Diseño de sistemas multiagente

Prometheus

Javier Béjar

ECSDI - 2021/2022 2Q

CS-GEI-FIB 

Introducción

- **Prometheus** es una metodología iterativa que cubre el proceso completo de ingeniería de software
  - Análisis
  - Diseño
  - Diseño detallado
  - Implementación

- Su objetivo es el desarrollo de agentes inteligentes, orientado en particular a agentes BDI
  - Usa como abstracciones objetivos, creencias, planes y eventos
- La especificación resultante puede ser desarrollada en cualquier implementación de agentes que cubra esas abstracciones
  - En particular está pensada para la implementación con el lenguaje de agentes JACK

- Es una metodología que ha evolucionado a partir de la experiencia práctica en el desarrollo de sistemas multiagente
- Está enfocada al desarrollo de software en la industria
- Ha recibido la experiencia de uso en proyectos tanto a pequeña escala (estudiantes) como en la industria (AOS Group http://aosgrp.com/)

- Esta enfocada en dar una guía y estructuración detallada para facilitar el desarrollo de herramientas de diseño
- Es una mezcla de:
  - Una notación gráfica que facilita una vista general del sistema
  - Una notación textual estructurada para una vista detallada
- Es jerárquica y modular
- Existe un prototipo de herramienta para el diseño (PDTool)

- Estamos diseñando sistemas distribuidos (no monolíticos)
- El sistema interactúa con múltiples entidades externas (también distribuidas) al mismo tiempo
- El sistema puede tener múltiples réplicas de cada entidad que lo compone
- Las entidades internas y externas pueden aparecer y desaparecer dinámicamente
- Las acciones que realiza el sistema no han de ser síncronas
- Puede haber múltiples flujos de ejecución a la vez

Fases de diseño

(1) Especificación del sistema:

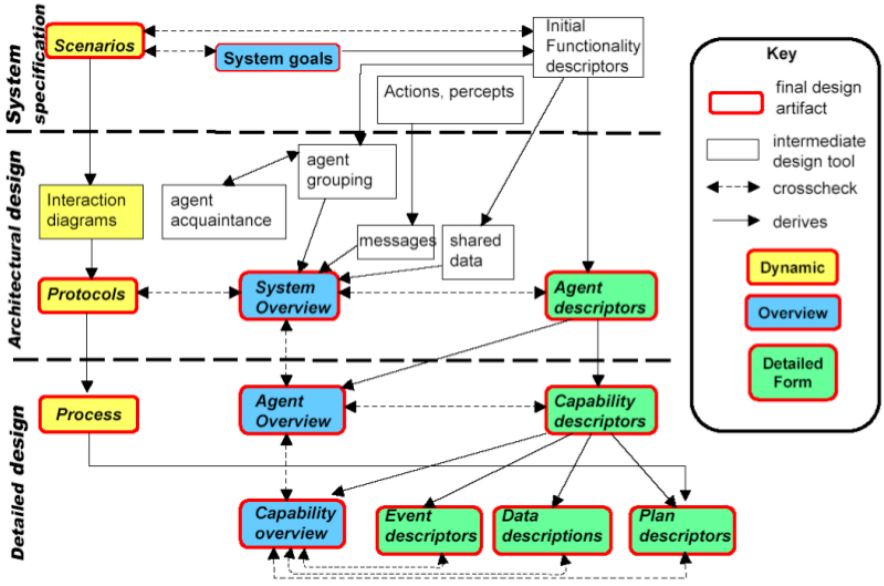
- Focalizada en la identificación de las funciones básicas del sistema:
  - Roles (funcionalidades)
  - Objetivos
  - Escenarios
  - Entradas (*percepts*), salidas (*actions*) y su procesamiento

(2) Diseño arquitectónico:

- Detemina qué agentes contendrá el sistema y cómo interaccionarán

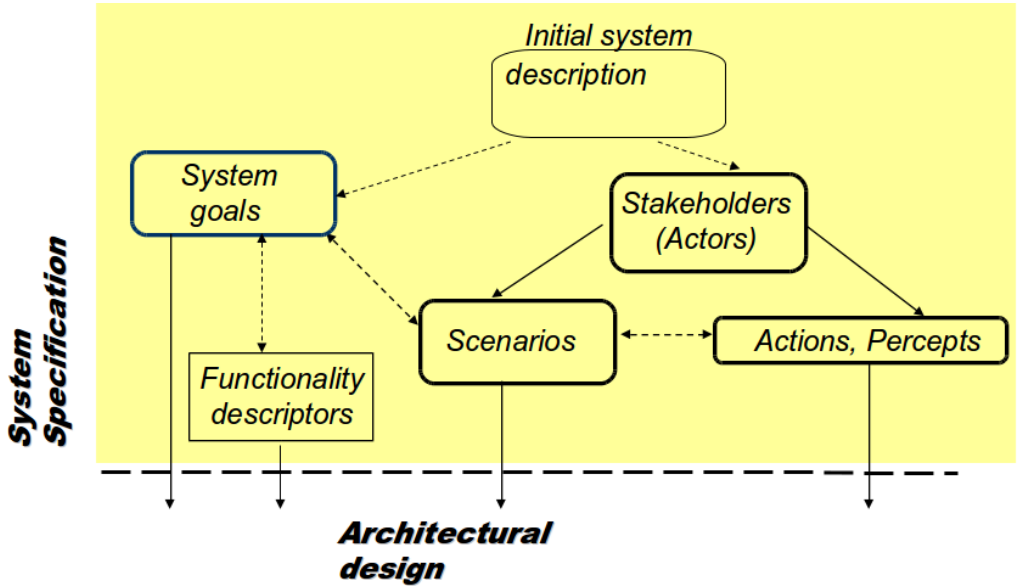
(3) Diseño detallado:

- Describe los elementos internos de los agentes
- La forma en la que realizarán sus tareas dentro del sistema
- Enfoque en describir capacidades (*capabilities*, módulos del agente), eventos internos, planes y estructuras de datos internas


System specification
Architectural design
Detailed design
Scenarios
System goals
Initial Functionality descriptors
Actions, percepts
agent grouping
Interaction diagrams
agent acquaintance
messages
shared data
Protocols
System Overview
Agent descriptors
Process
Agent Overview
Capability descriptors
Capability overview
Event descriptors
Data descriptions
Plan descriptors
Key
final design artifact
intermediate design tool
crosscheck
derives
Dynamic
Overview
Detailed Form

Especificación del sistema

- Sistema definido por:
  - Objetivos: Diagrama de objetivos
  - Escenarios: Escenarios de casos de uso
  - Roles: Descriptores de funcionalidades
- La interacción con el entorno se describe en términos de:
  - Acciones
  - Percepciones
  - Datos externos


Initial system description
System goals
Stakeholders (Actors)
Scenarios
Actions, Percepts
Functionality descriptors
System Specification
Architectural design

- Comenzar con una descripción a alto nivel del sistema (textual)
- Identificar los actores
- Identificar los objetivos del sistema (y subobjetivos)
- Identificar los roles (funcionalidades) que cubren los objetivos
- Identificar/definir escenarios
- Identificar entradas y salidas (percepciones y acciones)

¡Atención!

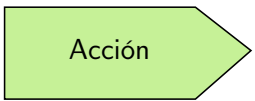
¡No es un proceso secuencial!

Cada elemento realimenta/se interrelaciona con los otros

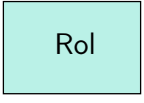
- ¿Para que se construye el sistema? Objetivos principales
- ¿Cuales son los subobjetivos que permiten conseguirlos?
- Descritos en el diagrama de objetivos
- **¿Cómo?** (subobjetivos)
- **¿Porqué?** (objetivos padre)

- Los agentes están en un entorno y recibirán información de él (**percepciones**)
- Habrá cosas que sucederán en el entorno que serán significativas para los agentes (**eventos**)
- Estos pueden ser directamente percepciones u obtenerse de percepciones después de un procesamiento

- Las **acciones** se hacen para modificar el entorno
- Pueden ser simples y directas o interacciones complejas
- Preguntas:
  - ¿Tienen una duración?
  - ¿Pueden fallar?
  - ¿Sabremos si lo hacen?
  - ¿Si fallan tendrán un efecto en el entorno?

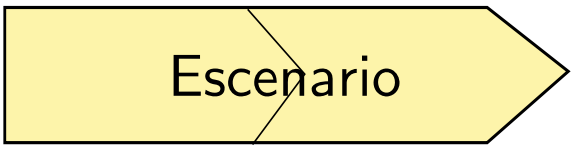


- Los **roles** describen funcionalidades que el diseñador piensa que debe tener el sistema
- Deben ser específicas y poder definirse en una o dos frases
- Los roles estarán ligados a los objetivos
  - Puede haber más de un objetivo por rol
  - Un rol puede aparecer en más de un objetivo
- Hay unos descriptores que deben usarse para detallarlos



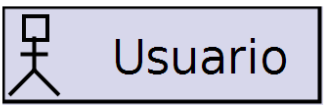
- Los **escenarios** muestran una instancia particular de la ejecución del sistema (alternativas son escenarios separados o variaciones)
- Un escenario consiste en una secuencia de pasos (acciones, percepciones, objetivos, escenarios)
- Cada paso está ligado a un rol y a los datos usados y producidos

- Cada objetivo del sistema debe aparecer en al menos un escenario
- Escenarios habitualmente involucran más de un rol (funcionalidad)



- Queremos un sistema que sea capaz de manejar los requerimientos de una biblioteca, básicamente:
  - Permitir sacar libros, informando al usuario de la fecha de retorno
  - Permitir retornar libros
  - Permitir la reserva de libros aún no disponibles
  - Permitir la notificación de libros prestados que están fuera de plazo
  - Permitir la notificación de llegada de libros reservados

- Podemos identificar dos actores
  - Biblioteca
  - Usuario
- No tienen porque ser agentes en el diseño final
- También pueden ser actores que no modelemos, pero que queramos tener para representar, por ejemplo, los elementos del entorno que generan las percepciones o reciben las acciones



