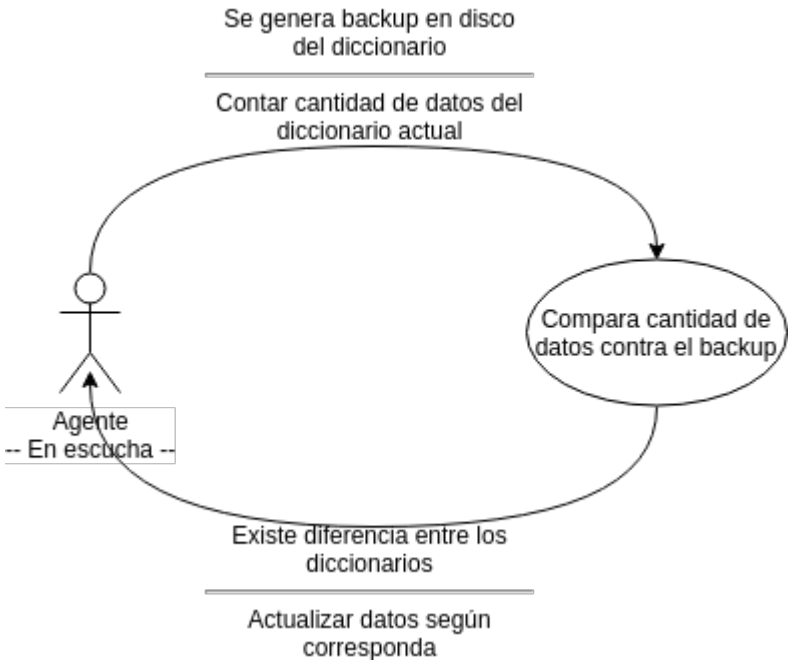

Figura 4-5: Diagrama de Estado del Agente de Soporte.

4.3.4. Diagrama de Secuencias

En el siguiente diagrama se muestra la secuencia de cómo se va a garantizar la robustez del módulo. El agente que representa estas actividades es el supervisor ya que contacta a cada uno de los agentes para asegurar la fiabilidad de la información.

Así mismo está la institución en el centro ya que es la que gestiona y administra todos los procesos referente a ella y el sistema multiagente quienes son los usuarios finales de todo el sistema y en favor de quienes se trabaja.

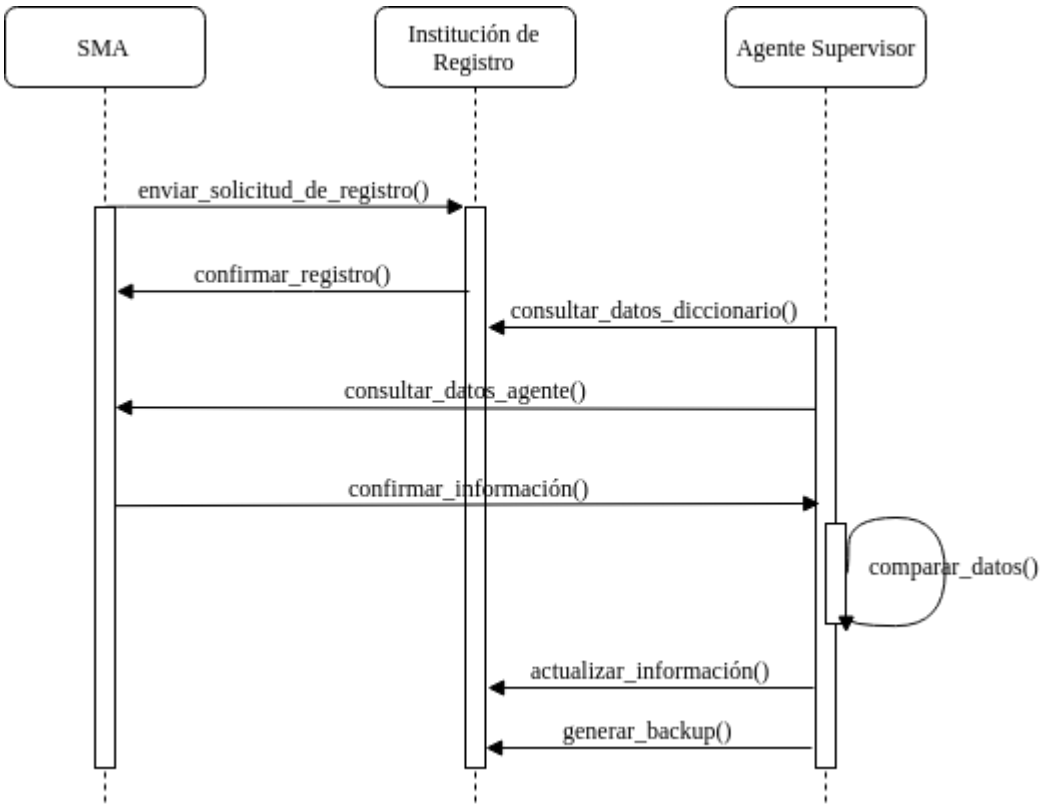

Figura 4-6: Diagrama de Estado del Agente Supervisor.

4.3.5. Diagrama de Clases

Con la información anterior se construye el diagrama de clases en donde se observan los atributos, las operaciones y las asociaciones de cada actor.

Como lo dice la norma sólo puede existir una institución de registro por ambiente, sin embargo los funcionarios asociados a ella como el supervisor, coordinador o de soporte, pueden ser tantos como sean necesarios para que se cumplan las metas para los que fueron creados.

Figura 4-7: Diagrama de Estado del Agente Supervisor.

A continuación, en los escenarios de prueba, se explica con mejor detalle cada uno de los agentes que trabaja para la institución de registro y los procedimientos que se aplicaron para que cumpliera con sus funciones. De igual forma se demuestra por qué cada uno de ellos aporta un factor importante para mantener el sistema robusto.

