Programación 4

Informe del
Modelo de Casos de Uso

Grupo 23

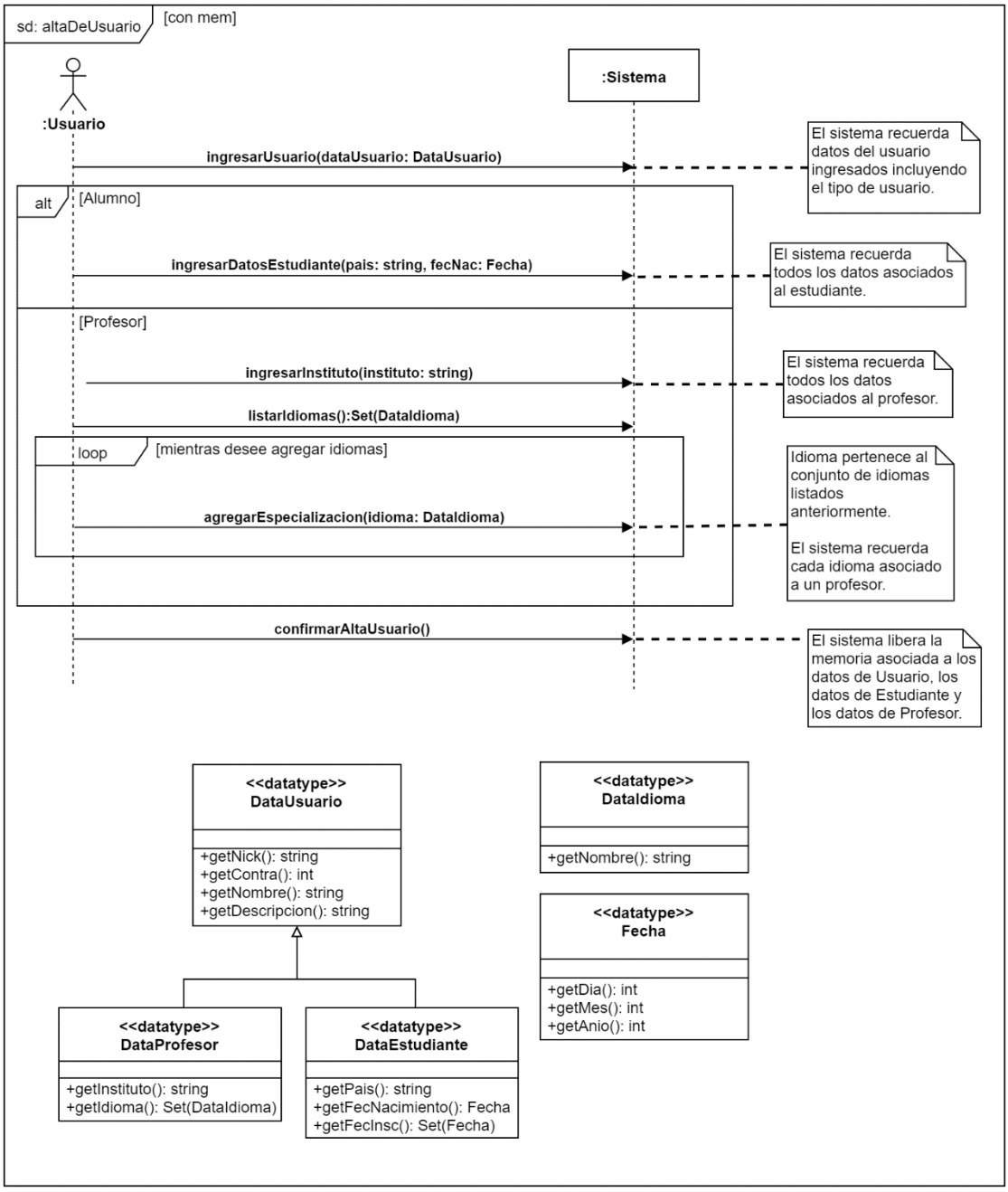
Integrantes:

Franco Sebastián Biardo Sienra	**CI:** 5.448.483-6
Luis Alejandro Ledezma Torrealba	**CI:** 6.401.425-5
Luca Ignacio Pisano Pereyra	**CI:** 5.290.708-6
Martín Terevinto Bidegain	**CI:** 5.135.210-7
Alejandro Daniel Vázquez Cirio	**CI:** 5.282.540-2

Docente: Miguel Machado

Alta de Usuario

Diagramas de Secuencia del Sistema



Contratos

Nombre	**Ingresar Usuario.**
Operación	`ingresarUsuario(usuario: DataUsuario)`
Entrada	`usuario` representa una instancia de Estudiante o Profesor y con valores en sus atributos que indican el nickname, la contraseña, el nombre y la descripción del Usuario a ingresar.
Salida	No hay.
Descripción	Guarda en memoria los datos del nuevo usuario que se añadirá al sistema.
Excepciones	Si no se cumple Pre1, se lanza la excepción ExYaExisteUsuario.

Precondiciones y Postcondiciones
Pre1: No existe en sistema un Usuario U con U.Nickname == usuario.getNick().
Pre2: El parámetro Contra tiene un largo mínimo de 6 caracteres.
Post1: Se almacenó en la memoria del sistema la información de usuario.

Nombre	**Ingresar datos de estudiante.**
Operación	`ingresarDatosEstudiante(pais: string, fecNac: Fecha)`
Entrada	`pais` representa el País de residencia de un estudiante y `fecNac` representa la fecha de nacimiento de este.
Salida	No hay.
Descripción	Añade los datos de entrada a la instancia de Estudiante almacenada en la memoria del sistema.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en memoria una instancia de Estudiante.
Post1: Se añade el valor de pais y fecNac a la instancia de Estudiante.

Nombre	**Ingresar Instituto.**
Operación	`ingresarInstituto(instituto: string)`
Entrada	`instituto` representa el Instituto donde trabaja un Profesor.
Salida	No hay.
Descripción	Añade el dato de entrada a la instancia de Profesor almacenada en la memoria del sistema.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en memoria una instancia de Profesor.
Post1: Se añade el valor de instituto a la instancia de Profesor.

Nombre	**Listar Idiomas.**
Operación	`listarIdiomas(): Set(DataIdioma)`
Entrada	No hay.
Salida	Un conjunto de DataIdioma.
Descripción	Lista los idiomas existentes en el sistema.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: No hay.
Post1: No hay.

Nombre	**Agregar especialización a profesor.**
Operación	`agregarEspecializacion(idioma: DataIdioma)`
Entrada	`idioma` representa una instancia de Idioma almacenado en el sistema.
Salida	No hay.
Descripción	Asocia el idioma ingresado al Profesor almacenado en la memoria del sistema.
Excepciones	No hay.

