ISSN 1870-4069

Uso de Sistemas Multiagente en Órdenes de Traba de Manufactura

José Antonio Gordillo Sosa

Universidad Tecnológica del Suroeste de Guanajuato,
México

antgor@antoniogordillo.com

Resumen. En este trabajo de investigación se presenta el proceso de desarrollo e implementación de un algoritmo de distribución de recursos aplicado a la programación de un sistema multi-agentes, en un entorno de fabricación de piezas cerámicas. La idea central de la misma gira en torno a la búsqueda del balance entre la distribución de órdenes de trabajo aplicando modelos de temperatura entre los agentes. Se desarrolló una simulación completa de la tarea anterior aplicando la herramienta de programación de agentes JADE.

Palabras clave: Multi-agentes, distribución dinámica, algoritmo de optimización, programación de órdenes de trabajo.

Application of Multi-Agent Systems in Manufacturing Work Orders

Abstract. This research paper presents the developing and implementation process of an algorithm applied to the scheduling of a multi-agent system in a ceramic manufacturing envirorment. The main focus revolves around the search for a balance in the distribution of work orders by applying temperature models among the agents. A complete simulation of the aforementioned task was developed using the JADE agent programming tool.

Keywords: Multi-agent, dynamic distribution, optimization algorithm, work order scheduling.

1. Introducción

Una de las actividades más importantes y al mismo tiempo, la más compleja a ser desarrollada dentro de los sistemas de manufactura actuales, es la asignación de tareas. Se requiere que sea lo suficientemente dinámica y adaptable como para enfrentar fallas mecánicas, órdenes emergentes y un sinnúmero de eventualidades más. Actualmente,

la industria está experimentando cambios muy importantes en lo que se refiere a la aplicación de elementos de software al interior de los procesos productivo.

Los agentes inteligentes integran una de las áreas con más rápido crecimiento en este rubro. En la red Internet, por ejemplo, un agente de software (también conocido como agente inteligente) se identifica como una aplicación decómputo que puede recolectar información o realizar servicios diversos sin necesidad de contar con una presencia real del usuario.

Las características principales de estos agentes de acuerdo a Jennings & Wooldridge [1] son:

- Autonomía: los agentes deben ser capaces de desarrollar la mayoría de sus tareas sin requerir la intervención directa de humanos y tener un cierto grado de control sobre sus propias acciones y su propio estado interno.
- Habilidad Social: los agentes deben ser capaces de interactuar con otros agentes.
- Responsividad: los agentes deben percibir su entorno, y responder con suficiente oportunidad a los cambios que puedan ocurrir en él.
- Proactividad: en el lapso de emisión de respuesta hacia el entorno, los agentes deben mostrar un comportamiento enfocado a cubrir sus metas y tomar la iniciativaen aquellos casos en los que sea apropiado.

Los agentes se utilizan regularmente en aplicaciones tales como administración personalizada de la información, de procesos industriales, de transacciones de comercio electrónico entre otros, haciendo énfasis especialmente en la administración del flujo de trabajo en un entorno de manufactura.

Una vez implementado un SMA (Sistema Multiagentes) en este entorno de trabajo, las características del mismo, tales como interacción local y dinámica no lineal – entre otras – permitirán que la distribución de órdenes a ser procesadas pueda realizarse en tiempos y con un número total de fallas cada vez menores.

El algoritmo implementado, presentado en el presente trabajo, se centra en tratar de reducir el tiempo de búsqueda del espacio que pueda recibir las piezas fabricadas, tratando de generar un equilibrio en la distribución de las órdenes.

Para lograr lo anterior, se utilizan conceptos tales como Temperatura, Calor Relativo y Calor Latente, los cuales se utilizan para programar los agentes en el sistema y, de este modo, revisar el impacto sobre el rendimiento global de la operación.

2. Estado del arte

El objetivo de este trabajo es el diseño, simulación y ejecución de la programación de órdenes de trabajo en las líneas de producción de una empresa del sector cerámico. Sepropone una arquitectura de agentes JADE [4] integrados al área de prensado como medio de enlace para distribuir dinámicamente los diferentes lotes producidos hacia el área de esmaltado, programados en base a un algoritmo de equilibrio de cargas.

