Diseño Socio-inspirado de la Máquina Virtual Distribuida (Örbis Virtüalis Maqüinus ÖVM) para la red Ad-Hoc TLÖN

Jhon Alexander López
Universidad Nacional de Colombia
Carrera 30 No. 45-03
Edificio Sindú 314, Oficina 110
(+57 1) 316 5000 ext:12206
jalopezfa@unal.edu.co

Joaquín Fernando Sánchez
Universidad Nacional de Colombia
Carrera 30 No. 45-03
Edificio 453, Oficina 107
(+57 1) 316 5000 ext:14075
jofsanchezci@unal.edu.co

Jorge Eduardo Ortiz
Universidad Nacional de Colombia
Carrera 30 No. 45-03
Edificio 453, Oficina 107
(+57 1) 316 5000 ext:14075
jeortizt@unal.edu.co

RESUMEN

Este artículo introduce el diseño de una máquina virtual distribuida basado en un modelo social para la implementación de una red ad-hoc denominada TLÖN. En este modelo los nodos representan planetas gobernados por determinado sistema operativo dónde la máquina virtual es la embajada del Estado (gobierno y control de la red ad-hoc) y a través de ésta puede extender sus recursos y emplearlos en el cumplimiento de los objetivos generales de la red.

De esta manera los conceptos extraídos de las estructuras sociales son traducidos en términos computacionales y aplicados en el diseño y arquitectura de todos los sistemas y subsistemas que conforman la red. En contraste con los modelos bio inspirados los modelos sociales han sido poco utilizados ya sea por su complejidad o la dificultad para abstraer factores sociales en aspectos técnicos y viceversa, no obstante a través de la investigación queremos descubrir sus propiedades y posibles aplicaciones en computación distribuida, sistemas inteligentes entre otros campos de la computación.

Palabras Clave

Máquina virtual distribuida, socio-inspiración, sistemas distribuidos, redes ad-hoc, lenguajes de programación distribuidos, sistemas inteligentes, sistemas multiagente.

ABSTRACT

This paper introduces the design of a distributed virtual machine what is based on a social model for the implementation of an ad-hoc network called TLÖN. In this model the nodes represents planets what are governed by an operative system and the virtual machine is the State embassy (The State is the government and control of the ad-hoc network) and through it the State may extend their resources and employ them to reach the network goals.

In this way the social concepts are translated to computational terms and applied in the design of every system and subsystem in the network. In opposition to bio-inspired models the social models have not been enough used yet because there is a complexity or difficulty to abstract social factors in technical aspects and vice versa. However with this research we want to find their properties and their future uses in distributed computing, intelligent systems and other topics of computing.

Categories and Subject Descriptors

C.2.4 [**Distributed Systems**]: Distributed Applications

D.3.2 [**Language Classifications]**: Design Languages

I.2.11 [**Distributed Artificial Intelligence]**: Coherence and Coordination, Language and structures, Multiagent Systems
D.3.3 [**Programming Languages**]: Language Constructs and Features – *abstract data types, polymorphism, control structures.*

General Terms

Algorithms, Design, Experimentation, Languages, Theory.

Keywords

Distributed virtual machine, socio-inspiration, distributed systems, ad-hoc networks, distributed programming languages, intelligent systems, multiagent systems.

INTRODUCCIÓN

La computación distribuida es una de las áreas que ha despertado mayor interés durante la última década[1], esto era de esperarse teniendo en cuenta que la mayoría de los sistemas de información se ha venido desarrollando de manera distribuida para responder a las necesidades cada vez más exigentes de la economía global y la evolución continua de las tecnologías de la comunicación[2], adicionalmente todos los sistemas son diseñados para ser accedidos a través de internet y las exigencias de calidad son cada vez mayores, así la seguridad, la escalabilidad, la disponibilidad, el desempeño y demás características son indispensables e implementadas de manera distribuida.

