Service Level Agreement Languages

Alessandro Raffio
Politecnico di Milano,
Dipartimento di Elettronica e Informazione
via Ponzio 34/5, 20133, Milano, Italy
+39 02 2399 3661
raffio@elet.polimi.it

Christina Tziviskou
Politecnico di Milano,
Dipartimento di Elettronica e Informazione,
via Ponzio 34/5, 20133, Milano, Italy
+39 02 2399 3649
tzivisko@elet.polimi.it

ABSTRACT
I *Service Level Agreement* (SLA) permettono a due soggetti di definire una sorta di contratto, che sia in grado di garantire il rispetto di alcuni vincoli (qualità del servizio, tipo dei parametri in ingresso e dei risultati in uscita, …) e di pattuire un prezzo per un servizio erogato da un fornitore (*service provider*) e consumato da un cliente (*service customer*).

In questo articolo descriviamo WSLA e WS-Agreement, due diverse architetture per la definizione e la gestione di Service Level Agreement per Web Services. Rivolgiamo il nostro interesse al mondo dei Web Service solo a causa della grande attenzione che la comunità scientifica sta ponendo nei servizi web, ma premettiamo che le tecnologie qui descritte sono applicabili a qualsiasi tipo di servizio – non obbligatoriamente web – per il quale sia definibile un contratto tra un fornitore e un consumatore.

General Terms
Management, Languages.

Keywords
Web services, WSLA, WS-Agreements, SLA

1. INTRODUZIONE
Essendo il Web diventato una piattaforma di esecuzione per il commercio elettronico, si presenta la necessità di definire (1) processi che gestiscano in modo automatico le interazioni business-to-business (B2B), e (2) strutture che stabiliscano contratti aziendali tra i partner e che controllino l'adempimento degli obiettivi descritti nel contratto.

Attualmente, per la scelta e la consegna di prodotti e di servizi in modo elettronico, sono utilizzati sempre più frequentemente tecnologie basate sui Web Service. Uno scenario tipico vede interazioni implementate con lo scambio di messaggi fra i partner in modo (parzialmente) automatico, basato su tecnologie Web; i servizi si descrivono in Web Services Defintion Language (WSDL) [7], un linguaggio di specificazione basato su XML per descrivere le interfacce del servizio e i suoi meccanismi d'accesso. Il fornitore dei prodotti pubblica i suoi servizi in indici appropriati. Universal Description Discovery e Integration (UDDI) [6] è un indice pubblico che (1) espone la descrizione del servizio, e (2) fornisce informazioni di contatto per il fornitore. I clienti possono ricercare negli indici UDDI e ottenere riferimenti a servizi del loro interesse. Quando il cliente sceglie un servizio, la comunicazione si realizza tramite lo scambio di messaggi SOAP [5]. Standard in fase di definizione, come Business Process Execution Language (BPEL) [1], gestiscono l'esecuzione di processi, mentre altri, come RosettaNet [4] e xCBL [9], definiscono il formato dei documenti che vengono scambiati.

Nonostante le tecnologie presentate si usino per automatizzare l'esecuzione di processi aziendali, il commercio elettronico non si sta diffondendo ampiamente. Un motivo è la mancanza di una struttura per definire i termini aziendali, che devono essere concordati reciprocamente dal fornitore e dal cliente del servizio prima del commercio. Questi termini possono coprire tanti aspetti di una relazione commerciale (il tempo di risposta del servizio, la frequenza delle richieste del cliente, l'affidabilità e la disponibilità del servizio) ed esprimono requisiti che definiscono i diritti e le obbligazioni di ogni partecipante durante l'esecuzione del processo. Il mancato adempimento di questi termini in un contratto può condurre al termine del processo. In questo articolo illustriamo gli approcci esistenti per trattare, creare e gestire contratti aziendali. In particolare, presentiamo il linguaggio WSLA [10][11] sviluppato da IBM e il linguaggio WS-Agreement [8][2] sviluppato dal Global Grid Forum. Entrambi linguaggi definiscono Service Level Agreements (SLA) per interazioni di business.

La struttura di questo articolo è la seguente. Nella sezione 2 introduciamo il concetto di Service Level Agreements nell'ambito Web. La sezione 3 descrive la specifica di WSLA e la realizzazione della sua architettura. Nella sezione 4 vengono presentati la specifica di WS-Agreement e la sua implementazione Cremona. La sezione 5 conclude il lavoro con alcune considerazioni finali.