5 Validación de Escenarios de Pruebas

Para la ejecución de las pruebas se implementó la red ad-hoc que se observa en la Figura **5-1**, en este caso cada nodo representa un Sistema TLÖN, lo que significa que cada uno debe tener su propia Institución de Registro y el principal objetivo es comunicar toda la red para conocer los agentes de cada sistema o ambiente.

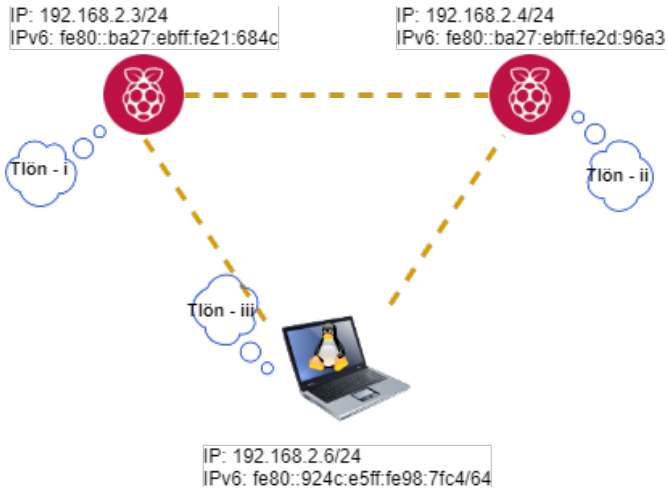

Figura 5-1: Diagrama de Red

El enrutamiento entre los nodos se realizó a través del protocolo Batman, el cual es un protocolo para redes ad-hoc que descentraliza la información de los nodos y garantiza un enrutamiento altamente adaptable y sin bucles, a la vez que reduce el costo de procesamiento y de tráfico [4].

Sin embargo para la comunicación entre agentes se utilizó el estándar XMPP para hacer más dinámica la interacción entre ellos. La comunicación solo pide que los usuarios se encuentren registrados en el servidor, por lo que una vez se registran correctamente frente a la institución, se habilita la inscripción para la comunicación XMPP con el mismo UID con el que trabaja [33].

El servidor utilizado para este fin fue ejjaberd que funciona sobre la plataformas UNIX y como cliente se ejecutó Sleekxmpp que es compatible con el servidor y desarrollado para

python3 [38] [39], lenguaje sobre el cual también fue realizado el sistema TLÖN.

Una vez establecida la interacción entre los ambientes, con cada uno de los escenarios que se plantean se pretende medir y demostrar que el sistema de registro diseñado e implementado es robusto. Para medir esta robustez en los Escenarios 1 y 2 se crea un servicio que grafica la cantidad de nodos que responden respecto a la cantidad de nodos que tiene el diccionario como 'Activos'.

$rate = \frac{Agentes_{contestaron}}{Agentes_{total_activos}}$

El cálculo se hace para cada ciclo del programa. Se define como ciclo cuando el software obtiene los diccionarios de los tres ambientes, valida cuáles están 'Activos', le envía un mensaje a cada uno de ellos y consolida los datos de cuántos contestaron. Estos test se pueden encontrar en el repositorio $https://github.com/jpospinalo/MAS_TLON/tree/register/test$.

La medida definida previamente se tomó como base ya que permite mostrar de manera sencilla cómo el sistema es capaz de recuperarse a fallos y mantener la información actualizada, de igual forma se hace el comparativo del comportamiento cuando no se implementa robustez respecto a cuando si se hace.

5.1. Escenario 1

En este escenario se simula la falla sobre varios agentes del sistema los cuales de forma súbita se desconectan de la red, allí se debe validar que el diccionario de datos en donde está almacenada toda la información de los agentes registrados se actualice con el estado 'Inactivo' correspondiente al nuevo status del agente.

El actor principal es el agente supervisor, encargado de validar que toda la información que se encuentre en el diccionario sea verídica, él es quien debe detectar que los estados, ubicaciones o descripciones han cambiado y actualizar los datos correspondientes.

Se espera que cuando los agentes "desaparezcan"del ambiente, a pesar de que se presente una perturbación en la información de registro, el agente supervisor note este cambio y actualice los registros según corresponda en el menor tiempo posible.

En la Figura **5-2** de observa el proceso de cómo actúa el agente supervisor frente a la información que contiene la institución de registro y los planes de acción para mantenerla actualizada. Todo este proceso está ceñido a las normas que definieron previamente.

