Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 1, No. 2, pp. 44-53, Diciembre 2014
Selección de los Mejores Artículos de CoNCISa 2014

Enfoque Formal de Verificación y Autómatas Temporizados Aplicados a Procesos Industriales

Luis Mendoza[1]
lmendoza@usb.ve

[1] Departamento de Procesos y Sistemas, Universidad Simón Bolívar, Caracas, Venezuela

Resumen: Una de las metas de la Ingeniería de Software y del Modelado de Procesos de Negocio es lograr que los diseñadores de negocios y los desarrolladores de sistemas construyan aplicaciones de software confiables, principalmente para aquellos *Procesos Industriales Críticos* (PIC); es decir, aquellos procesos que impactan directamente la misión de una industria. El uso de *Métodos Formales*, tales como la técnica de *Verificación Automática* y la teoría de *Autómatas Temporizados* (AT), ha demostrado que propicia la confiabilidad y la seguridad de PIC al aumentar su comprensión, revelando inconsistencias, ambigüedades e incompletitudes, que de otra manera pasan desapercibidas. Este trabajo presenta la integración de un *Enfoque Formal de Verificación* (EFV) con la teoría de AT para la especificación y la verificación automática del *Modelo de Tareas (MT) asociado a un PIC* (MTPI). Además, se introducen un conjunto de directrices para transformar modelos de PIC en AT. El resultado del uso del EFV con la verificación automática y los AT es una infraestructura metodológica que garantiza la exactitud del MTPI con respecto a la especificación de las propiedades iniciales derivadas de las buenas prácticas o reglas de negocio de una industria. Con el fin de mostrar la viabilidad de la propuesta, se discute el caso Celda de Producción, un típico ejemplo para evaluar metodologías para el diseño de sistemas industriales, el cual ilustra cómo integrar la verificación automática en las etapas tempranas del diseño de PIC.

Palabras Clave: Procesos Industriales Críticos; Verificación Automática; Autómatas Temporizados; Especificación; Métodos Formales.

Abstract: One of the goals of Software Engineering and Business Process Modeling is to enable that business designers and system developers build reliable software applications, mainly for those that support *Critical Industrial Processes* (CIP); i.e. those processes that directly impact the mission of an industry. The use of *Formal Methods*, such as the *Model Checking* technique and the *Timed Automata* (TA) theory, has shown that promotes the reliability and safety of CIP to increase their understanding, revealing inconsistencies, ambiguities and incompleteness, which otherwise so go unnoticed. This paper presents the integration of a *Formal Verification Approach* (FVA) with TA theory for the specification and model checking of the *Task Model (TM) associated with a CIP* (MTIP). In addition, a set of guidelines to transform CIP models in TA are introduced. The result of using the FVA with model checking and TA is a methodological infrastructure that guarantees the accuracy of MTIP with respect to the specification of the initial properties derived from good practices or business rules of an industry. In order to show the feasibility of the proposal, a Production Cell case, a typical example to evaluate methodologies for designing industrial systems, is discussed, which illustrates how to integrate model checking on the early stages of CIP design.

Keywords: Critical Industrial Processes; Model Checking; Timed Automata; Specification; Formal Methods.

I. Introducción

Con la automatización de procesos industriales y el soporte que las *Tecnologías de la Información y Comunicación* (TIC) le dan a las industrias, muchos procesos se han convertido en el elemento fundamental para el funcionamiento de las industrias, así como para la realización de gran cantidad de procesos y actividades de la vida diaria. Como resultado de esto, algunos de los procesos industriales que utilizan las organizaciones se han catalogado como críticos —aquellos procesos que impactan directamente la misión de una industria. Esto se debe a que su funcionamiento condiciona el trabajo diario de las industrias y de muchos aspectos de la vida humana, lo que incluye un buen tiempo de respuesta. Como consecuencia, estos *Procesos Industriales Críticos* (PIC) necesitan ser verificados, certificados u homologados, antes de su puesta en funcionamiento [1].

La verificación de los aspectos temporales de un PIC soportado por las TIC puede ser visto como una actividad de tres etapas

que tiene como objetivo validar la viabilidad de los requerimientos, el diseño y la implementación. Para examinar los requerimientos y el diseño, se hace necesario la especificación de un modelo del comportamiento temporal del PIC propuesto. De este modelo se comprueba su confiabilidad y seguridad. Con el fin de expresar los requerimientos temporales y las propiedades de diseño, el esquema de modelado debe contener dos primitivas clave: plazos de espera y tiempos de ejecución [2]; además, debe ser capaz de expresar la concurrencia inherente a los sistemas de tiempo real. La verificación de la implementación de un PIC consiste en predecir el peor comportamiento temporal del PIC cuando todos los retrasos se contabilizan, y la posterior afirmación de que en estas circunstancias todos los plazos se cumplen. Esto implica generalmente la planificación de los tiempos de respuesta y su análisis asociado.

Por ello, para minimizar posibles omisiones o inconsistencias con respecto a lo que se espera de los procesos críticos de una industria, tanto a nivel funcional como a nivel de rendimiento de acuerdo a restricciones temporales, es necesario invertir esfuerzo en la verificación de los PIC que van a ser implementados. Debido a la compleja naturaleza y dinámica de algunas industrias, los modelos basados en lenguajes y técnicas formales —técnicas, lenguajes y herramientas, definidos matemáticamente para especificar y verificar procesos y sistemas— son necesarios para el análisis del comportamiento, a través de la verificación automática y el diseño de PIC, así como para el mejoramiento del funcionamiento de los existentes.

En este trabajo se presenta la integración de un *Enfoque Formal de Verificación* (EFV) con la teoría de *Autómatas Temporizados* (AT) para la especificación y la verificación automática del *Modelo de Tareas asociado a un PIC* (MTPI). También se ofrece un conjunto de directrices de mapeo para transformar los modelos de PIC en AT que pueden ser verificados con UPPAAL [3]. El resultado del uso del EFV con la verificación automática y los AT es una infraestructura metodológica que garantiza la exactitud del MTPI con respecto a la especificación de las propiedades iniciales derivadas de las buenas prácticas o reglas de una industria. Con el fin de mostrar la viabilidad de nuestra propuesta, también se discute un caso de carácter industrial, la Celda de Producción, un típico ejemplo para evaluar metodologías para el diseño de sistemas industriales, el cual ilustra cómo integrar la verificación automática en las etapas tempranas del diseño de PIC. Como resultado, logramos como parte de nuestros objetivos soportar al ingeniero de modelado de PIC con métodos, modelos y herramientas que le permitan verificar los modelos de los PIC que construye, desde el mismo momento que los define y los diseña.

Se han desarrollado varios trabajos que permiten la verificación y el análisis de procesos industriales utilizando algunos lenguajes y formalismos. En [4] se presenta un estudio que examina los esfuerzos de la aplicación de la verificación formal en el modelado de procesos. El autor presenta las técnicas de verificación formal (autómatas, redes de petri y álgebras de proceso) que permiten simular y verificar modelos gráficos de PIC en la fase de diseño de los procesos; y reitera que estas técnicas pueden detectar y corregir errores de los modelos en etapas tempranas del diseño; antes de la implementación. Por otra parte, indica que hay estudios, por ejemplo en [5], para verificar diagramas de máquinas de estado de *Unified Modeling Language* (UML) con verificación automática, que deben ser tomados en cuenta. Por nuestra parte, hemos realizado algunos aportes orientados al diseño y la verificación de sistemas con criticidad, reflejados en [6]. En cuanto a la verificación composicional, está el trabajo expuesto en [7]. Todos estos aportes nos han permitido conformar un EFV. Este enfoque integra la verificación composicional y la técnica de verificación automática, aplicables a la verificación formal de PIC.

