Procesos de Desarrollo de Software

Domain-Driven Desing

Sesión Práctica 01

Contenidos

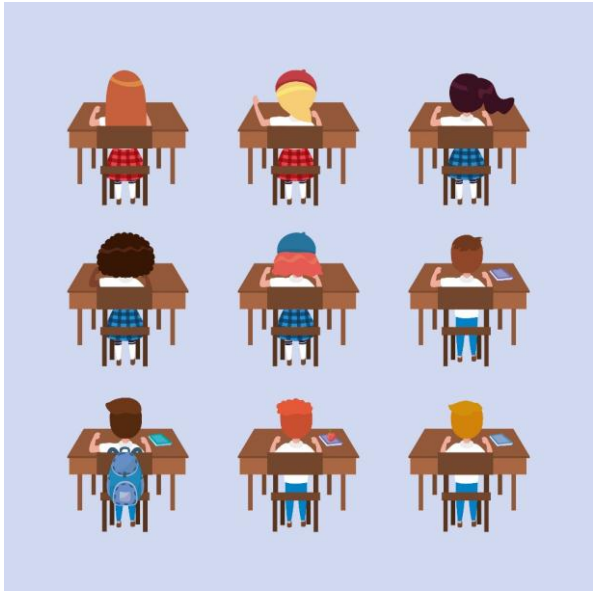
1. Caso práctico.
2. Resultados.

Caso práctico

Vigilancia de exámenes

2. Caso práctico – Enunciado

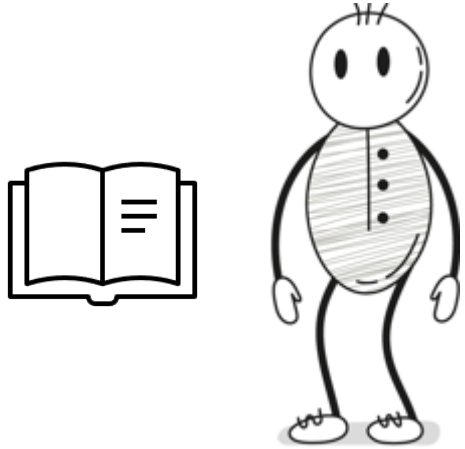
En la facultad se desea hacer una aplicación para gestionar profesores, asignaturas, exámenes y vigilancias realizadas por cada profesor en cada examen.



Fuente: http://www.freepik.com. Diseñado por stockgiu

2. Caso práctico – Enunciado

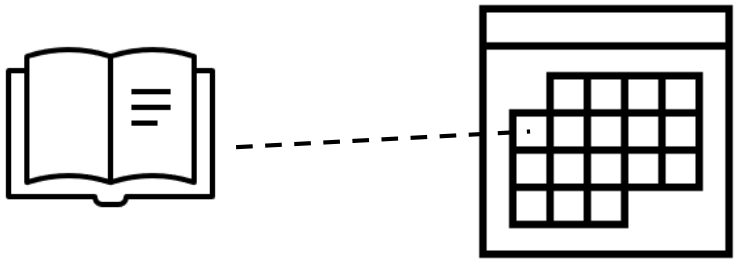
Cada asignatura está asociada a una titulación y tiene un profesor responsable.