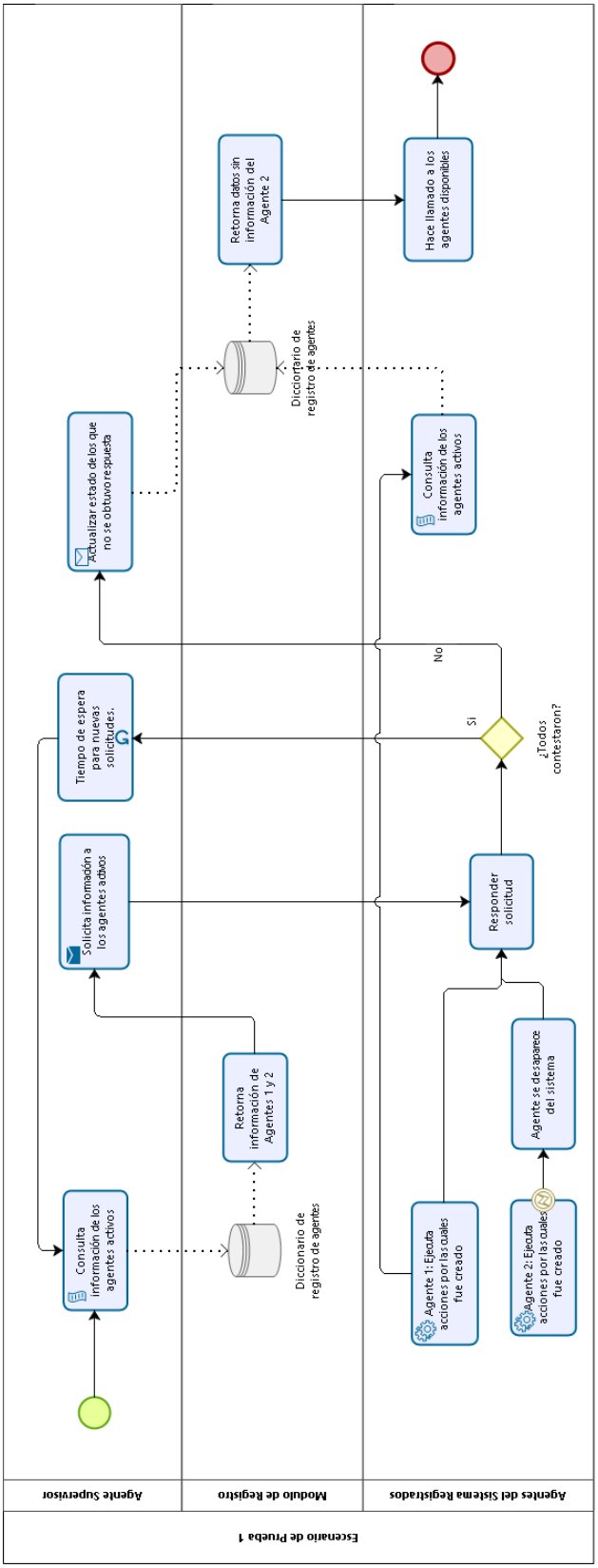

Figura 5-2: Diagrama de procesos del escenario 1.

Con el proceso definido se ejecuta el escenario de prueba de acuerdo a los datos de la Tabla **5-1**. Se debe tener en cuenta que la falla de los agentes se hace de manera aleatoria en diferentes ciclos del programa y que el total de agentes corresponde a todos los que se encuentran registrados en el diccionario independiente del estado.

Tabla 5-1: Datasheet Escenario 1

Agentes Iniciales Activos	30
Total Agentes en Diccionarios	72
Agentes Supervisores	3
Ciclos	50
Falla Simulada	Los agentes se detienen súbitamente

Una vez finalizada la prueba se obtiene la siguiente gráfica:

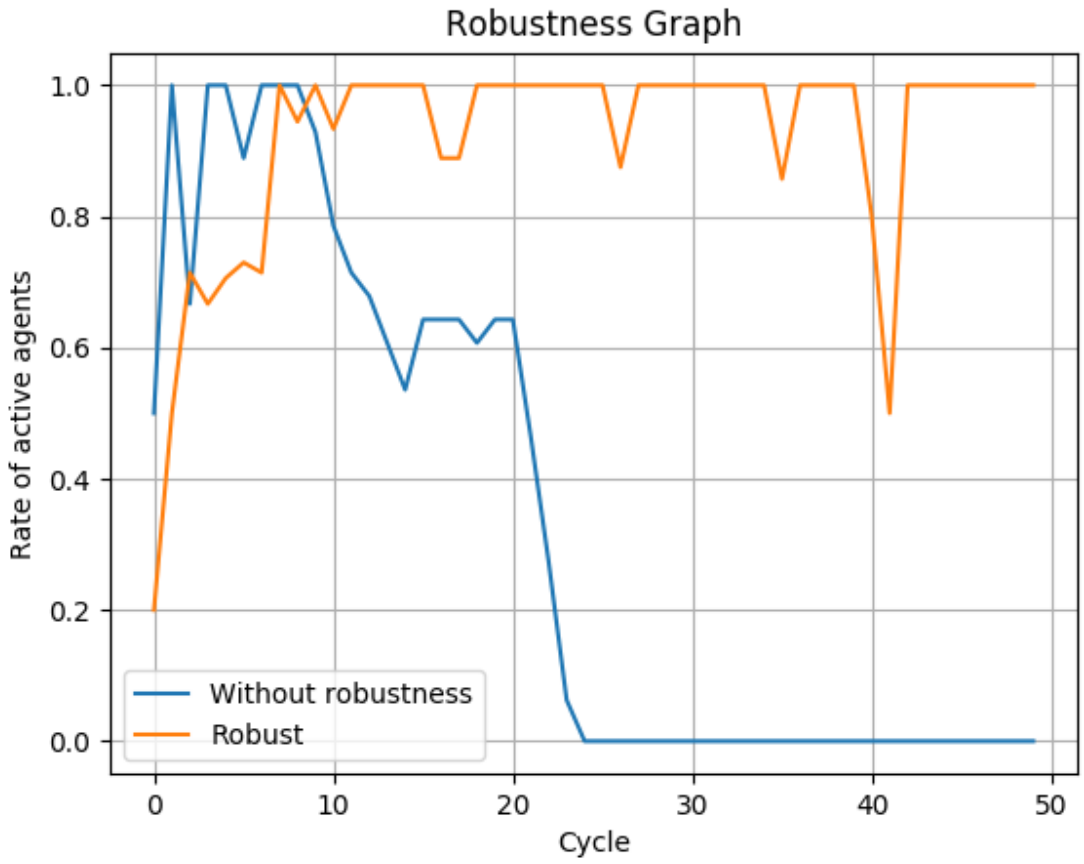

Figura 5-3: Gráfica de resultados para el escenario 1.

En ella se observan los comportamientos cuando el sistema es robusto respecto a cuando no lo es. El segundo caso significa que ningún agente supervisor se estaba ejecutando por lo tanto la integridad de la información va decreciendo hasta llegar a cero.

De acuerdo a la figura **5-3** cuando el sistema no tiene ningún proceso de robustez pareciera que por momentos se recuperara de una perturbación, sin embargo esto no es así. Esta novedad se da ya que como la base de la ecuación son los agentes que están activos, si uno de ellos finaliza correctamente este valor disminuye lo que aumenta el resultado del ratio. Al final de la prueba queda en cero ya que todos los agentes terminaron sus acciones o de manera súbita, por lo que ninguno contesta las peticiones, si embargo en el diccionario aún siguen apareciendo agentes como si estuvieran activos.