Este artículo está estructurado como sigue. En la Sección II se presentan los conceptos relacionados con la verificación de PIC, se expone la teoría de AT y se introduce la herramienta UPPAAL. En la Sección III se presenta el EFV y, a continuación, en la Sección IV, se indican las directrices para transformar un PIC en AT. Posteriormente, en la Sección V, se discute el caso de la Celda de Producción y el uso de la herramienta UPPAAL. Finalmente, en la Sección VI, se enumeran las conclusiones y se esboza el trabajo futuro.

II. MARCO CONCEPTUAL

A. Verificación de Procesos Industriales y Criticidad

Según [8], los procesos industriales tienen una vinculación con los flujos de trabajo (en inglés, workflows), al indicar que éstos son la automatización de un procesos industriales, total o parcialmente, durante la cual materia (prima o semiprocesada), información y/o tareas, son pasadas de un participante a otro por una acción conforme a un conjunto de reglas procedimentales. Para [9], los procesos son secuencias específicas de actividades de trabajo a través del tiempo y del espacio, con un inicio, un fin y unas entradas y salidas claramente definidas: *una estructura para la acción*.

Para este trabajo los procesos industriales con criticidad, denominados PIC, son los procesos industriales de una industria que son esenciales (es decir, críticos) para cumplir la declaración de su misión; es decir, aquellos que deben ser restaurados inmediatamente después de una interrupción para garantizar la capacidad de la industria afectada para proteger sus activos, hacer frente a sus necesidades fundamentales, y cumplir la normativa y los requisitos obligatorios [10]. Esta importancia típicamente se basa en una evaluación de las consecuencias que implicaría un fallo del equipo, actividad o proceso, en el servicio que presta la empresa. A menudo las necesidades de mejora de los PIC se deben a que éstos no llegan a brindar un rendimiento satisfactorio [11]; es decir, no propician el logro positivo de los factores críticos de la organización de la mejor manera. Como parte del rendimiento está el cumplimiento de plazos de tiempo que garanticen su correcta ejecución [11].

La criticidad está asociada con la fiabilidad o confiabilidad (en inglés, reliability) de procesos automatizados; es decir, con

la probabilidad de que un equipo o sistema opere sin fallo por un determinado período de tiempo, bajo unas condiciones de operación previamente establecidas [12]. Adicionalmente, a nivel industrial, se habla de la confiabilidad operacional, referida a la capacidad de una instalación o sistema integrado por procesos, tecnología (máquinas) y gente, para cumplir su función dentro de sus límites de tiempo y bajo un contexto específico de operaciones [13].

Por ello, los conceptos de verificación y validación son importantes en este trabajo. La verificación se enfoca en el tema de la consistencia interna de un modelo, mientras que la validación está relacionada con la correspondencia entre el modelo y la realidad [14]. La validación es el proceso de comparar la salida del modelo del PIC con el comportamiento organizacional. En otras palabras: comparar la ejecución del modelo del PIC con la realidad (física u otra cualquiera). La verificación es el proceso de comparar la implementación (una forma de ejecución) de un PIC (que puede ser un programa o una especificación formal detallada) con el modelo del PIC, para garantizar que la implementación es correcta con respecto al modelo del PIC. El término validación se aplica a aquellos procesos que buscan determinar si una implementación es correcta o no respecto al sistema real. De forma más sencilla, la validación trata sobre la cuestión ¿se está construyendo/modelando el PIC correcto?, mientras que la verificación responde a ¿se está construyendo/implementando correctamente el PIC?. La verificación comprueba que la implementación del modelo (una forma de ejecución detallada y formal) corresponde al modelo del PIC, mientras que la validación comprueba que el modelo del PIC corresponde con la realidad [14].

En este trabajo nos centramos en la verificación formal, ya que por las características de los PIC y el impacto de éstos en las industrias, además de lo complejo que son los sistemas y maquinarias que los soportan, es sumamente costoso y riesgoso esperar que el PIC esté implementado para validar que la implementación cumple con lo que se quiere en la realidad.

B. Teoría de Autómatas Temporizados

El recurso más utilizado actualmente en la verificación del modelo de un procesos industriales es el uso de autómatas que implementan la ejecución del proceso [15]. Los autómatas son modelos matemáticos de un sistema (o proceso) con entradas y salidas, pudiendo estar en cualquiera de un número finito de estados. Cada estado establece la relación entre entradas anteriores para alcanzar ese estado y las entradas posteriores (salidas) para alcanzar un estado siguiente [16]. La entrada es leída símbolo por símbolo (alfabeto del autómata), hasta que es *consumida* completamente —piense en ésta como una palabra que es leída por una cabeza lectora; la cabeza se mueve a lo largo de la palabra, leyendo un caracter (o símbolo) a la vez— una vez la entrada se ha agotado, el autómata se detiene.

El concepto de AT fue propuesto por primera vez por [17] como una extensión de la teoría de autómatas enfocada al modelado de sistemas en tiempo real [16]. De acuerdo con la teoría de AT, un autómata temporizado es un grafo dirigido finito anotado con condiciones y restablecimientos de los valores reales no negativos de relojes, donde un sistema se modela como una colección de máquinas de estados finitos y un conjunto finito de relojes [17]. En el esquema estándar, los relojes están sincronizados y pueden ser restaurados (*reset*) a través del paso de un estado a otro. Los relojes también se utilizan como guardas para las transiciones.

Definición 1: **Autómata temporizado.** Un *autómata temporizado* es una tupla $\mathcal{A} = \langle S, \Sigma, C, E, s_0 \rangle$ que consiste de los siguientes componentes:

- S es un conjunto finito. Los elementos de S son llamados los estados de $\mathcal{A}$.
- Σ es un conjunto finito llamado el alfabeto de acciones de $\mathcal{A}$.
- C es un conjunto finito llamado los relojes de $\mathcal{A}$.
- $E \subseteq S \times \Sigma \times B(C) \times P(C) \times S$ es el conjunto de transiciones de $\mathcal{A}$, donde
  - $B(C)$ es el conjunto de restricciones booleanas de los relojes de C, y
  - $P(C)$ es el conjunto potencia de C.
- s_0 es un elemento de S, llamado el estado inicial.

Una transición $\langle s, a, g, r, s' \rangle \in E$ es el cambio del estado s hasta s' con las acciones a, las guarda g y el reinicio del reloj r .

La teoría de AT estipula que el tiempo es continuo, pero la declaración de los relojes por lo general se limita a la utilización de valores enteros. El tiempo nunca es negativo así como que los relojes sólo pueden ser reiniciados a 0. El tiempo de vida está representado por el requerimiento de que algunos relojes específicos nunca pueden obtener un valor superior a un plazo determinado, o que no pueden alcanzarlo mientras no se supere un estado o una colección de estados. Las transiciones se definen para ser instantáneas. Por ejemplo, de acuerdo con la Figura 1, la transición de salida del estado S no puede ocurrir antes de $T = 3$.

