An agent and web services based e-government model validation: electronic identity card & health card creation case studies

EMNA KAROUI*, SAMEH HADOUAJ**, KHALED GHÉDIRA*

*Higher Institute of Management of Tunis, Tunis University, TUNISIA
**Faculty of Economic and Management Sciences of Nabeul, Carthage University, TUNISIA
Higher colleges of technology, Abu Dhabi, UAE

Abstract: - In E-Government, the provision of valuable services depends upon the capacity of Public Administrations to coordinate their services. The ability of computing systems owned and managed by these Public Administrations turns independently without adequate attention to the need to interact with other Public Administration systems, coordination of heterogeneous computing solutions is a difficult and costly task. Moreover, Some Public Administrations are not necessarily equipped with sophisticated systems especially in developing countries such as the case of Tunisia. To this direction, although several approaches have been proposed, all these approaches would be insufficient since they are theoretical solutions, centralized and costly. This paper presents an e-government model based on agent and web services. It introduces an Inter-Organizational Workflow (IOW) model based on the idea of interfacing. The e-Government Entities (GEs) are only required to describe their services using WSDL. The e-Government Entities (GEs) are distributed, a direct interaction is established between them with no need of a middleware. Two different case studies are carried out, to demonstrate the applicability of our approach.
Keywords: - Web servicesm e-government, electronic identity card, information and communication technologies.

1 Introduction

E-Government is defined as *"the use of technology to enhance access to and delivery of government services to benefit citizens, business partners and employees"* [1]**.** E-Government uses Information and Communication Technologies (ICT) for improving the interaction between citizens and government agencies. Recently, Different process-based systems or prototypes for Digital Government have been developed. Unfortunately, most of these works assume that all the involved partners are equipped with sophisticated Information Technology (IT) infrastructure, like for example workflow engines. These constraints do not permit emerging countries or under-equipped administrations to easily adapt this kind of systems. Besides, these systems are based on an ad-hoc or implicit coordination model and therefore do not provide means to engineer it (study, analyze, simulate, adapt, etc.). Indeed, the reconstruction of the entire system is very costly and this remains a big challenge, especially, in developing countries where funds are not sufficient to support E-Government projects.

Another challenge that is presented in the E-Government field is distribution. E-Government system involves an important number of processes, distributed in several administrations, and it has to coordinate these processes in order to carry out citizens and Public Administrations (PAs) requests [2]. Therefore, a solution that complies with e-government constraints should reduce the transaction time and the communication cost by adopting a distributed approach.

The communication between different governmental administrations raises another problem: interoperability. Indeed, the need of more complex services delivered by a collaboration of several public administrations led governments and institutions to encourage research and development to present solutions for the interoperability problem, dynamic environment and automatic discovery, composition, and execution of services.

Service Oriented Architecture (SOA) is an approach to build distributed systems that deliver application functionality as services which are language and platform independent [3]. A Web Service (WS) is a technology that realizes the SOA [4]. Web service is a key technology that can support PA technical interoperability. To deal with technical heterogeneity, Web Service provides a set of standards Simple Object Access Protocol (SOAP), Web Service Description Language (WSDL) and Universal Description Discovery and Integration (UDDI). However, Web Service technologies provide an insufficient degree of autonomy and ability needed to adapt automatically to changing situations. Intelligent Agents (IA) [5] are a suitable technology to deal with these drawbacks.

Recent E-government projects have exploited the combination of Agent and Web Service to enhance PA coordination. However, they are unable to face distribution or construction cost problems.

Recently, we presented a distributed e-government model [2] that is supported by Software Agents and Web Services. The proposed Inter-Organizational Workflow (IOW) model is based on the idea of interfacing and the distributed organization of e-Government Entities (GEs). We impose very few technology constraints on partners. They are only required to describe their services using WSDL. So, administrations are not supposed to be equipped by any complex IT infrastructure, which fits well the emerging countries administrations. Each administration will be equipped with an interface used for its services publication or services request. Citizens can use this interface via the E-Government portal to request services. Their executions can overtake more than one PA. So a distributed coordination of GEs interfaces is adopted. This allows a direct interaction between them with no need of a middleware. The need of a distributed model based on interface collaboration is justified in [2, 6]). In this paper, we present a final and complete version of a prototype that implements that model and a detailed description of the Implementation steps. Our work is a part of a government project in Tunisia named Secure System for E-Government (S2EG). The aim of the project is to secure inter-organizational interactions in E-Government through the developing of secure applications based on reuse and coordination of administrations services using an IOW coordination model.

Two different case studies are carried out to demonstrate the applicability of our approach in the E-Government domain and verify the expected benefits. The first used case study concerns the process of creating an Electronic Identity Card (EIC) in Tunisia. The second one concerns the process of creating a health card of national health insurance office (CNAM) in Tunisia.

The paper is organized as follows: in section Two, e-government challenges and constraints are presented. Section three introduces the related works. The proposed model is described in section four. In section five, scenarios of simple and composite services are illustrated. Section six is devoted to the experimental results: to illustrate the applicability of the proposed architecture, two case studies and a prototype implementation are presented. The paper ends with a conclusion.

2 E-Government challenges and constraints

E-government is facing a number of exceptional challenges because of its strict and committed nature of relationships of Government to Citizen (G2C), Government to Government (G2G), Government to Business (G2B) and Government to Employee (G2E) [7]. In spite of the considerable literature on E-Government, we still do not have a real, distributed and low-cost framework in place that assesses the potential of E-Government by taking into consideration all the critical dimensions. In this section, we introduce E-Government constraints.

2.1 Interoperability

Interoperability is a key issue in the development of current E-Government services [8]. It is concerned with linking computer networks as well as with sharing and reusing knowledge between networks. Two main levels of interoperability are presented in the literature: technical and semantic interoperability. The first one refers to the topics of connecting systems, defining standard protocols and data formats. The second one concerns the exchange of information in an understandable way, whether within or between administrations [8]. Our paper is concerned with technical interoperability.

2.2 Flexibility

This requirement concerns partners' selection, tasks allocation, and exceptions management [9]. In the context of E-Government, the services offered by each GE may vary. In fact, services can be added; deleted or updated. Coordination of these GEs must take into account changes of environment.

2.3 Autonomy

Autonomy refers to the degree of compliance of a partner to the global control rules. Partner systems may be autonomous in their design, communication, and execution. This means that

individual partners select the process and content description models, programming models, interaction models with the outside world, etc. In a fully autonomous collaboration, each partner is viewed as a black box that is able to exchange information (i.e., send and receive messages) [11]. Based on this definition, we conclude that each organization participating in the IOW has to:

• Determine the conditions of cooperation.

• Preserve the control of its local tasks and maintain their confidentiality.

2.4 Distribution

It concerns process, information used and resources. In E-Government, there are different architectures based on one of the three types of coordination: distributed, centralized or hybrid

• Centralized: one agent coordinates all the participant organizations. In E-Government, this agent is always included in a middleware.

• Distributed: Organizations are coordinated without interacting with a third partner.

• Hybrid: It combines centralized coordination and distributed coordination. Organizations are subdivided into groups and each group has a coordinator agent. Organizations communicate through coordinator agents.

A conducted study of these three coordination models [12] shows that distributed coordination is the most appropriate one for E-Government. It allows reducing transaction time and communication cost.

2.5 Project Cost

E-government projects aim to facilitate the provision of electronic services to citizens, business and government entities. However, realizing this vision is strictly depending on the ability to minimize the cost of e-government implementation.

The cost of the project depends on the approach adopted. Some of these approaches impose the rebuilding of the IT infrastructure. They entail a replacement of hardware and software as well as training employees on the new systems. Costs can be far above the ground [13]. The reconstruction of the entire system is very costly and this remains a big challenge, especially, in developing countries where funds are not sufficient to support E-Government projects. Consequently, it is very important to minimize project cost. It is crucial to point out the importance of the choice of e-government approach to deal with such a requirement.

