Web Service Scenarios and RO Models

Contents

Mashups

M1: Yahoo Pipes

The scenario is an example of creating a mashup using Yahoo Pipes. Yahoo Pipes is an interactive web application it enables the creation and execution of mashups. It offers a workspace in which a user can add widgets such as data sources, filters, and functions to refine and merge the data.

> *A user has built a stock quote watch mashup using Yahoo Pipes[1], this displays the last quote and chart for the stocks. In this example he uses the widgets provided to retrieve the original stock data from a .csv file stored at the Yahoo Finance downloads. He then uses a filter widget to filter the stock file for certain stock quotes. To loop through the obtained data he uses a loop widget that displays the results as a chart.*

Infrastructural and functional requirements

1. **Data retrieval interaction (Read only) -** The client can request data from the server, but cannot update or modify it. So the client cannot change the server's state.

2. **Stateless interactions with the server -** No interaction between the client and server involves maintaining client specific data on the server, and data requests are independent and isolated from each other.

3. **Proprietary Workflows -** The workflows description is not in an open format it is specific to the platform executing it.

4. **Workflows are controlled by and executed on one machine -** There is no need for a participation of multiple machines or a coordination of them.

5. **Server/Service provider ownership of data -** The data accessed by the client belongs to the service provider.

6. **Open Accessibility to the Data -** The data is accessible, there are no security restrictions.

7. **Creation of the workflows is done by the end user with a GUI -** Mashup Creator's level of expertise is minimal, the filtering and programming is through GUI no coding is required from the end user EU.

8. **No triggering of actions -** There are no tasks that are triggered as result of executing a mashup.

Non-functional requirements

1. Tolerance of failure - In this scenario, and many other mashup scenarios, mashups are used by end users for providing specialised data for non-critical tasks, so the failure of mashups do not have a large impact on other tasks.

<u>Scenario Breakdown</u>

The generic scenario of building mashups using Yahoo Pipes[1] is broken down to the following steps:

(1.) The client creates a mashup

(2.) It creates widgets that read inputs from other created widgets or external resources

(3.) The widget produces the results

(4.) The client reads the results

<u>Resource Oriented Model</u>

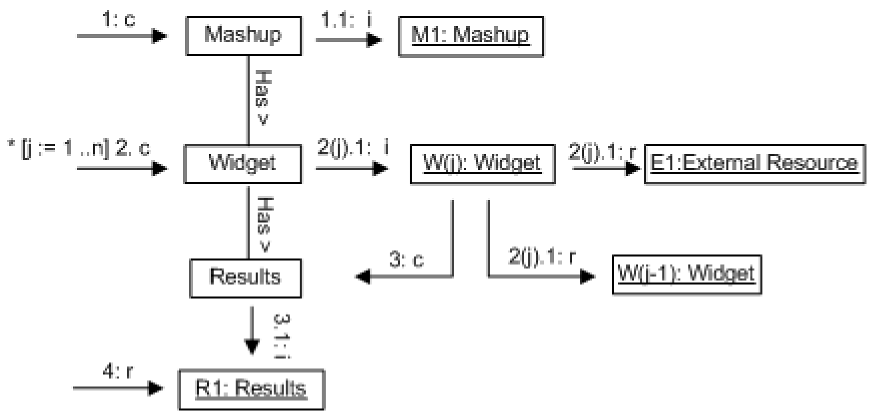

Figure 1 Modelling Mashups Creations with Yahoo Pipes

In this scenario step 2, (creating widgets) is iterative. We used the `*[j:= 1..n]` UML convention to indicate this. The `Has` links show the structural relationships between the mashup, its widgets, and the results.

The return messages are not shown in the modelling, that is because in ROA where HTTP is utilised, the server's response is standardised. This standardisation also applies to error messages.

M2: The MashMaker Scenario, Desktop Mashups

[2] describe MashMaker an interactive browser plug-in for creating mashups from Intel. The scenario provided explains how a user, who is planning to rent a house, uses MashMaker.
A user is interested in houses that have the best restaurants around. The user visits a housing website and adds it to MashMaker by clicking an icon in the browser. The houses are displayed in MashMaker as a tree where each house is a node, when a node is clicked MashMaker suggests appropriate queries like "things nearby". The user searches for food nearby, then applies a filter

widget to include only those within 0.5 a mile and having a rating of 3 or more. He adds a count widget to count how many restaurants match these criteria, and then copies this widget to the other houses, saves it, and publishes it.
The interaction occurs between the different web servers, where the data resides, and MashMaker on the client. The actual processing and aggregation of the data happens on the client. However in case of overlaying information on maps, Google Maps is utilised and some of the processing happens on the Google Maps Server then the results are transferred to the client.

Infrastructural and Functional Requirements
Similar to the requirements discussed in M1

1. No triggering of actions

There are no tasks that are triggered as result of executing a mashup.

Non-functional Requirements
Similar to the requirements discussed in M1

Technical Notes

1. Client/Intermediary Data processing, filtering and aggregation

The processing of the data is performed on the client or partially on an intermediary server like Google Maps.

2. Data aggregating compositions

The compositions involved in creating mashups are based on joining data providing services, where the composition depends on matching elements or attributes of the data.

3. Standards of data resources

The formats of the data sources vary, from HTML (web pages), RSS, JSON to RDF.

4. Scalability issue

Although the mashup is executed on the client there is a point that could affect the scalability of the architecture, this is the MashMaker server that hosts a database of extractors [3]. Extractors describe how to extract structured information from HTML pages. The creation and maintenance of extractors is done in a wiki collaborative manner. The scalability issue is minor if the extraction is executed on the client, which seems to be the case although is not explicitly stated.

5. Architecture
    1. Multiple Servers for Data Sources
    2. An intermediary server for maintaining extractors and mashup reuse
    3. An application on the client to create mashups

Scenario Breakdown
(1.) The user creates a mashup
(2.) The user creates two web resources that link to websites that s/he wants to mashup
(3.) The user updates the mashup to mash the two web resources
(4.) The user runs the mashup
(5.) The mashup returns the results

Resource Oriented Model

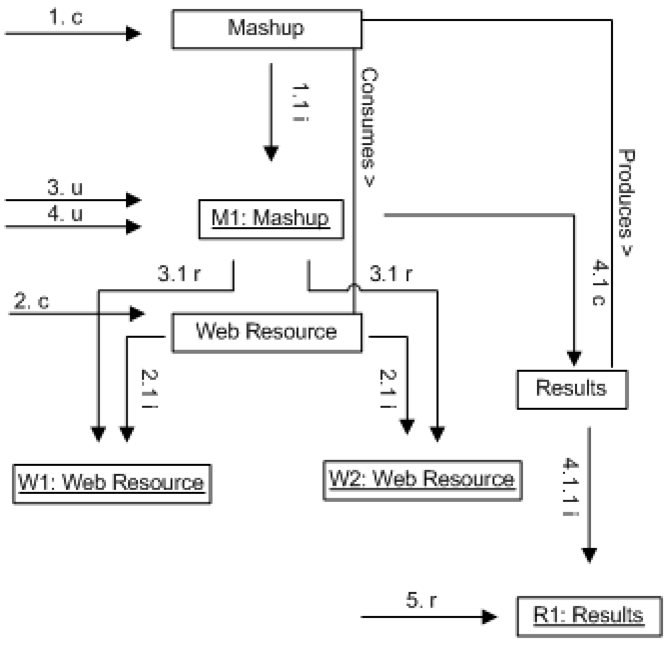

Figure 2 RO Model of M2

Modelling Issues

Creating more than one resource at the same time. (3)

There are widgets that the user adds to the Mashup that manipulate the web resources that were considered part of the mashup and not modelled as individual resources. This presents a different level of granularity to M2.

M3: Displaying the time and location of a Website's visitors using a layered mashup architecture

[4] explained their layered architecture for creating mashups from streaming data. Their approach is similar to Yahoo Pipes where the mashup architecture executes the mashup and the results are sent to the client. They provided an example of a mashup that combines a Web server's log file with a geolocation service.

In this scenario a user wants to display the geographic locations of a web site's visitors on a map, this map is constantly updated. He or she does that by using the system built on this architecture to access the web server's log through a secure shell socket (SSH) this provides real-time updates through a streaming push mechanism in contrast to a request/response mechanism using HTTP, which increases the latency and network traffic. The user then uses the system to create components to extract the IPs from the log, resolving the DNS, looking up the coordinates and overlaying them on a map which is the sent to the client.

