A register module for agents communities through robustness property on ad hoc networks

Lina J. Alfonso *
National University of Colombia
School of Engineering
Research Group TLÖN
ljalfonsos@unal.edu.co

Juan P. Ospina †
National University of Colombia
School of Engineering
Research Group TLÖN
jpospinalo@unal.edu.co

Jorge E. Ortiz ‡
National University of Colombia
School of Engineering
Grupo de investigación TLÖN
jeortizt@unal.edu.co

***Abstract*—Ad hoc networks are decentralized systems that work without a fixed infrastructure and exhibit dynamic and stochastic behaviors. These conditions allow us to analyze them through properties like self-organization, adaptation and the use of the agent paradigm for managing and controlling the network. In this article, socially inspired computing is used to implement a register module for software agents through the concepts of institution and robustness; our goal is to build a software application that keeps working under the unexpected operating conditions of the ad hoc networks. IEEE FIPA protocols were used to design the agents and the communication channels. Three scenarios were developed, and failures in the systems were simulated. The results show the robustness of the module; it can recover itself from failures and allow the services to keep operating in acceptable conditions.**

Keywords*—Ad hoc networks, FIPA, robustness, self-organization, socially inspired computing.

I. Introducción

Las redes ad hoc son sistemas computacionales descentralizados, formados por un conjunto de nodos que se comunican entre sí a través de enlaces inalámbricos y que no dependen de una infraestructura preexistente para funcionar [1], [2]. Su naturaleza auto-organizante facilita su despliegue en ambientes de operación adversos donde no es posible contar con una infraestructura preestablecida para el funcionamiento de la red. Esta propiedad se convierte en una ventaja frente a las redes tradicionales, ya que permite adaptar la estructura de la red a diferentes contextos de operación. Adicionalmente, este comportamiento auto-organizante permite estudiar este tipo de redes como sistemas complejos, ya que es difícil predecir su comportamiento únicamente a través del análisis de sus componentes (usuarios, protocolos, servicios, nodos) [3]. Para enfrentar estos desafíos, las ciencias de la complejidad y el paradigma de agentes permiten incluir propiedades como la adaptación, la robustez y la anticipación, lo que abre la posibilidad de modificar el comportamiento de un sistema en situaciones inesperadas, sin tener que realizar una nueva configuración [4].

Por otro lado, las redes ad hoc permiten construir sistemas escalables formados por un gran número de subsistemas, que buscan ofrecer servicios de cómputo a través de la coordinación y colaboración de sus componentes. Entre sus aplicaciones más destacadas se encuentran las redes de sensores, las redes vehiculares, las nubes móviles y la agricultura de precisión. Sin embargo, a pesar de las ventajas ofrecidas por este tipo de sistemas, construir redes ad hoc con una alta capacidad de despliegue presenta grandes desafíos en términos de disponibilidad, confiabilidad y gestión de la red, debido a su naturaleza dinámica y a la ausencia de control e información centralizada.

Con el propósito de abordar estos retos, este trabajo busca utilizar el concepto de robustez para generar un servicio de registro para agentes de software que, gracias a su autonomía y capacidad de respuesta, esté en la capacidad de funcionar en condiciones de operación dinámicas y recuperarse de fallas en tiempo de ejecución. Para esto, se combinarán los principios de diseño del estándar FIPA (Foundation for Intelligent Physical Agents) y el concepto de institución. El objetivo es mantener actualizada la información de los agentes de software disponibles en la red ad hoc encargados de ofrecer servicios y realizar tareas de gestión ante cambios inesperados en el ambiente de operación.

El resto del artículo se organiza en cinco secciones. En la sección II se presentan las generalidades de las redes ad hoc y el paradigma de agentes. La sección III presenta la necesidad de una institución de registro y la metodología de computación social inspirada. El modelo propuesto se presenta en la sección IV. En la sección V se presentan los escenarios de prueba sobre la red ad hoc y la validación de la propiedad de robustez. La sección VI presenta la discusión y conclusiones.

