Una Propuesta de Editor Gráfico para el Modelado y la Generación de Código de Patrones de Eventos sobre Drones

Juan Boubeta-Puig[1], Juan Hernández[2], Enrique Moguel[2], Juan Carlos Preciado[2], Fernando Sánchez-Figueroa[2]

[1] Dpto. de Ingeniería Informática, Universidad de Cádiz
juan.boubeta@uca.es
[2] Dpto. de Ingeniería de Sistemas Informáticos y Telemáticos, Universidad de Extremadura
{juanher,enrique,jcpreciado,fernando}@unex.es

Resumen. Los lenguajes de procesamiento de eventos (EPL) permiten declarar e implementar patrones de eventos que son procesados posteriormente por motores de procesamiento de eventos complejos (CEP) y así poder detectar situaciones de interés del usuario en tiempo real. Para llevar a cabo esta tarea, el usuario debe tener un alto grado de experiencia en estos lenguajes. Sin embargo, y en el ámbito de los drones, los usuarios suelen tener un vasto conocimiento en el dominio para el que se necesitan definir ciertos patrones de eventos (motores, dispositivos de navegación, pilotos automáticos, etc.), pero que son inexpertos tanto en EPLs como en el lenguaje requerido para implementar las acciones a llevar a cabo en el dron tras la detección de los eventos. En este artículo presentamos un editor de modelado de patrones con el propósito de facilitar a los usuarios finales un entorno amigable e intuitivo con el que poder definir gráficamente las situaciones críticas y relevantes que se requieran detectar en los drones, y sin necesidad de conocer ningún lenguaje de programación en particular. Además, este editor transforma estos modelos gráficos de patrones al código que los implementa.

Keywords: CEP, EPL, MDD, Editor de Modelado Gráfico, Dron.

1 Introducción y Motivación

El procesamiento de eventos complejos (*Complex Event Processing*, CEP) [1] es una tecnología emergente que permite procesar, analizar y correlacionar ingentes volúmenes de datos con el fin de detectar en tiempo real situaciones críticas o relevantes para un dominio en particular.

Los lenguajes de procesamiento de eventos (*Event Processing Languages*, EPL) permiten declarar e implementar los patrones de eventos que son procesados posteriormente por motores CEP y así poder detectar situaciones de interés del usuario en tiempo real. Para llevar a cabo esta tarea, el usuario debe tener un alto grado de experiencia en estos lenguajes. Sin embargo, los usuarios suelen tener un vasto conoci-

miento en el dominio para el que se necesitan definir ciertos patrones de eventos, pero que son inexpertos tanto en EPLs como en el lenguaje requerido para implementar las acciones a llevar a cabo tras la detección de los eventos.

Un estudio de la empresa ebizQ [2] concluye que un 84% de los encuestados consideran que la definición de estos patrones debe realizarse por expertos en el dominio, que son los que realmente disponen de todo el conocimiento necesario para ello, frente a un 16% que prefieren que sean programadores. El estudio realizado al amparo de la tesis doctoral de Boubeta-Puig [3] también corrobora esta necesidad.

Dada la cantidad de EPLs existentes, Boubeta-Puig et al. han propuesto un lenguaje de modelado específico de dominio (*Domain-Specific Modeling Language*, DSML), denominado ModeL4CEP [4], con el fin de definir un lenguaje común para que cualquier usuario pueda describir fácilmente un patrón, independientemente de su implementación. Gracias a las técnicas de desarrollo de software dirigido por modelos, este patrón definido como modelo podrá transformarse posteriormente a código de distintos EPLs con las consecuentes ventajas: oculta los aspectos técnicos de los EPLs a los usuarios finales, y la productividad mejora puesto que los modelos son más fáciles de mantener y el código generado automáticamente estará libre de errores.

Además, con objeto de acercar esta tecnología a cualquier usuario, eliminando las barreras existentes en cuanto a la programación requerida, Boubeta-Puig et al. proponen MEdit4CEP [5], una solución compuesta principalmente por un enfoque dirigido por modelos [6] y un editor gráfico e intuitivo para el modelado de patrones de eventos, así como su transformación automática tanto al código EPL a desplegar en el motor CEP, como al código de las acciones a ejecutar tras la detección de los mismos.