Fuente: http://www.freepik.com

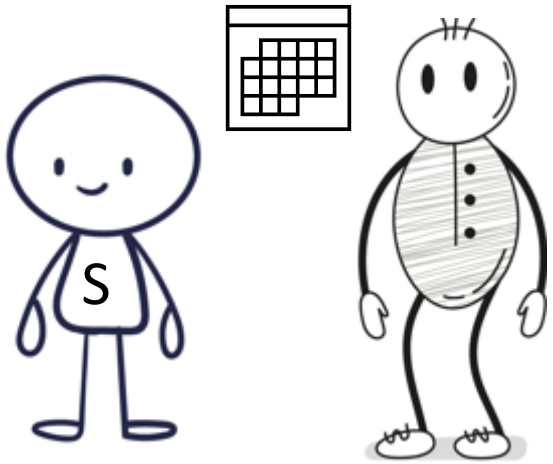
2. Caso práctico – Enunciado

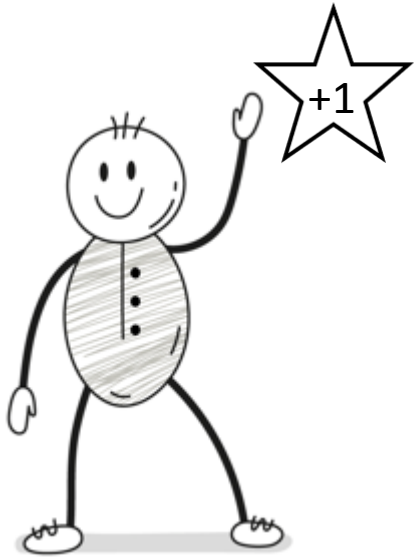
En cada convocatoria (*febrero, junio y julio*) se realiza un examen de cada asignatura, cada uno en una fecha determinada.



2. Caso práctico – Enunciado

Con suficiente antelación, antes del periodo de exámenes, el secretario de la facultad asigna a cada examen un turno (mañana, tarde), una o varias aulas y el profesor responsable del mismo. Este profesor responsable es el encargado de determinar si es necesario algún profesor más para ayudar a vigilar el examen, y de solicitarlo, en su caso, como mínimo con 15 días de antelación a la celebración del examen.

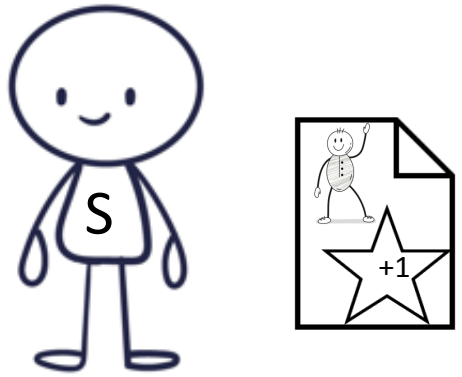




Fuente: http://www.freepik.com

2. Caso práctico – Enunciado

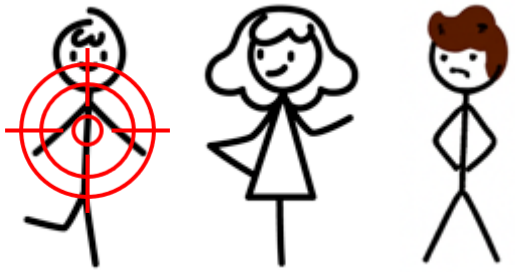
El secretario de la facultad recibe una notificación cada vez que se solicitan vigilantes para un examen. El secretario revisa periódicamente estas solicitudes y las valida asignando profesores vigilantes al mismo y avisando a los profesores implicados para que confirmen la vigilancia



Fuente: http://www.freepik.com

2. Caso práctico – Enunciado

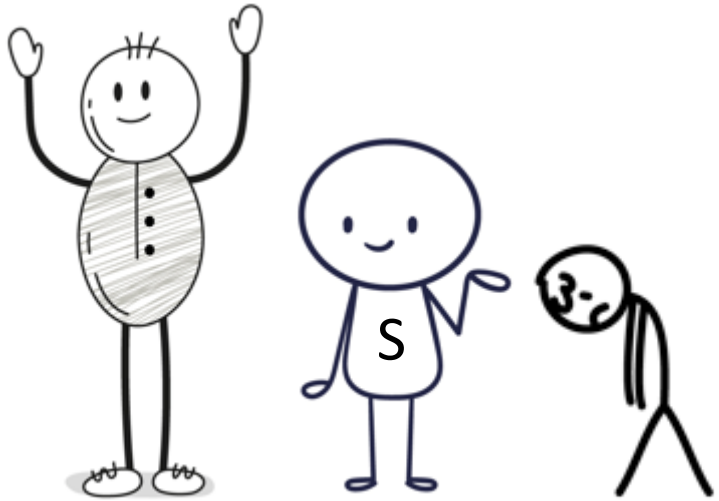
Esta revisión de solicitudes se realiza de la siguiente manera: el secretario de la facultad inicia la validación de la solicitud, tras lo cual el sistema asignará las vigilancias solicitadas a los profesores con menos vigilancias hasta el momento y que no tengan un examen en el mismo día y turno. En caso de empate el sistema seleccionará al profesor vigilante de manera aleatoria.

