WINK: a Web-based System for Collaborative Project Management in Virtual Enterprises

S. Bergamaschi, G. Gelati, F. Guerra, M. Vincini
Università degli Studi di Modena e Reggio Emilia

Abstract—**The increasing of globalization and flexibility required to the companies has generated, in the last decade, new issues, related to the managing of large scale projects within geographically distributed networks and to the cooperation of enterprises. ICT support systems are required to allow enterprises to share information, guarantee data-consistency and establish synchronized and collaborative processes.**

In this paper we present a collaborative project management system that integrates data coming from aerospace industries with two main goals: avoiding inconsistencies generated by updates at the sources' level and minimizing data replications. The proposed system is composed of a collaborative project management component supported by a web interface, a multi-agent data integration component, which supports information sharing and querying, and SOAP enabled web-services which ensure the whole interoperability of the software components.

The system was developed by the University of Modena and Reggio Emilia, Gruppo Formula S.p.A. and Alenia Spazio S.p.A. within the EU WINK Project (Web-linked Integration of Network based Knowledge - IST-2000-28221).

I. INTRODUCTION

The increasing of globalization and flexibility required to the companies has generated, in the last decade, new issues, related to the managing of large scale projects within geographically distributed networks and to the cooperation of enterprises. ICT support systems are required to allow enterprises to share information, guarantee data-consistency and to establish synchronized and collaborative processes.

In particular, management issues related to the aerospace industry, with specific regard to the production of scientific satellites and in-orbit infrastructures, are very specific even if compared to the traditional One-of-a-Kind Production models. Many critical factors are combined together: absolute reliability of materials, components, equipments and final assembled outputs; unique production processes and products for unique aims; huge investments and high risks related to the ROI (Return On Investment) factor and strict time constraints.

As for the reliability and uniqueness, they have led to the development of sophisticated and accurate procedures for requirements analysis, verification and testing. All of them are particularly detailed and require the accurate management of an enormous quantity of technical documentation.

The high quality of final products can only be assured by acquiring components from highly specialised companies; therefore, it is very rare that the entire space project (called space program) is carried out within the scope of a single organisation, but more often the prime contractor (typically a large company with adequate know-how) outsources specific components or activities to smaller firms through various forms of subcontracting. In these scenarios, the relations between main contractors and subcontractors are strategic and must be supported by adequate collaboration practices.

Finally, strict time constraints and huge investments require that all the activities of the entire product life-cycle (design, manufacturing, verification and testing, launch) be planned and monitored precisely, by adopting project management tools capable of taking into account several factors, like resource and product availability, budget and time constraints, personnel skills and availability, and so forth.

Traditionally, all these issues have been dealt by devoted information systems capable of managing one feature at a time, nevertheless requiring integration among them that was hard to automatically obtain. Nowadays, the integration of the diverse management tools and information sources is necessary for several reasons, of which the fast technology evolution is the first one: in fact, the overall product life-cycle has shortened and quite often, during a space program, the time elapsing between the design and the launch phases is so long that some of the involved technologies become obsolete in the meantime. Secondly, new collaboration paradigms such as Collaborative Project Management, Supply Chain Management and Knowledge Management are definitely mature enough to support the overall process and must be accompanied by adequate information systems [1,2]. Finally, the availability on the market of new technologies (XML for the data exchange between different systems, SOAP for the interoperability between different software platforms, mobile agents for accessing remote systems resources) allows a more powerful and potentially easier interoperability than in the past.

In this paper we present a collaborative project management system that integrates information coming from a real world scenario in aerospace industries, offered by Alenia Spazio SpA, the Italian leader of aerospace industry, with two main goals: avoiding inconsistencies generated by updates at the sources' level and minimizing data replications. In particular the proposed semi-automatic integration methodology follows a semantic approach by using intelligent Description Logics-techniques [3], clustering techniques, an ODM-ODMG [4]

extended data model to represent extracted and integrated information [5] and a multi-agent mediator system to support distributed queries over the virtual integrated information representation [6].
The WINK (Web-linked Integration of Network based Knowledge) system is based on a three tier architecture and exploits integrated data coming from several data sources to provide the users with a set of tools which increase the capability of managing large projects. In particular, the client tier supports operations such as alert firing, activity scheduling, and project planning structures, …, providing a customized and integrated web interface. The business logic then includes a multi-agent data integration component, which supports information sharing and querying, and SOAP enabled web-services, which ensure the whole interoperability of the software components.

The paper is organized as follows: in Section 2 we give an overview of the scenario and the benefits provided by the system. In section 3 we describe the overall WINK architecture, then in section 4 and 5 the detailed aspects of the architecture are illustrated. In section 6 and 7, by means of a real world scenario, the 'system at work' is shown. Finally, in section 8 we sketch some conclusions and the future work.