2.6 Type of requested services

A web service can be simple or composite. Simple services are Internet-based applications that do not rely on other Web services to full consumers' requests. A composite service is defined as a conglomeration of outsourced Web services (called participant services) working in tandem to offer a value-added service [11].

When requesting a simple service, the discovery approach is used to find services that fulfill client requirements. However, if the request is a composite service, the discovery approach can't find services that match the requested service. So a composition approach is needed.

Nowadays, a huge amount of e-government services is deployed on the Web and there are a lot of requests that cannot be fulfilled using just one web service. In most cases, composing individual services to create composite web service to fulfill the users' request is necessary.

3 Related Work

During the last few years, several projects under the E-government initiative have emerged. One of the main challenges of these e-government initiatives relies on efficiently integrating all heterogeneous public information systems and business processes of government organizations [14]. So finding a solution for technical interoperability is the main ambition of governments. Consequently, several research works adopted web service technology.

3.1 Web service-based approaches

For Example, in [14], a framework for an E-government Based on Service Oriented Architecture for Jordan is presented. The authors propose a novel integration mechanism based on the web service. The proposed architecture is examined using a case study in the context of implementing environmental license web service in the Jordanian ministry of environment. The proposed architecture is divided into four layers: client layer, presentation layer, application layer, and data layer. The proposed architecture is supported by a secure government network.

Even though the presented architecture of [14] was conceived to tackle interoperability challenge in E-Government as well as to handle simple and composite services, it is a centralized architecture based on a middleware.

In [15], the author presented an architecture based also on WS. The author developed a Service Oriented E-Government Architecture in order to

deal with Interoperability. Compared to existing works, the architecture of [15] was proposed to find a more practical solution to solve the architectural problem and to successfully implement E-Government interoperability.

There is a similarity between the work presented in this research work and our coordination solution especially when dealing with developing countries. According to [15], a key determinant of success in E-Government initiative is based on the ability of the isolated, independent, heterogeneous computing systems of the different ministries to cooperate and work together. Therefore, the proposed architecture shall take into consideration existing independent software systems deployments and technical implementation rather than trying to replace them. However, the presented architecture is a centralized one and not flexible.

3.2 Agent and web service based approaches

Technical interoperability is not the only challenge in e-government. E-government is a dynamic environment Intelligent agents are known to be able to face this challenge. In fact, agents are software entities characterized by their autonomy, reactivity, pro-activity and social ability. Intelligent Agents have been extensively applied in the past for handling the distributed management of a wide variety of services (e.g., e-commerce, e-learning, e-recruitment, and so on). Their adoption in the context of e-government received less attention [16]. Therefore, there are few works using Software Agents in E-government field.

In this section, we introduce agent and Web Services research projects and discuss the improvement of our model compared to them.

In [17], authors proposed a Multi-Agent System that suggests to citizens the most interesting services. The authors argue that their system is able to consider the needs and preferences of citizens. In the E-government context, service composition is an important task. However, such a feature is not handled by [17]

In [18] and [19], interoperability architecture for E-government based on Software Agents and web services is presented. Reference [19] offers an implementation prototype for the described model presented in [18]. Two types of agent behavior are implemented: The first one is a broker agent that allows individual Local Information System to access the architecture. The broker agents are responsible for services publishing and discovery. The second type of agent acts as an aggregator of services that communicates with agents of others PA proving services.

However, there is no detailed description of service discovery and composition. Services repositories are not implemented in the prototype, so services are not published.

3.3 Summary of related work based on E-Government constraints

Table 1 summarizes related works based on E-Government constraints: Interoperability, flexibility, autonomy, distribution, type of requested service and project cost. For each constraint, we attribute (+) to the model that supports it and (-) to the model that does not support it.

All the approaches mentioned supports interoperability, flexibility, and autonomy since they use web services:

• Interoperability: the web services can be used by any application.

• Flexibility: Web Service provides a set of standards such as SOAP, WSDL, and UDDI which lead to flexibility in describing, discovering and invoking services.

• Autonomy: Government entities can have local control over the logic a service encapsulates.

Some of these approaches [17,18,19] use Intelligent agents in addition to the web service which increases the degree of flexibility and autonomy. However, these approaches except [18] and [19] are centralized. Besides, all these approaches are costly.

The study of the existing approaches shows that no solution fulfills all constraints identified for E-Government. In the next section, we will present our model that will meet the different E-government constraints required.

Technology	References	Interoperability	Flexibility	Autonomy	Distribution	Composite Services	Low Project Cost
Web Services	[14]	+	+	+	-	+	-
	[15]	+	+	+	-	+	-
Web Services & Agent	[17]	+	++	++	-	-	-
	[18] [19]	+	++	++	+	+	-

Table 1. Related work comparison

4 Proposed model

The proposed model is drawn inspiration from the work of [13]. The "model perceives government units as a collection of autonomous entities where there is little or no integration among them" [13]. The architecture that we propose is composed of two different layers: Interface layer and Web Service Layer as shown in Figure 1.

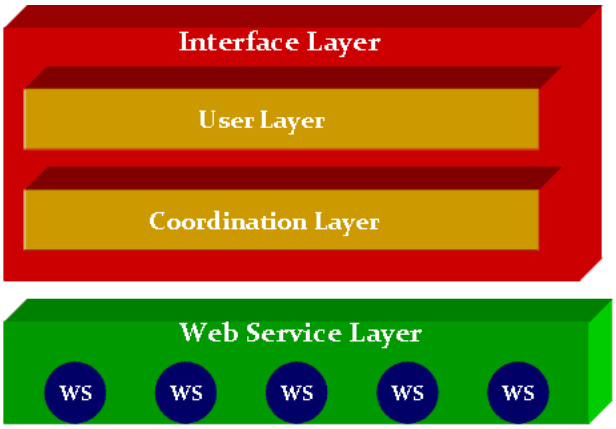

Fig.1. The architecture layout

4.1 Interface Layer or Government Entity Interface Layer

Interface Layer or Government Entity Interface Layer expresses the main contribution of our work. It presents the interface of each GE to be used by both citizen and Public Administration Employee. Each cooperating public administration can act both as provider of its own services and as requestor for services available on other public administrations. This Layer presents also the communication between GEs.

The interface layer is composed of a set of intelligent agents (IAs) that constitute a MAS as shown in Figure 2. An agent is "a real or virtual entity that is autonomous, it operates in its perceived environment able to act and interact with other agents" [20].

The studies from the literature highlight the following characteristics of an agent [5,20]:

• Autonomy: an agent has its internal state and controls its actions without the direct intervention of humans or others

• Reactivity: an agent is able to perceive its environment and acts rapidly in response to changes that can occur.

• Pro-activeness: agent does not simply act in response to its environment, it is able to take the initiative and exhibit goal-directed behavior.

• Social ability: an agent is able to cooperate and interact with other agents in order to realize collective tasks.

We have profited from these characteristics to represent our architecture by means of intelligent agents: (1) IOW and MAS show similar properties such as autonomy, flexibility, and distribution. (2) E-government environment is a dynamic one. E-Government services can be updated, removed or new ones can be added. Consequently, an intelligent agent is able to autonomously and dynamically adapt to e-government changing environment.

Roles of each type of agent are summarized in Table 2. It is divided into two Interfaces sub-layers: user layer and coordination layer. The UDDI registry is used by both sub-layers for service registration and service discovery.

4.1.1 User Layer

It constitutes an interface between the user and the system and is represented by a single agent, the Interface Agent. Two type of users can interact with this interface: Service provider (an administration Employee) and Service requester (An administration Employee or a citizen).

Two functionalities are assigned to the User Layer which are Service Publication and Service Request.

• Service Publication: The administration Employee can publish its services in the UDDI registry.

• Service Request: The Interface Agent is responsible for receiving Administration Employee or citizen requests, then sending them to the Discovery Agent (DA).