Una de las principales ventajas de este editor es que puede reconfigurarse mediante un modelo gráfico de dominio CEP –definido por un conjunto de tipos de eventos con sus propiedades, conforme al DSML de dominios CEP denominado ModeL4CEP. Así, el usuario dispondrá en todo momento de una interfaz gráfica común adaptada al contexto específico para el que desee definir los patrones, modificándose dinámicamente la paleta de herramientas dependiendo del dominio en cuestión. Sin embargo, las únicas herramientas sobre acciones a ejecutar tras la detección de patrones, actualmente proporcionadas por el editor, son el envío de eventos por *email* o Twitter.

En este artículo presentamos una extensión de MEdit4CEP para facilitar a los usuarios finales un entorno amigable e intuitivo con el que poder definir gráficamente las situaciones críticas y relevantes que se requieran detectar en los drones [7], y sin necesidad de conocer ningún lenguaje de programación en particular. Además, este editor transforma estos modelos de patrones al código que los implementa.

El resto del artículo se estructura de la siguiente forma. En la Sección 2 se propone el editor de modelado gráfico de patrones de eventos sobre drones y en la Sección 3 se presentan las conclusiones y el trabajo futuro.

2 Editor Gráfico Propuesto

En esta sección, se describe brevemente cómo se ha llevado a cabo la extensión de MEdit4CEP, implementado con *Graphical Modeling Framework* (GMF) y el proyec-

to Epsilon [8], para hacer posible el modelado y la generación de código de patrones de eventos sobre situaciones a detectar en los drones.

Por un lado, se ha extendido el metamodelo de patrones de eventos ModeL4CEP, especializando la metaclase *Actions* en las metaclases *ReturnToHome*, *Land* y *UAVConfiguration*. Esto soporta la definición de patrones asociados a acciones específicas a ejecutar en el propio dron: el regreso a casa (*ReturnToHome*) o el aterrizaje (*Land*); aparte de las acciones ya contempladas (el envío de eventos a una cuenta de correo o Twitter). Cabe destacar que la metaclase *UAVConfiguration* tiene dos atributos: *port* y *mode*, para indicar el número de puerto y el modo de conexión (telemetría o inalámbrico) a establecer con el dron. Además, se ha modificado la paleta de herramientas del editor, haciendo uso de *Epsilon Object Language* (EOL), para añadir dichas acciones específicas de drones como herramientas en la categoría *Actions*.

Por otro lado, se han creado nuevas reglas de validación en *Epsilon Validation Language* (EVL) que permiten validar si el patrón de eventos modelado, concretamente las acciones para drones modeladas, es conforme a su metamodelo. Por ejemplo, si el número de puerto indicado es correcto para el modo de conexión elegido.

También se han construido nuevas reglas en *Epsilon Generation Language* (EGL) para la transformación de las acciones de drones modeladas al código Python, que deberá desplegarse en el dron en cuestión para ser ejecutadas. El dron utilizado tiene un piloto automático ArduPilotMega, ya que se basa en el protocolo abierto MAVLink. La gran mayoría de drones existentes en el mercado están basados en dicho protocolo facilitando en gran medida el desarrollo de aplicaciones como la que se presenta en este artículo.

En la Figura 1 se muestra un ejemplo de patrón de eventos, modelado con el editor propuesto, que comprueba para cada evento sonoro medido por el dron si el nivel de batería es inferior al 30% de su capacidad máxima. En caso afirmativo, se enviará la orden al dron para que regrese a casa y, además, se notificará esta alerta a los usuarios interesados a través del correo electrónico.