Caso contrario ocurre con el sistema robusto, ya sea porque el agente finalizó correctamente o porque el supervisor ha actualizado los estados según las validaciones realizadas, el diccionario va mostrar los estados reales durante la mayor parte de los ciclos.

5.2. Escenario 2

En este escenario la falla se va a concentrar en el agente supervisor que se encarga de mantener actualizado el módulo, si esto se llega a presentar debe entrar en acción un segundo agente que valide los datos y siga cumpliendo con la propiedad de robustez.

El indicador de éxito es simular el escenario 1 (agentes que terminan inesperadamente) incluyendo a los agentes supervisores en los que presentan fallas. Cuando se realice una consulta la información que se retorne debe ser acorde al estado actual del sistema multiagente.

El agente supervisor al ser un software se puede ver como un proceso del sistema, en este escenario entra en función el agente coordinador quien vigila que ese proceso esté activo constantemente, en caso de que no esté lo inicia, lo que significa que un nuevo agente supervisor es creado y comienza a cumplir sus funciones.

Este proceso se ve con detalle en la Figura **5-4** y como su nombre lo indica el coordinador sólo coordina el agente supervisor, sin embargo no tiene interacción directa con los datos de la institución o con los demás agentes.

En la Tabla **5-2** se presenta con detalle los componentes que hicieron parte de este escenario, se específica los agentes de cada ambiente ya que el total varía según el número de veces que haya fallado el supervisor.

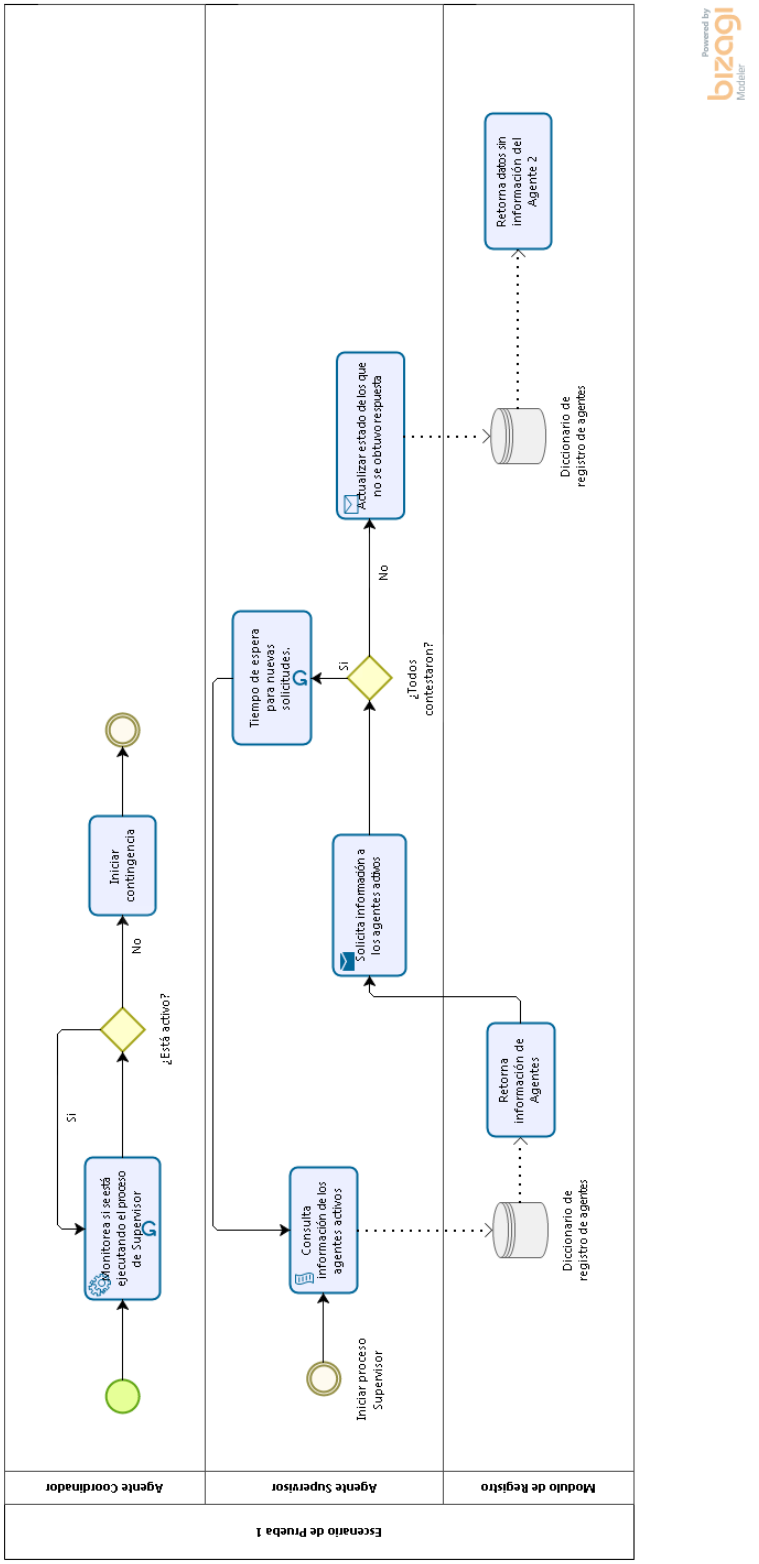

Figura 5-4: Diagrama de procesos del escenario 2.

