12 INDUSTRIAL APPLICATION INTEGRATION USING AGENT-ENABLED SEMANTIC SOA: CAPNET CASE STUDY

Leonid Sheremetov and Miguel Contreras
Mexican Petroleum Institute, Eje Central Lázaro Cárdenas #152,
07730, México, D.F. Mexico
{sher, mcontrer}@imp.mx

The paper addresses the issues of industrial application integration in business processes using agent-enabled Service Oriented Architectures (SOA). In this paper, we show that agent-enabled SOA can play an important role for service integration. Our architecture combines Web services and intelligent agent technologies orchestrated by a business process management system. This architecture is grounded in a semantic service integration model and supported by the CAPNET agent platform tools. We describe the architecture and illustrate the approach by an industrial application scenario from petroleum wells' drilling.

1. INTRODUCTION

An important aspect of software in industry lies in its ever-increasing complexity. The often need to integrate heterogeneous applications composing business processes into corporate-wide computing systems, and even to extend that beyond company boundaries into the Internet, introduces new levels of complexity. In recent years Enterprise Application Integration (EAI) systems have evolved towards Service Oriented Architectures (SOA) (Haller et al., 2005). The SOA model is based on the principle that business functionality is separated and published as self-contained components, called services. Though Web Services (WS) as the most commonly used implementation of traditional SOA reduces the number of point-to-point adapters because every interface is based on WS Description Language (WSDL), it still leaves open the question of semantic interoperability of these interfaces. That is why recently a major effort has been invested in semantic extensions of SOA known as Semantic SOA allowing for scalable and controlled EAI (Sycara et al., 2003; Preist, 2004, Ackland et al. 2005).

Multiagent systems (MAS) are ideally suited for open and dynamic environments, where automatic group formation, multiagent adaptation, agent coordination, etc. could likely be fruitfully adapted for EAI (Jennings, 2001; Tesauro et al., 2004). Therefore there is a great opportunity to join the Web Services and Agent Services (AS) technologies under the SOA paradigm, exploring several

Please use the following format when citing this chapter:

Sheremetov, L., Contreras, M., 2006, in IFIP International Federation for Information Processing, Volume 220, Information Technology for Balanced Manufacturing Systems, ed. Shen, W., (Boston: Springer), pp. 109–118.

open topics such as business processes modeling, interaction of AS and WS, semantic description of services, and dynamic service negotiation and composition.

Motivated by the above considerations, we have developed an extension to the CAPNET agent platform (AP) permitting to make a step ahead towards the development of industrial applications for open environments (Contreras et al., 2004). The specific software engineering objectives of CAPNET are: to enable a distributed system to self-configure at runtime initialization; to develop agent templates that enable rapid application development through service composition within the system; to improve the efficiency of businesses through automation of business processes in accordance with the system's overall business objectives. We have developed several prototype systems as a concrete testbed to pursue these objectives. One of them, multiagent hybrid intelligent system (HIS) for Lost Circulation Problem (LCP) is described in the case study section of the paper.

The rest of the paper is organized as follows. First we describe briefly the application requirements that motivated the research and analyze the current trends of EAI architectures in order to formulate the basic requirements to agent-enabled SOA. Further we describe the extensions to the architecture of the CAPNET enabling the integration of services provided by agents in a SOA. Finally we illustrate the developed architecture by example of multiagent HIS called SMART-Drill followed by conclusions.

2. MOTIVATION FOR RESEARCH

An oil company is a multi-business organization, which produces, manufactures, markets and transports crude oil, natural gas and petroleum products. The interest in information technology (IT) for decision support of the operational activities lays in the fact that in oil companies like the Mexican *Petroleos Mexicanos* (PEMEX), senior personnel daily have to solve problems based on extensive data analysis and their experience gained through years of field work. Operation of the oilfield is composed of many complex industrial processes, drilling is one of them. LCP – also known as lost returns - stands for the absence or reduction of drilling mud pumped through the drillstring, which filtrates into the formation instead of flowing up to the surface. It is one of the most common problems: drilling fluid may flow freely into the shallow unconsolidated formations because of high permeability or just because of a broken tube. Drilling may continue, or the mud can be thickened and lost circulation material (LCM) added, in an attempt to cure the problem. Sometimes, the LCP is cured easily. But in most cases, the intervention of experienced petroleum engineers is required to find the most appropriate and efficient solution.