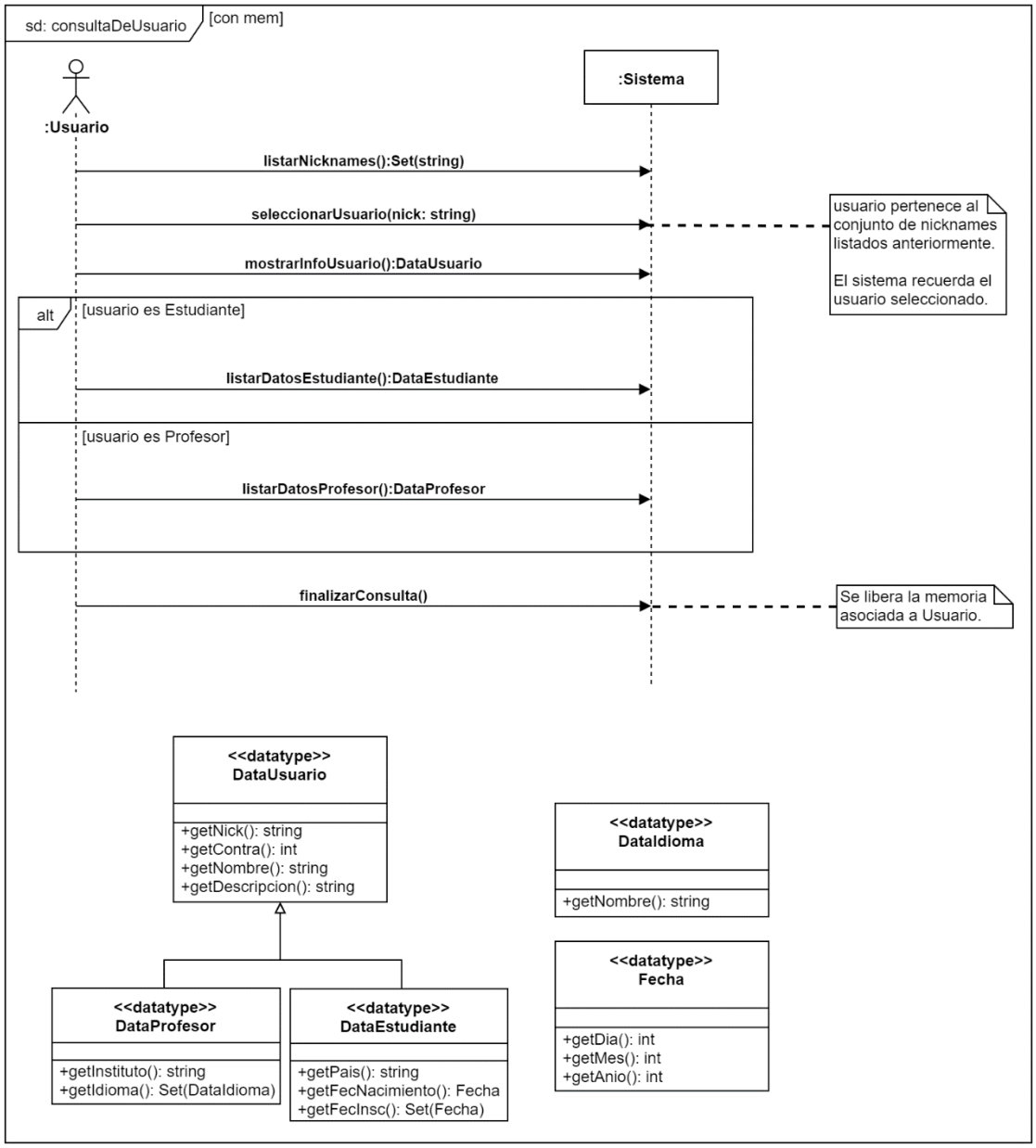
Precondiciones y Postcondiciones
Pre1: Existe en la memoria del sistema una instancia de Profesor.
Pre2: Existe en el sistema una instancia de Idioma I con I.nombre == idioma.nombre.
Post1: Se asoció la instancia de Idioma a la instancia de Profesor.

Nombre	**Confirmar el alta del usuario.**
Operación	`confirmarAltaUsuario()`
Entrada	No hay.
Salida	No hay.
Descripción	Confirma el alta del Usuario recordado por el sistema.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en memoria una instancia de Estudiante o Profesor.
Post1: Se creó una instancia de Estudiante o Profesor con los valores en su tipo y sus atributos correspondientes al usuario que existía en memoria.
Post2: Se liberó la memoria del sistema.