Figura 1: Ejemplo de AT

En este ejemplo, T es un reloj; que se está reiniciando cuando se alcanza el estado S. En un modelo simple (es decir, el invariante $T < 5$ no existe), la única salida del estado S es cuando T es superior a 3; es decir, el ejemplo, ilustra la imposición de un retraso (*delay*). Un estado puede tener también un invariante temporal para forzar una transición de salida. La Figura 1 ilustra esto porque el estado S no puede salir antes de $T = 3$, sino que debe salir después de 3 y antes de $T = 5$. Si por alguna razón la transición no se puede tomar, el autómata contiene una condición de error o punto muerto (deadlock). Cuando hay dos o más posibles transiciones de un estado a otro, cada una es una transición válida.

Tal como se introdujo anteriormente, la semántica de un autómata temporizado se define como un sistema de transición en la que un estado o configuración consiste en una ubicación

(o estado) dada y los valores de los relojes en ese momento. Hay dos tipos de transiciones entre estados. El autómata puede aplazar por un tiempo (una transición de retardo) o bien seguir un camino permitido (una transición de acción). Una *acción temporizada* es un par (t, a), donde $a \in \Sigma$ es una acción tomada por el autómata temporizado $\mathcal{A}$ después de $t \in \mathbf{R}_+$ unidades de tiempo desde que $\mathcal{A}$ se ha iniciado. El tiempo absoluto t se llama *captura de tiempo* de la acción a. Una *traza temporizada* es una secuencia (posiblemente infinita) de acciones temporizadas $x_i = (t_1, a_1)(t_2, a_2)\ldots(t_i, a_i)\ldots$ donde $t_i \leq T_{i+1} \forall i \geq 1$.

Para modelar sistemas concurrentes (como los PIC), los AT pueden ser extendidos con la composición paralela. En las álgebras de procesos, se han propuesto varios operadores de composición paralela para modelar diferentes aspectos de concurrencia —véase, por ejemplo *Calculus of Communicating Systems* (CCS) [18] y *Communicating Sequential Processes* (CSP) [19]. Estos operadores algebraicos pueden adoptarse en la teoría de AT, la cual permite el intercalado de las acciones, así como la sincronización entre AT. Esencialmente, la composición paralela de un conjunto de AT es el producto de los autómatas, denominado *red de AT*.

Definición 2: **Red de AT.** Una *red de AT* es la composición paralela $RAT = \mathcal{A}_1 \parallel \cdots \parallel \mathcal{A}_n$ de un conjunto de autómatas temporizados $\mathcal{A}_1 \ldots \mathcal{A}_n$, llamados sub–procesos, combinados en un único sistema (o proceso) por un operador de composición paralela con la ocultación de todas las acciones internas. La comunicación síncrona entre los sub–procesos es establecida mediante acciones de entrada y salida. Se asume que el alfabeto de acciones constará de los símbolos para las acciones de entrada $c?$, las acciones de salida $c!$ y las acciones internas representadas por el símbolo τ.

Los AT están diseñados para que puedan ser comprobados utilizando la verificación automática. En la verificación automática *se comprueba si un modelo formal es correcto respecto a los requerimientos expresados en una lógica temporal* [20]. Intuitivamente, la verificación automática funciona mediante la exploración de todas las posibles transiciones de estado desde el estado inicial del sistema. Todos las trazas posibles desde el conjunto de estados existentes al principio del autómata se exploran para ver si un estado inseguro puede ser alcanzado o una condición de vivacidad no se cumple. La verificación automática es una técnica que requiere el apoyo de herramientas. En este trabajo se utiliza la herramienta UPPAAL [3], porque es compatible con la representación gráfica de los AT y permite que el usuario de la herramienta pueda interactuar con un programa de edición para crear y modificar modelos de AT; es decir, redes de AT.

C. UPPAAL

UPPAAL[1] [3] es una herramienta de software que permite modelar, validar y verificar sistemas en tiempo real. UPPAAL sirve como un lenguaje de diseño o de modelado que permite describir el comportamiento de un sistema o proceso como redes de AT ampliadas con variables de datos y de reloj.

La herramienta cuenta con un simulador interactivo que es una herramienta de validación que permite examinar las posibles ejecuciones dinámicas de un sistema durante las fases de diseño y proporciona una sencilla detección de los fallos antes de que este sea verificado por el comprobador de modelos. Por su parte, el comprobador de modelos (o verificador automático) puede chequear las propiedades —especificadas en una lógica temporal basada en *Timed Computational Tree Logic* (TCTL), que es una especialización de la semántica de la lógica temporal Computational Tree Logic (CTL), partiendo de la misma sintaxis— explorando el espacio de posibles ejecuciones del autómata. En UPPAAL, el *autómata producto* se calcula *on–the–fly* durante la verificación. También proporciona diagnósticos que se pueden abrir automáticamente a través del simulador y pueden usarse para visualizar e investigar las trazas de símbolos leídos del alfabeto [3].

Desde el punto de vista de los analistas/diseñadores de negocios, UPPAAL es fácil de usar; aunque para modelos complejos se requiere una considerable capacidad de cómputo por parte de la herramienta. Tal como se indicó anteriormente, el formalismo subyacente en UPPAAL es el de redes de AT [3], que es de fácil comprensión para la gente de negocios y reflejan fielmente el comportamiento de los participantes involucrados en la ejecución de un PIC. El verificador de la herramienta soporta una lógica temporal bastante expresiva y el simulador es muy apreciado por los responsables de analizar/modelar procesos, ya que es muy visual para todo el proceso de verificación.

III. ENFOQUE FORMAL DE VERIFICACIÓN

Con el fin de contribuir a la formalización del complejo campo del modelado de PIC, consideramos adecuado la incorporación de un modelo ejecutable de los PIC, conocido como Modelo de Tareas del PIC (MTPI), para lograr este propósito. Un MTPI puede ser considerado como el modelo más básico posible para obtener una descripción completa, exacta, coherente y bien definida del conjunto de actividades y tareas que deben realizar los participantes/máquinas en un PIC para cumplir los objetivos de la industria [21]. La formalización de un MTPI establece las bases para la verificación automática de los modelos de PIC.

Bajo este enfoque, la falta de verificación de propiedades funcionales y no funcionales de propiedades de los MTPI se debe principalmente a problemas semánticos y sintácticos causada por la incorrecta integración de: (1) la formalización de propiedades (metas y objetivos de la industria, representados por un modelo del PIC) en tiempo de diseño [22], y (2) el modelo ejecutable (MTPI) correspondiente en tiempo de ejecución [22] que es necesario derivar de cualquier PIC para llevar a cabo la verificación automática del modelo del PIC. En este trabajo se propone la construcción de un MTPI como una red de autómatas temporizados. Así, las restricciones temporales del modelo del PIC en tiempo de diseño pueden ser especificadas por los autómatas temporizados en tiempo

[1] http://www.uppaal.org

de ejecución. Al proceder de esta manera, podemos obtener la verificación completa de un MTPI a partir de un modelo inicial del PIC de una manera confiable y soportado por una herramienta de verificación automática como UPPAAL.

Sobre la base de los conceptos e ideas anteriores, se propone una adaptación del esquema conceptual, conocido como EFV, que integra la teoría de AT con la verificación automática, el cual se presenta en la Figura 2.

