Development of knowledge-based web services to promote and advance Industrial Symbiosis in Europe (**eSymbiosis**)

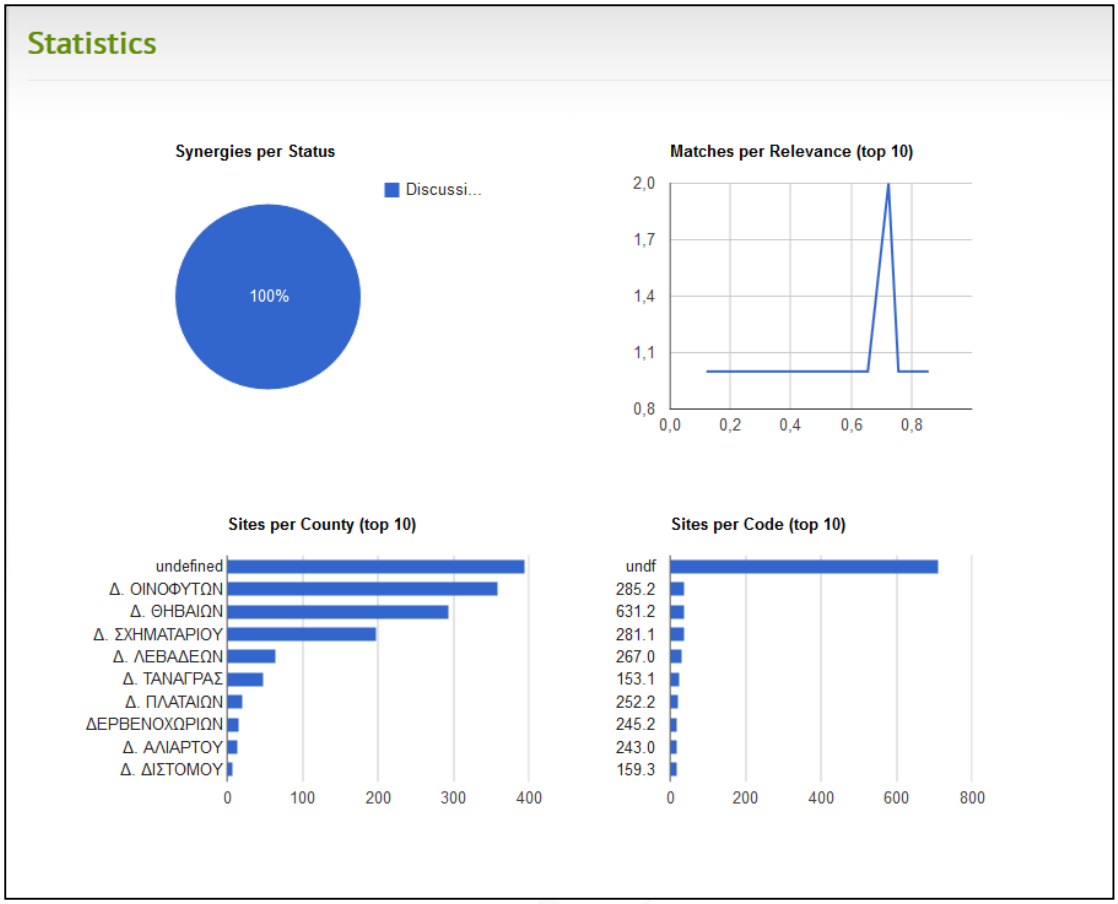
LIFE09/ENV/GR/000300

ACTION 2: Service architecture and implementation

D2.4 Design and implementation of semantic web service portal

May 2013

INDEX


Statistics
Synergies per Status
Discussi...
100%
Matches per Relevance (top 10)
2,0
1,7
1,4
1,1
0,8
0,0
0,2
0,4
0,6
0,8
Sites per County (top 10)
undefined
Δ. ΟΙΝΟΦΥΤΩΝ
Δ. ΘΗΒΑΙΩΝ
Δ. ΣΧΗΜΑΤΑΡΙΟΥ
Δ. ΛΕΒΑΔΕΩΝ
Δ. ΤΑΝΑΓΡΑΣ
Δ. ΠΛΑΤΑΙΩΝ
ΔΕΡΒΕΝΟΧΩΡΙΩΝ
Δ. ΑΛΙΑΡΤΟΥ
Δ. ΔΙΣΤΟΜΟΥ
0
100
200
300
400
Sites per Code (top 10)
undf
285.2
631.2
281.1
267.0
153.1
252.2
245.2
243.0
159.3
0
200
400
600
800

..... 70

Revision History

Revision	*Description*	*Date*	*Authors*	*Notes*
Draft v.0.1	Initial Version Document	30/05/2013	CLMS	Split documents for Activities 2.3, 2.4. Changed Formatting.

1 INTRODUCTION

1.1 WEB SERVICE PORTAL OF INDUSTRIAL SYMBIOSIS

This document examines in depth the design and implementation of the semantically enriched web portal that enables participants to register and to participate in the IS ecosystem (*Activity 2.4: Design and implementation of semantic web service portal*). The high level modules that materialize this aspect of functionality are the Information Portal, the CRM, and the Search facility.

Before examining in detail the aspects of the platform that refer to Activity 2.4, it is worth presenting the overall high level architecture and the consisting components of the platform from a functional perspective.

1.2 HIGH LEVEL ARCHITECTURE AND COMPONENTS

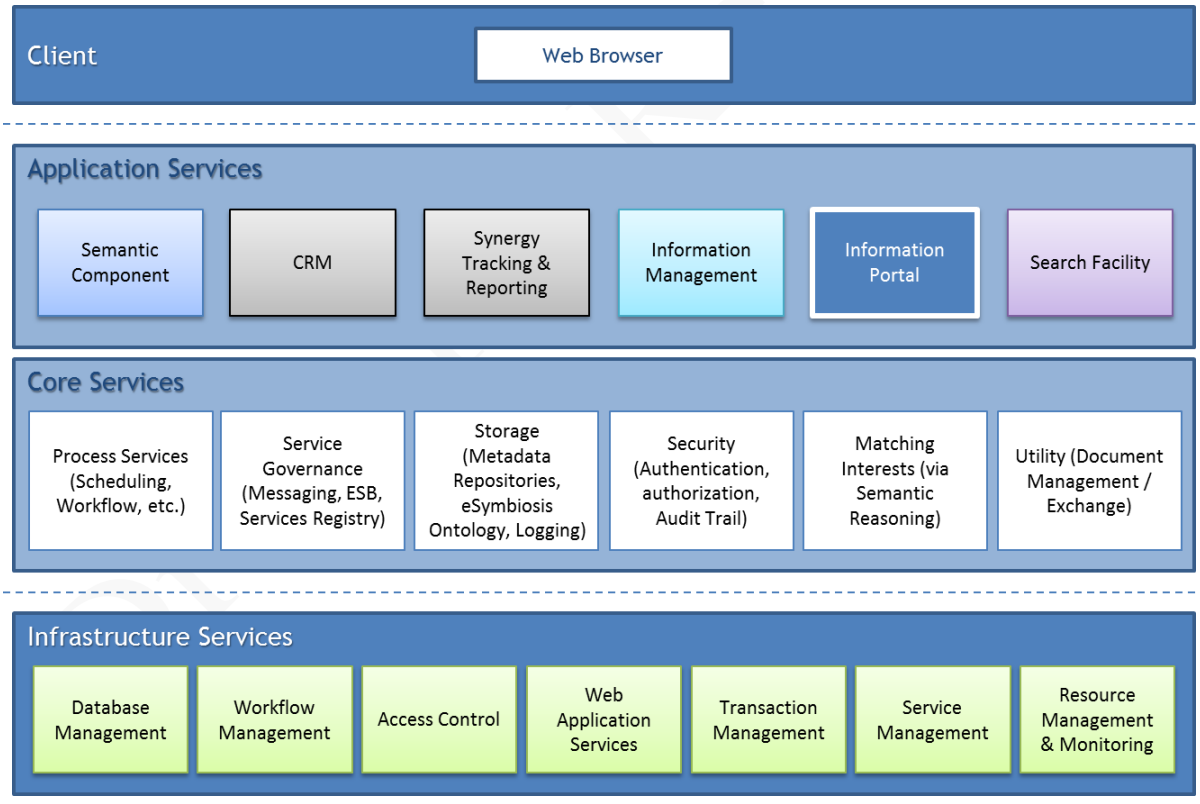


Figure 1: High level architecture of eSymbiosis platform

The above diagram is a technical diagram based on a number of assumptions following the SOA philosophy. It identifies mainly two categories of services:

1. *Application Services* and
2. *(Platform) Core Services.*

These services use a number of Infrastructure services provided by the various technologies that are used in the project, including semantic technologies (e.g. Jena), RDBS (e.g. SQL Server), Communication (e.g. WCF) etc.

The high level functional components of the eSymbiosis platform are presented in the diagram of Figure 1.

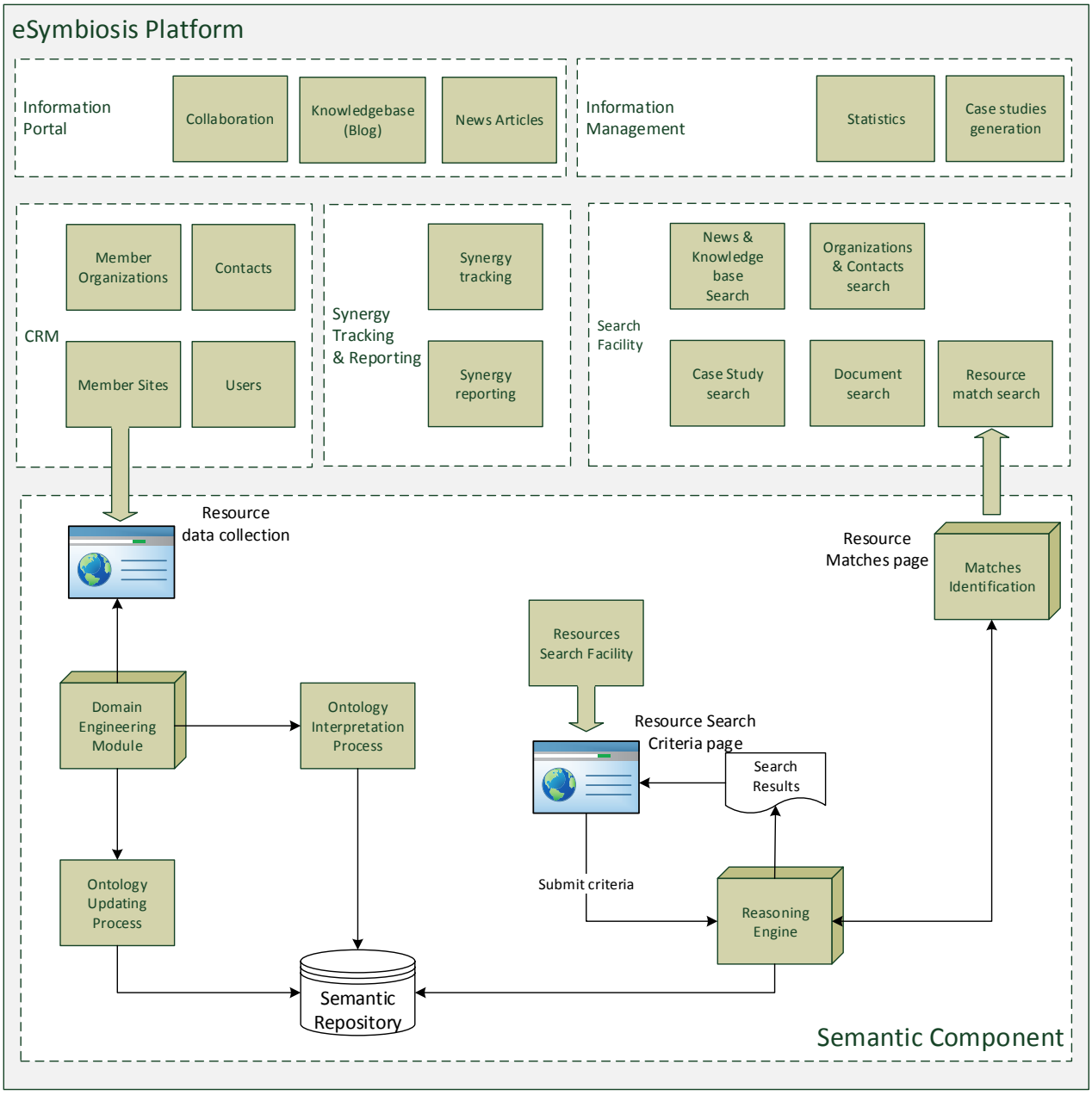


Figure 2: Functional components of eSymbiosis platform

The components and their services, as demonstrated in the above figure, are:

1. Semantic Component
2. Information Portal
3. CRM

4. Search Facility
5. Information Management
6. Synergy Tracking & Reporting

The current document focuses on the components of Information Portal, of CRM, and of Search Facility. A more detailed analysis of the aforementioned components follows.

2 INFORMATION PORTAL

The Information Portal component includes all modules that support the eSymbiosis Portal.

2.1 NEWS ARTICLES

Using this module the user is able to publish and view news articles. News is categorized using tags, so that users can browse directly to news relevant to a particular subject area. The ability to publish items is controlled by the security model.

An example of a News Article is shown in **Error! Reference source not found.**.

Figure 3: A sample News Article.

2.2 KNOWLEDGE BASE

The knowledge base includes a collection of the success stories of the project, which are based on the case studies of completed synergies. The case studies are generated by the system, (see the description of **Error! Reference source not found.** below), with minimal input from the IS Practitioner, and can be used as a reference point for the purposefulness of the overall project.

2.3 COLLABORATION

The system promotes the collaboration of participating organisations in multiple ways. This dedicated collaboration component handles the communication of participants or IS Practitioners that are cooperating within the context of a Synergy in two ways (which are presented in the runtime context in the respective section **Error! Reference source not found.**):

- Notification system, which informs the Organisation/IS Practitioner about important changes of a Synergy that he/she is involved in.

- Comments that the user provides for the other actors of the Synergy, in order to coordinate activities or resolve potential pending issues. The other users is notified about the new comments by notifications.

3 CRM

The CRM functionality is focused on the management of:

- IS Practitioners
- Member Organisations
- Member Sites
- Contacts
- Users

Depending on the access rights for the management of the aforementioned entities, the functionalities of CRM can be categorized into Public registration (for Organisation/Site/Resource) and Administrator registration.

3.1 PUBLIC REGISTRATION

The CRS component provides a public registration functionality, which is used to register as a member user and to define Organisations, Sites, and Resources. Each organization has an online 'record card', which will display a wide range of information about it, such as:

- Name
- Branch/Site
- Status (active member/dormant member/stakeholder etc.)
- Addresses (mail including postcode, email and web)
- Phone and fax numbers
- Region/area of Greece
- Associated matches
- Associated resources (haves/wants)
- Events which contacts within the organization have attended
- Recent activity (emails, documents, etc.) associated with the Organization

This organisation list provides live links to other details within the system. Lists of synergies, matches, etc. also include hyperlinks to referenced information. A security layer controls all access to the system's information so that detailed information of a commercial or confidential nature can be shared by an appropriate, controlled group of users.

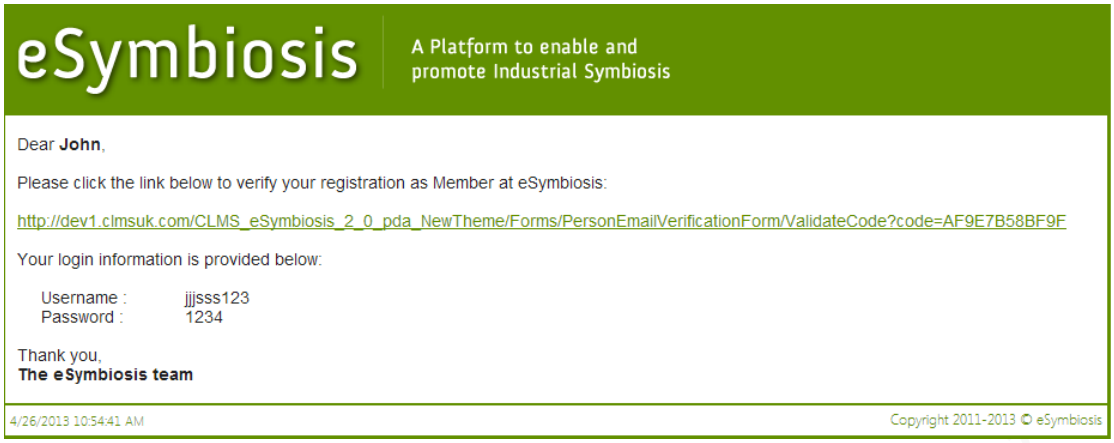


Figure 4: Example of the verification email sent to a new registrant

First, the CRM component handles the registration of a new member. Then, the CRM is responsible for the participant Organisation registration process. When a user first creates an account, a confirmation email is sent to the user, containing a URL that verifies the registration (see Figure 4). If the user accesses the link, the member user account is verified and the user can log in to the platform. After logging in, the portal allows the creation of a new Organisation under this user. As a consequent step, the user adds the Site information that belongs to this Organisation and the resources that are provided or requested by each Site. As the Site and Resource data need to be recorded in the ontology of the Semantic Component, the CRS system communicates via web services with the Semantic Component, to register the relevant information and to acquire a summary of each property of the resource. Communication with the semantic component is shown in the relevant sequence diagram (the CRM invokes two services of the Semantic Component, to register Site and Resource; the semantic component returns the stored resource summary by

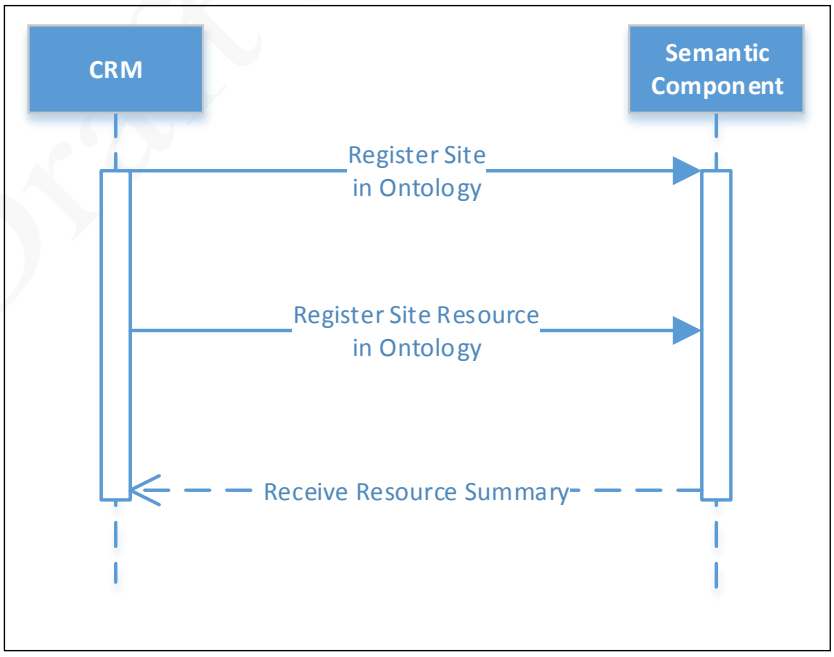


Figure 5: The sequence diagram of CRM and Semantic Component communication

invoking a service of the CRM).

The resource properties are provided by the semantic component in an xml string (which is named *resource summary*), in order to maintain a dynamic list of properties for each resource. An example of the resource properties xml is provided below.

```
<object xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns" type="ResourceProducer">
        <property name="confidentialityFlag" rdf:type="http://www.w3.org/2001/XMLSchema#boolean">
        False</property>
        <output type="ResourceByType">
                <property name="hasInterval" type="Random">
                        <label xml:lang="el">Τυχαίο</label>
                        <label xml:lang="en">Random</label>
                </property>
                <property name="hasStorageMethod" type="Open_topped_tank">
                        <label xml:lang="en">Open Topped Tank</label>
                        <label xml:lang="el">Ανοιχτή Δεξαμενή</label>
                </property>
                <property name="hasUnitOfMeasurement" type="m2">
                        <label xml:lang="en">Square metres (m2)</label>
                        <label xml:lang="el">Τετραγωνικά Μέτρα (m2)</label>
                </property>
                <property name="hasPatternOfSupply" type="Seasonal">
                        <label xml:lang="el">Εποχιακή</label>
                        <label xml:lang="en">Seasonal</label>
                </property>
                <property name="hasDeliveryMethod" type="Tipper">
                        <label xml:lang="en">Tipper</label>
                        <label xml:lang="el">Ανατρεπόμενο Όχημα</label>
                </property>
                <property name="isBatch" rdf:type="http://www.w3.org/2001/XMLSchema#boolean">
        False</property>
                <property name="deliveryCapability"
rdf:type="http://www.w3.org/2001/XMLSchema#boolean">          True</property>
                <property name="validFrom"
rdf:type="http://www.w3.org/2001/XMLSchema#date">2012/12/03</property>
                <property name="validTo"
rdf:type="http://www.w3.org/2001/XMLSchema#date">2012/10/03</property>
                <property name="hasName"
rdf:type="http://www.w3.org/2001/XMLSchema#string">ResourceName1</property>
                <property name="hasQuantity"
rdf:type="http://www.w3.org/2001/XMLSchema#float">3</property>
                <property name="isHazardous" rdf:type="http://www.w3.org/2001/XMLSchema#boolean">
        True</property>
                <property name="hasQuantityType" type="Solid">
                        <label xml:lang="el">Στερεό</label>
                        <label xml:lang="en">Solid</label>
                </property>
                <property name="needsSpecialHandling"
rdf:type="http://www.w3.org/2001/XMLSchema#boolean">True</property>
                <property name="hasStorageCapacity"
rdf:type="http://www.w3.org/2001/XMLSchema#float">34 </property>
        </output>
</object>
```

Each property has an XPath element, in order to extract the particular property's value from the xml of the resource summary. In this way, a dynamic set of properties can be provided for each resource by the semantic component. Moreover, the resource and its properties are maintained only in the Semantic Component and any changes are replicated to the other related components that need to access the resource details. The list of the resource properties is maintained by the administrator, within the eSymbiosis platform, with a user-friendly interface (see Figure 7 and Figure 6).