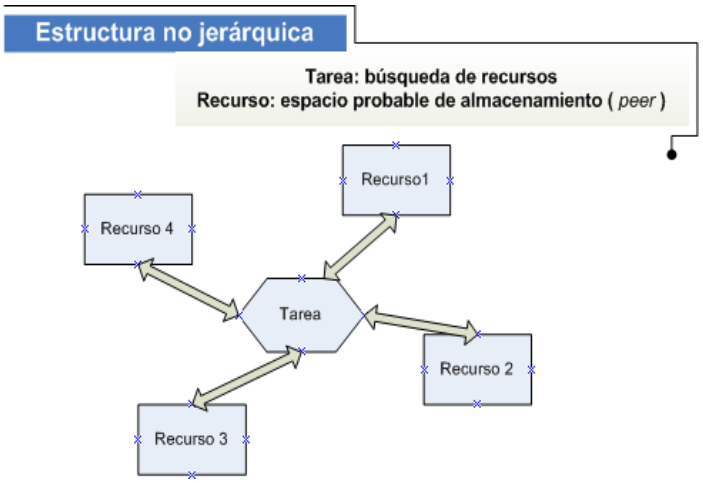

Fig. 1. Entorno más con ardillas.

3. Metodología

Debido al tipo de aplicación dentro del sistema, el estudio supone unaestructura descentralizada, no jerárquica, en la que se programará el comportamiento individual de cada agente en el sistema con base en su autonomía y cooperación interna (Ver figura 1).

La comunicación entre agentes se basa en la programación de comportamientos (Behaviours en el lenguaje de programación) mediante los cuales se simula la distribución de órdenes de trabajo del área de producción. Con esto, cada agente interactúa con el resto de manera dinámica, distribuyendo las piezas producidas en las líneas con capacidad de recepción.

El entorno debe adaptarse a la distribución dinámica de tareas. Al aplicar la programación de agentes, se tendrán dos comportamientos a ser observados: el del agente individual y el del sistema en general.

Para el agente individual, el objetivo será minimizar el tiempo de respuesta y en el caso de este algoritmo específico, la indicación correcta del estado de su temperatura.

El sistema deberá tender a una distribución uniforme de la carga de órdenes generadas sobre todos los agentes disponibles y con esto, estará mejor preparado para aceptar trabajo adicional y adaptarse más adecuadamente a fallas surgidas durante la operación.

La herramienta de desarrollo para implementar esta estrategia es la plataforma de agentes JADE, utilizando como base de distribución de tareas un algoritmo de optimización que asigna valor de temperatura a cada agente en el sistema,dependiendo de su carga de trabajo.

4. Entorno de trabajo

En torno a la sección de prensado, se asigna un agente en cada línea de trabajo.Cada uno de ellos, recibirá la orden de trabajo enviada por un agente de distribución y la aceptará o rechazará de acuerdo a si "temperatura", es decir, a la carga de trabajo que tenga en ese momento.

El agente de distribución maneja un "pizarrón de estado" en el cual se escriben las temperaturas de cada agente en el sistema. Ésta será la base mediante la cual la distribución de tareas tratará de mantenerse en equilibrio. Cuando un agente esté "caliente" el nivel calorífico correspondiente se asentará en el pizarrón y se buscará asignar el trabajo a un agente con menor temperatura, lo cual ayudará al proceso de "enfriamiento" del sistema.

La estructura anterior se implementa mediante tecnología de Agentes JADE con las siguientes características:

- Comunicación programada. Los agentes distribuyen las órdenes a lo largo de la red interna mediante protocolo TCP/IP
- Dinámica no-lineal. El agente de distribución lee la temperatura tanto individual como colectiva en el sistema y se decide por la opción de menor energía interna, que puede ser diferente en cada revisión.
- Reconfiguración sobre demanda. El lenguaje de programación permite ajustar, incluso en tiempo real la estructura multiagente para responder a las diferentes situaciones que puedan presentarse.

5. Experimentación

El algoritmo de solución propuesto, busca implementar una solución adecuada al problema de la distribución de órdenes de trabajo, mediante la comunicación y posterior comunicación de cada agente entre sí al interior del SMA. Las características generales implementadas en este sistema son:

- Generación aleatoria del total de órdenes de trabajo a ser procesadas, así como valores de referencia de las mismas. Estos últimos representan el nivel de prioridad de las tareas a desarrollarse.

- Generación aleatoria de valores de temperatura asociados a cada uno de los agentes en el sistema, referenciando con ellos su nivel de carga de trabajo. Comparativamente, para un mayor valor generado se tendrá un agente con más actividad y por tanto, con una temperatura más alta. Un valor bajo indicará un agente con menor temperatura, es decir, tendrá un menor nivel de carga en el sistema al momento de buscar opciones para la asignación de trabajos a ser procesados.

