CONTRACT BASED COOPERATION FOR AMBIENT INTELLIGENCE: PROPOSING, ENTERING AND EXECUTING CONTRACTS AUTONOMOUSLY

Fatma Başak Aydemir

Boğaziçi University
Department of Computer Engineering

November 4, 2011

Ambient Intelligence

- Indicates environments that are aware of and responsive to human presence.
- Should be:
  - Context aware.
  - Personalized.
  - Adaptive.
  - Anticipatory.
- Emerging technologies: sensors, radio frequency identification, affective computing, brain computer interfaces, nanotechnology and software agents.

Ambient Intelligence

- Indicates environments that are aware of and responsive to human presence. Consider an intelligent kitchen.
- Should be:
  - Context aware. How many people are going to be served?
  - Personalized. The user is allergic to strawberries.
  - Adaptive. The user goes on a diet.
  - Anticipatory. The user may go on a diet soon.
- Emerging technologies: sensors, radio frequency identification, affective computing, brain computer interfaces, nanotechnology and software agents.

Problem Definition and Proposed Solution

Problem:

- Constructing an ambient intelligence environment that is dynamic and flexible

Proposed Solution:

- An open multiagent system that
  - allows flexible cooperation among agents,
  - is regulated by dynamically created contracts.

Multiagent System

- Agents are autonomous systems that perceive the environment via their sensors and act upon the environment via their actuators.
- Social – consists of several intelligent agents that communicate.
- Autonomy – each agent has some degree of autonomy with respect to its actions.
- Cooperation vs collaboration – overall capability of the MAS may exceed the capabilities of the individual agents.
- Open – agents enter and leave the system freely.

Intelligent Kitchen

- There are users that benefit from the kitchen so there are agents that represent these users.
- There are agents that provides services such as preparing cake, brewing coffee and so on.
- The kitchen provides menus to the user of her choice.
- Menus consist of multiple services and all of the services should be served together.
- Most of the time, menus can be served by multiple agents' working together.

Contracts

- Commitments form a contractual relationship between the agents.
- Social commitments are contracts made from one agent to another in order to satisfy certain properties.
- CC(X,Y,Q,P)
  - X: Debtor.
  - Y: Creditor.
  - Q: Condition.
  - P: Proposition.

Contracts

- Commitments form a contractual relationship between the agents.
- Social commitments are contracts made from one agent to another in order to satisfy certain properties.
- CC(X,Y,Q,P)
  - X: Debtor.
  - Y: Creditor.
  - Q: Condition.
  - P: Proposition.
- CC(Fridge, User Agent, Cake Request, Cake)
- CC(Coffee Maker, User Agent, Coffee Beans, Coffee)

Statically Generated Contracts

- Closed system.
- Unlimited resources.
- Fixed desires and needs of the user.
- Unchanging services of the agents.

Statically Generated Contracts

- Closed system. None of the agents gets broken, no agent enters the system later.
- Unlimited resources. The coffee maker agent never runs out of coffee beans.
- Fixed desires and needs of the user. The user always ask for coffee and cake.
- Unchanging services of the agents. The coffee maker agent continues to make coffee.