4.1.2 Coordination Layer

Four functionalities are assigned to this layer which are service discovery, service composition, service invoking and service coordination. For service publishing and discovery, we use a UDDI registry.

UDDI: The UDDI is an XML-based standard that defines a standard method for publishing, discovering, and managing information about Web service providers such as contact information, Web services, and the technical interfaces of Web services. UDDI is the most known and used registry.

UDDI defines a standard data structure for representing company and service description information [21].

A set of intelligent agents compose the coordination layer: Discovery Agent (DA), Composition Agent (CA) and Service Agent (SA).

Table 2. Roles assigned to each type of agent

Interface Layer	Agents	Roles
User Layer	Interface Agent	- User representative (citizen or Public Administration) - Receiving Requests
Coordination Layer	Discovery Agent	- Discovery - Coordination
	Composition Agent	- Composition
	Service Agent	-Service representative -Invocation -Coordination

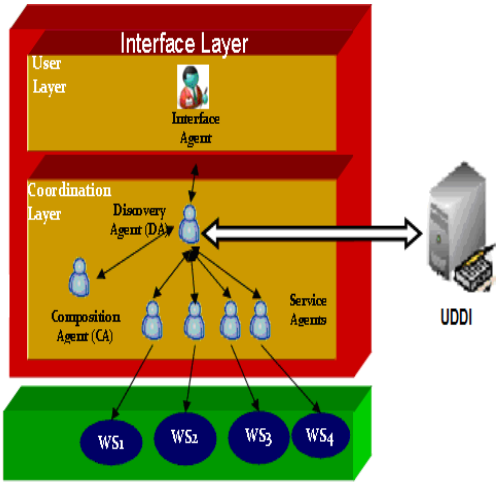

Fig. 1. The model architecture [2]

4.2 Web Service Layer

Web Services are software components that are self-containing, self-describing modular applications and can be published, located, and invoked across the Web. They allow applications to interoperate in a loosely coupled environment, discovering, and connecting dynamically to services without any previous agreements [22].

This Layer represents services offered by the GE. Public Administrations are service providers that describe and modify their services based on WSDL. In general, the service providers are the ones responsible for implementing the services they aim to offer.

5 Scenarios of simple and composite services

Each PA deploys an instance of the three layers as illustrated in Figure 3. The collaboration of the different PA involved in the execution of a requested composite service is realized through their Coordination Layer and in particular the communication between Discovery Agents and Services Agents.

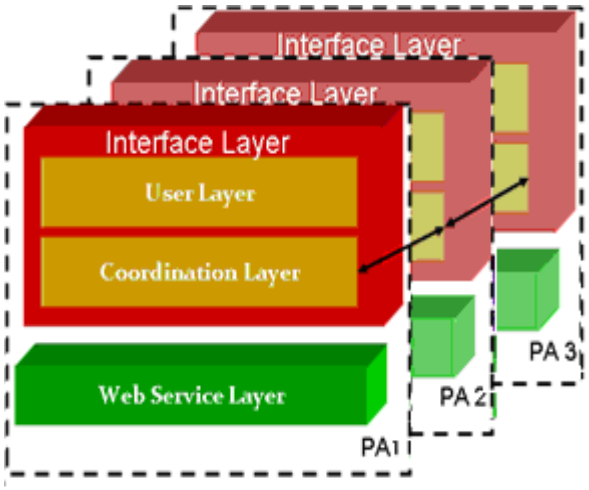

Fig. 2. Model instances and collaboration

Communication between agents is based on Foundation for Intelligent Physical Agent-Agent Communication Language (FIPA-ACL) messages. The Interface Agent receives a request from a citizen or an administration employee. Then, it sends it to the DA that discovers the service by mean of the UDDI Registry. Three scenarios can occur when discovering a service:

• **Scenario 1** (Figure 4): a simple service is executed by the same GE. If DA finds the requested service, this latter is a local simple one executed by the same Government Entity. The DA sends the service to the SA that will invoke the corresponding web service.

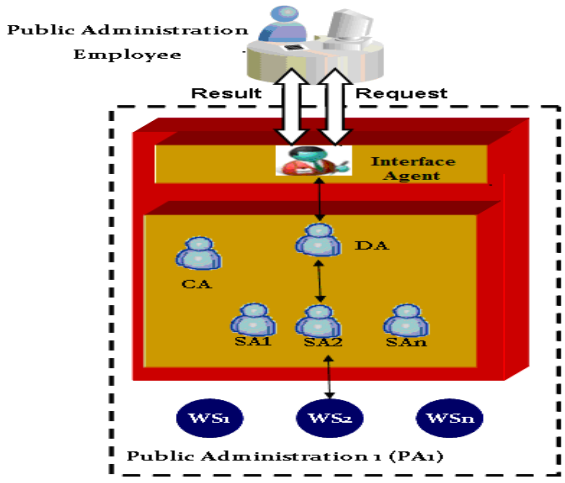

Fig. 3. Scenario 1 Simple service executed by the same GE

• **Scenario 2** (Figure 5): Simple service executed by another GE. DA finds that the discovered simple service is executed by another GE. So, it communicates with the corresponding SA.

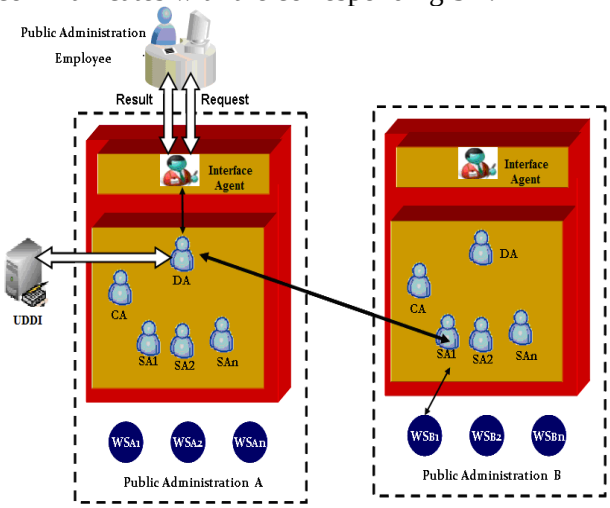

Fig.4. Simple service executed by another GE

• **Scenario 3** (Figure 6 and 7): Composite service

Case1: Composite service is composed of simple services executed by more than one Government entity. DA cannot find the service in the UDDI Registry. Therefore, it is a composite one. CA has the role of service composition. Once CA has completed the phase of composition, it sends the list of simple services that compose the service requested to the DA in order to be discovered. After that, the DA communicates with each SA of each GE responsible for the execution of a simple service as shown in Figure 6. Each SA returns the result of execution to the DA (Public Administration A).

Case2: If the execution of a simple service needs the output of execution of another service, then we use this type of coordination as shown in Figure 7.