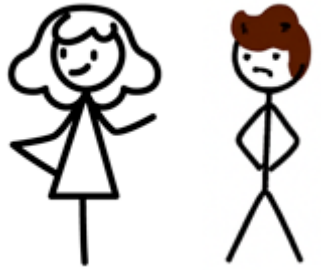


Fuente: http://www.freepik.com

2. Caso práctico – Enunciado

El secretario entonces validará esa asignación y el sistema avisará automáticamente a los profesores seleccionados para que verifiquen que pueden realizar la vigilancia y la confirmen o bien indiquen que no les es posible realizar dicha vigilancia.





Fuente: http://www.freepik.com

Resultados

Posible solución al problema

3. Resultados – Lenguaje ubicuo (1/2)

Término	Descripción
Profesor	Persona que enseña en la facultad
Asignatura	Materia que se enseña en la facultad
Examen	Prueba que los profesores realizan a los estudiantes
Vigilancia	Asignación de un profesor para que vigile un examen
Titulación	Título académico compuesto de asignaturas
Profesor responsable	Profesor encargado de una asignatura
Convocatoria	Periodo en el que se realizan unos determinados exámenes
Secretario	Persona encargada de validar y asignar vigilancias
Turno	Momento del día en el que se realiza un examen (mañana, tarde)
Aula	Lugar donde se realiza un examen
…	

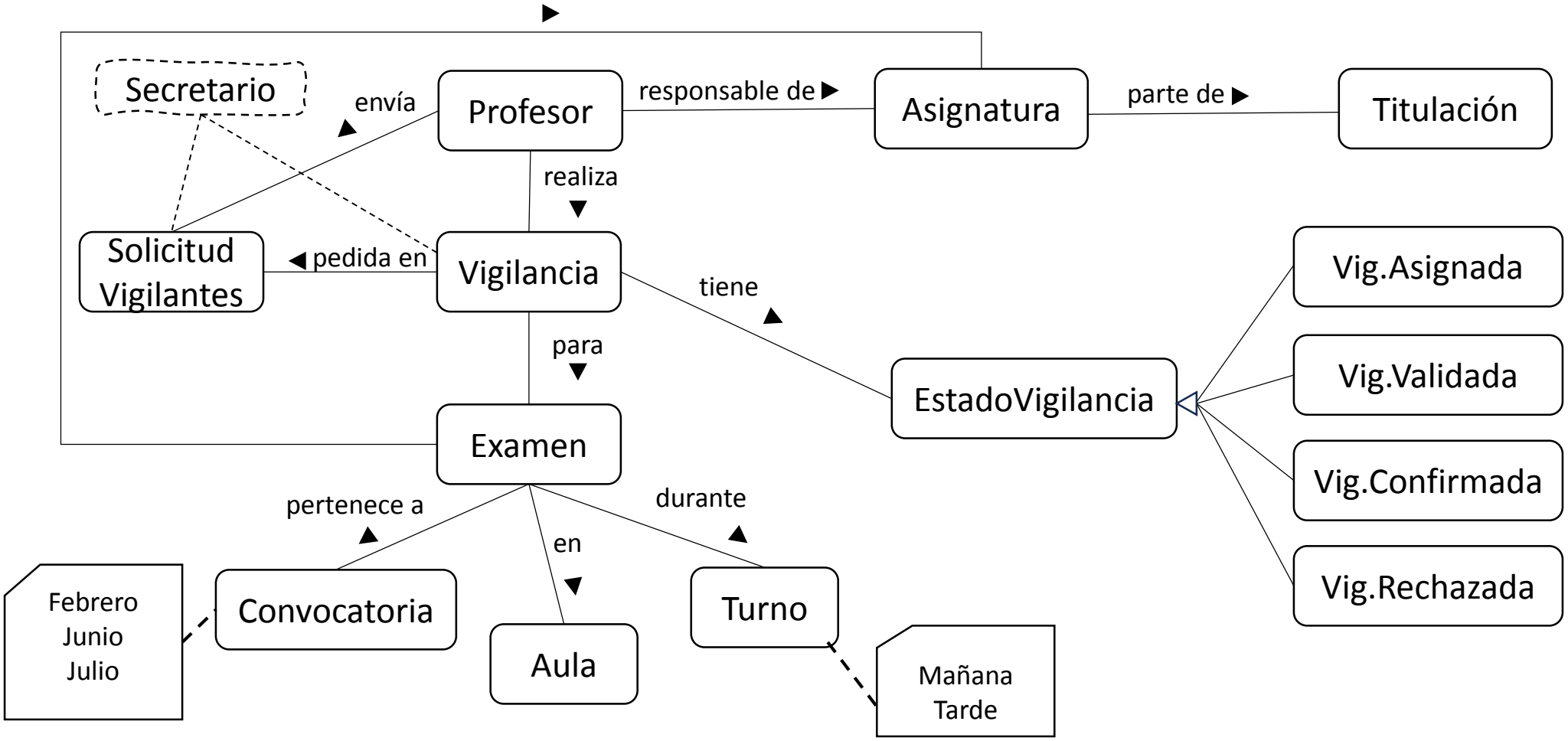
3. Resultados – Lenguaje ubicuo (2/2)