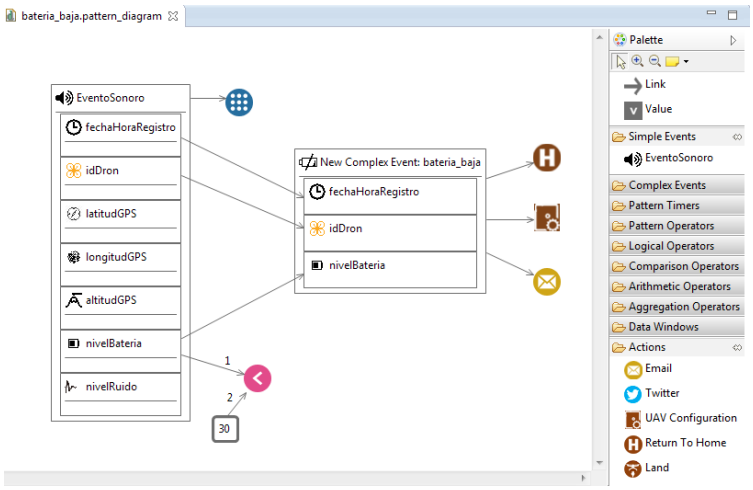

Figura 1. Patrón de eventos que detecta el nivel de batería baja en drones.

3 Conclusiones y Trabajo Futuro

En este trabajo hemos propuesto una extensión del editor gráfico de modelado MEdit4CEP, basado en ModeL4CEP, para unificar la descripción de patrones de eventos en el ámbito de los drones, representándolos como modelos; facilitando así al usuario final la definición de estos patrones sin necesidad de conocer detalles específicos sobre ningún EPL ni sobre ningún lenguaje requerido para implementar las acciones asociadas a estos patrones como, por ejemplo, el regreso del dron a casa o su aterrizaje. Además, este editor valida los patrones modelados, genera el código que los implementa, y permite importarlos y exportarlos.

Como trabajo futuro, pretendemos extender la paleta del editor gráfico, añadiendo nuevos tipos de acciones que podrán usarse durante el modelado de patrones de eventos en el ámbito de los drones, con el propósito de hacer posible la definición de situaciones críticas o relevantes que desencadenen otras acciones, aparte del regreso del dron a casa y su aterrizaje.

Agradecimientos. Este trabajo ha sido parcialmente financiado por el Ministerio de Economía y Competitividad (proyectos TIN2014-53555-REDT, TIN2015-65845-C3-3-R, TIN2015-6957-R), la Junta de Extremadura (GR15098) y los Fondos FEDER. Boubeta-Puig agradece la hospitalidad recibida durante su estancia de investigación con el Grupo Quercus de Ingeniería del Software de la Universidad de Extremadura, donde parte de este trabajo ha sido desarrollado.

Referencias

[1] D. Luckham, *Event Processing for Business: Organizing the Real-Time Enterprise*. New Jersey, USA: Wiley, 2012.

[2] BEA, "Event Processing Market Pulse 2007," ebizQ, 2007.

[3] J. Boubeta-Puig, "Desarrollo Dirigido por Modelos de Interfaces Específicas de Dominio para el Procesamiento de Eventos Complejos en Arquitecturas Orientadas a Servicios," Tesis Doctoral. Universidad de Cádiz, Cádiz, España, 2014.

[4] J. Boubeta-Puig, G. Ortiz, e I. Medina-Bulo, "ModeL4CEP: Graphical domain-specific modeling languages for CEP domains and event patterns," *Expert Systems with Applications*, vol. 42, no. 21, pp. 8095–8110, Nov. 2015.

[5] J. Boubeta-Puig, G. Ortiz, e I. Medina-Bulo, "MEdit4CEP: A model-driven solution for real-time decision making in SOA 2.0," *Knowledge-Based Systems*, vol. 89, pp. 97–112, Nov. 2015.

[6] J. Boubeta-Puig, G. Ortiz, e I. Medina-Bulo, "A Model-driven Approach for Facilitating User-friendly Design of Complex Event Patterns," *Expert Systems with Applications*, vol. 41, no. 2, pp. 445–456, Feb. 2014.

[7] Federal Aviation Administration, "Small UAS Notice of Proposed Rulemaking (NPRM)," 2015. https://www.faa.gov/uas/nprm/ [Accedido: 23/04/2016]

[8] Eclipse Foundation, "Epsilon," 2016. http://www.eclipse.org/epsilon/ [Accedido: 23/04/2016]