The requirements are similar to the ones in scenario M2; however there are some additional ones

Infrastructural and Functional Requirements

1. Accepts Streaming Data pushed by servers

Unlike other approaches it accepts data pushed to the mashup engine over open ports

Non-functional Requirements

1. Secure Access

In this scenario access is enabled to access secure files on remote web servers using SSH

Technical Notes

1. Standards of data resources
Web logs, RSS, JSON, accesses data from Web Services using SOAP.

Scenario Breakdown

(1.) The application reads a Web log
(2.) A local copy of the Web log is created
(3.) The client reads the local copy (The search is discussed in the modelling issues)
(4.) IPs are extracted from the web log
(5.) The IPs are read to be sent to the DNS
(6.) Resolve the IPs at the DNS
(7.) Create a resource representing the DNS coordinates
(8.) Getting the Coordinates from the DNS
(9.) Creates a Map
(10.) Overlay the Map with the coordinates

Resource Oriented Model

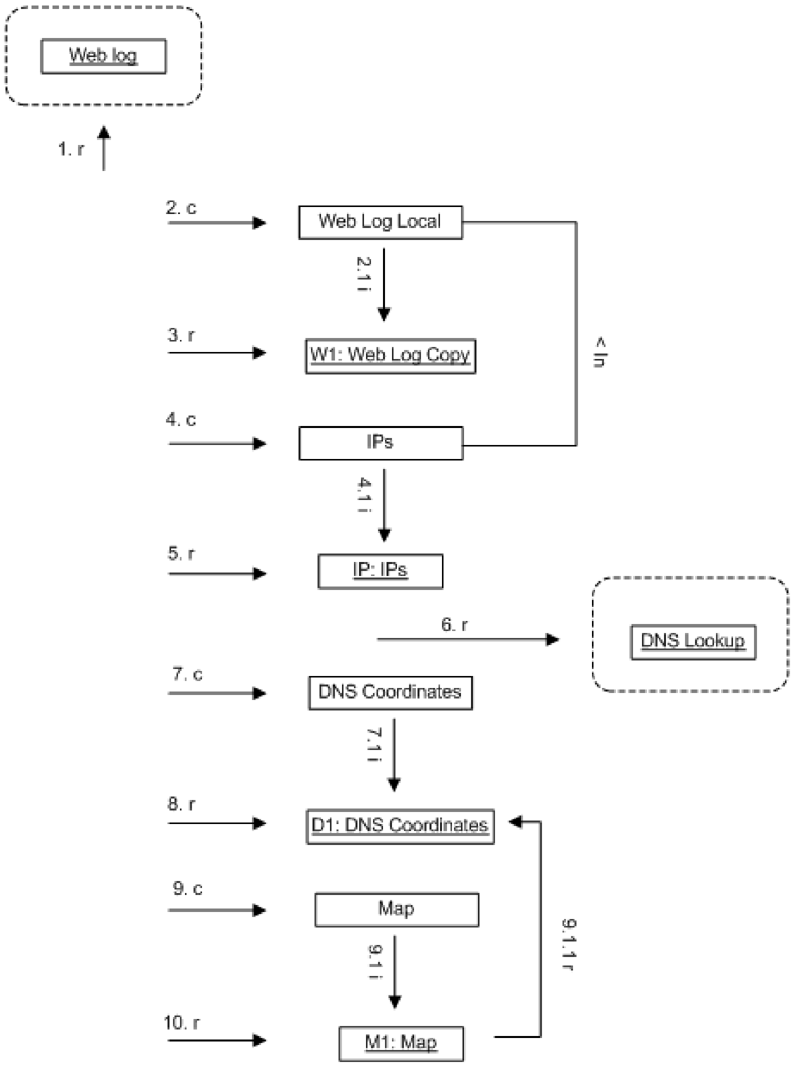

Figure 3 RO Model of M3

Modelling Issues

1. The exchange of roles between server and client, servers become clients of other servers, within the server it is not clear how the business logic is controlled, it is not like controlling the client transition through hyperlinks. It is more like an internal workflow within the server. Not sure how to model that.
2. Copying the Web log
3. The Web Log is pushed to the mashup server, it is not read, as the authors put it is a streaming input, however the initiation of the stream is requested.
4. How to model a search:
    - A resource with a query string URI which the client formulates then Gets
    - Create a query resource, this creates a result resource
    - Create a search results resource, structurally connected to the resources it searches

M4: Creating situational applications using the enterprise information mashup fabric

In [5] the author discusses the enterprises need for *Situational Applications* the author described them as "applications that come together for solving some immediate business problems". The paper described two scenarios to illustrate where it would be useful.
M4A: *In the first example a salesperson needs information on a client before making a call on prospect. The information needed is how much was sold to the customer during the last quarter, and did the customer have problems with sales.*
M4B: *A CFO that has a meeting with his CEO. The CFO wants to present a summary of the financial picture. This summary needs to be assembled from emails by finance personnel including presentations that contain embedded spreadsheets about the financial picture.*

Infrastructural and Functional Requirements
1. Information Assembly

In M4A there is no merging done on the data on information assembly

Non-functional Requirements
1. Closed

The system is to be used inside an enterprise.

Technical Notes
1. Standards of data sources

The data depends on the applications that the enterprise uses, the more the mashup engine understands the formats of enterprise data, the more useful it would be.

Scenario Breakdown
(1.) Query the Customer info
(2.) Reading the results of the query

Resource Oriented Model

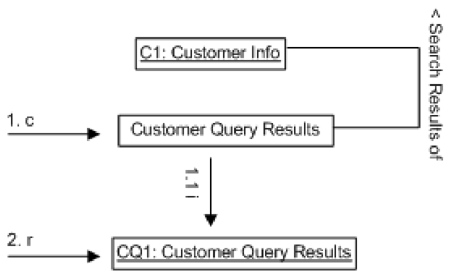

Figure 4 RO Model of M4

Modelling Issues

1. What is modelled is M4A, M4B is the same with the a change in the resource names.
2. However a requirement is having access to parts of resources, and although this is not related to modelling and is an implementation issue but it could be done by having dependent and independent URIs.
3. If a resource only exists as a part of another resource it is a dependent resource and its URI is an extension of the resource it belongs to URI

   Eg. www.example.com/resource1/resource2
   Otherwise it is independent
   Eg. www.example.com/resource2

Enterprise Services

E1: City University

The scenarios chosen were two integration projects from City university[6]. The first project was Single Sourcing of Programme Data (SSPD). The university uses information about the study programmes in different processes, like the producing student handbooks, publishing programme information on the website, producing prospectus and quality and approval processes for development of new programmes. These processes are using the same information but they were ope rating independently this lead to inconsistencies data and effort duplications.

> *SSPD is concerned with how programme information is created, updated and used. So that the processes mentioned above could be facilitated and any inconsistencies resolved. It enables academic and administrative staff to define and maintain module and programme specifications and submit them for approval.*

The other integration project from city is called Managed Learning Environment (MLE)

> *The University uses both SITS:Vision student information management system, and Virtual Learning Environment (WebCT Vista). The transfer of student information from SITS:Vision to WebCT Vista took place using a nightly scripting process, this was slow and had errors. MLE aims to have the SITS system trigger the updating process so new information is added to WebCT directly.*

Infrastructural and functional requirements

1. Complete control over the service providers and service consumers - The university systems are the service providers and the service consumers there are no external entities involved.

2. Actions are triggered as a result of service invocation, so it is not a read only situation - The state of resources can be altered because of the service invocation

3. The ability to deal with multiple systems and data formats – The services deal with legacy systems that use different technologies and formats to represent the data.

Non-functional requirements

1. Communication between the subsystems is controlled by the same entity - This implies that security measures are less stringent because the services are not available to external consumers.

Scenario Breakdown

The SSPD scenario from City University [6] can be decomposed into:

(1.) Academic Staff reads the program info
(2.) Creates a modification
(3.) Can update it, when it is finished
(4.) It is approved by the Administrative staff
(5.) The Program info is updated
(6.) It can be read by interested processes

Resource Oriented Model

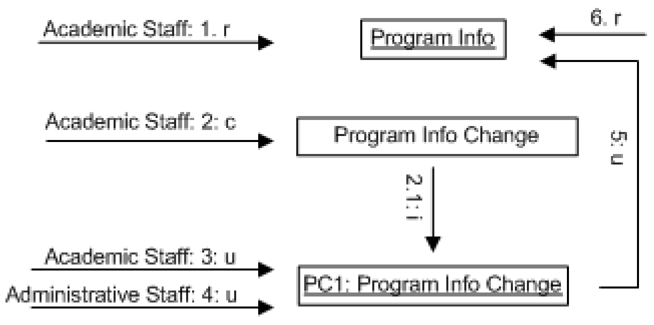

Figure 5 Modelling City University's SSPD

With step (3.) an update can also change the status of the modification to indicate it is ready to be submitted. **Error! Reference source not found.** shows how roles are modelled, with the name of the role associated with the action on the messages.

Scenario Breakdown

The other integration project from City (MLE) is modelled below. The steps involved in MLE are

(1.) The SITS system creates updates
(2.) The SITS system notifies WebCT
(3.) WebCT reads the changes and gets updated

Resource Oriented Model

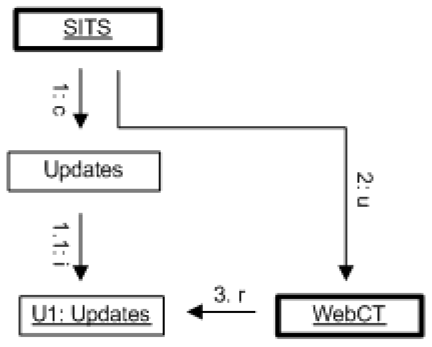

Figure 6 Modelling City University's MLE

WebCT and SITS are active resources, indicated by the heavy lines (a UML convention). This means they initiate control activity.

E2: Information Enterprise to Enable Net Centric Operations in the US DoD

The US DoD aiming to provide Net centric operations, where different systems coordinate in case of warfare or national security alerts. The issues facing this coordination are discussed in [7]:

In the DoD there is large number systems both legacy and emerging, these systems are aiming to provide Net centric operations. However these systems are large, monolithic with different infrastructures in terms of: transport, network, data, interface layers, etc). This limits data exchange among them and interoperability is a main concern. Moreover these systems are configuration intensive and difficult to manage. Maintaining these systems and keeping them up to date is very expensive and replacing them is usually cost prohibitive. Although these systems are for the DoD, however they are for several divisions, as a result they:

1. *Have different conceptual basis.*
2. *Stand alone because of the acquisition environments*
3. *Have different management*
4. *Do not share common funding*
5. *Have different customers*
6. *Have different evolvement rates*

Net Centric operations aim to prepare for the unknown. It is not known in advance what information is required or what collaborations and coordination must take place. This means that the system must be flexible and rapid in terms of customisation, re-configuration when required.

Infrastructural and Functional Requirements

1. Federated control over the service providers and service consumers
Although the subsystems are under the control of the DoD, however they are autonomous systems that were not designed to work together.