In addition, a member of an Organisation can invite other users in the same Organisation (i.e. an Organisation can be managed by multiple users). The process is as follows: the member user sends an invitation to a person using the platform; the invitation recipient receives an email with a unique registration URL; if the user proceeds to clicking the URL and filling the requested user account data, a new member is added as representative of the Organisation. Moreover, the Organisation user can review at any time the status of the sent invitations from the respective web form *My Invitations* of the system.

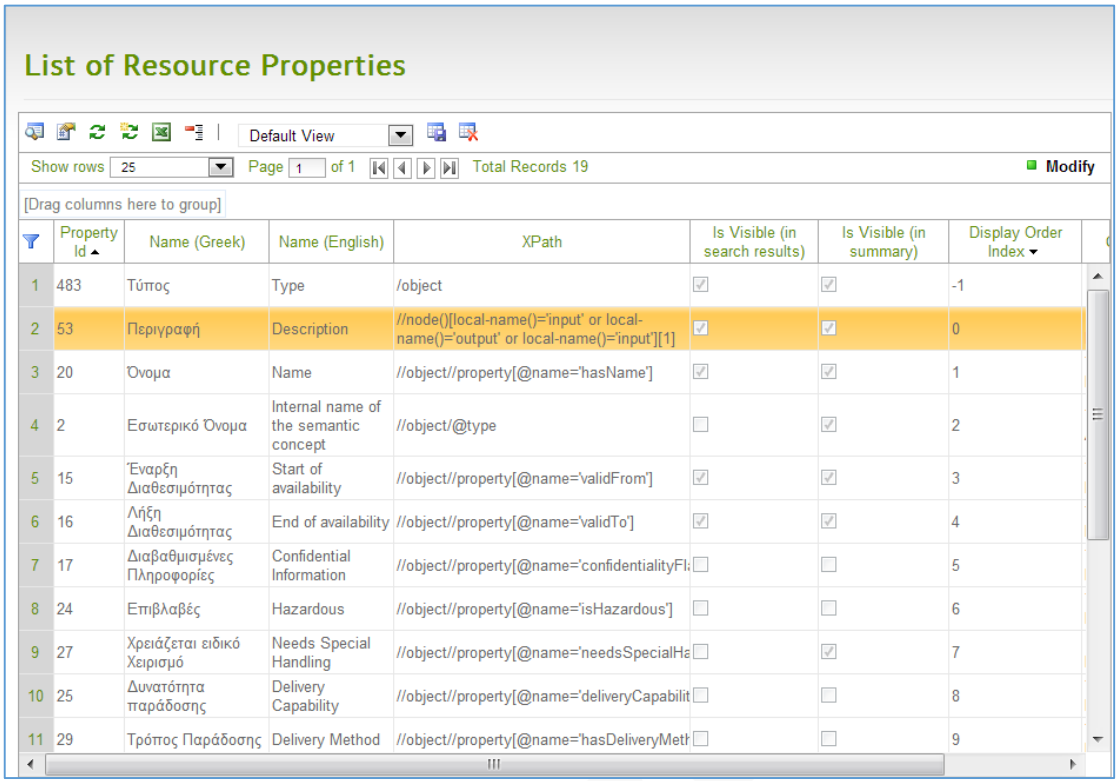


Figure 7: The list of defined resource properties (accessible only by administrator)

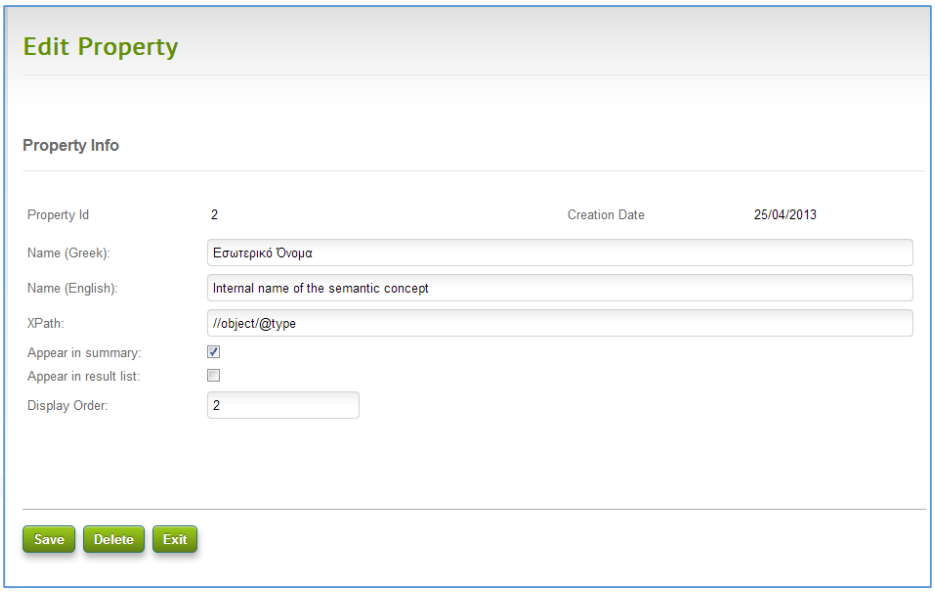


Figure 6: Editing a resource property (accessible only by administrator)

3.2 ADMINISTRATOR REGISTRATION

The registration of Administrators and IS Practitioners is done only by the Administrator, for security purposes. Thus, the CRM component also offers the functionality of registering new IS Practitioners in the system and for assigning the regions for the IS Practitioners. In more detail, these two processes are explained below.

3.2.1 *PRACTITIONER REGISTRATION*

The Administrator registers a new IS Practitioner by completing the respective form, which contains the person details, as well as the credentials for logging into the system. The process that is followed after the Administrator registers someone as a Practitioner is similar (the user is notified by an email, which contains the registration link – see Figure 4). After the new Practitioner user clicks the link, the registration is completed and the user can log into the system with the provided username and password.

3.2.2 ASSIGNMENT OF REGIONS TO PRACTITIONERS

The Administrator is also responsible for assigning regions to IS Practitioners. A region can have multiple Practitioners and one Practitioner can be responsible for multiple regions. The relevant functionalities are provided in separate tabs and are listed below:

- Assignments by Region: is this tab, the Administrator can view a listing of all regions and the assigned Practitioners for each region. From this tab, the Administrator can also unassign a Practitioner from a region.
- Assignments by Practitioner: from this tab, the Administrator views a listing of all Practitioners, with the Regions that are assigned to each one. The Administrator can unassign a region from the selected practitioner and can block/unblock the access of the specified Practitioner to the portal.
- Create New Region: from this tab, the Administrator can register a new region into the system, by specifying its geographical data.

4 SEARCH FACILITY

This is the starting point for many users of the system once they have completed the registration process. It provides a quick way of finding contacts, companies, resource matches and any of a wide range of predefined types of information. Typical searches might include such examples as:

- search for (Matches) in (Central Greece) in the last (month) containing (*text*)
- search for (Contacts) in (all) in the last (month) containing (*text*)
- search for (documents) in (Viotia) in the last (week)

From the search results list, users will be able to select individual items as links to detailed records.

4.1.1 SEARCH FOR RESOURCE MATCHES

An important functionality of the search component is the resource match functionality, which is the first step to a synergy creation.

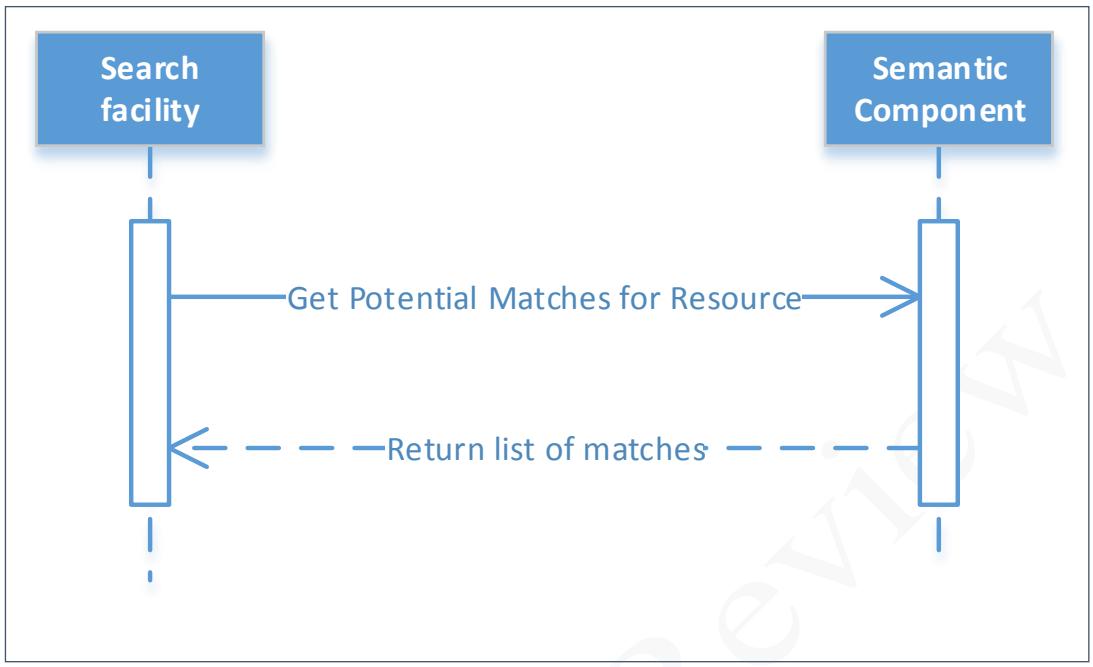


Figure 8: The sequence diagram of searching for compatible resources

There are two ways that potential matches for a selected resource can be identified:

- By semantic relevance of the resources (the Semantic Component calculates and returns a relevance value, i.e. a number that indicates compatibility of the two resources based on reasoning)
- By specific criteria that the user submits

The resource matches are managed by the Semantic component and more precisely by the dedicated Matches Identification module that directly communicates with the Reasoning Engine. From a technical perspective, the Search Facility component receives the potential matches' list from the Semantic Component via a web service. As analysed in the relevant section (see Section CRM), the returned resources have a dynamic list of properties, provided from the semantic component in xml format. The user can select one of the returned matches in order to initiate a new synergy, if the user considers this matched resource as compatible. This user action starts the synergy lifecycle, which is managed from this step on by the dedicated component (see Section **Error! Reference source not found.**).

5 TECHNICAL ARCHITECTURE AND DESIGN

In this section, we analyze in depth the technical architecture of the eSymbiosis platform, in terms of architectural design, design patterns used for the materialization of the platform, and technologies on which the implementation was based.

5.1 IMPLEMENTATION STRATEGY

The eSymbiosis platform follows a three-tier architecture:

- the first tier is the client workstation that accesses the application from a web browser;
- the second tier is the web server and can be further decomposed into 4 layers:

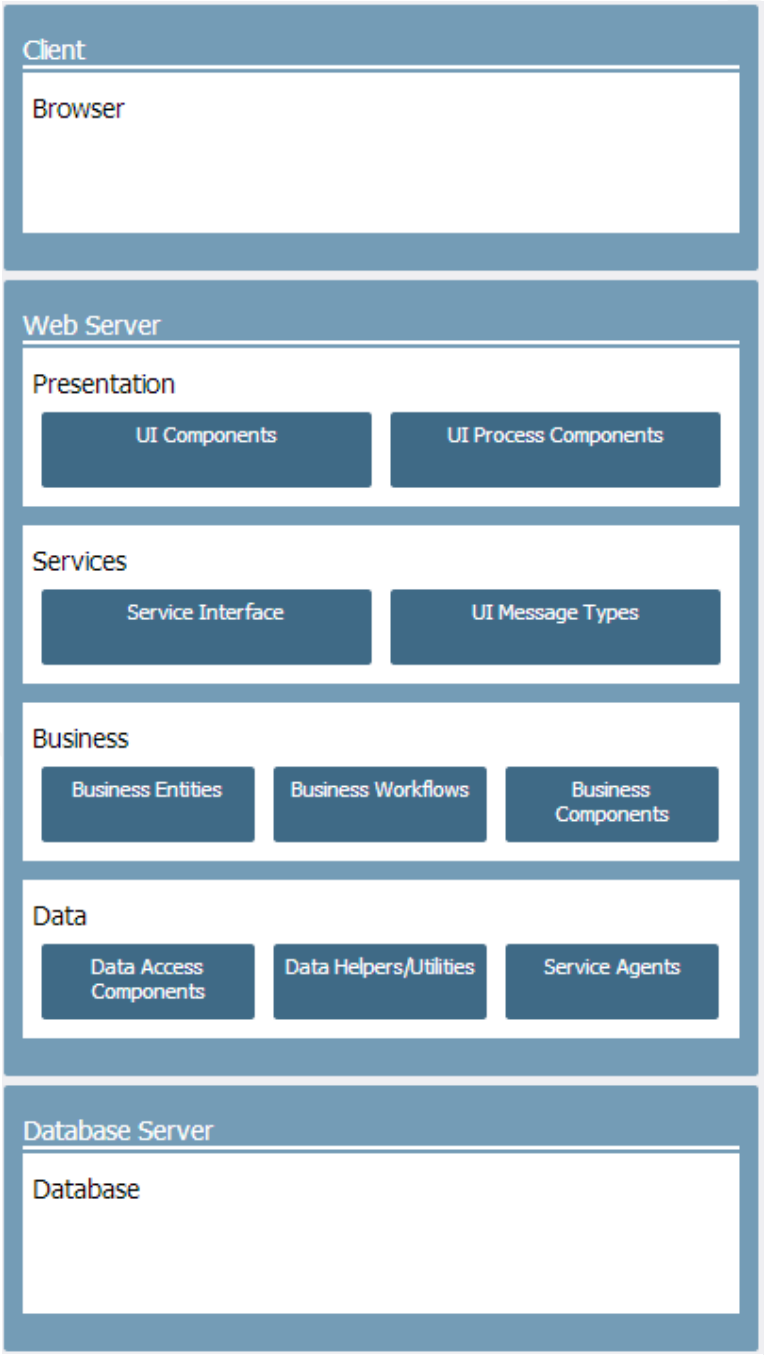


Figure 9: Technical Architecture of the eSymbiosis platform

Presentation (for accepting user input and managing the user interface content that is sent to the client application), services (implements the interfaces and messages that expose the core processes of the eSymbiosis platform to the external components), business (implements the core functionality of the system, and encapsulate the relevant business logic), and data layer (provides access to data that is hosted within the boundaries of the system, and data exposed by other back-end systems);

- the third and final tier is the database server, which directly writes/reads the eSymbiosis data entries stored in the corresponding database.

5.2 ARCHITECTURAL PATTERNS

The implementation strategy uses a number of architectural patterns, as the following diagram displays.

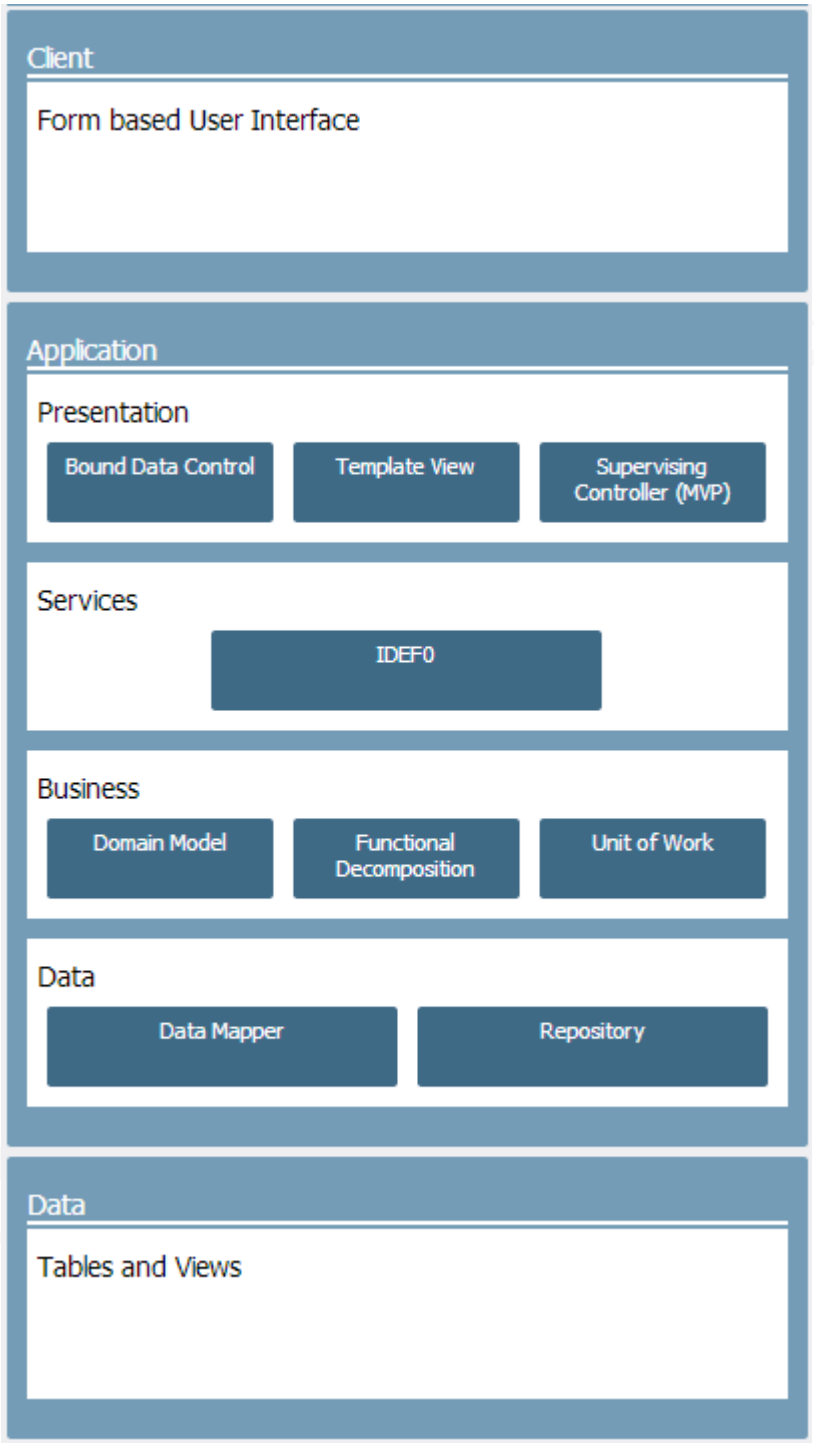


Figure 10: The design patterns that are used

For each of the three tiers presented in the previous Section (see Implementation Strategy), the respective Architectural Patterns are selected to define a best-practice

approach that will guide the actual implementation. A summary of the important points that the Architectural Pattern diagram demonstrates is the following:

- User Interface processing is handled by a Supervising Controller pattern.
- The Template View pattern is used to define a common look and feel.
- Controls are bound to objects that contain data.
- The business layer uses a Facade pattern to implement a message-based interface between the presentation and business layer.
- The Domain model pattern is used to model the application domain.
- The Unit of Work pattern is used to keep track of everything you do during a business transaction that can affect the database. When you're done, it figures out everything that needs to be done to alter the database as a result of your work.
- A Repository pattern is used to access the Data Mapper entities.
- A Data Mapper pattern is used to map domain entities to the database schema and make the Domain Model persistence ignorant.

These patterns are described in more detail in the following sections, where they are categorized according to the tier and the layer they refer to.

5.2.1 *WEB SERVER – PRESENTATION LAYER*

Supervising Controller	
User Interface processing is divided into three separate roles.	The three roles are Model, View, and Presenter. The Model represents data, the View represents the user interface, and the Presenter is responsible for processing requests.
The web page handles requests and passes them off to a controller	Requests are sent to the View (web page), which then passes control to a provider that is responsible for initializing the Model, returning control back to the View, or passing control on to a different View.
M-V-P do not depend directly on each other (Dependency inversion)	M-V-P do not depend directly on each other. Instead, they depend on interfaces (e.g. IView, IPresenter)
Template View	
An ASP.NET Master page is used to provide a common look and feel.	Common elements such as background, page layout, menus, header, and footer are defined in the master page.
ASP.NET pages focus on content that is specific to each page	Each page is associated with the master page, which renders the common content. As a result, the page only needs to contain user interface elements that are not common across all pages.
Bound Data Control	
ASP.NET Server and User controls are bound to business entities returned from the business layer.	Business entities returned from the business layer can be bound to web controls, which will use data from the entity when rendering the display.

5.2.2 *WEB SERVER – BUSINESS LAYER*

Domain Model	
The domain model is comprised of POCOs (Plain Old CLR Objects)	The Domain Model consists of POCOs that are related and fully describe the application domain. POCOs are "…ordinary classes where you focus on the business problem at hand without adding stuff for infrastructure-related reasons. ... The classes should focus on the business problem at hand. Nothing else should be in the classes in the Domain Model." (Nilsson, 2006)
POCOs also contain domain logic.	The POCOs also have methods that implement the business logic.
The POCOs are completely Persistence Ignorant (PI)	This allows us to : • Design the Domain Model independently from the Database Model. • Design, build, and test any business logic relatively independently of the database and the persistence infrastructure code.
Unit Of Work	
The Unit of Work keeps track of changes during a business operation, and saves the changes to the database.	When you're pulling data in and out of a database, it's important to keep track of what you've change. Similarly you have to insert new objects you create and remove any objects you delete. You can change the database with each change to your object model, but this can lead to lots of very small database calls, which ends up being very slow. Furthermore it requires you to have a transaction open for the whole interaction, which is impractical if you have a business transaction that spans multiple requests. The situation is even worse if you need to keep track of the objects you've read so you can avoid inconsistent reads. A Unit of Work keeps track of everything you do during a business transaction that can affect the database. When you're done, it figures out everything that needs to be done to alter the database as a result of your work.
Used for business operations that need to be executed as a single unit.	Within the Unit of work operation all changes a tracked, one or more business operations are performed, and the changes are only propagated to the database depending on the outcome of the business operations.
Can also be used to manage context information.	This pattern can be used to implement a single point of entry for each request where context can be initialized and used throughout the request processing.
Functional Decomposition	
Business Processes are implemented using the Functional Decomposition pattern. Each process is broken down to sub-processes and modeled using IDEF0 models.	A function is achieved by a sequence of sub-functions. A combination of a super-function and its sub-functions is called a functional decomposition pattern. Its definition consists of a super-function, sub-functions, functional relations among sub-functions, and behavioral conditions. These functions are described in terms of the functional concepts. For example, a super-function “heat object” has two sub-functions; “generate heat” and “give heat”. There should be a proportional-type functional relation among them. The behavioral condition is that the objects receiving the heat are identical. In general, a function has some functional decomposition patterns to achieve it. A super-function is decomposed into sub-functions by specifying something related to the ways to achieve it. (In the task context of the functional hierarchy understanding, the reverse operation of the functional decomposition, the information is lost.) According to what is specified, we can categorize the functional decomposition patterns as follows (the notation of the examples in the list is that super-function → sub-function + sub-function2, if any):

5.2.3 *WEB SERVER – DATA ACCESS LAYER*

Repository	
A repository provides an in-memory representation of domain entities.	In systems with a complex Domain Model, it can be worthwhile to build another layer of abstraction over the mapping layer where query construction code is concentrated. This becomes more important when there are a large number of domain classes or heavy querying. In these cases particularly, adding this layer helps minimize duplicate query logic. A Repository encapsulates the set of objects persisted in a data store and the operations performed over them, providing a more object-oriented view of the persistence layer.
A repository provides clean separation between layers.	Repository also supports the objective of achieving a clean separation and one-way dependency between the domain and data mapping layers.
Criteria objects can be used by the repository to generate database queries.	In cases where you have multiple complex queries to retrieve an entity, a criteria object can be used to define selection criteria that are passed into a single get operation.
Data Mapper	
Impedance mismatch between the Domain Model and relational data require mapping layer.	There are several factors that cause a mismatch between object and relational database structures. As a result, a mapping layer can be used to map object structures to database schemas. This allows developers to perform operations against the objects without having to know the database schema.