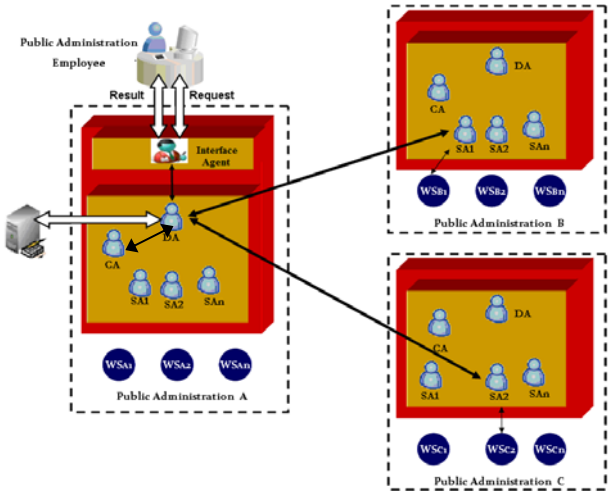

Fig.5: Composite service (case1)

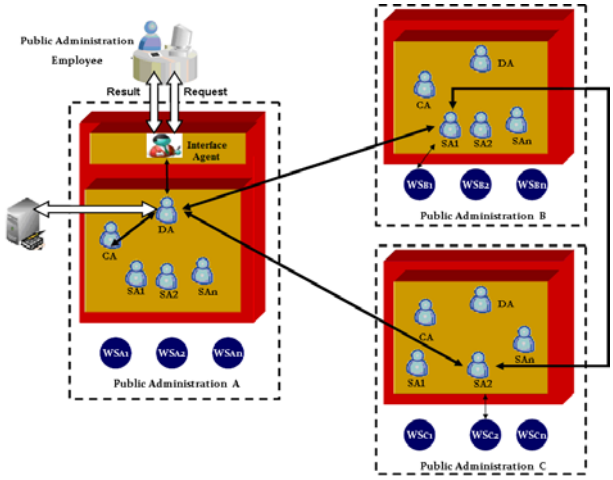

Fig.6: Composite service (case 2)

6 Case studies and experimental results

Two different case studies are carried out to demonstrate the applicability of our approach in the E-Government domain and verify the expected benefits. The first used case study is concerned with the process of creating an Electronic Identity Card (EIC) in Tunisia. The second one is concerned with the process of creating a health card for the national health insurance office (CNAM) in Tunisia (Figure 8).

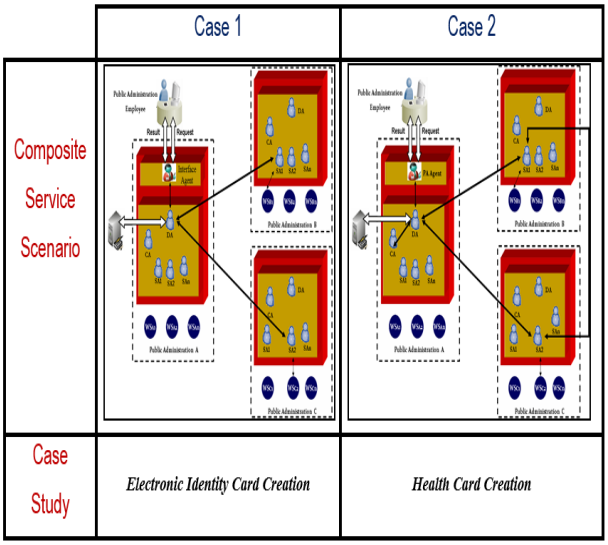

Fig.7. Case studies related to each case of composite service

6.1 Electronic identity card creation case study

The process of creating an electronic identity card is an inter-organizational process involving five government administrations: Ministry of the interior, Police Station, Municipality, Court and Work (or school). A detailed description of the case study is presented in [23]. Names and types of services associated with the above presented Public Administrations are shown in Table 3.

Table 1 Services offered by PAs

Public Administration	Service	type
Ministry of the interior	Identity Card Creation	composite
Police Station	address certificate	simple
Municipality	Birth certificate	simple
Court	nationality certificate	simple
Work /school	working certificate/ presence certificate	simple

6.1.1 Prototype implementation

The prototype of our system was created using Eclipse IDE Mars 4.5.1 (Integrated Development Environment for creation of java applications). One of the main reasons for choosing Eclipse was that it offers the possibility to integrate different frameworks such as JADE for Agent and Spring for agent services implementation.

1- Services description: Five services are developed using plug-in Spring of Java Enterprise Edition (J2EE): Birth certificate (municipality Service), address certificate (police station Service), nationality certificate (court Service), and working certificate (work Service), if the citizen is a worker, presence certificate (school service), if the citizen is a student.

The Service Description is based on the WSDL specification. This structure stores all the necessary information to be used in the architecture. It includes the identification of the service provider, as well as relevant information regarding the service delivery, such as data required and the service outcome.

Figure 9 contains an excerpt of the WSDL file of one of the services. The main elements of the 'getExtraitByIdentifiant' operation are shown inside the boxes. As input, this method receives a unique identifier of a citizen. The method returns information about citizen: Birthday (dateNaissance), City (lieu), Married to (marieA), LastName (nom), Mother's first name and lastName (nomPrenomMere), FirstName(prenom), father's first name and sex (sexe).

```
<?xml version="1.0" encoding="UTF-8" ?>
- <wsdl:definitions name="ExtraitServiceImplService" targetNamespace="http://spring.java.com/"
  xmlns:ns1="http://cxf.apache.org/bindings/xformat"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://spring.java.com/"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 - <wsdl:types>
  - <xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
     targetNamespace="http://model.java.com" xmlns:tns="http://model.java.com"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   - <xsd:complexType name="Extrait">
     - <xsd:sequence>
        <xsd:element minOccurs="0" name="dateNaissance" nillable="true" type="xsd:string" />
        <xsd:element minOccurs="0" name="id" nillable="true" type="xsd:int" />
        <xsd:element minOccurs="0" name="lieu" nillable="true" type="xsd:string" />
        <xsd:element minOccurs="0" name="marieA" nillable="true" type="xsd:string" />
        <xsd:element minOccurs="0" name="nom" nillable="true" type="xsd:string" />
        <xsd:element minOccurs="0" name="nomPrenomMere" nillable="true" type="xsd:string" />
        <xsd:element minOccurs="0" name="numRasm" nillable="true" type="xsd:int" />
        <xsd:element minOccurs="0" name="prenom" nillable="true" type="xsd:string" />
        <xsd:element minOccurs="0" name="prenomPere" nillable="true" type="xsd:string" />
        <xsd:element minOccurs="0" name="sexe" nillable="true" type="xsd:string" />
       </xsd:sequence>
     </xsd:complexType>
   </xsd:schema>
  - <xsd:schema attributeFormDefault="unqualified" elementFormDefault="unqualified"
     targetNamespace="http://spring.java.com/" xmlns:ns0="http://model.java.com"
     xmlns:tns="http://spring.java.com/" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
     <xsd:import namespace="http://model.java.com" />
     <xsd:element name="getExtraitByIdentifiant" type="tns:getExtraitByIdentifiant" />
    - <xsd:complexType name="getExtraitByIdentifiant">
      - <xsd:sequence>
```

Fig. 8. Birth certificate (Municipality Service)

6.1.2 MAS and sequence steps of the system

A MAS was developed based on the above framework using JADE (Java Agent DEvelopment Framework) plug-in of eclipse. JADE [24] is a FIPA compliant platform for developing Intelligent Agents (JADE version 4.5.0 has been used for the implementation of the framework). JADE is the most widely used agent platform for research projects worldwide and it is at a quite mature stage of development [25]. Agents communicate with each other using FIPA-ACL messages.

The sequence of the steps our system follows to send back either a success or a failure message of an EIC creation are described below:

• **Authentication:** If the citizen accesses the system for the first time, registration is mandatory. In the opposite, if he/she is already registered, an authentication process starts. We have used Spring security to connect to LDAP (Lightweight Directory Access Protocol) as a security framework for authentication. LDAP is an open, vendor-neutral, industry standard application protocol for accessing and maintaining distributed directory information services over an Internet Protocol (IP) network (Network Working Group). A common use of LDAP is to provide a central place to store usernames and passwords. This allows many different applications and services to connect to the LDAP server to validate users [26].

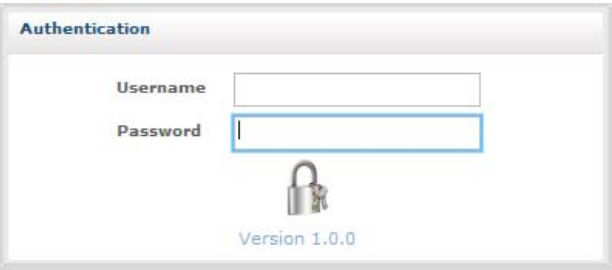

Fig.9. Authentication process

An Administrator of a public administration can access the system by using the interface of the application as depicted in Figure 11. After being authenticated, he has to choose his profile (Administrator, citizen).