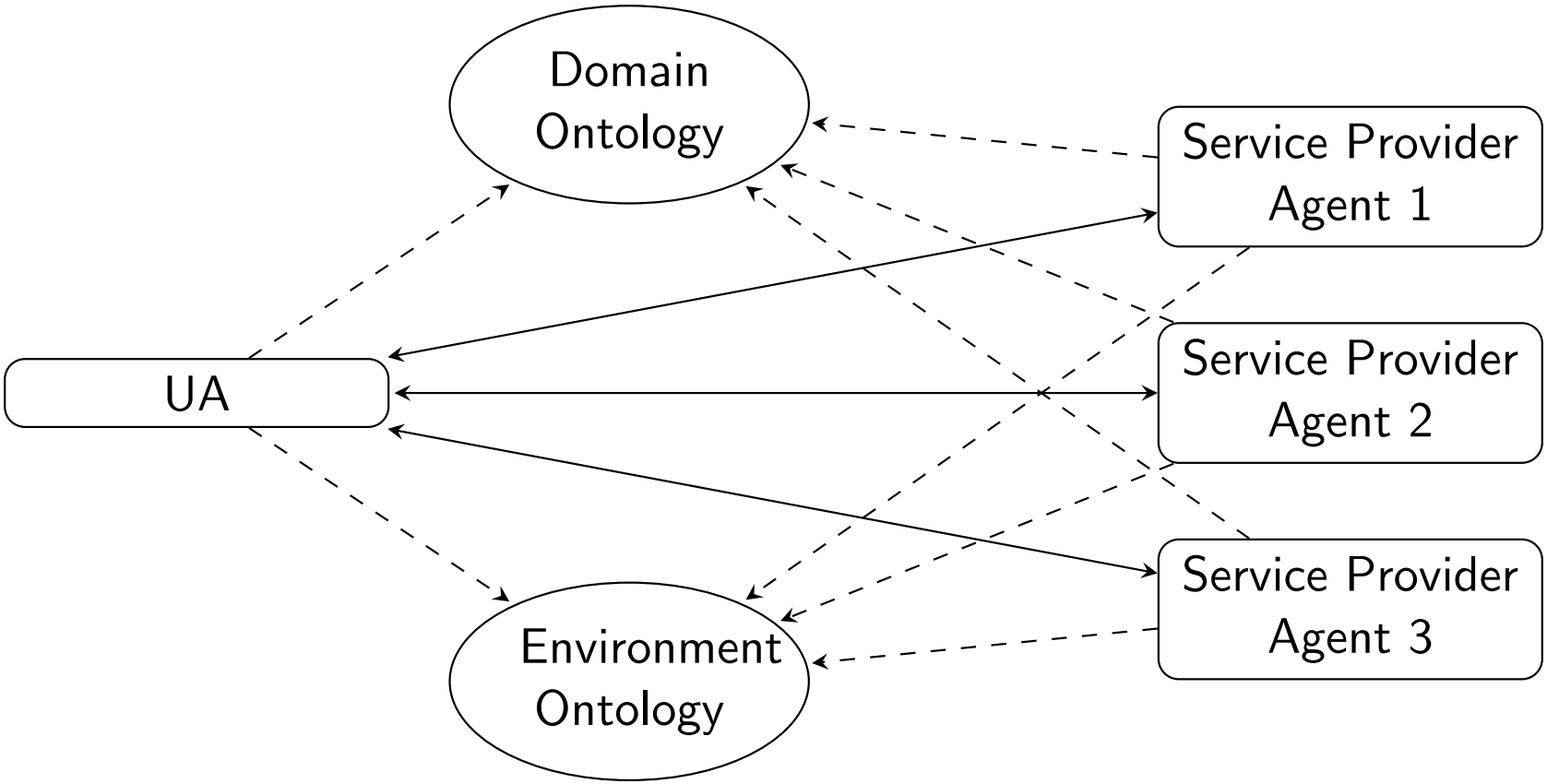
Dynamically Generated Contracts

- Open system.
- Scarce resources.
- Alterable desires and needs of the user.
- Changing services.

Dynamically Generated Contracts

- Open system. Agents may get broken, broken ones may get fixed or replaced.
- Scarce resources. The coffee maker agent runs out of coffee beans as it consumes them.
- Alterable desires and needs of the user. The user may switch between different menu combinations, cake and coffee, simit and tea, tea and cake, and so on.
- Changing services. The coffee maker agent starts serving cappuccino.

System Architecture



User Agent

- Represents the user.
- Takes the needs and desires of the user.
- Proactively communicates with other agents.
- Tries to establish the cooperation among agents and to serve the user.

Service Provider Agents

- Provide services.
- Reactive before communication starts.
- Proactively make requests during communication.
- Autonomously decide to take part in the contracts.

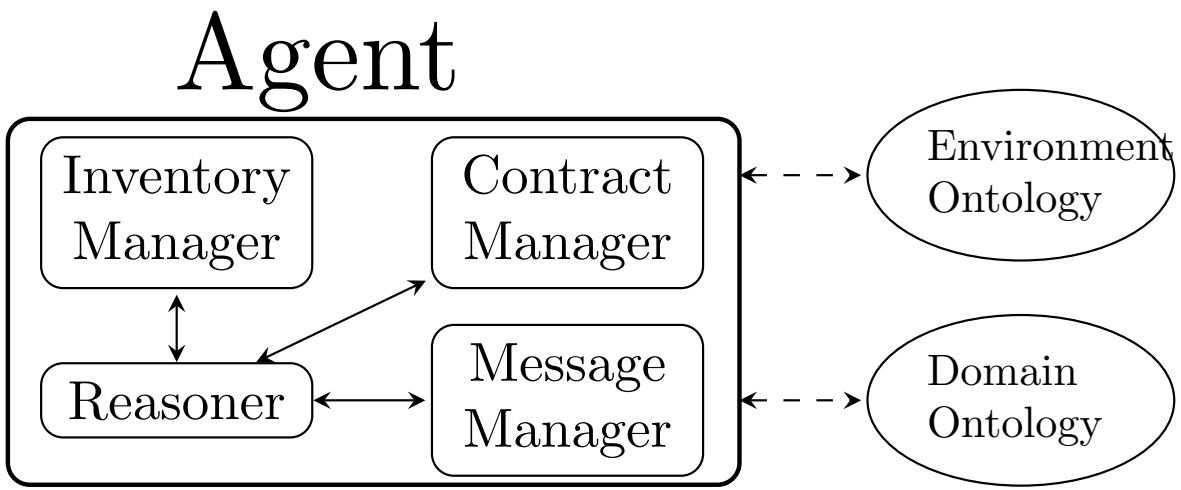
Environment Ontology

- Models the system.
- Describes the agent, contract, user and the physical environment
- Does not reveal the internal states of the agents.
- Keeps the contextual information about the user and the environment.

Domain Ontology

- Provides information about the services.
- Classifies the services.
- Describes the resources needed to serve the services.
- Used by agents in the process of making the decisions on entering the contracts by
  - providing the resource – service relation,
  - providing the similarities between resources.

Agent Architecture



Inventory Manager

- Manages the inventory of the agent.
- Does not reveal the information to the public.
- Main operations:
  - Updating resources
  - Querying resources