5.3 TECHNOLOGIES

In this Section we present how the aforementioned Architectural Patterns are replaced with technologies, interfaces, or objects, on the actual implementation level.

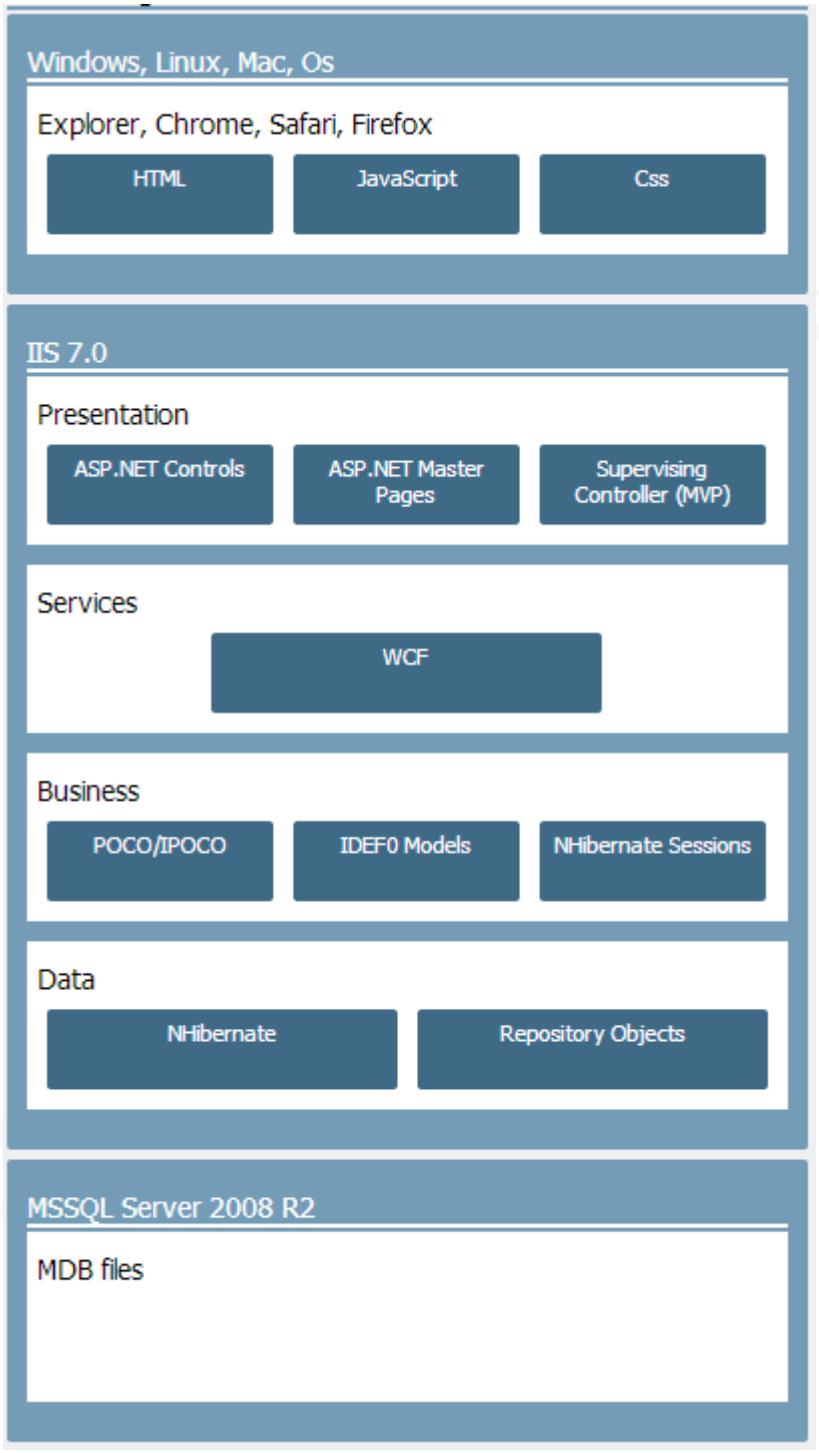


Figure 11: The technologies used for each tier of the eSymbiosis platform

The following is a summary of the technologies, interfaces or objects shown above:

- Any web browser can be used for accessing the eSymbiosis platform (IE, Firefox, Chrome, Safari).
- MVP libraries are used to implement the Model-View-Presenter pattern.
- ASP.NET Master Pages are used to define a Template View.
- ASP.NET Page, Server, and User controls are used to define the user interface.
- Ajax technologies are used to provide a richer user experience.
- POCOs are used to implement the Domain Model.
- Repository objects are used to provide a data access interface.
- NHibernate is responsible for mapping the Domain Model to the database schema.
- Mapping xml files are used to provide Domain Model's mapping information to NHibernate.
- The database used for this implementation is the Microsoft SQL Server 2208 R2.

5.3.1 *WEB SERVER – PRESENTATION LAYER*

Technology Description	**Example**
MVP	

Events in the view, ASP.NET Page, are passed onto the presenter, which is responsible for processing actions from the view.	`protected override void OnInit(EventArgs e)`
	`{`
	`  base.OnInit(e);`
	`  presenter = new CustomerPresenter(this, new CustomerController());`
	`}`
	
	`protected override void OnLoad(EventArgs e)`
	`{`
	`  base.OnLoad(e);`
	`  action = GetFormArgument("action");`
	`  requestedEditCustomerCustomerID = Request.QueryString["ID"];`
	`  if(!IsPostBack)`
	`  {`
	`  presenter.OnViewInitialized();`
	`  }`
	`  presenter.OnViewLoaded();`
	`}`

The presenter class is responsible for interacting with a controller, which then interacts with the business layer. Operations in the presenter are associated with events from the view to perform different actions.	`public void OnViewLoaded()`
	`{`
	`  //Add handlers to View Events`
	`  view.ControlChanged += view_ControlChanged;`
	`  view.cmdCustomerSaveClicked += view_cmdCustomerSaveClicked;`
	`  view.cmdDeleteCustomerClicked += view_cmdDeleteCustomerClicked;`
	`  view.CustomerTypes_DataBound += view_CustomerTypes_DataBound;`
	`  view.CustomerTypes_PreRender += view_CustomerTypes_PreRender;`
	`  PerformAction(view.RequestedAction);`
	`}`

```
private void PerformAction(string action)
{
    try
    {
        string actionresponse;
        LogEntry log = null;

        switch (action)
        {
            case "EditCustomer":

                if (!view.UserRoles.Contains("Administrator"))
                {
                    Apply security and log action
                }

                view.SetPageTitle("RES_PAGETITLE_EditCustomer");
                actionresponse
                    = controller.EditCustomer(view.RequestedEditCustomerCustomerID,
                                              view.UserRoles.ToArray(),
                                              view.UserHostAddress,
                                              view.UserName);

                PopulateView(controller.Customer);
```

```
public void EditCustomer(object RequestedCustomerID,
                         string[] Roles,
                         string UserHostAddress,
                         string UserName)
{
    try
    {
        Apply Security policy and log action

        Repository repository = new Repository();
        Customer = repository.GetCustomer(RequestedCustomerID.ToString());
    }
    catch(Exception)
    {
        // Rethrow
        throw;
    }
}
```

ASP.NET Master Page	
An ASP.NET Master Page is used to define a common layout.	A common use of the Master Page is to define the layout for all pages of the application. This approach provides a consistent look and feel across the web site.
Menus are defined in the Master Page.	Instead of using User controls for common menus they are defined directly in the Master Page.
ASP.NET Page, User and Server Controls	
ASP.NET Page controls are used to define each page of the web application.	The ASP.NET page control is used to render an HTTP page that will be sent back to clients.
ASP.NET User and Server controls are used to provide the interface	ASP.NET user and server controls are used to generate HTML interface elements, such as INPUT, that are used to provide a user interface into the application.

Ajax technologies are used for a richer user experience.	ASP.NET's Update panel Control is used to provide the user with a richer UI experience.

5.3.2 WEB SERVER – BUSINESS LAYER

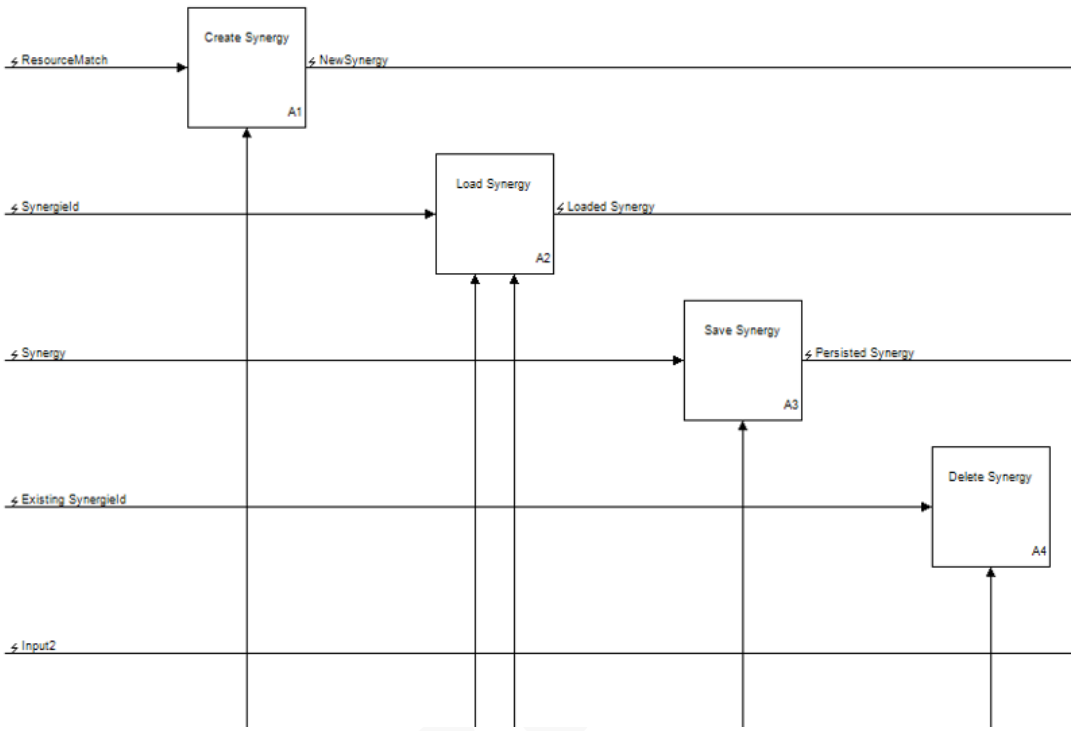
Technology Description	Example
Domain Model	
The Domain Model is implemented as POCOs which are completely persistence ignorant.	

```
public class Region : INotifyPropertyChanged, ICloneable
{
    #region Region's Fields
    [DataMember(Name="RegionID")]
    private int regionID;

    [DataMember(Name="RegionDescription")]
    private string regionDescription;
    #endregion

    #region Region's Properties
    public virtual int RegionID
    {
        get
        {
            return regionID;
        }
        set
        {
            regionID = value;
            OnPropertyChanged("RegionID");
        }
    }

    public virtual string RegionDescription
    {
        get
        {
            return regionDescription;
        }
        set
        {
            regionDescription = value;
            OnPropertyChanged("RegionDescription");
        }
    }
    #endregion

    Region's Participant Properties

    Constructors

    INotifyPropertyChanged Members

    Implementation of ICloneable

} //Class: Region
```

POCOs also contain domain logic.

```
public  class InventoryTransaction : INotifyPropertyChanged, ICloneable
{
    InventoryTransaction's Fields

    InventoryTransaction's Properties

    InventoryTransaction's Participant Properties

    Constructors

    INotifyPropertyChanged Members

    #region Business Logic
    private decimal UpdateDebit(decimal value)
    {
        if (this.inventory == null) return value;
        this.inventory.Balance = this.inventory.Balance - this.debit;
        this.inventory.Balance = this.inventory.Balance + value;
        return value;
    }

    private decimal UpdateCredit(decimal value)
    {
        if (this.inventory == null) return value;
        this.inventory.Balance = this.inventory.Balance + this.credit;
        this.inventory.Balance = this.inventory.Balance - value;
        return value;
    }
    #endregion

    Implementation of ICloneable
}
```

IDEF0 Models

Multiple business operations that need to be included in a transaction are combined in a single operation exposed by a business process object that is defined in an IDEF0 model that implements the Functional Decomposition pattern.	

The Workflow engine then executes the operations in a single transaction	`Dictionary<string, object> inputs = new Dictionary<string, object>();`
	`RequestParameters reqparams`
	`= new RequestParameters`
	`{`
	`Action = RequestedAction.ExecuteProcess,`
	`ServiceId = "Customer.SaveCustomer",`
	`Roles = Roles,`
	`RequestSourceIp = UserHostAddress,`
	`User = UserName`
	`};`
	`inputs.Add("Customer", Customer);`
	
	`Response response`
	`= CLMS.WorkflowEngine.WorkflowEngine.AcceptRequest(reqparams, inputs);`
	
	`Customer=(BO.Customer)response.Outputs["PersistedCustomer"];`

Unit Of Work	

Business operations that need to be included in a transaction are handled by the Unit of Work pattern.	`public void SaveProductCategory(NWS.BO.ProductCategory productcategory)`
	`{`
	`  if(productcategory == null)`
	`  {`
	`  throw new ArgumentNullException("productcategory",`
	`  "No ProductCategory was specified.");`
	`  }`
	`  using (TransactionScope scope`
	`  = new TransactionScope(TransactionScopeOption.Required))`
	`  {`
	`  UnitOfWork.CurrentSession.SaveOrUpdate(productcategory);`
	`  UnitOfWork.Current.Flush();`
	`  scope.Complete();`
	`  }`
	`}`

Unit of Work can be used to initialize a data context that spans across multiple requests.	This approach allows you to maintain context with one object while processing a long business operation. This ensures that all updates to business entities are saved when the operation is completed. The context is the NHibernate session object and Unit Of Work takes care of holding that context until the long operation is completed.
	`if (UnitOfWork.IsStarted && UnitOfWork.InLongConversation)`
	`{`
	`  UnitOfWork.EndLongConversation();`
	`  UnitOfWork.CurrentSession.Clear();`
	`}`
	`UnitOfWork.StartLongConversation();`

5.3.3 *WEB SERVER – DATA ACCESS LAYER*

<table>
<tr><th>Technology Description</th><th>Example</th></tr>
<tr><td colspan="2">Repository Object</td></tr>
<tr><td>The repository contains methods for each parent business entity in the Domain Model</td><td>Parent objects represent the top level object in a in a group of objects that are related.

<pre>
public class Repository : IRepository
{
    #region Implementation of IRepository

    #region ProductCategory Methods
    public NWS.BO.ProductCategory GetProductCategory(int CategoryID)...
    public void SaveProductCategory(NWS.BO.ProductCategory productcategory)...
    public void UpdateProductCategory(NWS.BO.ProductCategory productcategory)...
    public void DeleteProductCategory(NWS.BO.ProductCategory productcategory)...
    #endregion

    Supplier Methods

    Shipper Methods

    CustomerType Methods
</pre>
</td></tr>
<tr><td>For objects that can be retrieved using multiple complex queries generic methods are supplied by the repository which accept criteria expressed in Linq.</td><td>
<pre>
public List<T> Get<T>(Expression<Func<T, bool>> predicate)...

public List<T> Get<T>(Expression<Func<T, bool>> predicate,
                        int startRowIndex,
                        int pageSize,
                        Dictionary<Expression<Func<T, IComparable>>, bool> orderBy,
                        out int totalRecords)...

public List<T> GetAll<T>()...

public List<T> GetAll<T>(int startRowIndex, int pageSize, out int totalRecords)...
</pre>
</td></tr>
<tr><td colspan="2">NHibernate Mapping Files</td></tr>
</table>

XML files provide mapping information of the Domain model POCOs to NHibernate.	<hibernate-mapping xmlns="urn:nhibernate-mapping-2.2" schema="CLMSApps_NWSample_Alpha_Northwind.dbo" default-lazy="true" auto-import="false" assembly="NWS.BO" namespace="NWS.BO"> <class name="Region" table="`Region`"> <!-- Key --> <id name="RegionID" column="RegionID" type="Int32"> <generator class="assigned" /> </id> <!-- Properties --> <property name="RegionDescription" column="RegionDescription" type="String" /> <!-- Relationships --> <bag name="territories" access="field" inverse="true" cascade="all-delete-orphan" > <key column="RegionID" foreign-key="Region_To_Territories"/> <one-to-many class="Territory" /> </bag> </class> </hibernate-mapping>

5.4 DATABASE SCHEMA

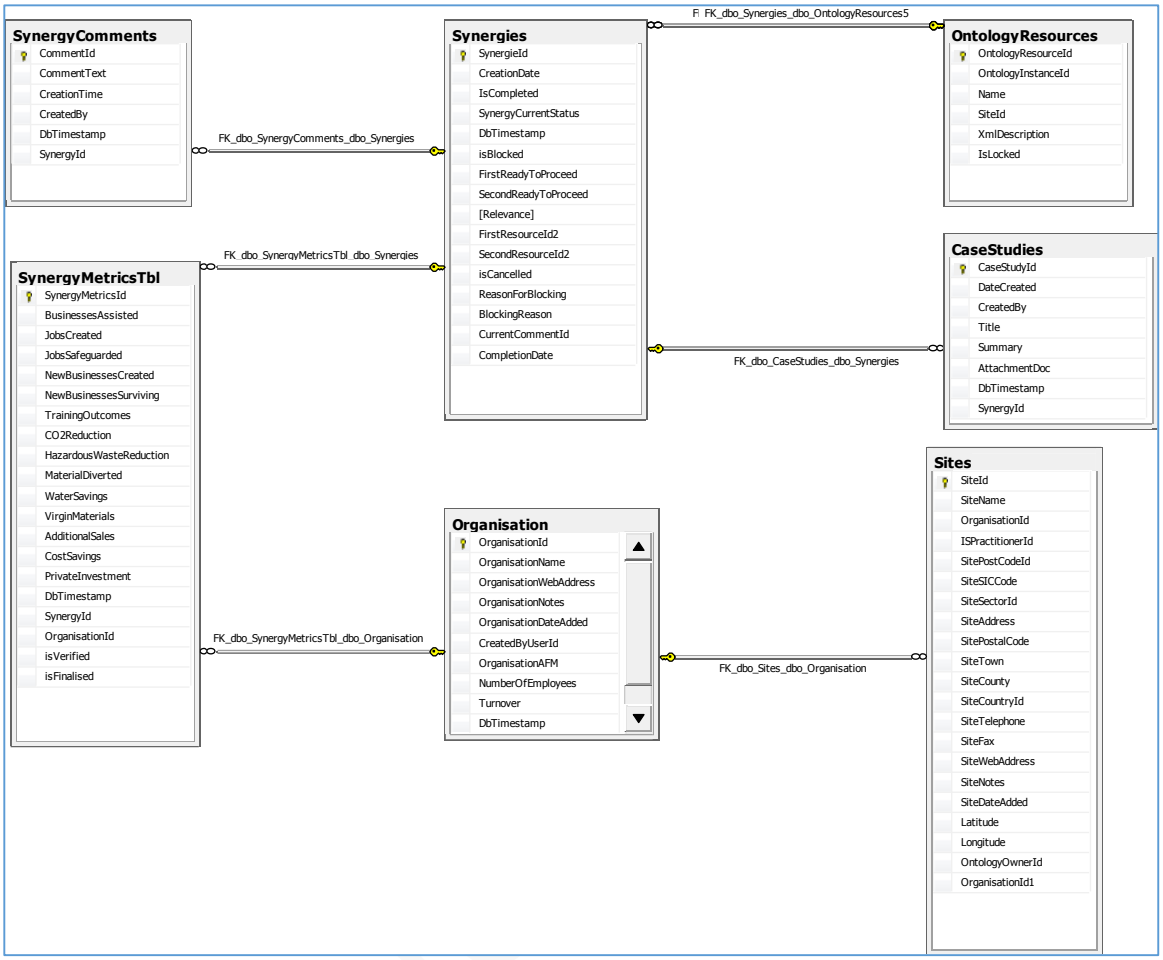


Figure 12: Database schema of main eSymbiosis tables

The main tables of the database tier for the eSymbiosis application are interconnected as shown in the relevant diagram, above.