II. Case Study and Expected Benefits

The life cycle of an Alenia space program (i.e. the plan related to design, manufacturing, assembling and launch of a scientific satellite or a International Space Station module) can last up to ten years. Among the scheduled processes, the Assembly, Integration and Verification phases are very critical in an Aerospace context due to the fact that many components and relative manufacturing and testing procedures are unique and high levels of quality must be guaranteed.
At Alenia Spazio S.p.A., the Assembly, Integration and Verification (AIV) Department is responsible for the supervision of the whole project life cycle. At the beginning of the project, the AIV manager builds a product tree according to the program requirements. The product tree can be then divided into sub-trees, each assigned to some external enterprise. An external enterprise will be considered a contractor or a sub-contractor depending on its responsibility and budget amount. The AIV Department analyses the project requirements and organizes them in a hierarchical tree where the lower levels typically contain the needed equipment, the intermediate levels represent the assembled components and the root level is the overall system perspective. In this way, the best verification procedures matching the requirements are defined. These activities are supported by different information systems (of different enterprises) and involve many complex and distributed processes:

- Project scheduling systems for Gantt definition;
- Project accounting systems for the definition of project costs, budget and final balance;
- Resource planning systems for personnel allocation capable of matching the right skills and the right activities according to the time and cost constraints;
- Requirement management systems usually rely on dedicated databases due to the complexity of the product and the high value of the materials;
- Document management system to manage Non Conformity Reports (NCRs) which are usually on dedicated databases;
- Supply Chain Management system.

Starting from the requirement tree, the best-practise procedures, project budget, goal and constraints, the AIV manager defines the so-called General Schedule Project (GS). The GS is then transformed into a Detailed Schedule, which holds a more operative function and usually has a short-medium term scope.
Due to strict time constraints, especially when launch date approaches, and the necessity in term of reliability for materials, components and assembled equipment, the AIV Department requires daily management meetings, where the accomplished verification activities are analyzed and a detailed plan for carrying out the next verification activities is formulated. The daily meeting, chaired by the AIV manager, represents an important step that reworks all the information coming from the various IT systems combined with the additional information of the various people involved.
This activity is very expensive in terms of employed resources and, to some extent, can lead to inefficiencies and ineffectiveness in the process itself.

The AIV activities have to take into account several vertical processes that are most likely managed with different information systems, therefore additional efforts have to be made in order to harmonize all data needed for activity monitoring. The WINK system tries to minimize this effort by providing the aforementioned semi-automatic data-integration toolkit.

From the AIV perspective, foreseen benefits of the system are the reduction of number and duration of activity verification meetings, re-scheduling simplification, employee travel reduction, performance monitoring improvement, resource allocation (with real-time visibility on intranet) improvement, reduction of testing time and costs related to AIV management.

III. Architecture

The WINK architecture, shown in Figure 1, is based on a three-tier model where the client tier makes available a Virtual Integrated Cockpit on which information is collected and presented as a customized web interface; the data tier manages the interactions with the data provided by the Enterprise Information Systems; and the business logic tier combines the capabilities of two separated modules, the Project Collaboration Portal and the Integration Framework. In particular, the first module supports the definition of business logic for monitoring, execution and planning of a project (resource management, non-conformities, alert, document organization and so on). The Integration Framework collects the data required by the implemented business processes in a very dynamic way, integrating information coming from heterogeneous and possibly distributed data sources.

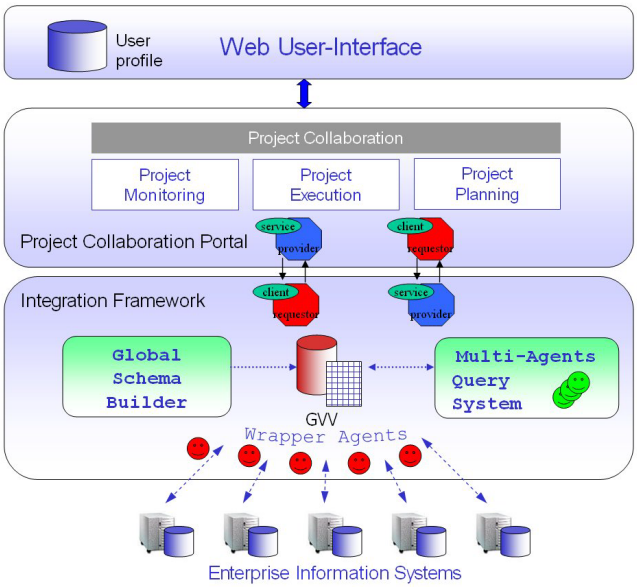

Figure 1: The WINK Architecture

This is achieved by exploiting MIKŞ an agent-based system [6], which allows a highly flexible and configurable data integration. In this way, the business logic tier of the WINK system is continuously fed by updated (and coherent) data for each of the implemented business processes.

In the following sections the Project Collaboration Portal and the Integration Framework modules are described in detail. Here we focus on the heterogeneity of the WINK tiers that are distributed on the net and have to communicate to each other. In particular, a communication managing system is needed within the business tier, where the Project Collaboration Portal and the Integration Framework reside on different systems and on different platforms. In our case, the first one has been implemented by Microsoft technology (asp pages and DCOM objects) while the second one is a JAVA compliant system that exploits agent and CORBA technologies for managing internal communication. In order to solve the interoperability issues, we chose to adopt web services, thanks to their flexibility and easy development features. A web service is understood as a software module deployed on an application server, making services available not depending on the kind of platforms, operating systems and programming languages. Technologically speaking, web services do not add any feature to the previously developed techniques (like CORBA or RPC) in distributing data management applications, but they provide an infrastructure based on W3C standards to connect systems that are more easily and less expensively implementable. This is achieved by using WSDL, the Web Service Description Language, a proposed standard that provides a model and an XML format to describe Web services. WSDL allows to separate the description of the abstract functionality offered by a service, from concrete details of a service description (e.g. "how" and "where" the functionality is offered) [7]. Web services allow applications to interact with each other by using SOAP, that provides the definition of the XML-based information which can be used for exchanging structured and typed information between peers in a decentralized, distributed environment [8].

