Semantic Web Fred – Agent Cooperation on the Semantic Web

Michael Stollberg[1], Uwe Keller[1], Peter Zugmann[2], Reinhold Herzog[2]

[1] DERI – Digital Enterprise Research Institute
University of Innsbruck, Austria
{michael.stollberg, uwe.keller}@deri.org

[2] Quarto Software GmbH
Vienna, Austria
{reinhold.herzog, peter.zugmann}@quarto.at

Abstract

Semantic Web Fred, SWF for short, is an environment for automated cooperation of agents on the Semantic Web that combines agent technology, ontologies, and Semantic Web Services. In alignment with the Web Service Modeling Ontology WSMO, SWF applies emerging Semantic Web Services technologies for advanced automated goal resolution with dynamic service usage. For demonstration purpose, we present a user interface that leads through the steps of cooperative goal resolution in SWF for explaining the framework and technical architecture with special attention to the mechanisms developed for dynamic cooperation establishment.

1 Introduction

Different key technologies for the Semantic Web have been identified: ontologies for semantically enhanced information exchange over the Internet, Web Services for reuse and interoperability of computational functionality, and agent technology for automated execution of tasks [Berners-Lee et al., 2001]. SWF exploits the potential of the Semantic Web by combining these technologies into a coherent system strongly aligned with the Web Service Modeling Ontology WSMO.

In SWF, software agents called *Freds*, perform tasks automatically on behalf of their owners. According to the paradigm of agents as autonomously acting entities in a software environment, Freds have to interact in order to resolve their distinct tasks. Therefore, a Fred has to find a suitable cooperation partner as well as the computational resources required for automated task resolution. With regard to a service-oriented architecture as envisioned for Semantic Web Services, the main building blocks of SWF are Goals and Services [Fensel and Bussler, 2002]. A Goal represents a task that a Fred is assigned, and a Service is a computational resource that allows automated resolution of Goals. SWF develops advanced mechanisms to identify possible cooperation partners, detect the services needed for automated goal resolution, and to execute such cooperations between agents.

The starting point for the SWF project is the FRED system developed by Quarto, an agent-based environment for task delegation to electronic representatives [Stollberg et al., 2004], and the Web Service Modeling Ontology WSMO which develops an overall framework for Semantic Web Service.[1] The SWF project is funded by Austrian government under the CoOperate 2003 programme, awarded as the 2nd best proposal in the call.

2 SWF Framework and Architecture

The SWF framework shown in Figure 1 relies on a real-world cooperation model. For solving a complex task, several parties have to interact; a cooperation will only take place if it is profitable for all partners, as nobody offers a service without getting something in return. Following this, each Fred in SWF has a Goal (the task

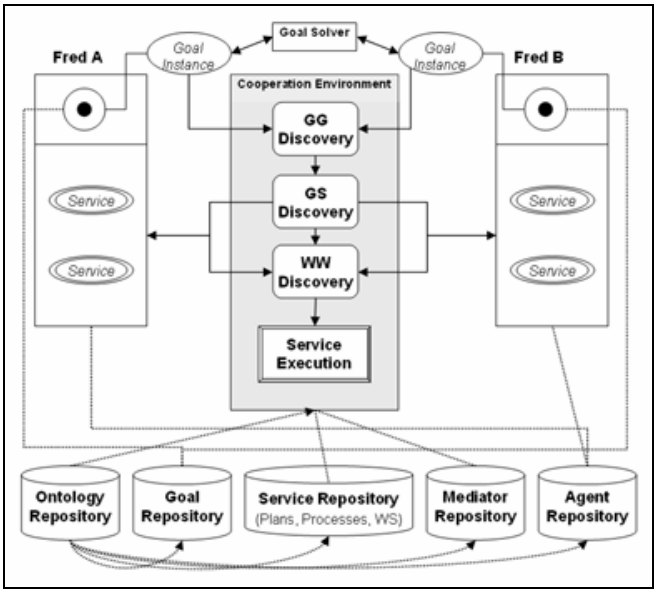

Figure 1: SWF Overview

[1] WSMO Standard: [Roman *et al.*, 2004], www.wsmo.org

assigned by its owner), and Services that provide the functionality the Fred needs for automated task resolution as a party participating in a cooperation.

For realization of SWF, WSMO-enabled technologies for those components that are directly WSMO compatible are combined with specialized mechanisms for SWF. This allows use of other WSMO-enabled technologies within SWF and ensures compatibility with emerging Semantic Web Service technologies around WSMO. The following depicts the technical building blocks of SWF, while we explain the core mechanisms and the demonstration use case later.

Agents (Freds). A *Fred* is a software agent that acts as an electronic representative on behalf of its owner. The FredBase is the FIPA-compliant agent runtime environment of the FRED system with facilities for management of agents. Interactions between Freds take place in Meeting Rooms, wherein all computational resources are available for Service Execution.