II. Redes ad hoc y paradigma de agentes

A. Redes 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 según las necesidades del sistema. Como resultado, la topología de la red puede

* M.Sc (c) in telecommunications engineering. National University of Colombia
† Ph.D (c) in Systems and Computing Engineering. National University of Colombia
‡ Associate Professor. Computing Systems and Industrial Engineering Department

978-1-5386-8131-2/18/$31.00 ©2018 IEEE

cambiar rápida y de manera impredecible, modificando constantemente las condiciones de operación [5]. De igual forma, los servicios funcionan de forma distribuida, lo que hace necesario realizar procesos de coordinación y cooperación para habilitar la comunicación y las funcionalidades de la red [6]. Bajo estas condiciones, los componentes del sistema deben operar de manera autónoma, por esto su funcionamiento es más complejo que en las redes tradicionales.

Como resultado, es recomendable tener en cuenta las siguientes características al momento de diseñar y construir soluciones para redes ad hoc [7]:

- La redes ad hoc funcionan como sistemas auto-organizantes: cada nodo opera independiente y no existen mecanismos de control o información centralizados.
- Los nodos deben realizar procesos de cooperación y coordinación 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 debido a que la ruta de enlace end-to-end es usada por múltiples nodos.
- Los dispositivos son móviles y obtienen su energía de baterías, las cuales son un recurso limitado y deben utilizarse de manera eficiente [8].
- La topología de la red es dinámica. Los nodos pueden cambiar su posición de manera frecuente, lo que dificulta el enrutamiento.

B. Sistemas auto-organizantes

"La auto-organización es la forma de observar los sistemas, no una clase absoluta de sistema" [3].

Antes de hablar de auto-organización, es necesario definir qué es un sistema complejo. En [9], se define como la suma de los componentes y las relaciones de un sistema que mediante sus interacciones produce comportamientos emergentes. Cabe resaltar que el resultado de estas interacciones no sigue un patrón definido y no son fáciles de predecir. Como consecuencia, se deben utilizar nuevos métodos para analizar estos comportamientos, ya que no es posible hacerlo a través de métodos tradicionales. Generalmente, al trabajar con sistemas auto-organizantes artificiales, como las redes ad hoc, se utilizan combinaciones de los siguientes enfoques para mantener la integridad del sistema en un ambiente de operación cambiante e inesperado [3]:

- **Adaptación:** El sistema modifica su estructura o comportamiento para enfrentar el cambio. Le permite al sistema modificarse así mismo para "ajustarse" mejor a las condiciones del ambiente.
- **Anticipación:** El sistema anticipa un cambio al cual debe hacer frente y se ajusta según sea necesario. Esto prepara al sistema para enfrentar los cambios antes de que ocurran, adaptando el sistema sin sufrir perturbaciones.
- **Robustez:** Un sistema es robusto si continúa funcionando frente a una perturbación. Le permite al sistema resistir perturbaciones sin perder su función o propósito.

C. Agentes de software

Un agente es un sistema computacional que se encuentra situado en un ambiente, y que es capaz de modificarlo para cumplir sus objetivos de diseño [10]. De igual forma, los agentes deben ser construidos con las características y atributos que le permitan realizar las acciones para las cuales fue programado. Para esto, se espera que un agente sea una pieza de software con las siguientes propiedades [11]:

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

Adicional, se espera que un agente exhiba comportamientos racionales, es decir, no debe tener objetivos de diseño que posteriormente pueden entrar en conflicto o disputa. Para esto se han realizado estudios sobre cómo debe ser el diseño de un agente artificial, los cuales proporcionan explicaciones ontológicas, mecánicas y operacionales [12] [13] [14] .

D. 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 busca generar un esquema de computación inspirado en comportamientos biológicos y sociales para redes de comunicaciones auto-organizantes [15]. El objetivo de este sistema es contar con los modelos formales y computacionales que permitan a los miembros de una red ad hoc compartir recursos y servicios de forma dinámica en ausencia de control centralizado. En la Figura 1 se pueden observar los componentes del modelo propuesto, el cual tiene como sistema base las redes ad hoc con todas las condiciones mencionadas anteriormente.