Synergy

	Name	Data Type	Max Length (Bytes)	Allow Nulls	Identity
	SynergieId	int	4	☐	1 - 1
	CreationDate	datetime	8	☐	
	IsCompleted	bit	1	☐	
	SynergyCurrentStatus	int	4	☐	
	DbTimestamp	timestamp	8	☐	
	isBlocked	bit	1	☐	
	FirstReadyToProceed	bit	1	☐	
	SecondReadyToProceed	bit	1	☐	
	Relevance	decimal(18,8)	9	☐	
	FirstResourceId2	int	4	☐	
	SecondResourceId2	int	4	☐	
	isCancelled	bit	1	☐	
	ReasonForBlocking	nvarchar(500)	1000	☐	
	BlockingReason	int	4	☐	
	CurrentCommentId	int	4	☐	
	CompletionDate	datetime	8	☐	

Organisation

	Name	Data Type	Max Length (Bytes)	Allow Nulls	Identity
	OrganisationId	int	4	☐	1 - 1
	OrganisationName	nvarchar(255)	510	☐	
	OrganisationWebAddress	nvarchar(50)	100	☐	
	OrganisationNotes	nvarchar(max)	max	☐	
	OrganisationDateAdded	datetime	8	☐	
	CreatedByUserId	nvarchar(50)	100	☐	
	OrganisationAFM	int	4	☐	
	NumberOfEmployees	int	4	☐	
	Turnover	decimal(18,2)	9	☐	
	DbTimestamp	timestamp	8	☐	

Synergy Metrics

	Name	Data Type	Max Length (Bytes)	Allow Nulls	Identity
	SynergyMetricsId	int	4	☐	1 - 1
	BusinessesAssisted	int	4	☐	
	JobsCreated	int	4	☐	
	JobsSafeguarded	int	4	☐	
	NewBusinessesCreated	int	4	☐	
	NewBusinessesSurviving	int	4	☐	
	TrainingOutcomes	int	4	☐	
	CO2Reduction	decimal(8,2)	5	☐	
	HazardousWasteReduction	decimal(8,2)	5	☐	
	MaterialDiverted	decimal(8,2)	5	☐	
	WaterSavings	decimal(8,2)	5	☐	
	VirginMaterials	decimal(8,2)	5	☐	
	AdditionalSales	decimal(18,2)	9	☐	
	CostSavings	decimal(18,2)	9	☐	
	PrivateInvestment	decimal(18,2)	9	☐	
	DbTimestamp	timestamp	8	☐	
	SynergyId	int	4	☐	
	OrganisationId	int	4	☐	
	isVerified	bit	1	☐	
	isFinalised	bit	1	☐	

Site

	Name	Data Type	Max Length (Bytes)	Allow Nulls	Identity
	SiteId	int	4	☐	1 - 1
	SiteName	nvarchar(255)	510	☐	
	ISPractitionerId	int	4	☐	
	SitePostCodeId	int	4	☐	
	SiteSICCode	nvarchar(50)	100	☐	
	SiteSectorId	int	4	☐	
	SiteAddress	nvarchar(255)	510	☐	
	SitePostalCode	nvarchar(50)	100	☐	
	SiteTown	nvarchar(50)	100	☐	
	SiteCounty	nvarchar(50)	100	☐	
	SiteCountryId	int	4	☐	
	SiteTelephone	nvarchar(50)	100	☐	
	SiteFax	nvarchar(50)	100	☐	
	SiteWebAddress	nvarchar(50)	100	☐	
	SiteNotes	nvarchar(256)	512	☐	
	SiteDateAdded	datetime	8	☐	
	Latitude	decimal(10,6)	9	☐	
	Longitude	decimal(10,6)	9	☐	
	OntologyOwnerId	nvarchar(200)	400	☐	
	OrganisationId1	int	4	☐	

Case Study

	Name	Data Type	Max Length (Bytes)	Allow Nulls	Identity
	CaseStudyId	int	4	☐	1 - 1
	DateCreated	datetime	8	☐	
	CreatedBy	nvarchar(100)	200	☐	
	Title	nvarchar(1000)	2000	☐	
	Summary	nvarchar(max)	max	☐	
	AttachmentDoc	nvarchar(500)	1000	☐	
	DbTimestamp	timestamp	8	☐	
	SynergyId	int	4	☐	

Synergy Comments

	Name	Data Type	Max Length (Bytes)	Allow Nulls	Identity
	CommentId	int	4	☐	1 - 1
	CommentText	nvarchar(1000)	2000	☐	
	CreationTime	datetime	8	☐	
	CreatedBy	nvarchar(100)	200	☐	
	DbTimestamp	timestamp	8	☐	
	SynergyId	int	4	☐	

5.5 DOMAIN MODEL

In this section we present a number of the most important Business Objects of the eSymbiosis platform.

5.5.1 ORGANISATION

The business object *Organisation* represent a member of the eSymbiosis ecosystem that either has resources to offer (“haves”) or requests resources of other participants (“wants”). An organization has multiple member sites, which need or produce the relevant resources. From the usability point of view, the organization is represented by a user, who accesses the relevant functionalities of the platform.

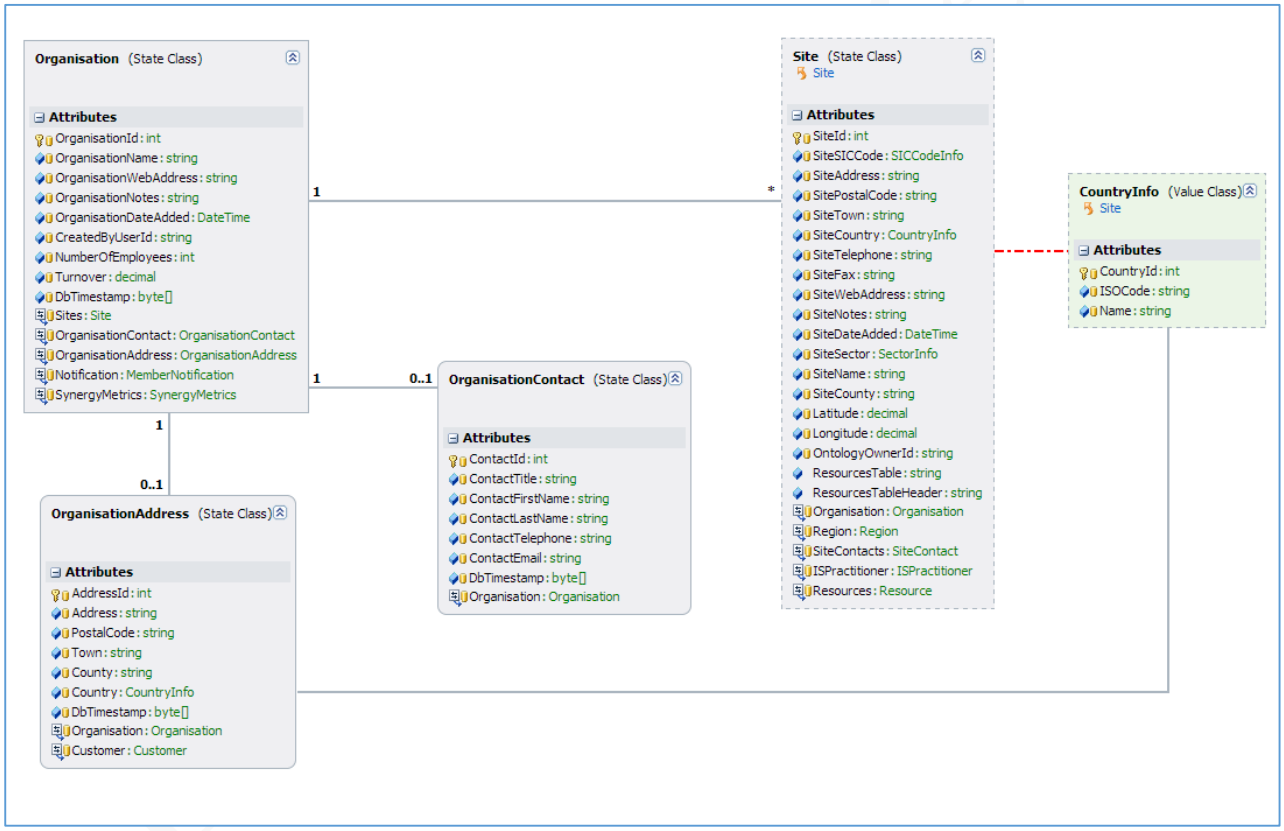


Figure 13: The domain model of the Organisation business object

5.5.2 SITE

The Site business object represent an industrial unit of the eSymbiosis ecosystem, which belongs to one particular Organisation. The site also belongs to a specified region and an IS Practitioner is responsible for facilitating the synergies that the Site is involved into.

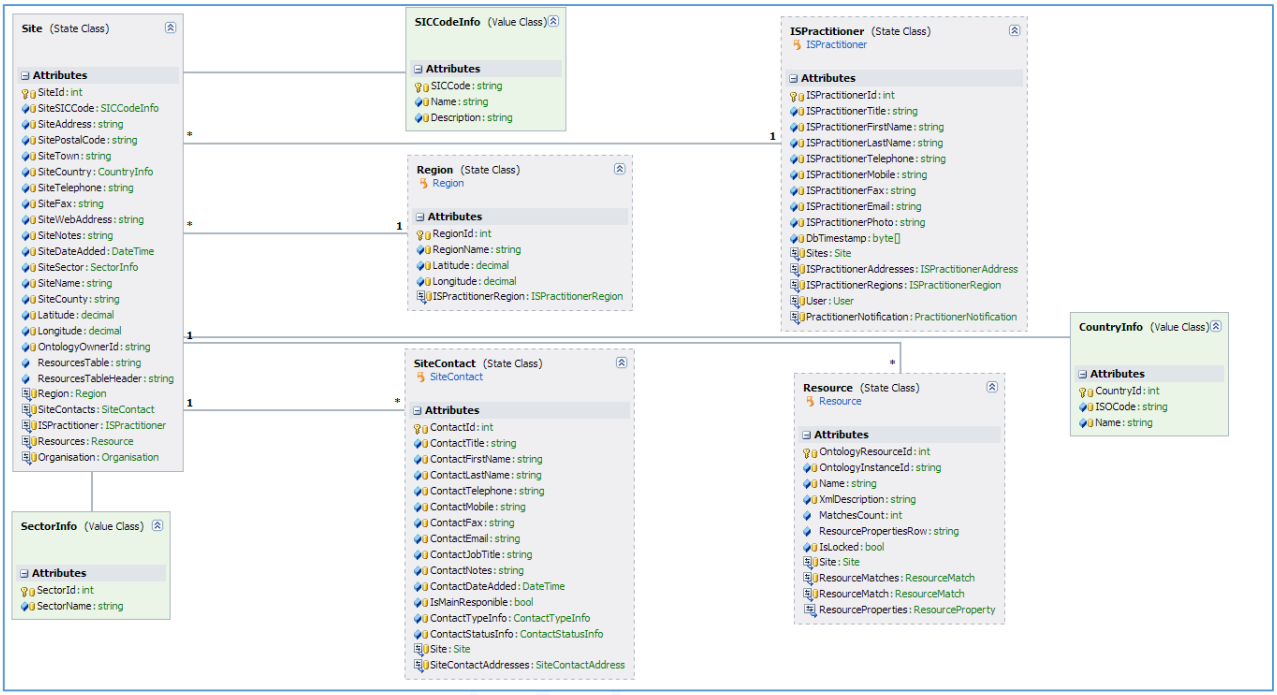


Figure 14: The domain model of the Site business object

5.5.3 SYNERGY

The Synergy business object represents the process of the collaboration between a member site that offers a resource to a member site that needs a resource.

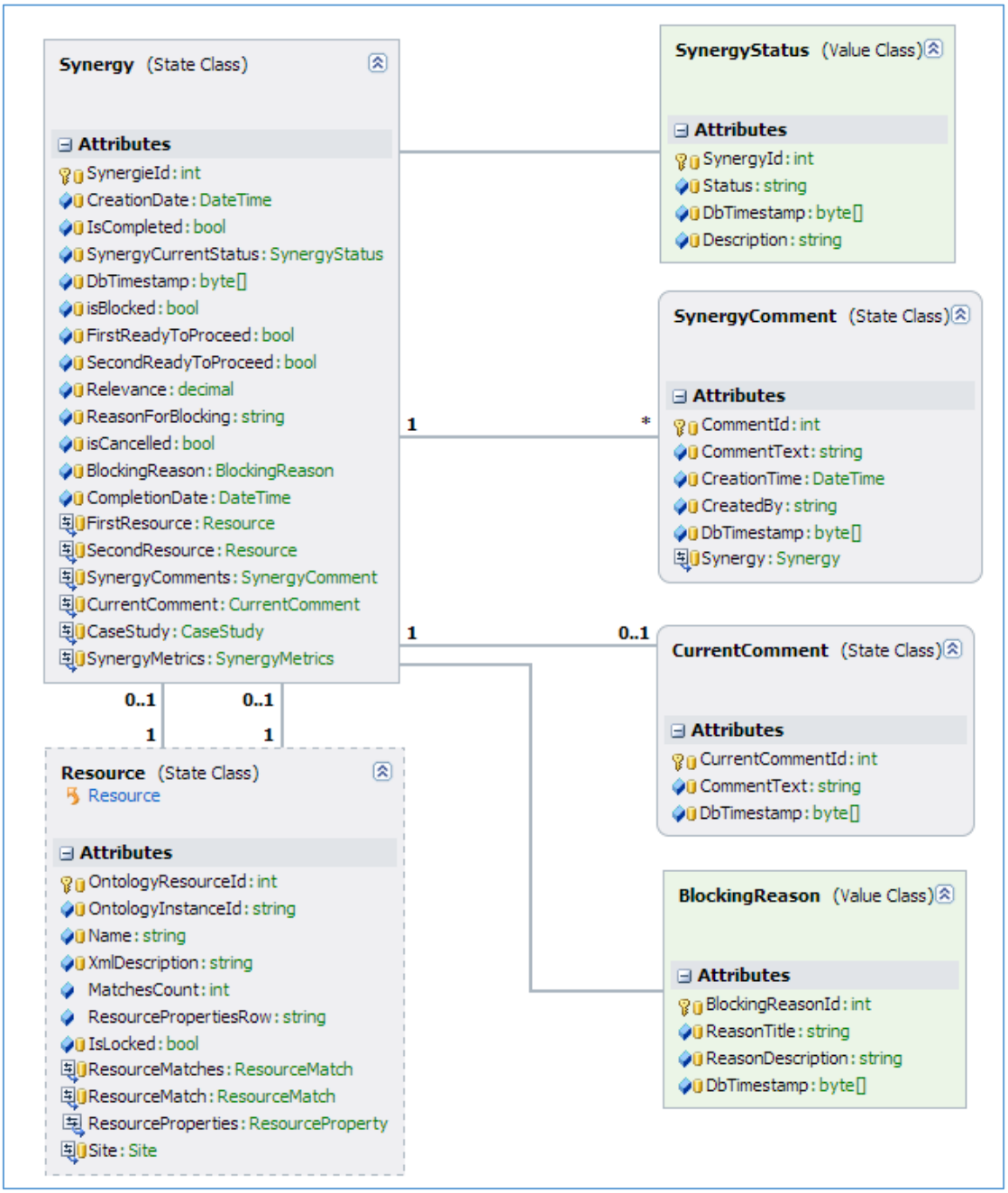


Figure 15: The domain model of the Synergy business object

5.5.4 SYNERGY METRICS

The Synergy Metrics business object represents a set of metrics that are submitted by each synergy participant in relation to the measurable results of the process.

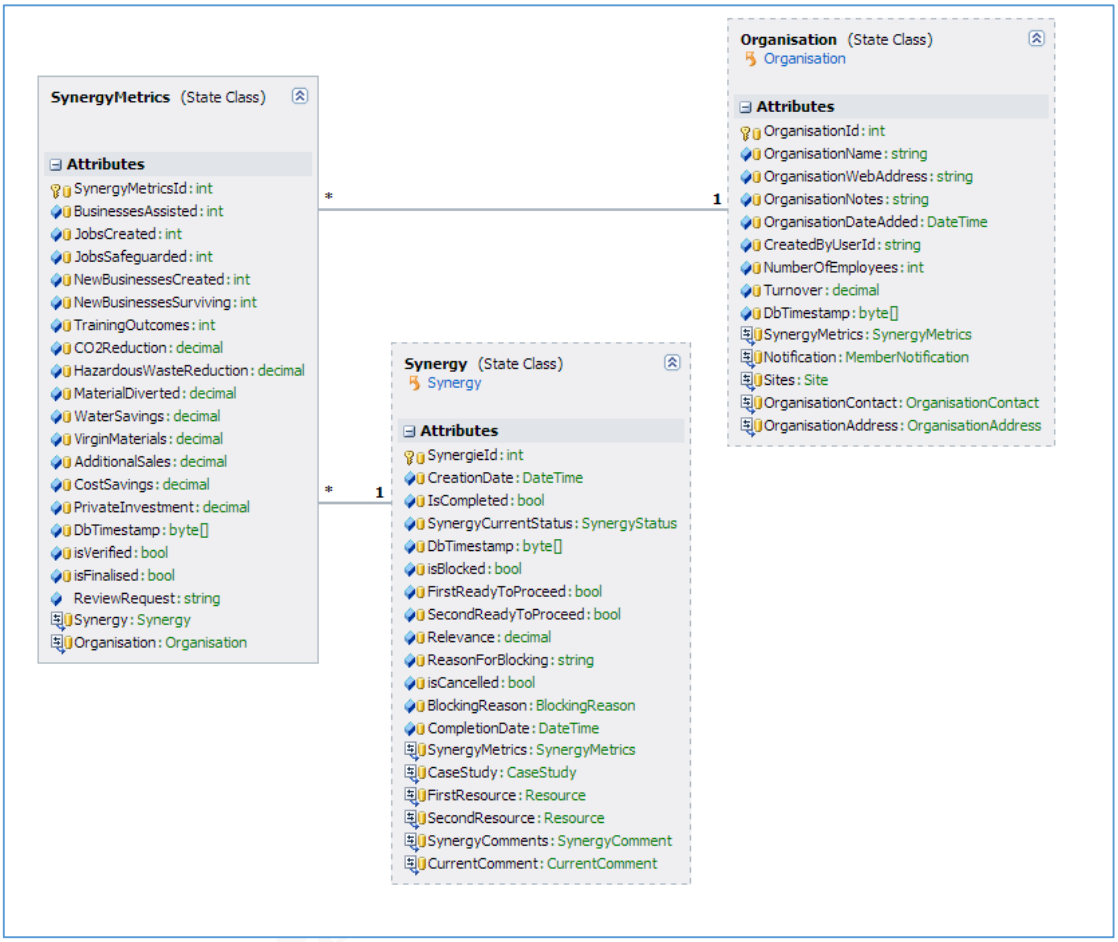


Figure 16: The domain model of the SynergyMetrics business object

5.5.5 CASE STUDY

The Case Study business object represents the summary that is produced after the successful completion of a synergy, from the input of the IS Practitioner, the synergy details, and the related synergy metrics.

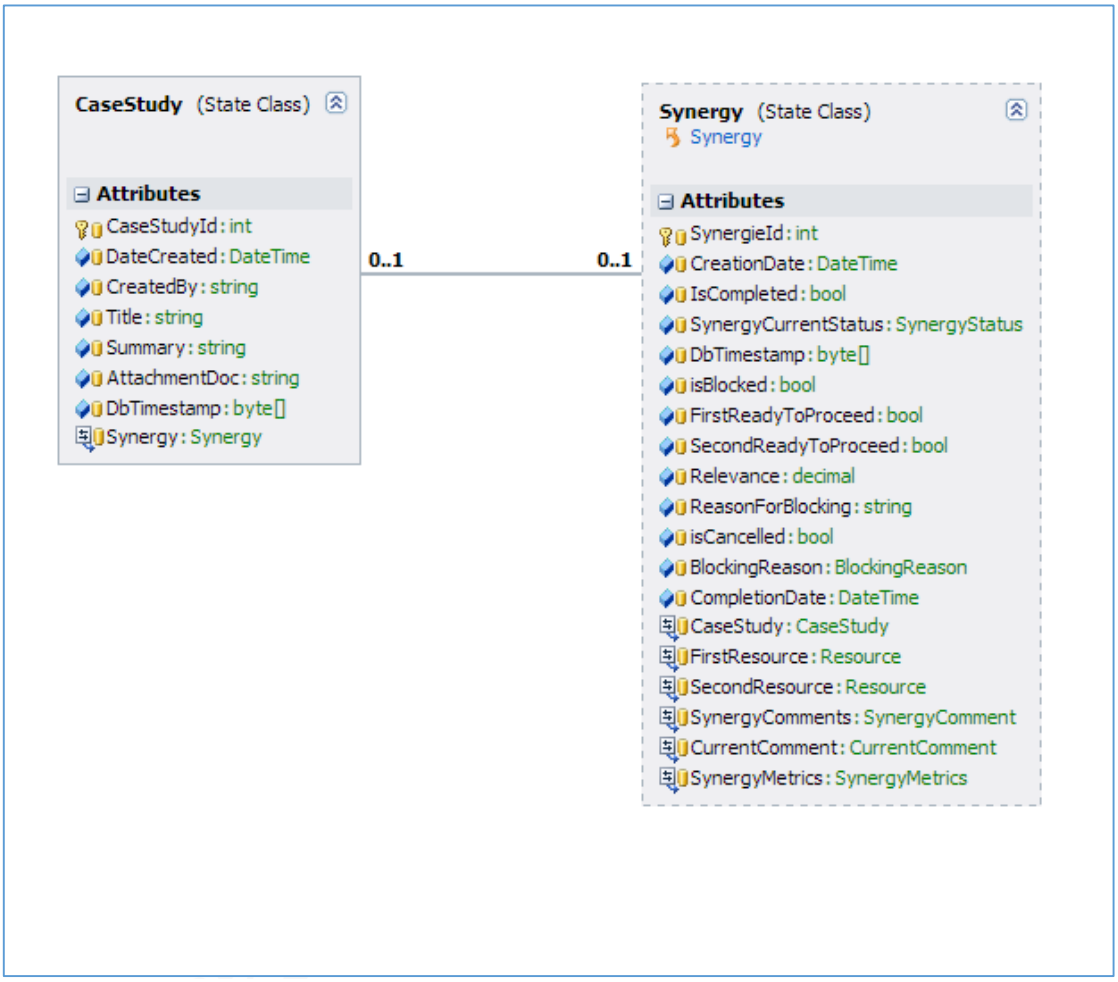


Figure 17: The domain model of the Case Study business object

6 SUMMARY AND CONCLUSIONS

The semantic web service portal offers a wide range of functionalities that allow to the user to become part of the Industrial Symbiosis (IS) ecosystem and take advantage of the existing sources of IS knowledge stored in the portal. The purpose of this deliverable is to introduce these functionalities to the reader and familiarize them with the underlying technical infrastructure of the portal.

In more detail, this document summarized the architecture and the design of the eSymbiosis platform modules that implement the semantic web portal functionalities (Activity 2.4). The relevant components are:

- the Information Portal, which provides and manages access to the various knowledge sources in the portal,
- the CRM, which manages the profiles of the Organisation Member Users, the IS Practitioners, and the Administrators with the relevant managerial functionalities, and
- the Search Facility (which allows the user to search for and navigate to the requested information sources or functionalities).