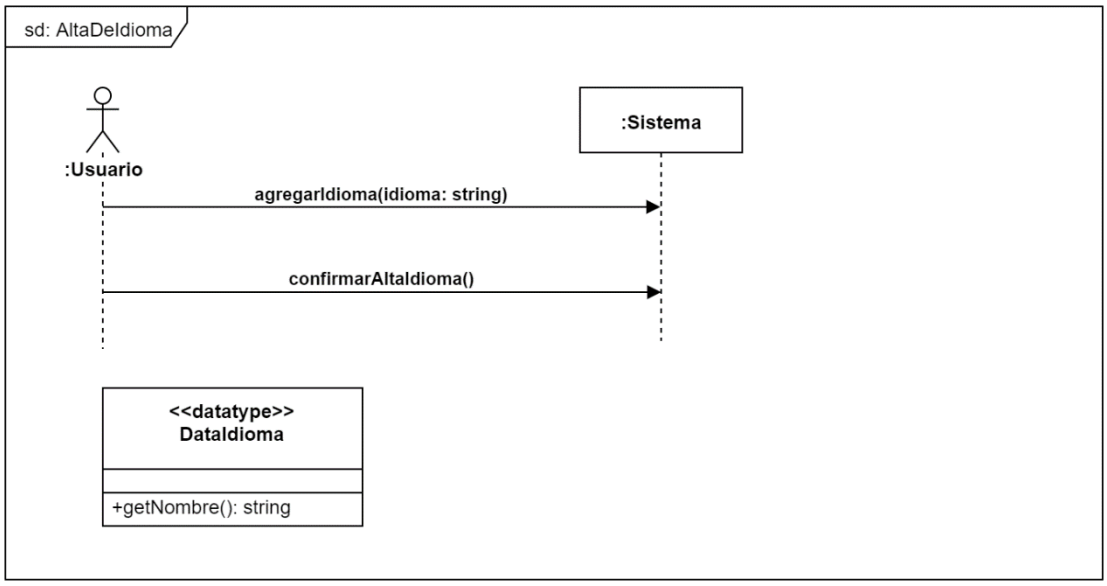
Consulta de Usuario

Diagramas de Secuencia del Sistema



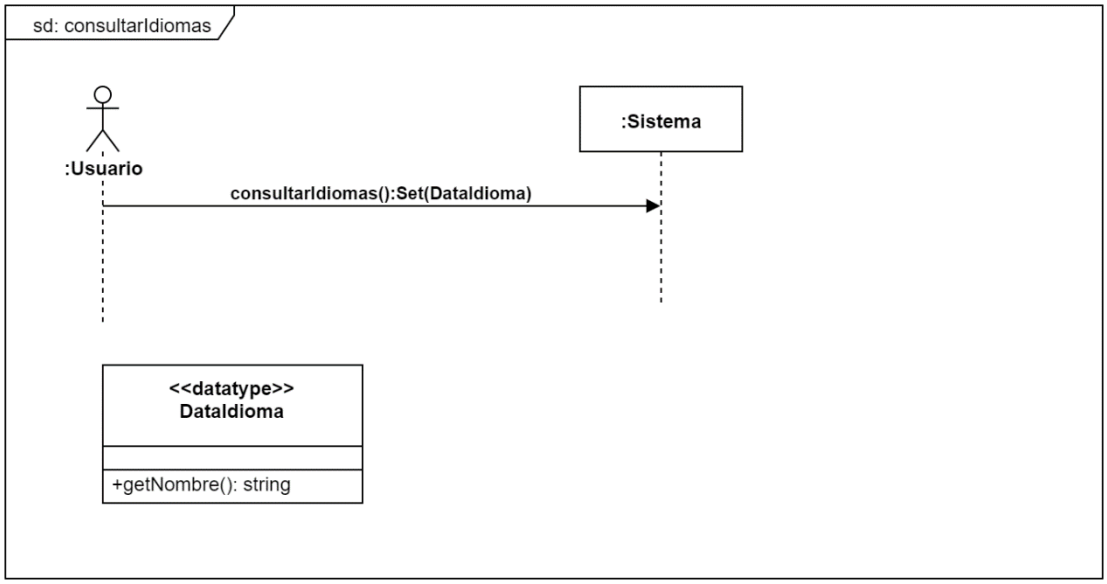
Alta de Idioma

Diagramas de Secuencia del Sistema



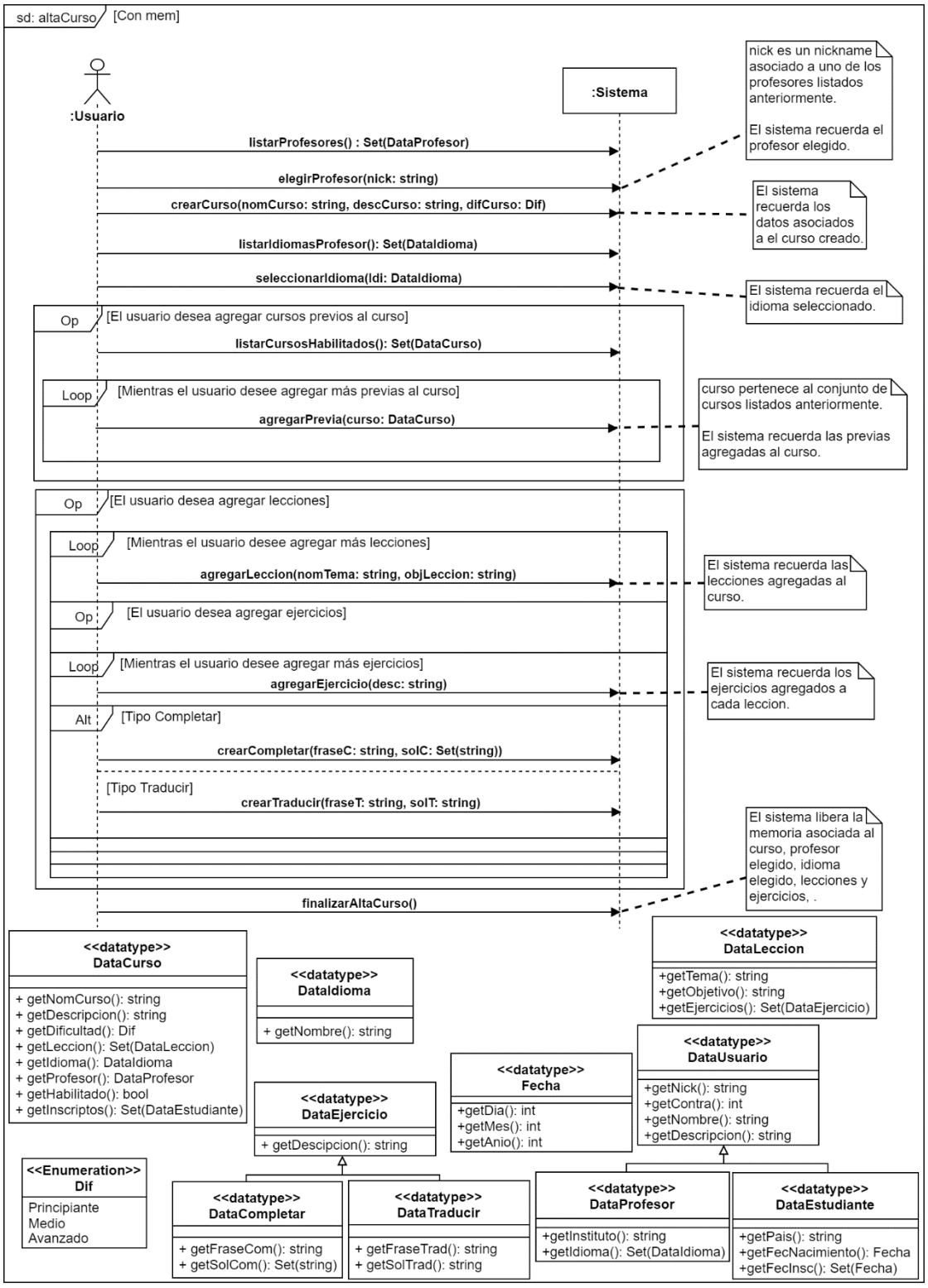
Consultar Idiomas

Diagramas de Secuencia del Sistema



Alta de Curso

Diagramas de Secuencia del Sistema



Contratos

Nombre	**Listar Profesores.**
Operación	`listarProfesores():Set(DataProfesoor)`
Entrada	No hay.
Salida	Un conjunto de DataProfesor que representa todos los profesores existentes en el sistema.
Descripción	Se listan todos los profesores ingresados en el sistema.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: No hay.
Post1: No hay.

Nombre	**Elegir Profesor.**
Operación	`elegirProfesor(Nick: string)`
Entrada	`nick` representa el nickname del profesor a elegir
Salida	No hay.
Descripción	Marca un profesor en específico para ser utilizado en futuras operaciones
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en el sistema una instancia P de Profesor con P.nick == nick
Post1: Se almacenó en la memoria la información asociada al profesor cuyo nick fue ingresado.

Nombre	**Crear Curso.**
Operación	`crearCurso(nomCurso: string, descCurso: string, difCurso: Dif)`
Entrada	`nomCurso` representa el nombre que se le asignará al curso, así como `desCurso` representa la descripción del curso y `difCurso` representa la dificultad de este.
Salida	No hay.
Descripción	Guarda en la memoria del sistema un DataCurso con los datos ingresados y el profesor elegido previamente.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en la memoria del sistema una instancia de Profesor.
Post1: Se almacenó en la memoria la información asociada al curso.

Nombre	**Listar Idiomas de Profesor.**
Operación	`listarIdiomasProfesor():Set(DataIdioma)`
Entrada	No hay.
Salida	Un conjunto de DataIdioma que representa todos los idiomas que un profesor en especifico domina
Descripción	Retorna los idiomas en los que se especializa un profesor.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en la memoria del sistema una instancia de Profesor.
Post1: No hay.

Nombre	**Seleccionar Idioma.**
Operación	`seleccionarIdioma(Idi: DataIdioma)`
Entrada	`Idi` representa una instancia del Idioma a elegir.
Salida	No hay.
Descripción	La operación asocia el idioma ingresado con un curso sin idioma asociado.
Excepciones	Si no se cumple Pre1 se lanza la excepción ExProfesorNoElegido. Si no se cumple Pre2 se lanza la excepción ExNoExisteIdiomaEnProfesor. Si no se cumple Pre3 se lanza la excepción ExNoExisteCursoLibre.