Término	Descripción
…	
Solicitud de vigilantes	Solicitud de profesores adicionales para la vigilancia de un examen
Profesor vigilante	Profesor al que se le ha asignado una vigilancia
Notificación de vigilancia	Notificación de que se ha realizado una solicitud de vigilantes
Vigilancia asignada	Estado de vigilancia indicando que ha sido asignada a un profesor
Vigilancia validada	Estado de vigilancia indicando que ha sido validada por secretario
Vigilancia confirmada	Estado de vigilancia indicando que ha sido confirmada por profesor vigilante
Vigilancia rechazada	Estado de vigilancia indicando que ha sido rechazada por el profesor asignado

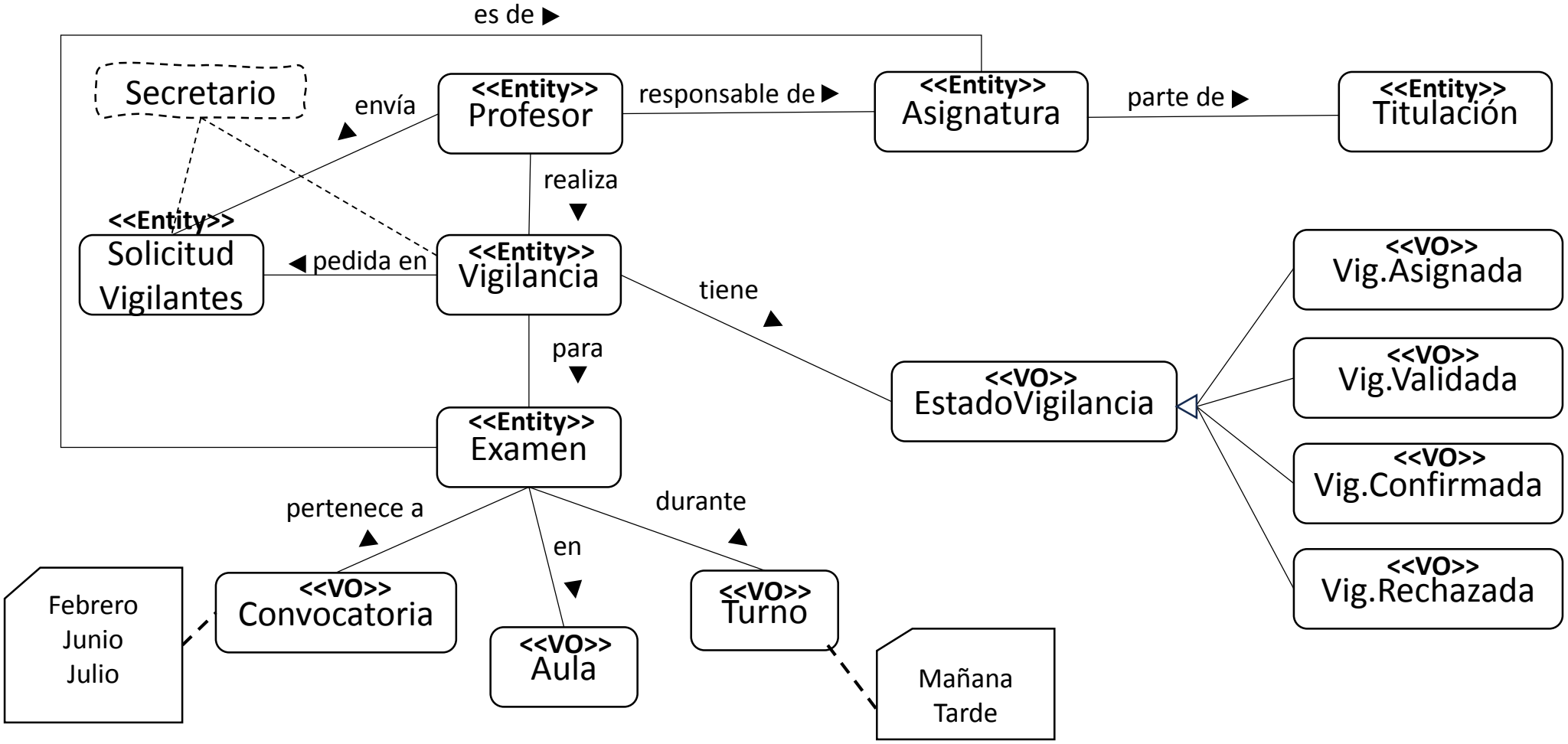
3. Resultados – Historias de usuario

- Como secretario quiero asignar un examen a un aula y fecha para planificar su realización.
- Como profesor responsable quiero solicitar vigilantes para asegurarme de que el examen de mi asignatura se realiza justamente.
- Como secretario quiero revisar y validar solicitudes de vigilancia para asegurar que todos los exámenes tienen los vigilantes que necesitan.
- Como secretario quiero lanzar el proceso de asignación automática para que se elijan automáticamente profesores que cubran una vigilancia.
- Como secretario quiero validar una asignación automática para asegurarme de que es correcta.
- Como profesor vigilante quiero confirmar/rechazar una vigilancia para que el secretario sepa que puedo/no puedo realizarla.