Las redes ad-hoc son un buen ejemplo de la complejidad de los sistemas distribuidos, su variedad, su versatilidad y de la infinidad de aplicaciones para las que se puede implementar este tipo de tecnología [3]. De ese modo la tecnología móvil y la tendencia de las redes de computadores denominada Internet de las cosas (IoT) e Internet de Todo (IoE) cada vez se acercan más a la estructura de las redes ad-hoc pero con algunas salvedades, por ejemplo aún existen elementos fijos de la red que posibilitan la comunicación entre los nodos.

Por otro lado los lenguajes de programación distribuidos son un paradigma reciente en técnicas de programación para sistemas distribuidos, existen dos formas de implementación conocidos: la primera es la programación a través de un Sistema Operativo Distribuido [4] y la segunda es la programación a través de un

Middleware[5]. Ambas estrategias son válidas y eficientes dependiendo del contexto en el cual se les utilice, los sistemas operativos distribuidos por ejemplo son útiles si se tiene una infraestructura centralizada de computadores en un mismo espacio y el middleware es útil si se tiene una infraestructura descentralizada de computadores en diferentes espacios, pero el caso de las redes ad-hoc es mucho más complejo que eso porque supone que los nodos de la red no solo están dispersos en diferentes lugares sino que están continuamente en movimiento.

La máquina virtual distribuida ha sido una solución alternativa que funcionaría como un contrato entre los nodos que se afilian a la red, esta es una manera en la cual cualquier tipo de objeto que se quiera unir a la red acepta los términos y condiciones explicitas del contrato y al unirse se compromete a cumplir los deberes asignados.

Por último el diseño socio-inspirado ha surgido como una respuesta a las necesidades del proyecto TLÖN, teniendo en cuenta las características estocásticas que se presentan tanto en una red ad-hoc como en cada uno de sus nodos y la similitud con los mecanismos y roles que se ejecutan en los sistemas sociales. Así toda la programación será orientada por agentes, dónde cada uno cumple un rol y una labor que contribuye al desarrollo de todo el sistema al tiempo que se incluyen comportamientos inteligentes en cada uno de ellos para permitirles tomar decisiones, asumir responsabilidades y un cierto nivel de autonomía en sus labores, la mayoría de estas características hacen parte de nuestro objeto de investigación por lo que hasta el momento sólo contamos con los desarrollos generales de Inteligencia Artificial realizados hasta el momento y la teoría de agentes[6] para fundamentar nuestro proyecto de investigación. Por supuesto que el aspecto social de nuestra metodología es una extensión de la teoría de agentes pero desde el punto de vista de comunidades de agentes.

1. METODOLOGÍA

El proyecto TLÖN está presupuestado para desarrollarse en 10 años, todos los integrantes del grupo de investigación forman parte integral del proyecto donde todos se alinean con el objetivo general pero trabajando desde el área del conocimiento con el cual cada uno es más afín, de esa manera hay colaboración intergeneracional e interdisciplinaria en la dinámica de trabajo.

Existe un curso en la Facultad de Ingeniería dictado por el profesor del grupo denominado Lenguajes de Programación, durante el desarrollo del curso se aprenden los principales aspectos que se deben contemplar en el diseño de un compilador y el proyecto de clase es diseñar un compilador e implementar alguna de las partes, como el analizador léxico, el analizador sintáctico o el analizador semántico, los requerimientos consideran las características de las redes ad-hoc y un concepto fundamental dentro del proyecto TLÖN que es lo que denominamos agentes dispersos.

Los agentes dispersos son entidades de software que a diferencia de los agentes comunes, tienen la propiedad de estar segmentados por partes y distribuidos ya sea por segmentos o por unidades en los diferentes nodos de la red, de esta manera puede haber un agente por partes distribuido en la red o un agente completo dentro de un nodo pero copias de este en otros nodos. Estos agentes tienen como propósito encapsular las propiedades de la red y alcanzar sus objetivos, por ejemplo la administración de energía de la red es hecha por uno de los agentes, la organización jerárquica de la red es controlada por otro agente, el rol que desempeña cada nodo dentro de la red es controlado por otro agente y así muchas de las propiedades que discutiremos más adelante.