Precondiciones y Postcondiciones
Pre1: Existe en la memoria del sistema una instancia de Profesor.
Pre2: Existe en el sistema una instancia I de Idioma con I.nombre == Idi.getNombre() asociada al profesor elegido.
Pre3: Existe en la memoria del sistema una instancia de Curso sin Idioma asociado.
Post1: Se asoció Idi a la instancia de Curso en memoria.

Nombre	**Listar Cursos Habilitados.**
Operación	`listarCursosHabilitados():Set(DataCurso)`
Entrada	No hay.
Salida	Un conjunto de DataCurso con su atributo habilitado == true.
Descripción	Retorna los cursos que están habilitados en el sistema.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: No hay.
Post1: No hay.

Nombre	**Agregar Previa.**
Operación	`agregarPrevia(curso: DataCurso)`
Entrada	`curso` representa una instancia de Curso con atributo habilitado == true
Salida	No hay.
Descripción	Crea una asociación entre `curso` y la instancia de Curso almacenada en memoria
Excepciones	Si no se cumple Pre1 se lanza la excepción ExNoExisteCurso. Si no se cumple Pre2 se lanza la excepción ExCursoNoHabilitado. Si no se cumple Pre3 se lanza la excepción ExCursoNoElegido.

Precondiciones y Postcondiciones
Pre1: Existe en el sistema una instancia de Curso que se corresponde con curso.
Pre2: Se cumple que curso.habilitado == true.
Pre3: Existe en la memoria del sistema una instancia de Curso.
Post1: Se asoció curso a la instancia de Curso en memoria.

Nombre	**Agregar Lección.**
Operación	`agregarLeccion(nomTema: string, objLeccion: string)`
Entrada	`nomTema` representa el tema de la lección a agregar y `objLeccion` representa el objetivo de esta.
Salida	No hay.
Descripción	Se crea una instancia de Leccion con los datos ingresados y se asocia a la instancia de Curso en memoria.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en la memoria del sistema una instancia de Curso.
Post1: Se almacenó en la memoria la información de la Lección.
Post2: Se asoció la Lección a la instancia de Curso en memoria.

Nombre	**Agregar Ejercicio.**
Operación	`agregarEjercicio(desc: string)`
Entrada	`desc` representa el atributo descripción del ejercicio a crear.
Salida	No hay.
Descripción	Se crea una instancia E de Ejercicio con E.descripcion = desc y se asocia a la instancia de Lección en memoria.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en memoria una instancia de Leccion.
Post1: Se almacenó en la memoria la información de la instancia de Ejercicio.
Post2: Se asoció la instancia de Ejercicio a la instancia de Leccion en memoria.

Nombre	**Crear Tipo Completar.**
Operación	`crearCompletar(fraseC: string, solC: Set(string)`
Entrada	`fraseC` se corresponde al atributo fraseCom y `SolC` se corresponde al atributo solCom de una instancia de Completar.
Salida	No hay.
Descripción	Se crea una instancia de Completar con los atributos ingresados y la descripción de una instancia de Ejercicio en memoria y se le asocia la instancia de Leccion de la misma.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en memoria una instancia de Ejercicio.
Post1: Se asoció la instancia de completar a la instancia de Leccion asociada a la instancia de Ejercicio.

Nombre	**Crear Tipo Traducir.**
Operación	`crearTraducir(fraseT: string, solT: string)`
Entrada	`fraseT` se corresponde al atributo fraseTrad y `SolT` se corresponde al atributo solTrad de una instancia de Traducir.
Salida	No hay.
Descripción	Se crea una instancia de Traducir con los atributos ingresados y la descripción de una instancia de Ejercicio en memoria y se le asocia la instancia de Leccion de la misma.
Excepciones	No hay.

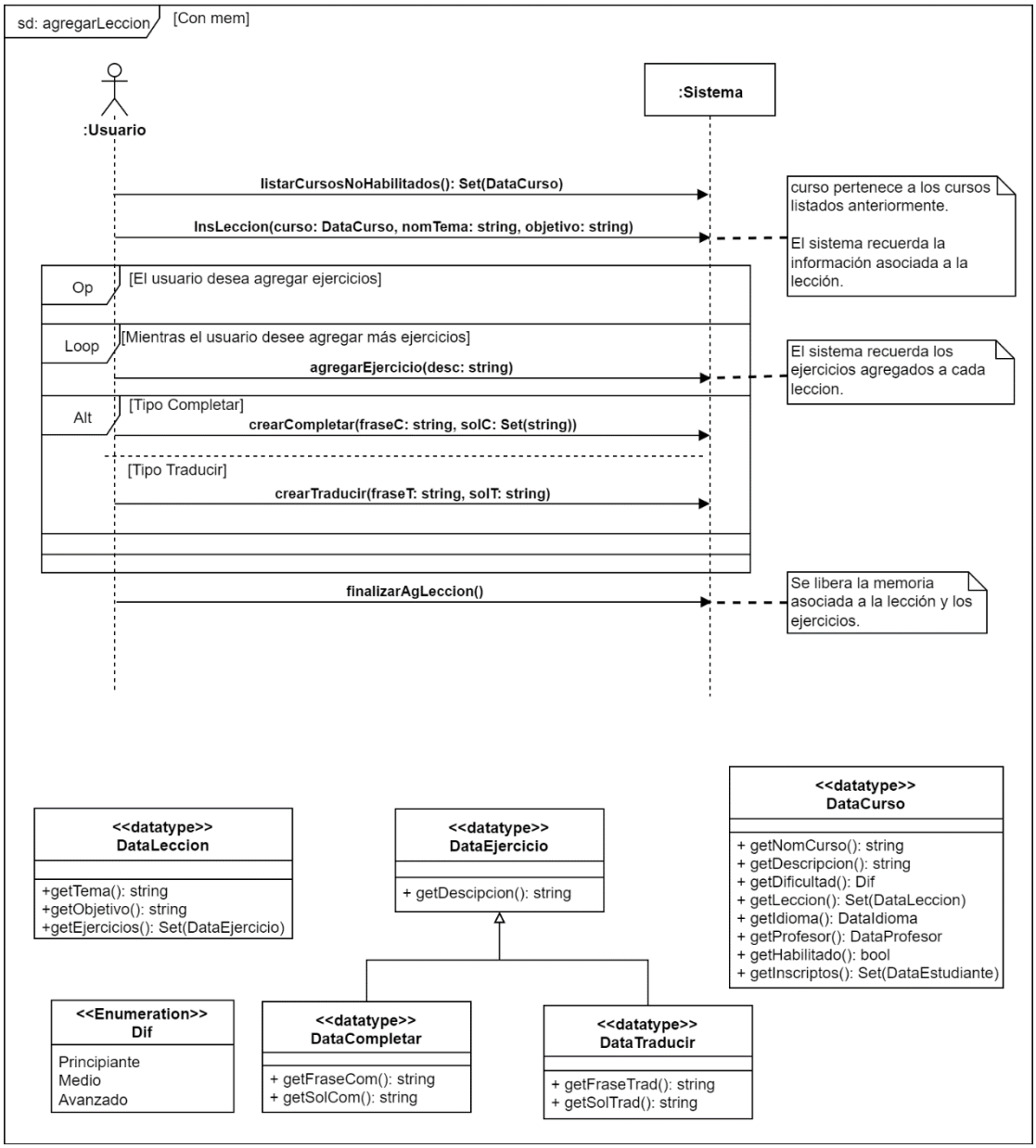
Precondiciones y Postcondiciones
Pre1: Existe en memoria una instancia de Ejercicio.
Post1: Se asoció la instancia de Traducir a la instancia de Leccion asociada a la instancia de Ejercicio.