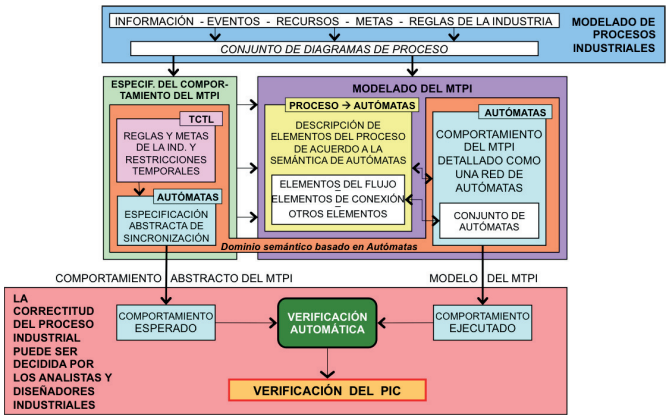

Figura 2: Enfoque Formal de Verificación Utilizando AT

El enfoque presentado en este artículo se basa en el hecho de que el desempeño de un PIC P está estructurado por el trabajo llevado a cabo por varios participantes o componentes (que pueden ser humanos o máquinas) verificados que trabajan en paralelo, $P = ||_{i:1..n} P_i$, donde el comportamiento de cada participante P_i satisface la propiedad ϕ_i, lo que representa la especificación del comportamiento que el participante debe exhibir. El principal objetivo es hacer posible la verificación del comportamiento de un PIC completo a partir de la verificación del comportamiento de sus participantes. En este sentido,

Definición 3: **Composicionalidad de propiedades.** Una propiedad ϕ es composicional si y sólo si para cualesquiera dos AT $\mathbf{A}_1$, $\mathbf{A}'_1$, y $\mathbf{A}_2$ con $\mathbf{L}(\mathbf{A}_2) \cap \mathbf{L}(\phi) = \emptyset$ se mantiene

$$(\mathbf{A}_1 \models \phi) \Rightarrow ((\mathbf{A}_1 \| \mathbf{A}_2 \models \phi) \vee \mathbf{A}_1 \| \mathbf{A}_2 \models \delta)) \quad \text{y} \qquad (1)$$

$$((\mathbf{A}_1 \sqsubseteq \mathbf{A}'_1) \wedge (\mathbf{A}'_1 \models \phi)) \Rightarrow (\mathbf{A}_1 \models \phi) \qquad (2)$$

Es decir, las propiedades locales son preservadas por la composición paralela de componentes cuando el etiquetado es disjunto:

Lema 1: Para dos ATs $\mathbf{A}_1$ y $\mathbf{A}_2$ y propiedades ϕ_1 y ϕ_2 con $\Sigma_1 \cap \Omega_2 = \emptyset$, $\Sigma_2 \cap \Omega_1 = \emptyset$, $\mathbf{L}(\mathbf{A}_1) \cap \mathbf{L}(\mathbf{A}_2) = \emptyset$ se mantiene:

$$((\mathbf{A}_1 \models \phi_1) \wedge (\mathbf{A}_2 \models \phi_2)) \Rightarrow (A_1 \| A_2 \models \phi_1 \wedge \phi_2). \qquad (3)$$

Prueba: Como $\mathbf{L}(\mathbf{A}_1) \cap \mathbf{L}(\mathbf{A}_2) = \emptyset$, podemos concluir que si ϕ_2 se mantiene, ésta no es influenciada por $\mathbf{A}_1$. Dada la asunción $\Sigma_1 \cap \Omega_2 = \emptyset$, $\Sigma_2 \cap \Omega_1 = \emptyset$, $\mathbf{L}(\mathbf{A}_1) \cap \mathbf{L}(\mathbf{A}_2) = \emptyset$, ambos AT son ejecutados independientemente en paralelo y así $\mathbf{A}_2 \models \phi_2$ implicará $A_1 \| A_2 \models \phi_2$, y $\mathbf{A}_1 \models \phi_1$ implicará $A_1 \| A_2 \models \phi_1$. Por consiguiente, $A_1 \| A_2 \models \phi_1 \wedge \phi_2$ ha sido probado.

Cada participante P_i debe satisfacer la invariante (ψ_i), la cual representa el comportamiento de los otros participantes que colaboran en el proceso con respecto a P_i. Se usa el símbolo especial $\neg\delta$ para denotar que el bloqueo del participante (es decir, que el participante caiga en un estado sin ninguna transición de salida) no puede producirse en ninguna ejecución posible en la que colabore. La propiedad ϕ y el invariante ψ que son satisfechos por el sistema P son obtenidas a partir de las propiedades locales ϕ_i ($\bigwedge_{i:1..n} \phi_i \Rightarrow \phi$) y de los invariantes locales ψ_i ($\bigwedge_{i:1..n} \psi_i \Rightarrow \psi$), respectivamente. Como resultado, podemos obtener la verificación completa del PIC utilizando el Teorema 1[2]:

Teorema 1: **Verificación de PIC.** Sea el PIC P estructurado en varios participantes trabajando en paralelo, $P = \|_{i:1..n} P_i$. Para un conjunto de $AT(P_i)$ que describen el comportamiento de los participantes P_i, las propiedades ϕ_i, los invariantes ψ_i, y el bloqueo δ, con $\bigcap_{i:1..n} \Sigma_i = \emptyset$, $\bigcap_{i:1..n} \Omega_i = \emptyset$, y $\bigcap_{i:1..n} \mathbf{L}(AT(P_i)) = \emptyset$, la siguiente condición se comple:

$$AT(P) \models (\phi \wedge \psi \wedge \neg\delta) \Leftrightarrow \|_{i:1..n} AT(P_i) \models \bigwedge_{i:1..n} (\phi_i \wedge \psi_i) \wedge \neg\delta, \qquad (4)$$

donde $AT(P) = \|_{i:1..n} AT(P_i)$.

Prueba: Asumimos que el comportamiento del participante P_i puede ser descrito por el $AT(P_i)$, a fin de:

$$AT(P_i) \models (\phi_i \wedge \psi_i) \wedge \neg\delta \ .$$

Por el operador de composición paralela de los cálculos de procesos, la siguiente relación debe ser satisfecha:

$$AT(P) = \|_{i:1..n} AT(P_i) \ .$$

Si los $AT(P_i)$ satisfacen,

1) $\bigcap_{i:1..n} \Sigma_i = \emptyset$ y $\bigcap_{i:1..n} \Omega_i = \emptyset$, y
2) $\bigcap_{i:1..n} \mathbf{L}(AT(P_i)) = \emptyset$,

podemos afirmar que éstos pueden ser compuestos y, por el Lema 1, escribimos:

$$\|_{i:1..n} AT(P_i) \models \bigwedge_{i:1..n} (\phi_i \wedge \psi_i) \wedge \neg\delta$$

Dado que la relación de satisfacción es cerrada con respecto al operador de conjunción, concluimos $\bigwedge_{i:1..n} \phi_i \Rightarrow \phi$, $\bigwedge_{i:1..n} \psi_i \Rightarrow \psi$, y por consiguiente:

$$AT(P) \models \phi \wedge \psi \wedge \neg\delta \Leftrightarrow \|_{i:1..n} AT(P_i) \models \bigwedge_{i:1..n} (\phi_i \wedge \psi_i) \wedge \neg\delta \ . \qquad (5)$$

La condición suficiente $\phi \Rightarrow \bigwedge_{i:1..n} \phi_i$, $\psi \Rightarrow \bigwedge_{i:1..n} \psi_i$, puede ser demostrada considerando que aun en el caso donde todos los participantes P_i presenten un comportamiento diferente,

[2]Se denota $AT(P) \models \phi$ cuando el $AT(P)$ mantiene la propiedad ϕ para todos sus estados iniciales, es decir, $\forall s \in I \Rightarrow (AT(P), s) \models \phi$. Para los invariantes tenemos $AT(P) \models \psi$ cuando para todos los $s \in S$ se mantiene $(AT(P), s) \models \psi$. Para denotar que $AT(P)$ no contiene algún bloqueo, escribimos $AT(P) \models \neg\delta$.