2.1 Application Requirements

Expert systems technology is a feasible option to support drilling operations (Garrouch and Lababidi, 2001). In (Sheremetov et al., 2005), a first desktop version of the HIS for LCP diagnostics was described. During the field-testing phase with the PEMEX Company, the following requirements were identified:

- The need for a real time system's feeding with data from distributed information sources: locating and typing data manually was "a torture" for the operators.

- The need to standardize input data description in order to exchange information with other drilling applications. Data should be described in XML-based Wellsite Information Transfer Standard Markup Language (WITSML) (WITSML, 2006) and be available in a server-based repository.
- The need to contact third parties in order to select the most appropriate solution.
- The need to enable a collaborative interaction between the well operator, petroleum and chemical engineers and project manager, since decision-making process is distributed.
- The need to operate over different oil & gas assets (North, South and Marine zones in the case of PEMEX) and over a common case-base.

Therefore, while developing a second version of the HIS, we considered a distributed architecture to overcome the limitations of the stand-alone approach. This new version was implemented over an agent platform and legacy WS, orchestrated by a business process management system. Even though there was no need for automatic discovery and dynamic service composition (but the need for dynamic instantiation), this solution implied several changes to the CAPNET AP allowing its agents to work within a heterogeneous IT environment in a service-oriented fashion.

2.2 Enterprise Application Integration Using Semantic SOA

Integration of IT is crucial for companies as only integrated information systems can deliver business values, such as efficient decision-making support, instant access to information, data integrity, along with decreased cost of software development and maintenance. Traditional EAI systems provided three types of integration levels (Ruh, 2000): process, transformation and transportation layers. Migration of EAI towards SOA changes these layers in order to provide: (i) a standardized way to expose and access the functionality of applications as services, (ii) an enterprise bus infrastructure for communication and management of services, including message interception, routing, transformation, etc, (iii) an integration architecture between the various services and existing and newly developed applications used in business processes and (iv) a specialized language (like the Business Process Execution Language for Web Services -BPEL4WS-) for composition of exposed functionalities of applications into business processes (Juric et al., 2006).

Semantics should be included as the fifth layer of the integration scheme since adding semantics to the service description, firstly, provides a formal description of the functionality of a service. This description allows the developer to base the manual integration on the knowledge about the meaning of the data. Secondly, the model permits decentralizing of semantics of different systems overcoming one of the principal drawbacks (centralized semantics) of traditional EAI. Finally, semantics bring closer the possibility of composing services dynamically by discovering them at runtime.

The centerpiece of semantic integration is an ontology that conceptualizes and codifies knowledge that can be mapped as a knowledge domain. In the context of this paper all the agents share an ontology of their domain of expertise - their *domain model* - that establishes the terminology for interacting with the agent and its services.

3. SEMANTIC SERVICE ORIENTED ARCHITECTURE OF THE CAPNET

The main objective of CAPNET is to bring an integrated infrastructure for MAS that covers the programming, deployment, administration and integration of agents with enterprise applications within the semantic SOA. In this section we present how this integration was achieved. Due to the space limitations, the integration details are covered at the architectural level only and not at the implementation level.

3.1 Making the Agent Platform Service Oriented

The requirement of integration of services provided by agents in a SOA where service composition can be achieved requires an AP to communicate effectively with agent service requestors and providers. However the current notion of services in the agent community is unstructured and does not provide the required elements for service composition which is crucial for expressing service relationships and defining service flows. Additionally agents do not expose standard interfaces that are consumable by the majority of non-agent software and as a result agent technology is not widely adopted in industrial applications, mainly because of its lack of support for integration with business processes. However even with these limitations, agents are ideally suited to representing problems that have multiple problem solving methods, multiple perspectives and/or multiple problem solving entities and incorporate semantics and knowledge management in a natural way.

An AP is a software architecture that controls and manages an agent community allowing the survival of an agent in a distributed and heterogeneous environment (FIPA, 2005). Based on the FIPA specifications that considers an AP as a set of four components - Agents, Directory Facilitator (DF), Agent Management System (AMS), and Message Transport System (MTS) - representing a set of logical capacities or services, it is possible to make a mapping between the concepts in APs and the features required for an architecture to be considered as Service Oriented. These concepts are: *services* provided by agents, *service provider* - the agents implementing services, *service consumer* (or requestor) - end-user application or another agent, and *service locator* - the DF that acts as a registry and allows for the lookup of service provider interfaces (agents) and service locations.