Within the WINK system, the interoperability between the Project Collaboration Portal and the Information Framework is assured by a set of web services built on the SOAP protocol that guarantee the data flow.

IV. The Project Collaboration Portal

The Project Collaboration Portal (PCP) addresses issues related to decentralisation of project and production activities with the related concentration on the core business in the specific industrial sector of the One-of-a-kind Production (e.g.: industrial equipment, ship building, aerospace). These activities assure both final high quality and low overall logistics and production costs of the final products. In these distributed contexts, where the product life-cycle is characterized by activities that are not repeated, the only way to guarantee high levels of quality of service within distributed manufacturing processes is to adopt a strategy of *collaboration,* extending the Concurrent Engineering techniques to the entire network of partners (main contractor, subcontractors and suppliers), as shown in figure 2.

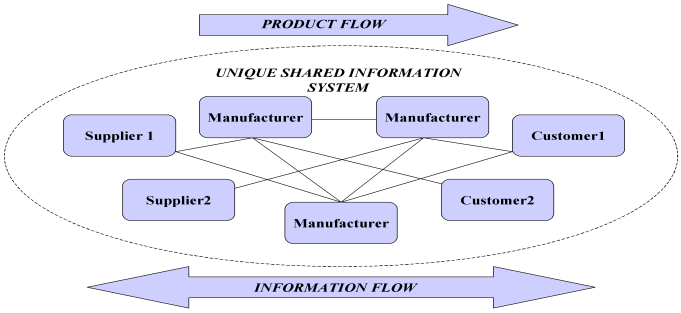

Figure 2: The Network Enterprise Model

This model implies not only a series of formal agreements among different partners but, from a technological perspective, it also implies the existence of a collective shared information system integrated with the different local legacy application to create a truly shared and collaborative workspace [9].

As depicted in the WINK architecture (figure 1), the PCP is composed of four modules: the Project Collaboration module, the Planning module, the Execution module, and the Monitoring module. The *Project Collaboration* module allows visibility of data presentations in aggregate and detailed views, searching, filtering and reports printing facilities and links between different data for each node or actor according to their visibility rights. A documental system has been developed, permitting a large number of documents related to each data object (Products, Bills Of Material, Project scheduling and so on) to be managed at a distributed level. Moreover a smart configurable workflow automation system has been developed to allow interactions between users in order to negotiate specific aspects (orders, activity or phase duration and so on) in all the project life cycle phases (planning, execution, monitoring).

The *Project Planning* module permits to define two transversal project structures called respectively *Extended*

Project Organisational Structure and *Activity Plan*. The *Extended Project Organisational Structure* describes the temporary, multi-site and multi-company hierarchical organization created to carry out a particular project, while the *Activity Plan* describes the project in terms of operational phases and activities. This module supports the management of activity plans directly inserted by the WINK user web-interface, as well as generated by other applications (such as MS Project and Windchill).

The *Project Execution* module allows tracking project steps in terms of consumed resources, exception management, and performance (time and costs) to identify variances from the plan, generating alerts if the consumption overcomes the budget, according to the rules defined for the WINK Alert System.

Finally the *Project Monitoring* module allows reporting all the relevant data information by means of OLAP functionalities and printable reports.

V. The Integration Framework

The Integration Framework consists of a web service architecture that encapsulates a multi-agent mediator-based system. The mediator provides an integrated access over the Enterprise Information System, exploiting the functionalities previously developed within the MOMIS [5] and MIKS [6] systems.

A. Integration Process

The proposed Integration Framework relies on a semantic approach based on the conceptual schema -or metadata- of the information sources, and on the Intelligent Integration of Information (I^3).

The methodology follows a GAV approach [10], whose result is a global schema which provides a reconciled, integrated and virtual view of the underlying sources, called Global Virtual View (GVV). The GVV is composed of a set of (global) classes that represent the information contained in the sources being used, together with the mappings establishing the connection among the elements of the global schema and those of the source schemas.

Within the framework, a common language ODL_I^3, which describes source schemas for integration purposes, is defined. ODL_I^3 is a subset of the corresponding ODL-ODMG language -according to the proposal for a standard mediator language developed by the I^3-POB working group, augmented by primitives to perform integration. In particular, ODL_I^3 can express inter- and intra-source intentional and extensional relationships among classes, mapping tables (to establish a connection between the global and the local view), integrity constraints and some further operators to handle heterogeneity.

The ODL_I^3 relationship types are the following:

- syn (synonym of) is a relationship defined between two terms ti and tj (where ti ≠ tj) that are synonyms in every involved source. For example, you can use ti and tj in every source to denote a single concept.
- bt (broader terms) is a relationship defined between two terms ti and tj, where ti has a broader, more general meaning than tj. bt relationships are not symmetric. The opposite of bt is nt (narrower terms).
- rt (related terms) is a relationship defined between two terms ti and tj that are generally used together in the same context in the considered sources.

To accomplish the integration process, the Global Schema Builder of the Integration Framework exploits a common ontology for the sources (the Common Thesaurus) generated using lexical knowledge derived from WordNet [11], schema derived relationships and integration knowledge inferred by exploiting Description Logics techniques.