2. Service Coordination
Several services from different subsystems must be made to work together.

3. Workflow Support
Service coordination requires workflow support from the service infrastructure

4. Actions are triggered as a result of service invocation, so it is not a read only situation
The state of resources can be altered because of the service invocation

5. Service Discovery

To provide the flexibility needed, the infrastructure should provide means for automated or semi-automated discovery

6. The ability to deal with multiple systems and data formats

The services deal with legacy systems that use different technologies and formats to represent the data.

7. Notification

Running services send notifications to the clients or to the service coordinators.

Non-functional Requirements

1. Flexibility

The functional and infrastructural requirements are changing and not known in advance

2. Rapid development

The development of the integration is a response to urgent situations and must be done quickly

3. Timely response

Net centric operations are time critical, delay is not tolerable.

4. Security

The security involves:

   a. The authentication and authorisation of the service consumers and service providers
   b. The encryption of the communication to guarantee confidentiality

5. Reliability

This includes

   a. The availability of services
   b. The recoverability of data and applications

Resource Oriented Model

There is no clear scenario to model, however the key requirements are:

1. Flexibility as the requirements are changing
2. Rapid development
3. Security

Resource oriented architecture can fulfil these requirements because:
For flexibility, realising the resources in a system will facilitate combining them in different ways, it will enable reuse since the Because of the uniform interface accessing and designing the interaction with external systems will be easier than connecting to external systems which have different interfaces, and hence more rapid. Because of fine grained Role Based Access Control (RBAC) controlling access to the system will be broken down to the finest level, and restrictions can be controlled easily because it is reduced to:

1. identifying resources
2. identifying roles
3. specifying what action each role can perform on a resource

E3: Integrating BT's Operational Support Systems (OSS)

BT used Web Services to integrate core operational support systems (OSS) which are legacy subsystems to enhance existing services or provide new ones. The following scenarios mentioned in [8] illustrate this:

BT.com Online website

BT.com offers many customer services such as 'View my bill', 'Friends and Family', etc. BT would like its customers to use the website because it reduces the cost of operator-assisted services. BT.com needs access to core services from multiple internal heterogonous sub-systems.

Project SCORe (Service Consolidation and Operational Revitalisation)

A problem that was identified with the call-centres is the complexity of retrieving the data relative to a customer's contact. SCORe aims at reducing costs and increasing customer satisfaction. Because the data is held in multiple databases and controlled by several systems; this means that several calls to these systems were needed using different technologies.

<u>Infrastructural and Functional Requirements</u>

1. Complete control over the service providers and service consumers
   BT systems are the service providers and the service consumers there are no external entities involved.
2. Actions are triggered as a result of service invocation, so it is not a read only situation
   The state of resources can be altered because of the service invocation
3. The ability to deal with multiple systems and data formats
   The services deal with legacy systems that use different technologies and formats to represent the data.

<u>Non-functional Requirements</u>

1. Communication between the subsystems is controlled by the same entity

This implies that security measures are less stringent because the services are not available to external consumers.

<u>Scenario Breakdown</u>

BT.com

- The customer can read the bill, this will invoke reads to the subsystems
- The customer can update and read Family and Friends options, this will also invoke update and read requests to the system

Score

- The operator can retrieve customer information, which then retrieves it from the subsystems.

<u>Resource Oriented Model</u>

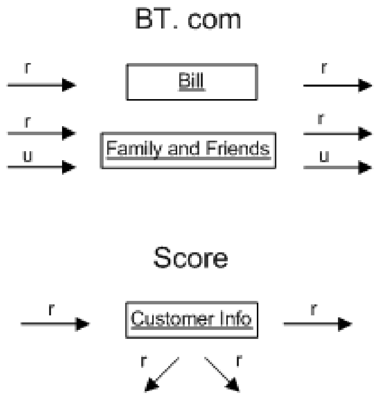

Figure 7 RO Model of E3

<u>Modelling Issues</u>

No sequence numbering.

E4: Documenting Software Components

In [9] the author discusses the problem of reusing software components in large enterprises and the automation of this reuse. He explains that current technologies aren't sufficient. The following is a scenario based on the issues he mentioned:

A programmer creates an internal software component then generates a WSDL description of it using a tool such as axis. This WSDL description doesn't help the next programmer hired to work on this component, as it doesn't describe how to use this component. The automatically generated WSDL file is tens of pages long which is only useful to specific programs, that is because an entire process has been converted to a web service. Since it is hard for a programmer to understand and use such a component it would be impossible for some other program to do so automatically.

So the main concerns in the scenario

1. Making components reusable for other developers, by having a clear descriptions
2. This could facilitate the automation of the reuse

Infrastructural and Functional Requirements

Automatic/Semiautomatic discovery

Clear service descriptions are needed to enable discovery and hence reusability.

Non-functional Requirements

Readability and reusability of software components

Resource Oriented Model

There is no clear scenario to model, however the key requirements are the reusability and clarity of the software design.

So the key 3 things that must be described in the documentation of the system are the

1. Representation of the resource
2. Transitions allowed from 1 resource action to another
3. Allowed actions on the resources

The more the design and implementation reflects these descriptions, the more the system becomes understandable and reusable.

B2B

B1: Reverse Auctioning

The scenario modelled here is a reverse auctioning scenario mentioned in [10]:

> *"A buyer (e.g., car manufacturer) uses reverse auctioning for procuring specially designed components. In order to get help with selecting the right suppliers and organizing and managing the auction, the buyer outsources these activities to an auctioning service. The auctioning service advertises the auction, before different suppliers can request the permission to participate in it. The suppliers determine the shipper that would deliver the components to the buyer or provide a list of shippers with different transport costs and quality levels, where the buyer can choose from. Once the auction has started, the*

suppliers can bid for the lowest price. At the end, the buyer selects the supplier according to the lowest bid. After the auction is over, the auctioning service is paid."

Infrastructural and functional requirements

1. Registration - The auctioning service deals with many participants/clients that need to register before using the service. This implies the need for authentication and authorisation

2. Support for different client roles - There are two different roles for users of this service: buyers and suppliers.

3. The service provider and the service consumers are different entities
The service provider is the auctioning services, and the consumers are the buyer and the suppliers.

Non-functional requirements

1. Security
This involves authentication and authorisation for service consumers and encryption of payment transactions.

<u>Scenario Breakdown</u>

The reverse auctioning scenario mentioned in [10] could be broken down into these steps

(1.) The buyer creates an auction
(2.) The buyer starts the auction
(3.) The suppliers place their bids
(4.) The buyer selects a bid
(5.) The buyer pays for the service
(6.) The buyer deletes the auction

<u>Resource Oriented Model</u>

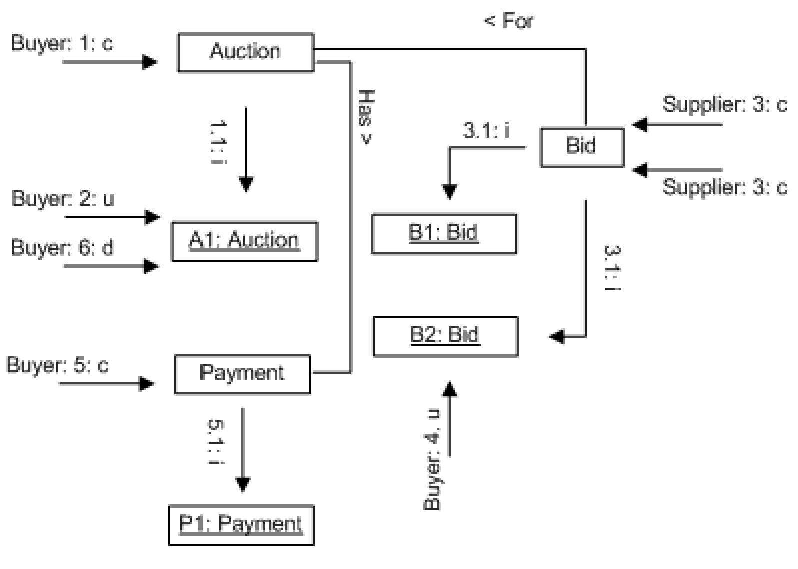

Figure 8 Modelling Reverse Auctioning

As in **Error! Reference source not found.** the messages are annotated with roles.

B2: Telecommunications Wholesaler

In [11] the authors discussed an IBM project that aims to enable a large telecommunications wholesaler to supply services to more than 150 customers. The wholesaler owns the physical network. The customers are either telecommunications companies extending their own network infrastructure, or companies that want to bundle telecommunication services with their products. These customers will use the order management services of the wholesaler to connect, configure, or disconnect telephone services for end users. The order management application should offer two main processes:

1. Provide a new Public Switched Telephone Network (PSTN) telephone service.
2. Move a PSTN telephone service to a new address.

A customer needs to follow the next steps, summarised from [11], in order to perform the aforementioned processes:

1. *Identify the service to be moved and its current location or site address.*
2. *Identify the new address for the service. This has to be the address as recognized by the systems that record telecommunications plant and service information. Hence search aids are required.*
3. *When a recognized address is identified, the next step is to search for transmission cable plant which exists at the target address and could be reused for provisioning this service.*
4. *Having identified a particular copper transmission path, this result has to be recorded.*
5. *Determine the features of the service at the new address which depends on a complex set of factors. Some features may already exist from a previous service at this address, some transferred from the old address, and some may be requested.*
6. *Next, determine a phone number for the service at the new address and reserve it. The old number maybe kept if the network at the new address permits, otherwise a list of numbers available must be supplied.*
7. *If a visit is required, then a time must be negotiated which suits both the customer and the field staff to be assigned to the task.*
8. *The request to move and the reservation is confirmed, allowing the commercial transaction to proceed.*

Infrastructural and Functional Requirements

1. Negotiation

The service infrastructure should support conventions that enable the service provider and service consumers to negotiate.

2. Workflow support

The processes needed involve the invocation of several services in a certain order.

3. Conversational services

The service infrastructure should enable execution of services where the all inputs the inputs cannot be known upfront.