In order to implement this mapping and be able to establish relations, cross invocation and composition of agent and Web services, several extensions to a basic FIPA compliant AP should be addressed that include:

1. An agent service model compatible with the current standards for SOA.
2. A common model and related language for the semantic description of services.
3. A mechanism for publishing agent provided services in UDDI registries.
4. A mechanism for finding services published in UDDI registries.
5. A common transport protocol for communication.
6. A mechanism for exposing the functionality of agent provided services to Web service clients.
7. A mechanism that enables agents to consume Web services.

The architecture implementing these mechanisms within the CAPNET agent platform is addressed in the following section.

3.2 CAPNET Agent Service Model

The first requirement for the integration, to provide AS model compatibility with the current standards for WS, and its implementation implied the design of a new class of agent for CAPNET that was heavily based on the WS architecture for its specification. The architecture of the "component agent" (Figure 1) was designed as a .NET component that can be easily hosted either by the application that will use its services or by a CAPNET container that can provide it with mobility services.

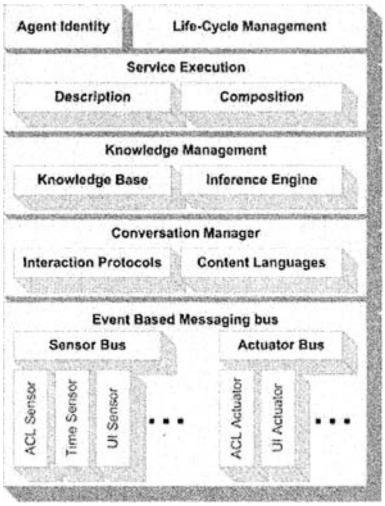

Figure 1 - CAPNET Agent's Architecture.

The "component agent" is composed of the following parts:

- An event based messaging bus that is connected to several components, which constitute the agent's sensors and actuators for the "outside world". Those sensors currently include (but are not limited to) one for receiving Agent Communication Language (ACL) encoded messages, a sensor for time awareness and a sensor for perceiving events generated by the user interface (UI) of the system where the agent resides. Currently, only two actuators are available but more can be implemented and plugged in: an ACL actuator that sends ACL encoded messages to the AP and an actuator that communicates changes in the agent and/or the system state to the UI of the system that contains the agent.
- A Conversation Manager that is capable of handling complex structured conversations based on FIPA Interaction protocols and the common patterns of service invocation (both synchronous and asynchronous) and using a content language for negotiation of services. The currently supported languages include: FIPA-SL0, FIPA-SL1 and propriety CAPNET-KRF.
- A Knowledge Management component that uses a Knowledge Base for representing the agent's domain (based on CAPNET-KRF) and an inference engine to handle this internal knowledge representation.

- A Service Description, Composition and Execution component that enables the specification of the services and their model (both syntactically and semantically) along with its particular implementation. This component allows for basic composition of services inside the agent by providing a specification language (a subset of BPEL) that handles sequence, parallel and loop service execution to form composed services. For the semantic description of services OWL-S is used to define the service profile, process and groundings.
- A component that handles the lifecycle of the agent in terms of the state transitions specified by FIPA and also in terms of its particular functionality. This component tracks the state of the agent and conducts its behavior based on the transitions of a finite state automaton fired by the inputs perceived by the sensors, the internal knowledge and the results of its services.
- A component that handles the agent identity and capabilities.

The second requirement deals with the semantic description of services and it is implemented by adopting the OWL-S ontology model and language allowing service description in terms of profiles, models, and groundings, where the service profile tells "what the service does", the service model tells "how the service works", and the service grounding specifies the details of how a client can access a service. In the case of the CAPNET AP it is a common practice to specify at least two service groundings for a particular service, where one maps to the SOAP interface of the platform and is meant to be used by external clients and another one intended for intra-platform use, maps directly to the agent identity and home address. This adoption of the OWL-S language required changes to be made to the agents' internal representation of services and to the DF service, which should be able to handle this extra information for storage and matching/retrieval operations.