The corresponding components were examined both from a functional and from a technical perspective.

Moreover, the portal was presented from a technical point of view: from the abstract level of implementation strategy to the specific technologies that were used. The domain model and its data representation were introduced and analysed, in order to cover all aspects of the technical infrastructure that materialize the provided portal functionalities.

7 ΠΕΡΙΛΗΨΗ ΚΑΙ ΣΥΜΠΕΡΑΣΜΑΤΑ

Η σημασιολογική δικτυακή πύλη παρέχει ένα μεγάλο εύρος από λειτουργικότητες, οι οποίες επιτρέπουν στο χρήστη να συμμετάσχει στο οικοσύστημα Βιομηχανικής Συμβίωσης (ΒΣ) και να αξιοποιήσει τις υπάρχουσες σχετικές πηγές γνώσης της πύλης. Σκοπός αυτού του παραδοτέου είναι να παρουσιάσει αυτές τις λειτουργικότητες στον αναγνώστη και να τον εξοικειώσει με την υφιστάμενη τεχνική υποδομή της πύλης.

Πιο αναλυτικά, αυτό το έγγραφο συνοψίζει την αρχιτεκτονική και το σχεδιασμό των λειτουργικών μονάδων της πλατφόρμας eSymbiosis που υλοποιούν τις λειτουργικότητες της σημασιολογικής διαδικτυακής πύλης (Δραστηριότητα 2.4). Οι σχετικές λειτουργικές μονάδες είναι:

- η Πληροφοριακή Πύλη, που παρέχει και διαχειρίζεται την πρόσβαση στις διάφορες πηγές πληροφοριών της πύλης,
- η Διαχείριση Πελατολογίου, που διαχειρίζεται τα προφίλ των χρηστών των συμμετεχόντων Οργανισμών, των Υπευθύνων ΒΣ, και των Διαχειριστών με τις σχετικές λειτουργικότητες διαχείρισης, και
- η Λειτουργία Αναζήτησης (που επιτρέπει στον χρήστη να αναζητά και να προηγείται στις ζητούμενες πηγές πληροφοριών ή λειτουργικότητες.

Οι σχετικές λειτουργικές μονάδες εξετάστηκαν τόσο από λειτουργική όσο και από τεχνική πλευρά.

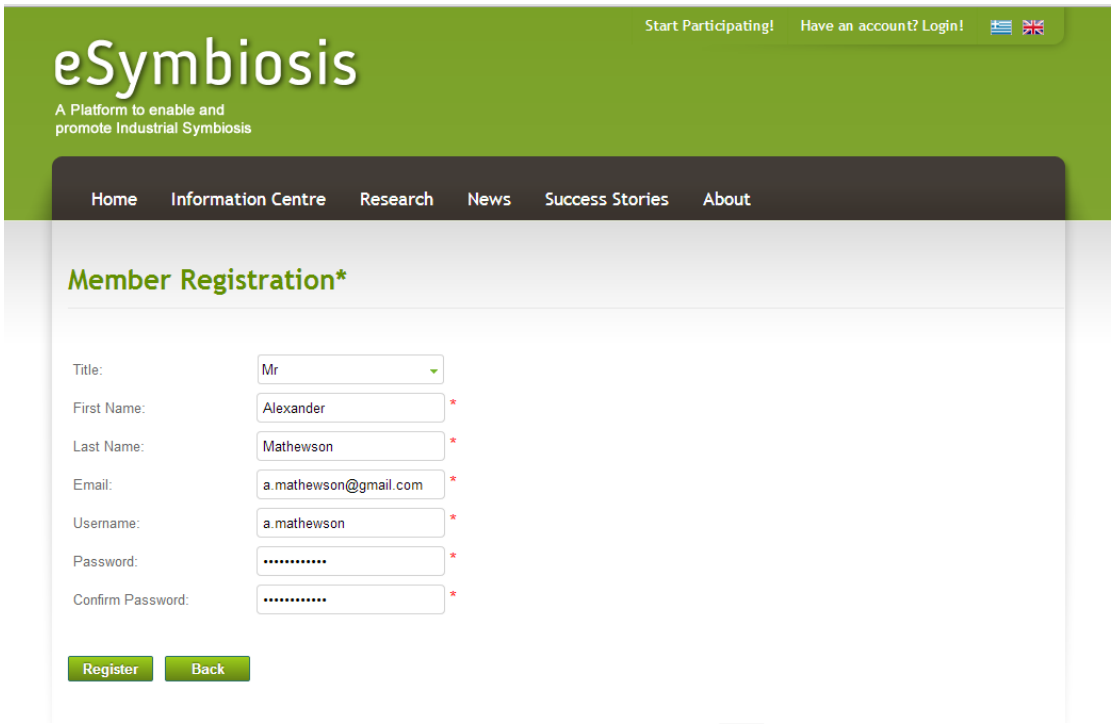
Επιπλέον, η πύλη παρουσιάζεται από την τεχνική σκοπιά: από το γενικότερο επίπεδο της στρατηγικής υλοποίησης ως τις συγκεκριμένες τεχνολογίες που χρησιμοποιήθηκαν. Το μοντέλο των οντοτήτων και η αναπαράσταση του σε επίπεδο δεδομένων παρουσιάζονται και αναλύονται, ώστε να καλυφθούν όλες οι πλευρές της τεχνικής υποδομής, οι οποίες υλοποιούν τις παρεχόμενες λειτουργικότητες της πύλης.

APPENDIX: USER MANUAL

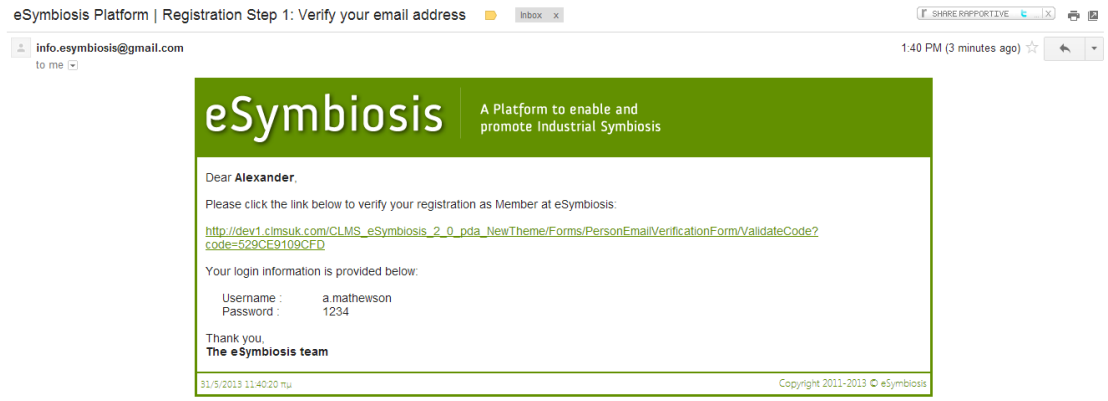
Step 1: Create account for first time user

A user visits the site for the first time and needs to register, so they select “Start Participating!”

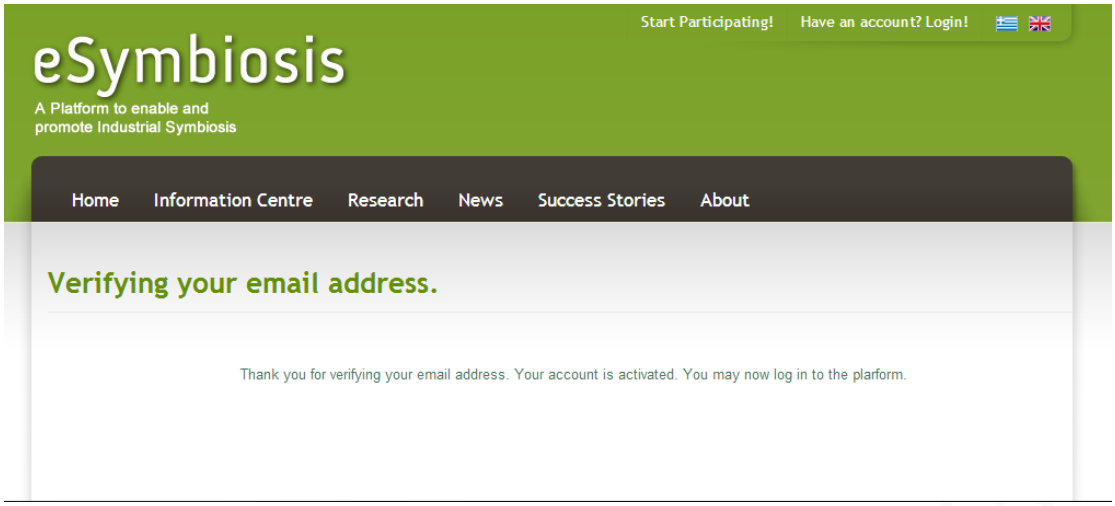
In the screen that follows, the user that represents the Organisation registers his/her user account, by providing his basic information.



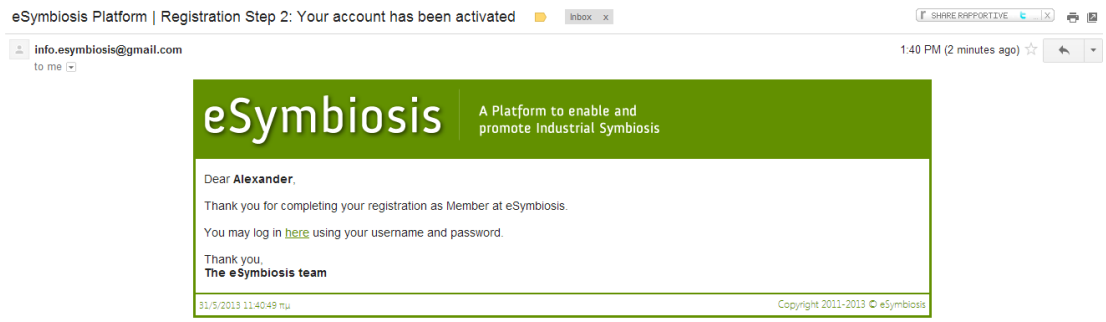
Right after this page, the user receives the following email message with a link to verify the specified email address.



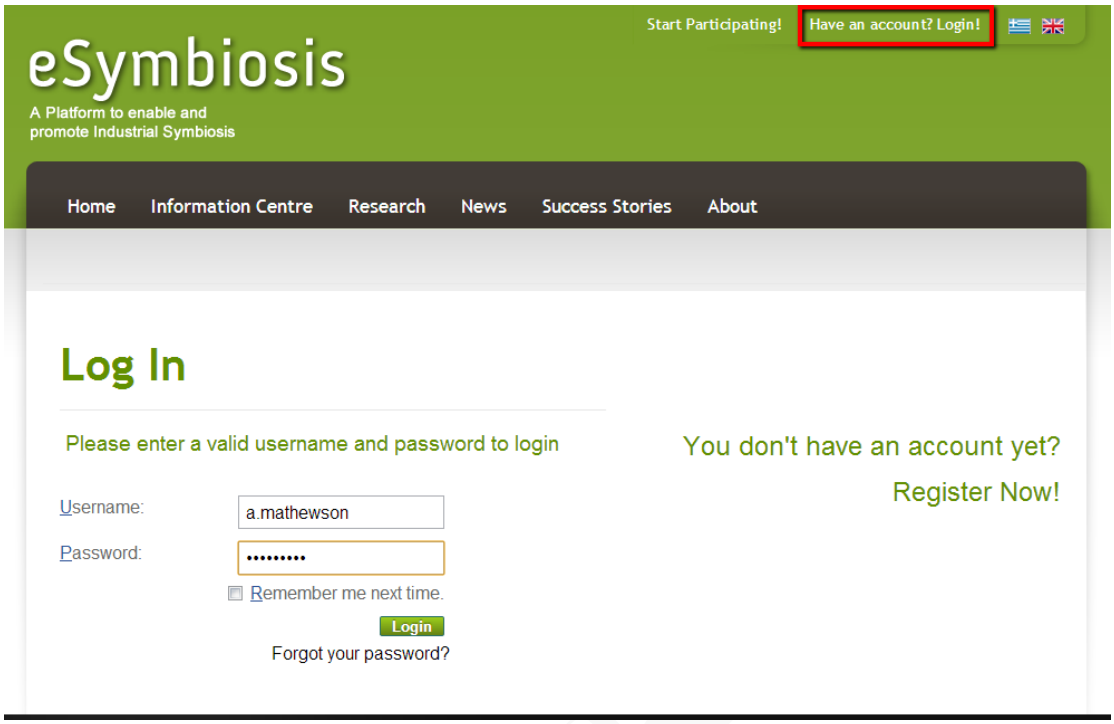
Clicking on the link, will navigate the user to the website, and the email address will be registered as verified.



A second email message is received informing the user regarding the email address verification success and the registration completion.



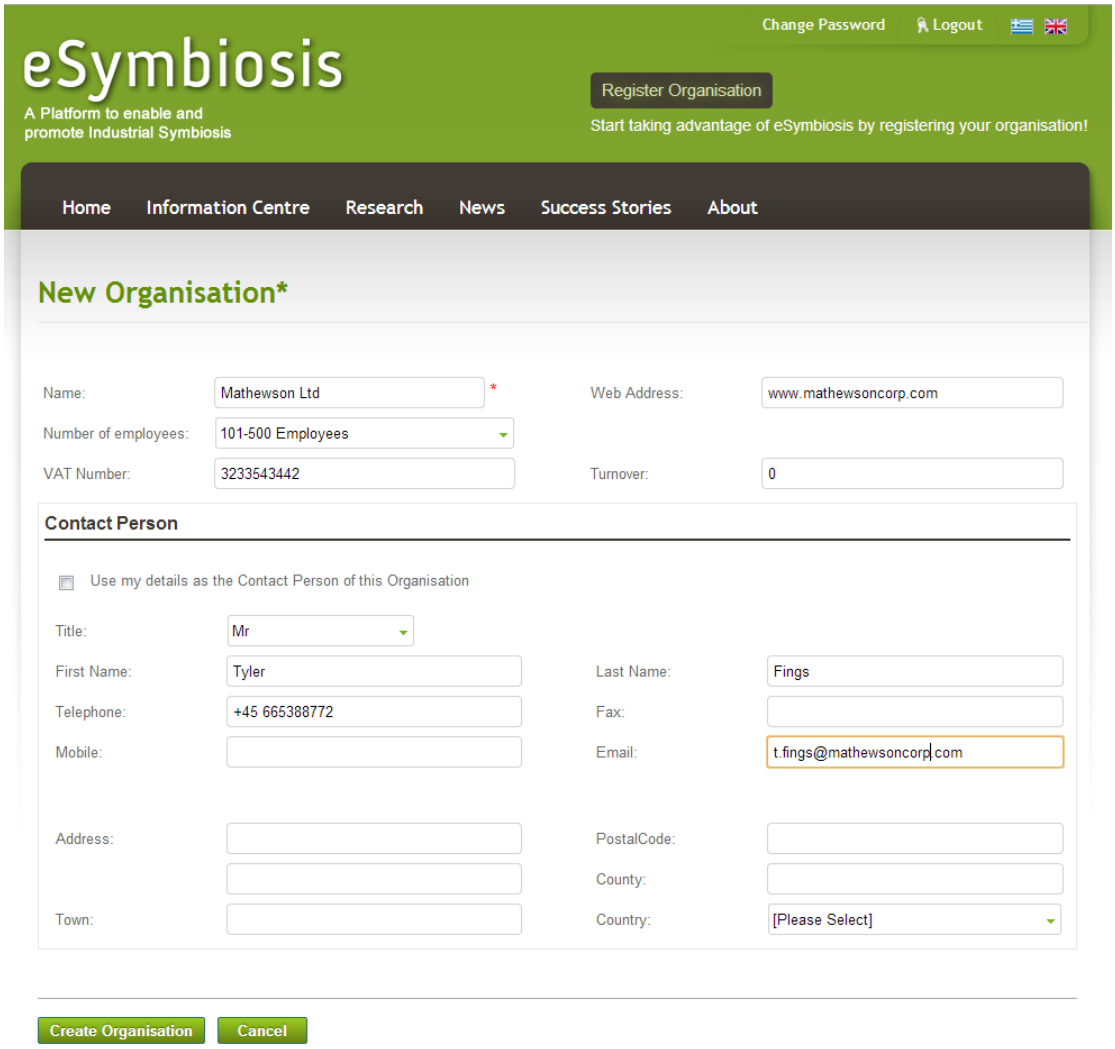
The User may now login with his/her credentials, in order to start using the website to a greater extent.



The User is provided with suggestions based on the current state of completion regarding their profiles.

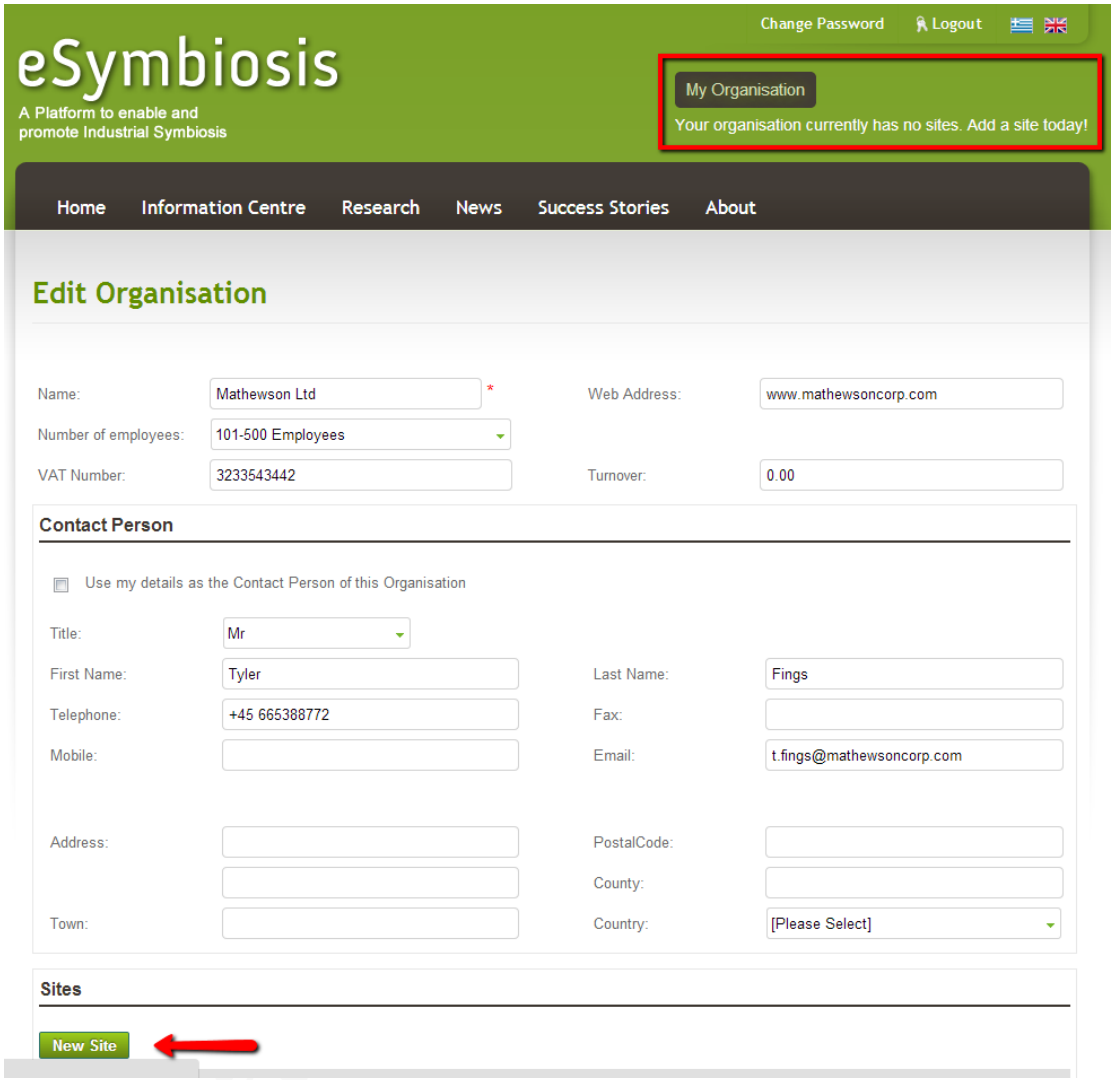
Step 2: Register Organisation

In the following screen, the user has to provide the basic Organisation information.