ϕ y ψ podrán ser definidas como la conjunción de las propiedades locales de cada componente (ϕ_i, ψ_i).

La aplicación práctica del Teorema 1 incluye realizar un proceso de *chequeo de satisfacción* inductivo en el rango del número de participantes ($i : 1..n$) de nuestro PIC. La herramienta UPPAAL nos soporta en este chequeo, de acuerdo a los siguientes pasos de aplicación del EFV y del Teorema 1. El EFV consta de los siguientes procesos integrados de acuerdo a la técnica de verificación automática y la teoría de AT (ver Figura 2):

1) *Modelado del MTPI*. Se modela y especifica el comportamiento del MTPI con AT, obteniéndose la descripción completa de su comportamiento. Para lograr esto, se utilizan las directrices que se exponen en la Sección IV de este artículo. De esta forma, se obtiene la especificación del comportamiento de los participantes (personas y/o máquinas) que colaboran en el MTPI en tiempo de ejecución, de acuerdo a la secuencia discreta de eventos en los cuales los participantes del MTPI están involucrados.
2) *Especificación del comportamiento de MTPI*. En paralelo a lo anterior, los requerimientos y las restricciones temporales que el MTPI debe cumplir son especificados a través de fórmulas de la lógica temporal TCTL, que es interpretada por UPPAAL. Posteriormente, estas propiedades son expresadas como AT de manera automática por UPPAAL . Como consecuencia, las propiedades del MTPI se expresan en el mismo lenguaje de especificación que el modelo del MTPI (es decir, AT), pudiendo razonar y ejecutar la verificación automática en un mismo dominio semántico.
3) *Verificación del MTPI*. Finalmente, obtenidos los resultados de los procesos anteriores realizados en paralelo, procedemos a la verificación del comportamiento del MTPI vs. el comportamiento abstracto esperado de éste. Esto se realiza a través de la incorporación de la herramienta de software UPPAAL, la cual permite simular y analizar este comportamiento de una manera cómoda para los diseñadores de PIC.

IV. DIRECTRICES PARA TRANSFORMAR PIC A AT

Una de las contribuciones de este trabajo es proponer un conjunto sencillo de pasos para derivar el MTPI correspondiente al modelo de un PIC, como una red de AT. Esta transformación es la primera etapa necesaria para realizar la evaluación (es decir, el análisis cualitativo) de un PIC, tal como se mostró en la Sección III. Después, la red de AT generada por la transformación, lo que representa el MTPI, se analiza con el apoyo de UPPAAL, para proporcionar diversos indicadores de rendimiento empresarial (por ejemplo, tiempo de servicio, tiempo o tamaño de la cola) de un PIC.

Tal como ya se ha dicho, un PIC consiste en un conjunto de actividades coordinadas que se ejecutan a lo largo un entorno organizacional y técnico [23]. Un PIC se estructura en una serie de tareas que deben ser realizadas por participantes (personas o sistemas automatizados) limitados por un conjunto de condiciones de negocio y en un plazo de tiempo específico. Una tarea es una unidad atómica y lógica del trabajo que se lleva a cabo por completo por un recurso [24]. En estas tareas, los participantes colaboran siguiendo un flujo de trabajo (o proceso) definido por un servicio dado. Además, los PICs se ven limitados por un conjunto de reglas de negocio, las cuales deben ser acatadas y determinan la estructura de la información y las políticas de la industria [25].

El proceso (o sub–proceso) que ejecuta un participante pasa por varios estados. Estos estados están unidos por la secuencia de ejecución que representan las transiciones de estado especificados por el proceso. Las transiciones pueden entrar o salir de un estado y tienen asociadas las guardas. Los pases de mensajes corresponden a la sincronización de sub–procesos (o procesos separados).

Para transformar el modelo de un PIC en una red de AT, el modelo del PIC es separado en tres aspectos: el sub–proceso que modela el comportamiento de cada participante, representado por un autómata, las relaciones entre los sub–procesos que colaboran para lograr el comportamiento del proceso, representadas por la red de AT, y las restricciones temporales del PIC, derivadas de las reglas de negocio. En pocas palabras, los dos primeros son mapeados en los AT y en la sincronización entre AT (es decir, para conformar la red de AT correspondiente), respectivamente, y las restricciones son mapeadas como invariantes, guardas y asigaciones temporales en los AT que conforman la red.

A. Participantes

El comportamiento de un participante en un PIC se modela como un sistema de transición de estados o autómata, que se caracteriza por sus acciones por los canales de sincronización, las invariantes, guardas, y asignaciones. La siguiente definición formaliza el mapeo propuesto.

Definición 4: **Participante.** Un *participante* modelado en un PIC, corresponde a un autómata temporizado $\mathcal{A} = \langle S, \Sigma, C, E, s_0 \rangle$ que consiste en los siguientes componentes:

- S es un conjunto finito de los estados de $\mathcal{A}$.
- Σ es un conjunto finito llamado el alfabeto o acciones (entrada y salida) de $\mathcal{A}$, que es la comunicación sincrónica entre participantes.
- C es un conjunto finito de los relojes de $\mathcal{A}$.
- $E \subseteq S \times \Sigma \times B(C) \times P(C) \times S$ es el conjunto de transiciones de $\mathcal{A}$, donde
  - $B(C)$ es el conjunto de restricciones booleanas de los relojes de C, que representan las restricciones temporales, y
  - $P(C)$ es el conjunto potencia de C.
- s_0 es un elemento de S, llamado el estado inicial.

Una transición $\langle s, a, g, r, s' \rangle \in E$ es el cambio del estado s hasta s' con acciones a, guarda g y el reinicio del reloj r.

Las constantes referidas a los invariantes, las guardas y las asignaciones, son parámetros temporales; son valores que se deciden en función de otros aspectos; es decir, las relaciones

entre los participantes y las restricciones temporales establecidas por el modelo del PIC. Por ejemplo, para especificar el comportamiento de las tareas, los invariantes ($t \leq max$), las condiciones de las guarda ($t \geq max$) y las asignaciones o reinicio de los relojes ($t := 0$) se definen en los estados y transiciones correspondientes en el autómata temporizado. Por lo tanto, t es una variable de reloj que tiene el tiempo transcurrido de la tarea y el invariante y la guarda especifican que la transición de la tarea a cualquier otra tarea del flujo no puede ocurrir hasta que el tiempo de ejecución mínimo haya transcurrido (min), y debe ocurrir antes de que el tiempo máximo de ejecución haya transcurrido (max). Las constantes max y min son parámetros temporales.