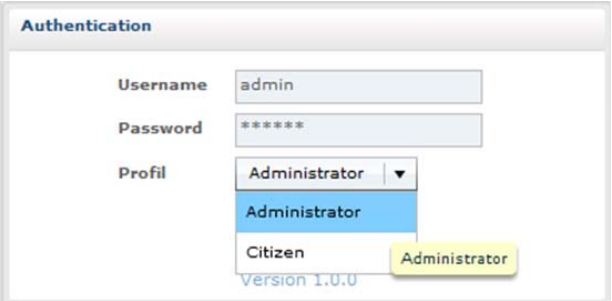

Fig. 10. User's Profile

• **Service publication:** Every PA exposes and registers its services with the UDDI registry. The service provider has to publish specifications about the service namely input/output parameters, methods implemented and access point (where it has been hosted and deployed). Only simple services are published. In the identity card case study, the administrator of each PAs (municipality, police station, court, work, and school) can use the interface of our framework (Figure 12) to publish their services in the UDDI registry. We used JUDDI V3.3.4 for service publication and discovery. JUDDI requires the use of a relational Data Base Management System. For this purpose, we have used Oracle 11g Express Edition.

To publish a web service, the administrator of the PA has to register his demand (Figures 12-13). After being accepted, he can publish the web service in question (Figures 14-15).

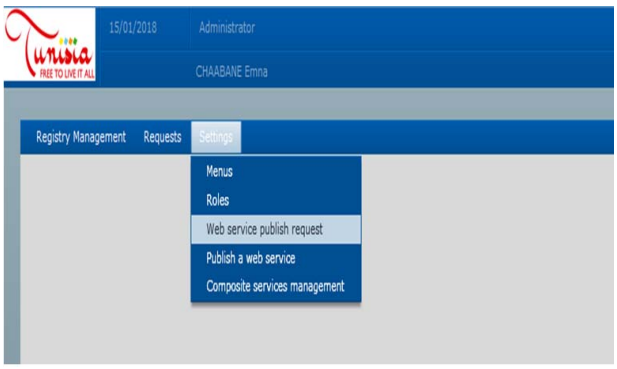

Fig. 11. Web Service publication demand

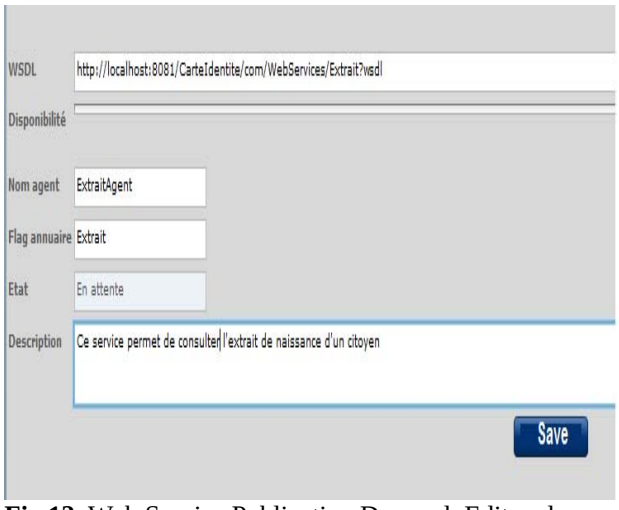

Fig.12. Web Service Publication Demand: Edit and save

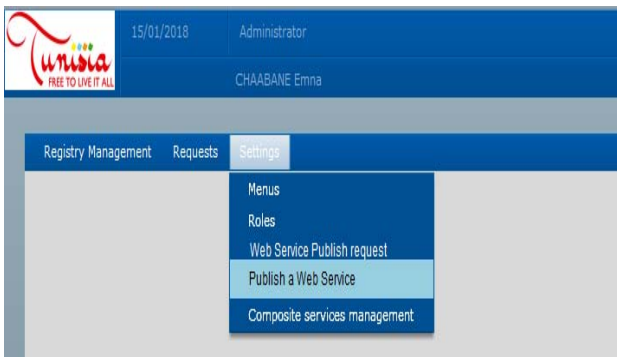

Fig. 13. Web service publication

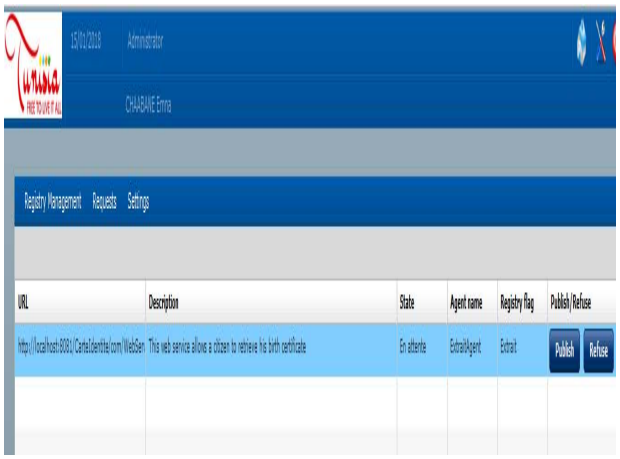

Fig. 14. Web service publication

• **Service request**: A Citizen or an administrator of a Public Administration can use the interface Agent to request a service. A request can be for a simple service such as Birth Certificate Service (Figure 16) or for a composite one such as in the case study of the Identity Card Creation (ICC). In this latter, the requester is a citizen. After being authenticated, the citizen chooses Identity Card Creation Service from a list of e-government services (Figure 17). The Interface Agent receives the ICC service request and sends it to the Discovery Agent. A sequence diagram (Figure 18) illustrates the messages exchanged between agents.