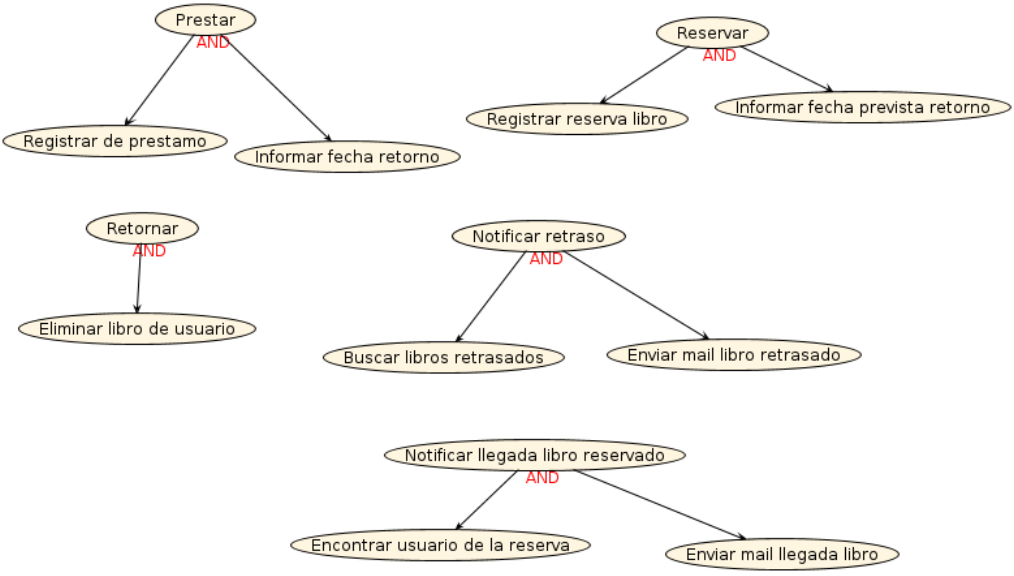
- Podríamos identificar como objetivos de alto nivel del sistema:
  - Prestar libros (notificando fecha retorno)
  - Retornar libros
  - Reservar libros no disponibles
  - Notificar libros fuera de plazo
  - Notificar la llegada de libros reservados

Para cada uno de los objetivos nos preguntamos **cómo** son obtenidos

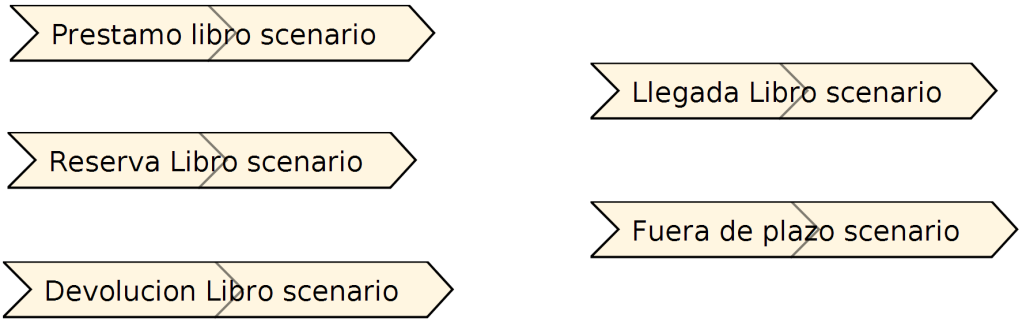
- Prestar libro $\Longrightarrow$
  - Registrar ID del libro a ID del usuario
  - Informar de la fecha de retorno
- Retornar libro $\Longrightarrow$
  - Borrar ID de libro del ID de usuario
- Reservar libros no disponibles $\Longrightarrow$
  - Registrar ID del libro como reservado por ID del usuario
  - Informar de la fecha de retorno del libro al usuario

Para cada uno de los objetivos nos preguntamos **cómo** son obtenidos

- Notificar libro fuera de plazo $\Longrightarrow$
  - Acceder al registro de libros al inicio del día
  - Enviar correo para los libros no entregados a tiempo
- Notificar la llegada de libros reservados $\Longrightarrow$
  - Acceder a la lista de libros reservados por los usuarios
  - Enviar correo informando de la llegada del libro


Prestar
AND
Registrar de prestamo
Informar fecha retorno
Reservar
AND
Registrar reserva libro
Informar fecha prevista retorno
Retornar
AND
Eliminar libro de usuario
Notificar retraso
AND
Buscar libros retrasados
Enviar mail libro retrasado
Notificar llegada libro reservado
AND
Encontrar usuario de la reserva
Enviar mail llegada libro

1. Cuando el usuario viene a pedir un libro (Préstamo Libro)
2. Cuando el usuario retorna un libro (Devolución Libro)
3. Cuando un libro está fuera de plazo (Fuera de plazo)
4. Cuando el usuario pide reservar un libro (Reserva Libro)
5. Cuando el libro reservado llega (Llegada libro)



- Subescenarios/variantes: Descomposición del escenario o variantes
- Objetivos: Cuáles son los objetivos que se persiguen en este escenario
- Percepciones: Qué eventos exteriores ponen en marcha el escenario
- Actividades y acciones: Cuáles son las funcionalidades que se desarrollan dentro del escenario
- Mensajes: Cuáles son las comunicaciones (generador $\longrightarrow$ receptor) que se realizan dentro del escenario
- Roles: Cuáles son los roles que participan en el escenario

Podemos esbozar los pasos que componen cada escenario

- **Préstamo Libro**
  1. Petición de libro
  2. Dar fecha de retorno
  3. Registrar el código de libro como prestado
  4. Dar el libro
- **Devolución Libro**
  1. Devolver libro
  2. Borrar el código del libro de los prestados
- ...

1. Procesador de préstamos de libros
2. Procesador de libros retornados
3. Procesador de libros fuera de plazo
4. Procesador de reserva de libros
5. Procesador de llegada de libros reservados

Los roles pueden estar asignados al mismo actor/agente o varios pueden compartir roles

