**Preguntas e ideas varias**

* Problema de enfoque de reintentos con Conector: Los requests son **sincrónicos**, por lo tanto en caso de necesitar reintentos, habría que demorar el response hasta obtener la respuesta. De querer hacerlo, una opción es hacer un polling interno en el conector, hasta tener un resultado, y recién ahí devolver el thread. Notar que no se puede tener varios reintentos, ya que el invocante va a estar bloqueado esperando la respuesta, y por lo tanto un timeout es algo factible de ocurrir.
* A mi parecer, el enfoque de reintentos para servicios sincrónicos debería ser responsabilidad del que invoca, y no del Conector.
* En caso que el servicio que se invoca sea **sincrónico, pero en realidad no se necesita la respuesta inmediatamente** (algo como un soap send and forget), entonces en este caso una política de reintentos tiene sentido. Una sugerencia de implementación sería **marcar el Conector como un request asincrónico, y el request genera un mensaje en una cola**, e inmediatamente devuelve la respuesta al invocante. Internamente, se tiene un listener que procesa el request, lo ejecuta, y guarda el resultado, de manera que quede traza de que la invocación fue exitosa. En caso de problemas, se puede volver a intentar (el mensaje no se marca como consumido hasta tener éxito), hasta lograr la invocación o haber utilizado todos los reintentos disponibles.
* Para el punto anterior, sería necesario definir políticas de reintento, visualizador de estado actual de invocaciones, persistencia, etc.

Notas de reunión

* El enfoque se ajusta más al del conector de trazabilidad, que funciona como un send and forget
* Agregar un “módulo” al conector para que permita procesar este tipo de mensajes
* Revisar código actual del conector de trazabilidad, para ver cómo lo está haciendo
* Guardado de la respuesta? En ese caso, proveer una base más robusta que derby
* Revisar si el conector de trazabilidad tiene un callback, luego de haber invocado exitosamente
* Política de reintentos como configuración global
* Checkbox para indicar si es asincrónico o no (se llamaría “Tiene reintentos”)
* En una primera versión, no hay visualización, y se accede por bd para ver el estado actual de invocaciones
* Para analizar: en caso de usar reintentos, pasar a usar una base más robusta, y considerar una potencial migración de derby a otra base en caso de instalar la funcionalidad (pensar en la forma de tener dualidad si no es muy complicado)