B. Sincronización de Participantes

Un participante dentro de cualquier industria *debe* interactuar con otros participantes dentro de la misma industria o con otros participantes de diferentes empresas para llevar a cabo por completo un PIC. Por lo tanto, para obtener la representación formal del MTPI (ver la Figura 2), que describe cómo los participantes realizan las tareas y colaboran entre ellos, la red de AT debe ser construida. Como resultado, se obtiene una red de AT que especifica y establece los aspectos del comportamiento y de las restricciones temporales de los participantes involucrados en la ejecución del MTPI.

Definición 5: **Sincronización de participantes.** La *sincronización de participantes* se obtiene como la composición paralela $RAT = \mathcal{A}_1 \parallel \cdots \parallel \mathcal{A}_n$ de un conjunto de AT $\mathcal{A}_1 \ldots \mathcal{A}_n$, denominados participantes, combinados en un único sistema por un operador de composición paralela con la ocultación de todas las acciones internas. La comunicación síncrona entre los participantes se establece mediante acciones de entrada y salida a través de un canal c. Se asume que el alfabeto de acciones consta de los símbolos para las acciones de entrada $c?$, las acciones de salida $c!$ y las acciones internas representadas por el símbolo τ.

C. Guía para Realizar la Transformación

Según nuestra experiencia, la transformación de un modelo/diagrama de un PIC en una red de AT se puede lograr a través de los siguientes pasos:

1) *Incluir restricciones temporales en el modelo/diagrama del PIC*. Para la verificación de las propiedades temporales de un PIC, como el tiempo de respuesta de un participante, y las restricciones temporales que puedan derivarse de las reglas de negocio, tales como el tiempo de ejecución de las actividades, éstas deberán especificarse en el modelo/diagrama del PIC. Se recomienda utilizar algún tipo de anotación permitida por la técnica de modelado utilizada para diagramar el PIC, con la finalidad de facilitar el trabajo de los analistas/diseñadores de PICs.
2) *Obtener el autómata temporizado correspondientes a cada participante*. Con referencia a la Definición 4, se construye el autómata temporizado necesario para especificar el comportamiento de cada trabajador y que formaliza el flujo de trabajo que debe seguir el trabajador para colaborar en la ejecución del PIC.
3) *Asignar a los parámetros correspondientes del autómata temporizado los valores de las restricciones temporales indicados en el modelo/diagrama del PIC*. De acuerdo con las anotaciones indicadas en el modelo/diagrama del PIC, se definen los invariantes, las guardas y las asignaciones en el autómata temporizado.
4) *Obtener la red de TAs correspondiente a la colaboración entre participantes*. A partir de la Definición 5, se construye la red de AT que conforma el MTPI que especifica los aspectos de comportamiento y las restricciones temporales de los participantes que colaboran en el MTPI. La asignación de los canales en los AT por donde se transmiten las acciones de salida y de entrada se establece de acuerdo a la sincronización de los participantes. En función de las restricciones de concurrencia podría ser necesario cambiar la estructura de la red de AT.

Como resultado de aplicar los pasos anteriores, se construyen los AT (como los que se muestran en la Figura 5 y en la Figura 6 de la Sección V) y la red de AT es integrada (como se muestra en la Figura 7 de la próxima sección), con el fin de llevar a cabo el análisis cualitativo del PIC de una manera fácil, a un adecuado nivel de formalidad.

V. Celda de Producción

La Celda de Producción es un ejemplo típico para evaluar metodologías para el diseño de PIC automatizados [26]. Es un modelo de una instalación industrial para el procesamiento de metal en Karlruhe, Alemania. Desafortunadamente, la especificación original de la Celda de Producción no contiene ninguna restricción temporal y se ignora la sincronización temporal. Se han propuesto una serie de estudios de caso avanzados de la Celda de Producción, pero muchos de ellos son bastante diferentes (incluso de acuerdo a cuestiones temporales). En este trabajo nos quedamos con la especificación realizada en [27], la cual es una simplificación y le añade el tiempo en todas las operaciones y requerimientos. Esto tiene un impacto importante en cualquier diseño propuesto. Como se puede ver en la Figura 3, el modelo incluye varias máquinas que deben coordinarse con el fin de forjar piezas de metal [27].

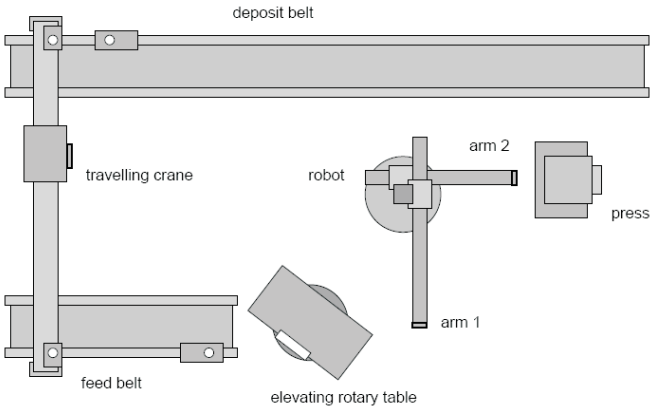

Figura 3: Vista Superior de la Celda de Producción [26]

Para modelar con realismo el comportamiento de la Celda de Producción, hay que considerar que todas las tareas toman un tiempo para su ejecución. Suponemos que la mayoría de las operaciones tienen un tiempo fijo; sin embargo, como un ejemplo de una especificación más real, para las tareas de la prensa se especifican los valores máximos y mínimos de ejecución. Por tanto, cualquier valor entre estos límites es aceptable.

Los requerimientos temporales sobre la Celda de Producción están vistos según el movimiento de una placa desde que entra hasta que sale de la celda. Por ello, se establece que el tiempo del peor caso para una placa (desde la llegada de la banda de alimentación hasta su remoción de la banda de depósito) es 200 unidades de tiempo [27]. Esto representa un tiempo límite para el modelo; es decir, una propiedad de vivacidad a verificar. En la definición original de la Celda de Producción no hay ninguna propiedad de seguridad; sin embargo, en el diseño que se hace en la próxima sección se introducen algunas obligaciones, que se traducen en invariantes de seguridad.

Tenga en cuenta que la definición textual o con alguna anotación gráfica sobre la Celda de Producción corresponde a la etapa 1 de la guía para la realización de la transformación presentada en la Sección IV. En la siguiente sección, estas anotaciones se utilizan para cumplir con el paso 3 de esta guía y para obtener el MTPI del PIC.

A. *Modelado y Verificación*

Ahora procedemos a modelar el MTPI; es decir, la ejecución y la sincronización de la red de AT de los participantes implicados en el MTPI de la Celda de Producción. Para obtener este modelo, se aplicaron las Definiciones 4 y 5, de acuerdo a los pasos 2, 3 y 4, de la guía para la realización de la transformación presentada en la Sección IV.

A continuación se muestra de forma resumida el modelado de uno de los elementos centrales del sistema con la herramienta UPPAAL: la prensa. La tarea de la prensa es forjar piezas de metal. La prensa consiste de dos planchas horizontales, donde la plancha inferior se mueve a lo largo del eje vertical. La prensa opera presionando la plancha inferior contra la plancha superior. Debido a que los brazos del robot están colocados en diferentes planos horizontales, la prensa tiene tres posiciones (ver Figura 4). En la posición inferior, la prensa es descargada por el brazo 2 del robot, mientras que en la posición del medio es cargada por el brazo 1.