Tabla 5-2: Datasheet Escenario 2

Agentes Activos	40
Agentes en diccionario Tlön-i	60
Agentes en diccionario Tlön-ii	68
Agentes en diccionario Tlön-iii	51
Agentes Supervisores	74
Agentes Coordinadores	3
Total Agentes en Diccionarios	179
Ciclos	50
Falla Simulada	Los agentes particulares y supervisor se detienen súbitamente

Una vez finalizada la prueba se obtienen los siguientes resultados:

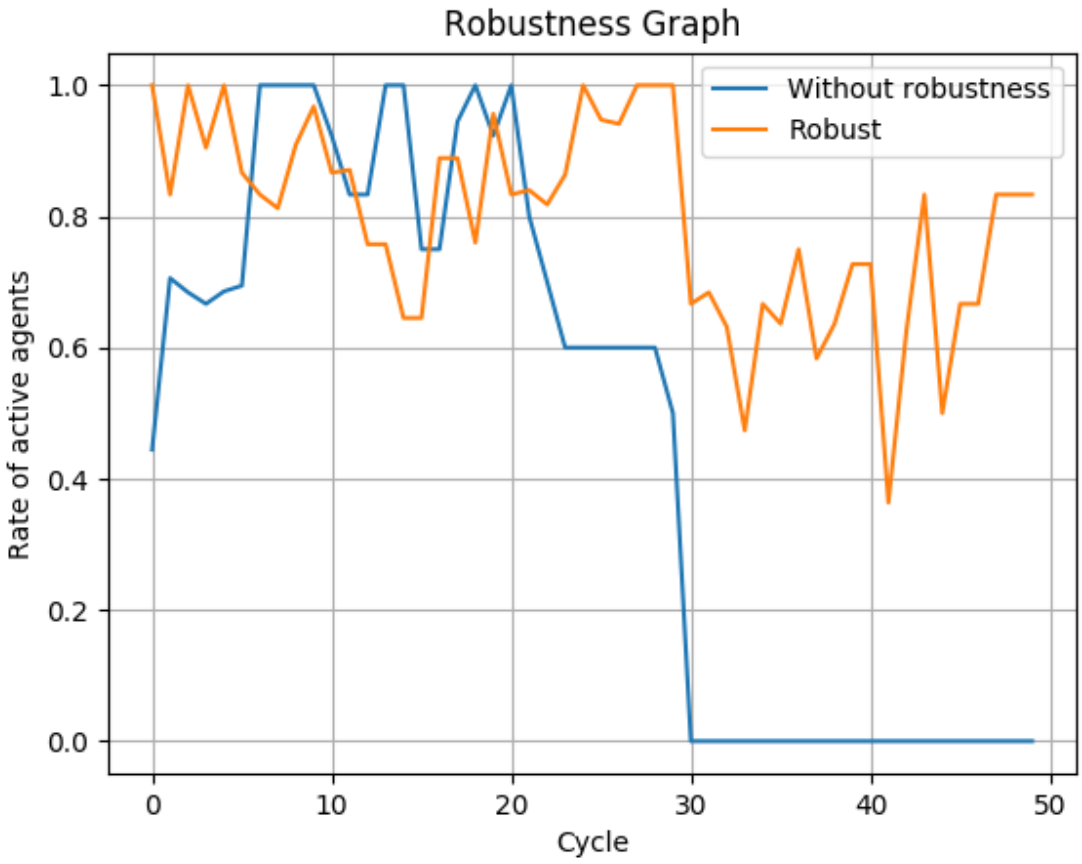

Figura 5-5: Gráfica de resultados para el escenario 2.

Según la figura anterior, cuando el sistema no cuenta con un agente coordinador (línea sin robustez) pero sí con un agente supervisor, el servicio llega a ser robusto hasta el ciclo 20 que es hasta cuando permaneció activo el agente. De ahí en adelante si el supervisor llega a presentar un error el sistema deja de mantener esa robustez y comienza a decaer hasta llegar nuevamente a 0.

Por otro lado cuando se tiene un agente coordinador, siempre trata de mantener los componentes para que tengan la capacidad de operar a niveles sub-óptimos y se siga manteniendo dentro de la definición de sistema robusto tal como se precisa en [34].

Entrando en detalle se observan que las perturbaciones son mayores y toman más tiempo recuperarse de ellas, algo que se suponía ya que se debe considerar el tiempo que tarda el agente supervisor en crearse de nuevo y que debe iniciar desde cero la interacción con la institución y los demás agentes para empezar a tomar decisiones sobre estos.

5.3. Escenario 3

Una de las formas de hacer el módulo de registro robusto es almacenar los datos que se tienen en memoria al disco local de una máquina. En caso de que se reinicie la instancia en donde se está guardando los datos o de que el diccionario sufra de alguna perturbación, la información no se perderá por completo y se tendrá un punto de recuperación.

El tercer escenario va a simular esa perdida de datos y se mostrará cómo se restaura la información sin que los agentes del sistema se percaten de que hubo una intermitencia. Aunque el rendimiento se vea afectado o los tiempos de respuesta de las consultas, el éxito de este escenario es que cuando sea que se consulte el diccionario, la información que se extraiga sea verídica.

Cuando el agente supervisor valida que la información es correcta, genera un backup del diccionario y este se toma como base para que el agente de soporte compare los datos. Si llega a detectar que el diccionario actual tiene menos datos que el backup (significa que parte de la información fue suprimida) restaura este último backup junto con la información que tenga actualmente para mantener la integridad de los datos.

En caso contrario, si el backup tienen menos información, actualiza este último para tener un punto de respaldo confiable. De igual forma todo este proceso se puede ver en la Figura **5-6**.