Requirements 3 and 4 for the integration were also addressed by modifying the basic functionality of the DF agent that is now capable of storing service descriptions fully compatible/transformable with that of WS. If an agent requests to publish its service as a WS, the DF handles its registration with one or several UDDI directories. The DF is also capable of performing federated search for services on UDDI directories on behalf of the agents, providing them with service descriptions to enable direct service invocation by these agents using the platform's MTS.

The requirements 5, 6 and 7 are handled directly by the MTS, by the addition of a SOAP transport manager that implements the necessary mechanisms for SOAP based communication. It is implemented as a server extension for MS Internet Information Services (IIS) responsible for providing the access point for the services provided by agents, and to translate between AS and WS descriptions. For more details on the MTS description and the transport manager factory mechanism see (Contreras et al., 2004). The functionality of this transport manager includes ACL to/from SOAP encoding and transformation, service description and invocation conversion to/from WSDL and session handling for service invocation from the platform to external WS.

3.3 Domain Model

The primal intention of the CAPNET knowledge representation model is to provide a common and consistent symbolic representation for the agent domain. CAPNET Knowledge Representation Format (KRF) is based on the FIPA-RDF (FIPA, 2005).

This model solves the ambiguity in defining relationships among entities and introduces the possibility of expressing rules. KRF is used to specify the domain model; the service model is based on OWL-S descriptions.

The CAPNET KRF represents entities from the application domain by using structures known as Objects. Objects instantiate Resources. Properties set the value of some feature or attribute that belongs to the Object/Resource. A Property is composed of: a name, data type and value. If values for some property are restricted to a list of possibilities or valid ranges, then it is said that it has Constraints.

In CAPNET KRF the rules define relationships between known information (antecedents or premises) and information that can be concluded (consequents or conclusions). Antecedents and consequents are propositions that relate properties from resources to some value by one of the following operators: equal to, greater than, greater than or equal to, lower than, lower than or equal to and not equal.

CAPNET Knowledge Acquisition Tool (KAT) and CAPNET Expert System Shell (ESS) are complementary tools enabling knowledge acquisition and encoding in CAPNET KRF and fuzzy reasoning capabilities. CAPNET ESS can be used by the CAPNET agents as an internal (as in the "component agent") or external component. It implements conjunctive, disjunctive and additive algebras of strict monotonic operations on finite ordinal scales represented as multi-sets over the CAPNET KRF (Sheremetov et al., 2005). This representation gives the possibility to take into account the change of plausibility of premises in the rules and to differentiate and refine the uncertainties of conclusions.

4. CASE STUDY DESCRIPTION

The general solution scheme for the LCP consists of several stages. Once the problem is detected, the data stored in daily perforation reports (DPR), laboratory deposits and perforation program (PP) are analyzed in order to determine loss severity. In the case of a total loss severity, the action to perform is placing LCM; else, the solution consists of lowering pump rate and rate of penetration. If the problem persists, additional data from lab tests and the drill log are to be analyzed regarding the viability of lowering density as the next measure. In case of failure, two remaining measures could be applied: placing LCM or squeezing cement plugs. Finally, if none of these controlled the problem, critical measures must be applied.

The business process model developed for the orchestration of the Web and agent services follows the diagnostic LCP process model described above according to the business process scheme shown in Figure 2. The first step in the process is the specification of a user requests for the solution of a LCP problem. An iterative process takes place producing a possible diagnostics and a set of proposed measures at each iteration corresponding to one of the five phases of the problem solution. This diagnostic process involves the use of the IMP_SmartDrill_HIS service, which is actually an orchestration of several services provided by "normal programs" and agents, specified in BPEL4WS and implemented using MS Biztalk Server.

In the case study we modeled three main distributed information sources: DPR, PP and laboratory results. To get information from these sources we developed three agents that expose their functionality as Web services using the CAPNET AS-WS integration mechanism. Each asset has instances of these agents associated with it.