Message Manager

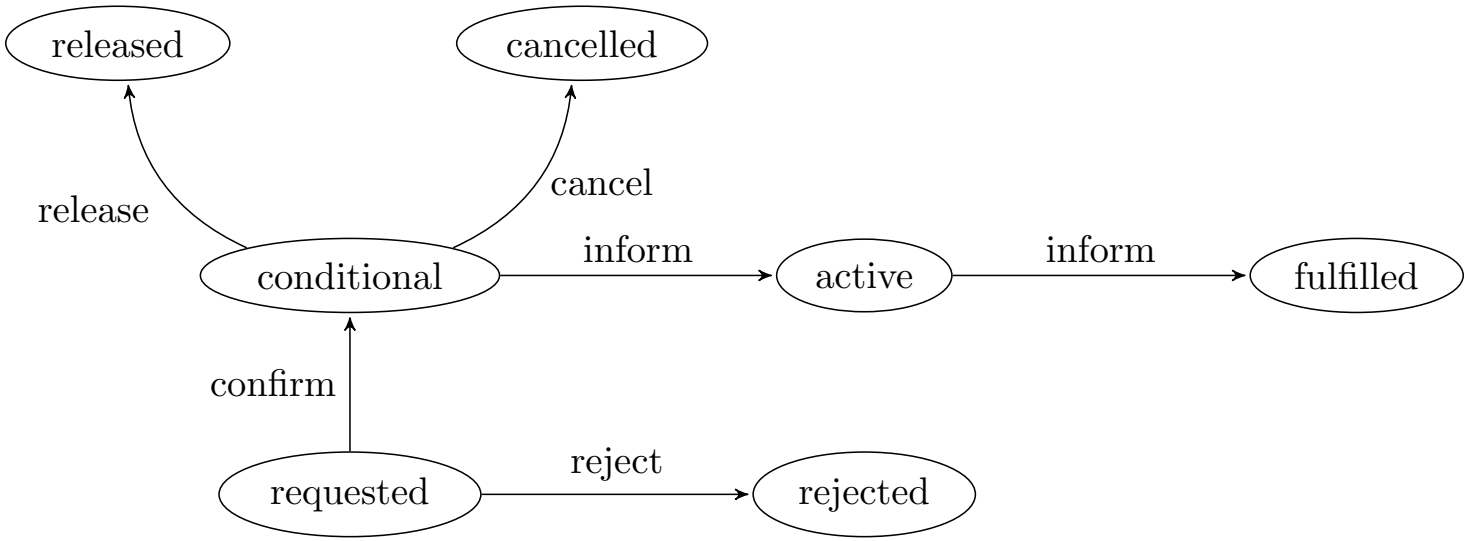
- Sends messages to other agents.
- Receives messages from other agents.
- Sorts messages according to their type and pass them to the reasoner.
- Traces conversations.

Contract Manager

- Handles the contract operations of the agent.
- There is no common contract base.
- Main operations:
  - Updating contract states
  - Querying contracts according to their state, debtor, creditor, proposition and condition etc.

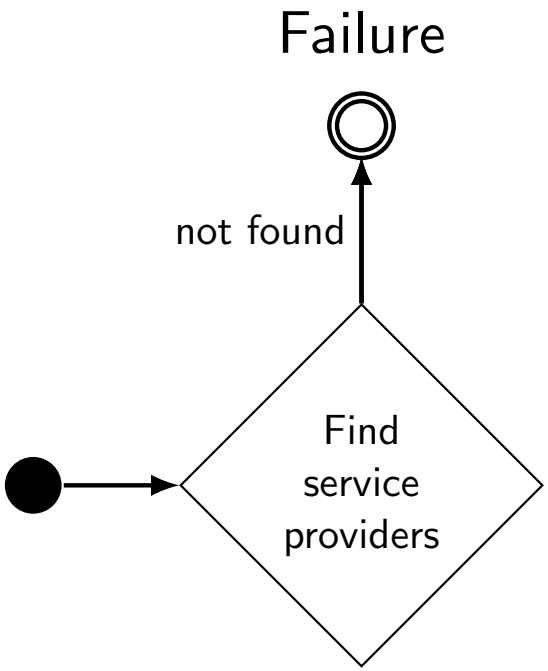
Contract Life Cycle

CC(X,Y,Q,P)



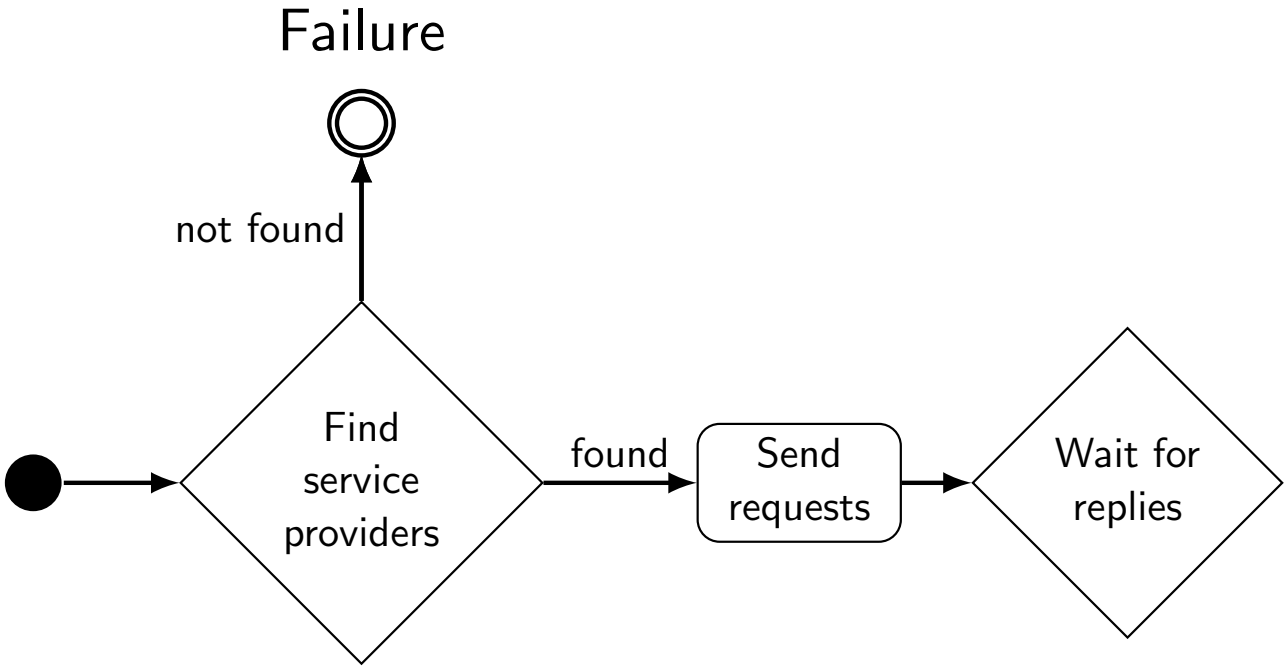
Reasoner

- Makes the decision of entering into contracts interacting with the inventory manager and the contracts manager as well as reasoning on the domain ontology.
- Processes the message contents, updates the contract base and the inventory together with the contract manager and the inventory manager.
- Responds to messages and takes necessary actions such as fulfilling the proposition when the condition is fulfilled and so on.