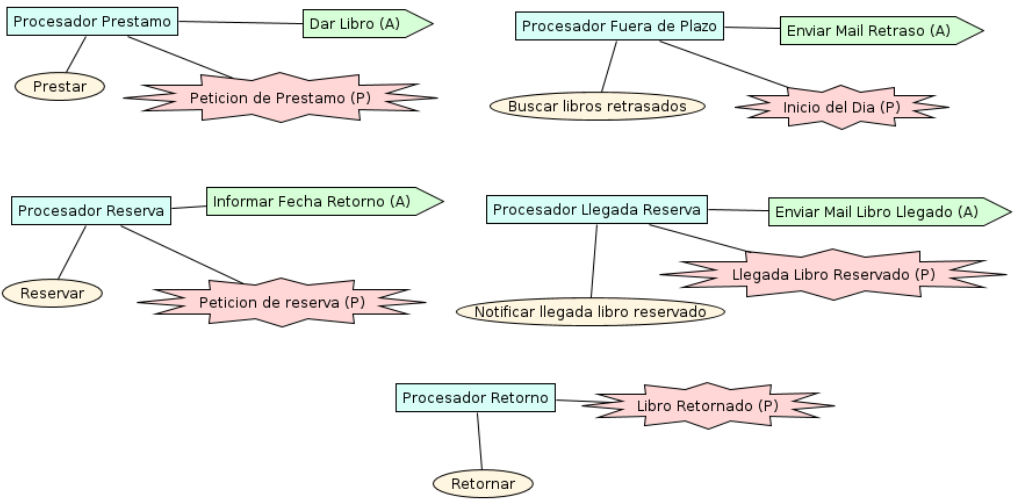
Hemos de identificar las entradas y salidas del sistema en forma de percepciones y acciones y asignarlas a los roles

- Préstamo de libros:
  - **Percepción**: Petición de libro
  - **Acción**: Dar libro
- Retorno de libro:
  - **Percepción**: Libro retornado
- Libros fuera de plazo
  - **Percepción**: Inicio del día
  - **Acción**: Enviar correo de libro retrasado

Hemos de identificar las entradas y salidas del sistema en forma de percepciones y acciones y asignarlas a los roles

- Reserva de libro
  - **Percepción**: Petición de reserva
  - **Acción**: Informar de fecha de retorno del libro
- Llegada de libros reservados
  - **Percepción**: Llegada de libro reservado
  - **Acción**: Enviar correo de llegada de libro

- Hemos de ligar los objetivos con los escenarios y los roles
- Hemos de ligar las percepciones y acciones a roles



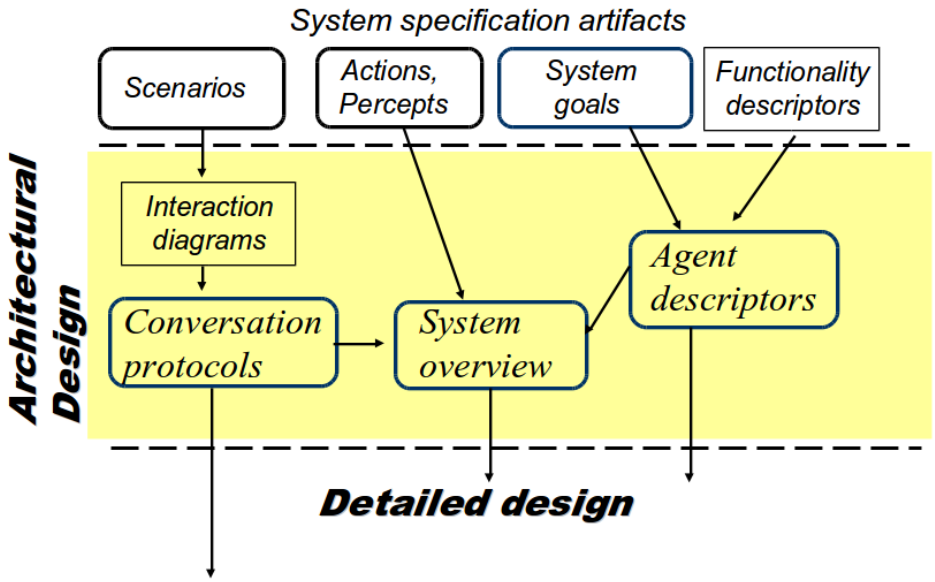
Rol	
Nombre	Procesador Fuera de Plazo
Descripción	Se encarga de informar a los usuarios que no han devuelto un libro a tiempo
Evento Iniciador	Inicio del dia
Acciones	Enviar un mail avisando del retraso
Información usada	Fecha de retorno de los libros prestados
Información Producida	ninguna
Objetivos	Notificar retraso

Escenario			
Nombre	Fuera de plazo		
Descripción	Hay libros prestados que no han sido retornados a tiempo		
Evento	Inicio del día		
Pasos			
Tipo	Nombre	Rol	Datos
Percepción	Inicio Día	P. Libro fuera Plazo	
Objetivo	Notificar Retraso	P. Libro fuera Plazo	
Objetivo	Buscar Libros Retrasados	P. Libro fuera Plazos	Libros Prestado
Objetivo	Enviar Mail Libro Retrasado	P. Libro fuera Plazos	
Acción	Enviar e-mail	P. Libro fuera Plazo	e-mail usuario

Percepción	
Nombre	Inicio del día
Descripción	Evento que indica el inicio de un nuevo día
Información	Ninguna
Conocimiento Actualizado	Hay un nuevo día
Fuente	Entorno
Procesamiento	Ninguno
Agentes que responden	Por determinar
Frecuencia	Una vez al día

Acción	
Nombre	Enviar mail retraso
Descripción	Enviar un mail a un usuario que no ha retornado un libro
Parámetros	E-mail usuario, código del libro
Duración	Instantáneo
Fallo	Puede no llegar al usuario
Efectos Laterales	Ninguno

Diseño Arquitectónico


System specification artifacts
Scenarios
Actions, Percepts
System goals
Functionality descriptors
Architectural Design
Interaction diagrams
Conversation protocols
System overview
Agent descriptors
Detailed design