Non-functional Requirements

1. Security

This involves authentication and authorisation for service consumers

Scenario Breakdown

(1.) The client creates a service request
(2.) Adds the new address of the service
(3.) Determines the features of this service

(4.) A number is created

(4.A) A list of new numbers

(4.B) The old number is kept

(5.) Increase the RAM in the machines

(5.A) The client chooses a number

(5.B) The old number is read

(6.) [Optional] A visit is arranged

(7.) The client pays for the service

Resource Oriented Model

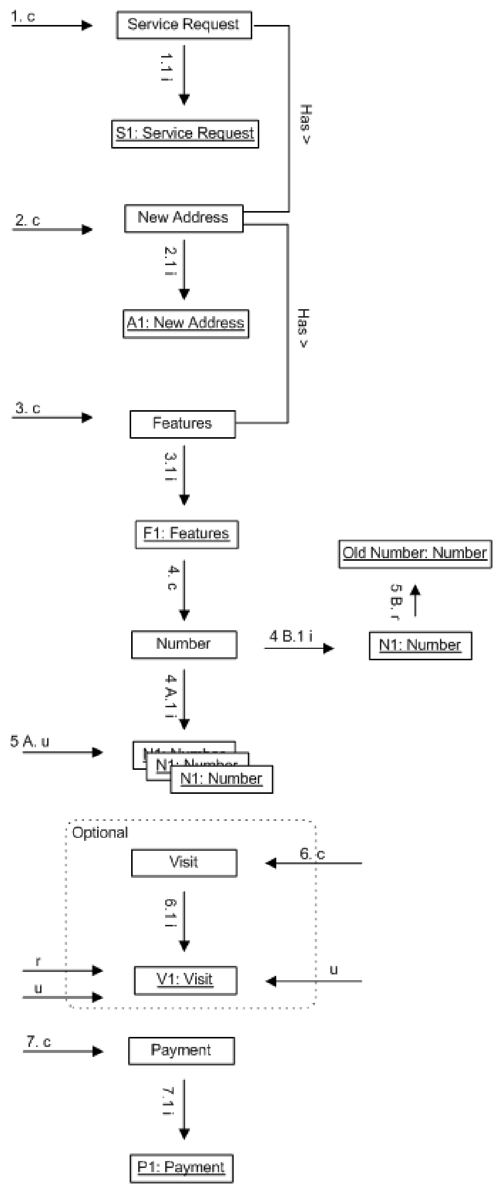

Figure 9 RO Model of B2

Modelling Issues

1. Not all structural relationships have been modelled e.g. Payment and Service Request
2. 4.c, 6.c and u who initiates them?

3. How to represent optional parts in a better way
4. (7.) Negotiating the visit? How is that represented?

B3: E-Procurement

[12] presents and e-procurement general scenario:

"E-procurement has a buy side, a sell side, and the connection of the two. On the buy side, a customer such as a company purchasing agent needs to access information on all relevant products, including product specifications, comparisons with all competitive products, pricing including discounts, delivery arrangements, and promises. The seller must have all relevant information on the buyer, including company, finance, credit, contact, logistics, preferences, and legal. On the sell side, the vendor must provide all relevant, up-to-date catalogue information from hundreds or thousands of suppliers together with real-time inventories and pricing. For a sale, transaction details must be irrefutably committed on both sides, and reflected in the inventory and financial systems."

Infrastructural and Functional Requirements

The characteristics are identical to the ones in scenario B2

Non-functional Requirements

1. Security

This involves authentication of buyers and sellers and the encryption of payment transactions.

Scenario Breakdown

(1.) The buyer reads the catalogue
(2.) The buyer places the order
(3.) The seller provides the pricing for that order
(4.) The buyer reads the pricing
(5.) The buyer provides the payment

Resource Oriented Model

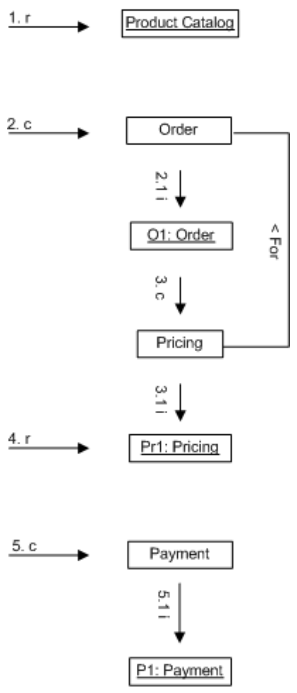

Figure 10 RO Model of B3

B4: Supply Chain Management

A scenario mentioned in [13] illustrates an example of a supply chain and the different entities and interactions involved:

> *"We consider a manufacturing company in Bristol, UK which needs to distribute its goods internationally. It does not maintain its own transportation capability, but instead outsources this to other companies, which we refer to as Freight Forwarders. These companies provide a service to the manufacturing company – they transport crates on its behalf. However, the manufacturing company still needs to manage relationships with these service providers. One role within this company, which we refer to as the Logistics Coordinator, is responsible for doing this. Specifically, it carries out the following tasks;*
>
> 1. *Commissioning new service providers, and agreeing the nature of the service they will provide. (E.g. locating a new freight forwarder in Poland, and agreeing that it will regularly transport crates from Gdansk to Warsaw.)*
> 2. *Communicating with service providers to initiate, monitor and control shipments. (E.g. informing the Polish freight forwarder that a crate is about to arrive at Gdansk; receiving a message from them that it has been delivered in Warsaw, and they want payment.) This is done using one of the messaging standards, EDIFACT.*
> 3. *Coordinating the activity of service providers to ensure that they link seamlessly to provide an end-to-end service. (E.g. making sure the shipping company plans to deliver the crate to Gdansk when the Polish transport company is expecting it. Informing the Polish company when the shipping company is about to drop it off.)*
> 4. *Communicating with other roles in the company to coordinate logistics with other corporate functions. (E.g. sales to know what to dispatch; financial to ensure payment of freight forwarders.)*
>
> *In our scenario, we consider a specific logistics supply chain from Bristol, UK to Warsaw, Poland. It consists of three freight forwarders: The first is a trucking company, responsible for transporting crates from the manufacturing plant in Bristol to the port of Portsmouth, UK. The second is a shipping company, responsible for shipping crates from Portsmouth to the Polish port of Gdansk. The third is another trucking company, which transports crates to the distribution warehouse in Warsaw. We assume that the Logistics Provider communicates with the Freight Forwarders using the EDIFACT standard, and is already successfully using this logistics chain."*

A problem arises when the shipping company becomes unavailable and the Logistics Coordinator must find an alternative company, the new potential provider uses RosettaNet as a standard for communication and the Logistics Coordinator needs to negotiate with it.

<u>Infrastructural and Functional Requirements</u>

The requirements are identical to the ones in scenarios B2 and B3 however there are others:

1. Mediating between different standards

In this example EDIFACT and RosettaNet, and this involves both the mediation of data and the mediation of protocols used.

2. Discovery of services

   In this example EDIFACT and RosettaNet, and this involves both the mediation of data and the mediation of protocols used.

<u>Non-functional Requirements</u>

Identical to B1s requirements

Scenario Breakdown

(1.) Logistics coordinator creates a supply chain
(2.) Read the offered services from the shipping company
(3.) Logistics coordinator creates a service request
(4.) The shipping company creates an offer
(5 .)Logistics coordinator agrees to that offer
(6.) The shipping company starts a shipment
(7.) Logistics coordinator updates the supply chain with info from the agreement
(8.) Logistics coordinator updates the shipping monitor with info from the shipment
(9.) The shipping monitor monitors the shipment