En la primera y segunda capa del sistema TLÖN se hace referencia a la red física (red ad hoc) y la virtualización de esta red respectivamente. En la tercera capa se encuentra alojado un sistema multi-agente. Esta capa tiene como propósito facilitar las tareas de gestión de la red y ofrecer servicios de cómputo a través de la cooperación y coordinación de agentes de software. Este trabajo se enmarca en esa capa, y tiene como objetivo construir un módulo de registro con propiedades de robustez que permita facilitar la identificación y la gestión de los agentes disponibles en el sistema durante la operación de la red.

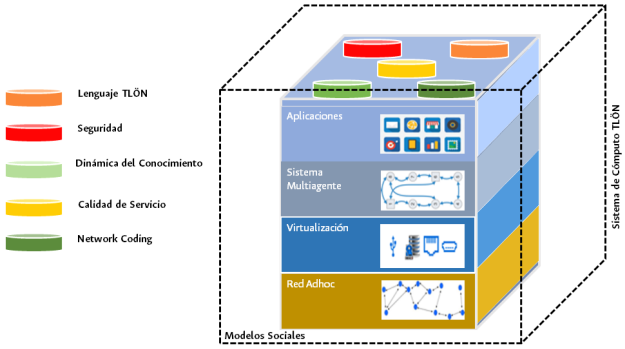

Figure 1. Componentes del Proyecto TLÖN [15].

III. Necesidad de una institución de registro

Dado que este trabajo tiene como base modelos computacionales social inspirados, se utilizará el concepto de institución para construir el módulo de registro. Como se argumenta en [16], los agentes pueden considerarse como ciudadanos de primera clase en las aplicaciones de sistemas multi-agentes. De esta forma, el módulo de registro desempeñará un papel análogo al que realiza la Registraduría Nacional. De ahí su relevancia en el sistema, ya que todos los agentes deberán registrarse para ser reconocidos en el ambiente, ofrecer servicios y estar en la capacidad de realizar cualquier proceso de cooperación.

Por otro lado, los modelos computacionales basados en el concepto de institución han demostrado ser un mecanismo poderoso para lograr interacciones más efectivas, estructuradas, coordinadas y eficientes en sistemas multi-agente [17]. Estos modelos permiten crear organizaciones sociales dinámicas y regular y controlar el comportamiento del sistema. Es importante mencionar que un agente por sí solo no puede representar una organización social. Sin embargo, la interacción de un conjunto de agentes puede formar comunidades (también conocidas como coaliciones) que les permita trabajar de manera conjunta para alcanzar sus objetivos de diseño. Como consecuencia, es de vital importancia contar con información actualizada de los agentes disponibles en el sistema para ejecutar tareas de gestión y ofrecer servicios en la red.

De igual forma, en esta investigación se tuvieron en cuenta las recomendaciones de estándar FIPA de la IEEE para la construcción de sistemas multi-agentes. En este estándar, la institución de registro hace referencia al AMS (Agent Management System), el cual es un componte obligatorio del ambiente. Así, el módulo de registro ejerce un control de supervisor sobre el acceso y el uso del ambiente. En este sentido, solo existirá un módulo de registro por ambiente, donde habrá un directorio con las direcciones de transporte (direcciones lógicas, comunidad, servicios ofrecidos, etc.) para todos los agentes registrados en el sistema [18].

Dado que el módulo de registro es un factor de vital importancia para el sistema, la contribución de este trabajo es lograr que dicho módulo exhiba propiedades de robustez. Varias investigaciones evidencian que esta propiedad es vital en la implementación de sistemas multi-agente. Así lo menciona Russel en [19], donde señala que en la medida en que un sistema opera de manera autónoma, es necesario que se comporte de manera robusta.

