II ENCUENTRO DE JÓVENES INVESTIGADORES

"UCEFlow – Sintaxis para estructurar el Flujo de Eventos de Casos de Uso"

Sabrina Cruz Introini

Becario Estudiante Avanzado – CICITCA

Instituto de Informática

FCEFyN

1 ANTECEDENTES Y FUNDAMENTACIÓN

El grupo de investigación trabaja en esta temática desde hace algunos años, en un principio se desarrolló una técnica que identificó indicadores que permiten evaluar el diseño de un sistema, a través del modelo de CU, basándose en documentación mínima que deben contener las descripciones de los CU [1].

Luego, debido a la necesidad de una plantilla común para documentar CU, se elaboró la plantilla CUPIDo, cuyo proceso de generación y detalle de la misma fue publicado en [2] y se realizó una validación estadística [3]. CUPIDo se basa e integra otros formatos para documentar CU y se adapta a necesidades y ámbitos de desarrollo de software local y/o regional, en el contexto de los sistemas administrativos, donde la formalización y documentación de sistemas es escasa o nula o, si existe, es netamente informal.

Por su parte, I.Díaz y A. Matteo [4] proponen directrices para disminuir el impacto de la utilización de lenguaje natural en la especificación de CU, debido a los diferentes estilos de redacción de la lengua española. Plantean un patrón o guía para redactar una descripción o escenario de CU, estableciendo algunas restricciones, controlando los múltiples estilos de redacción y la variedad de términos característicos del idioma.

C. Rolland, et. al. [5] exponen un framework que captura características para describir los escenarios de CU desde dos perspectivas: la descripción en lenguaje natural y la presentación en prototipos, gráficos, etc. Aplican un marco en doce enfoques y muestran que todos comparten algunas de las propiedades que caracterizan a los escenarios. Los escenarios son las descripciones concretas de las situaciones o conductas expresadas, la mayoría de las veces, a través de textos en lenguaje natural.

Más tarde I. Díaz, et. al. [6] extienden el trabajo citado anteriormente y proponen un patrón para la especificación de CU, incluyendo meta-información relacionada. Establecen nueve grados de refinamiento en un CU, cada uno expresa distintos niveles de abstracción. Por último, presentan el metamodelo que restringe los elementos conceptuales relacionados con el marco de la especificación propuesta.

Todas estas propuestas son apropiadas y útiles, pero ninguna plantea estructurar, a priori, la escritura del flujo de eventos a través de una sintaxis definida para tal fin.

Un flujo de eventos (FE) [7], también llamado escenario, es la parte principal de la plantilla de CU, por lo que merece especial cuidado y atención. Consiste en la declaración de los pasos que sucesivamente ocurren en la interacción entre los actores y el sistema, en el contexto de un CU. Está compuesto por una serie de sentencias escritas en una sola forma gramatical y donde cada oración es un paso, a través del cual un actor llega al resultado esperado y poseen un orden cronológico. Un buen paso ha de ser siempre una oración con sentido, escrita en voz activa, iniciada con el nombre del actor o bien del objeto del sistema que realiza la acción e indicando exhaustivamente los elementos de información que se intercambian.

Motiva esta investigación la dificultad en la construcción del FE evidenciada en alumnos de 4º año de la materia "Diseño de Software" de las carreras de Informática de la FCEFN de la UNSJ. En esta materia los alumnos utilizan, desde el 2005, plantillas para describir los CU de los sistemas que modelan y desde el 2010 se usa la plantilla CUPIDo. La experiencia de todos estos años demuestra que, al momento de completar las plantillas, los estudiantes expresan confusión en el llenado del campo referido al FE y corroborado a través de encuestas. Se planteó la hipótesis de que puede ser debido a la falta de estructuración y guía para completar el FE. Por este motivo se trabajó en una sintaxis, a la que se denominó "UCEFlow", que estructura el FE aplicando reglas gramaticales que guían y formalizan su escritura.