Resource Oriented Model

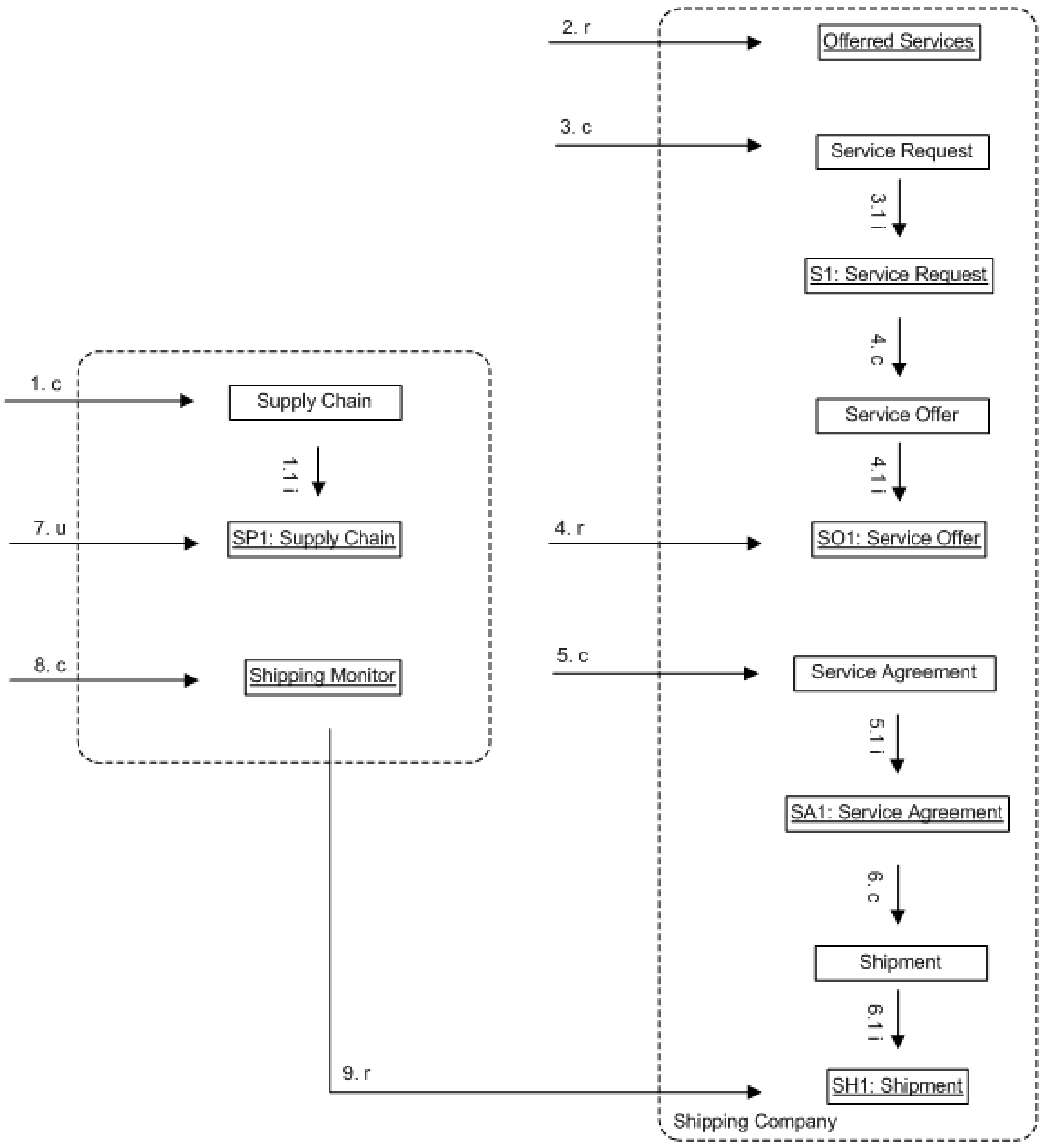

Figure 11 RO Model of B4

Modelling Issues

1. Structural relationships are not modelled

2. There is more than 1 shipping company and the process is the same, so it is iterative and is similar to M2. But this is not shown in the model
3. When the update, changes the structural relationships, what happens e.g. (7.u)

Cloud Computing

C1: NYT Times Machine

The cloud computing scenario we chose is the New York Times project called TimesMachine, which is discussed in [14], it aims to provide access to issues dating back to 1851, adding up to 11 million articles.

> *The technical team wanted to generate the PDF files from TIFF images. The generation was done based on request; however this solution would not work for high traffic. The team decided to generate all the PDF files and serve them on request. The size of TIFF files was 4 Terabytes. So they used Amazon's Elastic Compute Cloud (EC2) and Simple Storage Service (S3). The TIFF files were uploaded to S3 and they started a Hadoop cluster of 100 customized EC2 Amazon Machine Images. They transferred the conversion application. That resulted in the conversion to PDFs and storing the results to S3 taking 36 hours only.*

Infrastructural and functional requirements

1. **Configuration of Virtual Machines -** In this scenario the Amazon Machine Images (AMI) were configured to form a Hadoop cluster. This can be done through a web-based control panel or through Web Services. EC2 offers a SOAP interface and a query interface.

2. **Transferring large amounts of data to and from the servers -** This implies the need for reliable, efficient and secure data transfer. This is discussed in the following 3 points.

3. **The data is owned and manipulated by the client -** In contrast to mashups where the client requests the data, here clients request resources to manipulate their data.

4. **The client transfers the job/application to the servers -** In this scenario the client uploads to the cloud the application that manipulates the data.

5. **Multitenancy -** This means that the services and resources are used by multiple clients other than the New York Times and this implies a stronger need for security and for resource virtualisation.

6. **Batch processing -** Interaction with the server does not need to happen during the processing

Non-functional requirements

1. **Service Level Agreements -** There is no formal specification for the agreement, the SLA is a webpage. Therefore, the negotiation of SLA not automated.

2. **Reliability -** This should be based on the SLA and include
   a. The availability of services
   b. The recoverability of data and applications

   Since it is built on a business model what are the penalties in the case the reliability criteria are not met.

3. **Security -** The security involves:
   a. The authentication and authorisation of the service consumer in this case the technical team at The New York Times
   b. The encryption of the communication to guarantee confidentiality
   c. The encryption of the data and applications on the client which are owned by the clients to ensure that no one else can access them

4. **Monitoring -** Amazon offers a web console, command line tools, and a Web API (Web Service) to monitor the instances.

Scenario Breakdown

The New York Times project scenario TimesMachine [14] is decomposed into the following steps:

(1.) Create the data items, upload the images
(2.) Create a Hadoop Cluster
(3.) Create an application and upload the converter
(4.) The application returns the results
(5.) The client reads the results

Resource Oriented Model

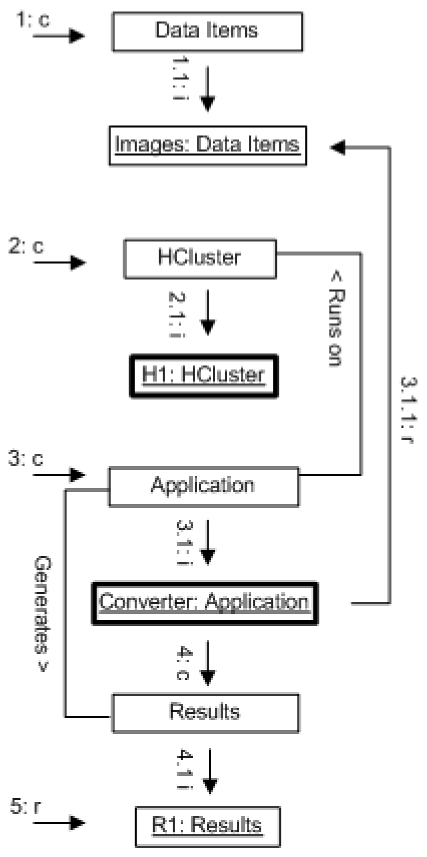

Figure 12 NYT Cloud Computing RO Model

The client sends the representation of the resource when creating or updating it, the client receives a resource representation when it reads a resource.

C2: Major League Baseball MLB Website's Chat System

Another scenario that was mentioned in [14], the MLB Advanced Media a company the develops and maintains the MLB websites wanted to add a chat service.
The technical team faced the problem that this chat service has to be up and running at a very short notice, there was no time to buy and set up new equipment. So they decided to use machines from Joynet a cloud computing provider, the machines acquired were used to test and launch the new product. At the development stage they needed 10 virtual machines and 20 for the chat clusters. When they launched the chat system they needed extra RAM for the machines, when the playoff and World Series started they needed extra machines with extra RAM and processing power. When the season ended they could scale down on the resources required.

Infrastructural and Functional Requirements

The requirements are similar to C1 however it differs in some technical issues

1. Flexible Scalability
   The resources utilised efficiently, acquired when needed or released otherwise
2. Standards used
   There is no Web API (Web Service) interface to Joynet services
3. Used as hosting server
   The scenario described here is more like a hosting server than cloud computing.

Non-functional Requirements

Identical to C1s requirements

Scenario Breakdown

(1.) Create Machine instances
(2.) Increase the number of machines and increase their RAM
(3.) Install "Create" the chat system
(4.) Run the Chat system
(5.) Increase the RAM in the machines
(6.) Scale down the machines

Resource Oriented Model

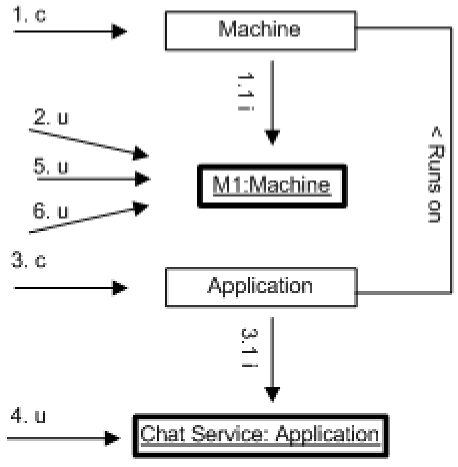

Figure 13 RO Model of C2

Modelling Issues

1. There are multiple machines, should they be considered a collection? And how will the application be modelled in that case

C3: Colorado State University using Google Apps

In [15] the author discusses Colorado State University's use of Google Apps such as Google Mail, Google Calendar, and Google Talk, Google Docs, Google Sites and Google Video.

In 2009, Colorado State University (CSU) used Google Apps as an e-mail hosting solution for its undergraduate students. Google Apps Education Edition, is free for colleges and universities. CSU wanted to replace their old system with an outsourced e-mail and collaboration solution. The important issues were: cost, reliability and the scope of services. Google Apps was selected because the main it offered e-mail, calendar, and personal web site services for students. Moreover the interoperability between those applications was a plus. This has increased students collaboration and communication. The faculty and staff are now moving their accounts to use the suite because of its potential.

Infrastructural and Functional Requirements

The requirements are similar to C2 however it differs in

1. Software as a service

Instead of acquiring software solutions the university used of Google Apps.

2. The client does not transfer applications to the server

The client uses the services as applications existing on their systems.

Non-functional Requirements

Identical to C1s requirements

Scenario Breakdown

(1.) Create a user account
(2.) Pay for the service
(3.) The Apps are created for this account

Resource Oriented Model

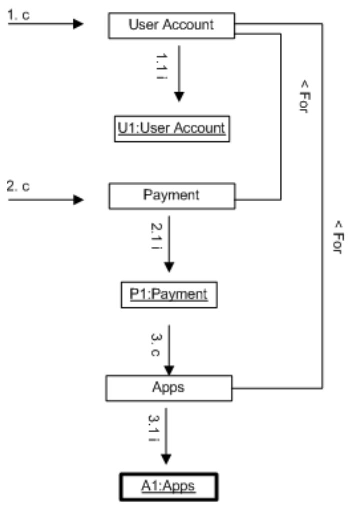

Figure 14 RO Model of C3

Modelling Issues

1. Is there an application specification resource or is it part of the user account?
2. Can I model the payment in more detail? What is the granularity of the modelling
3. Alternative paths?
4. In this scenario I assumed there would be a payment, however in Google Apps, there are different options, depending on the account type.