Nombre	**Finalizar Alta de Curso.**
Operación	`finalizarAltaCurso()`
Entrada	No hay.
Salida	No hay.
Descripción	Confirma el alta del Curso con las lecciones y ejercicios que se le hayan añadido.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en memoria una instancia de Curso.
Post1: Se creó una instancia C de Curso en el sistema con C.habilitado = true.
Post2: Se liberó la memoria del sistema.

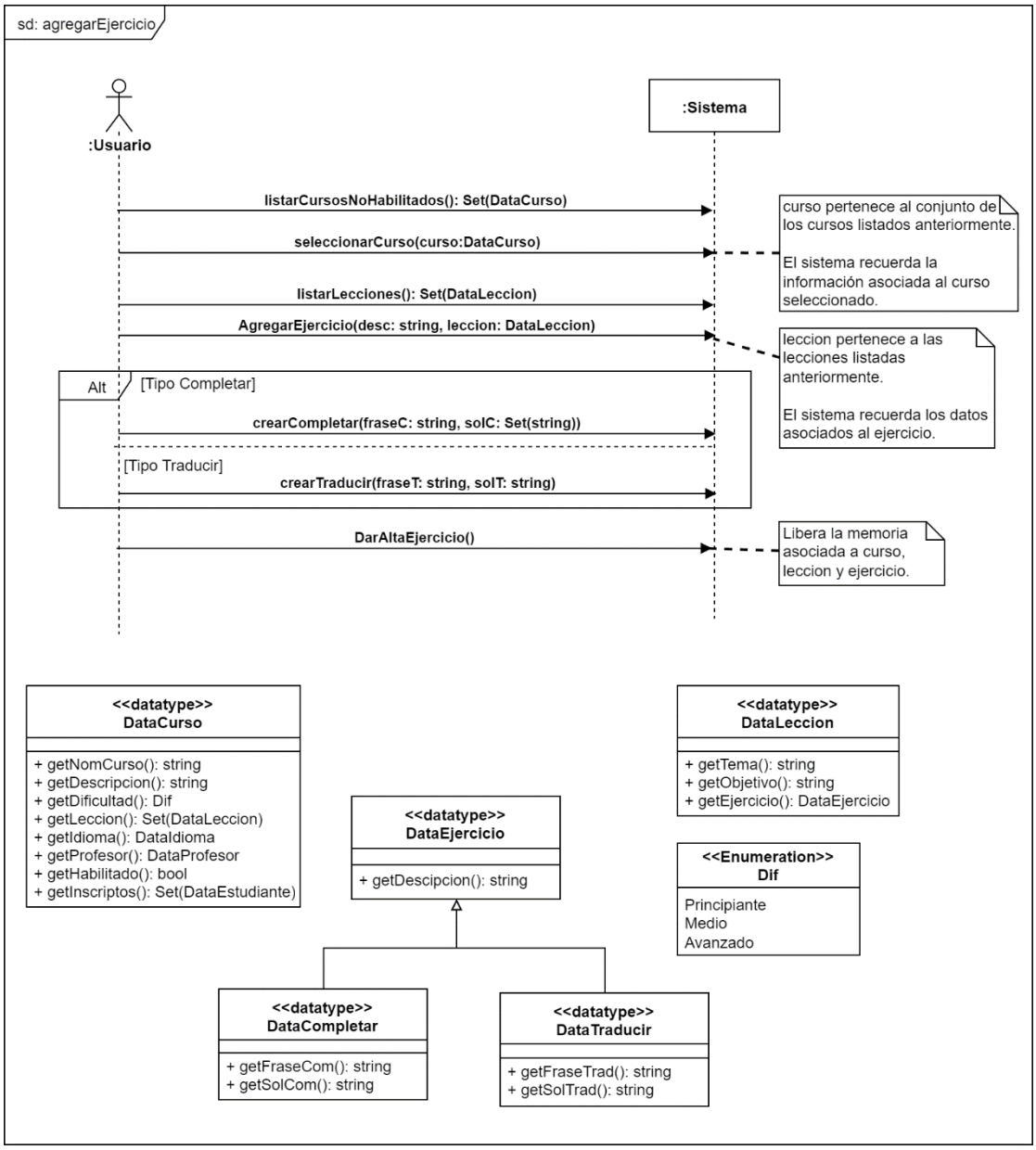
Agregar Lección

Diagramas de Secuencia del Sistema



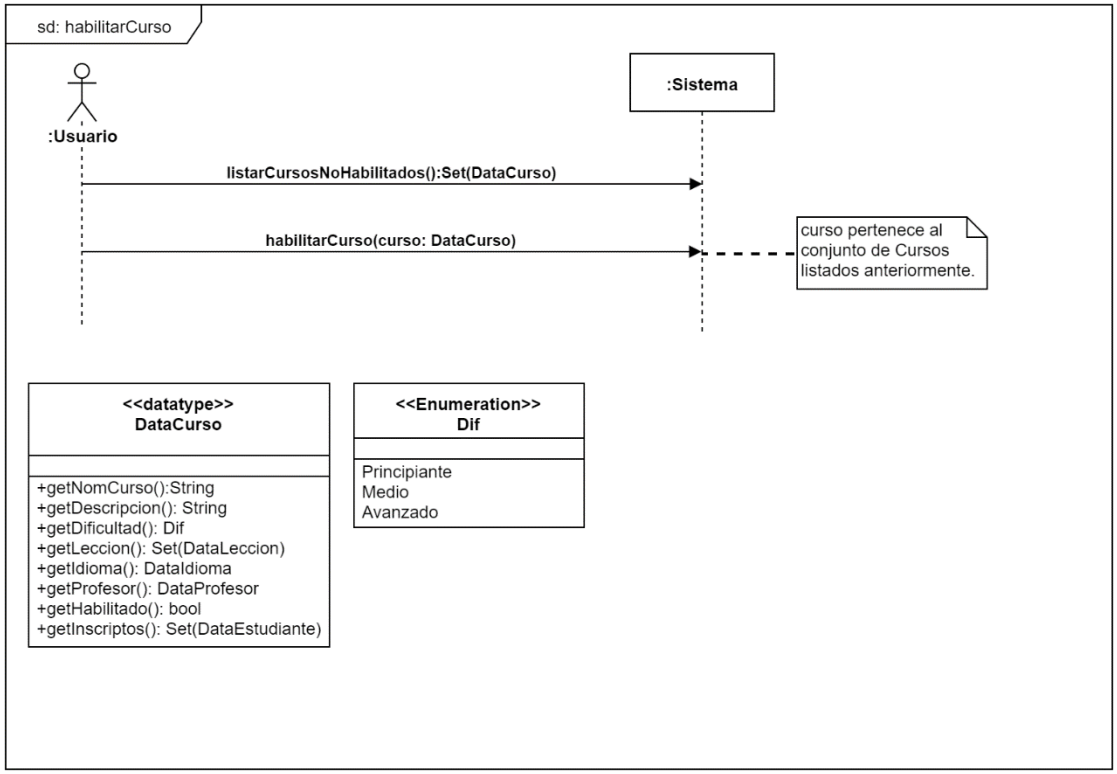
Agregar Ejercicio

Diagramas de Secuencia del Sistema



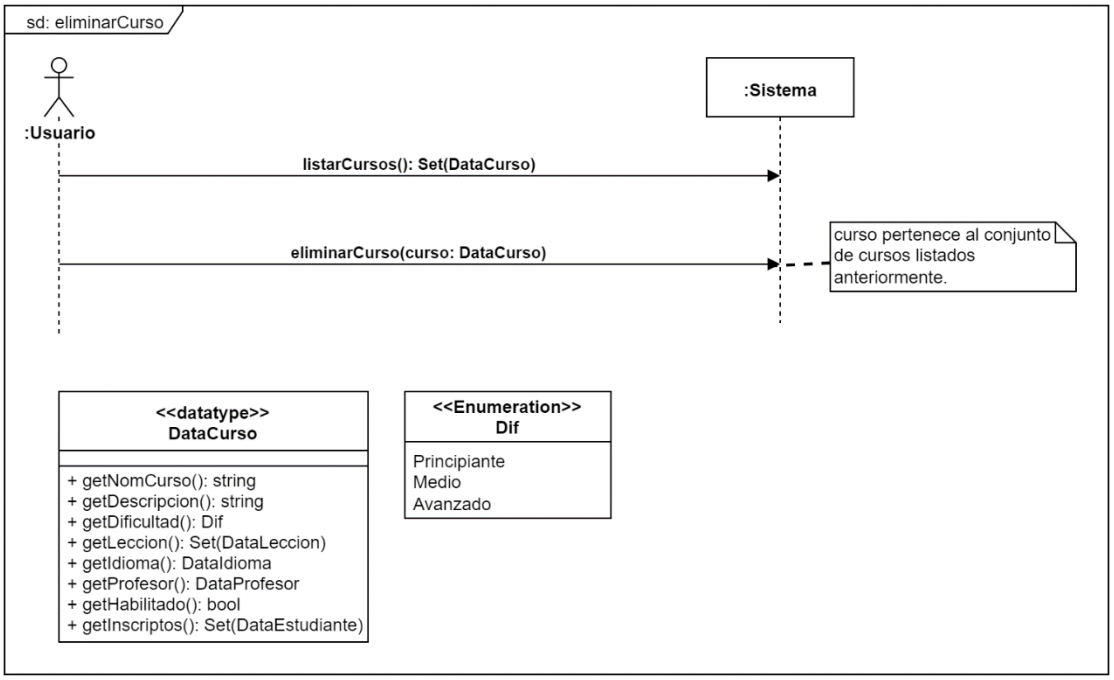
Habilitar Curso

Diagramas de Secuencia del Sistema



Eliminar Curso

Diagramas de Secuencia del Sistema



Contratos

Nombre	**Listar cursos.**
Operación	`listarCursos():Set(DataCurso)`