Based on the relationships in the common ontology, affinity coefficients giving a measure of the level of matching among the concepts in the data sources are computed. Then, a threshold-based hierarchical clustering technique is used to classify concepts into groups of different levels of affinity. Finally, the designer selects (or unifies/slits) the clusters providing a unified view of the integrated domain and defines a mapping rules set for each cluster to express the relationships holding between the global abstraction of the cluster (global class) and the local representation.

The framework expresses the GVV using the ODL_I^3 language and may export all the integration results (i.e. the GVV and the Common Thesaurus) in XML in order to guarantee interoperability with other open integration systems.

B. A Multi-Agent Query System for supporting global query execution

The GVV gives users an integrated view over data that are scattered over different places and applications. By means of web services, an external application interacts with the Integration Framework by querying directly the global schema using a common query language (an extension of the OQL standard language). Similarly to other semantic approaches, the querying phase consists of three steps [12]:

- Semantic optimization
- query plan execution
- fusion of local, partial answers.

We have designed and implemented a Multi-Agent System (MAS) for supporting the whole phase of global query execution. The system has been built using the JADE environment (http://jade.cselt.it), a FIPA-compliant development tool (http://www.fipa.org). Agents perform activities for manipulating global queries to create queries at lower level of abstraction (local queries) that are hence executable on data sources. Local answers have then to be synthesized into a global answer. Notice that, while the integration process is essentially a one-way bottom-up information flow starting from the source contents and ending up with the generation of a GVV, the querying phase is a two-way process: top-down when users submit queries over the GVV, and bottom-up when local answer are made available and have to be merged to compose the global answer. Our

MAS reflects the nature of the querying process.

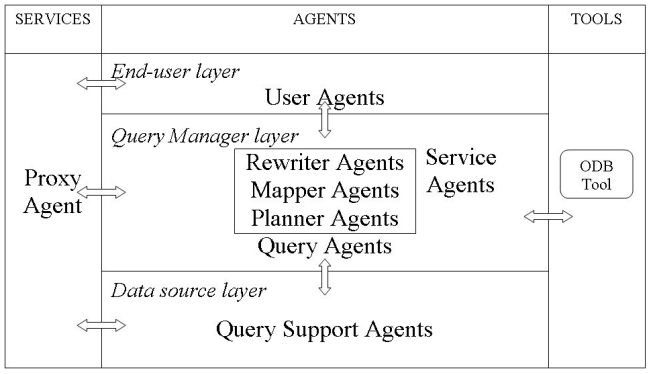

Figure 3: The Multi Agent Query System

Figure 3 illustrates the organization of agents. The agents that carry out global query decomposition and partial answer merging (globally called *Service Agents*) and the agents responsible for query execution at local level (Query Agents) are grouped in the Query Manager layer.

Service Agents can be divided into three different classes of cooperative agents: the *Rewriter Agents*, the *Mapper Agents* and the *Planner Agents*.

Rewriter Agents (RAs) operate on the assigned query by exploiting the semantic optimization techniques provided by ODB-Tools [13] in order to reduce the query access plan cost. The query is rewritten incorporating any possible restriction, which is not present in the global query but is logically implied by the GVV (class descriptions and integrity rules).

Mapper Agents (MAs) express the rewritten global query in terms of local schemas. Thus, a set of sub-queries for the local information sources is formulated. To this end, MAs dialogue with Proxy Agents that hold the knowledge about mapping tables and global and local schema. In order to obtain each local query, the mediator checks and translates every predicate in the where clause. The other important task performed by MAs is the rewriting of the original global query in terms of the local queries (as join query), in order to produce the final data answer.

Planner Agents (PAs) are charged to take the set (or subsets) of local queries and produce the executable query plan. The goal of PA is to establish how much parallelism and workload distribution is possible. Considering that queries are assigned to *Query Agents (QAs)* that move to local sources, creating a plan means trying to balance different factors:

- how many queries have to be assigned to each single QA
- which sources and in which order each QA has to follow in order to solve the assigned queries or to fuse partial results.

The choice of the number of query agents to be used can be determined by analyzing each query. In some cases, it is better to delegate the search to a single query agent, which performs a "trip" visiting each source site: it can start from the source that is supposed to reduce the further searches in the most significant way, then continue to visit source sites, performing queries on the basis of the already-found information. In other cases, sub-queries are likely to be quite independent, so it is better to delegate several query agents, one for each source site: in this way the searches are performed concurrently with a high degree of parallelism. This allow for decentralization of the computational workload due to collecting local answers and fuse them into the final global answer to be carried to the user.

QAs move to local sources where they pass the execution of one or more queries to Wrapper Agents (WAs). Moving to local sources implies a number of advantages. In particular, users can also query sources that do not have continuous connections: QAs moves to the source site when the connection is available, performs locally the search even if the connection is unstable or unavailable, and then returns as soon as the connection is available again. WAs afford translation services between the global query language and the native query language of the data source. This step is required to make queries executable by local information management system.

When the local answer is available, the corresponding QA has to map these data (whose structure follows the local schema) to the global schema. To do this, the QA interacts with PAs to know the set of mapping rules related to the source.