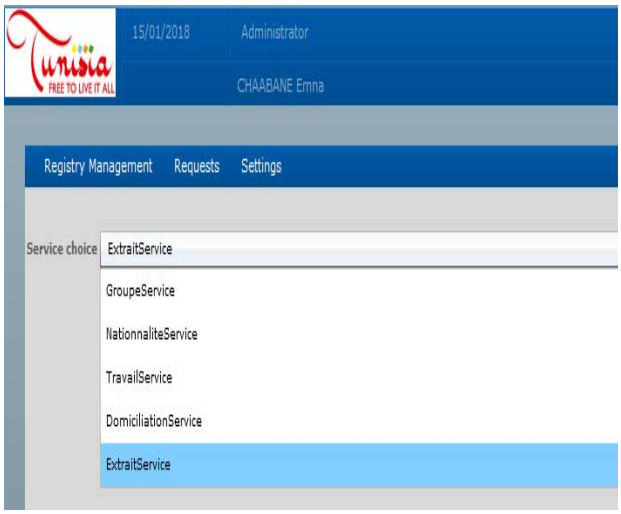

Fig. 15. Simple service request (Birth Certificate Service) by citizen

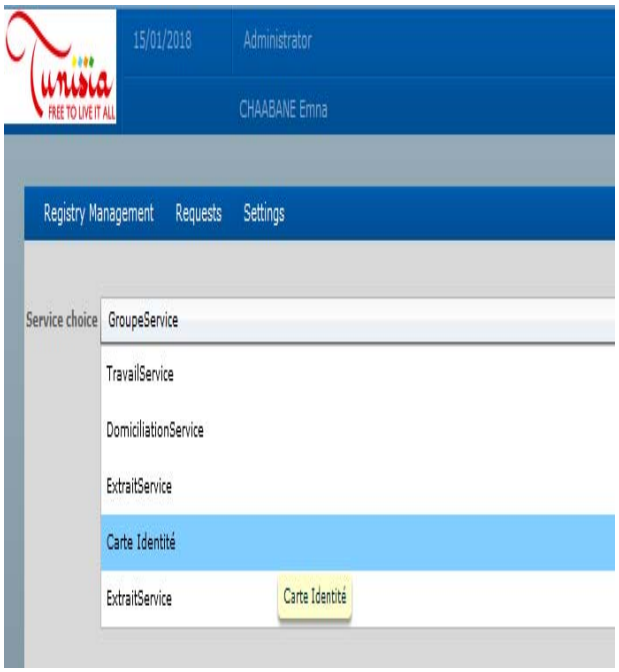

Fig. 16. Composite Service (Identity Card Creation) request

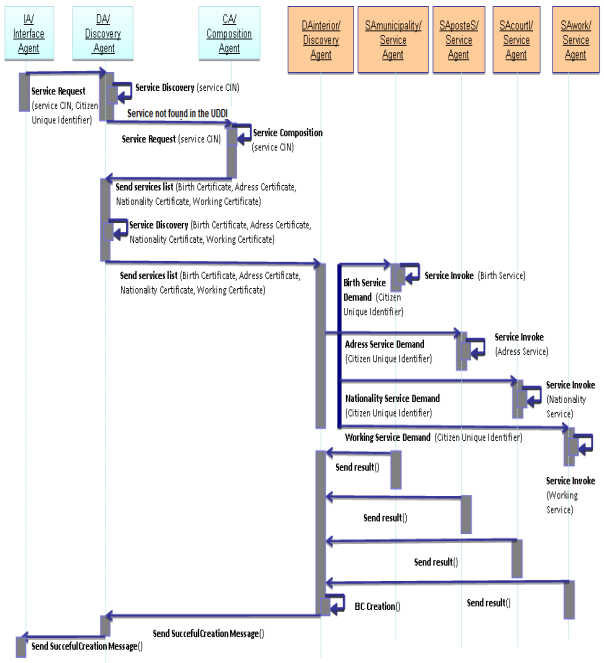

Fig. 17. Sequence diagram of agent's communications

• **Service Discovery:** The Discovery Agent can access the UDDI registry to search the most suitable service that fulfills the service request. To find services our approach uses the Java API for XML Registries (JAXR), which provides a uniform and standard Java API for accessing different kinds of XML registries including the UDDI. The client needs not to know the implementation details of a service. The agent gathers service parameters, from the service description file, namely service name, method name, input method parameters, and SOAP URL. In the current Implementation, the discovery process is based only on syntax similarities. The Discovery Agent has to find the same name of the service requested. In the case of a simple service request (BirthCertificate Service), the discovery agent searches the requested web service in the UDDI registry and finds it then it communicates with the corresponding Service agent to be invoked as shown in Figure 19. In our case (Identity Card creation), there is no service that can match the desired request. Therefore, it is a composite service that needs a service composition.

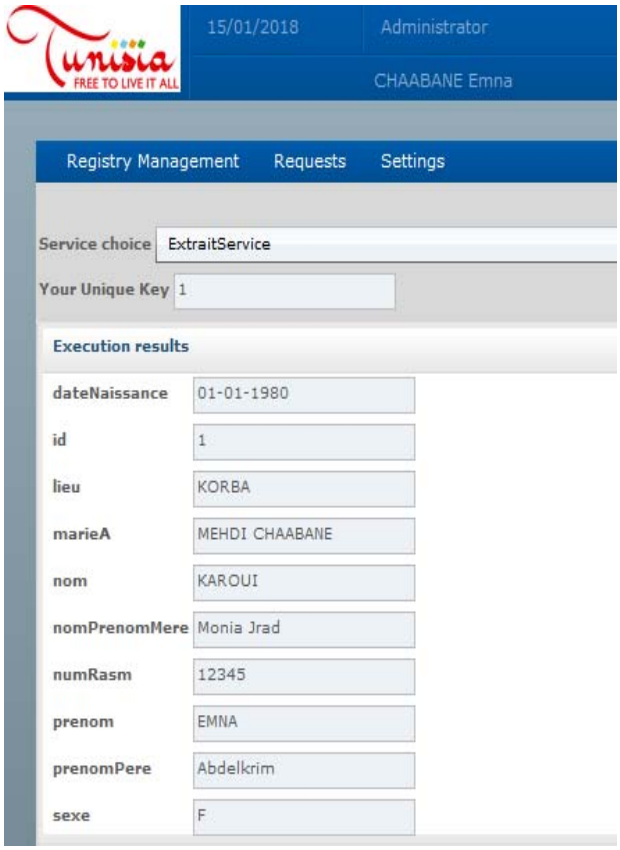

Fig. 18. The execution result of BirthCertificate Web Service

• **Service Composition:** Web service composition refers to the process of creating a composite service, offering new functionality, from simple existing Web services. Given that none of the available services matches the requested one, the discovery agent sends the composite service request to the Composition Agent that decomposes the service into simple services. The list of simple services is sent back to the discovery agent to be discovered (Figure 20). Once simple services are discovered, a list is sent to the DA of the ministry of the interior. This PA is responsible for Identity Card Creation. The DA of the ministry of the interior has to coordinate with others PAs Agents. Therefore, it communicates with the different PAs involved in the execution of the ICC. It sends each simple service to the corresponding SA of the four PAs (Municipality, Police Station, Court and Work).

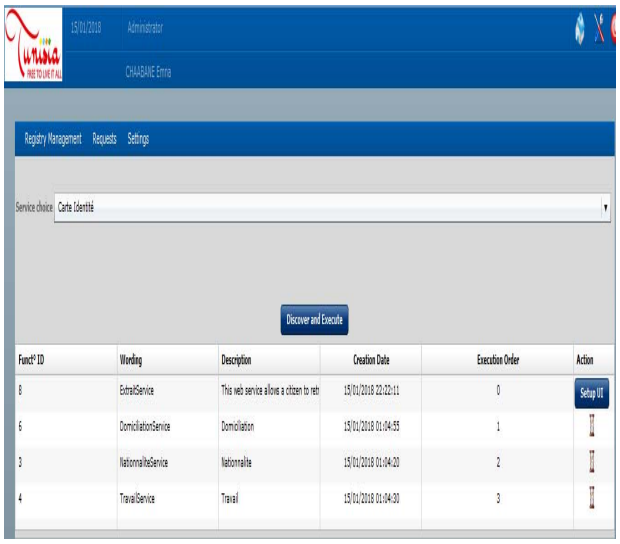

Fig. 19. List of simple services to be discovered and executed

• **Service Invocation:** Each SA invokes the corresponding service using a unique identifier as input. The services outcomes are then sent back to the DA of the ministry of the interior (Figure 21) which creates the EIC and presents the successful creation message to the final user (citizen) (Figure 22).

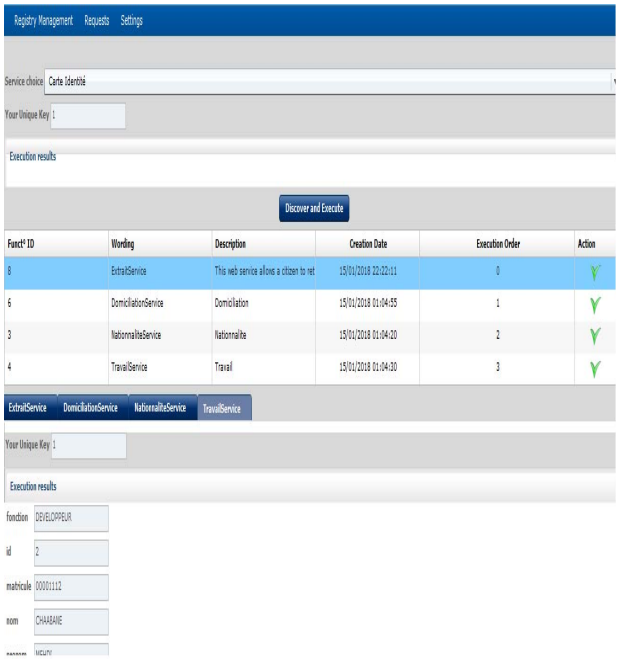

Fig. 20. Invocation of the simple services that compose the Identity Card Creation Service