IV. Modelo propuesto

Tomando como base el trabajo de Ostrom sobre instituciones auto-organizantes [20], es posible definir una institución como un conjunto de reglas al interior de un contexto social, que permiten determinar quién es elegible para tomar decisiones en algún contexto, qué acciones están permitidas o limitadas, qué procedimientos se deben seguir, qué información debe o no se debe proporcionar, y qué pagos se asignarán a los agentes de acuerdo con sus acciones. A partir de esto, y siguiendo las recomendaciones del estándar FIPA, se determinaron las siguientes normas para el módulo de registro:

A. Normas Generales de la Institución de Registro

1) Solo existirá una única institución de registro para agentes y servicios por ambiente.
2) La institución debe contar como mínimo con un agente con tareas de gestión.
3) La institución debe proveer la información de los agentes disponibles en el sistema cuando sea requerido.
4) La institución tiene la autoridad para aceptar o denegar el registro de un agente en el sistema.
5) Las funciones de la institución le permiten ejecutar las siguientes acciones sobre sí mismo: registrar, desregistrar, modificar y buscar.
6) La información almacenada en el archivo de registro debe permanecer a lo largo del tiempo. No se permitirá la eliminación de registros bajo ningún criterio.

B. Administración de Datos

Los datos en el módulo de registro serán almacenados en un diccionario de datos. La información que contendrá cada registro está basada en las recomendaciones del estándar FIPA y se compone de lo siguientes atributos:

- **UID** (Unique Identifier): identidad propia de cada agente proporcionada por un identificador, en este caso la institución de registro. Este debe ser único en todo el sistema.
- **Name**: nombre con el cual se va a conocer el agente en el sistema.
- **Community**: comunidad a la cual va a pertenecer.
- **Function**: descripción del trabajo o tareas que puede desempeñar un agente para cumplir sus objetivos de diseño.
- **State**: estado en el cual se encuentra el agente. Este debe permanecer actualizado en línea.
- **Location**: cada agente debe ser capaz de ser ubicado en el sistema. Este parámetro se almacenará y actualizará constantemente.

Una vez determinadas las normas y el proceso de registro, se definen los agentes que van a cumplir con las tareas asociadas al módulo de registro.

1) Agente Supervisor: este agente es el encargado de construir un mensaje y enviarlo directamente a otros miembros del sistema con el propósito de corroborar que la información registrada sea correcta y se encuentre actualizada.

2) Agente Coordinador: verificará que el supervisor se esté ejecutando 24/7. Mientras el sistema esté activo debe existir un supervisor que garantice la calidad de los datos disponibles en el módulo.

3) Agente de Soporte: es el encargado de velar por la integridad del diccionario de datos. Su propósito es verificar que la información sea respaldada y en caso de existir una alteración, el agente deberá restaurar el diccionario minimizando la afectación al sistema.

V. Escenarios de prueba y validación

Para la ejecución de las pruebas se implementó la red ad hoc que se observa en la Figura 2. En este caso, cada nodo representa una instancia del sistema TLÖN y, a su vez, cada instancia del sistema tiene su propio módulo de registro. El principal objetivo de este escenario es comunicar toda la red y verificar la disponibilidad de los agentes en el sistema.

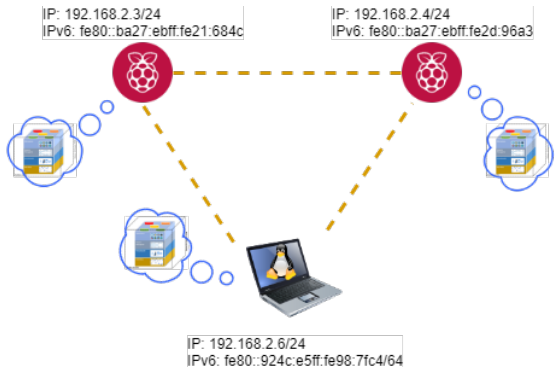

Figure 2. Diagrama de red para los escenarios de prueba

El enrutamiento entre los nodos se realizó a través del protocolo B.A.T.M.A.N (Better Approach To Mobile Adhoc Networking), el cual se encuentra disponible en el Kernel de Linux. Este protocolo permite construir redes ad hoc de una manera flexible garantizando un enrutamiento altamente adaptable, sin bucles y con un bajo costo de procesamiento [21]. Adicional, para la comunicación entre agentes se utilizó el estándar XMPP para garantizar el envío de mensajes [22]. Este proceso solo requiere que los agentes se registren en el servidor, así una vez se realiza el proceso de registro, se habilita la inscripción para la comunicación XMPP con el mismo UID asignado por la institución.