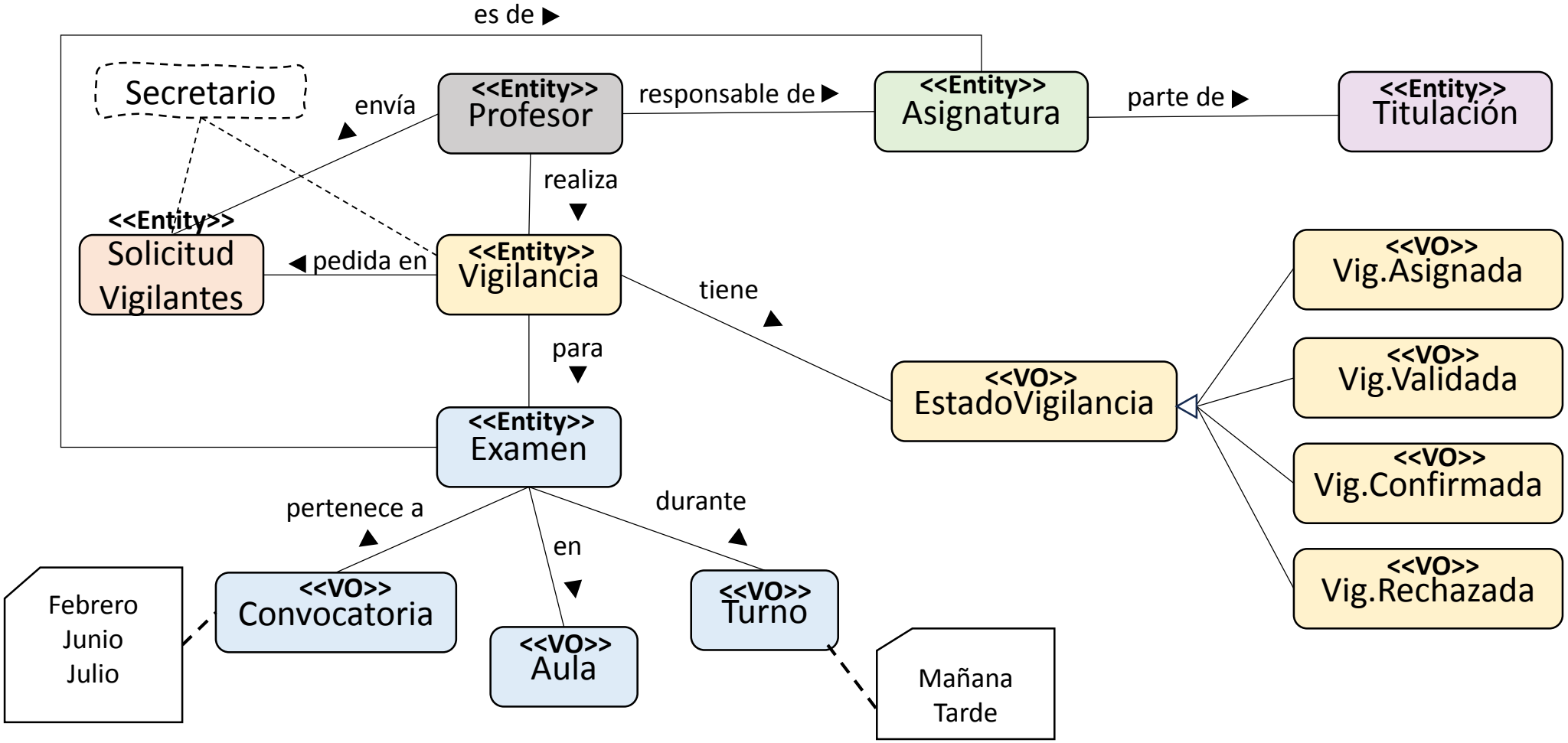
3. Resultados – Modelo del dominio



3. Resultados – Entidades y Value Objects



3. Resultados – Agregados



3. Resultados – Discusión

- Secretario no es una entidad puesto que no la registramos en el sistema. Se ha dibujado en el modelo de dominio para completar la información, pero no es parte del mismo de manera formal.
- ¿Examen y Vigilancia en el mismo agregado?
  - Una Vigilancia representa una asignación para vigilar un examen para un profesor. Una vigilancia es un agregado raíz (de una sola clase) porque tiene un ciclo de vida propio (aceptado, rechazado, etc.)
  - Imaginemos que el algoritmo de asignación de vigilancias necesita comprobar datos de vigilancias pasadas. Si Vigilancia fuera parte del agregado Examen (siendo Examen raíz) entonces tendría que cargar todos los exámenes y sería costoso.

3. Resultados – Discusión

- Cómo implementar “Asignar vigilancias a profesores que han vigilado menos exámenes en el pasado”
  - Necesita manejar varios agregados. Esta responsabilidad se asignará a un servicio de dominio.
  - Dos formas de implementar esta regla de negocio:
    - Anotar en el profesor la carga de trabajo (capturando eventos de vigilancias aceptadas). Es decir, hay denormalización.
      - Cuando una vigilancia se acepta -> generar evento `VigilanciaAsignada(profesorId)`
      - Un manejador de eventos incrementa la carga de vigilancias del profesor
      - El RepositorioProfesor puede tener una consulta para obtener los profesores con menos vigilancias
    - Crear un servicio de consultas CargaVigilanciaQueryService cuya implementación acceda a las tablas involucradas en una única consulta