VI. Space Program Integration

The activities of the AIV manager require the AIV Manager is constantly kept up-to-date on diverse aspects of the projects he is managing, from personnel to materials and components, from costs to non-conformities. Starting from the requirements we identified by interviewing AIV managers in collaboration with the IT department of Alenia, we selected a set of sources that are necessary data to support AIV managers throughout their job. Due to the internal organisation of Alenia, the needed data sources have been created and managed by different units. Each unit has been managed data following different styles and criteria, resulting in an heterogeneous collection of information sources.

The selected set of sources to be integrated was:

- `Storage DB`: serves the logistic management of the AIV department. It stores information about material and equipment and the requests submitted to the storage unit and the subcontractors. It has been implemented using MS Access;
- `AIV DB`: stores data related to product trees, project requirements and project activities. This source is under the direct control of the AIV manager and has been implemented using Oracle 8i;
- `SAP DB`: this is a classical SAP system. The portion of data belonging to this source that is interesting in our application scenario is mainly related to the project organisation (cost centers, resources, workpackages) and supply management (order, billing, and other);

- NCR DB: collects data related to non-conformities and their impact on the project schedules. It has been implemented using Lotus Notes;
- WHALES: is the data source managed by the PCP module. Its structure is mainly application independent, with a few tuning paramenters. It is a data source we decided to create to materialize data on the WINK system related to specific project management functionalities not present within Alenia systems. It has been implemented using MS SqlServer.

The integration process has been carried out over 70 relations distributed in 5 data sources. In the following, we present our integration process by means of some relevant relationships examples obtained in the space program domain.

Schema-derived relationships
First, the schema-derived relationships holding at intra-schema level are automatically extracted by analysing each ODL_{I3} schema separately. These relationships are determined using the information on foreign keys holding between the relations of a schema.
As an example, we report a few relationships extracted from the AIVDB schema. A relation of the AIVDB schema is PRODUCT_TREE. It contains the data related to the product tree of a project. A product tree is identified by the field PT_ID. In the same schema there are other relations declaring a foreign key to the identifier of a product tree. Just to cite a few, relations that have this constraint are CI_PRODUCT, that stores the information on each item of the product tree, and CI_PHASE_DEFINITION, that stores the phases to be accomplished in order to realise the tree. We thus obtain the following RT relations:

1. AIVDB.CI_PRODUCT_TREE **RT** AIVDB.CI_PRODUCT
2. AIVDB.CI_PRODUCT_TREE **RT** AIVDB.CI_PHASE_DEFINITION

In the case we have the additional property of the attribute being the primary key of both relations, we have an ISA relationship identified with a BT/NT link. A BT relationship is extracted for the relations REQUIREMENT - that stores the requirement of each activity to be executed – and IMAGE_LINK – that stores the links to a technical document for each requirement (such as drawings) :

3. AIVDB.REQUIREMENT **BT** AIVDB.IMAGE_LINK

Lexical-derived inter-schema relationships
In this step, terminological and extensional relationships holding at intra-schema level are extracted by analysing ODL_{I3} schemas together. The extraction of these relationships is based upon the lexical relations holding between classes and attribute names. This is a kind of knowledge, which is not based on rules of a data definition language but derives from the names assigned by the integration designer. This is achieved during the annotation phase, when the designer assigns a meaningful word and meaning to each relation and attribute name. This phase is crucial in the integration process, as much attention has to focus in the selection of the meaning to be assigned to a name: this presupposes a correct and complete knowledge about the content of all schemas.
In our case, we annotated approximately 1400 terms and obtained 900 relationships. A few examples are:

4. WHALES.PHASE **SYN** AIVDB.CI_PHASE_DEFINITION
5. StorageDB.request **SYN** WHALES.MyPR
6. NCR.NCR.item **SYN** AIVDB.CI_PRODUCT.CI_ID
7. StorageDB.request.Program **SYN** SAP.ODA.PROGRAM

In 4) and 5) both the antecedent and subsequent elements are relations. Relationship 5) explicates for instance that requests of equipments stored in the request relation of the StorageDB schema are synonym of the requests stored in the MyPR relation of the WHALES schema.
In the next two relationships, both the antecedent and the subsequent elements are attributes. Thus, an item in a non-conformity stored in the NCR schema is synonym of an item of the product tree of the AIVDB schema. The same goes for the attribute Program which identifies the space program that a request in the StorageDB schema and an order in the SAP schema refers to.

All relations have been built starting from the annotated schema and exploiting the ODB-Tools inference engine.

Clustering and global mapping

Once the relationships among the classes of the schemas have been inferred, the integration process goes on with the clustering phase. During this phase, we identify classes that describe the semantically related information, grouping them in the same cluster. The level of semantic matching is measured by means of the affinity function [14].

In our test case, the Integration Framework automatically recognized twelve clusters. A cluster included from two to six classes, being the average three. Significantly, there clusters were built for personnel, resources, material orders, equipment requests, non-conformities, product tree, requirements, procedures and project documents.

As an example, let us take the cluster where all information concerning orders was grouped. The cluster (named ORDER) comprises six relations covering different aspects related to order management within the AIV Department. First, we find the relation ODA from the SAP schema, which stores very general information about an order (buyer, program, item, description). Then, we have two relations taken form the WHALES schema that store additional information such as request and delivery dates, quantity, and supplier. As order is intended here to be an item of the product tree, the cluster includes also three relations from the AIVDB source, reporting the description of the requirements related to the particular

item. All these data provide a comprehensive view of the concept of order as meant within the AIV Department.
Given its semantic relevance, the cluster was chosen to form a global class during the last stage of the integration process. Mappings are defined by means of a table where columns represent the set of the local classes, which belong to the cluster, and rows represent the global attributes (see Figure 3).