Failure
not found
Find
service
providers

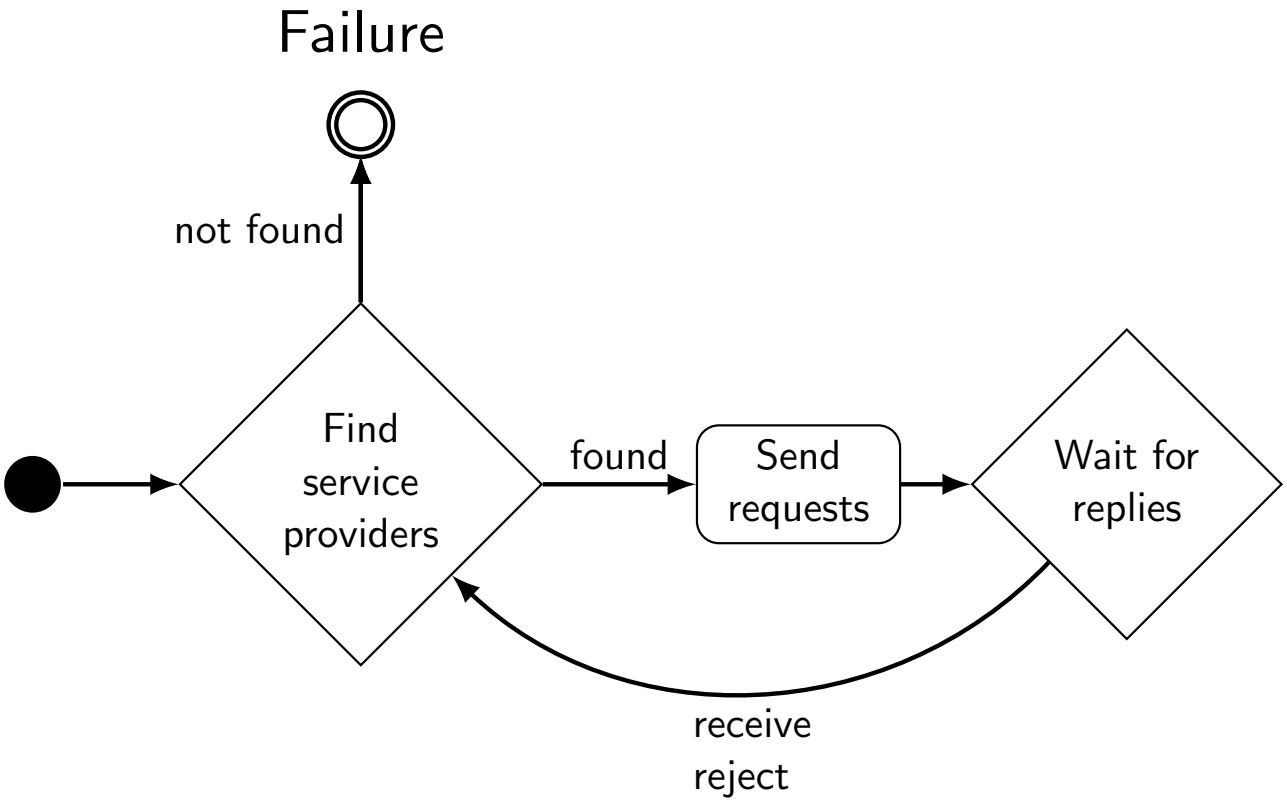
Service Provider Agent

Can I serve now?
Do I have the service ready?
What are the resources?
Are there any missing resources?