Este trabajo se enmarca en el proyecto de investigación "Formalización de descripciones de Casos de Uso a través de metamodelos. ForCUPIDo" (Proy Nº E-883), de la FCEFN de la UNSJ (Argentina).

2 OBJETIVOS

2.1 Objetivo general

El objetivo del presente trabajo es presentar una sintaxis para estructurar la redacción del Flujo de Eventos de Casos de Uso y la validar la misma mediante un análisis sintáctico, con un parser desarrollado en lenguaje Java.

2.2 Objetivos específicos

- Identificar elementos comunes en un conjunto acotado de escenarios de Casos de Usos.

- Definir la sintaxis denominada UCEFlow a partir de los elementos identificados.
- Especificar la sintaxis UCEFlow a través de la Notación de Backus-Naur extendida.
- Definir el modelo conceptual mediante un diagrama de clases.
- Validar la propuesta a través de un analizador sintáctico (parser) desarrollado en Java.

3 DESARROLLO DE LA PROPUESTA

Se estructuró el FE, a través de reglas gramaticales que guían y formalizan su escritura, por medio de una sintaxis denominada UCEFlow (Estructuración del Flujo de Eventos de Casos de Uso, se lee "iu si i flou", que sería como decir "you see e-flow" o tú ves el flujo de eventos) [8]. Esto contribuirá a la construcción semiautomática de los diagramas de secuencia y por medio de transformaciones automáticas obtener la prototipación del sistema, fundamentándose en la teoría de Desarrollo de Software Dirigido por Modelos (MDD) [9]. Por otro lado, se proyecta integrar el FE estructurado a la plantilla CUPIDo sistematizada.

En la Fig. 1 Se muestra el modelo conceptual de la sintaxis propuesta. Se puede observar que el FE es un conjunto de pasos, estos pueden ser pasos de camino base o pasos de camino alternativo. A su vez cada paso puede ser una sentencia simple, una sentencia especial, o una llamada a "include". En el camino alternativo se puede especificar el número del paso al que retorna. Las sentencias simples se componen de un sujeto (actor) y un predicado. Este último puede contener o no un destinatario, debe tener una acción y un complemento el cual podrá ser almacenado como objeto del dominio. En cuanto a las sentencias especiales deben tener una condición y pueden tener más de un paso. Este modelo preliminar está sujeto a posibles mejoras según se vaya validando la sintaxis con más ejemplos de FE de CU reales y de variados contextos.