      - No ponemos esta lógica en un repositorio porque no encaja de forma natural en ninguno
      - La implementación hará un “join” de las tablas implicadas obtentiendo los profesores con menos vigilancias

3. Resultados – Servicios de dominio

```java
public interface ServicioAsignacionVigilancias {
  public List<Vigilancia> asignarVigilancias(
    Examen examen, RepositorioProfesores repositorioProfesores, int numVigilancias);
}

public class ServicioAsignacionVigilanciasRankingInverso implements ServicioAsignacionVigilancias {
  public List<Vigilancia> asignarVigilancias(
    Examen examen, RepositorioProfesores repositorioProfesores, int numVigilancias) {
    // Ranking (inverso) de profesores por vigilancias pasadas.
    var profesores = repositorioProfesores.profesoresConMenosVigilancias()
    // Filtrar profesores según disponibilidad si es necesario y extraer N profesores
    var profesoresFiltrados = ...;

    // Devuelve vigilancias para los N profesores primeros del ranking
    return profesoresFiltrados.stream().map(p -> new Vigilancia(p.id(), examen.id()));
  }
}
```

* Esta versión está implementada asumiendo denormalización en Profesor

3. Resultados – Servicios de aplicación

- Posibles servicios de aplicación (de las H.U.)
- ServicioAplicacionSolicitarVigilantes
  - 1. Validar permisos de profesor responsable de asignatura
  - 2. Recuperar datos examen
  - 3. Modificar número de vigilantes necesarios
  - 4. Persistir solicitud
  - 5. Notificar secretario

- ServicioAplicacionValidarSolicitudes
  - 1. Validar permisos de secretario
  - 2. Recuperar solicitudes de vigilantes
  - 3. Marcar solicitudes como validadas
  - 4. Persistir solicitudes
- Etc.