2. SLA
Prima di presentare i motivi che giustificano l'integrazione dei Service Level Agreements nei sistemi di commercio elettronico, descriviamo l'approccio tradizionale che si usa per invocare servizi Web.

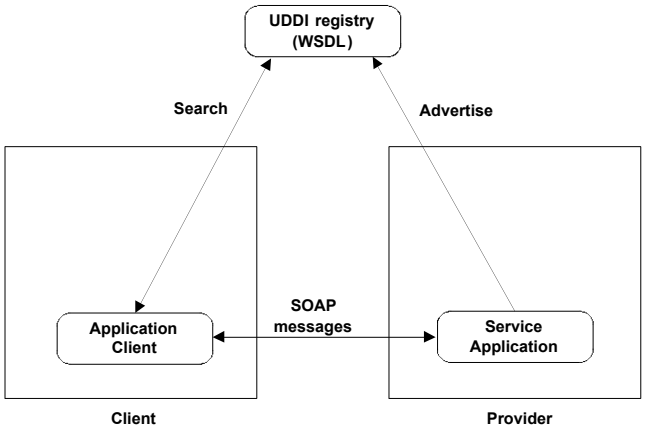

Figura 1. Interazioni fra servizi Web

La Figura 1 descrive uno scenario d'interazioni fra servizi Web basato sul modello client/server. Il fornitore del servizio implementa l'applicazione, espone le sue funzionalità con interfacce definite in WSDL e pubblica il servizio tramite una registrazione in UDDI. Il cliente fa una ricerca nel registro UDDI e riceve informazioni sui servizi che corrispondono ai suoi criteri di ricerca. In seguito, seleziona il servizio di suo interesse e l'interazione può procedere attraverso lo scambio di messaggi codificati in formato SOAP.

In genere, in interazioni aziendali nel mondo reale, il cliente vuole garanzie sull'esito del processo, assicurazioni sulla disponibilità del servizio e sul tempo di risposta. I fornitori probabilmente vorranno addebitare la prestazione di servizi che soddisfano alcuni dei criteri concordati. Per fornire valutazioni obiettive sulla *qualità del servizio (Quality of Service, QoS)*, abbiamo bisogno di estendere il modello client/server presentato sopra con il concetto di Service Level Agreements (SLA). Di conseguenza, il cliente deve poter descrivere i suoi requisiti mentre sta cercando per un servizio, e il fornitore deve essere in grado di valutare le caratteristiche del proprio servizio e di definire priorità sulla ripartizione delle risorse, per fornire il servizio soddisfacendo gli obiettivi definiti nel SLA. Quindi, in questo scenario l'esecuzione del processo viene guidata dall'adempimento del contratto firmato tra il cliente e il fornitore.

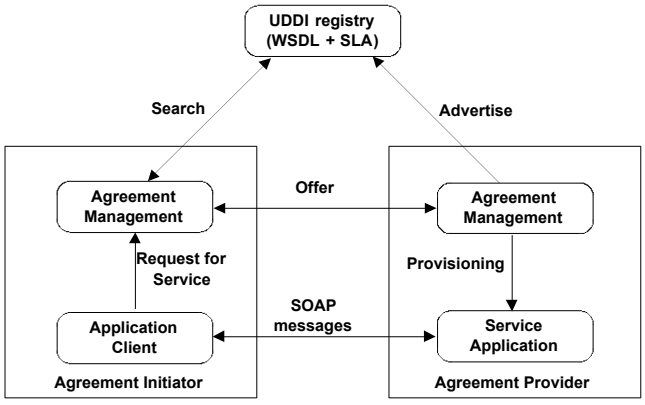

Figura 2. Interazioni fra servizi Web sotto condizioni di SLA

La Figura 2 presenta una possibile interazione che supporta definizione di condizioni SLA. Inizialmente, il fornitore pubblica il servizio e include criteri SLA nella sua descrizione. La parte del cliente che si chiama Agreement Management cerca fra i registri UDDI, riceve una lista con i servizi che soddisfano i suoi criteri di ricerca e costruisce un'offerta di contratto basata sugli elementi SLA ricavati per il servizio che gli interessa. Il servizio che ha il ruolo dell'Agreement Management sulla parte del fornitore misura le sue risorse, valuta le condizioni dell'offerta e accetta o rifiuta l'accordo offerto dal cliente. Questo processo di trattativa può continuare finché le due parti non si concordano sulle condizioni del contratto. Una volta stipulato il contratto le interazioni possono iniziare, e durante la loro esecuzione entrambe le parti sono obbligate a soddisfare le condizioni dell'accordo.