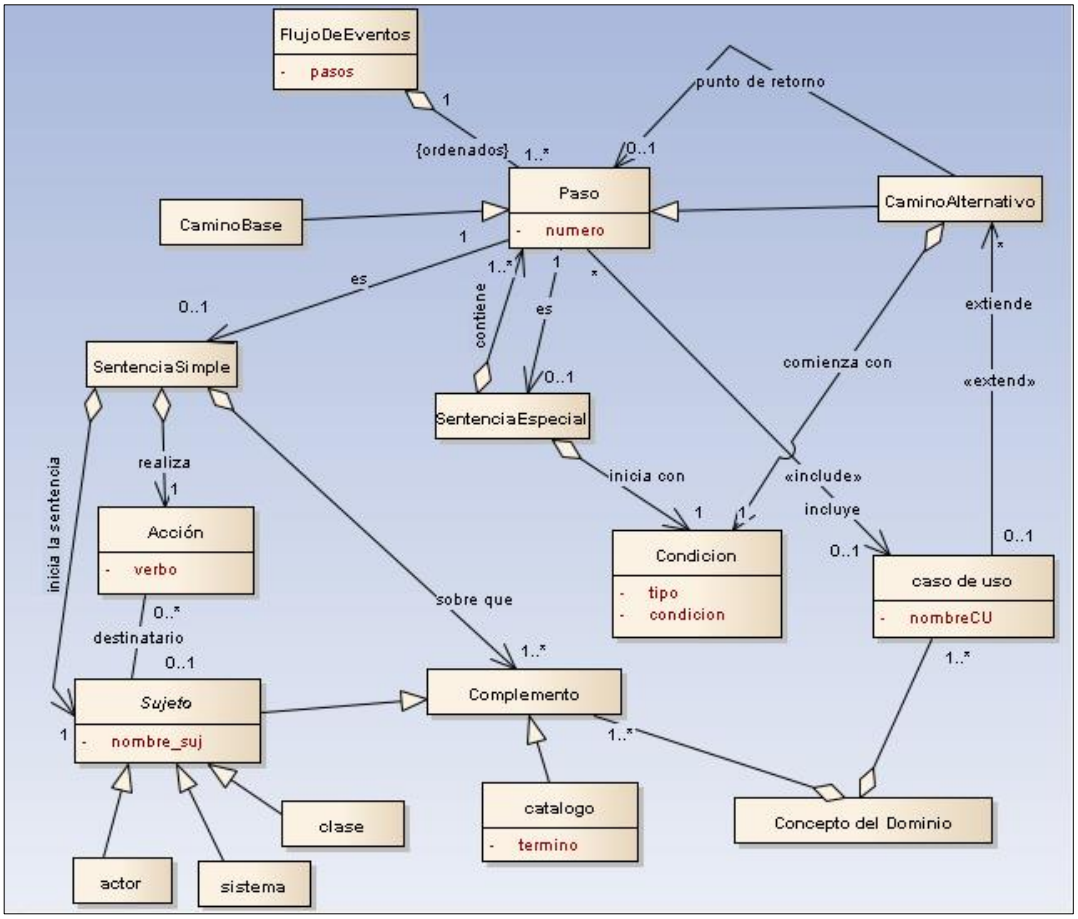

Fig. 1 – Modelo de la sintaxis de UCEFlow.

3.1 Sintaxis del Lenguaje

La sintaxis se planteó en EBNF [10] (Extended Backus-Naur Form) y comprende conceptos relevantes a un FE.

Los símbolos terminales se muestran entre corchetes angulares (por ejemplo, <code>), los símbolos no terminales en cursiva (por ejemplo, camino base); las sentencias textuales entre comillas dobles (por ejemplo, “incluye”).

Los caracteres, entre comillas dobles (por ejemplo, “]”, “incluye”). El símbolo ‘|’ indica alternativas, los elementos que se repiten 1 o más veces se escriben entre llaves ‘{}’ mientras que aquellos que se repiten 0 o más veces van entre llaves ‘{}’ y seguidos del símbolo ‘|’. La concatenación se denota por espacios entre los símbolos, mientras que el punto y coma (“;”) marca el final de la sentencia.

En Fig. 2 se presenta la sintaxis de UCEFlow.

```
FlujoDeEventos = {CaminoBase | CaminoAlternativo |};
CaminoBase = "base" {Paso|};
Paso = <code> (SentenciaSimple | SentenciaEspecial | Include);
SentenciaSimple = Articulo Sujeto Accion (Destinatario|) {Complemento |};
Articulo = ("el" | "los" | "la" | "las" | "un" | "una" | "unos");
Sujeto = ("sistema" | Actor | Clase);
Actor = "ac" <string>;
Clase = <string>;
Include = "incluye" NombreCU;
Extend = "extiende" NombreCU;
NombreCU = ("caso" "de" "uso") <string>;
CaminoAlternativo = "alternativo" Condicion ({Paso|} Extend) Retorno;
Condicion = "[" <string> "]";
Retorno = "return" <code>;
Accion = "v" <string>;
Destinatario = Preposicion ("sistema" | Actor | Clase |);
Complemento = Articulo ("sistema" | Actor | Clase ("y"|);
Preposicion = ("al" | "en" | "a");
SentenciaEspecial = ("mientras" | "si") Condicion {Paso|}"fin";
```