Global class	AIVDB			SAP	WHALES	
ORDER	DOCST_ DEFINITIO N	DOCS_ LINK	VER_DOC _LINK	ODA	MYORDER	SUPPLY ORDER
ORDERID				order	ordernumber	orderoid
PROJECT		ci_id	ci_id			projectitem code
MATERIAL				material	material_id	
REQUEST_ DATE						reqdate
STATUS				status		orderstatus
DELIVERY_ DATE				delivery date		
WORK PACKAGE			wp		supply	
DOC_ID	docs_id	docs_n um				
DOC_LINK	webpage					
REQUIREME NT			req_seq			

Figure 3. ORDER mapping table

VII. WINK INFORMATION FLOW

The typical usage scenario of the WINK system foresees the AIV Manager and other users operating the WINK web interface to view and manage project information. The first operation is the logon where the user specifies the node (that represents the user point of view for accessing and interacting with other nodes), the role (which is the organizational position he/she wants to impersonate for the current session), username and password. After having stated the logon credentials the WINK system enables the use of the proper functionalities and presents the personalized home page. For example figure 4 presents the WINK Personal Home Page for an AIV manager, who can see current alerts coming from relevant project events, currently ongoing workflow activities, a list of relevant links for easy access of the user's projects and frequently used functions.

In order to propose the entry point for any collaboration process in WINK, the main areas in the WINK Personal Home Page are the following:

- My alerts: contains the notification of relevant events occurred in the project to the actor of the system, regarding the project and position he chooses to select. The actor can have a look to the data that caused the alert, and can finally decide to get rid of it, by ignoring it.
- Open NCR: contains the list of currently open non-conformities that have to be solved. The actor can navigate the list and access documents that accompany the non-conformity generation.
- My Orders: contains the list of all order that have been submitted but no yet closed. The actor can thus monitor the execution of the orders he submitted or the orders for which an authorisation is required.
- My Requests: contains the list of internal equipment requests, reporting the status and tracking any change in the data related to them. The actor can thus know whether a requested instrument can be available on time and subsequently decide alternative actions or requests.
- My activities: contains the list of open negotiation that the logged on actor must consider, since he is requested for authorization or negotiation. The actor must follow the linked workflow interaction in order to comply with the negotiation activities he is involved in.
- My Projects: contains the list of the organizational positions that the logged on actor has in the moment of logging on. The actor can choose among the different projects and organizational positions he is in charge of. Whenever he selects another position, the home page reloads in order to present the above mentioned collaboration alerts and activities for the specific project and position.
- What's new: contains a series of static information that are common to the project network the user choose to log on.
- My Link: contains the preferred links (typically to external web sites or application) for the actor, regarding the chosen position.
- My Frequent Tasks: contains the most frequently used WINK function of the logged in organizational position, along with the workflow activities it is in charge to activate.

Many of these operations require the execution of queries in order to retrieve up-to-date data, to be subsequently processed. The analysis of the WINK system requirements brought to a classification of the query types according to two orthogonal dimensions. The first captures the design perspective, i.e. whether the query responds to explicit and well-known application requirements or is introduced by users for contingent needs. The second dimension concerns an operative perspective, i.e. the times a specific query has to be submitted. In addition, queries can be submitted either in response to explicit user's requests or as scheduled operations required to keep data up-to-date in an automatic fashion.

Combining the two dimensions, we have four kinds of queries:

- Designed and user-submitted queries: these are defined at design time to meet explicit application requirements and are executed only when the user explicitly calls an operation that relies on the query execution;
- Designed and scheduled: these are defined at design time to meet user requirements and consist in the automatic execution of queries on a regular basis (to materialize distributed data at scheduled time);
- User defined and user submitted: while operating the system, new queries can be composed and executed under explicit user requests;
- User defined and scheduled: while operating the system,

new requirements may emerge and determine the introduction of new queries to be scheduled on a regular basis. This type of query is important for designers when new application requirements are unfolded.

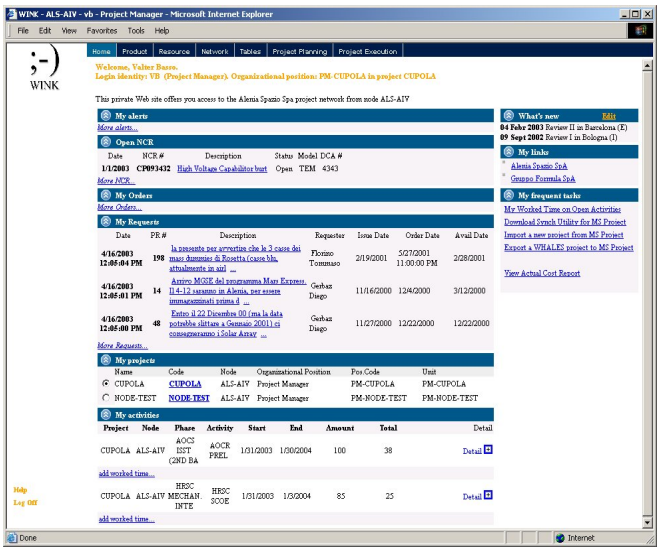

Figure 4. The WINK web user-interface

All these kind of queries are executed by the WINK system by exploiting the multi-agent query system included within the Integration Framework. The whole query processing for a single "designed and user-submitted query" is shown in figure 5. First, the user with the right grant composes the query by means of a parametric dynamic web page (realized by Microsoft ASP pages) of the web user-interface. For example, a daily task of the AIV manager consists in checking the opened (or closed) material orders referred to its managed projects and requested in a specific period: this implies to know the order number and material, the date it was requested, the expected date of delivery, the workpackage, the requirement number and documents it was associated to.