Entrada	No hay.
Salida	Un conjunto de DataCurso que representa todos los Cursos existentes en el sistema.
Descripción	Retorna todos los cursos existentes en el sistema.
Excepciones	No hay.

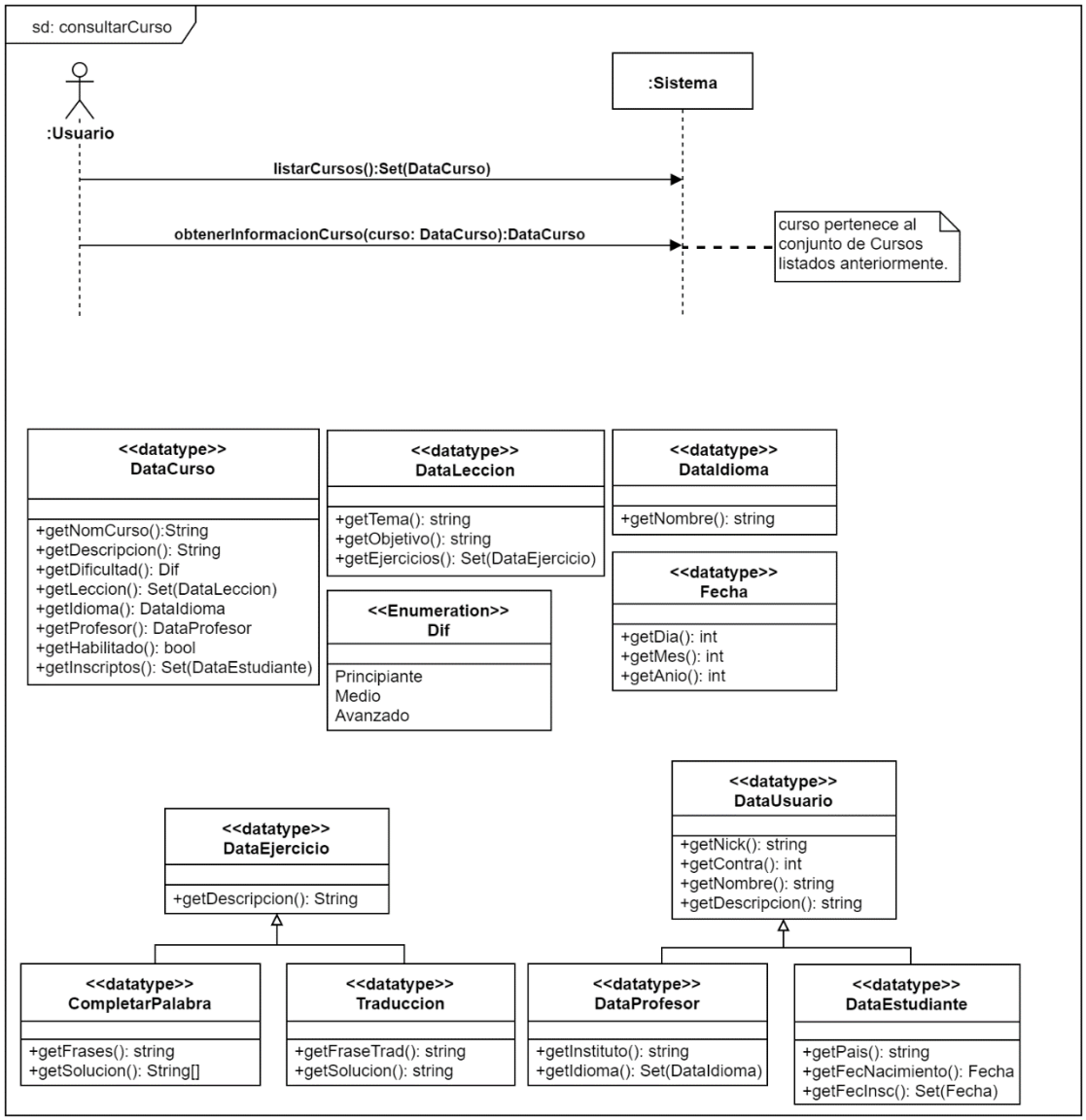
Precondiciones y Postcondiciones
Pre1: No hay. **Post1: No hay.**

Nombre	**Eliminar curso.**
Operación	`eliminarCurso(curso: DataCurso)`
Entrada	`curso` representa una instancia de un Curso existente en el sistema.
Salida	No hay.
Descripción	Elimina el curso seleccionado, así como todas sus lecciones, ejercicios, inscripciones y registros de aprobaciones de los estudiantes vinculados con el curso.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en sistema una instancia de Curso C tal que C == curso.
Post1: Se eliminó del sistema toda la información asociada a curso.