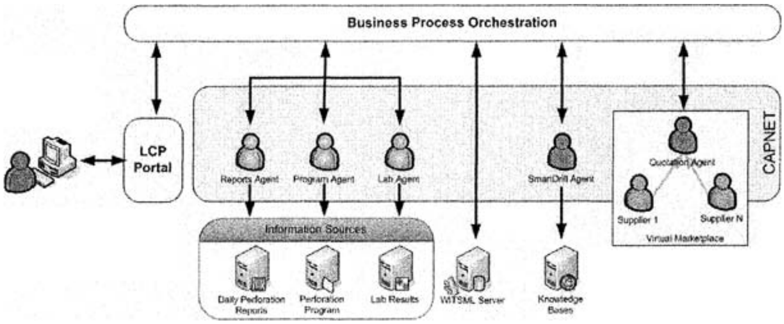

Figure 2 - Business process implementation scheme

The Reports Agent provides the DPR_WITSML_service responsible for converting a DPR format into WITSML objects. The Program Agent implements the PP_WITSML_service responsible for converting a PP format into WITSML objects and the Lab Agent implements the LR_WITSML_Service.

The WITSML Server is a legacy WS implementation of the API that is able to store, manage and retrieve WITSML objects from a central repository. This repository is used as the knowledge source for the inference process that would lead to a diagnostics.

Expert system module is implemented as a service provided by a single agent (SmartDrill Agent) that uses the CAPNET ESS as its engine, a set of knowledge bases as reasoning base and the knowledge from the WITSML Server as the source of facts for the inference. At each phase the agent may propose different measure for solving the problem: lowering pump rate and rate of penetration, lowering density, LCM, squeezing cement plugs or eventually taking extreme measures.

If the proposed measure involves the use of LCM or cement plugs, the Quotation Agent will attend a virtual marketplace for quotations of the required materials. The marketplace is implemented entirely by agents that represent different vendors of the materials that may be needed in an LCP problem. CAPNET agents that behave according to the FIPA-Contract-Net-Protocol and make proposals based on private catalogs currently are only simulating this marketplace. However no actual contracts are made, since the quotation agent only "collects" the proposals and reports the results to the LCP Portal.

When the system completes the iteration it would keep the state and ask the user if the problem has been solved. If this is not the case, it will proceed to the next iteration where the SmartDrill Agent will shift knowledge base and infer from a different set of rules, up to the fifth phase where the process ends. The application is developed as LCP Portal implemented as a set of MS Sharepoint WebParts indicating the current situation of the well, the relevant LCP data, and diagnostics and solution details (Figure 3).

From the server side the following tools were used:

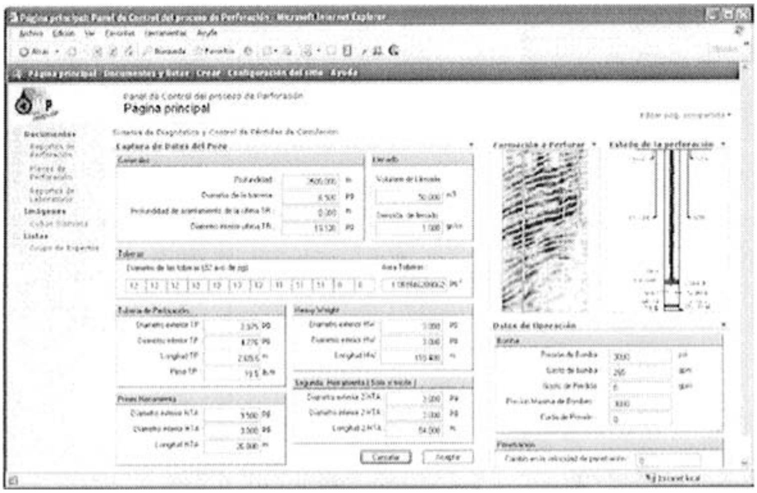

Figure 3 – LCP portal interface

- MS .Net Framework 1.1.
- MS SQL Server 2000.
- MS Biztalk Server 2004 for orchestration and choreography of BP.
- CAPNET Agent Platform.
- MS IIS Web server used as WS container for CAPNET and Sharepoint.
- MS Sharepoint Portal server.
- CAPNET Expert System Shell as the inference component for HIS agents.

5. CONCLUSIONS

In this paper, we descried a CAPNET service model featuring some innovative elements like seamless integration with the model of WS, the most widely adopted implementation of SOA. This integration allows taking advantage of the capabilities of WS-related standards such as service composition while maintaining the MASs' advantages for organization and dynamic coordination enabling easy integration of agent services in industrial applications. Compared to SOA kernel of the JADE (Bellifemine, 2001), CAPNET service model permits a transparent integration mechanism between AS and WS that does not require agents inside the platform to request the service from a local "gateway agent", instead a virtual agent reference is created by the DF when a search is performed and instantiated by the MTS for each invocation of an external WS.