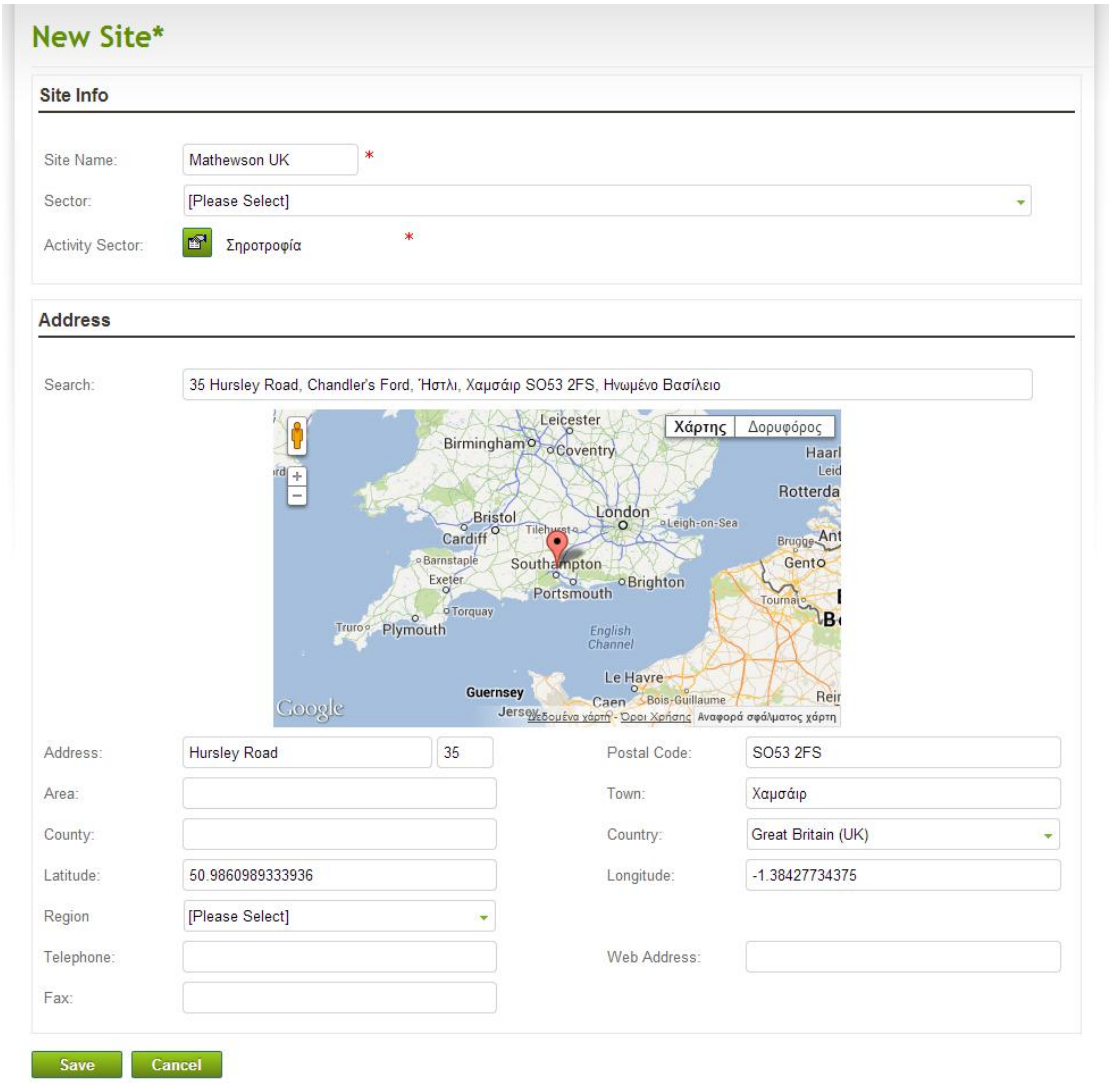
Organisation name, web address, size of the organisation, as well as some contact details.

After registering an organisation, the User is suggested to add sites to the newly-created organisation.

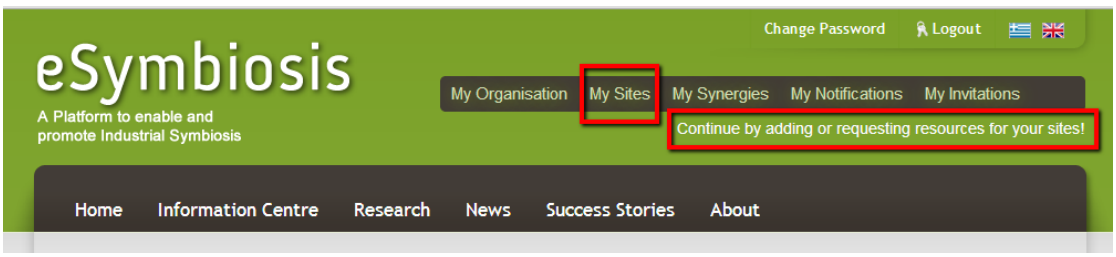


Step 2: Add Sites

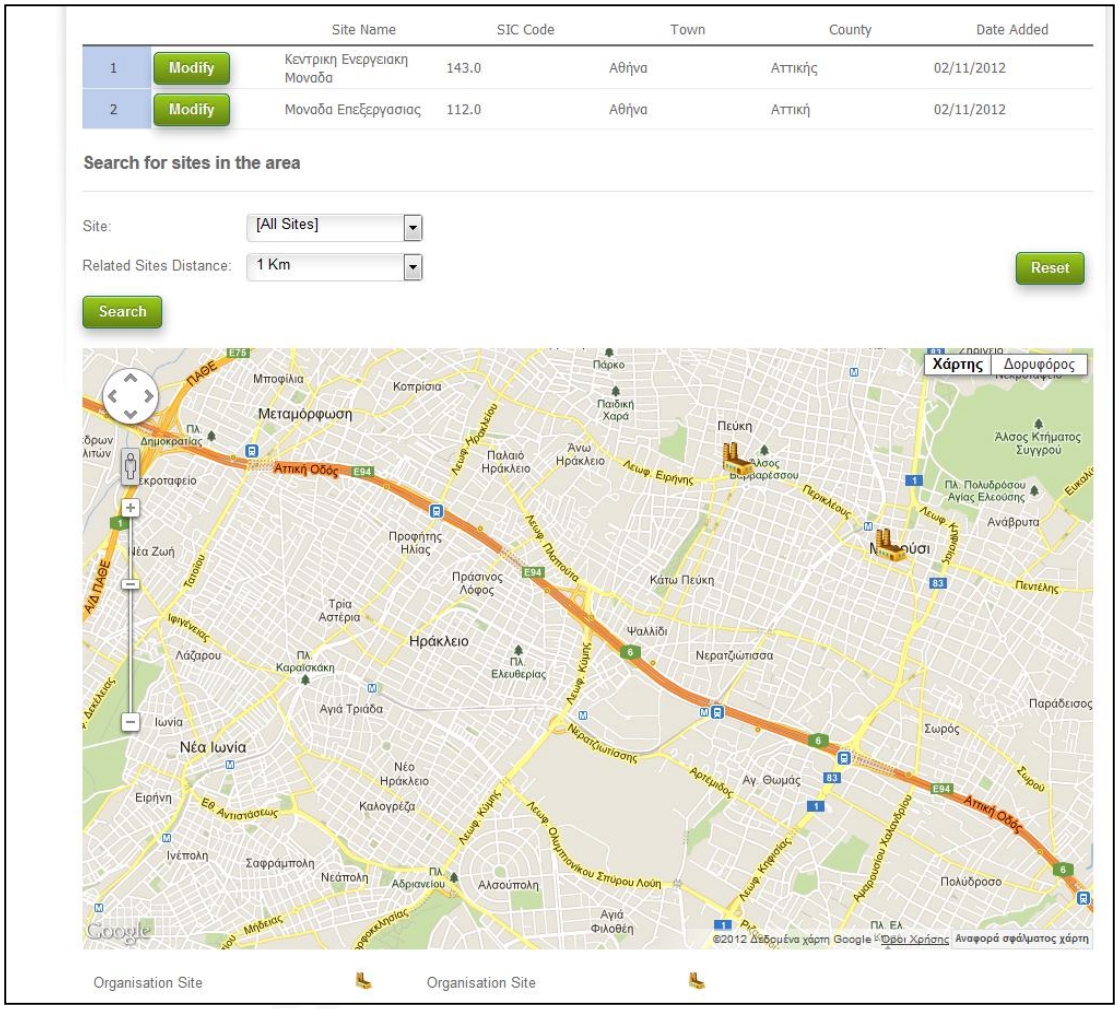
The following screen in used for adding the Sites of the Organisation that participate in the eSymbiosis project. Apart from the Site details, such as name and activity sector, the user provides the geographical information of the site. The corresponding location is shown in the provided map.



Upon saving the Site information, the user can then start registering and/r requesting Resources. The user may also access the Sites' details at any later point by clicking the corresponding menu item *My Sites* at the upper right of the screen.

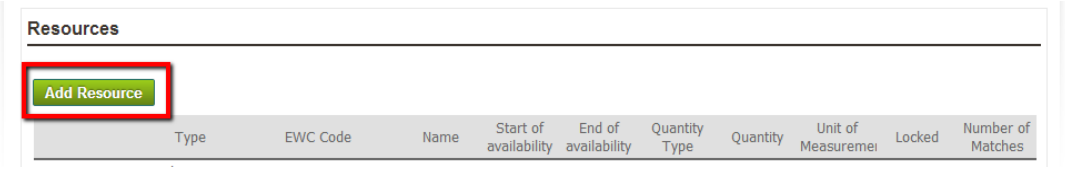


In the “My Sites” page, the user may modify each Site’s details and also view how Sites are placed geographically, on a map.



Step 3: Add Further Site Details

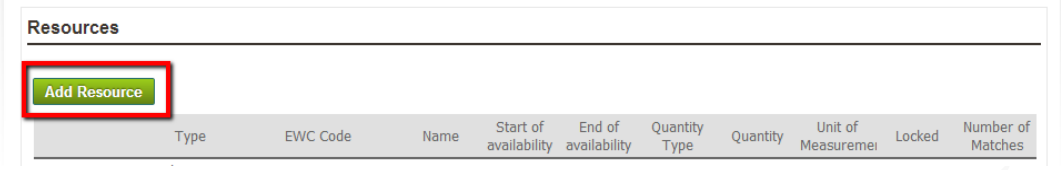
For the selected Site, the user can submit the Resources of the Site. For the Resources that this Site needs/provides, the Organisation user can select the corresponding option as shown below.



This action initiates the Resource Registration process for the selected Site, as explained in the following section. And upon saving the changes, the resource is shown in the “Edit Site” page as in the following form

Step 4: Add resource

Once the user has registered a site, a resource can be added using the *Add Resource* button at the bottom of the site details page, as mentioned previously.



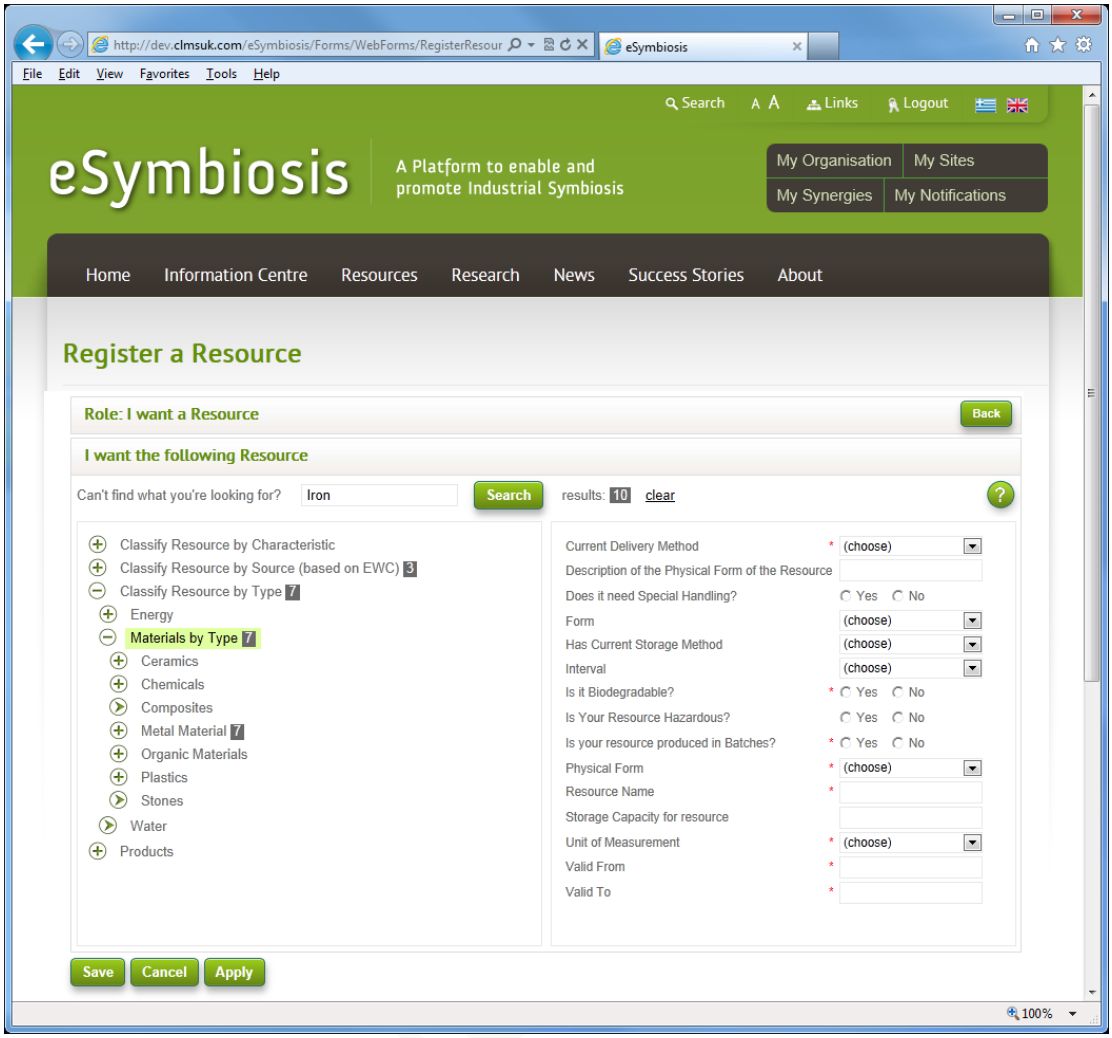
This button then takes the user to the resource registration page which allows them to follow a sequence of prompts that are designed to first classify and then add details to the resource they would like to register.

This page originally has multiple sections, as the Organisation user can specify if the Organisation's site is:

1. Offering a resource
2. Having a Technology, or
3. Wanting a Resource

Selecting each option will change the right-hand pane, which allows to the user to specify the Resource details for the selected need.

By selecting the *Back* button the user will return to this view.



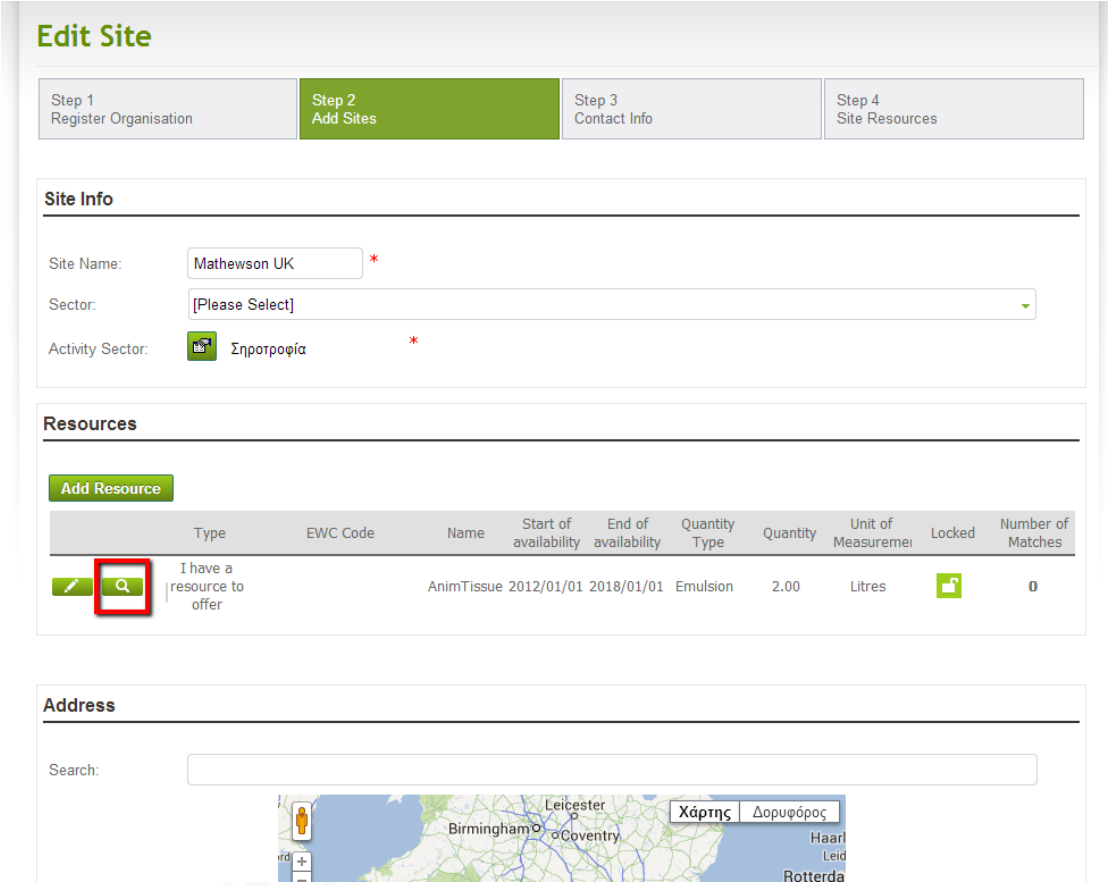
Under this is the current section, which allows the user to identify the resource that is closest to theirs using the list to the left. The user has the option of searching this list using the search box above – in this example the word “Iron” shows 10 entries and the highlighted numbers show where these can be located.

To the right of the classification list can be seen the properties that have been associated with the chosen item, where red asterisks represent those fields that are required to be filled in before the registration can be approved.

Finally the user can choose to store their responses using the “Apply” button, discard unsaved responses using the “Cancel” button, or both save and return to the site overview page using the “Save” button.

Search for Resource Matches

From the Site's screen, where the Resources of the Site are displayed, the user can initiate a search for potential synergy matches with other available Resources. For this action, the user has to click on the Search button, in the Resource details, as shown below.

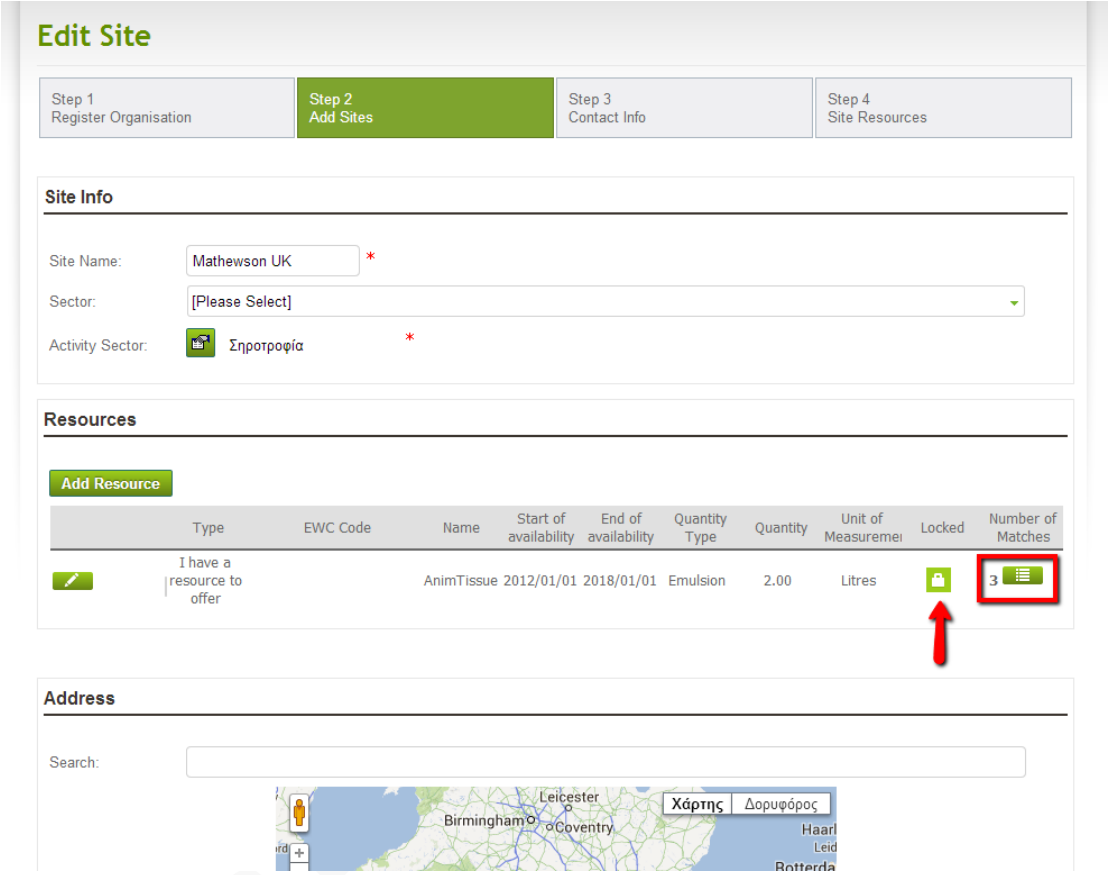


When this action is selected, the Resource is locked until the completion of the search process and the mechanism retrieves matching Resources for potential Synergies.

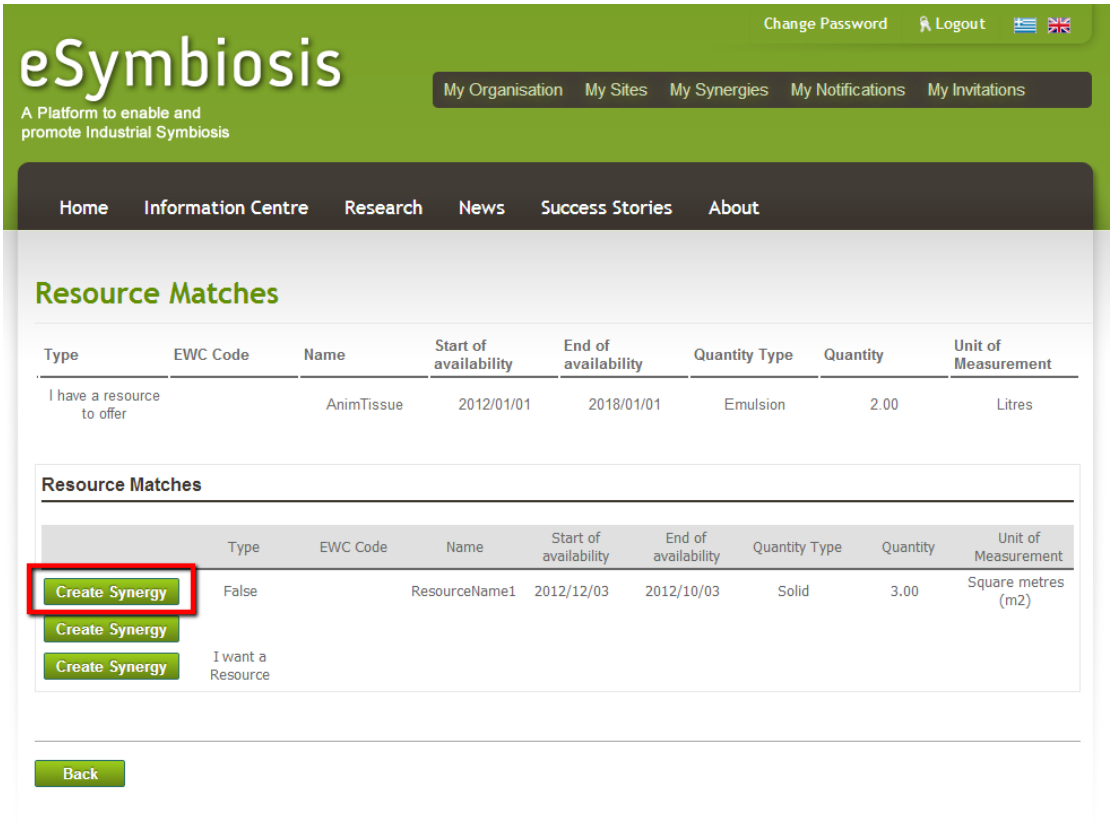
Resource Matches

For a selected Site, the user can see the retrieved matches for each of the Site's available resources, by clicking the View List of Matches button from each row of the Resources table. Also, note that the resource is now locked

For example, the following screen shows that for each of the Site resources, there are multiple potential resource matches.