La forma inicial de la máquina virtual era simplemente un compilador, pero a medida que se han estudiado los resultados de cada semestre y las discusiones entre los integrantes del grupo de investigación se ha determinado que el mejor camino para alcanzar los objetivos del proyecto y que la red funcione correctamente bajos las condiciones planteadas es hacerlo a través de una máquina virtual[7].

Después de planteado el problema se ha invitado a estudiantes de diferentes semestres y programas académicos para trabajar en el tema como tesis o trabajo de grado, durante un año se ha recopilado la información de los requerimientos basados en el proyecto general y en las ideas generadas a través de los espacios académicos descritos previamente para realizar el respectivo diseño.

El proceso de diseño ha sido retroalimentado continuamente en un ejercicio académico, teórico y de discusión para encontrar una metodología de diseño y desarrollo que se ajuste a las necesidades del grupo y del proyecto, por tanto consideramos algunos elementos de arquitectura de software para describir la máquina virtual y algunas herramientas visuales para comunicar las ideas durante todo el proceso.

Inicialmente se consideró utilizar Python como lenguaje de programación del compilador considerando su versatilidad y facilidad de uso y aprendizaje, pero los cambios en el diseño de un compilador convencional a una máquina virtual nos ha llevado a contemplar la posibilidad de usar java en la programación e inclusive utilizar la JVM como software libre para no desarrollar nuestra máquina desde cero sino aprovechar sus características y ahorrar tiempo, no obstante las propiedades de la red, las diferencias entre ambos lenguajes de programación (java y tlonés), la arquitectura de software del proyecto general y de la propia máquina virtual nos han forzado al diseño e implementación de la máquina virtual desde cero.

De esa manera para el diseño de la ÖVM se emplea la metodología SADT (Structural Design and Desing Technique)[8], donde el proyecto general hace parte del primer nivel de arquitectura y la máquina virtual se encuentra en el tercero.

2. DISEÑO DE LA ÖVM

El bosquejo inicial de la maquina virtual se ilustra en la figura 1, allí la función que tiene la máquina es una interfaz de comunicación con el lenguaje de programación y el sistema operativo de la máquina, también se contempla desde el diseño la necesidad de darle a la máquina algunos permisos para interactuar con los recursos físicos, esto con el propósito de optimizar algunas funciones, de igual manera se han contemplado los riesgos que supone hacer este proceso, por ello la máquina virtual tendrá un componente de seguridad que se encargará de validar la autenticidad del emisor de las

instrucciones para evitar que agentes externos asuman el control de un nodo y sucesivamente el control de la red.

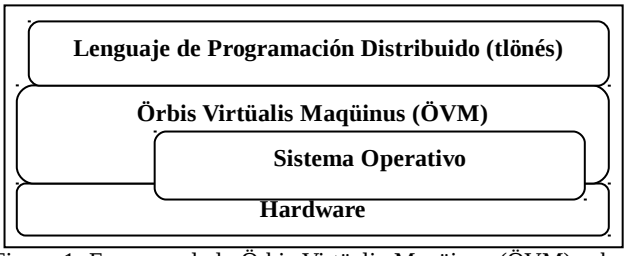

Figura 1. Esquema de la Örbis Virtüalis Maqüinus (ÖVM) sobre un solo nodo.

La figura 2 es un clúster formado por un nodo central y una configuración básica de 6 nodos agrupados a su alrededor, todos los nodos tienen la misma estructura de la figura 1, pero en su conjunto forman la ÖVM Distribuida.

Esta estructura en forma de clúster representa la relación entre el nodo central y los demás para formar un sistema distribuido, pero la comunicación puede ser hecha entre los nodos punto a punto sin pasar por el nodo central.

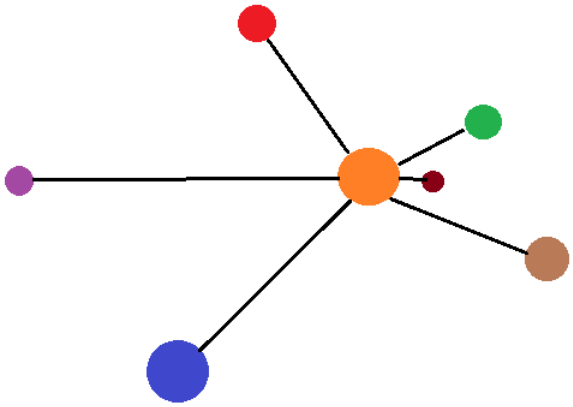

Figura 2. Esquema distribuido de la Örbis Virtüalis Maqüinus (ÖVM).