Fig. 2. Sintaxis de UCEFlow

3.2 Elementos del lenguaje

A continuación se describen algunos de los conceptos que definen el lenguaje:

1) *Flujo de Eventos:* es la enumeración de pasos sucesivos que ocurren en la interacción entre los actores y el sistema, en el contexto de un CU, con el fin de satisfacer un requisito de los usuarios del sistema.
2) *Camino Base:* Es la secuencia o curso normal de ejecución del CU. Se inicia con la palabra reservada **"base"** para indicar el comienzo del curso normal del CU.
3) *Code*: es el número que indica el orden en que ella se lleva a cabo.
4) *Paso:* representa cada una de las interacciones entre el usuario y el sistema. Un paso puede ser una *sentencia simple*, una *sentencia especial* o un *include*.
5) *Sentencia Simple:* es una oración o unidad lingüística con sentido completo, escrita en voz activa, sólo tiene un sujeto y un predicado. Inicia con el nombre del actor o el "sistema" que realiza la acción e indica los elementos de información que se intercambian. Al inicio se coloca un artículo como palabra reservada para formar el sintagma nominal.
6) *Sujeto:* es el actor, clase o la palabra clave "sistema". Quien realiza la acción.
7) *Complemento:* Es quién recibe o en quién se cumple la acción del verbo. Son en general términos del dominio de aplicación del sistema.
8) *Destinatario:* es hacia quien va dirigida la acción, es opcional.

9) *Acción*: la forma verbal; la acción que lleva a cabo el sujeto de la sentencia.

10) *Include:* cuando se relacionan dos CU con un "include", se dice que el primero incluye al segundo. Se inicia con la palabra reservada **"incluye"** seguida del nombre del CU.

11) *Sentencia Especial:* es un conjunto de pasos, donde su ejecución depende de una *condición*. Se inicia con la palabra reservada **"mientras"** o **"si"** y debe finalizar con la palabra reservada **"fin"**.

12) *Camino Alternativo*: cuando una ocurre una condición poco común, el curso normal del CU puede derivar en pasos condicionales o cursos alternativos, estas desviaciones cortan la secuencia para llevar a realizar un proceso inusual. Se indica con la palabra reservada **"alternativo"**.

13) *Extend:* cuando se relacionan dos CU con un "extend", se dice que el segundo extiende al primero (en un conjunto de pasos que ocurren sólo en algunas oportunidades). Se indica mediante la palabra reservada **"extiende"** seguida del nombre del CU.

14) *Retorno:* indica el punto al que debe retornar el flujo luego de finalizado el camino alternativo. Se inicia con la palabra reservada **"return"** seguida del número de paso del *Camino Base* al cual debe retornar.

15) *Condición*: es aquella propiedad o circunstancia que debe cumplirse necesariamente (resultado "verdadero") para que se ejecute una secuencia de pasos. Dicha condición debe ir encerrada entre corchetes **"["**, **"]"**.

3.3 Validación de UCEFlow

3.3.1 Graficador Léxico

Al inicio de la investigación, las pruebas de la sintaxis se realizaron con AntlrWorks que permite editar, visualizar, interpretar y depurar la sintaxis. En él se introducen las reglas gramaticales y a medida que se van escribiendo dichas reglas de producción el debugger va informando los errores de sintaxis. Cuando están correctamente escritas se van generando los diagramas de sintaxis como se muestran en la Fig. 3.

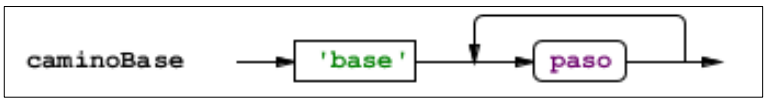

Fig. 3. Sintaxis del camino base.