By clicking *View*, the user sees the details for the retrieved matches of the selected Site resource. If a resource match seems compatible, the Organisation user can select the *Create Synergy action* with the specific resource match.

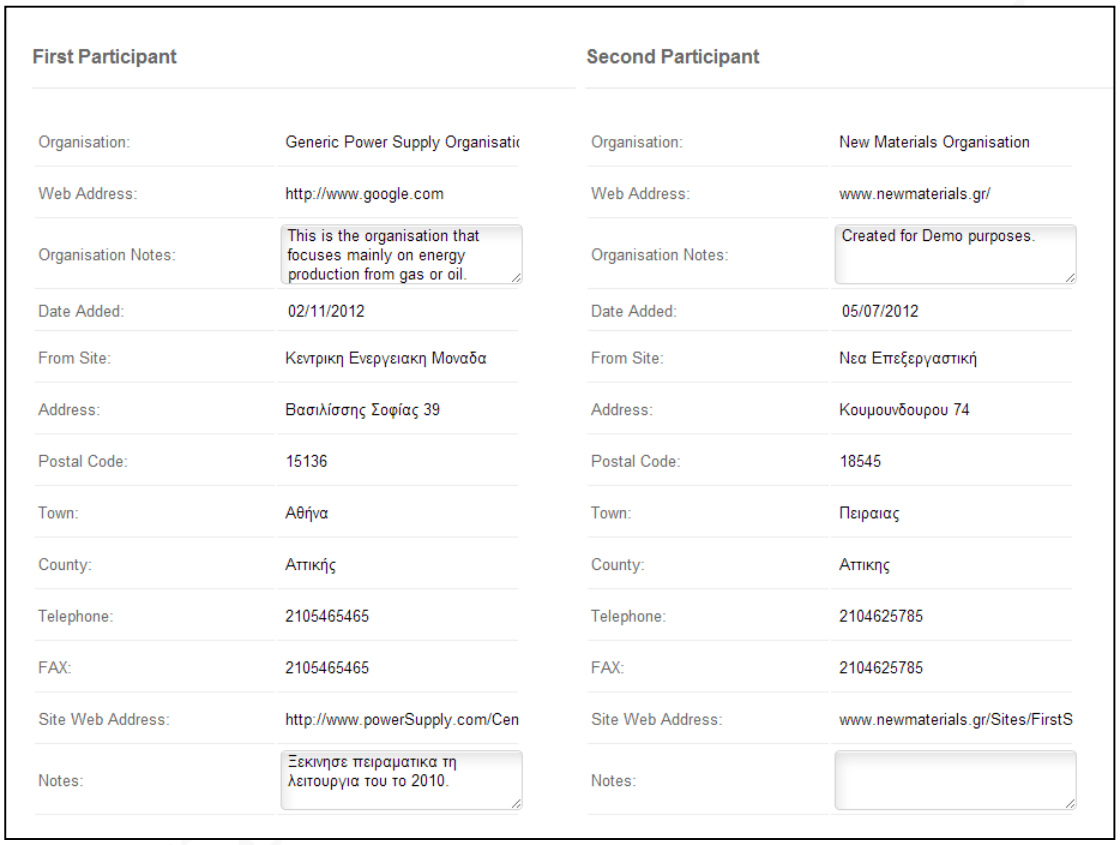


This action creates a new Synergy and the corresponding Synergy form is shown to the user, as explained in more detail in the following section.

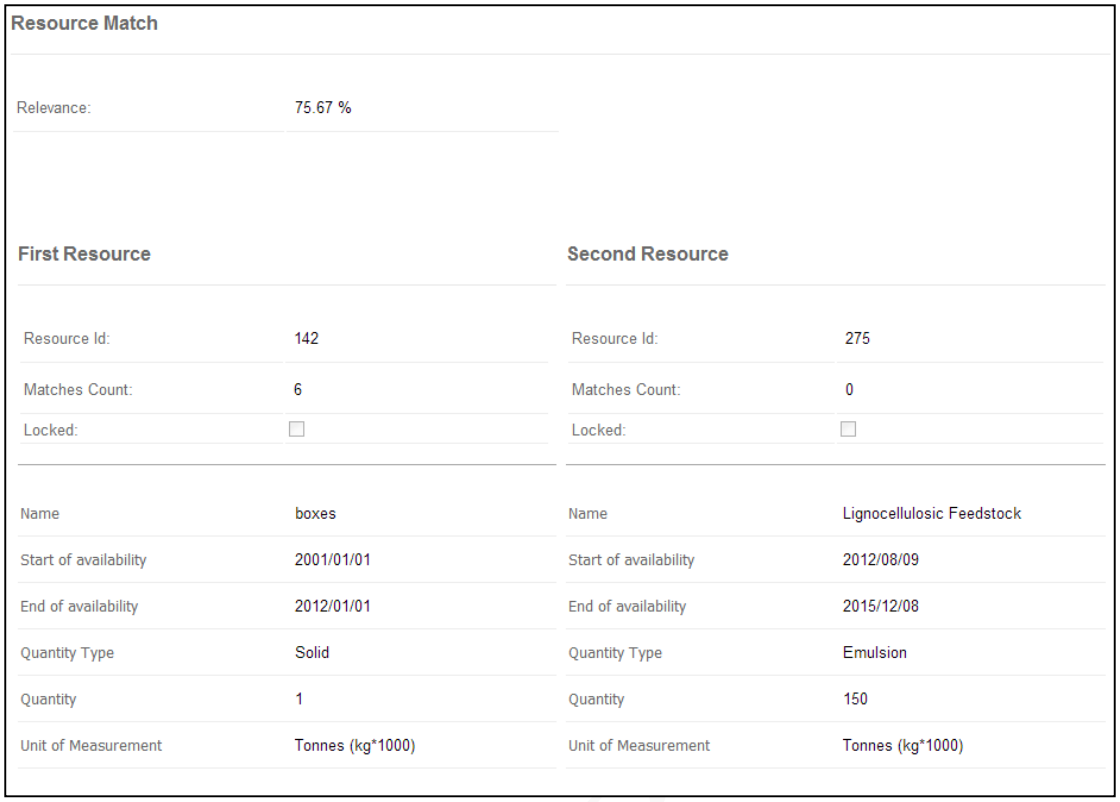
Synergy Lifecycle

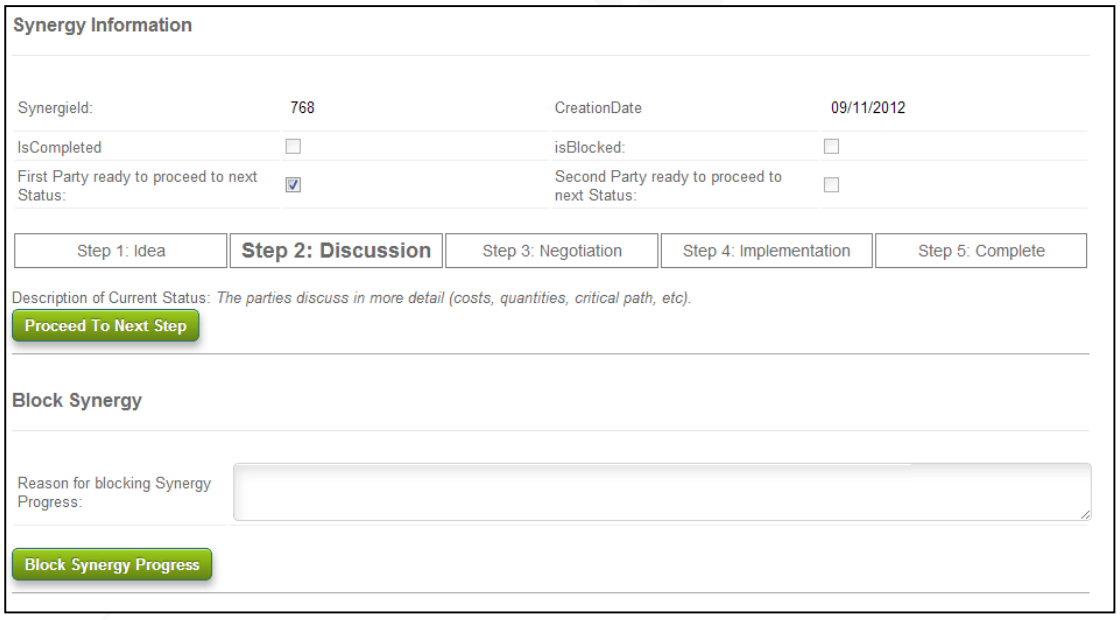
In the Synergy form, the two participating Organisations can:

- View details about the other participant
- View details about the resource match, the properties of the resources, etc.
- Submit their agreement to proceed to the next step of the Synergy lifecycle.
- Block the Synergy process for a specified reason, so that the IS Practitioner will have to be involved to resolve the issue and unblock the Synergy.
- Submit comments for the other participating Organisation.



First Participant		Second Participant	
Organisation:	Generic Power Supply Organisatic	Organisation:	New Materials Organisation
Web Address:	http://www.google.com	Web Address:	www.newmaterials.gr/
Organisation Notes:	This is the organisation that focuses mainly on energy production from gas or oil.	Organisation Notes:	Created for Demo purposes.
Date Added:	02/11/2012	Date Added:	05/07/2012
From Site:	Κεντρικη Ενεργειακη Μοναδα	From Site:	Νεα Επεξεργαστική
Address:	Βασιλίσσης Σοφίας 39	Address:	Κουμουνδουρου 74
Postal Code:	15136	Postal Code:	18545
Town:	Αθήνα	Town:	Πειραιας
County:	Αττικής	County:	Αττικης
Telephone:	2105465465	Telephone:	2104625785
FAX:	2105465465	FAX:	2104625785
Site Web Address:	http://www.powerSupply.com/Cen	Site Web Address:	www.newmaterials.gr/Sites/FirstS
Notes:	Ξεκινησε πειραματικα τη λειτουργια του το 2010.	Notes:	





The Synergy lifecycle has the following steps, as indicated in the corresponding screen:

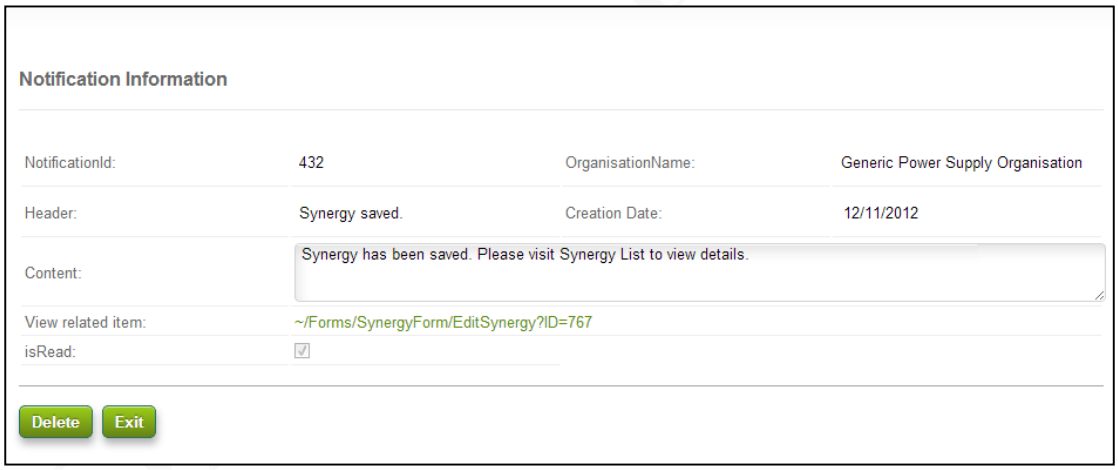
1. Idea
2. Discussion
3. Negotiation
4. Implementation
5. Completion

To progress from one step to the next, both Organisations have to agree that they

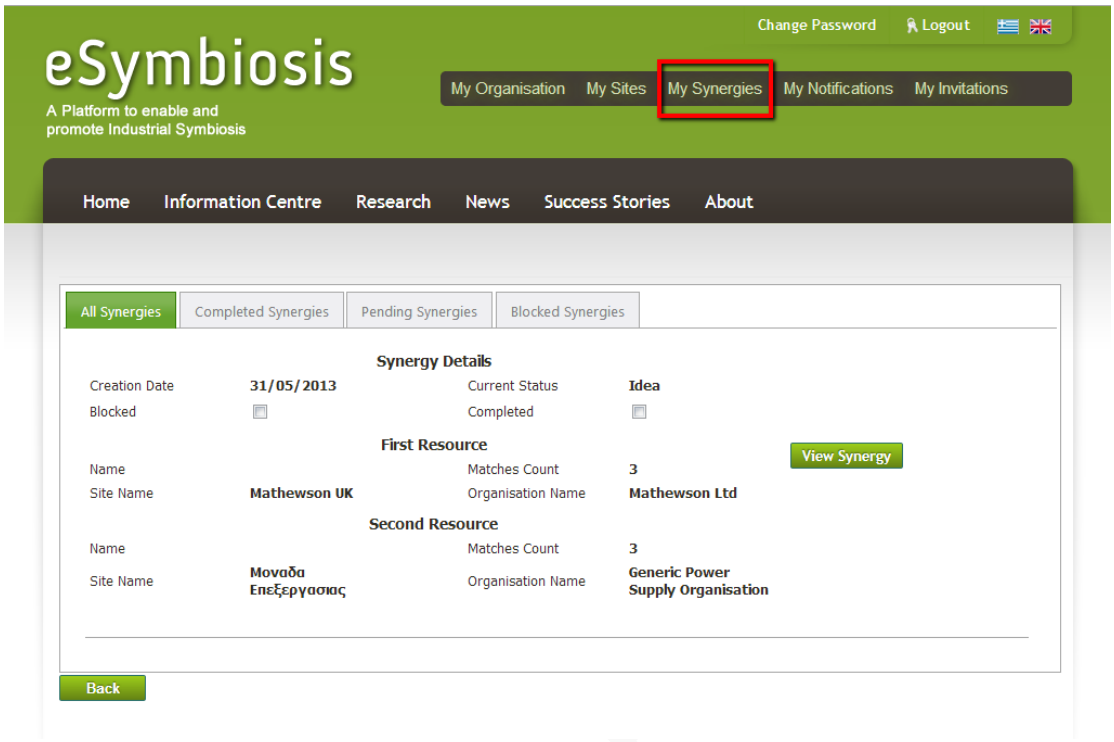
are ready to proceed, by selecting the corresponding checkbox (i.e. *First party ready to proceed to next Status*) and clicking *Proceed to Next Step*.

Then a notification is sent to the other Organisation user, in order to review the Synergy and maybe agree to the continuation of the process. When this agreement is mutual, the Synergy progresses to the next step and a notification is sent to both parties. A notification is also sent to participants when the Synergy has been updated or blocked.

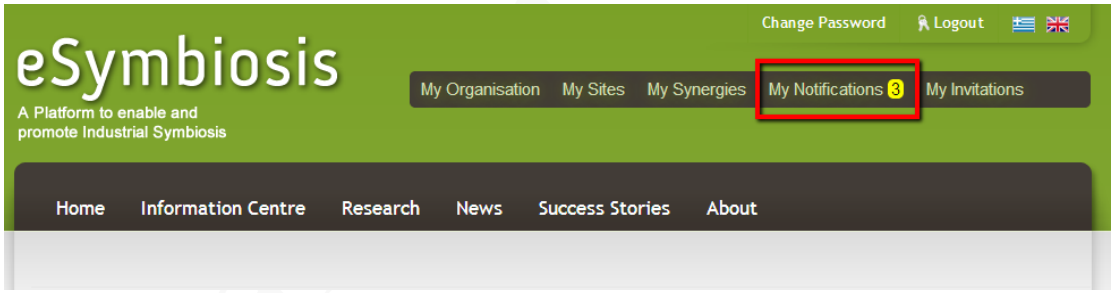
An example of the notification is shown below:



The participating organisations can access all their Synergies from the corresponding option of the upper right menu:



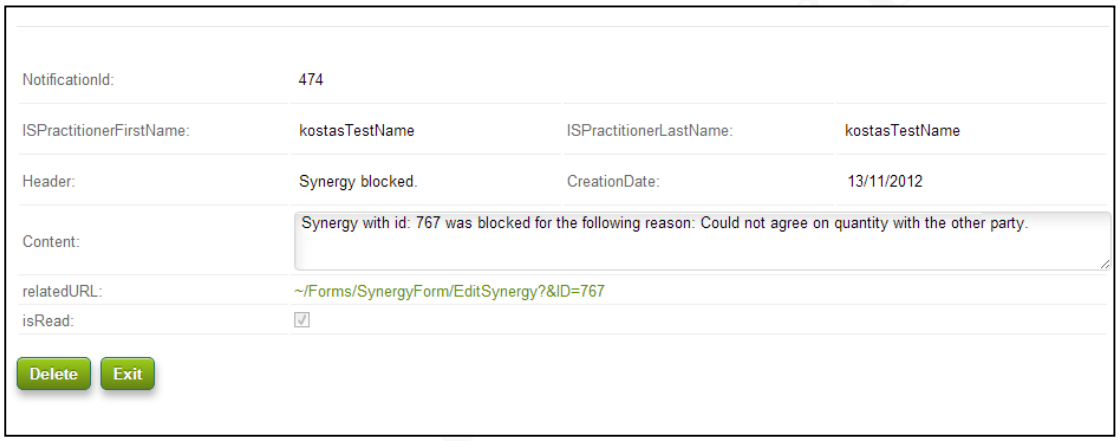
Also, the Notifications can be viewed at any point, from the upper right menu as well:



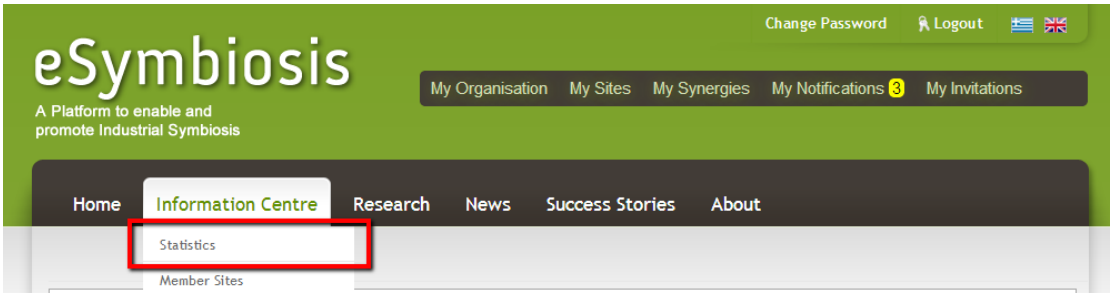
Blocking/Unblocking Synergy

As mentioned previously, the participating Organisations can block the Synergy progress from the corresponding option.

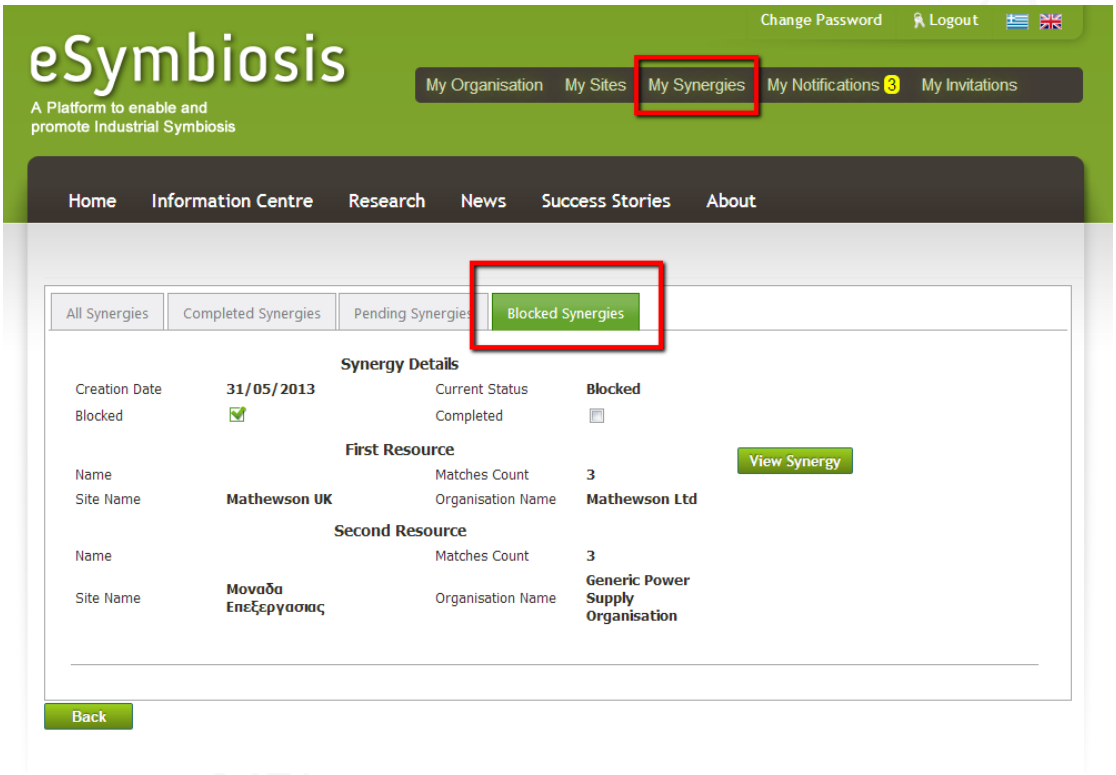
In case one of the two participants selects to block the Synergy progress, the IS Practitioner is notified about the issue. [Note that the IS Practitioner users are created from the Administrator].



The IS Practitioner is responsible for resolving the issue and can then unblock the Synergy from the corresponding button in the Synergy screen. After the Synergy is unblocked, its lifecycle can continue as normal.