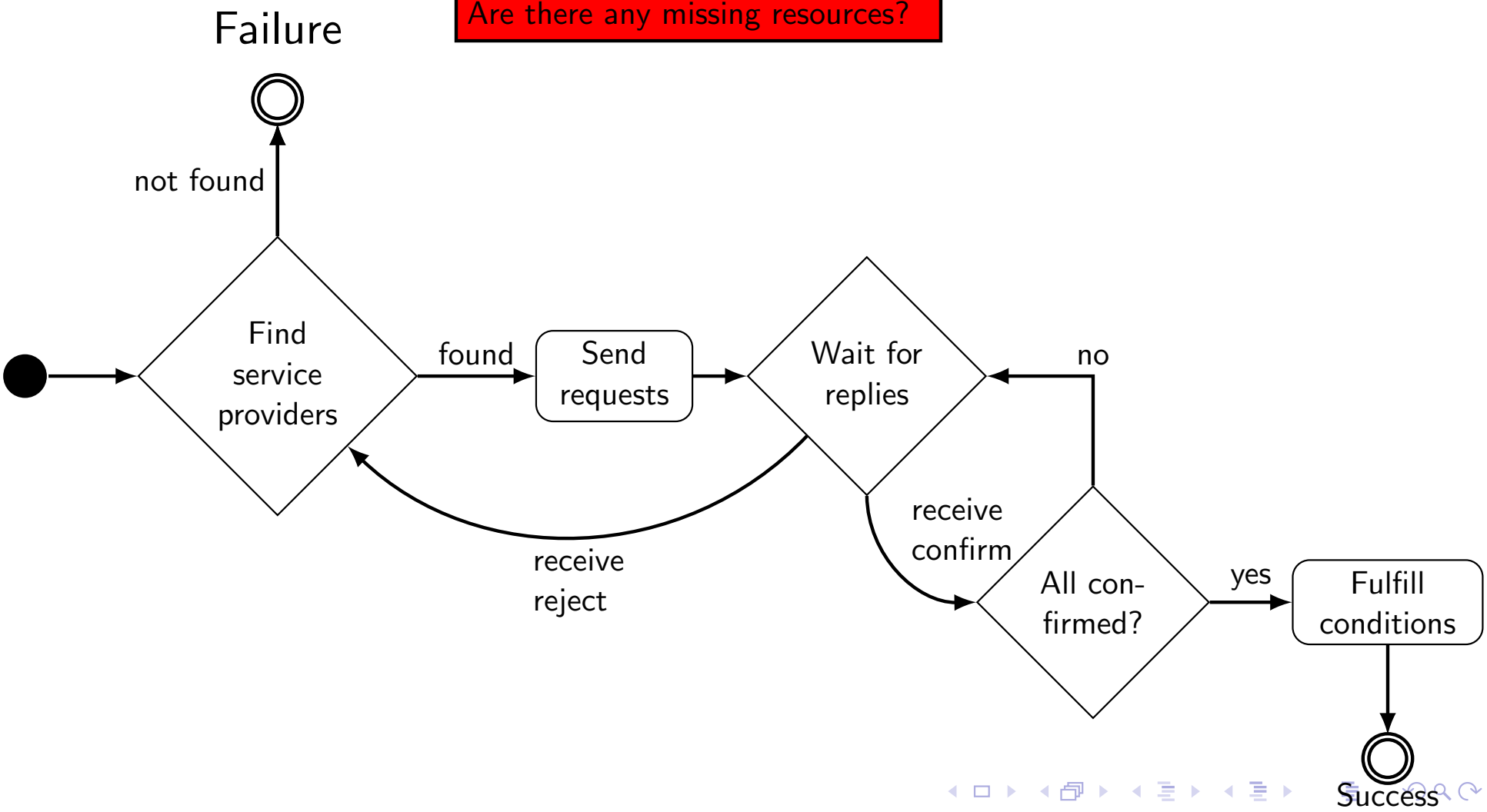
Service Provider Agent

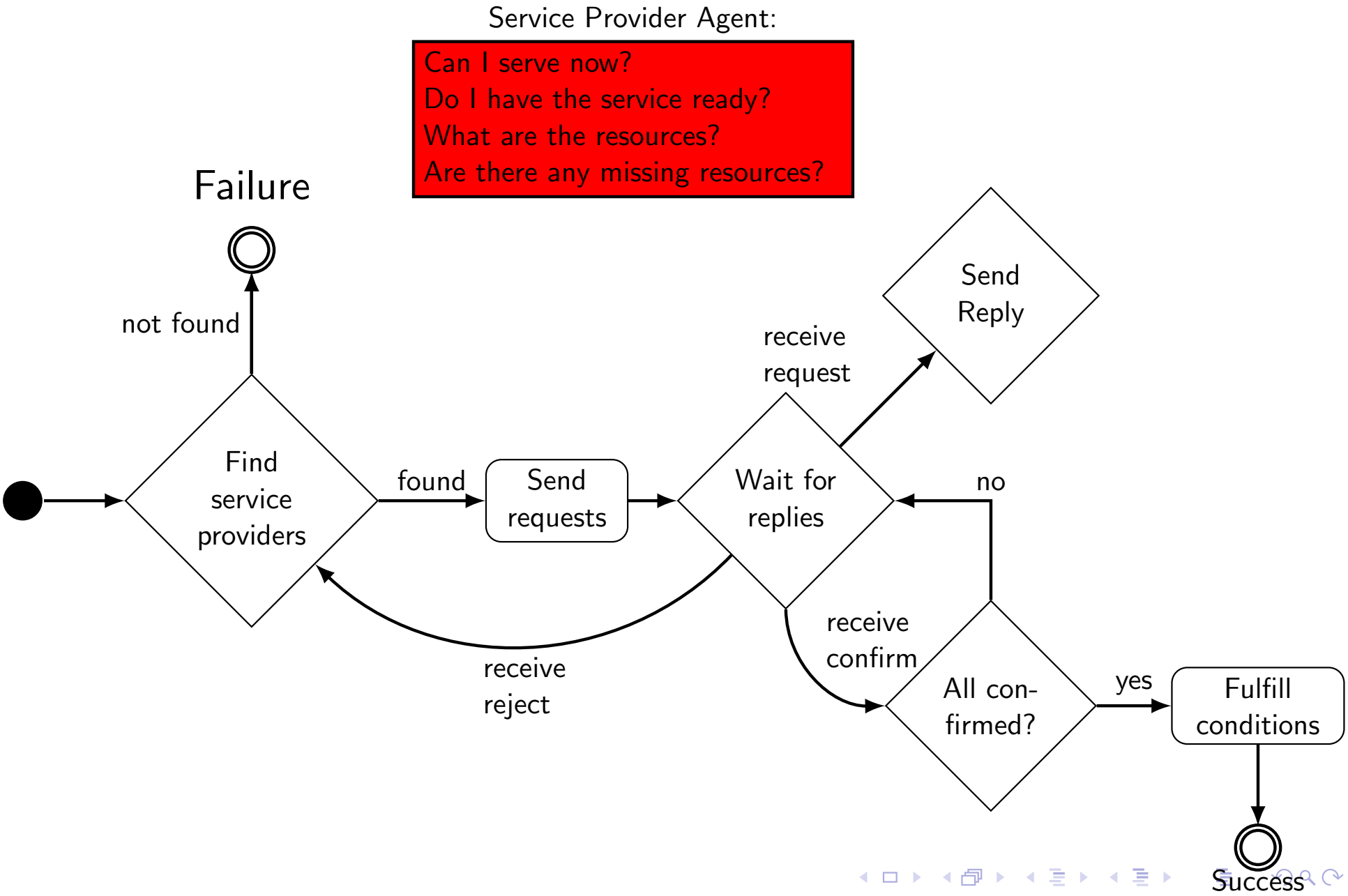
Can I serve now?
Do I have the service ready?
What are the resources?
Are there any missing resources?