Ahora bien, una vez establecida la comunicación entre las instancias del sistema TLÖN, se diseñaron tres escenarios de prueba donde se simulan fallas del módulo de registro y se verifica su nivel de robustez. Para esto, en los escenarios 1 y 2 se crea un servicio que verifica el número de agentes que responden al mensaje, con respecto al número de agentes que tienen el estado "Activos" en el diccionario (ver Ecuación 1). Este cálculo se realiza para cada ciclo del programa. De esta forma, se define un ciclo como el tiempo en el cual se obtienen los diccionarios de los tres ambientes, se valida cuáles están activos, se envían los mensajes y se consolidan los datos de los agentes que respondieron.

$$rate = \frac{Agentes_{contestaron}}{Agentes_{total\_activos}} \quad (1)$$

A. Escenario 1: Auditoria de agentes

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 se actualice con el estado "Inactivo", correspondiente al nuevo estado del agente. En este caso, el actor principal es el agente supervisor: él es el encargado de validar que toda la información que se encuentre en el diccionario sea correcta. Además debe detectar los estados, ubicaciones o descripciones que han cambiado y actualizar los registros. Se espera que si un agente "desaparece" de manera inesperada, el módulo esté en la capacidad de actualizar su estado en el menor tiempo posible.

Table I. Parámetros escenario 1

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

El escenario de prueba se configura según los parámetros presentados en la Tabla I. Se debe tener en cuenta que la falla de los agentes se hace de manera aleatoria en diferentes ciclos del programa y el total de agentes corresponde a todos los que se encuentran registrados, independientemente de su estado.

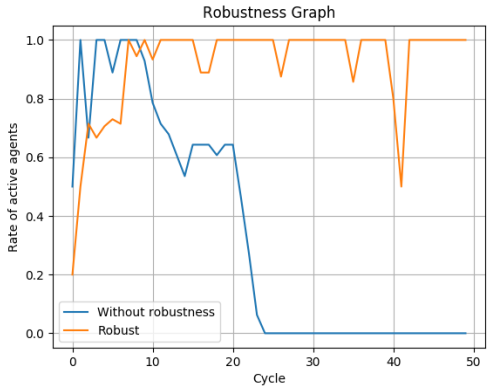

Figure 3. Resultados escenario 1

En la Figura 3 se comparan los resultados del escenario de prueba cuando el sistema es robusto y cuando no lo es. En el segundo caso, no existe ningún agente supervisor, por lo tanto la integridad de la información va disminuyendo gradualmente. Por otro lado, al final de la prueba, la tasa de agentes queda en cero ya que todos terminaron sus acciones o finalizaron de manera súbita. En este caso, a pesar que no hay ningún agente en el sistema, el módulo

de registro no actualiza la información y reporta agentes activos, de ahí, la importancia del agente supervisor. Por el contrario, cuando el sistema es robusto, ya sea porque el agente finalizó correctamente o porque el supervisor ha actualizado los estados, el módulo reporta los estados reales durante la mayor cantidad de ciclos.

B. Escenario 2: Seguimiento a funcionarios

En este escenario, la falla se simulará sobre el agente supervisor, encargado de mantener actualizado el módulo de registro. Sin embargo, si el agente supervisor deja de funcionar de manera inesperada, debe entrar en acción un segundo agente que valide los datos y siga cumpliendo con sus funciones. El indicador de éxito es similar al presentado en el escenario 1 (agentes que terminan inesperadamente), incluyendo a los agentes supervisores en los que presentan fallas. Se espera que cuando se consulte el estado del sistema multi-agente, la información retornada sea acorde a su estado real.

El agente supervisor, al ser una aplicación de software, puede verse 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 no ser así, este agente coordinador lo inicia, lo que significa que un nuevo agente supervisor es creado y comienza a cumplir sus funciones.

Table II. Parámetros 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

En la Tabla II se presentan con detalle los componentes que hicieron parte de este escenario y se especifican los agentes de cada ambiente, donde Tlön-i, Tlön-ii y Tlön-iii corresponden a un sistema multiagente que se muestra en la Figura 2, ya que el total varía según el número de veces que haya fallado el supervisor.