Ontologies. Ontologies provide the formally specified terminology used by the other components. For handling ontologies, they are transformed into Java Objects (called Smart Objects) that are as expressive for ontology specification as OWL-DL. The main benefit of Smart Objects is that conventional technologies can be used for ontology data handling. For ontology management, the Smart Object technology includes a scalable storage facility and techniques for evolution support, mismatch-handling, as well as import / export to standard ontology languages (WSMO-Ontologies, RDF, OWL is planned).

Goals. A Goal represents a desire that a client delegates to a Fred for automated resolution. SWF distinguishes Goal Schemas and Goal Instances: the former specifies the ontological structure of goals as pre-defined desires, while the latter instantiates a Goal Schema as a concrete a desire at a certain point in time. Goal Schemas in SWF are WSMO-Goals, described by a postcondition (the desire) and by effects (condition on the world that shall hold after goal resolution) [Roman et al., 2004]. SWF Goal Instances hold additional information: a submission, i.e. data that are submitted as input to a service, as well as the owner, creation date, and resolution status.

Services. A Service is a computational resource for automated goal resolution. SWF supports three types of Services: *Plans* are Java-programs; *Processes* are multi-step services allowing complex, nested services wherein each activity can be resolved arbitrarily; and *external Web Services*. SWF Services are described as WSMO Web Services, i.e. by *non-functional properties* for management, a *Capability* as the functional description, and a *Choreography Interface* as the behavior interface for service consumption.

Mediators. In WSMO, Mediation is the central concept for handling heterogeneities that might occur between resources that ought to interact. WSMO Mediators connect one or more *sources components* with a *target component*, whereby a *mediation service* is included for resolving the mismatches between the connected components. SWF supports WSMO Mediators, using OO Mediators to handle terminological mismatches between ontologies and the other types of WSMO Mediators within specific discovery mechanisms. The development of mediation facilities is out of scope for the SWF project, thus existing Mediators are applied.

Repositories. An UDDI registry that is aligned with the WSMO Registry [Herzog *et al.*, 2004] holds all SWF components, apart form Freds. It realizes a hybrid architecture, thus combining the benefits of central and peer-to-peer repositories. The non-functional properties of each component are mapped to equivalent UDDI information types and are stored centrally in the registry, while the detailed semantic descriptions are kept locally in a persistent repository at the respective owner. Publishing and retrieval is supported by the regular UDDI functionalities, whereby specific SWF components are transformed into the corresponding WSMO component.

3 SWF Mechanisms

The center of Figure 1 shows the general structure of the SWF mechanisms for establishing and executing cooperations between Freds. This section explains the workflow of cooperative goal resolution in SWF and the technical realization of the distinct mechanisms.

The first step in cooperative goal resolution is to detect suitable cooperation partners, i.e. Freds that have compatible Goals which can be resolved by a cooperation of the agents. To determine potential cooperation partners, the compatibility of the Goals assigned to Freds has to be checked, which is done in *Goal-to-Goal Discovery* (short: GG Discovery). Then, each potential partner has to find available services that allow automated execution of the cooperation role of the agent. This is performed in *Goal-to-Service Discovery* (short: GS Discovery) which returns a set of Services that a Fred can use for automation of his cooperation role. For completely automated cooperation, the Services to be used by cooperation partners need to have compatible Choreography Interfaces, which is checked in *Service-to-Service Discovery* (short: WW Discovery, for 'Web Service') as the last step for cooperation establishment. After successful cooperation establishment, a cooperation contract is defined that is comprised of all needed information (the cooperation partners, their Goal Instances, and the Services to be used). Then, the Freds are sent into a Meeting where the cooperation contract is executed; a meeting is considered to be *productive* when the Goals assigned to the cooperation partners are resolved successfully, and unproductive otherwise.

The SWF Discovery Mechanisms for cooperation establishment significantly increase the rate of productive meetings between Freds. The discoverers are realized as modules wherein inference engines work on the semantic descriptions of SWF components (Goals, Services, and the Ontologies used). The approaches for matchmaking in the discoverers described below are aligned with the

generic technologies for Semantic Web Services developed around WSMO, thus SWF presents a prototypical solution of a WSMO-enabled environment for Semantic Web Services.

3.1 GG Discovery

As the first step in the cooperative goal resolution, GG Discovery detects potential cooperation partners by determining the compatibility of their respective Goal Instances. The compatibility is given when the objects of interest (i.e. the Goal postconditions) match, and when the cooperation roles of the Goal Instance owners are compatible.