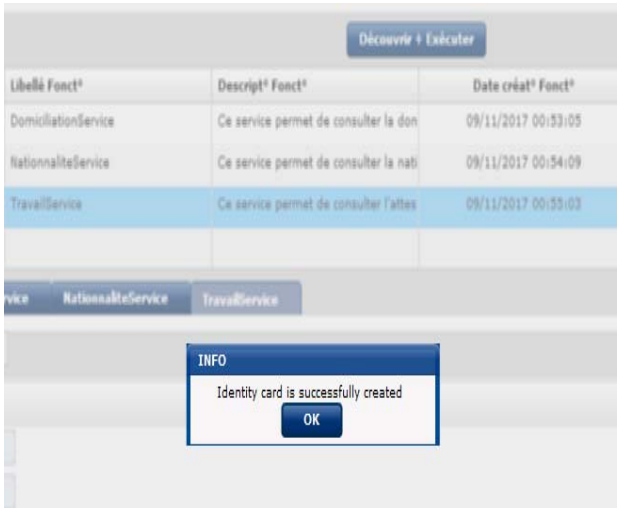

Fig. 21. Successful Identity Card Creation

6.2 Health card creation case study

The second used case study concerns the process of Health Card Creation in Tunisia.

The process of creating a Health Card involves two Government entities, which are Municipality and National Health Insurance Office (NHIO) Known as CNAM in Tunisia.

To create a Health card, a citizen needs to present his/her birth certificate, his/her children's birth certificates (if he/she has children) and his security social number to the NHI Office.

To automate the process of creating an Identity card, a service requester (citizen in our case) uses the user interface of our model to send his request as shown in Figure 23.

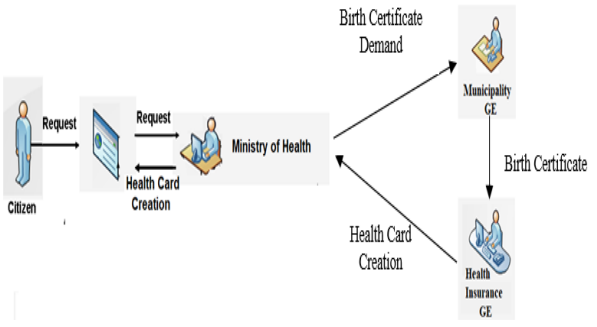

Fig. 22. Health Card Creation Case Study

The creation of the health card is a composite service that is composed of two sub-services: Birth certificate Service and Health Card Folder Creation Service (Table 4).

Table 2: Services offered by PAs

Public Administration	Service	type
Ministry of Health	Health Card Creation	composite
Municipality	Birth certificate	simple
NHIO	Health Card Folder Creation	simple

In opposition to the first case study, the sequential execution of the sub-services is mandatory. For this reason: the output of the execution of the first simple service (Birth certificate) is the input for the second simple service (Health Card Folder creation).

After being authenticated and after publishing the simple services (Birth Certificate and Health Card Folder Creation) by their corresponding Public Administrations, the citizen chooses Health Card Creation Service (CarteSoin) from a list of e-government services as shown in Figure 24. The Interface Agent receives the service request and sends it to the Discovery Agent.

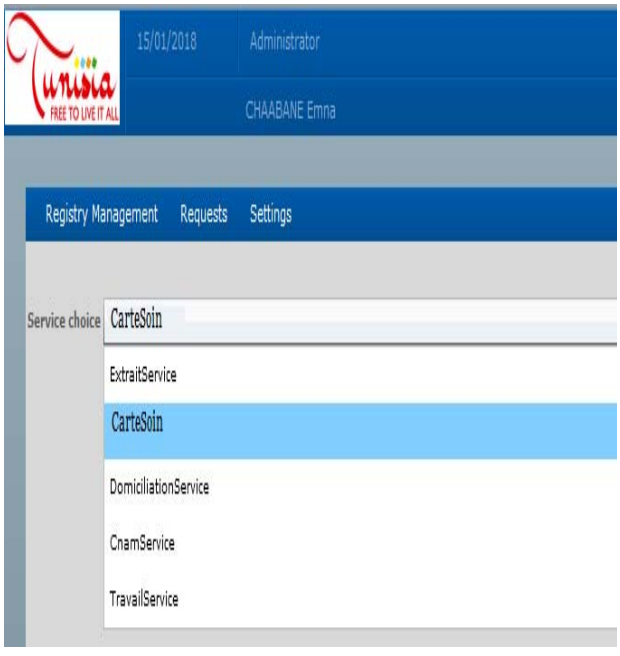

Fig. 23. Composite service (Health Card Creation) request

- The Discovery Agent accesses the UDDI. Yet, it cannot find the requested service. Consequently, it is a composite service that needs a composition stage.

- The Discovery Agent sends the composite service request to the Composition Agent that decomposes the service into simple services. The list of simple services is sent back to the discovery agent to be discovered (Figure 25). Once simple services are discovered, the DA communicates with the SA of the Municipality PA and sends the first simple service (Birth Certificate service).

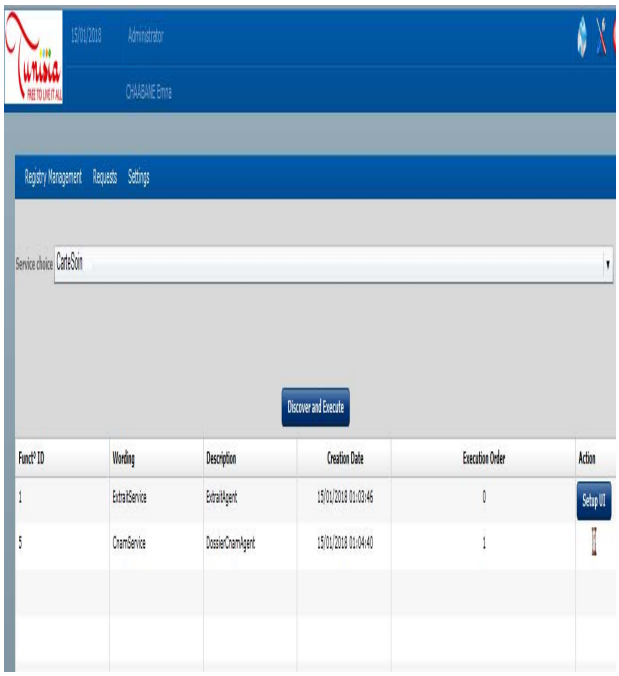

Fig. 24. List of simple services to be discovered and executed

The SA of the Municipality PA invokes Birth Certificate service using a unique identifier as input. The service outcome (Birth certificate) is sent to the SA of NHIO which invokes the Health Card Folder Creation service (Figure 26).

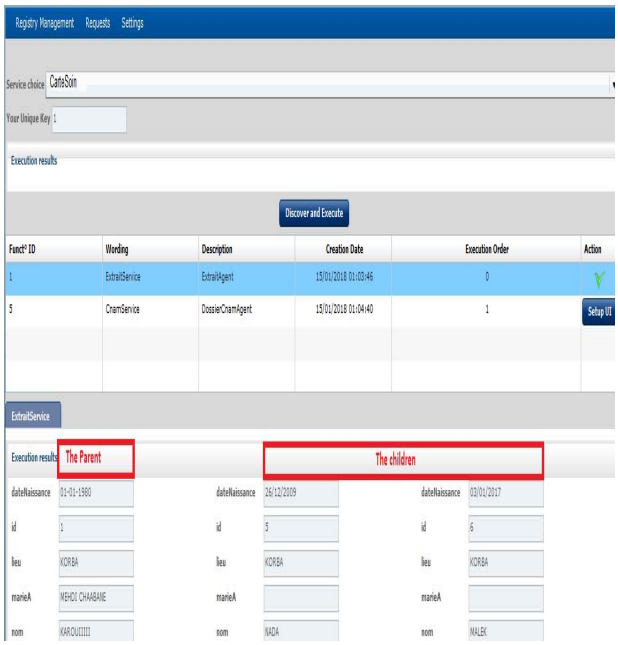

Fig. 25. Invocation of the simple services that compose the Health Card Creation Service

After Confirming the creation of the health card (Figure 27), the latter is then created (Figure 28).