This request invokes a specific function of the business logic with run-time parameters (for example 'opened order of last month): the business logic combines this parameters with the user profile information (for example the managed project by the user, let us suppose 'CUPOLA'), produces the global query over the GVV and requests the query execution to the Integration Framework. In the example, the globalquery over the GVV is the following:

Q: Select ORDERID, MATERIAL, DELIVERYDATE,
WORKPACKAGE, REQUIREMENT, DOC_LINK
from ORDER
where STATUS-='opened'
and PROJECT = 'CUPOLA'
and REQUESTDATE > Date() - 30

The web service enables a Service Agent to perform the rewriting, mapping and planning operation over the global query *Q*. The Service Agent exploits the GVV and the Mapping Tables to know which data sources are involved by the posed query.

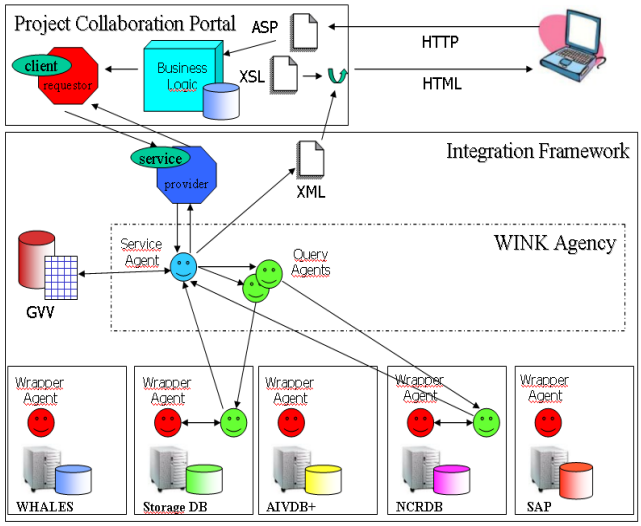

Figure 5: The WINK information flow

In the example all the three different data sources are involved and the following local queries are generated:

SAP source: Q1
Select order as ORDERID, material as MATERIAL,
deliverydate as DELIVERY_DATE
from ODA
where status = 'opened'

WHALES source: Q2
Select ordernumber as ORDERID,
material_id as MATERIAL
reqdate as REQUEST_DATE
supply as WORKPACKAGE
from MYORDER, SUPPLYORDER
where ordernumber = orderid
and projectitemcode = 'CUPOLA'
and orderstatus = 'opened'
and reqdate > Date() - 30

AIVDB source: Q3
Select wp as WORKPACKAGE,
req_seq as REQUIREMENT,
webpage as DOC_LINK
ProjectitemCode, DeliveryDate
from VER_DOC_LINK A, DOCS_LINKS B,
DOCST_DEFINITION C
where A.ci_id = B.ci_id
and B.docs_num = C.docs_id
and A.ci_id = 'CUPOLA'

According to this mapping and to the contingent system workload, the Service Agents will spawn a number of Query Agents. At this stage, the Query Agent will move to the data source(s)/container(s) the query refers to, will interact with the

Wrapper Agent(s) in order to execute the local query(ies) and finally will report the answer to a Service Agent.

The Service Agent composes the results and delivers them to the Project Collaboration Portal. In the example, the following query is executed by the Service Agent to perform the fusion:

```
Select Q1.ORDERID, Q1.MATERIAL,
       Q1.DELIVERYDATE,Q2.WORKPACKAGE,
       Q3.REQUIREMENT, Q3.DOC_LINK
from Q1, Q2, Q3
where  Q1.ORDERID = Q2.ORDERID
and Q1.MATERIAL = Q2.MATERIAL
and Q2.WORKPACKAGE =Q3.WORKPACKAGE
```

In order to deliver results so as to update the correct information, Service Agents reports query answers in the desired format (in our case, an xml file). The calling web service reports to the business logic the query acknowledgement and the URI of the resulting xml file: then the business logic applies the desired XSL stylesheet and dynamically produces a web page reporting the information.

For "designed and scheduled" queries, the extraction process is similar. In addition, a scheduler agent is spawn in the multi-agent system. A scheduler agent manages all details (such as connection pools and data storage) for querying a source on a regular basis. Scheduler agents had been activated during the initial start-up procedures of the WINK system. The configuration of a scheduler agent includes the query to be executed, the data source to be queried, the required drivers to access the source and how results should be communicated back to the WINK system. Communication of results can happen for instance by means of files stored on a given host of the network or by calling a published web service on a given url. This last case is the most suitable any time results have to be further processed. For instance, in our application, scheduler agents call web services whenever modified or new data appear in some particular relation (such as MyReports) in order to fire alerts on the WINK system.

For the two types of "user defined" queries, the extraction process follows the same operations as their respective "designed query" type. What changes is the interface that allows users to compose queries by navigating the metadata of the GVV.