Figure 2 shows the structure of GG Discovery. A so-called *Cooperative Goal* defines compatible Goal Schemas, thus specifying compatible cooperation roles and constituting a pre-selection of possible cooperation partners at design time. Optionally, a WSMO GG Mediator can be defined that resolves mismatches between Goal Schemas that are not compatible a priori. For checking the compatibility of the objects of interest of Goal Instances created from compatible Goal Schemas, the ontology objects defined in the Goal Instances have to be either the same or be a superset or subset of each other. The matchmaking between Goal Instances is realized in the FOL-theorem-prover VAMPIRE [Riazanov and Voronkov, 2002] by checking whether the Goal Instance of the initiating Fred (Fred *A* is the figure) logically entails the Goal Instances of Goal Schemas that are compatible to the one of Fred *A*.

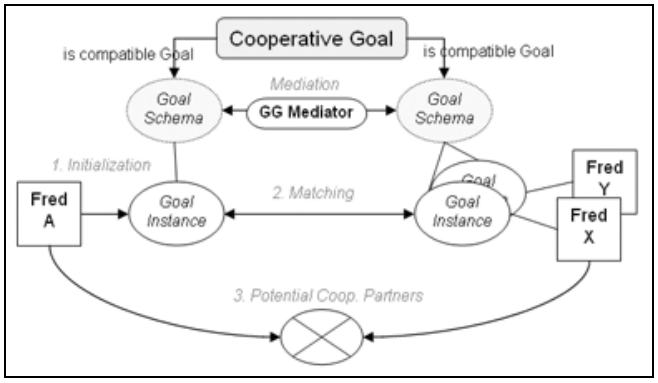

Figure 2: GG Discovery Overview

The result of this mechanism is a set of Freds (the Goal Instance owners) as potential partners for cooperative goal resolution. Three separate search engines initiate GG Discovery: one explores the system for new Goal Instances created by Freds permanently, the second one triggers GG Discovery on basis of explicit events, and the third one initiates GG Discovery dynamically during runtime, e.g. when a new Goal Instance is created by a process.

3.2 GS Discovery

The second step in the resolution process is GS Discovery which detects suitable services for automated goal resolution separately for each cooperation partner identified in GG Discovery. Analogous to Web Service Discovery, GS Discovery returns a set of services that a partner can use for automated goal resolution and thus realizes the WSMO approach for Web Service Discovery [Keller et al., 2004].

Figure 3 shows the structure of GS Discovery for Fred *A* as a potential cooperation partner. The proof obligation for matchmaking between the Goal Instance of Fred *A* and Service Capabilities as functional descriptions of available Services is that under consideration of all Ontologies and Mediators used in the Goal and the Capability description, if the submission of the Goal Instance satisfies the precondition and assumption of the Capability, and if the Goal Instance postcondition implies the Capability postcondition as well as if the Goal effects imply the Capability effects, then that the Service matches the Goal. For resolving partial Goal-Capability matches into exact matches, a WSMO WG Mediator defines the required reduction similar to the usage of GG Mediators in GG Discovery.

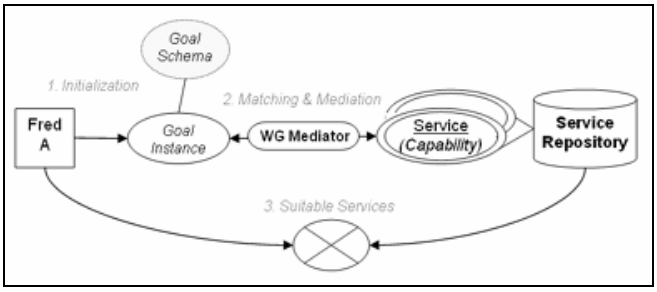

Figure 3: GS Discovery Overview

The conceptual model of GS Discovery relies on logical entailment of the Service Capability by the Goal, and thus it realized in VAMPIRE similar to checking compatibility to objects of interest in GG Discovery. With regard to performance improvement, GS Discovery is also subsequently performed at design time and runtime: when a new Goal Schema is created (design time), a set of suitable services it detected. Out of this, a subset is determined for specific Goal Instances during runtime.

3.3 WW Discovery

For automated interaction of services as the realization of cooperative goal resolution, the Choreography Interfaces of cooperation partners have to be compatible with regard to behavioral models and messaging sequences. This is checked in WW Discovery as the last step in cooperation establishment determines, resulting in a global interaction model of the partners' services that allows automated execution of the cooperation.

Figure 4 illustrates the structure of WW Discovery for two Freds (*A* and *B*) that have been determined as potential cooperation partners, and each of them has discovered a set of possibly usable Services. The oval boxes represent the Choreography Interfaces with the external visible behavior and the related messages as defined in WSMO Choreography [Roman et al., 2004b].