Los resultados del experimento se presentan en la Figura 4. Es posible observar que 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 (hasta ese punto permaneció activo el agente supervisor). Desde ese ciclo, si el agente supervisor llega a presentar un error la integridad de la información, empieza a decaer paulatinamente. Por el contrario, cuando se tiene un agente coordinador, siempre se busca mantener el estado del módulo dentro de unos límites aceptables y mantener así la definición de sistema robusto tal como se precisa en [23]. Adicional, es posible observar que las perturbaciones son mayores y toma más tiempo recuperarse de ellas, como resultado del tiempo que tarda el agente supervisor en crearse de nuevo y e iniciar el proceso desde cero.

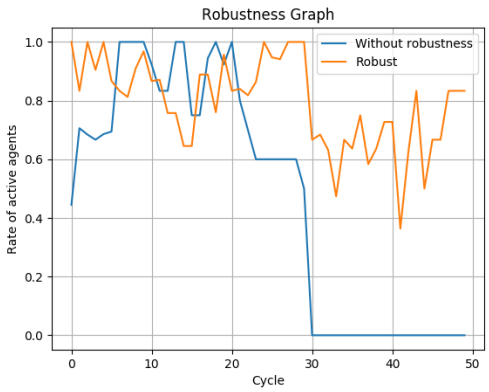

Figure 4. Resultados escenario 2

C. Escenario 3: Monitoreo de respaldo

Una de las formas de hacer el módulo de registro robusto es almacenar en disco los datos que se encuentran en memoria. Así, en caso de que se reinicie la instancia en donde se están guardando los datos, o que el diccionario sufra alguna perturbación, la información no se perderá por completo y se tendrá un punto de recuperación. En ese sentido, el tercer escenario va a simular una perdida de datos y mostrará cómo la información es restaurada sin que los agentes del sistema perciban fallas en el servicio de registro. Ahora bien, aunque el rendimiento o los tiempos de consulta se vean afectados, el éxito de este escenario radica en que cuando se consulte el diccionario, la información reportada sea correcta.

De esta forma, cuando el agente supervisor valida que la información es correcta, genera un respaldo del diccionario, el cual será la base para que el agente de soporte compare los datos. Si este agente de soporte llega a detectar que el diccionario actual tiene menos datos que el de respaldo (significa que parte de la información fue suprimida), restaura este último respaldo junto con la información que tenga actualmente para mantener la integridad de los datos. En caso contrario, si el respaldo tienen menos información, actualiza el segundo para tener un punto de respaldo confiable.

Table III. Parámetros 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.

Para este este escenario, es importante que la información se respalde por el ambiente, ya que sólo pertenece a su institución y es independiente de las demás instancias del sistema TLÖN. En la Tabla III se muestran los parámetros de configuración enfocados a un solo ambiente.

Los resultados del experimento se presentan en la Figura 5. En este caso, sólo existen dos estados: 1 si el respaldo y el diccionario actual son exactamente iguales y 0 si no lo son. Se decidió hacerlo de esta forma, ya que cuando estos dos objetos son diferentes significa que la información actual

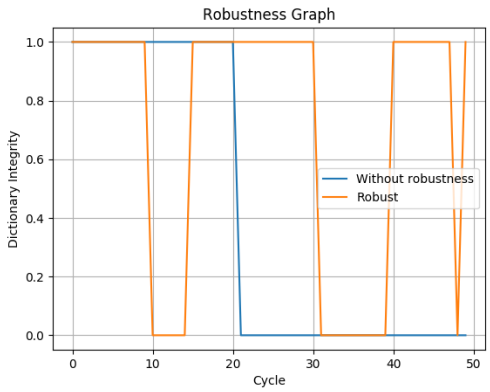

Figure 5. Resultados escenario 3

no ha sido respaldada correctamente o que el diccionario ha sufrido algún tipo de pérdida. Cualquiera de los dos escenarios representa una falla del servicio.