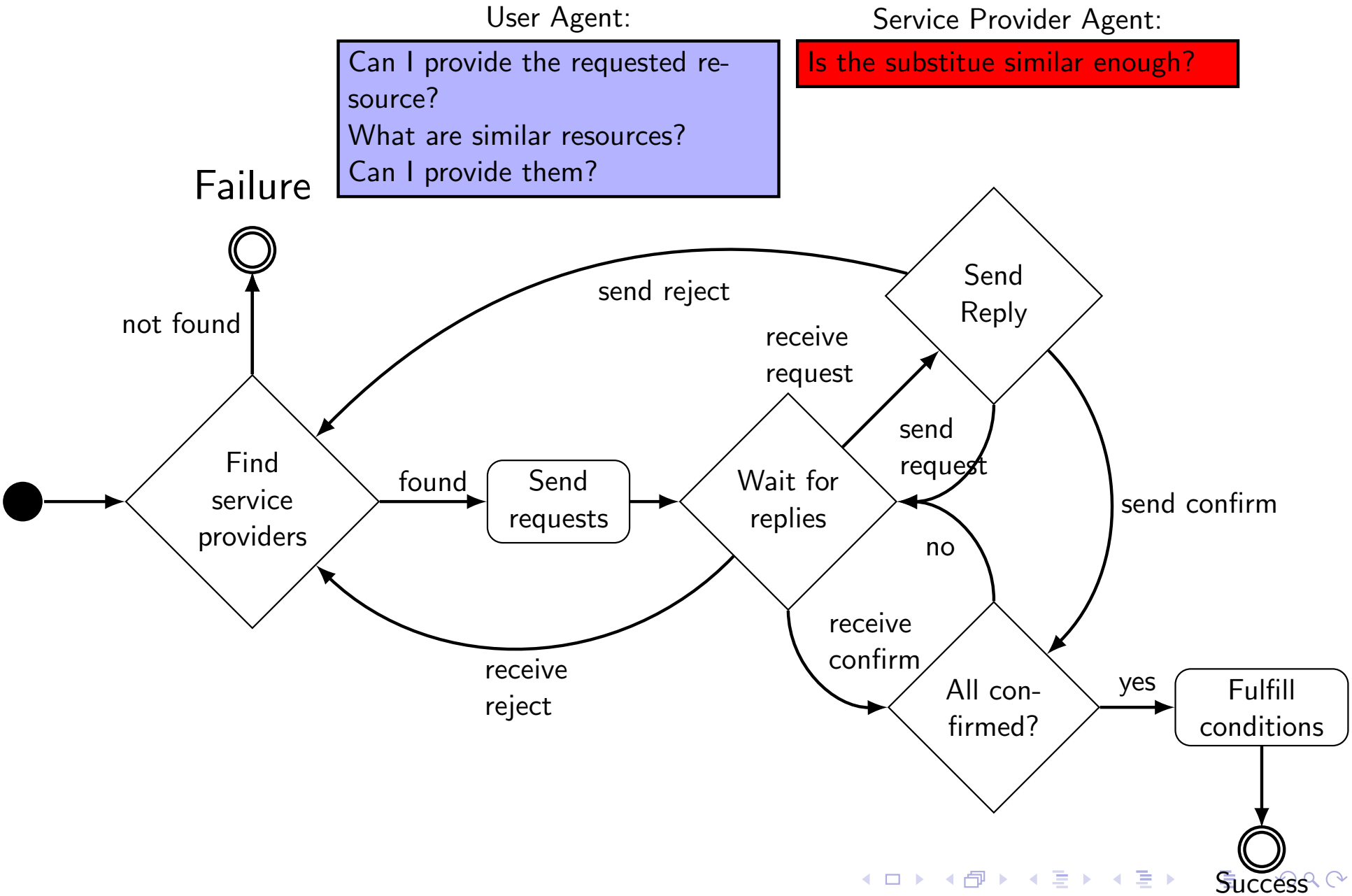


Service Provider Agent:

Can I serve now?
Do I have the service ready?
What are the resources?
Are there any missing resources?




Service Provider Agent:
Can I serve now?
Do I have the service ready?
What are the resources?
Are there any missing resources?
Failure
not found
Find service providers
found
Send requests
Wait for replies
receive request
Send Reply
no
receive confirm
All con-firmed?
yes
Fulfill conditions
receive reject
Success


User Agent:
Can I provide the requested resource?
What are similar resources?
Can I provide them?
Service Provider Agent:
Is the substitue similar enough?
Failure
not found
Find service providers
found
Send requests
Wait for replies
send reject
receive request
Send Reply
send request
send confirm
no
receive reject
receive confirm
All confirmed?
yes
Fulfill conditions
Success

JADE

- Open–source.
- Java is used as the programming language.
- Directory Facilitator provides yellow pages services.
- Agent Management System handles agent registration to the system.
- Provides FIPA compliant messages.
- Message structure:
  - Performative: The type of the message.
  - Sender: The agent that sends the message
  - Receiver: The agent that receives the message
  - Conversation ID: The identification that used to mark messages in the same conversation.