C4: LingoSpot a business built using Google App Engine

LingoSpot is one of the case studies mentioned in Google App Engine's documentation[1]
Lingospot provides services for online publishers to help readers discover more of their content, including virally-distributed widgets for related videos and articles, as well as smart discovery links within context. We use App Engine to scale our services to Web audiences limitlessly, ranging from a million+ users in 30 minutes at large sites, to supporting hundreds of smaller sites that have installed our viral widgets, without worrying an iota about provisioning capacity for the traffic and growth.
Google App Engine enables users to run programs written in Python or Java, it also offers APIs to access datastore, Google Accounts, URL fetch, Google Maps, and email services. It offers a Web-based Administration Console to manage applications.

Infrastructural and Functional Requirements

1. Platform as a service

LingoSpot used Google Apps Engine as a development and hosting platform for its application

2. Dynamic Scalability

The system autonomously responds to the peaks on demand.

Non-functional Requirements

[1] Google App Engine, App Engine Developer Profiles, http://code.google.com/appengine/casestudies.html

Identical to C1s requirements

Scenario Breakdown

(1.) Create a user account
(2.) Read the SDK
(3.) Upload the application
(4.) Run the application

Resource Oriented Model

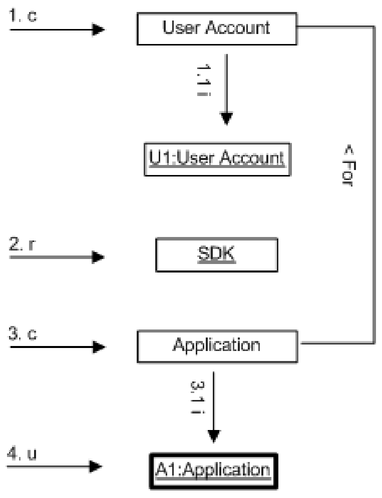

Figure 15 RO Model of C4

Grid Computing

G1: NEESgrid

NEES is an NSF funded project to build a virtual laboratory for earthquake engineers. Using grid technologies it enables remote access and control to observational sensors, experimental data, computational resources, and earthquake engineering control systems such as shake tables, reaction walls, and robots. NEESgrid also enables access to collaboration tools. [16]

Earthquake engineers wanted to study the effect of an earthquake on different types of substances and structures, these different structures and their shake tables are distributed across a number of labs, the aim was to coordinate these experiments with computer simulations. So the (Multi-site Online Simulation Test) MOST was devised to test and illustrate this capability using the NEESgrid system. MOST coupled physical experiments testing the effect of an earthquake on the interior of a multi-story building at 3 different sites each testing a part of the structure. MOST linked the physical experiments *at the University of Illinois at Urbana-Champaign (UIUC) and at the University of Colorado, Boulder (CU) with a numerical simulation at National Centre for Supercomputing Applications (NCSA). A simulation coordinator coordinates the overall experiment. [17]*

Infrastructural and functional requirements

1. Remote access to instruments - Services can be interfaces to instruments; in this case lab instruments such as shake tables.

2. Notifications - Running services send notifications to the clients or to the service/job scheduler.

3. Batch Processing - When a service or job is run, there is no need for the client to interact and results are delivered when it stops.

4. Coordination between running services - The services communicate to ensure correct synchronisation.

5. Negotiation - It involves interactions between the client and the server to ensure compliance between the client's requirement and the server's policies.

6. Support of sending and receiving large volumes of data - Large volumes of data are being transferred between different services requiring reliable, efficient, and secure transfer.

7. Service Scheduling - Services are invoked and controlled by schedulers in this case the Experiment Coordinator is controlling several experiment executions.

Non-functional requirements

1. Security - The security involves:

a. The authentication and authorisation of the researchers and scientists to protect sensitive data and applications

b. The encryption of the messages and transferred data to guarantee confidentiality

2. Monitoring - This is needed to ensure that the different components are functioning

3. Reliability - Reliable data transfer and service execution, no delays, interruptions or outages.

The main reason of analysing these scenarios was to inform design decisions when developing EXPRESS by gathering requirements from real scenarios. From this analysis there is an evident need to represent the requirements in unified way which will map easily to EXPRESS and this will be presented in the next chapter.

<u>Scenario Breakdown</u>

The NEESgrid scenario [16] discussed, consist*s* of the following steps:

(1.) Create experiments and the simulation

(2.) Create an experiment coordinator

(3.) The coordinator starts the experiments

(4.) The coordinator retrieves experiment results

(5.) The coordinator reads the results

(6.) The coordinator aggregates the results

(7.) The results are read

<u>Resource Oriented Model</u>

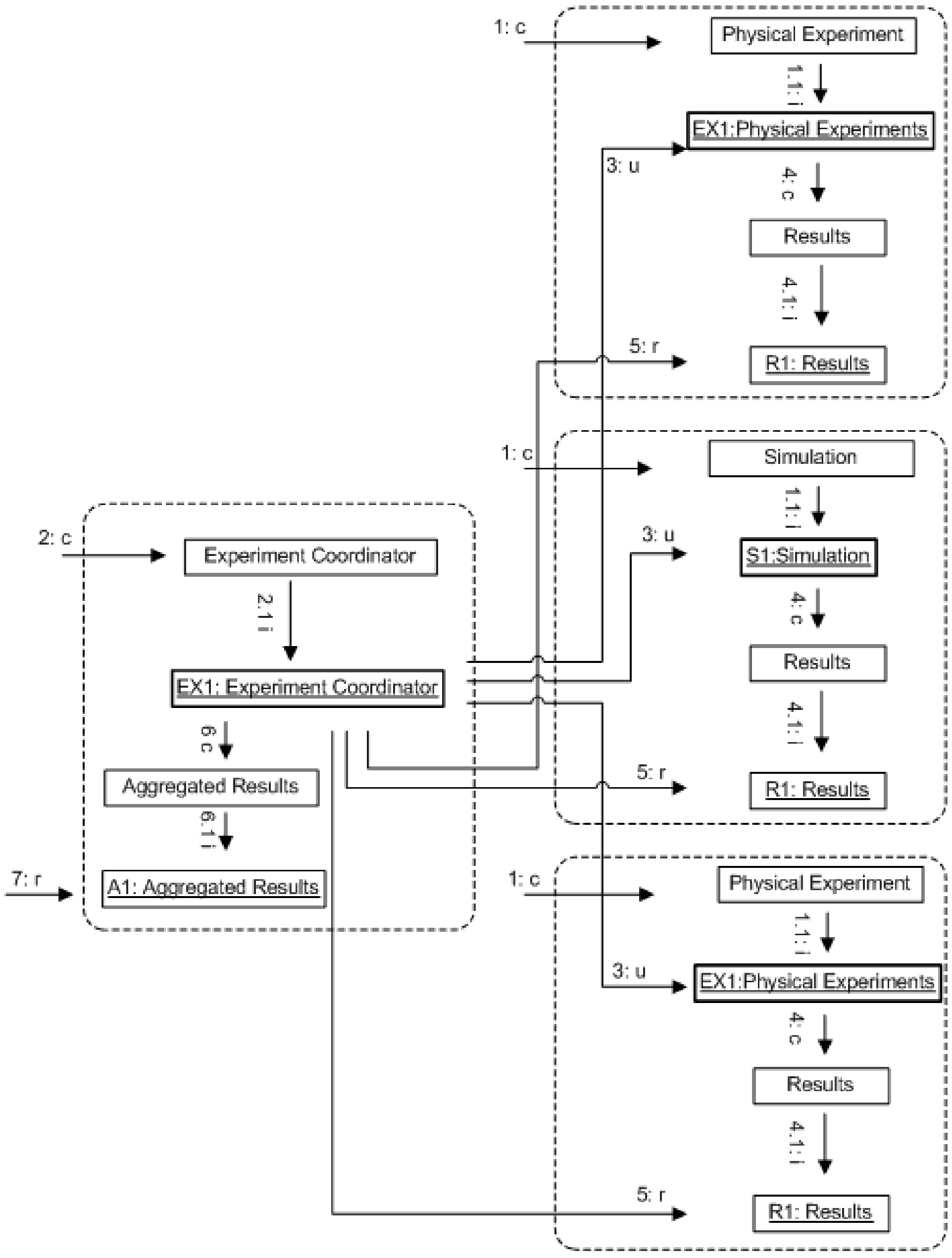

Figure 16 NEESgrid Experiment RO Model

Due to the complex nature of the NEESgrid scenario, and the limited space structural links between resources were not modelled.

G2: Distributed Aircraft Maintenance Environment

The DAME (Distributed Aircraft Maintenance Environment) project [18], is a Grid enabled system for aeroengine faults diagnosis and prognosis. The aim of the project is to use Grid technology to manage and analyse the vast amounts of data to diagnose existing anomalies and predict potential problems in aircraft engines. [19] states the challenges for DAME which are: the huge amount of data captured by the monitoring tool, the need for advanced pattern matching and data mining of the captured and historical data, the requirement of collaboration from diverse actors, the heterogeneity and distributiveness of the data assets and tools.

Work on DAME is currently further researched in BROADEN (Business Resource Optimisation for Aftermarket and Design Engineering on Networks) [20] which investigate the use of SOA techniques to achieve their goals. The main usage scenarios for DAME are:

There is a QUICK monitoring service installed on the aircrafts. This service captures the engines monitoring data. QUICK can produce up to 1 Gigabyte of data for each engine, an aircraft can have two or more engines; this can scale to many Terabytes each year for a fleet. Downloading and storing this amount of data efficiently requires a huge number of distributed repositories at different airports and these repositories must be available for the health monitoring of the engines. DAME's Engine Data Service is responsible for the downloading and storage of that data. The scenario mentioned in [21] illustrates the challenge.