Esta configuración está inspirada en la forma de nuestro sistema solar, como hemos abstraído la noción de TLÖN como un mundo y extendido al nivel de un universo, la ÖVM es un tipo de unidad dentro de ese gran sistema puesto que un nodo aislado no tiene sentido de existencia, como no lo tendría un planeta si no orbitara alrededor de una estrella.

Los colores y variedades de tamaño representan la diversidad de las características tanto de software como de hardware respectivamente que podría tener cada nodo, esta cualidad heterogénea de nuestro sistema distribuida lo distingue de los clúster tradicionales en los que los nodos interconectados poseen características homogéneas.

Dentro del diseño de la máquina virtual determinamos algunos subsistemas fundamentales en los que las funciones serán realizadas por agentes como se detalla en la Tabla 1.

TABLA I

Subsistemas de la Máquina Virtual (ÖVM)

Subsistema	Función	Entradas	Salidas
Proceso Arquitectónico	Escanear las características de cada nodo y segmentar los recursos (Disco, memoria, procesador, etc) para las actividades de la ÖVM y de la red TLÖN. Agentes: Arquitectos	Inventario de Recursos	Arquitectura de Recursos
Proceso de Administración	Analizar las actividades y los recursos disponibles de cada nodo y asignar las actividades a dichos recursos con base en las reglas establecidas por el proceso Arquitectónico. Agentes: Administradores	Arquitectura de Recursos Tareas Agentes	Orden de Trabajo Solicitud de Recursos Externos (Cuando el nodo no cuenta con los recursos se buscan en otro u otros nodos)
Proceso de Seguridad	Escanear todos los programas que llegan a la máquina virtual para validar su origen y función, el propósito es evitar intrusiones y software malicioso en la ejecución de la máquina virtual. Agentes: Guardianes	Programa	Programa revisado
Proceso de Gobierno	Comunicarse con el nivel central de gobierno y controlar los agentes de cada nodo. Agentes: Gobernadores	Agentes Recursos Órdenes de Gobierno	Acuerdos Reglas Resoluciones
Cuna de agentes	Crear los agentes según los requerimientos de la administración.	Requerimien-tos	Agentes Registro de nacimientos
Cementerio de agentes	Los agentes viven en la memoria, por tanto cuando han cumplido su función son removidos y su registro se guarda en el cementerio.	Agentes	Registro de muertes
Proceso de Compila-ción	El programa se va traduciendo de bycode a binario en tiempo de ejecución. Agentes: Compiladores	Bycode	Binario

Hasta el momento solo se han definido los procesos y tipos de agentes mencionados en la Tabla I, éste sistema social ha sido solamente definido al nivel de nodo, al nivel de la ÖVM distribuida se ha contemplado un nivel superior de gobierno denominado Federación, pero es necesario analizar los costos y beneficios de una jerarquía de control tan compleja o más compleja si se tiene en cuenta que la ÖVM distribuida es una unidad dentro de todo el sistema distribuido, ahora bien el ente de gobierno de más alto nivel es el Estado y para este no se han hecho las especificaciones de diseño respectivas.

3. PROBLEMAS Y TRABAJO FUTURO

Existen muchos tipos de sistemas sociales y desconocemos cual configuración podría ser más eficiente en el desarrollo de las tareas para la red, el proceso desarrollado contempla algunos elementos de psicología social como teoría de grupos y algunas consideraciones de sentido común sobre los sistemas sociales y su abstracción en los subsistemas de la máquina virtual ÖVM.

