The main purpose of this work was to propose a new methodology that would relieve the large effort required for the development and set-up of agent based solutions by extending the scope of the applications.

Our representation and usage of agents are part of a repository prototype for managing reusable agent-based artifacts. Agents are but one kind of reusable components that developers can search for, browse, and integrate into their own applications.

Below it is illustrated a step by step how the repository system can be used.

Figure 1 shows a screenshot of the home page. Users can perform different types of agent artifact searches, e.g. introducing the keywords in the textbox, using the tag cloud, the advanced search (More Searches link) or by applications domains (by default Health care is selected). In addition, there is a link to the administration area and a RSS feed, both are explained later.

There is only an agent component classified in the category Health care. Its tags can be seen in the list of tags at the right. The visualization of each tag depends on its weight in the system, that influences in the style of the font.
Figure 2 illustrates another example of the search by categories. In this case, all agents under the domain e-commerce are listed. The union of their respective tags is showed in the right panel. Under a specific application domain if we select a tag, we are doing a filtered search pointing out which agents are assigned by these tags. Figure 3 depicts how this process names custom search facets, works.
The advanced search is based on some characteristics of the agent components, like programming language they were developed with, platform, dimensions and contexts (cooperation, coordination, negotiation). Figure 4 illustrates an example of platform-based search by Jadex, where the resulting agents implemented with this platform are listed next.

Figure 4. Platform-based search.
If we click a link that refers to a specific agent, we can see its characteristics. An example is depicted in Figure 5, which shows the description of a Virtual Market.

![Agent Component Details](image)

**Agent Component Details:**

Name: Virtual Marketplace
Version: 1.0
Development Date: 2012-09-11
Language: Java
Platform: ASP
Description: Virtual Marketplace Instance. This project included agents that play the roles of seller and buyer.

Previos:
Categories: e-commerce
Tagged: used/second hand items, buying, speciality goods, e-commerce, selling
Dimensions: Interactive Agent, Autonomous Agent

Related: -

Roles:
- Description: To buy and immediately sell speciality goods. An speciality item is an extraordinary or unique product enough to motivate people to make an unusual effort to get it. Examples are designer clothes, exotic perfumes, limited-edition cars, stunning designs, works of famous painters.
  
  Precondition: the product is a speciality item
  
  Operations:
  - Description: To check if there is a buyer of speciality items.
    
    If failure: if there is no buyer, it will try again to look for in a while
    
    if failure: to notify to the user there is no available the speciality item that the users wants to buy
    
    if failure: to inform the user how much his desired speciality item costs. If the users accepts the proposal, it will buy the product in the shop and later re-send it to the user in a higher value.
  
  Precondition: the product is not new
  
  Description: buying imported books
  
  Precondition: the book are imported.
  
  Description: selling imported books
  
  Precondition: the book are imported.
  
  Description: selling second hand books
  
  Precondition: the book are used by someone else before.
  
  Description: buying second hand books
  
  Precondition: the book are used by someone else before.

Developed By: Andrew Costa
Diagram: -

Sender Message Interface: -
Receiver Message Interface: -

Download File

![Figure 5. Agent Component Details.](image)

To manage agent components in the repository, it is necessary to be logged in the system. Users can registry in the system for free as Figure 6 shows.

![Figure 6. Login and User registration screens.](image)
Figure 7 illustrates how a user can register an agent component.

After the agent is registered in the repository, the next step is to add its interfaces and to know with which agents it interacts and how. Figure 8 shows how add the interfaces to a specific agent. The agents listed as participants are in the same application domains of the new agent component.

Figure 8. Adding Interfaces to the Agent Component.
The user can establish relationships among agent components, how it is illustrated in Figure 9.

Figure 9. Adding Relationships to the Agent Component.

A user can subscribe to an application domain to be updated about how it is happening in the current category, as Figure 10 shows. There is a description of each application domain and a description of each agent belonging to its category for users to be guided by. The user can choose whatever feeder, now built them into email clients and browsers.

Figure 10. Subscription to Categories of Agents.