Application Domain

- An AmI kitchen domain.
- There is one user, represented by User Agent (UA).
- Service Provider Agents:
  - Coffee Maker Agent (CMA) brews coffee.
  - Fridge Agent (FA) serves cake.
  - Mixer Agent (MA) serves cake.
- The user asks for menus such as coffee and cake menu and all the items of the menus are served together.

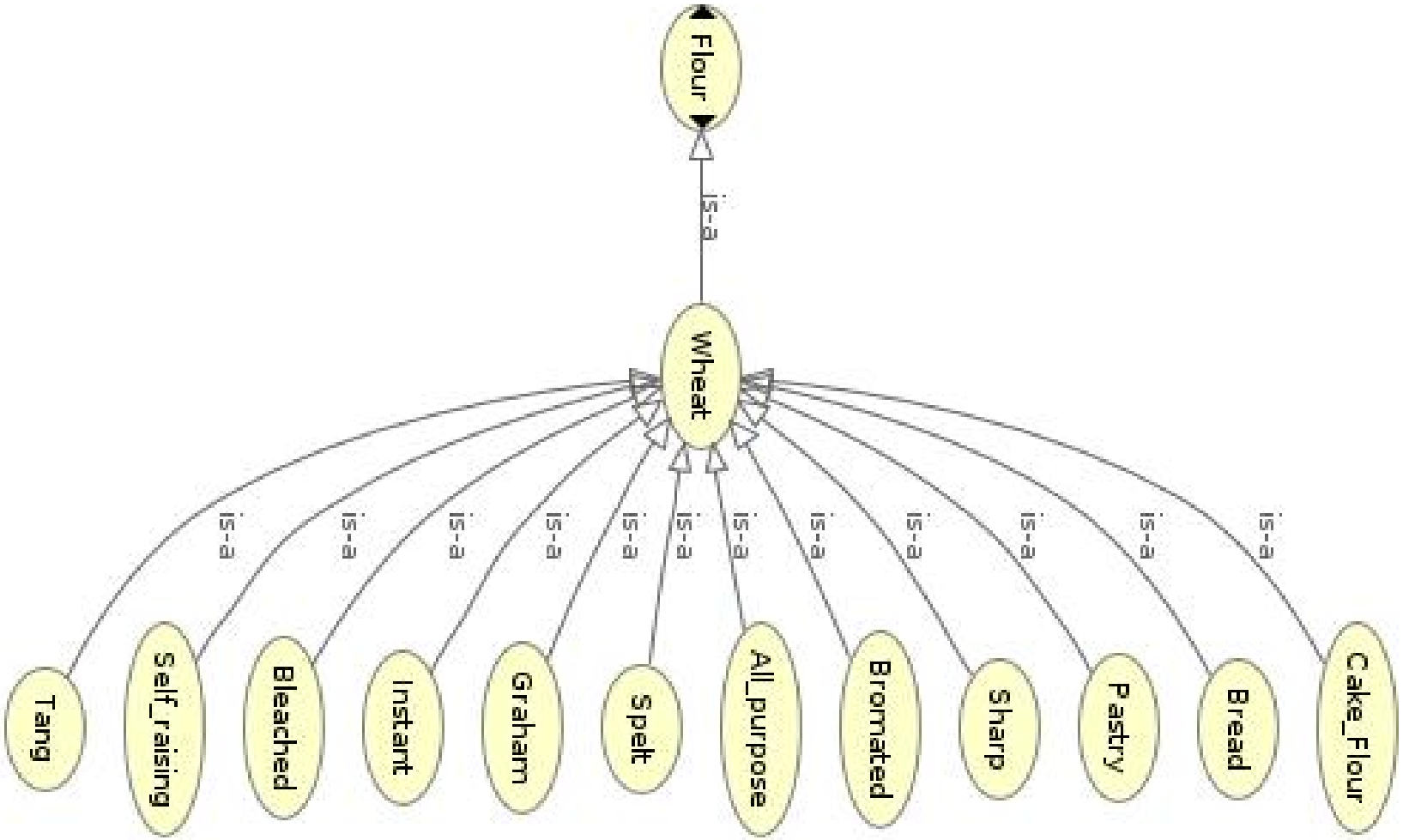
Environment Ontology

- Models the kitchen environment.
- The kitchen class has the following properties: hasHumidity, hasTemperature, hasAgent, hasUser.
- The agent class has the following properties: hasName, has InventoryManager, hasContractManager, hasMessageManager, hasReasoner and serves.
- The contract class has the following properties: hasDebtor, hasCreditor, hasProposition, hasCondition and hasState.

Domain Ontology

- Describes the services and the resources.
- hasResource property links the services to the resources.
- hasSimilarity property links the similar resources.
- Agents reason on this ontology to:
  - Get the necessary resources to serve a service
  - Calculate the similarities between to services or resources.