3.3.2 Parser JAVA

En un principio las pruebas de los diferentes FE se realizaron en el mismo AntlrWorks, utilizando el intérprete de dicha herramienta, en el cual se introduce el FE y por medio de un intérprete de las reglas probadas anteriormente con el graficador léxico se genera el árbol de derivación, sin embargo este parser no es flexible y no se adapta a las necesidades requeridas. Esto motivó la adaptación y uso de un parser en lenguaje JAVA en el IDE NetBeans, desarrollado en un proyecto anterior por el equipo de investigación.

El procedimiento es el que sigue: primero se escribe la sintaxis para que sea cargada en el parser. Luego se escribe el FE siguiendo la estructura prefijada por UCEFlow. Se crea un archivo "test_file.txt" con el FE para que el parser lo analice en base a la sintaxis (el uso de palabras reservadas es para que el parser pueda identificar los elementos) y genere el árbol de derivación.

Este árbol de derivación, si bien no es gráfico, permite identificar claramente los elementos claves de un FE como por ejemplo actores, sistema, objetos, clases, acciones.

A partir de esta identificación se pretende desarrollar una aplicación en la cual se ingrese el FE de la plantilla CUPIDo y se obtenga una clasificación de elementos para el modelado del sistema, si bien no será definitiva servirá a los alumnos para tener desde la redacción del FE una somera idea de las posibles clases que corresponden al sistema que se desea desarrollar.

Se presentan a continuación algunos ejemplos de FE testeados con la sintaxis propuesta.

En la Fig. 4 un ejemplo correspondiente al FE del CU "Presupuestar nuevo servicio". A este FE se lo analizó con el parser desarrollado y se obtuvo como resultado el árbol de derivación mostrado en la Fig. 5, donde se pueden identificar los actores y objetos que intervienen, las sentencias simples y las sentencias especiales.

base
1. El **ac** "encargado de Atención al Público" **v** "solicita" **a** "sistema" **el** "presupuesto de nuevo servicio"
2. El sistema **v** "solicita" **la** "fecha del evento, horario, cantidad de comensales y tipo de evento"
3. El **ac** "encargado de Atención al Público" **v** "ingresa" **los** "datos"
4. **mientras [**"existan productos a alquilar"**]**
   4.1 El sistema **v** "solicita" **la** "cantidad de productos a alquilar"
   4.2 El **ac** "encargado de Atención al Público" **v** "ingresa" **la** "cantidad"
   4.3 El sistema **v** "calcula" **el** "importe total a abonar" **fin**
5. El sistema **v** "genera" **el** "presupuesto"
6. El sistema **v** "almacena" **el** "presupuesto"
7. El sistema **v** "muestra" **un** "mensaje de éxito"