La operación de la prensa es coordinada con los brazos del robot como sigue: primero, la prensa se abre en la posición inferior y espera hasta que el brazo 2 retire la pieza de metal y deje la prensa; a continuación, se mueve la plancha inferior a la posición del medio y espera hasta que el brazo 1 cargue la prensa y se retire; finalmente, la prensa se cierra forjando la pieza de metal. Esta secuencia de procesamiento es ejecutada cíclicamente. En la Figura 5 y en la Figura 6 se pueden observar los autómatas, realizados en la herramienta UPPAAL, que modelan el comportamiento de la prensa y el robot, respectivamente. Se puede apreciar que el robot es el componente más complejo de la Celda de Producción.

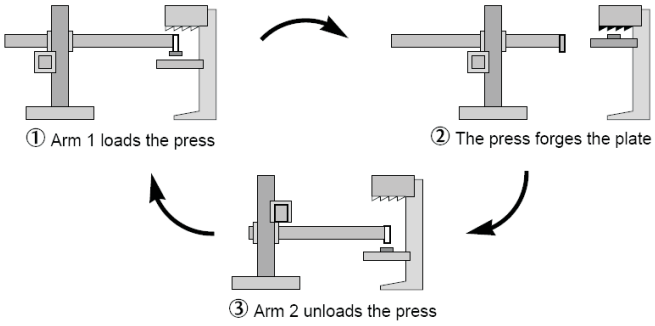

Figura 4: Vista Lateral de la Prensa y de los Brazos del Robot de la Celda de Producción [26]

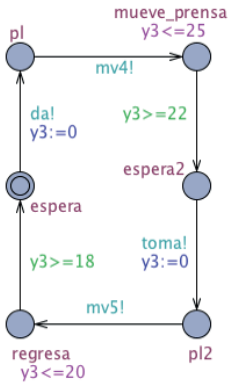

Figura 5: Autómata de la Prensa de la Celda de Producción

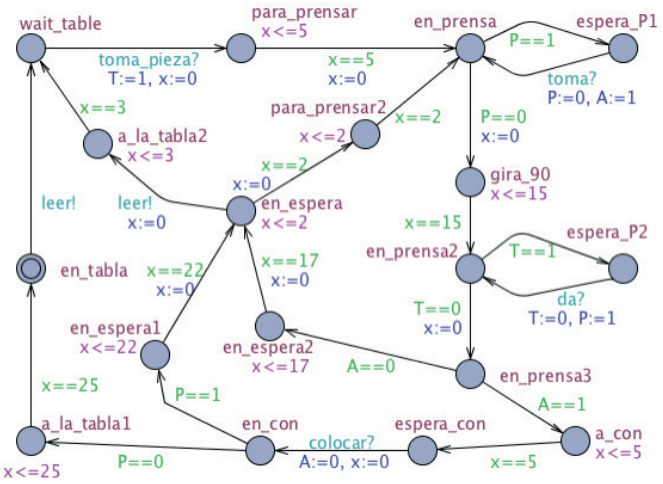

Figura 6: Autómata del Robot de la Celda de Producción

En la Figura 7 se muestra una pantalla del simulador de la herramienta UPPAAL, el cual contiene la red de AT que conforman el MTPI de la Celda de Producción. A través del simulador es posible apreciar la ejecución del MTPI completo, lo que permitió validar que el modelo funciona y que se puede proceder a la verificación de las restricciones temporales que debe satisfacer, en función de las reglas de la industria. En el caso de una contradicción o bloqueo al momento de simular el PIC a verificar (es decir, no se ha procedido a introducir las fórmulas en TCTL al verificador, tal como se muestra más adelante), el analista puede ajustar los parámetros de las restricciones temporales hasta conseguir los valores ideales

para el PIC modelado. Esto, de manera informal, corresponde a la verificacin de la ausencia de interbloqueos del MTPI.

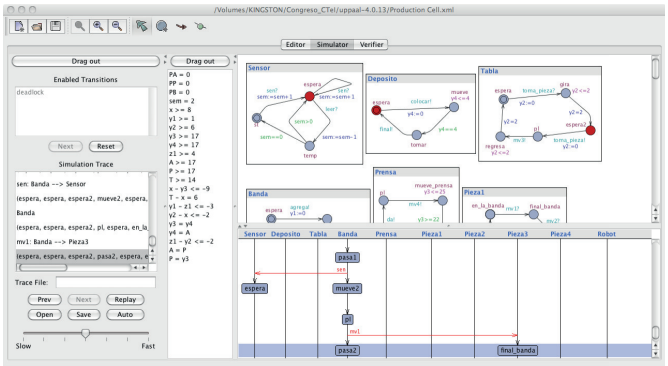

Figura 7: Captura de Pantalla del Simulador de UPPAAL para la Celda de Producción

En la Figura 8 se presenta la pantalla del verificador de la herramienta UPPAAL, donde se muestra la verificación de algunas propiedades del MPTI. Estas propiedades se especifican con una variación de la lógica temporal TCTL [3].

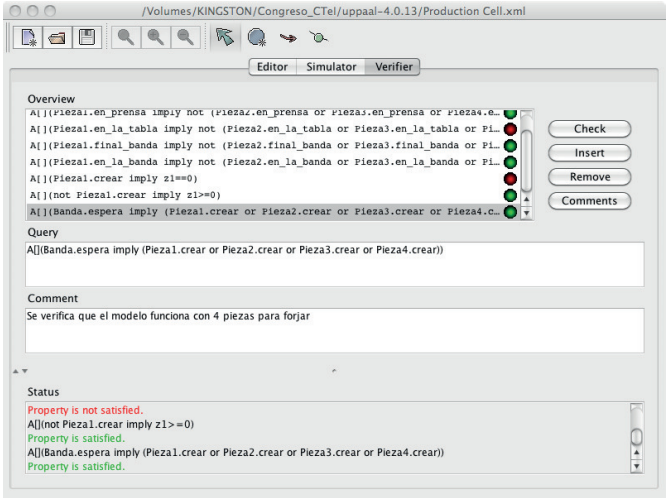

Figura 8: Captura de Pantalla del Verificador de UPPAAL para la Celda de Producción

Tal como se puede ver en la Figura 8, aquellas propiedades que aparecen con una señal verde en el lado derecho han sido verificadas con éxito; mientras que las que aparecen con la señal roja no han sido satisfechas por el modelo presentado en la Figura 7. Por ejemplo, UPPAAL indica que la siguiente propiedad se cumple (ver última propiedad en la lista con señal verde):

```
A[](Banda.espera imply (Pieza1.crear
or Pieza2.crear or Pieza3.crear or
Pieza4.crear)) ,
```

ya que se encontró que un mínimo de 4 piezas es necesario para mantener el modelo trabajando para analizar su comportamiento. Si se trabaja sólo con tres piezas, la Celda de Producción no se mantiene ocupada. Para asegurar que las piezas no se pueden adelantar entre sí, es necesario demostrar que sus ubicaciones están ocupadas bajo exclusión mutua. Es decir, si la Pieza1 está en el estado `en_tabla`, entonces otra pieza no puede estar en ese mismo estado. La excepción de esta regla es, por supuesto, estar en el estado `crear`. El resultado de verificación de estas propiedades son las que aparecen en la parte superior de la propiedad anterior.