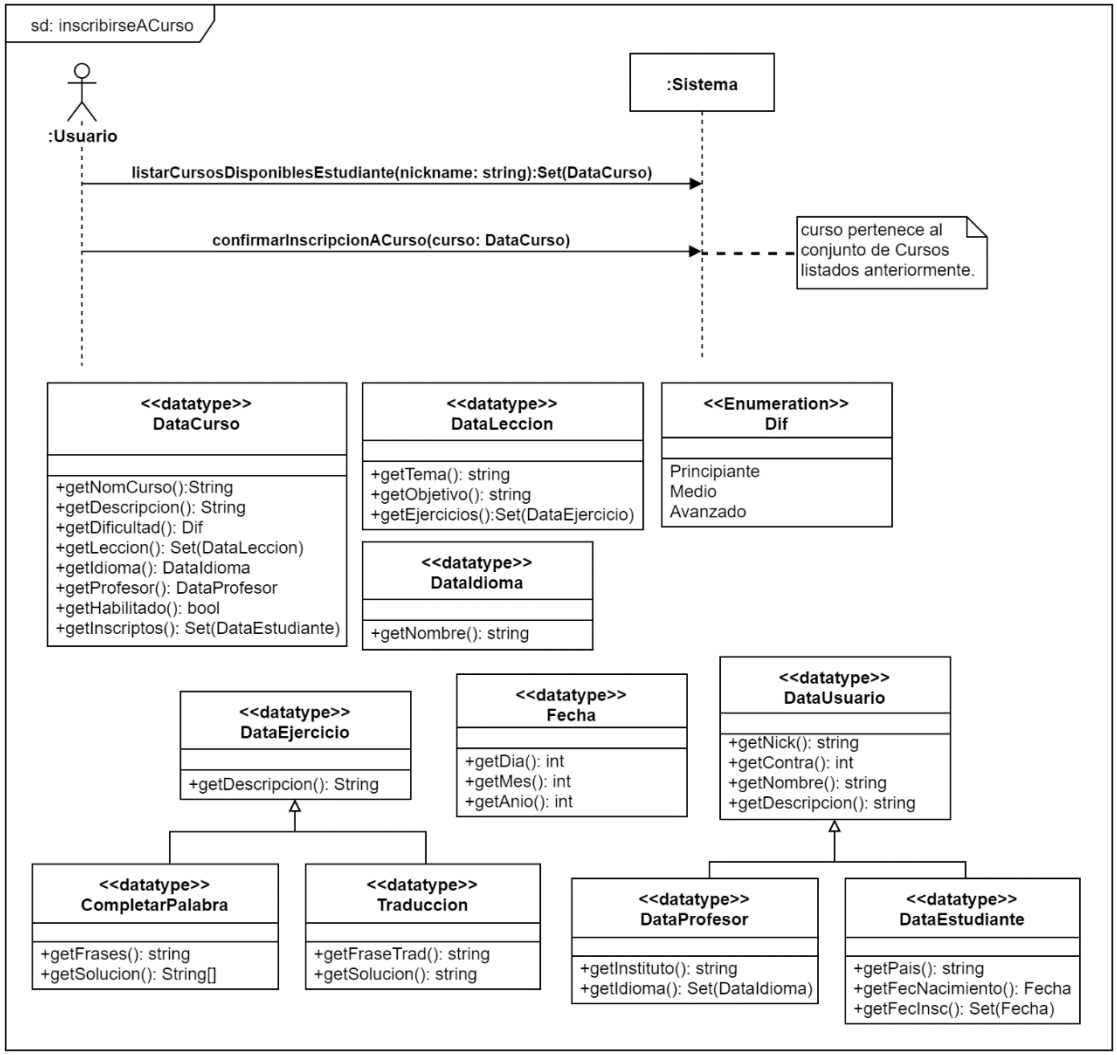
Consultar Curso

Diagramas de Secuencia del Sistema



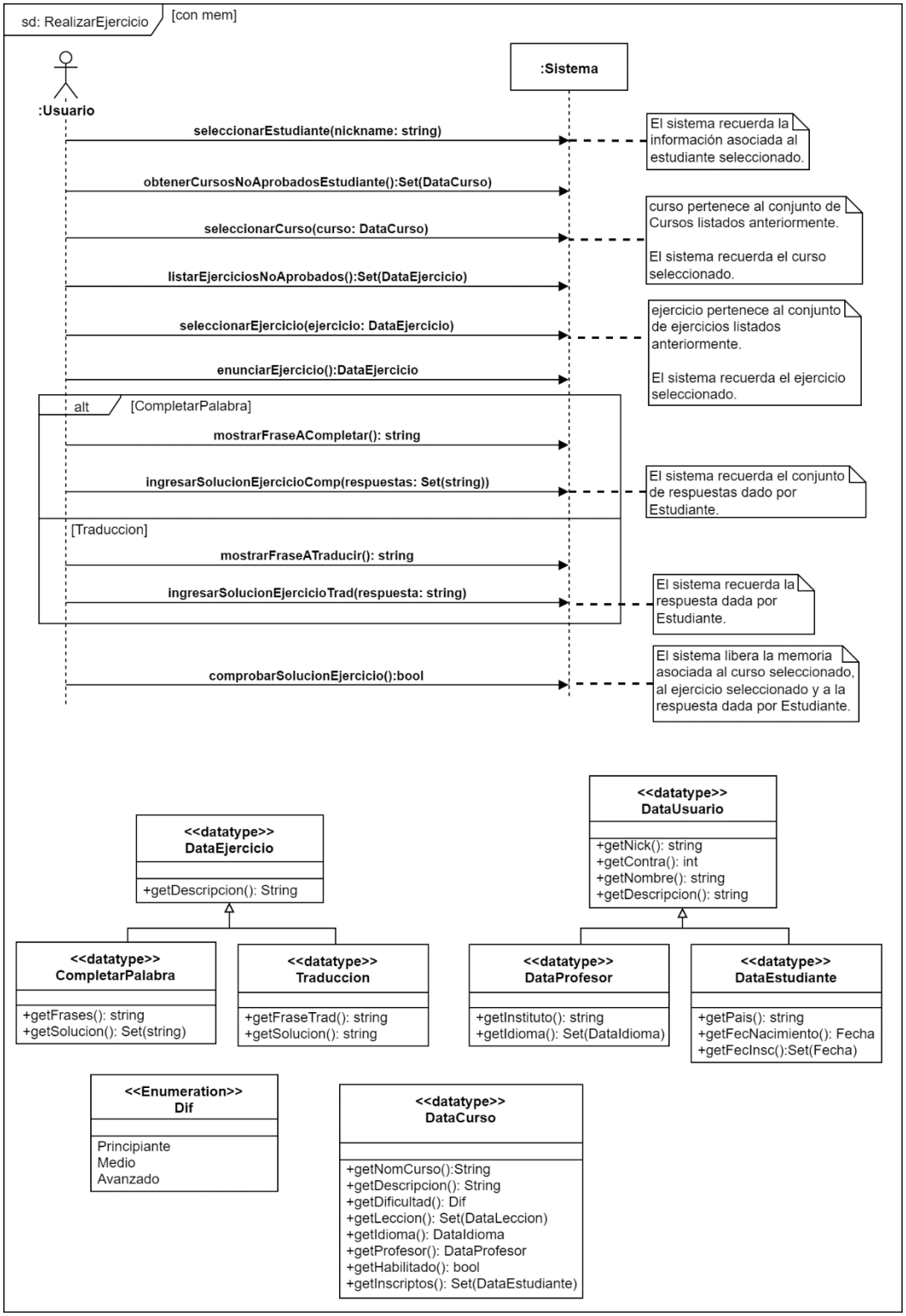
Inscribirse a Curso

Diagramas de Secuencia del Sistema



Realizar Ejercicio

Diagramas de Secuencia del Sistema



Contratos

Nombre	**Seleccionar estudiante.**
Operación	`seleccionarEstudiante(nickname: string)`
Entrada	`nickname` representa el Nickname de un Estudiante existente en el sistema.
Salida	No hay.
Descripción	Marca en la memoria del sistema un Estudiante para ser utilizado en futuras operaciones.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en el sistema una instancia de Estudiante E con E.Nick == nickname.
Post1: Se almacenó en la memoria del sistema la información asociada al Estudiante seleccionado.

Nombre	**Obtener cursos no aprobados del estudiante.**
Operación	`obtenerCursosNoAprobadosEstudiante():Set(DataCurso)`
Entrada	No hay.
Salida	Un conjunto de DataCurso que representa los Cursos existentes en el sistema a los cuales un Estudiante se encuentra inscripto y aún no ha aprobado.
Descripción	Retorna los cursos a los cuales el Estudiante marcado en memoria está inscripto y no ha aprobado.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en la memoria de sistema una instancia de Estudiante.
Post1: No hay.

Nombre	**Seleccionar Curso.**
Operación	`seleccionarCurso(curso: DataCurso)`
Entrada	`curso` representa una instancia de un Curso existente en el sistema.
Salida	No hay.
Descripción	Marca en la memoria del sistema un curso en el cual un determinado Estudiante se encuentra inscripto.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en sistema una instancia de Curso C tal que C == curso.
Post1: Se almacenó en la memoria del sistema la información asociada al Curso seleccionado.

Nombre	**Listar ejercicios no aprobados**
Operación	`listarEjerciciosNoAprobados(): Set(DataEjercicio)`
Entrada	No hay.
Salida	Un conjunto de DataEjercicio que representa los Ejercicios en la primera lección con ejercicios no aprobados por un estudiante del curso en memoria.
Descripción	Lista el conjunto de ejercicios no aprobados por el estudiante en memoria de la lección actual de un curso seleccionado existente en el sistema.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en la memoria del sistema una instancia de Curso.
Pre2: Existe en la memoria del sistema una instancia de Estudiante.
Post1: No hay.

Nombre	**Seleccionar Ejercicio**
Operación	`seleccionarEjercicio(ejercicio: DataEjercicio)`
Entrada	`ejercicio` es una instancia de CompletarPalabra o Traduccion, que existe en el sistema.
Salida	No hay.
Descripción	Marca en la memoria del sistema el ejercicio seleccionado por Usuario que el Estudiante en memoria aún no ha completado.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en la memoria del sistema una instancia de Ejercicio E tal que E == ejercicio.
Pre2: Existe en la memoria del sistema una instancia de Estudiante.
Post1: Se almacenó en la memoria del sistema la información asociada del Ejercicio seleccionado.