VI. Conclusiones

Una de las características principales de un sistema social inspirado es que los agentes estén en la capacidad de interactuar entre sí. Para esto, primero deben conocer quién se encuentra en su ambiente y contar con un mecanismo de comunicación que les permita intercambiar mensajes. Una posible solución es realizar la gestión de un directorio que cuente con toda la información de los agentes disponibles en el sistema. De allí la motivación de construir un módulo de registro con propiedades de robustez.

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 afecta el tiempo en que se puede recuperar de un fallo y, por lo tanto, la capacidad para operar un servicio en la red. En la ejecución de los escenarios de prueba se observa este factor, ya que la respuesta a las solicitudes de los agentes presentó un retraso 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.

Por su parte, el concepto de institución permite regular el funcionamiento y comportamiento de un grupo de agentes auto-organizantes por medio de un conjunto de reglas. Dichas reglas 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.

References

[1] F. Dressler, "Self-Organization in Ad Hoc Networks: Overview and Classification," University of Erlangen, Erlangen, Germany, Tech. Rep., 2006.

[2] J. P. Ospina and J. E. Ortiz, "Estimation of a growth factor to achieve scalable ad hoc networks," *Ingeniería y Universidad*, vol. 21, no. 1, pp. 49–70, 2017.

[3] C. Gershenson, *Design and Control of Self-organizing Systems*, copit arxi ed. Mexico: CopIt ArXives, 2007.

[4] T. Qiu, N. Chen, K. Li, D. Qiao, and Z. Fu, "Heterogeneous ad hoc networks: Architectures, advances and challenges," *Ad Hoc Networks*, vol. 55, pp. 143–152, 2017.

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

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

[7] J. Loo, J. Lloret Mauri, and J. H. Ortiz, *Mobile ad hoc networks : current status and future trends*. CRC Press, 2016.

[8] L. Feeney and M. Nilsson, "Investigating the energy consumption of a wireless network interface in an ad hoc networking environment," in *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, pp. 1548–1557.

[9] Y. Bar-Yam, "General Features of Complex Systems," *Encyclopedia of Life Support Systems (EOLSS UNESCO Publishers, Oxford, UK, 2002)*, vol. I, 2002.

[10] M. J. Wooldridge, *An introduction to multiagent systems*. John Wiley & Sons, 2002.

[11] L. Padghman and M. Winikoff, *Developing Intelligent Agent Systems*. John Wiley & Sons, 2004.

[12] D. G. Johnson and M. Noorman, "Principles for the future development of artificial agents," in *2014 IEEE International Symposium on Ethics in Science, Technology and Engineering*. IEEE, may 2014, pp. 1–3.

[13] J. N. Karigiannis and C. S. Tzafestas, "Robustness and generalization of model-free learning for robot kinematic control using a nested-hierarchical multi-agent topology," in *2012 4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*. IEEE, jun 2012, pp. 1140–1147.

[14] N. T. M. Khue, "Developing an Intelligent Multi-Agent System based on JADE to solve problems automatically," in *Systems and Informatics (ICSAI), 2012 International Conference on*, may 2012, pp. 684–690.

[15] "Tlön-grupo de investigación en redes de telecomunicaciones dinámicas & lenguajes de programación distribuidos." [Online]. Available: http://www.tlon.unal.edu.co/

[16] G. Sartor, "Why Agents Comply with Norms, and Why They Should." Springer, Boston, MA, 2001, pp. 19–43.

[17] O. Cliffe, M. De Vos, and J. Padget, "Embedding Landmarks and Scenes in a Computational Model of Institutions," 2008.

[18] Foundation For Iintelligent Physical Agents, "FIPA Agent Management Specification," 2004.

[19] S. Russell, D. Dewey, and M. Tegmark, "Research Priorities for Robust and Beneficial Artificial Intelligence," *AI Magazine*, vol. 36, no. 4, p. 105, dec 2015.

[20] E. Ostrom, *Governing the commons : the evolution of institutions for collective action*. Cambridge University Press, 1990.

[21] C. Aichele, S. Wunderlich, A. Neumann, and M. Lindner, "Better Approach To Mobile Ad-hoc Networking (B.A.T.M.A.N.)," 2008.

[22] P. Saint-Andre, "Jabber Component Protocol," jan 2012.

[23] Santa Fe Institute, "Complexity Explorer."