Fig. 4. Ejemplo de FE "Presupuestar Servicio"

```
run:
FlujoDeEventos (CaminoBase ([keyword "base"]
    Paso ([code "1."]
    SentenciaSimple (Articulo ([keyword "el"])
        Sujeto (Actor ([keyword "ac"][string "encargado de Atención al Público"]))
        Accion ([keyword "v"] [string "solicita"])
        Destinatario (Preposicion ([keyword "a"])[string "sistema"])
        Complemento (Articulo ([keyword "el"]) Clase ([string "presupuesto de nuevo servicio"])))
    Paso ([code "2."]
    SentenciaSimple (Articulo ([keyword "el"])
        Sujeto ([keyword "sistema"])
        Accion ([keyword "v"] [string "solicita"])
        Complemento (Articulo ([keyword "la"] )
        Clase ([string "fecha del evento, horario, cantidad de comensales y tipo de evento"])))
    Paso ([code "3."]
    SentenciaSimple (Articulo ([keyword "el"])
        Sujeto (Actor ([keyword "ac"] [string "encargado de Atención al Público"]) )
        Accion ([keyword "v"] [string "ingresa"] )
        Complemento (Articulo ([keyword "los"]) Clase ([string "datos"])))
    Paso ([code "4."]
    SentenciaEspecial ([keyword "mientras"]
        Condicion ([keyword "["] [string "existan productos a alquilar"] [keyword "]"])
        Paso ([code "4.1"]
        SentenciaSimple (Articulo ([keyword "el"])
            Sujeto ([keyword "sistema"])
            Accion ([keyword "v"] [string "solicita"])
            Complemento (Articulo ([keyword "la"] )Clase ([string "cantidad de productos a alquilar"])))
        Paso ([code "4.2"]
        SentenciaSimple (Articulo ([keyword "el"])
            Sujeto (Actor ([keyword "ac"][string "encargado de Atención al Público"]))
            Accion ([keyword "v"][string "ingresa"])
            Complemento (Articulo ([keyword "la"]) Clase ([string "cantidad"])))
        Paso ([code "4.3"]
        SentenciaSimple (Articulo ([keyword "el"])
            Sujeto ([keyword "sistema"])
            Accion ([keyword "v"] [string "calcula"])
            Complemento (Articulo ([keyword "el"]) Clase ([string "importe total a abonar"]) ))
      [keyword "fin"]))
    Paso ([code "5."]
    SentenciaSimple (Articulo ([keyword "el"])
        Sujeto ([keyword "sistema"])
        Accion ([keyword "v"][string "genera"]          )
        Complemento (Articulo ([keyword "el"]) Clase ([string "presupuesto"])))
    Paso ([code "6."]
    SentenciaSimple (Articulo ([keyword "el"])
        Sujeto ([keyword "sistema"])
        Accion ([keyword "v"][string "almacena"])
        Complemento (Articulo ([keyword "el"])Clase ([string "presupuesto"])))
        Paso ([code "7."]
  SentenciaSimple (Articulo ([keyword "el"])
        Sujeto ([keyword "sistema"])
        Accion ([keyword "v"][string "muestra"])
        Complemento (Articulo ([keyword "un"]) Clase ([string "mensaje de éxito"])))))
  BUILD SUCCESSFUL (total time: 0 seconds)
```

Fig. 5. Resultado del parser para FE del ejemplo

3.3.3 Ejemplo de FE con error en la sintaxis

Para mostrar lo que sucedería si se ingresa un FE sin respetar la sintaxis, en el siguiente ejemplo (Fig. 6) se omiten deliberadamente palabras reservadas en la línea 3. Luego, en la Fig. 7 se puede ver la ventana del parser indicando errores. Así se logra ir depurando el FE para que cumpla con la estructura propuesta.

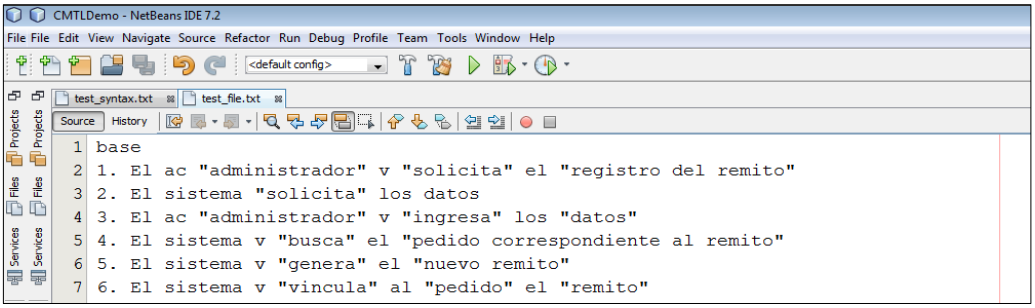

Fig. 6. – FE con error de sintaxis

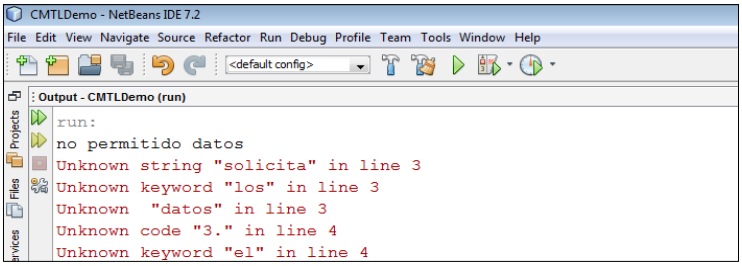

Fig. 7 – Resultado en el parser por el error en la sintaxis

3.3.4 Otros ejemplos analizados

En la Fig. 8, se puede observar otro ejemplo de FE analizado con el parser con su correspondiente árbol de derivación en la Fig. 9.