- ¿Cuando un componente del sistema es un agente?
  - ¿Es autónomo?
  - ¿Tiene objetivos?
  - ¿Es activo? (tiene procesos internos en ejecución de manera concurrente)
  - ¿Hace varias cosas a la vez? ¿Debe razonar respecto a la interacción entre ellas?
  - ¿Debe cambiar la manera de hacer las cosas basándose en cambios en el entorno?
- Si la respuesta es mayoritariamente sí, deberíamos modelarlo como un agente

- Agrupar las funcionalidades en tipos de agentes basándose en:
  - Cohesión (forman parte del mismo contexto)
  - Acoplamiento (tienen dependencias funcionales fuertes)
- Agrupar funcionalidades que:
  - Están relacionadas según el sentido común
  - Requieren la misma información

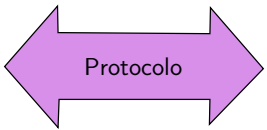
- No agrupar funcionalidades que
  - No están claramente relacionadas
  - Han de ejecutarse en plataformas separadas
  - No pueden compartir información por seguridad/privacidad
  - Han de ser modificadas por diferentes desarrolladores

- ¿Cuántos agentes de cada tipo debe haber (uno, muchos, dinámico)?
- ¿Cuál es el tiempo de vida del agente (permanente/ocasional)?
- ¿Cuál es el estado inicial del agente?
- ¿Qué pasa cuando un agente desaparece del sistema?
- ¿Cuáles son los datos usados/producidos por el agente?
- ¿A qué eventos reaccionan los agentes?
- ¿Cuáles son sus interacciones? ¿Cómo se realizan?

- Usar fuentes de datos centralizadas destruye el propósito de hacer un sistema distribuido (introduce acoplamientos) hay que buscar la máxima localidad
- Los roles/funcionalidades ven los datos desde diferentes perspectivas, eso permite partirlos
- Diferentes roles/funcionalidades tienen diferentes derechos de acceso a los datos
- No es un problema que ciertos datos estén replicados si eso ayuda a la distribución del sistema
- En la práctica el mantener los datos de un sistema distribuido es complejo y necesita sopesar ventajas y desventajas de las alternativas
- **CAP theorem** (Consistency/Availability/Partition Tolerance)

- El diagrama de **agrupación de roles a agentes**
- El diagrama de **relación entre agentes**
- El diagrama de **acoplamiento entre datos y roles**

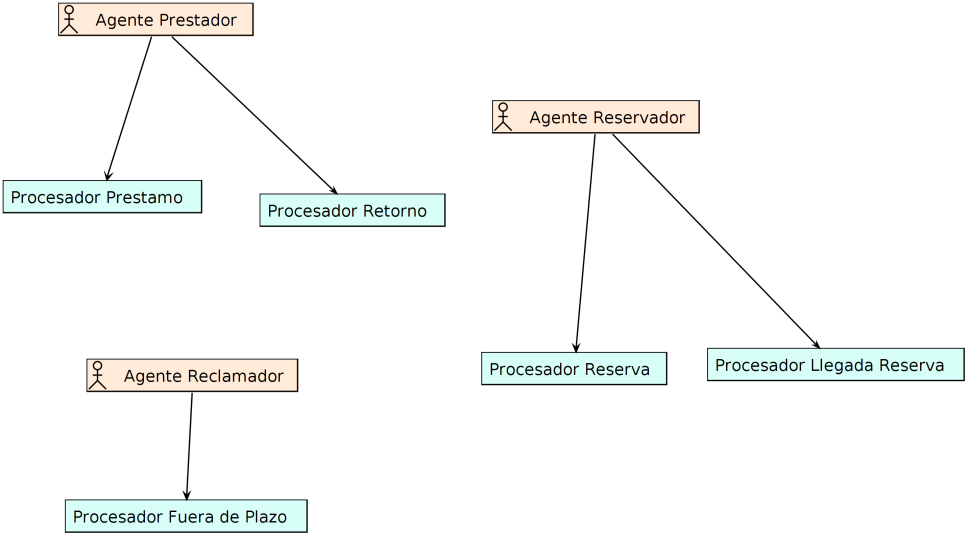
- Identificamos y diseñamos la interacción entre los agentes
- Usamos como base los pasos de los escenarios:
  - Si un paso involucra un rol asignado a un agente y el siguiente involucra otro rol asignado a otro agente, tendrá que haber un intercambio de mensajes
- Definir los protocolos de interacción



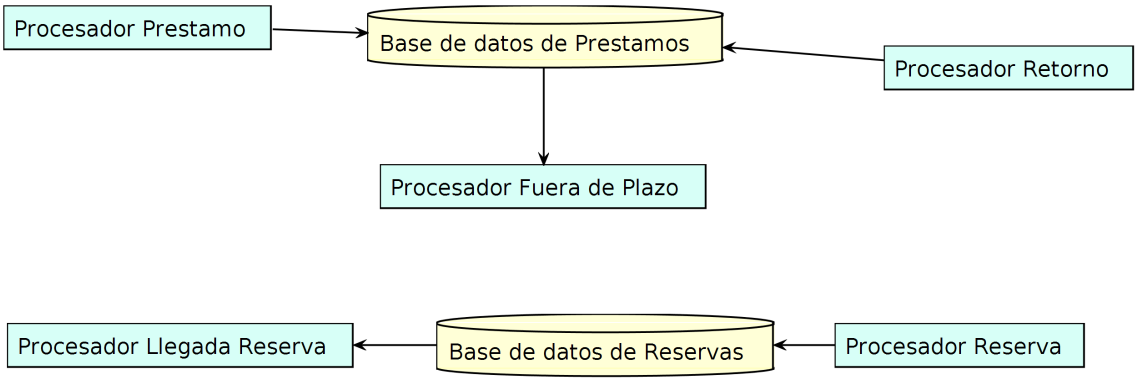
- Identificamos como creencias (datos) que han de manejar los agentes del sistema la información de todos los libros prestados y reservados
- El rol del procesador de préstamo y procesador de retorno modifican la creencia del préstamo de un libro
- El rol de procesador de fuera de plazo utiliza esa creencia para realizar su acción