Los fenómenos sociales son el otro aspecto importante a considerar, si bien las ciencias sociales han avanzado mucho en su estudio la teoría social es generalmente deductiva opuesto a lo que sucede con las ingenierías y las ciencias exactas que son principalmente inductivas, esto supone un reto en la adaptación de aquellos conceptos que abarcan aspectos globales tales como democracia, política, economía y otros en conceptos más específicos como byte, memoria RAM, procesador, almacenamiento, entre otros.

Por otro lado las teorías sociales pueden diferir dependiendo del autor y la escuela de pensamiento que las incuba, mientras que los aspectos técnicos son esencialmente permanentes en sus fundamentos y variantes en su tecnología. Nuestro trabajo pretende integrar estos aspectos opuestos pero igualmente interesantes de ambas corrientes de pensamiento.

Nuestro ideal es implementar una red ad-hoc no solo fuertemente fundamentada para asegurar su vida útil en un futuro distante sino también darle una amplia utilidad en múltiples campos como por ejemplo situaciones catastróficas o de emergencia, aplicaciones espaciales, la adecuación de ciudades inteligentes, entre otros.

La máquina virtual es solo uno de los pasos que tenemos que dar para alcanzar nuestro objetivo, esperamos que nuestra intuición nos guie por el camino correcto en los conceptos que empleamos y las teorías que formulamos. Es una estrategia osada alejarnos del confort ofrecido por los fundamentos de la ingeniería y las ciencias de la computación para navegar en el océano del conocimiento, atravesando las tormentas y tempestades características de las ciencias sociales.

Requerimos ayuda de profesionales en esas áreas de estudio (sociología, psicología, antropología, geografía, etc) y de la participación de nuevos integrantes que aporten ideas y tiempo de trabajo para la conclusión exitosa del proyecto, la falta de capital humano hace más lento nuestro progreso y pone en riesgo los avances obtenidos hasta el momento.

Una vez implementada la máquina virtual y el compilador del lenguaje tlönés es necesario profundizar en otros aspectos del proyecto como la conformación de la Nación y el Estado, la revisión de la especificación del lenguaje, la compilación de la máquina virtual para diferentes plataforma de hardware, la implementación de las comunidades de agentes, etc.

En el futuro próximo (10 años) esperamos contar con algunas redes ad-hoc TLÖN dispersas geográficamente, completamente móviles y conectadas a internet. Una vez alcanzado este objetivo procederemos a profundizar en aspectos específicos y las posibles aplicaciones de este proyecto.

REFERENCIAS

[1] A.S. Tanenbaum, M.Van Steen. (2007) Distributed Systems: Principles and Paradigms. Pearson Education. Second Edition.

[2] Zárate, H., & Nuñez, J. (2014). Metodología para el diseño de redes de Comunicaciones de Emergencia. X Congreso Internacional de Electrónica y Tecnologías de Avanzada. Pamplona. Santander.

[3] Zárate, H., & Ortiz, J. (2013). Servicios en Telecomunicaciones de Emergencia. III Congreso Internacional de Instrumentacion Control y Telecomunicaciones. Tunja.

[4] Bal, H., Kaashoek, F., Tanenbaum, S. Orca: A Language For Parallel Programming of Distributed Systems. IEEE Transactions on Software Engineering. Vol. 18. No 3. March 1992

[5] Al-Jaroodi, J., Mohamed, N. Jiang, H. Swanson, D. Middleware Infrastructure for Parallel and Distributed Programming Models in Heterogeneous Systems. IEEE Transactions on Parallel and Distributed Systems. Vol. 14 No. 11. November 2003.

[6] Russell, S. J., & Norvig, P. (1996). Inteligencia Artificial: un enfoque moderno.

[7] Aho, A. V., Sethi, R., & Ullman, J. D. (1998). Compiladores: principios, técnicas y herramientas. Pearson Educación

[8] Ross, D. T. (1977). Structured analysis (SA): A language for communicating ideas. Software Engineering, IEEE Transactions on, (1), 16-34.

Bal, H., Kaashoek, M., & Tanenbaum, A. (1992). Orca: A language for parallel programming of distributed systems. IEEE Transactions on Software Engineering IIEEE Trans. Software Eng., 190-205.

Tanenbaum, A., & Steen, M. (2007). Distributed systems: Principles and paradigms.