base
1. El **ac** "encargado de Atención al Público" **v** "solicita" **al** "sistema" **la** "verificacion de un presupuesto existente"
2. El sistema **v** "solicita" **el** "numero de presupuesto"
3. El sistema **v** "busca" **el** "presupuesto"
4. **mientras [**"existan productos asociados al presupuesto"**]**
    4.1. El sistema **v** "verifica" **las** "cantidades del producto para la fecha del evento"
    4.2. **Si [**"cantidad < cantidad solicitada"**]**
        4.2.1. El sistema **v** "informa" **la** "cantidad existente" **fin**
    4.3. **Si [**"cantidad >= cantidad solicitada"**]**
        4.3.1. El sistema **v** "actualiza" **el** "estado del presupuesto" **fin**

fin

Fig. 8 - Ejemplo de FE "Verificar Presupuesto"

run
FlujoDeEventos (CaminoBase ([keyword "base"]
Paso ([code "1."]
SentenciaSimple (Articulo ([keyword "el"])
    Sujeto (Actor ([keyword "ac"][string "encargado de Atención al Público"]))
    Accion ([keyword "v"][string "solicita"])
    Destinatario (Preposicion ([keyword "al"])[string "sistema"])

```
    Complemento (Articulo ([keyword "la"])  Clase ([string "verificacion de un presupuesto existente"])))
Paso ([code "2."]
SentenciaSimple (Articulo ([keyword "el"])
    Sujeto ([keyword "sistema"])
    Accion ([keyword "v"][string "solicita"])
    Complemento (Articulo ([keyword "el"])  Clase ([string "numero de presupuesto"])))
Paso ([code "3."]
SentenciaSimple (Articulo ([keyword "el"])
    Sujeto ([keyword "sistema"])
    Accion ([keyword "v"][string "busca"])
    Complemento (Articulo ([keyword "el"])  Clase ([string "presupuesto"])))
Paso ([code "4."]
SentenciaEspecial ([keyword "mientras"]Condicion ([keyword "["][string "existan productos asociados al presupuesto"][keyword
"]"])
    Paso ([code "4.1."]
    SentenciaSimple (Articulo ([keyword "el"])
        Sujeto ([keyword "sistema"])
        Accion ([keyword "v"][string "verifica"])
        Complemento (Articulo ([keyword "las"]) Clase ([string "cantidades del producto para la fecha del evento"])))
    Paso ([code "4.2."]
    SentenciaEspecial ([keyword "si"]Condicion ([keyword "["][string "cantidad < cantidadsolicitada"][keyword"]"])
        Paso ([code "4.2.1."]
        SentenciaSimple (Articulo ([keyword "el"])
            Sujeto ([keyword "sistema"])
            Accion ([keyword "v"][string "informa"]) Clase ([string "cantidad existente"])))[keyword "fin"]))
    Paso ([code "4.3."]
    SentenciaEspecial ([keyword "si"]Condicion([keyword "["][string"cantidad >= cantidad solicitada"][keyword "]"])
        Paso ([code "4.3.1."]
        SentenciaSimple (Articulo ([keyword "el"])
        Sujeto ([keyword "sistema"])
        Accion ([keyword "v"][string "actualiza"])
        Complemento (Articulo ([keyword "el"]) Clase ([string "estado del presupuesto"])))) [keyword"fin"]))
  [keyword "fin"])))
BUILD SUCCESSFUL (total time: 0 seconds)
```

Fig. 9 - Resultado del parser para FE del ejemplo

4 Conclusiones y Trabajo a Futuro

En este trabajo se ha presentado una sintaxis para estructurar el Flujo de Eventos de Casos de Uso y una herramienta desarrollada especialmente para probarla. Se pretende que la propuesta planteada evolucione, permitiendo obtener un FE bien formado, para ser utilizado luego a través de alguna herramienta Case que permita agilizar la construcción de diagramas de secuencias y de clases a partir de él.