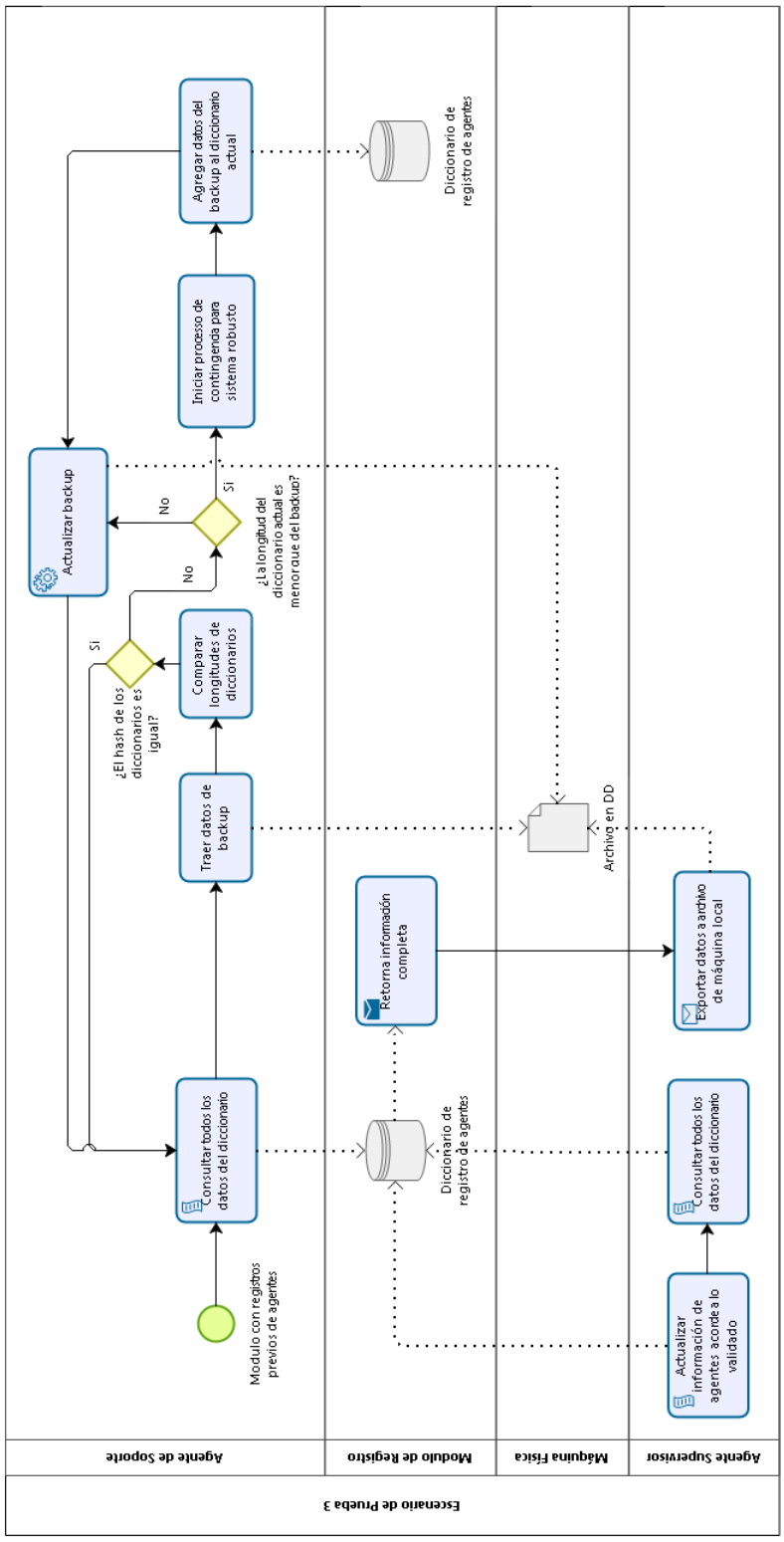

Figura 5-6: Diagrama de procesos del escenario 3.

Para este este escenario, la información es importante que se respalde por ambiente ya que sólo pertenece a su institución y es independiente de los demás sistemas externos. Es por ello que en la Tabla **5-3** se muestra con detalle los componentes que se usaron pero enfocados a un solo nodo.

Tabla 5-3: Datasheet Escenario 3

Agentes en diccionario Tlön-iii	51
Agentes de Soporte	1
Agentes Supervisores	2
Ciclos	50
Falla Simulada	Se borran registros del diccionario de manera aleatoria.

Al finalizar las pruebas se obtienen los siguientes resultados:

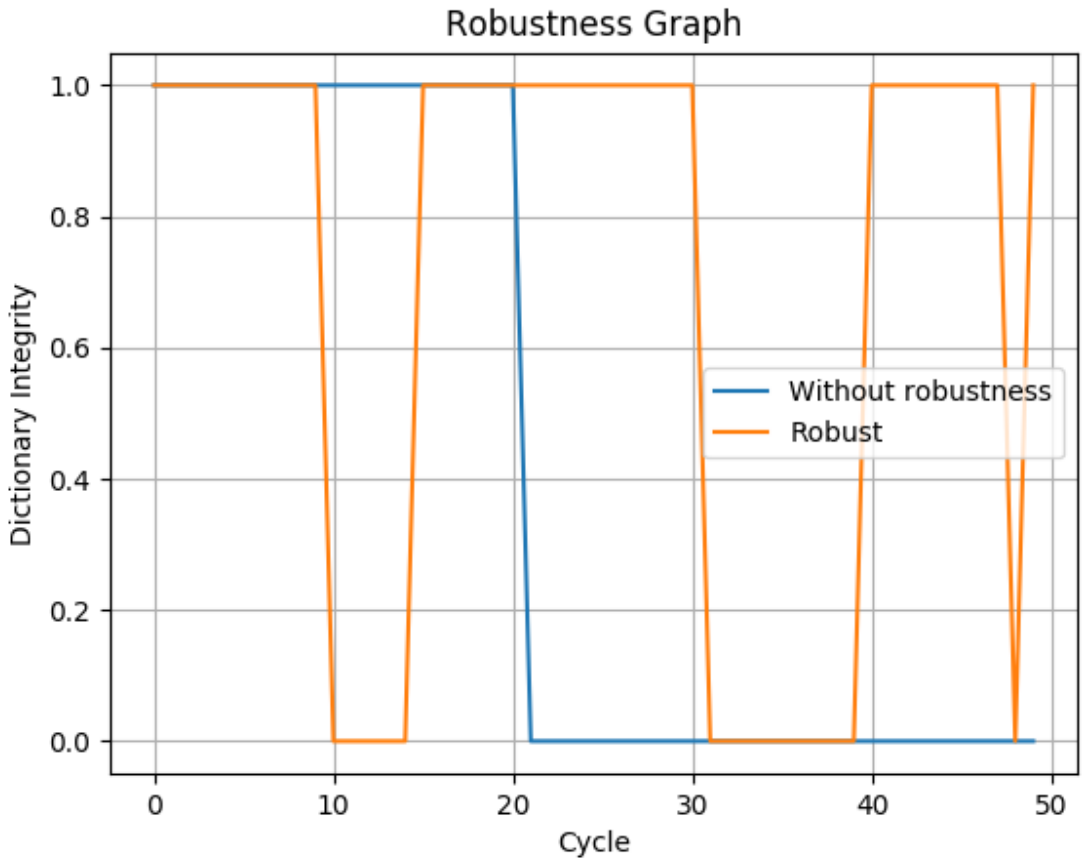

Figura 5-7: Gráfica de resultados para el escenario 3.