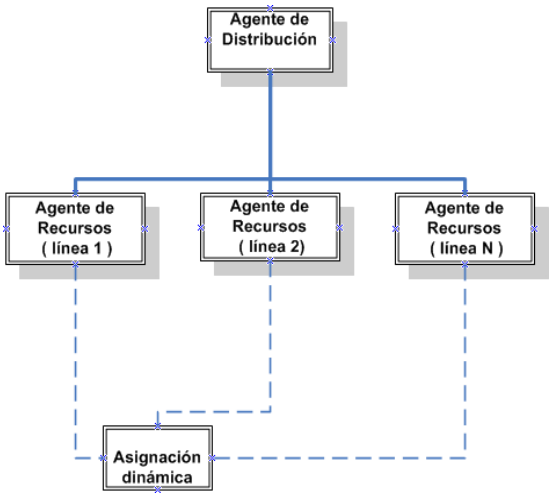

Fig. 2. Distribución de agentes en el sistema.

– Al terminar el proceso de generación y asignación de valores de carga y temperaturas, el sistema realiza una autoevaluación en busca de un equilibrio general: al presentarse un agente con mayor carga de trabajo (y por consecuencia, un incremento de temperatura) distribuye una parte del mismo hacia otro con menor carga. Esto permite una distribución más adecuada de las cargas de trabajo en el área de procesamiento.

En este contexto, el simulador de almacenamiento de recursos permite manejar los parámetros siguientes:

– Cantidad de piezas para distribuir. Desde 1 hasta *n* piezas.

– Cantidad de espacios de almacenamiento. Desde 1 hasta *n* espacios.

– Capacidad de almacenamiento en cada espacio. El simulador puede adaptarse a la capacidad específica requerida del sistema y, en tiempo de ejecución, comunicar a cada agente la capacidad admitida en ese momento.

– Cantidad de piezas almacenadas en cada espacio. Varía durante la ejecución. Al igual que la capacidad de almacenamiento, el simulador comunicará en el momento requerido, la cantidad almacenada.

7. Conclusión

En esta etapa del estudio, no se pretende hacer un desarrollo complejo o crear mecanismos de alto nivel para cooperación y negociación, solamente centrarse en un mecanismo simple, confiable y práctico para programación dinámica de tareas, debido a que se intenta utilizar como parte de un diseño integral basado en agentes y sistemas de manufactura para una fábrica real. Con esto se simplifica la complejidad del sistema,

se reduce el retardo de comunicación y por consecuencia, el tiempo de programación/reprogramación.

Los resultados obtenidos demuestran que integrando un esquema SMA acoplado al área de proceso bajo el algoritmo de Equilibrio de Temperaturas, se logra desarrollar un esquema de trabajo real con un desempeño eficiente de distribución de tareas.

Asimismo, una vez observado lo anterior, puede predecirse que es posible responder aposibles situaciones de emergencia, tales como:

- Reconfiguración para nuevos pedidos de emergencia Reconfiguración tras cancelación de pedidos.
- Reconfiguración para optimización de la programación actual Reconfiguración por ruptura de la máquina o producto destruido.

8. Trabajos futuros

Como líneas futuras de investigación dentro de este trabajo pueden destacarse lassiguientes:

- Implementación y comparación de eficiencia de otros algoritmos de programación como medida de reducción de tiempos de proceso.
- Desarrollo de interfaz gráfica del simulador para una mayor facilidad de uso.
- Reestructuración del código para permitir la introducción manual de datos en tiempo real, para enfrentar contingencias.
- Estudio detallado de requerimientos mínimos de *hardware* y *software* que permita un funcionamiento óptimo del simulador.
- Implantación en la fábrica de un sistema multiagente para la programación de producción en tiempo real.

Referencias

1. Jennings, N.R., Wooldridge, M.: Software Agents. IEEE Review, pp. 17–20 (1996)
2. Poli, R., Di Chio, C., Langdon, W.: Exploring Extended Particle Swarms: a Genetic Programming Approach. In: Proceedings of the conference on Genetic and Evolutionary Computation, pp. 169–176 (2005)
3. Parsopoulos, K.E., Vrahatis, M.N.: Evolutionary Computing and Optimization: Particle Swarm Optimization Method in Multiobjective Problems. In: Proceedings of the ACM Symposium on Applied Computing (2002)
4. Chien-Chung, S., Chaiporn, J.: Ad Hoc Multicast Routing Algorithm with Swarm Intelligence. Mobile Networks and Applications, 10(1-2) (2005)
5. JADE: Java Agent Development Framework (2005) http://jade.tilab.com.
6. Camorlinga, S., Barker, K.: Multiagent Systems Storage Resource Allocation in a Peer-to-Peer Distributed File System (2003)