- El rol de procesador de reserva modifica la creencia de la reserva de un libro
- El rol de procesador de llegada de reserva usa esta creencia para realizar su acción

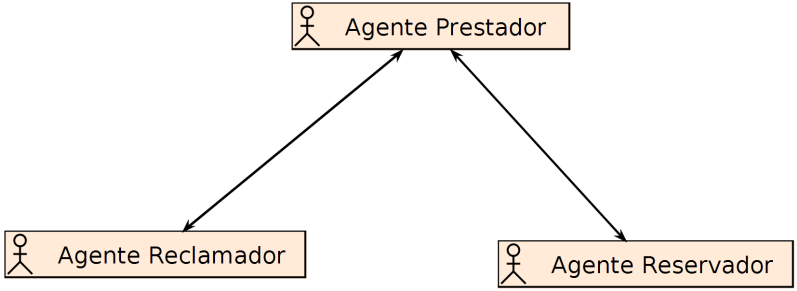
- Identificamos tres agentes diferentes:
  - **Prestador**
  - **Reclamador**
  - **Reservador**
- Agrupamos los roles según su coherencia semántica


Agente Prestador
Procesador Prestamo
Procesador Retorno
Agente Reservador
Procesador Reserva
Procesador Llegada Reserva
Agente Reclamador
Procesador Fuera de Plazo

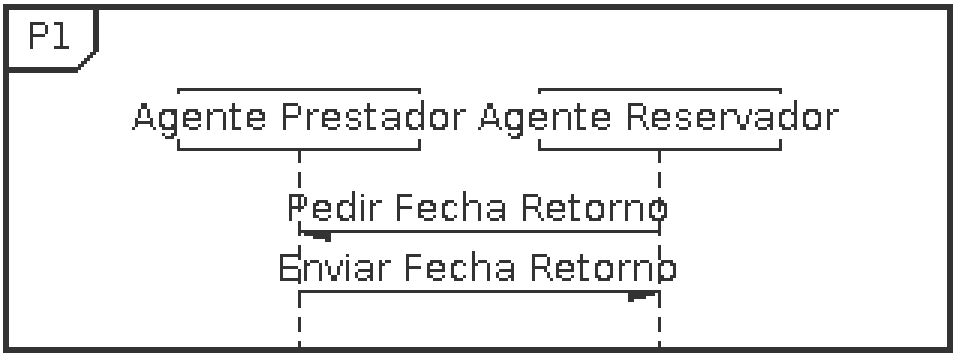
- Identificamos dos fuentes de datos (préstamos y reservas)
- Asignamos los roles a las fuentes de datos indicando en que dirección (entrada/salida) va la relación
- No tienen por qué corresponder a fuentes de datos físicamente separadas
- Tampoco una fuente de datos tiene por qué corresponder a una fuente de común entre todos los roles


Procesador Prestamo
Base de datos de Prestamos
Procesador Retorno
Procesador Fuera de Plazo
Procesador Llegada Reserva
Base de datos de Reservas
Procesador Reserva

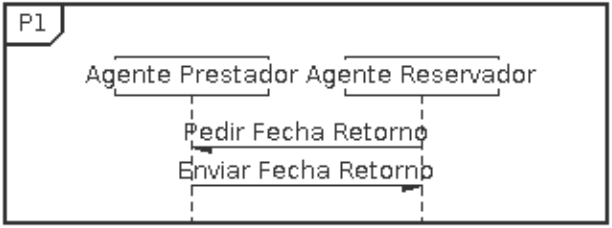
- Decidimos que los agentes se vayan comunicando
  - Reservador pide a Prestador las fechas de devolución de los libros prestados cuando se piden
  - Prestador informa de la fecha de devolución de los libros prestados
- Esto relaciona a los agentes prestador con el reservador y el prestador con el reclamador



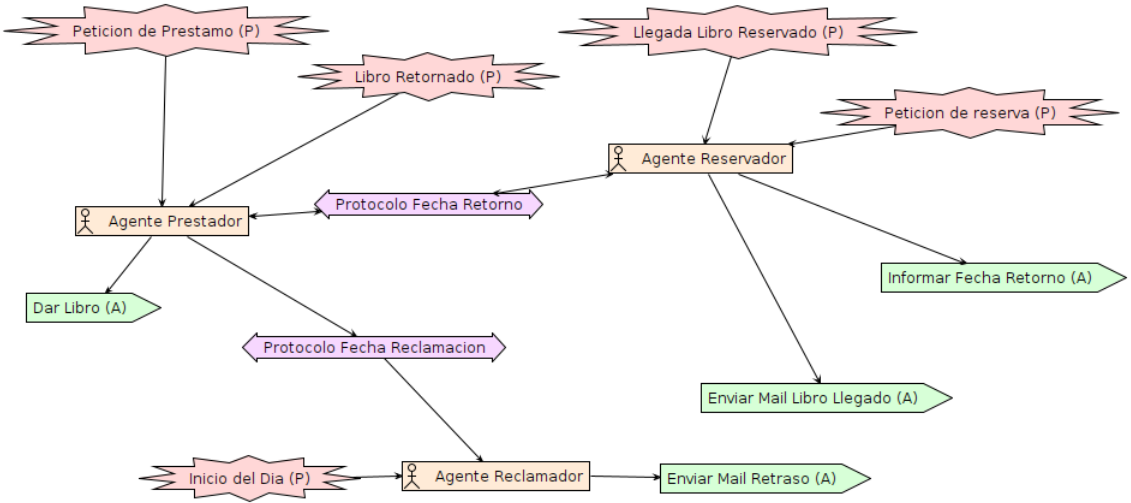
- La comunicación entre los agentes nos obligará a definir:
  - Mensajes: ¿Cuál es su contenido y su significado?
  - Protocolos: ¿Quién interviene en el protocolo? ¿Cómo se realiza el intercambio de mensajes?