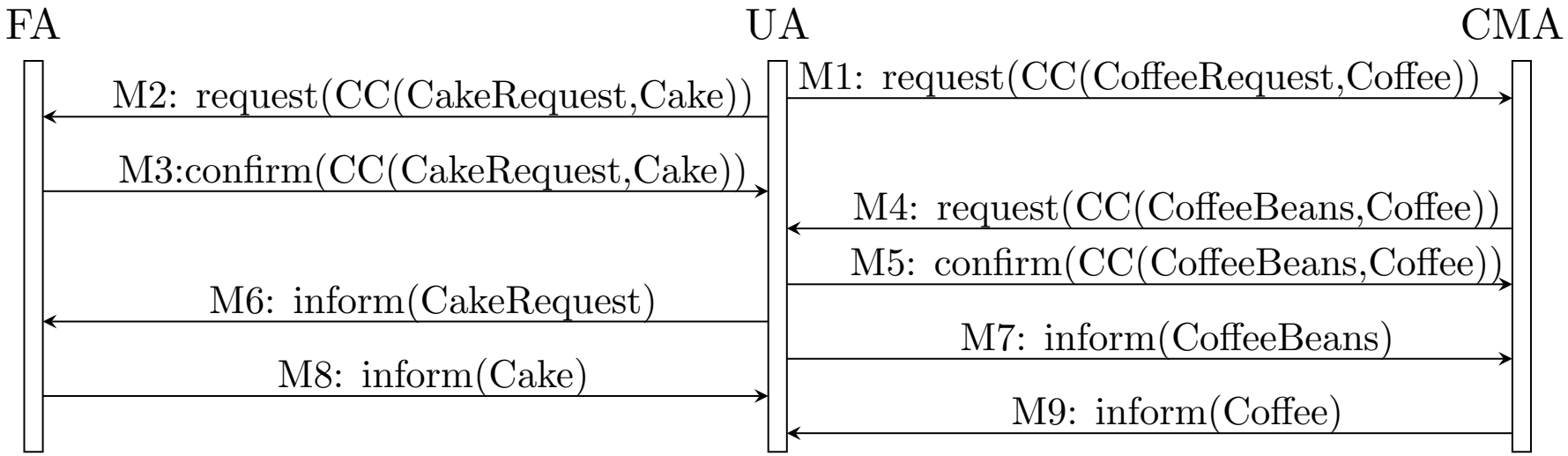
Domain Ontology



Scenario 1: UA Provides a Missing Resource.

- CMA serves coffee, coffee beans and water are needed to serve coffee. CMA lacks coffee beans.
- FA serves cake and has a cake ready to be served.
- UA can provide coffee beans.

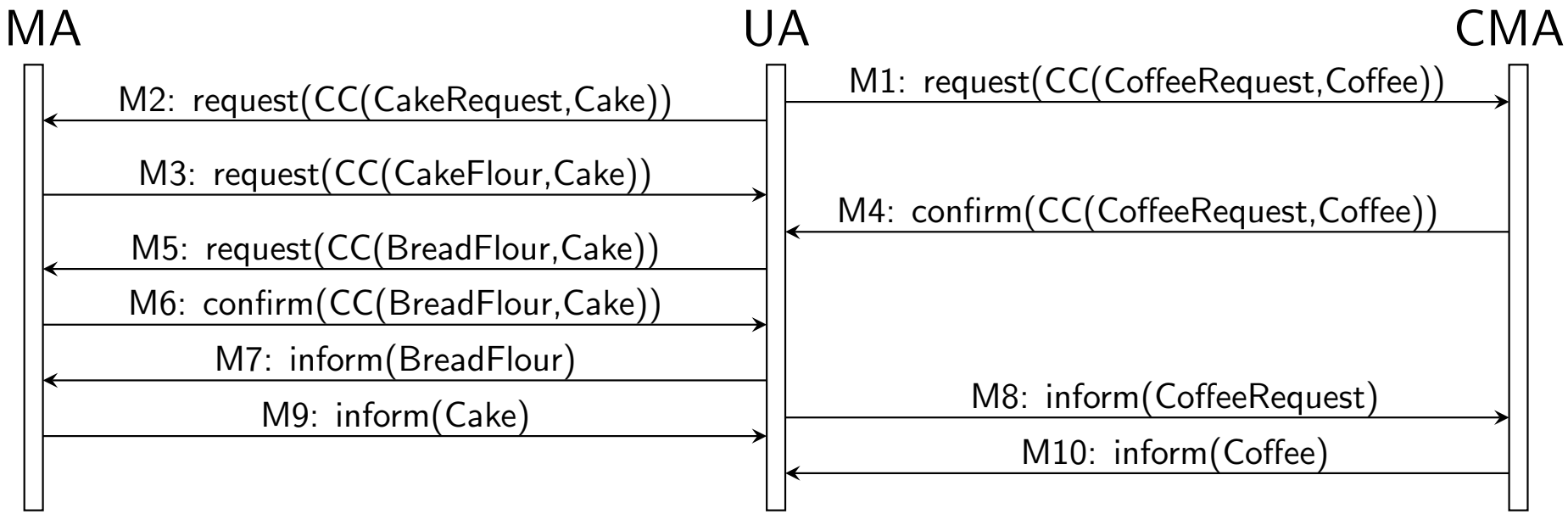
Scenario 1: UA Provides a Missing Resource.



Scenario 2: A Substitute is Proposed instead of the Original Resource

- CMA serves coffee, coffee beans and water are needed to serve coffee. CMA has all ingredients to serve coffee.
- MA serves cake, cake flour is one of the resources to serve cake. MA lacks cake flour to serve cake.
- UA can provide bread flour.

Scenario 2: A Substitute is Proposed instead of the Original Resource



Contributions

- We do not force rigid collaboration structures.
- We form cooperative structures by generating contracts dynamically in run time.
- The decision on entering a contract is made by reasoning on the knowledge that is relevant and accessible by the agent.
  - State of the system.
  - State of the agent.
  - Relevant domain information.