> *"Heathrow, with its two runways, is authorized to handle a maximum of 36 landings per hour. Let us assume that on average half of the aircraft landing at Heathrow have four engines and the remaining half have two engines. In future, if each engine downloads around 1 GB of data per flight, the system at Heathrow must be capable of dealing with a typical throughput of around 100 GB of raw engine data per hour, all of which must be processed and stored. The data storage requirement alone for an operational day is, therefore, around 1 TB, with subsequent processing generating yet more data."*

Due to the vast amounts of data the choice for DAME was to be highly distributed, having the airports as the units of distribution. The monitoring data from an aeroplane arriving at an airport is stored at that airport. Therefore the search queries are distributed across airport nodes, where each node deals with the data it stores. This means that data relating to one engine is found in the different airports it landed in. To make DAME work each airport node has a data repository, pattern matching service, and a data catalogue.

An engine specialist wants to analyse a particular engine's data. The specialist provides the engine's identifiers; the system submits it to a global catalogue, which returns a handle to the data in the repository and also provides access to a pattern matching control (PMC) service which can distribute the search process across different nodes. The specialist searches for a feature in the engine data; the PMC becomes the master node and distributes the query to the other nodes. The search is performed in parallel; the PMC collects the results and returns them to the specialist.

The requirements are similar to G1 however it differs in the following issues

<u>Infrastructural and Functional Requirements</u>

1. Clients and Servers are controlled and managed by the same entity

Although DAME is implemented on a grid infrastructure, all the different components belong to the same entity.

2. Service Brokering

There is a service broker which forwards services to different machines (servers/nodes), in this scenario the PMC.

Non-functional Requirements

1. Support of sending and receiving large volumes of data

Large volumes of data are being transferred between different services requiring reliable, efficient, and secure transfer.

Scenario Breakdown

(1.) Downloading the engine monitoring data
(2.) Copying it to the nodes
(3.) A client reads the Global Catalogue
(4.) Sends a query to the node, the node distributes the query
 (4.1) The query is run on the data
 (4.2) Reads the results
 (4.3) Aggregates the results
(5.) The client reads the results

Resource Oriented Model

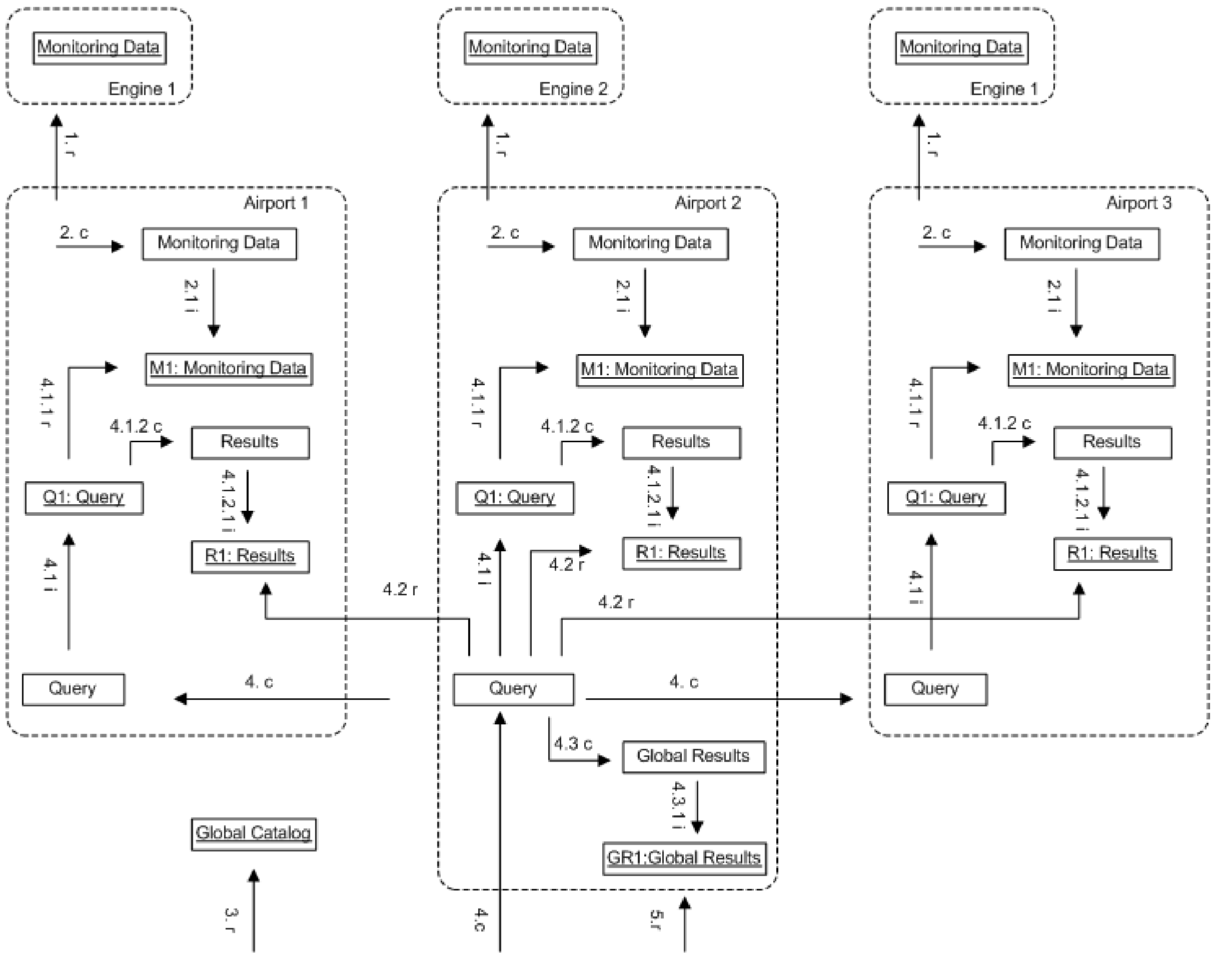

Figure 17 RO Model of G2

Modelling Issues

1. Who is doing (1.) & (2.), the node?
2. The multilevel bullets get confusing, especially if it indicates the timing
3. Who distributes the query is it the “Query” factory or should it be “Query 1”
4. Structural relationships are not shown in the model

G3: Virtual Screening with Desktop Grids

[22] Entropia is an architecture for desktop grids. Desktop grids utilise the idle commodity computing resources (desktops) to perform highly distributed and computing intensive tasks. A binary virtual machine is installed on each desktop. These communicate with a job manager and resource scheduler to receive jobs, execute them, and return results. Desktop grids are effective when there is high need for parallel processing power and there is no need for communication between nodes during processing or the communication is minimal. [22] describes “Virtual Screening” as one of the scenarios that make use of desktop grids:

In virtual screening, for drug discovery, a vast number of potential drug molecules are tested ranging from hundreds of thousands to millions. The aim is to discover if these drugs affect the activity of a studied protein. Testing involves a process called docking that assesses the binding affinity of the test molecule to a specific place on a protein. Each potential molecule can be evaluated independently making the process suitable for desktop grids. The results are binding scores.

So a scenario based on that would be: an end-user submits a computation to the Job Manager for example evaluating 50000 potential molecules. The Job Manager divides the computation to independent subjobs, in this scenario evaluating every five molecules together resulting 10000 subjobs. The subjobs are submitted to the Subjob Scheduler. Any available resources are periodically reported to the Node Manager that informs the Subjob Scheduler. Results of the subjobs are sent to the Job Manager then handed back to the end-user.

Infrastructural and Functional Requirements

1. Virtual Machines installed on clients/participants

For the desktop grid to work, virtual machines need to be installed on the nodes or desktops forming the grid computational resources.

2. Job Management

Managing breaking down the jobs into independent sub-jobs that are assigned to nodes, then assembling the results and returning them to the client.

3. Job Scheduling

The scheduling involves having knowledge of the numbers and sizes of tasks/jobs and the availability of resources. The VMs on the nodes informs the scheduler of the availability

4. There are three entities in this scenario
   a. Desktop grid service provider: in this scenario Entropia
   b. Nodes/participants: the desktops which become grid resources after installing the VMs
   c. Client: who has a computationally intensive task to be run

Non-functional Requirements

1. Security

In addition to the security issues mentioned in G1, another security measure is unobtrusiveness, meaning that the virtual machines and any jobs running on them do not harm or access confidential data or applications on the nodes they executed on.

2. Tolerance of failure

In this scenario, tasks are being submitted to desktops, which are volatile and it is likely that they could be switched off or cut off the network.

Scenario Breakdown

(1.) Create VMs on nodes
(2.) Create the Job
(3.) Submit the job to the Job Manager

(4.) The Job Manager splits the Job into subjobs

(5.) The Job Scheduler reads the subjobs
(6.) The Job Scheduler sends them to the nodes
(7.) The subjobs have results
(8.) The Job Manager reads the results
(9.) Aggregates the results
(10.) The client reads the results