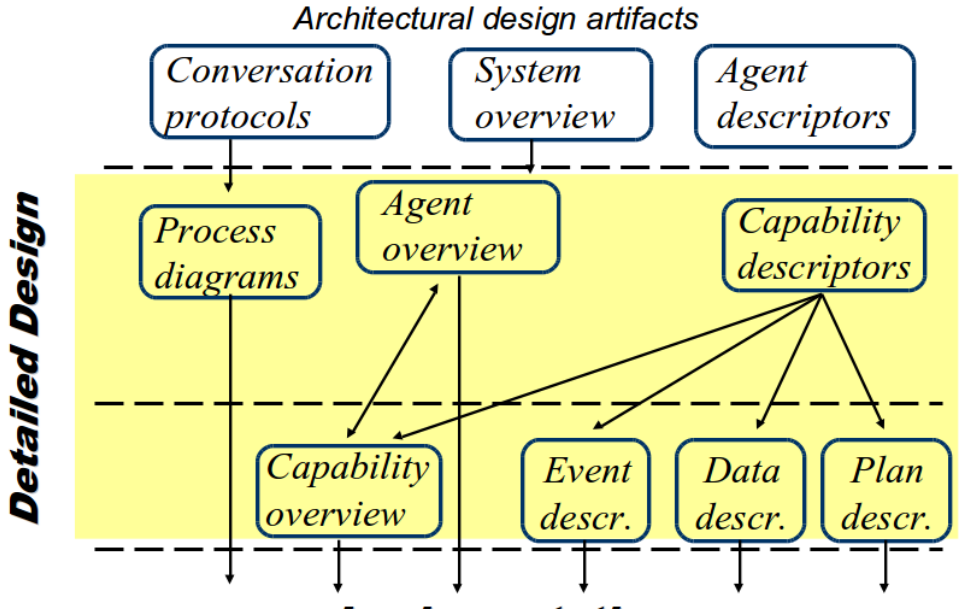
Agente			
Nombre	Agente Reclamador		
Descripción	Se encarga de procesar los libros retrasados		
Cardinalidad mínima	1	**Cardinalidad máxima**	1
Duración	ilimitada	**Inicialización**	Ninguna
En caso de fallo	Nada	**Percepciones**	Inicio de día
Acciones	Enviar mail retraso a usuario	**Usa datos**	Libros prestados a usuarios
Produce datos	Ninguno	**Datos internos**	Por definir
Objetivos	Notificar retraso...	**Roles**	Procesador Fuera plazo
Protocolos	Protocolo Fecha		

Protocolo	
Nombre	Pedir Fecha
Descripción	Obtener la fecha de retorno de un libro
Mensajes	Pedir Fecha Retorno, Enviar Fecha Retorno
Escenarios	Fuera de plazo, Reserva libro
Agentes	Prestador, Reservador, Reclamador
Notas	

Mensaje	
Nombre	Pedir Fecha Retorno
Descripción	Pedir la fecha de retorno de un libro
Distribución	De Reservador a Prestador, De Reclamador a Prestador
Propósito	Obtener la fecha de retorno de un libro
Información contenida	Identificador del usuario y libro


Peticion de Prestamo (P)
Libro Retornado (P)
Llegada Libro Reservado (P)
Peticion de reserva (P)
Agente Reservador
Agente Prestador
Protocolo Fecha Retorno
Informar Fecha Retorno (A)
Dar Libro (A)
Protocolo Fecha Reclamacion
Enviar Mail Libro Llegado (A)
Inicio del Dia (P)
Agente Reclamador
Enviar Mail Retraso (A)

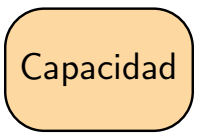
Diseño Detallado


Architectural design artifacts
Conversation protocols
System overview
Agent descriptors
Detailed Design
Process diagrams
Agent overview
Capability descriptors
Capability overview
Event descr.
Data descr.
Plan descr.

- En esta fase se desarrollan los **elementos internos** de los agentes
  - Se definen en función de capacidades, datos, eventos y planes
  - Se usan diagramas de proceso como un paso intermedio entre protocolos de interacción y planes