Nombre	**Enunciar ejercicio.**
Operación	`enunciarEjercicio():DataEjercicio`
Entrada	`No hay.`
Salida	Un elemento de tipo DataEjercicio para ser completado.
Descripción	Devuelve la descripción del ejercicio en memoria.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en la memoria del sistema una instancia de Ejercicio.
Post1: No hay.

Nombre	**Mostrar frase a completar.**
Operación	`mostrarFraseACompletar():string`
Entrada	`No hay.`
Salida	Un elemento de tipo string que representa la frase a completar.
Descripción	Devuelve la frase a ser completada del ejercicio en memoria.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en la memoria del sistema una instancia de Ejercicio.
Post1: No hay.

Nombre	**Ingresar Solución de ejercicio de tipo completar.**
Operación	`ingresarSolucionEjercicioComp(respuestas: Set(string))`
Entrada	`respuestas` es un conjunto de string que representa la solución dada por Usuario a un determinado Ejercicio almacenado en memoria.
Salida	No hay.
Descripción	Almacena en la memoria del sistema el conjunto respuesta para un determinado ejercicio almacenado en memoria.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en la memoria del sistema una instancia de Ejercicio.
Post1: Se almacenó en la memoria del sistema los valores de respuestas.

Nombre	**Mostrar frase a traducir.**
Operación	`mostrarFraseATraducir():string`
Entrada	`No hay.`
Salida	Un elemento de tipo string que representa la frase a traducir.
Descripción	Devuelve la frase a ser traducida del ejercicio en memoria.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en la memoria del sistema una instancia de Ejercicio.
Post1: No hay.

Nombre	**Ingresar Solución de ejercicio de tipo traducir.**
Operación	`ingresarSolucionEjercicioTrad(respuesta: String)`
Entrada	`respuesta` representa la solución dada por Usuario al ejercicio marcado en memoria.
Salida	No hay.
Descripción	Almacena en la memoria del sistema la respuesta para el ejercicio almacenado en memoria.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en la memoria del sistema una instancia de Ejercicio.
Post1: Se almacenó en la memoria del sistema el valor de respuesta.

Nombre	**Comprobar solución de ejercicio.**
Operación	`comprobarSolucionEjercicio():bool`
Entrada	No hay.
Salida	Retorna un elemento de tipo bool que representa si el Estudiante respondió o no correctamente al ejercicio.
Descripción	El sistema verifica la solución ingresada por el Estudiante en memoria con la solución del ejercicio almacenado en memoria, si coinciden, el ejercicio es marcado como completado por Estudiante.
Excepciones	No hay.

Precondiciones y Postcondiciones
Pre1: Existe en la memoria del sistema una instancia de Ejercicio.
Pre2: Existe en la memoria del sistema una instancia de Estudiante.
Pre3: Existe en la memoria del sistema una solución dada por Estudiante.
Post1: Si se retornó true, se marcó como completado el ejercicio el cual se realizó, de lo contrario, no.
Post2: Se liberó la memoria del sistema.

Consultar Estadísticas

Diagramas de Secuencia del Sistema