Registro de cambios

Registro de cambios respecto a la D05 del proyecto.

ChangeLog D06

10/01/2019

**D&T Grupo 11**

Albalat Heredia, Carlos

del Carmen Fuentes, Lucía

Martínez Figueroa, Pablo

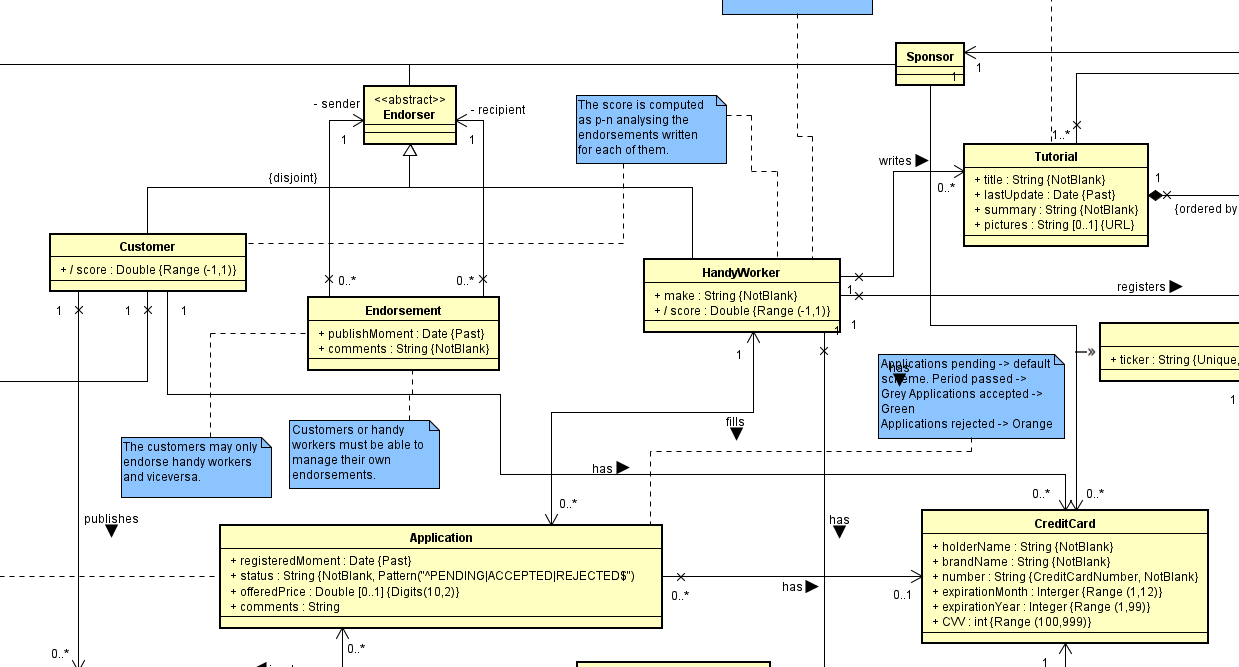
Montaño Aguilera, Antonio Manuel

de Paz Galán, Francisco

Rico Catalán, Jesús Andrés

# Cambios en los modelos de dominio

* Se ha añadido la relación Customer 🡪 CreditCard y Sponsor 🡪 CreditCard con la finalidad de conocer el propietario específico de una tarjeta de crédito.



* Se ha añadido en los modelos UML la bidireccionalidad Note <-> Report, que ya estaba puesta anteriormente en el modelo de dominio Java.
* Eliminados los cascade = CascadeType.ALL de Message y MessageBox.
* Actor::getSocialProfiles() se ha eliminado cascade=CascadeType.ALL ya que no es una composición.
* HandyWorker::getCurriculum(): se ha eliminado la etiqueta @NotNull.

# Cambios en los repositorios

* ComplaintRepository: se ha añadido una consulta que devuelve las quejas a partir de un handy worker.

# Cambios en los servicios

* En WarrantyService::save() ahora se comprueba si la garantía está en modo final o no.
* PhaseService::save(): ahora las fases solo pueden crearse cuando una solicitud esté aceptada. También se cumplen ahora las restricciones de tiempo.
* PhaseService::delete(): anteriormente no borraba correctamente la fase, ahora sí lo hace.
* AdministratorService: se ha añadido un método que calcula la puntuación de un cliente o un manitas. También se ha añadido un método que calcula las puntuaciones de todos ellos.
* FinderService: se ha modificado el método de búsqueda que devuelve un nuevo buscador que incluye las chapuzas que cumplen los filtros indicados.
* Se han modificado los test para que, en lugar de buscar un elemento por su id, funcionen siempre a pesar de que se vuelva a poblar la base de datos.
* ComplaintService::save(): cuando se crea una queja, esta se añade a la lista de quejas del cliente en cuestión.
* FixUpTaskService::save(): se ha añadido un nuevo método que devuelve al cliente creador de esa chapuza. También se ha añadido en este método la creación del momento de publicación y del ticker, en lugar de en el método create(). Por otra parte, se ha añadido la nueva forma de comprobar si una chapuza contiene spam.
* En varios servicios se ha utilizado el método instaceOf para comprobar que el usuario que está logueado puede pertenecer a varios roles.
* CurriculumService: se ha eliminado el método delete() ya que al considerarse que el currículum es opcional, no debería ser borrado.
* PersonalRecordService: se ha eliminado el método delete() ya que es obligatorio en un currículum y se ha considerado que este no puede ser borrado.
* UtilityService::generateTicker(): se ha modificado el método ya que con anterioridad daba fallos al obtener el día y el mes correspondiente.
* NoteService::save(): se ha modificado el método y ya se guarda la nota correctamente en la base de datos.
* NoteService::delete(): se ha eliminado el método ya que se considera que una nota no debe ser borrada una vez ha sido creada.
* Se han añadido en varios servicios comprobaciones para que, al guardar una entidad, no se actualicen sus atributos fijos.
* ApplicationService::save(): se ha añadido la nueva forma de comprobar si una solicitud contiene spam.
* MessageBoxService::save(): se ha añadido la nueva forma de comprobar si un mensaje de una carpeta contiene spam.
* ReportService: se han rehecho los métodos create y save y ahora funcionan de forma correcta.