Resource Oriented Model

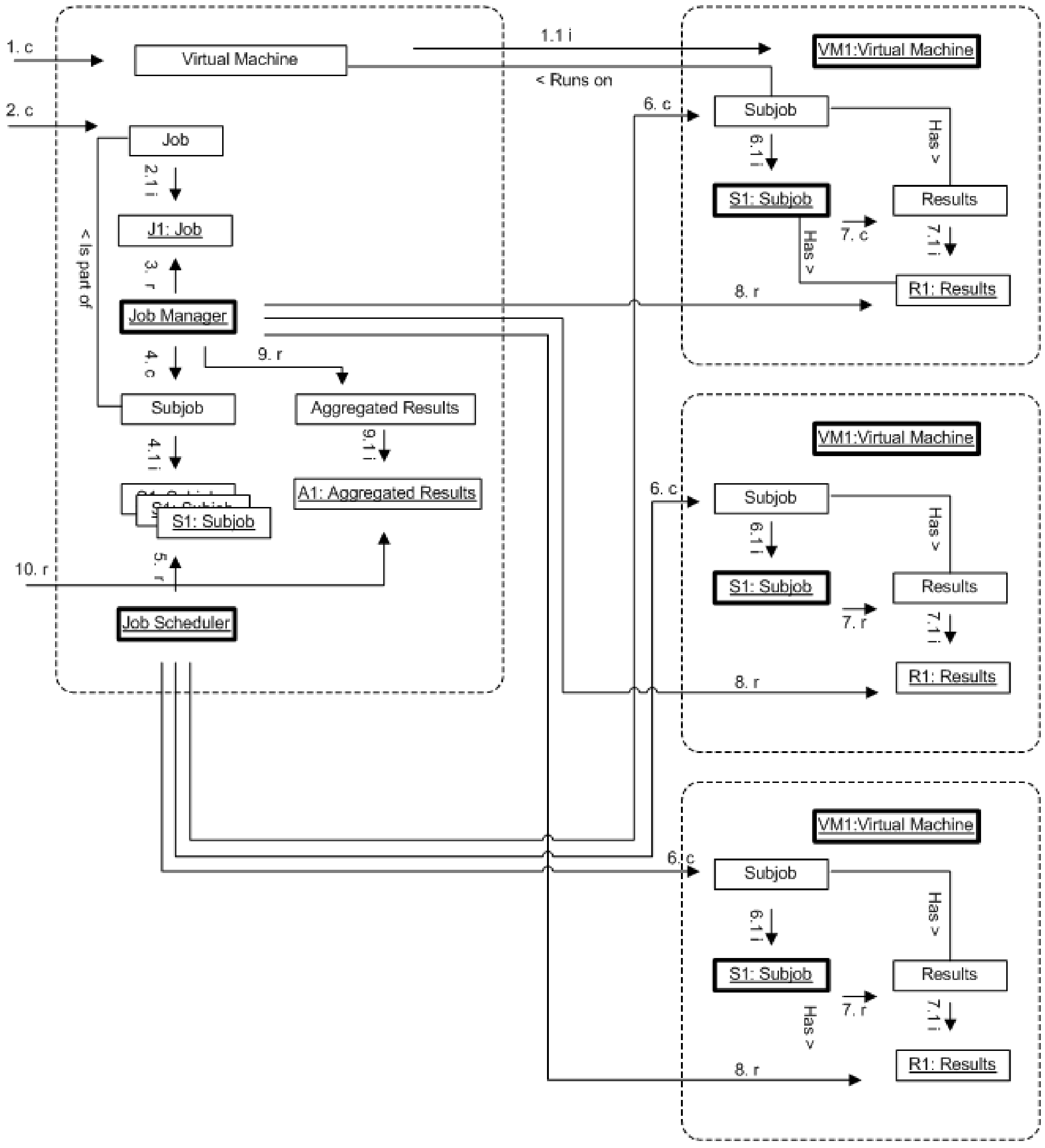

Figure 18 RO Model of G3

Modelling Issues

1. There are notifications when processing the jobs end, not modelled
2. The monitoring is not modelled

G4: CombeChem testbed on the Grid

The CombeChem project [23] developed a testbed to combine structure data sources and property data sources using the grid technologies to create a knowledge sharing environment. The grid infrastructure enriches laboratory devices and supports provenance and automation techniques.

As part of the CombeChem project Smart Lab was developed, it is intended to aid chemists during the different stages of an experiment, i.e. planning the experiment, performing the experiment, and analysing the results. The following scenario of using Smart Lab is built upon the description of the Smart Lab in [24].

A chemist uses the tablet PC to plan an experiment, gets it authorised by his/her supervisor. After the plan is authorised, the chemist follows it through to perform the experiment, during the experiment the chemist can observe and make notes that will be stored with experiment process. Moreover sensors and devices in the lab will store observations related to the experiment while it is being executed. After the experiment is performed results are recorded.

The requirements are identical to the ones in scenario G1 and G2 however there are others:

<u>Infrastructural and Functional Requirements</u>

1. Workflow support
The different processes that are executed can be coordinated and saved as workflows, so new workflows can be generated from them by changing processes or parameters.

2. Provenance Maintenance
The workflows provide means to link results to the steps they were generated from, thus providing a trial record and a method to reproduce the results.

<u>Scenario Breakdown</u>

(1.) The chemist creates the plan
(2.) The chemist creates the experiment process that is based on the plan
(3.) The process is updated by sensors and the chemists observations
(4.) The chemist can retrieve the process which contains all the information about the process

<u>Resource Oriented Model</u>

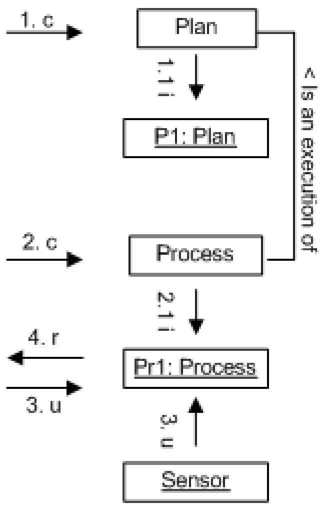

Figure 19 RO Model of G4

<u>Modelling Issues</u>

The sequence numbering (3.) does it happen at the same time, what if multiple requests are happening. Alternative scenarios

References

[1] P. Donnelly. (2010, 26/02/2010). *Yahoo Finance Stock Quote Watch List Feed*. Available: http://pipes.yahoo.com/31337/watchlist

[2] R. J. Ennals and M. N. Garofalakis, "MashMaker: mashups for the masses," in *the 2007 ACM SIGMOD international conference on Management of data*, Beijing, China, 2007, pp. 1116-1118.

[3] R. Ennals*, et al.*, "Intel Mash Maker: Join the web," *SIGMOD Record,* vol. 36, pp. 27-33, Dec 2007.

[4] B. Biornstad and C. Pautasso, "Let It Flow: Building Mashups with Data Processing Pipelines," *Service-Oriented Computing - Icsoc 2007 Workshops,* vol. 4907, pp. 15-28, 2009.

[5] A. Jhingran, "Enterprise information mashups: integrating information, simply," in *the 32nd international conference on Very large data bases*, Seoul, Korea, 2006, pp. 3-4.

[6] C. University, "Introducing SOA at City University, London," October 2008 2008.

[7] R. Miller and R. Cherinka, "Engineering a Complex Information Enterprise: A Case Study Architecting the Department of Defense Hourglass," in *International Conference on Enterprise Information Systems and Web Technologies, EISWT-07*, Orlando, Florida, USA,, 2007.

[8] J. Calladine, "Giving legs to the legacy - Web Services integration within the enterprise," *BT Technology Journal,* vol. 22, pp. 87-98, 2004.

[9] C. Petrie, "Practical Web Services," *IEEE Internet Computing,* vol. 13, pp. 93-96, Nov-Dec 2009.

[10] G. Decker and M. Weske, "Behavioral consistency for B2B process integration," *Advanced Information Systems Engineering, Proceedings,* vol. 4495, pp. 81-95, 2007.

[11] O. Zimmermann*, et al.*, "Service-oriented architecture and business process choreography in an order management scenario: rationale, concepts, lessons learned," in *Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, San Diego, CA, USA, 2005, pp. 301-312.

[12] M. L. Brodie, "The B2B e-commerce revolution: Convergence, chaos, and holistic computing," *Information Systems Engineering,* pp. 15-36, 2000.

[13] C. Preist*, et al.*, "Automated business-to-business integration of a logistics supply chain using semantic web services technology," *Semantic Web - Iswc 2005, Proceedings,* vol. 3729, pp. 987-1001, 2005.

[14] M. Klems*, et al.*, "Do Clouds Compute? A Framework for Estimating the Value of Cloud Computing," *Designing E-Business Systems,* vol. 22, pp. 110-123, 2009.

[15] D. R. Herrick, "Google this!: using Google apps for collaboration and productivity," in *Proceedings of the ACM SIGUCCS fall conference on User services conference*, St. Louis, Missouri, USA, 2009, pp. 55-64.

[16] S. Gullapalli*, et al.*, "Showcasing the features and capabilities of NEESgrid: A grid based system for the earthquake engineering domain," in *the 13th IEEE International Symposium on High Performance Distributed Computing*, Honolulu, Hawaii USA, 2004, pp. 268-269.

[17] L. Pearlman*, et al.*, "Distributed hybrid earthquake engineering experiments: Experiences with a ground-shaking grid appllication," in *the 13th IEEE International Symposium on High Performance Distributed Computing*, 2004, pp. 14-23.

[18] T. Jackson*, et al.*, "Delivering a grid-enabled distributed aircraft maintenance environment (DAME)," in *the UK e-Science All Hands Meeting*, Nottingham, UK, 2003.

[19] T. Jackson*, et al.*, "Distributed health monitoring for aero-engines on the GRID: DAME," in *IEEE Aerospace Conference 2005*, 2005, pp. 3738-3747.

[20] T. Jackson*, et al.*, "A virtual organisation deployed on a service orientated architecture for distributed data mining applications," in *Grid-Based Problem Solving Environments*. vol. 239, ed, 2007, pp. 155-170.

[21] J. Austin*, et al.*, "DAME: Searching large data sets within a grid-enabled engineering application," *Proceedings of the IEEE,* vol. 93, pp. 496-509, Mar 2005.

[22] A. Chien*, et al.*, "Entropia: architecture and performance of an enterprise desktop grid system," *Journal of Parallel and Distributed Computing,* vol. 63, pp. 597-610, May 2003.

[23] J. G. Frey*, et al.*, "Combinatorial Chemistry and the Grid," in *Grid Computing: Making the Global Infrastructure a Reality*, F.Berman*, et al.*, Eds., ed: Wiley, 2004.

[24] K. R. Taylor*, et al.*, "The Semantic Grid and chemistry: Experiences with CombeChem," *Journal of Web Semantics,* vol. 4, pp. 84-101, Jun 2006.