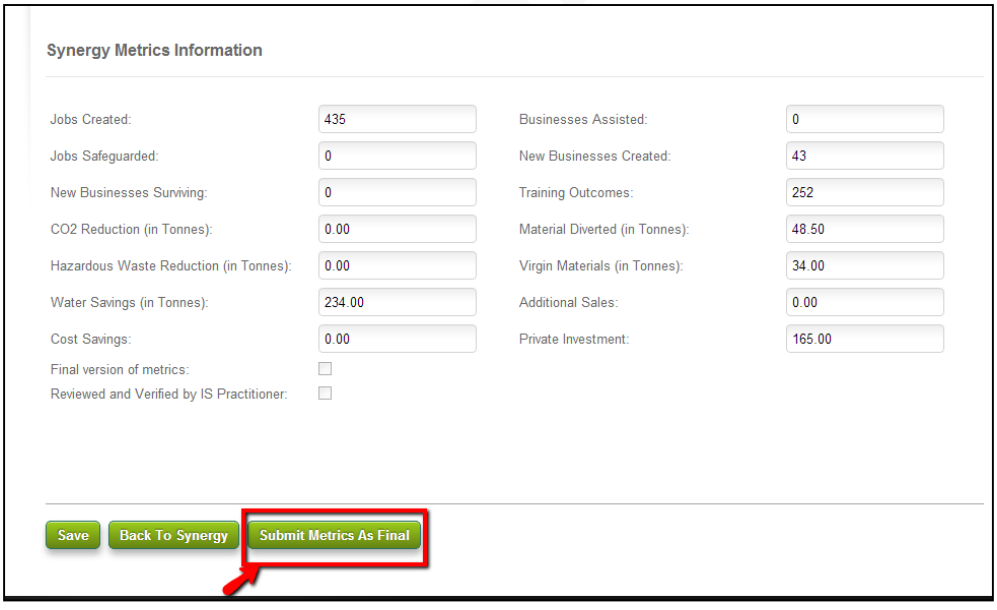


The IS Practitioner can access his/her list of Blocked Synergies through the Synergies page accessible through the upper right menu option and then in the last tab of that page.

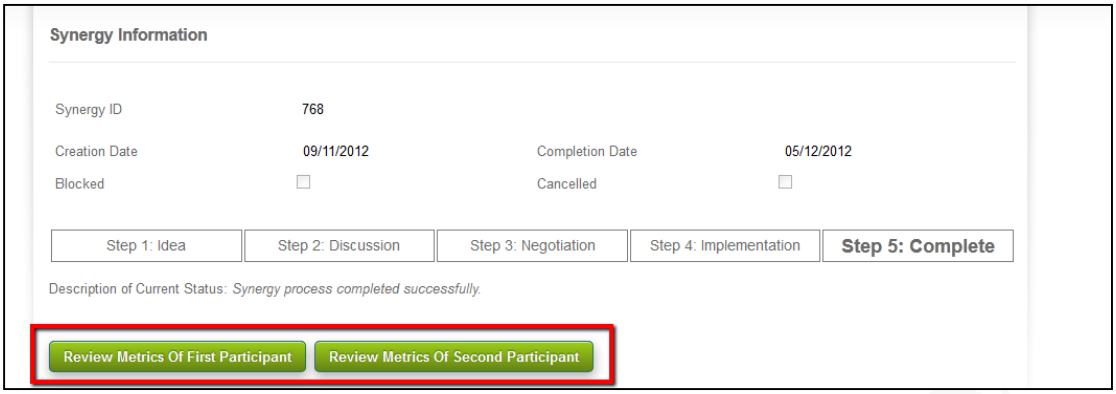


Metrics

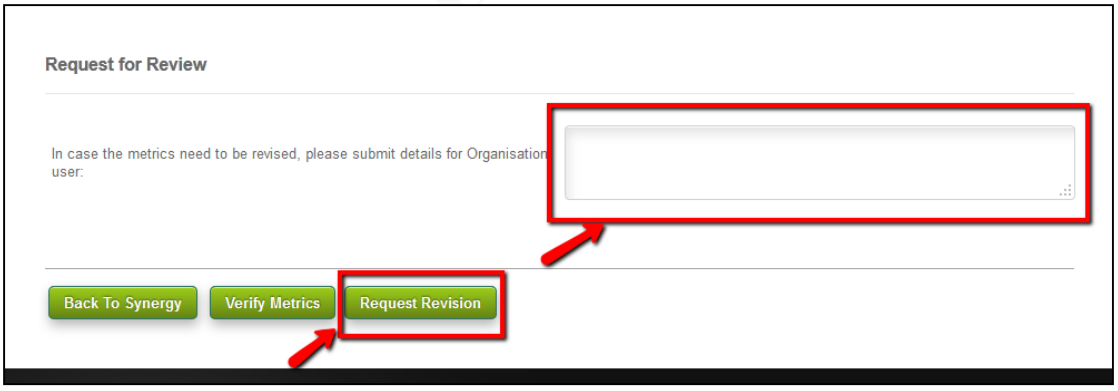
From Organisation user's perspective: The Organisation users have to submit a number of metrics, in relation to the Synergy. These metrics can be updated during the synergy progress (by clicking the "Review Metrics" button in the Synergy form) and they have to be finalised when the Synergy is completed; then, the Organisation user submits the finalised version of the metrics to the IS Practitioner for review (in the metrics form, the user clicks "Submit Metrics As Final").



From IS Practitioner's perspective: The IS Practitioner can review both metrics' info that the two members of the Synergy provide.



The IS Practitioner is notified that the user has submitted a finalised Metrics version for review. If the Metrics seem valid, the IS Practitioner confirms them by clicking "Verify"; otherwise, the Practitioner requests a revision from the Organisation user, by providing comments on the points that need correction and then clicking "Request Revision". This action will unlock the metrics' form the Organisation user, so that the metrics can be corrected and resubmitted. The platform informs each respective user (Organisation user or Practitioner) when an action is needed, with a notification.

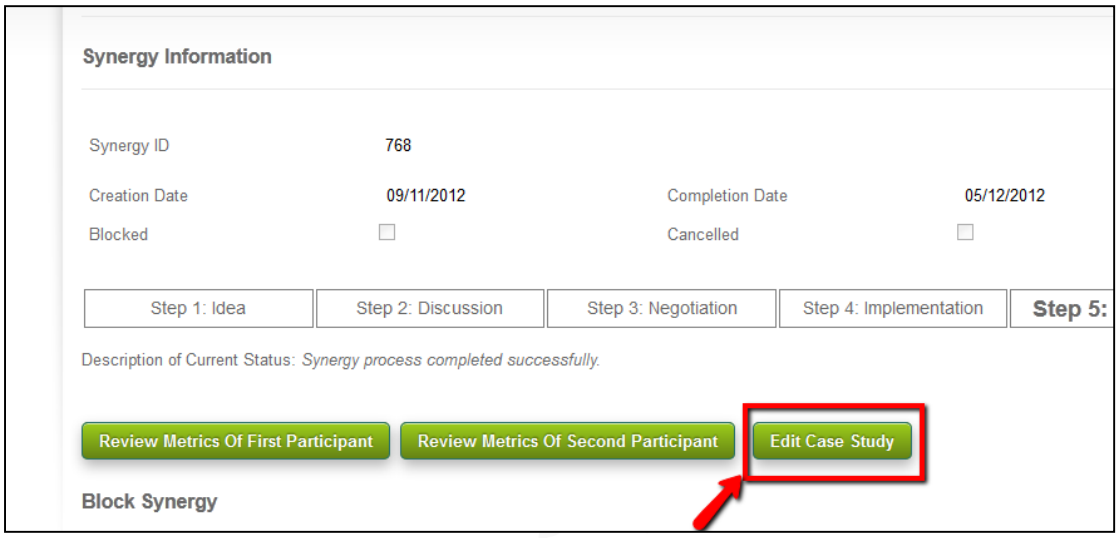


When both participants of the Synergy have completed the metrics submission and the IS Practitioner has validated them, the Practitioner can proceed to the generation of a case study based on the metrics, as the following section explains in detail.

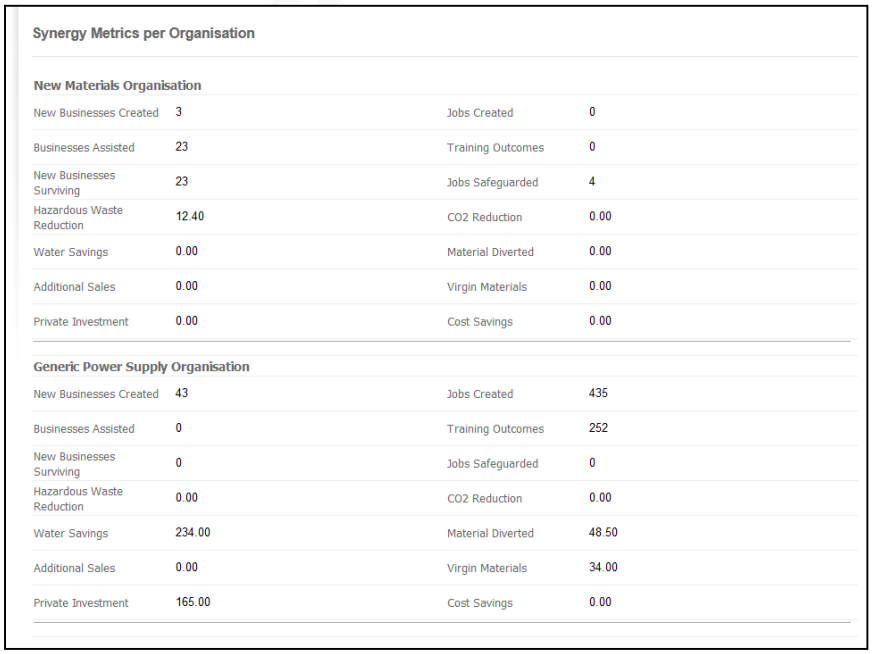
Case Study

After the Synergy has been completed and the metrics of the two participating organisations have been validated, the IS Practitioner can create a new Case Study based on the Synergy and its relevant metrics.

The IS Practitioner can access the case study form from the synergy form, as shown below (please note that the synergy must have been completed and both submitted metrics must have been verified by the IS Practitioner first).



In the case study form, the Practitioner can view the metrics of the Synergy (first screenshot below) and can add descriptive details in relation to the Synergy's



Synergy Metrics per Organisation

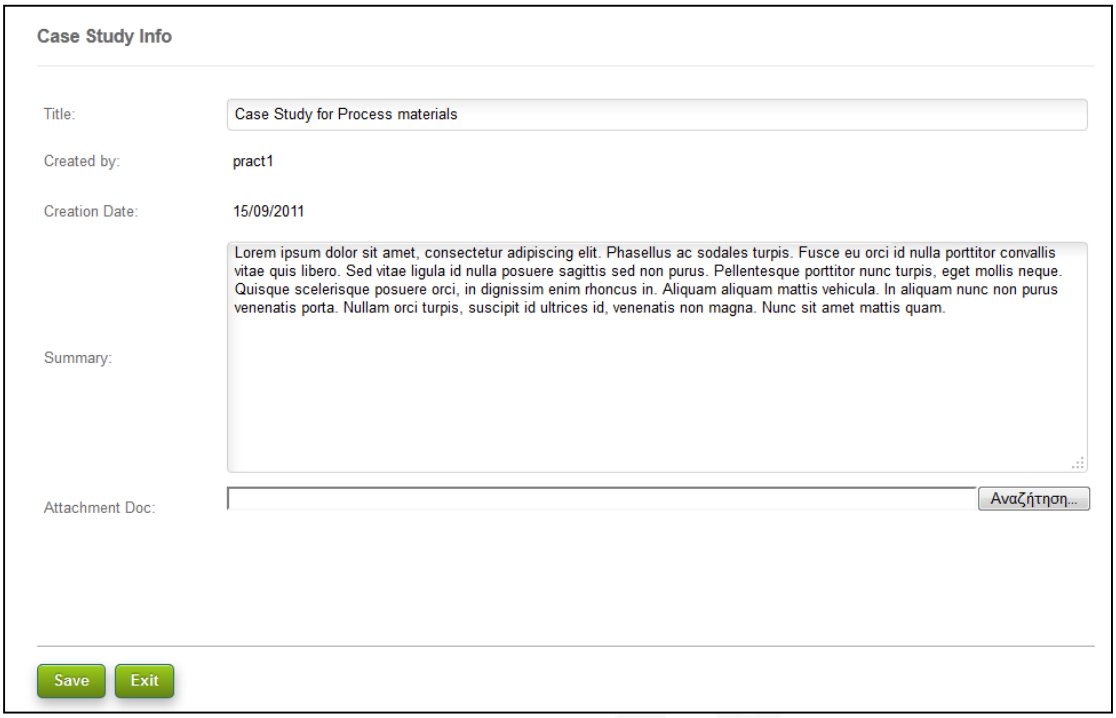
New Materials Organisation

New Businesses Created	3	Jobs Created	0
Businesses Assisted	23	Training Outcomes	0
New Businesses Surviving	23	Jobs Safeguarded	4
Hazardous Waste Reduction	12.40	CO2 Reduction	0.00
Water Savings	0.00	Material Diverted	0.00
Additional Sales	0.00	Virgin Materials	0.00
Private Investment	0.00	Cost Savings	0.00

Generic Power Supply Organisation

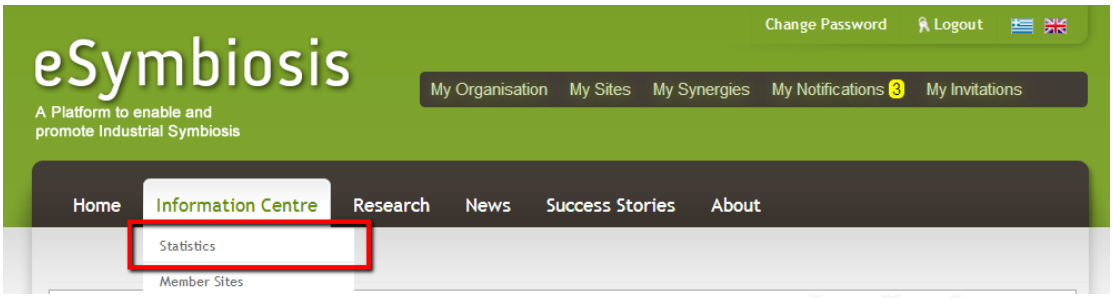
New Businesses Created	43	Jobs Created	435
Businesses Assisted	0	Training Outcomes	252
New Businesses Surviving	0	Jobs Safeguarded	0
Hazardous Waste Reduction	0.00	CO2 Reduction	0.00
Water Savings	234.00	Material Diverted	48.50
Additional Sales	0.00	Virgin Materials	34.00
Private Investment	165.00	Cost Savings	0.00

successful outcomes (second screenshot).

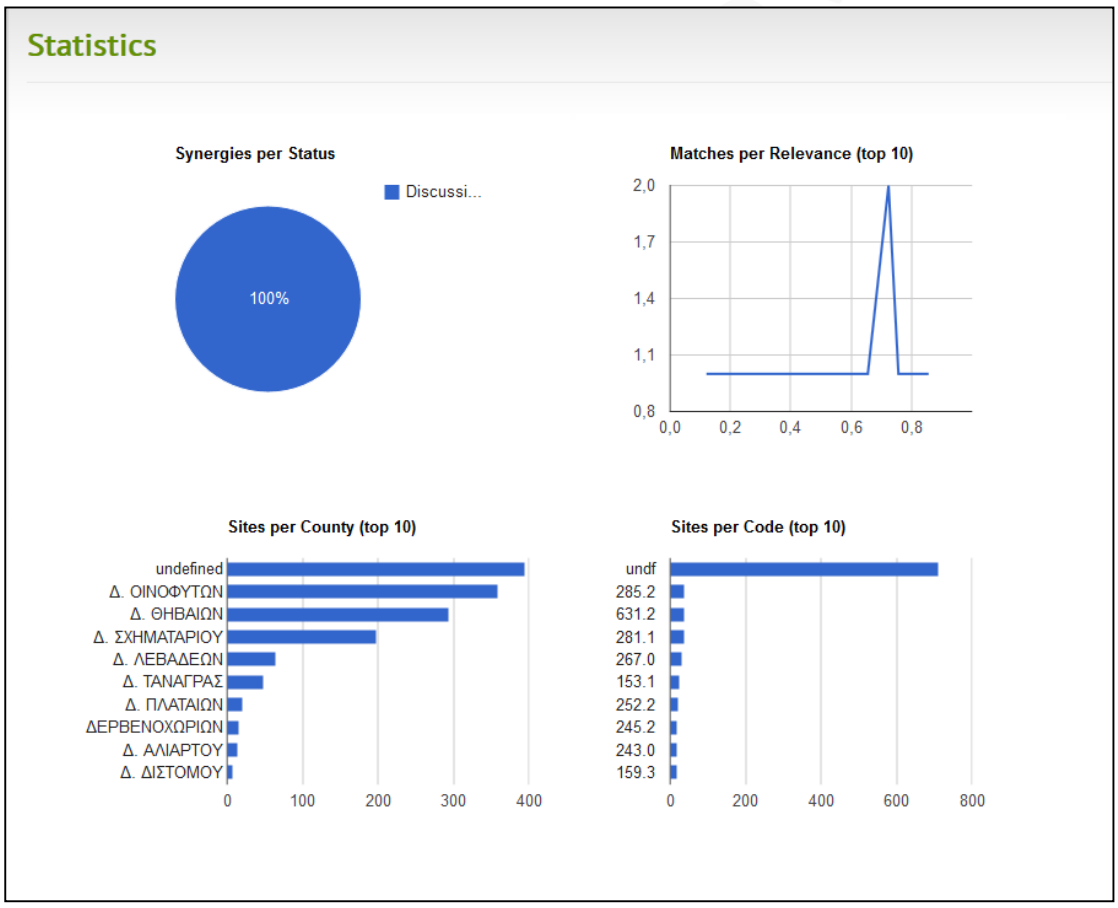


Statistics

The Organisation user can view Statistics in relation to the Synergies created, the matches, the Sites per county, and the Sites per code. To access this screen, the Organisation user clicks from the top menu: *Information Center* > *Statistics*.

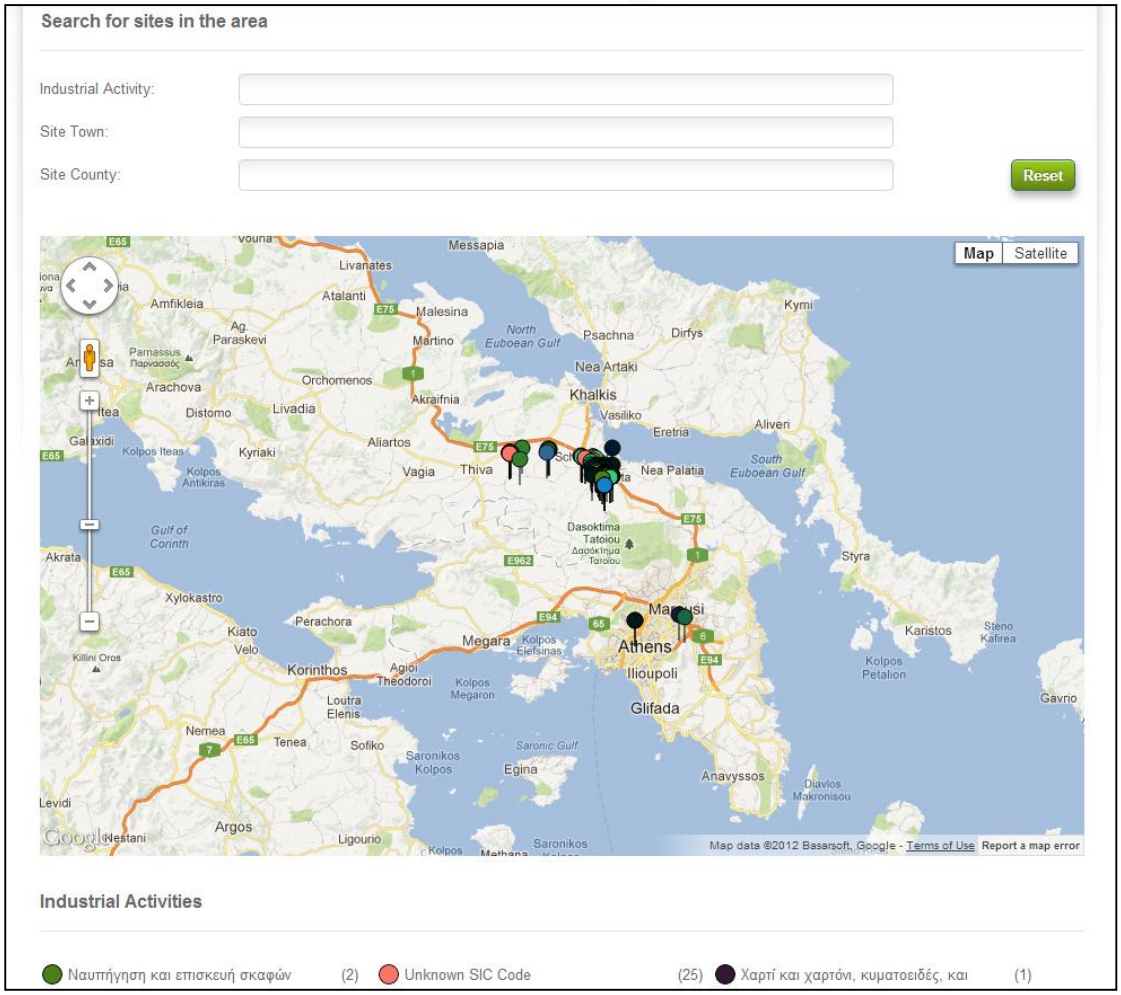


A view of the statistics is provided below:



Member Sites

Moreover, the user can go to: *Information Center* > *Member Sites*, to see the Sites located on a map and search for sites with specific criteria.



As shown in the relevant screen, the criteria that can be specified by the user are:

1. Industrial Activity
2. Site Town
3. Site County

The user can also select one specific Industrial Activity (by clicking on the corresponding Activity rectangle), to view only the Member Sites of this type of Activity, or select multiple Activities (by holding Ctrl and clicking multiple activity rectangles), to filter the displayed Member Sites for the selected Industrial Activities.

An example is shown below, where three Industrial Activities have been selected and only the corresponding Member Sites are shown on the map: