Sagardui, G., Etxeberria, L., Agirre, J. A., Arrieta, A., Nicolas, C. F., Martin, J. M.: Entorno de Validación Configurable para Software Embebido Refactorizado y su Aplicación en Ascensores. In: Sánchez-Figueroa, F. (ed.) Actas de las XXIII Jornadas de Ingeniería del Software y Bases de Datos (JISBD 2018). Sistedes (2018). https://hdl.handle.net/11705/JISBD/2018/081

Entorno de Validación Configurable para Software Embebido Refactorizado y su Aplicación en Ascensores

Goiuria Sagardui[1], Leire Etxeberria[1], Joseba A. Agirre[1], Aitor Arrieta[1], Carlos Fernando Nicolás[2], and Jose María Martín[3]

[1] Mondragon Goi Eskola Politeknikoa
{gsagardui, letxeberria, jaagirre, aarrieta}@mondragon.edu
[2] Ik4- Ikerlan
cfnicolas@ikerlan.es
[3] Orona EIC
jmmartinc@orona-group.com

Abstract. Los ascensores son sistemas complejos que integran además de software, componentes eléctricos, mecánicos, etc. La complejidad de estos sistemas aumenta además al tener en cuenta la variabilidad: un ascensor se puede instalar en edificios de más o menos pisos, puede tener diferentes número y tipo puertas, los actuadores y sensores pueden variar, etc. La validación del software de estos sistemas es compleja. Este artículo presenta un trabajo industrial para la validación de software embebido configurable refactorizado en el contexto del sector del transporte vertical.

1 Introducción y Problema a Resolver

Los ascensores son sistemas complejos que integran diferentes campos de ingeniería, como por ejemplo, la ingeniería electrónica, ingeniería mecánica, ingeniería de control, etc. Dichos sistemas son dependientes del edificio donde los ascensores van a ser instalados, lo que requiere alta variabilidad. Por ejemplo, se tiene que considerar el número de pisos del edificio, el número de ascensores que tendrá el edificio, si todos los ascensores pueden atender todos los pisos, el número de puertas de cada ascensor (y en qué piso se abre qué puerta), tipos de sensores y actuadores, etc. En este artículo, nos centramos en el control de la puerta del ascensor y su validación.

El principal problema del software de control de la puerta es que se ha de considerar variabilidad debido a diferentes factores: (1) variabilidad del sistema dependiendo del entorno de aplicación (como se ha explicado en el párrafo previo), (2) cambios de legislación, (3) diferentes estándares dependiendo del país donde el ascensor será instalado, (4) nuevas funcionalidades y (5) adaptación constante del software para ser embebido en nuevos procesadores [3]. Por ello, Orona, empresa líder en el sector de transporte vertical, refactorizó el software de control de la puerta por las siguientes razones: (1) reducir la complejidad,

Sagardui, G., Etxeberria, L., Agirre, J. A., Arrieta, A., Nicolas, C. F., Martin, J. M.: Entorno de Validación Configurable para Software Embebido Refactorizado y su Aplicación en Ascensores. In: Sánchez-Figueroa, F. (ed.) Actas de las XXIII Jornadas de Ingeniería del Software y Bases de Datos (JISBD 2018). Sistedes (2018). https://hdl.handle.net/11705/JISBD/2018/081

(2) aumentar la capa de abstracción, (3) mejorar la comprensión y (4) mejorar la mantenibilidad. Este artículo explica la estrategia utilizada para el testeo y validación del código refactorizado.

En la Sección 2 se presenta la propuesta para la validación del software configurable refactorizado. La sección 3 presenta los resultados obtenidos y las lecciones aprendidas. Por último concluimos el artículo en la sección 4.

2 Entorno de Validación Configurable

La propuesta presentada para la validación del software configurable del control de puertas está basada en tres partes diferentes, como puede verse en la Figura 1: (1) Generación de casos de test, (2) gestión de la variabilidad y (3) generación del entorno de validación. En el caso de la generación de casos de test, se ha utilizado la técnica de "Testeo Basado en Modelos". En este caso, se realiza una máquina de estados que modela el comportamiento del sistema. Dicha máquina de estados también modela aspectos relacionados con el tiempo. Después, se generan los casos de test haciendo uso de la herramienta ModelJUnit y el algoritmo "GreedyTester". Una vez que se generan los casos de test abstractos, se tienen que generar los casos de test ejecutables. Los casos de test deben de ser ejecutables en Simulink ya que se utiliza esta herramienta para simular la puerta. Para ello, se ha desarrollado un módulo que transforma casos de test abstractos en señales de Simulink.

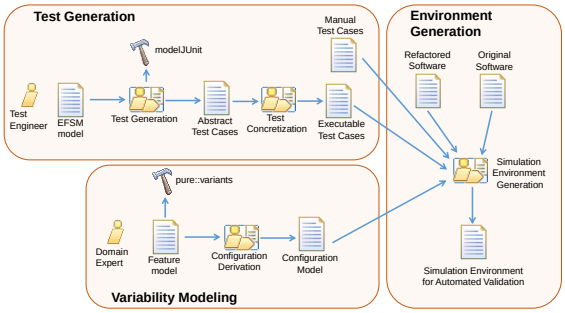

Fig. 1: Proceso para la generación del entorno de simulación para la validacin automática de software embebido refactorizado [3]

La segunda parte corresponde a la gestión de la variabilidad, en la que se hace uso de modelos de características con cardinalidad. El modelo de características incluye los diferentes elementos que se han de considerar al validar el sistema de control así como el resto del sistema (por ejemplo, sensores, elementos de movimientos horizontales, etc.). La herramienta utilizada ha sido pure::variants [2], ya que permite (1) la gestión de variabilidad con cardinalidad y (2) integración con modelos Simulink. La figura 2 muestra una parte de este modelo

Sagardui, G., Etxeberria, L., Agirre, J. A., Arrieta, A., Nicolas, C. F., Martin, J. M.: Entorno de Validación Configurable para Software Embebido Refactorizado y su Aplicación en Ascensores. In: Sánchez-Figueroa, F. (ed.) Actas de las XXIII Jornadas de Ingeniería del Software y Bases de Datos (JISBD 2018). Sistedes (2018). https://hdl.handle.net/11705/JISBD/2018/081

de características. A partir de este modelo de características es posible generar productos válidos del sistema.

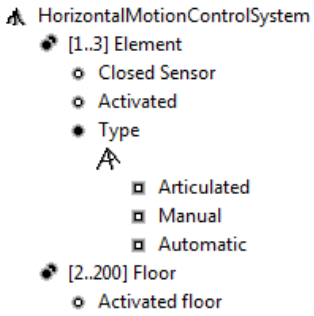

Fig. 2: Detalle del modelo características del sistema[3]

La última parte consiste en la generación automática del entorno de validación. Como el sistema contiene variabilidad, el entorno de validación tiene que ser configurable. El software de control antiguo se utiliza como oráculo del sistema (es decir, el evaluador que determina si un caso de test es correcto o no), ya que ha sido previamente testeado y está ejecutándose en producto. Por lo tanto se comparan las salidas de los dos sistemas (es decir, el refactorizado y el que está en producció) para determinar si el comportamiento del sistema refactorizado es el esperado. Para la generación del entorno de validación se tienen en cuenta (1) el software original, (2) el software refactorizado, (3) el fichero de la configuración del producto derivado a partir del modelo de características y (4) los casos de test ejecutables. La generación de este entorno es automática y consiste en los siguientes pasos: Primero, se parsea el modelo de configuración, el cual es un fichero xml. Con este fichero, se configuran el software original y el refactorizado. Después, se crea un nuevo fichero Simulink y los dos códigos de control de puertas se despliegan en forma de S-Function (código ejecutable en Simulink). Además, el generador del entorno de validación despliega los elementos mecánicos y eléctricos que simulan la parte física de la puerta. Como último paso, los casos de test ejecutables se crean, resuelve su variabilidad y se despliegan e integran con el resto de componentes en el fichero Simulink. Después de estos pasos, el modelo está preparado para ejecutarse y testearse.

3 Resultados y Lecciones Aprendidas

La herramienta desarrollada integra métodos para la generación automática de casos de test con herramientas para la gestión de variabilidad y simulación. Todo ello ha permitido automatizar el proceso de ejecución de casos de test para diferentes configuraciones del sistema. El uso de la versión antigua del controlador ha permitido detectar errores de manera automática. Por otra parte, de este estudio se han obtenido las siguientes lecciones aprendidas:

Sagardui, G., Etxeberria, L., Agirre, J. A., Arrieta, A., Nicolas, C. F., Martin, J. M.: Entorno de Validación Configurable para Software Embebido Refactorizado y su Aplicación en Ascensores. In: Sánchez-Figueroa, F. (ed.) Actas de las XXIII Jornadas de Ingeniería del Software y Bases de Datos (JISBD 2018). Sistedes (2018). https://hdl.handle.net/11705/JISBD/2018/081

- *Importancia de uso de las herramientas de simulación:* El uso de herramientas de simulación a la hora de testear software ha sido muy importante. El despliegue del software en un sistema real es extremadamente caro, especialmente en este sector. El testeo basado en simulación ha permitido numerosas ventajas a la hora de testear código embebido refactorizado. Primero, permite la ejecución de test suites largos. Segundo, permite testear diferentes configuraciones (con diferentes características, diferentes parámetros, etc.). Tercero, facilita la automatización a la hora de testear un sistema. Por último, permite testear funcionalidad críticas inyectando faltas sin riesgos ni daños (ya sean económicos o humanos).
- *Utilidad de la versión antigua del software de control:* Al refactorizar código, el código original (es decir, el no refactorizado) se testeó exahustivamente cuando se desarrolló y está funcionando en producción. Esto significa que muchos errores se han corregido y por lo tanto, se asume que su comportamiento es adecuado. Esto permite comparar el código refactorizado con el código original y así automatizar la validación. De tal manera, es posible comparar las salidas de ambos sistemas e informar a los expertos de dominio si los resultados no son iguales.
- *Importancia de la gestión de la variabilidad:* El uso de herramientas para la gestión de la variabilidad desde las primeras etapas de refactorización permiten diferentes ventajas. Una de las ventajas es la generación automática del entorno de validación, tal y como hemos propuesto en este artículo y en otros [1]. Otra ventaja importante es la derivación sistemática de configuraciones a testear tanto automática como manualmente. Además, en este estudio se ha incorporado variabilidad en casos de test, que también puede ser gestionada con modelos de características.

4 Conclusiones

En este artículo se presenta el trabajo llevado a cabo para la validación automática de un controlador de puertas de ascensor. Dicho controlador es complejo y con variabilidad, por lo tanto se han integrado técnicas de testeo basado en modelos con técnicas de gestión de la variabilidad. Los resultados han sido positivos, permitiendo la automatización de una parte amplia de verificación y validación.

References

1. Arrieta, A., Sagardui, G., Etxeberria, L., Zander, J.: Automatic generation of test system instances for configurable cyber-physical systems. Software Quality Journal 25(3), 1041–1083 (2017)
2. Pure-Systems: pure::variants. http://www.pure-systems.com (July 2014)
3. Sagardui, G., Etxeberria, L., Agirre, J.A., Arrieta, A., Nicolas, C.F., Martin, J.M.: A configurable validation environment for refactored embedded software: An application to the vertical transport domain. In: 2017 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW). pp. 16–19. IEEE (2017)