VIII. Conclusion and Future Work

The concept of "Collaboration" characterizes the most recent organizational business paradigm and e-business applications: the WINK project fully addresses the Collaborative-Commerce model and in this paper we described WINK's main features.

WINK intends to represent a common collaboration platform for main contractors and their subcontractors in a sector like that of the aerospace industry, where it is important both to preserve the quality and reliability of components and equipments, and to reduce the entire spatial program life-cycle in order to exploit the advantages offered by the rapid technological evolution and reduce the costs.

In particular, we described the business tier of the system architecture, whose main components are the Project Collaborative Portal, the Integration Framework (implemented by means of a multi-agent system) and a set of web services to guarantee interoperability.

Finally, we illustrated the flexibility and the easy customisation of the WINK system by using a real scenario provided by Alenia Spazio S.p.A, that is currently deploying the system at its Turin and Rome sites. A first testing phase has shown that on average the number of people taking part in the daily meeting has dropped to 15-20, i.e. a fall between 20% and 40% if compared to a daily meeting not supported by the WINK usage. Furthermore, daily meeting duration is on average half an hour time, representing a 50% cutback if compared to daily meeting duration not supported by the WINK system..

Alenia is planning to extend the access to the WINK web client to some of its main sub-contractors. These are usually charged of smaller Verification and Integration tasks and till now the communication between a sub-contractor and Alenia Spazio has been based on very traditional channels, like phone calls and e-mails. The purpose here is twofold: on one hand, sub-contractors have to become active players in the project management, allowing them a degree of access to the project information, on the other hand, they must get the possibility to report their internal partial results through an easy-to-use, configurable means (like the WINK web interface) in a timely and traceable fashion. For this reason, some of the sub-contractors have been assigned not only visibility rights on some project information but have been invited to use the Project Collaborative Portal to update important system data, such as order or workflow data. Such an initiative is helping in obtaining a closer collaboration, as the information systems of the participating partners have to be integrated within the WINK platform.

Along with this, Gruppo Formula is also proposing to its customers the WINK system as a solution for their business purposes. Clients mainly belong to the textile and multi-utilities sectors. From these early approaches to new markets, the most appreciated features turn out to be the flexibility of the system configuration and the interoperability with existing or third parties' applications, mainly due to the deployment of agent technology. This means that strong benefits can be foreseen in terms of time required to tune the system and define the data integration in accordance to the configuration requirements in place. This encourages Gruppo Formula to recommend the adoption of the WINK system to customers, as a solution for projects collaborative management.

References

[1] J. Roberto Evaristo and Bjørn Erik Munkvold, "Collaborative Infrastructure Formation in Virtual Projects" Journal of Global Information Technology Management, Vol. 6 No. 2, April 2003.

[2] L. Wegner and C. Zirkelbach , "Collaborative Project Management with a Web-based Database Editor", in MIS'99 Fifth International Workshop on MULTIMEDIA INFORMATION SYSTEMS 1999, California.

[3] D. Beneventano, S. Bergamaschi, C. Sartori, “Description logics for semantic query optimization in object-oriented database systems”, ACM Transaction on Database Systems, Vol. 28, Issue 1 pp 1-50.

[4] R. G. G. Cattell (Editor), Douglas K. Barry (Editor), et. al. “The Object Data Standard: ODMG 3.0”, ISBN: 1558606475.

[5] I. Benetti, D. Beneventano, S. Bergamaschi, F. Guerra, M. Vincini, An Information Integration Framework for e-commerce, IEEE Intelligent Systems Magazine, (January/February 2002).

[6] D. Beneventano, S. Bergamaschi, J. Gelati, F. Guerra, M. Vincini: "MIKS: an agent framework supporting information access and integration", Intelligent Information Agents - The AgentLink Perspective, March 2003, Lecture Notes in Computer Science N. 2586 - Springer Verlag, ISBN 3-540-00759-8

[7] W3C, Standards, www.w3c.org.

[8] W3C, Simple Object Access Protocol (SOAP) 1.2, W3C Working Draft - 26 June 2002.

[9] D.Gazzotti, M. Felice, P. Paganelli, R. Stevens: WHALES: a Web-based Collaborative Environment for Concurrent Project life-cycle Management in Networked Enterprises. In: Enterprises e-Business applications: results of applied research on e-Commerce, SCM and Extended Enterprises, Springer-Verlag, 2001.

[10] M. Lenzerini: “Data Integration: A Theoretical Perspective”. PODS 2002: 233-246.

[11] G. Miller, Wordnet: A lexical database for English, Communications of the ACM, 38(11):39–41, 1995.

[12] D. Beneventano, S. Bergamaschi, F. Guerra, M. Vincini: "Exploiting extensional knowledge for a mediator based Query Manager", Proceedings of the Convegno Nazionale Sistemi di Basi di Dati Evolute, 2001.

[13] D. Beneventano, S. Bergamaschi, C. Sartori, M. Vincini "ODB-QOptimizer: a tool for semantic query optimization in OODB." Int. Conference on Data Engineering ICDE97, Birmingham, UK, April 1997.

[14] Silvana Castano, Valeria De Antonellis, Sabrina De Capitani di Vimercati: Global Viewing of Heterogeneous Data Sources. TKDE 13(2): 277-297 (2001)