La propiedad de vivacidad que se indicó al principio de esta sección, está representada por la siguiente fórmula de TCTL: `A[](not Pieza1.crear imply z1 <= 200) .`

Para este tipo de propiedad (acotada) de vivacidad, es posible reducir el límite superior hasta que la afirmación se convierta en falsa. El contra–ejemplo contrario puede ser observado con la condición:

`A[](not Pieza1.crear imply z1 < I97) ,`

que falla y que el simulador muestra la razón. En esta situación, tres piezas están delante la Pieza 1 y dos están por detrás, lo cual implica que hay 5 piezas para mantener trabajando el modelo bajo esta condición; pero anteriormente establecimos que trabajamos con 4 piezas, lo cual genera una contradicción.

VI. CONCLUSIONES Y TRABAJO FUTURO

En este trabajo hemos presentado la integración de un EFV con la teoría de AT para la especificación y la verificación automática del MTPI. El uso del EFV con la verificación automática y los AT ha resultado en una infraestructura metodológica que garantiza la exactitud del MTPI con respecto a la especificación de las propiedades iniciales derivadas de las buenas prácticas o reglas de una industria. También se han propuesto un conjunto sencillo de lineamientos para la transformación de modelos de PIC a redes de AT. Este mapeo es la primera etapa necesaria para realizar la evaluación (es decir, el análisis cualitativo) de un PIC.

Se ha mostrado la viabilidad de la propuesta, a través de incorporación de la herramienta UPPAAL en la verificación de la Celda de Producción, un ejemplo de carácter académico para evaluar metodologías para el diseño de sistemas y procesos industriales. De esta manera, se ha ilustrado cómo integrar la verificación automática en las etapas tempranas del diseño de PIC. Finalmente, se la logrado demostrar que es posible soportar al ingeniero de modelado de procesos con métodos, modelos y herramientas que le permitan verificar los modelos de los PIC desde el mismo momento que los diseña.

Como trabajo futuro, se estima la conformación de las reglas de transformación que permitan concebir una herramienta de software para automatizar la obtención de los AT que reflejan fielmente los modelos de PIC realizados en alguno de los lenguajes gráficos no formales que normalmente se utilizan en la industria.

AGRADECIMIENTOS

Esta investigación fue parcialmente soportada por el Fondo Nacional de Ciencia, Tecnología e Innovación (FONACIT), mediante el proyecto G–2005000165.

REFERENCIAS

[1] P. Rodríguez, R. Alonso, and J. Snchez, *¿Cuál es la Madurez que Necesitarían los Procesos para el Desarrollo de Sistemas de Software Crítico?*, Revista Española de Innovación, Calidad e Ingeniería del Software, vol. 1, no. 2, pp. 31–41, 2005.

[2] A. Burns and A. Wellings, *Real–Time Systems and Programming Languages*, 3rd edition, Addison–Wesley Longman Publishing Co., Inc., 2001.

[3] G. Behrmann, A. David, and K. Larsen, *A Tutorial on* UPPAAL, Lecture Notes in Computer Science (LNCS), vol. 3185, pp. 200–236, 2004.

[4] S. Morimoto, *A Survey of Formal Verification for Business Process Modeling*, Lecture Notes in Computer Science (LNCS), vol. 5102, pp. 514–522, 2008.

[5] W. Yeung, K. Leung, J. Wang, and W. Dong, *Modelling and Model Checking Suspendible Business Processes Via Statechart Diagrams and CSP*, Science of Computer Programming, special Issue on: Increasing Adequacy and Reliability of EIS, vol. 65, no. 1, pp. 14–29, 2007.

[6] L. Mendoza, M. Capel, M. Pérez, and K. Benghazi, *Compositional Model–Checking Verification of Critical System*, Lecture Notes in Business Information Processing (LNBIP), vol. 19, pp. 213–225, 2009.

[7] M. Capel, L. Mendoza, and K. Benghazi, *Automatic Verification of Business Process Integrity*, International Journal of Simulation and Process Modelling, vol. 4, no. 3/4, pp. 167–182, 2008.

[8] L. Fischer, *2009 BPM and Workflow Handbook: Spotlight on Government*, BPM Handbook Series, Future Strategies, 2009.

[9] W. Kettinger, J. Teng, and S. Guha, *The Process Reengineering Life Cycle Methodology: A Case Study*, in Business Process Change: Reengineering, Concepts, Methods and Technologies, Idea Group Pub., pp. 211–244, 1998.

[10] M. Mijares and L. Mendoza, *Conceptual Model for the Specification of the Quality Properties of the Critical Business Process*, in proceedings of the 8th Iberian Conference on Information Systems and Technologies (CISTI 2013), vol. 1, pp. 315–322, Silvares, Portugal, June 2013.

[11] L. Meade and K. Rogers, *Selecting Critical Business Processes: A Case Study*, Engineering Management Journal, vol. 13, no. 4, pp. 41–46, 2001.

[12] ISO/IEC JTC 1/SC 7, *ISO/IEC 25000:2014 Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE) – Guide to SQuaRE*, ser. 35.080: Software, International Organization for Standardization, 2014.

[13] ISO/TC 176/SC 1, *ISO 9000:2005 Quality Management Systems – Fundamentals and Vocabulary*, ser. ISO 9000 Quality Management, International Organization for Standardization, 2005.

[14] R. Pressman, *Software Engineering: A Practitioners Approach*, 7th edition, McGraw–Hill, 2009.

[15] C. Baier and J.-P. Katoen, *Principles of Model Checking*, Cambridge, The MIT Press, 2008.

[16] J. Hopcroft, R. Motwani, and J. Ullman, *Introduction to Automata Theory, Languages, and Computation*, 3rd edition, Prentice Hall, 2006.

[17] R. Alur and D. Dill, *A Theory of Timed Automata*, Theoretical Computer Science, vol. 126, no. 2, pp. 183–235, 1994.

[18] R. Milner, *A Calculus of Communicating Systems*, Lecture Notes in Computer Science (LNCS), vol. 92, 1980.

[19] C. Hoare, *Communicating Sequential Processes*, International Series in Computer Science, Prentice–Hall International Ltd., 1985.

[20] E. Clarke, O. Grumberg, and D. Peled, *Model Checking*, The MIT Press, 2000.

[21] C. Duursma and U. Olle, *Task Model Definition and Task Analysis Process*, Technical Report, Brussels KADSII /M5/VUB/RR/004/2.0., Vrije University, 1994.

[22] W. Aalst, *Challenges in Business Process Analysis*, Lecture Notes in Business Information Processing (LNBIP), vol. 12, pp. 27–42, 2009.

[23] M. Weske, *Business Process Management: Concepts, Languages, Architectures*, Springer Berlin Heidelberg, 2007.

[24] C. Ouyang, E. Verbeek, W. Aalst, S. Breutel, M. Dumas, and A. Hofstede, *Formal Semantics and Analysis of Control Flow in WS–BPEL*, Science of Computer Programming, vol. 67, no. 2–3, pp. 162–198, 2007.

[25] M. Ortín, J. García, B. Moros, and J. Nicolás, *El Modelo de Negocio como Base del Modelo de Requisitos*, in Actas de las Jornadas de Ingeniería de Requisitos Aplicada, Sevilla, Spain, 2009.

[26] J. Lilius and I. Porres, *The Production Cell: An Exercise in the Formal Verification of a UML Model*, Technical Report, vol. 288, Turku Centre for Computer Science, 1999.

[27] A. Burns, *How to Verify a Safe Real–Time System: The Application of Model Checking and Timed Automata to the Production Cell Case Study*, Real–Time Systems, vol. 24, no. 2, pp. 135–151, 2003.