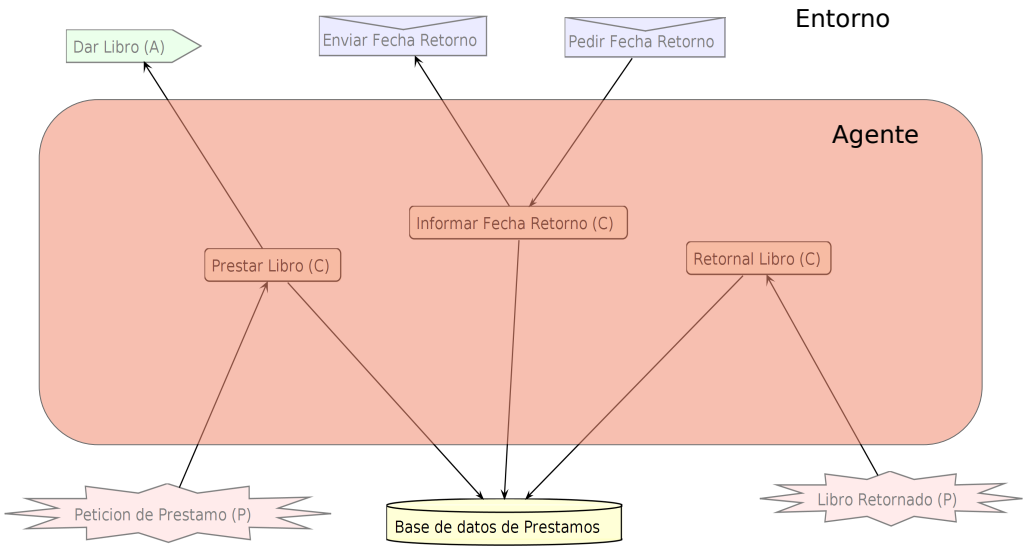
- Desarrollar la estructura interna de cada agente individual
- Identificar las capacidades de cada agente empezando con sus funcionalidades
- Generar los descriptores de capacidades
- Generar los diagramas de visión general del agente
- Generar los descriptores de los planes
- Generar los descriptores de los eventos
- Generar los descriptores de los mensajes externos e internos
- Generar los descriptores de los datos

- Las **capacidades** se definen como módulos (un trozo de código que hace una cosa en particular)
- Se parte de los roles (funcionalidades) de la primera fase, dividiéndolos hasta obtener capacidades primitivas
- Las funcionalidades de bajo nivel que son comunes entre varios agentes pueden usarse también como capacidades
- Hay que tener presente la reusabilidad al determinarlas

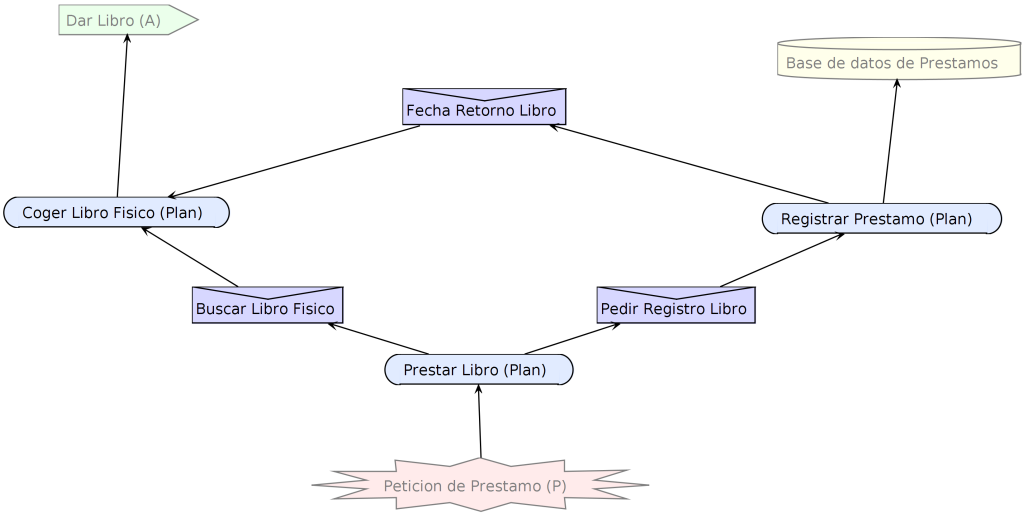


- Las capacidades se descomponen a su vez en **planes**, que son los que llevan a cabo las tareas
- Estos estarán:
  - Conectados a las **percepciones**/eventos
  - Recibirán los **mensajes** generados por los protocolos de interacción entre los agentes
  - Generarán **mensajes** internos para comunicarse con otros planes dentro de la capacidad
  - Realizarán las acciones

Diagrama del agente **Prestador**



Planes de la capacidad **Prestar Libro**



Capacidad	
Nombre	Prestar Libro (C)
Descripción	Obtiene un libro físico, determina la fecha de retorno...
Objetivos	Prestamo...
Protocolos	Ninguno
Mensajes Entrantes	Ninguno
Mensajes Salientes	Ninguno
Mensajes Internos	Pedir registro libro, fecha retorno libro...
Percepciones	Petición de libro (P)
Acciones	Dar Libro (A)
Datos usados	Base de datos usuario
Datos producidos	Fecha de retorno libro, registro de préstamo
Datos Internos	Por definir
Planes incluidos	Prestar Libro, Registrar préstamo...
Capacidades incluidas	Ninguna

Plan	
Nombre	Prestar Libro (P)
Descripción	Desarrolla las acciones necesarias para prestar un libro
Iniciador	Percepción de petición de préstamo
Mensajes Entrantes	Ninguno
Mensajes Salientes	Pedir registro libro, Coger libro físico
Datos usados	Base de datos usuario
Datos producidos	Ninguno
Fallo	Falta de acceso a los datos
Recuperación de fallo	Enviar mensaje a mantenimiento
Procedimiento	Código del plan

Datos	
Nombre	Base de datos de préstamos
Descripción	Contiene los registros de los préstamos de libros a usuarios
Tipo de datos	Registro préstamo de libro a usuario
Campos	Id usuario, Id Libro, Fecha retorno
Persistente	Sí
Externa al sistema	No
Incialización	Vacía
Producida por	Agente prestador, plan registrar préstamo
Usada por	Agente Prestador, Agente Reclamador...
Usada cuando	El usuario pide un libro...