The SMART-Drill application developed using the proposed approach has benefited from the agent technology for working with distributed knowledge sources, for fuzzy reasoning to get LCP diagnostics and for negotiation of services (using FIPA-CNP) by solution provider agents organized in marketplaces. Marketplaces also permitted to take advantage of role assignment techniques managed within agent organizations. Semantic capabilities of agents (domain model, KRF content language and ACL) were used mainly for the application description, while OWL-S extended Agent Service Description permitted agents to discover particular agent instances at the DF (data access agents for the assets). For the application at hand there was no need in automatic WS discovery mechanism and dynamic service composition.

Therefore a key contribution of this paper is twofold: (i) semantic SOA of an agent platform and (ii) a working demonstration of agent-enabled semantic SOA operating in real-life context of industrial applications. Several months ago, the second version of the SMART-Drill was also installed in PEMEX for field-testing.

6. ACKNOWLEDGMENTS

Partial support for this research work has been provided by the IMP within the projects D.00006 and D.00322. The last author would also like to acknowledge the support provided to him by the IMP Postgraduate Studies Program. Special thanks to J. Martinez for his invaluable contribution to the SMART-Drill programming.

7. REFERENCES

1. Ackland, R., Taylor, K., Lefort, L., Cameron, M. and Rahman, J. "Semantic Service Integration for Water Resource Management". In Proc. of the 4th International Semantic Web Conference, Gil, Y.; Motta, E.; Benjamins, V.R.; Musen, M. (Eds.), LNCS 3729, Springer, 2005.
2. Bellifemine F., Poggi A., Rimassa G., Developing multi-agent systems with a FIPA-compliant agent framework, Software - Practice And Experience, 2001 no. 31, pp. 103-128.
3. Contreras, M., Germán, E., Chi, M. and Sheremetov, L. Design and implementation of a FIPA compliant Agent Platform in .NET. J. of Object Technology, ETH Zurich, 3(3), March-April, 2004
4. Foundation for Intelligent Physical Agents – FIPA Specifications, (2005), http://www.fipa.org.
5. Garrouch A. and Lababidi H., "Development of an expert system for underbalanced drilling using fuzzy logic", J. of Petroleum Science and Engineering, vol. 31, pp. 23–39, 2001.
6. Jennings, N. An Agent-based Approach for Building Complex. Software Systems. Comm. ACM, 44, No. 4 (2001) 35-41.
7. Juric M., Sarang P. and Mathew B., Business Process Execution Language for Web Services Second Edition, Packt Publishing, pages 6-30, 2006.
8. Haller A., Gomez J. M., Bussler C. "Exposing Semantic Web Service principles in SOA to solve EAI scenarios". In Proc. of the WWW2005, May 10–14, 2005, Chiba, Japan.
9. Oberle D., Lamparter S., Eberhart A., Staab S., Grimm S., Hitzler P., Agarwal S., and Studer R. "Semantic Management of Web Services using the Core Ontology of Services". In Proc. of the W3C Workshop on Frameworks for Semantics in Web Services. 2005.
10. Preist C. "A Conceptual Architecture for Semantic Web Services". In Proc. of the International Semantic Web Conference (ISWC), 2004.
11. Ruh W. A., Maginnis F. X., and Brown W. J. Enterprise Application Integration: A Wiley Tech Brief. Wiley, 2000.
12. Sheremetov L., Batyrshin I., Martinez J., Rodriguez H., and Filatov D. "Fuzzy Expert System for Solving Lost Circulation Problem". In Proc. of the 5th IEEE Int. Conf. on Hybrid Intelligent Systems, Rio de Janeiro, Brasil, Nov. 6-9, pp. 92-97. IEEE, 2005.
13. Sycara, K. et al. Automated discovery, interaction and composition of semantic Web services. J. of Web Semantics. 1(1):27–46. 2003.
14. Tesauro G., Chess D., Walsh W., Das R., Whalley I., Kephart J. and White S., "A multiagent systems approach to autonomic computing", In Proc. AAMAS, 2004.
15. WITSML Web Site, http://www.witsml.org/, 2006.