En este caso sólo existen dos estados, 1 si el backup y el diccionario actual son exactamente iguales y 0 si no lo son, se hizo así ya que cuando estos dos objetos son diferentes significa o que la información actual no ha sido respaldada correctamente o que el diccionario ha sufrido algún tipo de perdida, lo que en cualquiera de los dos escenarios representa una falla del servicio.

Consolidando estos datos es claro que existe una diferencia en la fiabilidad de la información cuando se presentan perturbaciones, que es lo más común en la red ad-hoc, y no se ha aplicado las medidas necesarias para mitigarlos. De allí la importancia de que un servicio tan vital para la comunidad como es el registro se haga robusto en esta clase de organizaciones dinámicas. Estas y otras conclusiones se describen en el Capitulo 6.

6 Conclusiones y Recomendaciones

6.1. Conclusiones

- Lo principal para un sistema social-inspirado es que los agentes puedan relacionarse entre sí pero primero deben conocer quién se encuentra en su ambiente. La forma para obtener esta información es mediante la gestión de un directorio que tenga consolidado todos los datos. De allí la motivación de este trabajo para el desarrollo de un módulo de registro y lo más importante que fuera robusto.

- Un sistema puede llegar a ser tan robusto como las condiciones de la red se lo permitan. Esto se debe a que la comunicación entre los diferentes componentes afectan el tiempo en que se puede recuperar de un fallo y por lo tanto la capacidad de operar de un servicio. En la ejecución de los escenarios de prueba se denota este factor ya que la respuesta a las solicitudes de los agentes presentó un delay que se debió tener en cuenta a la hora de obtener los resultados.

- La acogida e implementación de los aspectos generales del modelo de referencia FIPA permite que el sistema sea interoperable entre plataformas y estructurado bajo un estándar que se ha venido desarrollando en los últimos años.

- Según lo evidenciado en los escenarios de prueba, un sistema va a ser más robusto si el funcionario principal de la institución no presenta inconsistencias o si hay más de uno que de manera simultanea se esté ejecutando para que lo reemplace. La creación y puesta en acción de uno nuevo, aunque se haga de forma automática, afecta en el tiempo de recuperación frente a una perturbación.

- El concepto de institución permite regular el funcionamiento y comportamiento de un grupo auto-organizante de agentes. Estas reglas o normas se conforman por las necesidad que surge de brindar un orden a la interacción entre individuos y de estandarizar los procedimientos requeridos para la identificación de los agentes.

6.2. Recomendaciones

- Es necesario aumentar la cantidad de agentes y ambientes para la ejecución de pruebas en donde se determine hasta qué punto el sistema continúa siendo robusto y opera con los mínimos requerimientos de acuerdo a las soluciones planteadas.
- Es importante diseñar e implementar la función de organización de elecciones y ponderación de resultados para los altos niveles de la institución de registro. Esto permite ir evolucionando en el concepto de institución al segundo nivel de elecciones colectivas con el cual se pretende establecer la forma de hacer las políticas con las que se van a regir los agentes del primer nivel.
- Se recomienda el diseño de otra institución con la que se pueda interactuar para seguir aumentando los niveles de institucionalidad y definir a qué normas van a estar sujetas y cuáles las pueden modificar.
- Es necesario validar más puntos de falla que pueda sufrir el sistema para crear las acciones necesarias que permitan mitigar de manera autónoma la indispobilidad del servicios.

Bibliografía

[1] *DECRETO 1010 DE 2000*

[2] *Registraduría Nacional del Estado Civil*

[3] *Tlön-grupo de investigación en redes de telecomunicaciones dinámicas & lenguajes de programación distribuidos.*

[4] Aichele, Corinna ; Wunderlich, Simon ; Neumann, Axel ; Lindner, Marek: Better Approach To Mobile Ad-hoc Networking (B.A.T.M.A.N.). (2008)

[5] Alippi, Cesare: *Intelligence for Embedded Systems.* Cham : Springer International Publishing, 2014. – ISBN 978–3–319–05277–9

[6] Bakht, H.: *Redes móviles Ad-hoc:.* 2018

[7] Bar-Yam, Yaneer: General Features of Complex Systems. En: *Encyclopedia of Life Support Systems (EOLSS UNESCO Publishers, Oxford, UK, 2002)* I (2002)

[8] C. Siva Ram Murthy.: *Ad Hoc Wireless Networks : Architectures and Protocols.* Prentice hall, 2012. – ISBN 0133007065

[9] Cardoso, Jorge. ; Ferguson, Donald ; Méndez Muñoz, Víctor ; Helfert, Markus: *CLOSER 2016 proceedings of the 6th International Conference on Cloud Computing and Services Science.* SCITEPRESS = Science and Technology Publications, feb 2016. – 333–338 p.. – ISBN 9789897581823

[10] Cliffe, Owen ; De Vos, Marina ; Padget, Julian: Embedding Landmarks and Scenes in a Computational Model of Institutions. (2008)

[11] Dressler, Falko: Self-Organization in Ad Hoc Networks: Overview and Classification / University of Erlangen. Erlangen, Germany, 2006. – Informe de Investigación