Dentro del proyecto se encuentra en marcha la implementación de una herramienta computacional que permitirá sistematizar el llenado de la plantilla CUPIDo y acelerar el proceso de documentación del modelo de CU. Se proyecta integrar el FE estructurado con la sintaxis presentada, a la misma.

Si bien este trabajo nace con un propósito educativo, no está fuera del alcance plantear que esta estructuración, junto con la plantilla CUPIDo sistematizados, puedan ser utilizados fuera del ámbito académico, ya sea en empresas del medio o cualquier otra entidad que se dedique al diseño de software, y ayudar a los diseñadores a documentar sus casos de uso en forma ordenada y eficiente, permitiendo obtener en forma automática y semiautomática, basados en el Desarrollo Dirigido por Modelos [11], diagramas de secuencia, de objetos y de clases.

Bibliografía

[1] M. I. Lund, C. Ferrarini, E. Ormeño, y S. Zapata, «Indicadores para la Evaluación de Diseños de Sistemas a través del Modelo de Casos de Uso en la Educación Superior», presentado en ICECE - International Conference on Engineering and Computer Education, 2009.

[2] Lund, María Inés, Ferrarini, Cintia, Aballay, Laura Nidia, Romagnano, María Gema, y Meni, Ernesto, «CUPIDo - Plantilla para Documentar Casos de Uso», presentado en V Congreso de Tecnología en Educación y Educación en Tecnología, El Calafate, Santa Cruz, Argentina, 2010.

[3] M. I. Lund, L. Aballay, E. Torres, M. Herrera, y E. Ormeño, «Validación de Usabilidad de una plantilla para documentar Casos de Uso – Estudio Exploratorio», presentado en XVII Congreso Argentino de Ciencias de la Computación, Buenos Aires - Argentina, 2011.

[4] I. Diaz y A. Matteo, «Directrices para la especificaciÓn de casos de uso en el idioma espanol», *Acta cientifica Venezolana*, vol. 53, n.º 2, pp. 139-148.

[5] C. Rolland, C. B. Achour, C. Cauvet, M. Jarke, P. Haumer, K. Pohl, y et al, *A Proposal For A Scenario Classification Framework*. 1996.

[6] I. Díaz, F. Losavio, A. Matteo, y O. Pastor, «A specification pattern for use cases», *Information & Management*, vol. 41, n.º 8, pp. 961-975, nov. 2004.

[7] C. Larman, *UML y patrones: una introducción al análisis y diseño orientado a objetos y al proceso unificado*. Pearson Prentice Hall, 2003.

[8] L. Aballay, S. Cruz Introini, M. I. Lund, y E. Ormeño, «UCEFlow: a syntax proposed to structuring the event flow of use cases», en *Proceedings 8CCC*, Armenia, Colombia, 2013.

[9] C. F. Pons, *Desarrollo de Software Dirigido por Modelos. Conceptos teóricos y su aplicación práctica*. La Plata, 2010.

[10] «EBNF - Extended Backus Naur Form». [En línea]. Disponible en: http://www.macs.hw.ac.uk/~rjp/Coursewww/Cwww/EBNF.html.

[11] O. Pastor, S. España, J. I. Panach, y N. Aquino, «Model-Driven Development», *Informatik Spektrum*, vol. 31, n.º 5, pp. 394-407, oct. 2008.