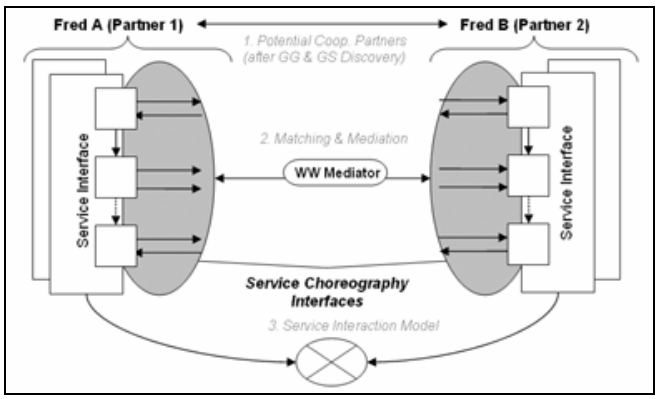

Figure 4: WW Discovery Overview

Determining the compatibility of Choreography Interfaces consists of two aspects: first, the workflow of the Service Choreographies have to compatible (process level compatibility), and second the expected messaging sequence has to be compatible (protocol level compatibility). The former is considered to hold if either the process models are the same (sequence of activities and transitions), or if one of them subsumes the other. The latter is given if the message sequences of Service Choreography Interfaces are symmetric or inverse, in the simplest case; a more advanced technique for determining protocol level compatibility is under construction at the time of writing. In order to resolve possible mismatches, a WSMO WW Mediator can be included to establish compatibility on the process and protocol level.

4 Demonstration Use Case

For showcasing the SWF technology, we specify a typical buyer-seller scenario for cooperative goal resolution. In an imaginary virtual marketplace for purchasing furniture: buyers want to purchase a certain type of chair, and several sellers offer different types of furniture: each Seller has the goal to sell its products This presents a typical scenario for cooperative goal resolution – the goal of the Buyer and the one of the Seller can only be achieved if they cooperative.

We will present the components of the Use Case, i.e. the needed domain ontologies, the Goal Schema and Goal Instance descriptions for buyers and the sellers, a collection of services and Freds as electronic representatives for the marketplace participants. These are implemented as resources in the SWF system, so that we can show how the resources are processed to establish cooperative goal resolution (the complete use case is available at: http://www.deri.at/research/projects/swf/usecase/).

For demonstration purpose, we provide a graphical user interface that subsequently shows the steps for cooperation establishment and execution: this allows explaining the SWF framework and mechanisms contained by a real world example, so that attendees can follow the complex architecture of the system and perceive the results of each mechanism in detail.

5 Conclusion and Future Work

SWF combines ontologies, Semantic Web Services, and agent technology into a coherent system that represents a prototypical solution for integrated environments as envisioned for the Semantic Web. The advanced mechanisms for establishing cooperative goal resolution increase the rate of successful meetings between agents, thus improve the quality of the FRED system. The alignment of the SWF technology with the Web Service Modeling Ontology WSMO allows usage of other WSMO-enabled technologies for Semantic Web Services and ensures compatibility with emerging technologies. In conclusion, SWF provides a significant contribution to Semantic Web and Semantic Web Service technology.

Currently, SWF is designed for a closed FRED environment. It is planned to extend the SWF technology to a web environment; therefore, only specific components have to be adopted, while the general architecture and components do not have to be changed.

References

Berners-Lee, T.; Hendler, J.; Lassila, O.: *The Semantic Web. A new form of Web Content that is meaningful to computers will unleash a revolution of new possibilities.* In: Scientific American May 2001.

Fensel, D.; Bussler, C.: *The Web Service Modeling Framework WSMF*. Electronic Commerce Research and Applications, 1(2), 2002.

Herzog, R.; Zugmann, P.; Stollberg, M.; Roman, D.: *WSMO Registry.*, WSMO Working Draft D10, 26 April 2004.

Keller, U.; Lara, R.; Polleres, A. (ed.): *WSMO Discovery*. WSML Working draft D5.1, 13 September 2004.

Riazanov, A.; Voronkov, A.: *The design and implementation of VAMPIRE*. AI Communications 15(2), Special issue on CASC, pp. 91 -110, 2002.

Roman, D.; Lausen, H.; Keller, U. (eds.): *Web Service Modeling Ontology WSMO v 1.0*, 20 September 2004.

Roman, D., Stollberg, M; Vasiliu, L. Bussler, C.: *Choreography in WSMO*, WSMO Working 17 July 2004.

Stollberg, M.; Lausen, H.; Arroyo, S.; Herzog, R.; Smolle, P.; Fensel, D.: *Fred Whitepaper*. DERI Technical Report DERI-TR-2004-01-09.

Used Software

FRED System, commercial product developed by Quarto Software GmbH, homepage: http://www.quarto.at

SWF Discoverers, open source modules available at: http://cvs.deri.at/cgi-bin/viewcvs.cgi/swf/src/org/deri/swf/

VAMPIRE Theorem Prover,
homepage: http://www.cs.man.ac.uk/~riazanoa/Vampire/