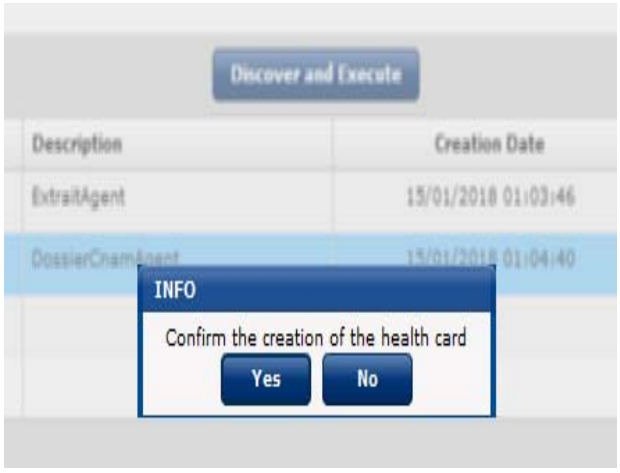

Fig. 26. Health Card Creation Confirmation

Fig.27. Health Card is created successfully

7. Conclusion

In this paper, we proposed a new IOW architecture applied in the e-government field. This architecture enables us to create a framework for e-government processes that meets the distribution criterion of GE. It can be easily used by citizens and administrations employees since they are able to express their request by a simple choice of the desired service from a list of government services. Moreover, the system is distributed over different GE so that it reduces communication cost and transaction time. The collaboration between government administrations is based on interfaces communication. Our architecture is suitable for developing countries such as Tunisia where GEs are not necessarily equipped with Workflow systems. When analyzing WS and IA based E-Government architectures, we found that: (1) a composite service request is not treated by some of them. The e-government processes are generally composite ones. These systems fail when the requested service is a composite one. (2) Even though a composition approach is adopted by some other approaches, they fail to meet an important requirement which is distribution. Many problems arise: (1) the use of a middleware centralizes the web service life cycle: discovery, composition, selection, and invocation in one component. (2) There is no direct collaboration between government administrations. Consequently, the transaction time and the communication cost increase. (3) High operational and maintenance cost.

Compared to the analyzed architectures, our model successfully meets all the E-Government requirements.

References

1. R. Silcock, What is E-Government' Hansard Society for Parliamentary Government, Parliamentary Affairs, 2001, 54, pp. 88–101 (2001)
2. E. Karoui Chaabane, S.Hadouaj, K. Ghedira, Distributed Government Architecture Based on Intelligent Agents and Web Services. *Proceedings of IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology* (WI-IAT'2015) (2015)
3. R. Thangavel, B. Palanisamy, Efficient *Approach* Towards an Agent-Based Dynamic WebService Discovery Framework with QoS Support. *International Symposium on*

Computing Communication, and Control(ISCCC), pp74-78 (2011)
4. G. Amirthayogam, M. Rathinraj, A. Gayathri, WebService Discovery with QoS An Agent-Based Approach. J. IJIRSET, **1**, pp1-5 (2013)
5. M.J. Wooldridge, N. Jennings, Intelligent agents: Theory and practice. KER, **10**, pp. 115-152(1995)
6. E. Karoui Chaabane, S. Hadouaj, K. Ghedira, Multi-Agents Coordination Model in Inter-Organizational Workflow: Applying in Egovernment.J. WASET, **7,** pp.372-379 (2013).
7. M. Ansari, M. A. Usman, MZA. Nadeem, S. Raza, Multi-agent Based Semantic E-government Web Service Architecture Using Extended WSDL. IAT Workshops, pp. 599-604 (2006)
8. A. Gugliotta, L. Cabral, J. Domingue, V. Roberto, A semantic web service-based architecture for the interoperability of e-Government services. *Proceedings of the International Workshop on Web Information Systems Modeling*, Sydney, Australia (2005).
9. M. Divitini, C. Hanachi, C. Sibertin-Blanc, Inter-organizational Workflow for Enterprise Coordination. in A. Omicini, F. Zambonelli, M. Klusch, & R. Tolksdorf, (eds), *Coordination of Internet Agents: Models, Technologies, and Applications*, pp. 369-398, Springer (2001).
10. I. Chebbi, S. Dustdar, S. Tata, The view-based approach to dynamic inter-organizational workflow cooperation. J.DKE, **56**, pp.139-173 (2006).
11. B. Medjahed, A. Bouguettaya, *Service composition for the semantic web*, Springer, Berlin (2011)
12. A. Berrais, Modèles de coordination multi-agents pour le workflow interorganisationnel: Application à l'e gouvernement. Master thesis, Higher Institute of Management, Tunis, Tunisia (2006).
13. A. Rabaiah, E. Vandijck, Federation of E-government: A Model and Framework, JISAC, **4**, pp. 1-4 (2007)
14. Z. Saleh, R. Obeidat, Y. Khamayseh, A Framework for an E-government Based on Service Oriented Architecture for Jordan. J. IJIEEB, **3**, pp.1-10 (2013)
15. Z. Al-Khanjari, N. Al-Hosni, N. Kraiem, Developing A Service Oriented EGovernment Architecture Towards Achieving E-Government Interoperability. J. IJSEIA, **8**, pp. 29-42 (2014)
16. A. Tebib, M. Boufaida, Interoperability of Services in E-Government using Intelligent Agent, EGP. *Proceeding of the Third International Conference on Computer Science and its Applications* (CIIA'11), Algeria (2011)
17. P. De Meo, G. Quattrone, G. Terracina, D. Ursino, Utilization of Intelligent Agents for supporting citizens in their access to e-government services.J.WIAS, **5**, pp. 273–310 (2007).
18. F. Marques, GP. Dias, A. Zúquete, A General Interoperability Architecture for e-Government based on Agents and Web Services. *Proceeding of the 6th Iberian Conference of Information Systems and Technology*, pp. 338-343 (2011)
19. F. Marques, G. Dias, A. Zúquete, Agent-Based Interoperability for e-Government. *Distributed Computing and Artificial Intelligence*, pp. 561-568 (2013).
20. J. Ferber, Les systems multi-agents : vers une intelligence collective. Paris (France), InterEditions (1995).
21. M.P. Papazoglou, WEB SERVICES: PRINCIPLES AND TECHNOLOGY. 1st ed. [ebook] England: Pearson Education Limited. Available at:http://www.nortonaudio.com/Ficheiros/Web. services...principles.and.technology.pdf (2008)
22. P. Palathingal, S. Chandra, Agent Approach for Service Discovery and Utilization. *Proceedings of the 37th Annual Hawaii International Conference on System Science HICSS-37*, Hawaii, pp.1-9 (2004)
23. E. Chaabane, S. Hadouaj, K. Ghedira, Agent and Semantic Web Service Based Model in Inter-Organizational Workflow for EGovernment. *Proceedings of the 14th European Conference on E-Government*, Romania (2014)
24. F. Bellifemine, G. Caire, T. Trucco, G. Rimassa, Jade Programmer's Guide. JADE 2.5 (2002)
25. F.G. Sanchez, L.A. Sabucedo, R. Martinez-Bejar, L.A.Rifon, R.V. Gacia, J.M. Gomez, Applying intelligent agents and semantic web services in eGovernment environments. I*JES*, **28**, pp.416-436 (2011)
26. OpenLDAP, Introduction to OpenLDAP Directory Services. Available from http://www.openldap.org/doc/admin24/intro.html (2011)