[12] Dunin-Keplicz, Barbara ; Verbrugge, Rineke: *Teamwork in Multi-Agent Systems.* Chichester, UK : John Wiley & Sons, Ltd, jun 2010. – ISBN 9780470665237

[13] Feeney, L.M. ; Nilsson, M.: Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. En: *Proceedings IEEE INFOCOM*

2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No.01CH37213) Vol. 3, IEEE, 2001. – ISBN 0–7803–7016–3, p. 1548–1557

[14] FOUNDATION FOR INTELLIGENT PHYSICAL AGENTS. *FIPA Agent Management Specification*. 2004

[15] Gershenson, Carlos: *Artificial Societies of Intelligent Agents*, Fundación Arturo Rosenblueth, Tesis de Grado, 2001

[16] Gershenson, Carlos: *Design and Control of Self-organizing Systems*. CopIt ArXi. Mexico : CopIt ArXives, 2007

[17] Grimaldi, Ralph P.: *Discrete and combinatorial mathematics : an applied introduction*. Pearson Addison Wesley, 2004. – 24 p.. – ISBN 9780201726343

[18] Guo, Chao ; Xiong, Wei: Research on model description method of multi-agent system. En: *2015 6th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, IEEE, sep 2015. – ISBN 978–1–4799–8352–0, p. 603–606

[19] Johnson, Deborah G. ; Noorman, Merel: Principles for the future development of artificial agents. En: *2014 IEEE International Symposium on Ethics in Science, Technology and Engineering*, IEEE, may 2014. – ISBN 978–1–4799–4992–2, p. 1–3

[20] Karigiannis, John N. ; Tzafestas, Costas S.: Robustness and generalization of model-free learning for robot kinematic control using a nested-hierarchical multi-agent topology. En: *2012 4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, IEEE, jun 2012. – ISBN 978–1–4577–1200–5, p. 1140–1147

[21] Khue, Nguyen Tran M.: Developing an Intelligent Multi-Agent System based on JADE to solve problems automatically. En: *Systems and Informatics (ICSAI), 2012 International Conference on*, 2012, p. 684–690

[22] Kumar Sarkar, Subir ; Basavaraju, T.G ; Puttamadappa, C.: *Ad Hoc Mobile Wireless Networks*. CRC Press. 2013. – ISBN 978–1–4665–1446–1

[23] Loo, Jonathan. ; Lloret Mauri, Jaime. ; Ortiz, Jesus H.: *Mobile ad hoc networks : current status and future trends*. CRC Press, 2016. – 528 p.. – ISBN 9781439856512

[24] Lopes Cardoso, Henrique ; Oliveira, Eugénio: A Contract Model for Electronic Institutions. (2008), p. 27–40

[25] Lott, Steven.: *Functional Python Programming*. Packt Publishing, 2015. – ISBN 9781784397616

[26] Luotsinen, Linus J. ; Khan, Majid A. ; Boloni, Ladislau: A study of the robustness of agent performance in nine popular agent implementation paradigms. En: *2007 IEEE International Conference on Intelligent Computer Communication and Processing*, IEEE, sep 2007. – ISBN 1–4244–1491–1, p. 233–236

[27] Muntaner-Perich, Eduard ; Lluís De La, Josep ; Esteva, Rosa: Towards a Formalisation of Dynamic Electronic Institutions. (2008), p. 97–109

[28] Ostrom, Elinor.: *Governing the commons : the evolution of institutions for collective action*. Cambridge University Press, 1990. – 280 p.. – ISBN 0521405998

[29] Padghman, Lin ; Winikoff, Michael: *Developing Intelligent Agent Systems*. John Wiley & Sons, 2004. – ISBN 0–470–86120–7 (HB)

[30] Python Software Foundation. *5. Data Structures — Python v3.1.5 documentation*. 2012

[31] Russell, Stuart ; Dewey, Daniel ; Tegmark, Max: Research Priorities for Robust and Beneficial Artificial Intelligence. En: *AI Magazine* 36 (2015), dec, Nr. 4, p. 105. – ISSN 0738–4602

[32] Russell, Stuart J. ; Norvig, Peter: *Artificial Intelligence: A Modern Approach*. Pearson Education, 2003. – ISBN 0137903952

[33] Saint-Andre, Peter: Jabber Component Protocol. (2012), jan

[34] Santa Fe Institute. *Complexity Explorer*

[35] Sartor, Giovanni: Why Agents Comply with Norms, and Why They Should. Springer, Boston, MA, 2001, p. 19–43

[36] Scioscia, Floriano ; Ruta, Michele ; Di Sciascio, Eugenio: A swarm of Mini-MEs: Reasoning and information aggregation in ubiquitous multi-agent contexts. En: *CEUR Workshop Proceedings* Vol. 1387, CEUR-WS, 2015. – ISSN 16130073, p. 15–22

[37] SERUGENDO, GIOVANNA DI M. ; GLEIZES, MARIE-PIERRE ; KARAGEORGOS, ANTHONY: Self-organization in multi-agent systems. En: *The Knowledge Engineering Review* 20 (2005), Nr. 2, p. 165–189

[38] SleekXMPP. *Sign in, Send a Message, and Disconnect*

[39] SleekXMPP. *SleekXMPP Quickstart - Echo Bot*

[40] Takahashi, Kenichi ; Mitsuyuki, Yoshiki ; Mine, Tsunenori ; Sakurai, Kouichi ; Amamiya, Makoto: Design and Implementation of Security Mechanisms for a Hierarchical Community-Based Multi-Agent System. (2009), p. 134—-145

[41] Truszkowski, Walt ; Karlin, Jay: LNAI 1860 - A Cybernetic Approach to the Modeling of Agent Communities. (2000), p. 166–178

[42] Weiss, Gerhard: *Multiagent systems*. 2013. – 867 p.. – ISBN 9780262315166

[43] Wooldridge, Michael J.: *An introduction to multiagent systems / Michael Wooldridge*. John Wiley & Sons, 2002. – ISBN 9780470519462 (rustica)