3. WSLA Framework

WSLA è un'architettura proposta e sviluppata da IBM per la specifica e il monitoraggio di Service Level Agreement per Web Services [10]. In questo capitolo presenteremo le caratteristiche salienti del sistema (sezione 3.1), accenneremo brevemente alla struttura del linguaggio formale (sezione 3.2) e descriveremo il toolkit che realizza il sistema WSLA (sezione 3.3).

3.1 Architettura del framework

Descriviamo qui l'architettura del framework WSLA, introducendone dapprima la terminologia appropriata.

3.1.1 Terminologia

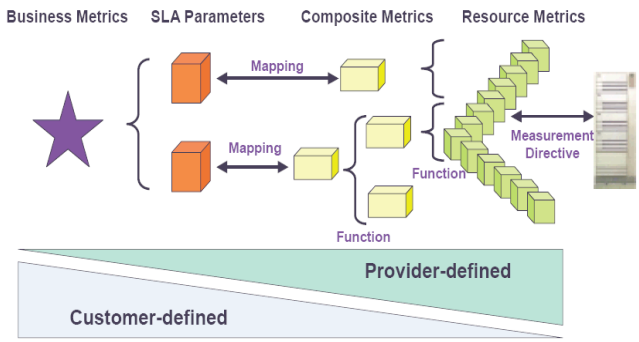

Figura 3. Livelli di gestione delle informazioni

La Figura 3 mostra un'astrazione dei diversi livelli di informazione legata ai SLA in un sistema distribuito:

- **Metriche delle risorse (resource metrics):** ricavate direttamente dalle risorse gestite dal fornitore di servizi (routers, server, strumentazioni varie). Integrate in un WSLA tramite **direttive di misurazione**, assegnate ad ogni risorsa per tenere traccia del contesto e dei comandi necessari per eseguire la misura.
- **Metriche composite (composite metrics)**: composizione di metriche base (o composte a loro volta) tramite un opportuno algoritmo. L'operazione di composizione è solitamente svolta dal fornitore del servizio, ma può essere affidata anche ad una terza parte.
- **Parametri SLA**: parte fondante di un SLA, rappresentano l'interfaccia tra il fornitore e il consumatore di un servizio. Un SLA deve definire tutti i parametri, insieme con i rispettivi intervalli di validità e con i mapping verso una metrica opportuna. Deve essere possibile specificare se la valutazione di un parametro debba avvenire quando questo

soddisfi o meno un obiettivo a livello di servizio. La valutazione dei parametri può essere delegata ad una terza parte, per garantirne l'obiettività.

- **Metriche aziendali (business metrics)**: sono la base della strategia di gestione del rischio dei clienti e restano interne ad essi. Esistono metriche aziendali anche per il fornitore del servizio, che deve verificare che i SLA dei servizi forniti non vadano in conflitto con le proprie politiche aziendali.

Da destra a sinistra, i quattro livelli sono di competenza crescente per il consumatore dei servizi e decrescente per il fornitore.

Il framework WSLA è in grado di gestire in vario modo i quattro diversi tipi di parametro sopra illustrati, anche se si rivolge principalmente agli aspetti tecnici e meno alle metriche aziendali. Del resto, la separazione tra i diversi livelli non è sempre così netta, e ciò è vero in particolare tra i *Parametri SLA* e le *metriche composite*.

3.1.2 Obiettivi del progetto

Il framework WSLA è progettato in maniera flessibile, per essere in grado di operare in ambienti eterogenei nei quali i diversi parametri introdotti in precedenza assumono peso diverso e compaiono in forme diverse. Il linguaggio formale di WSLA permette di definire qualsiasi tipo di accordo indipendentemente dai parametri utilizzati per definire la qualità del servizio (QoS), che possono variare liberamente in base al contesto.

La flessibilità del framework, stando a quanto dichiarato dagli autori, consente di integrarlo facilmente in sistemi di commercio elettronico, e permette tra le altre cose di definire una terza parte incaricata di monitorare il rispetto degli accordi da parte dei due firmatari.

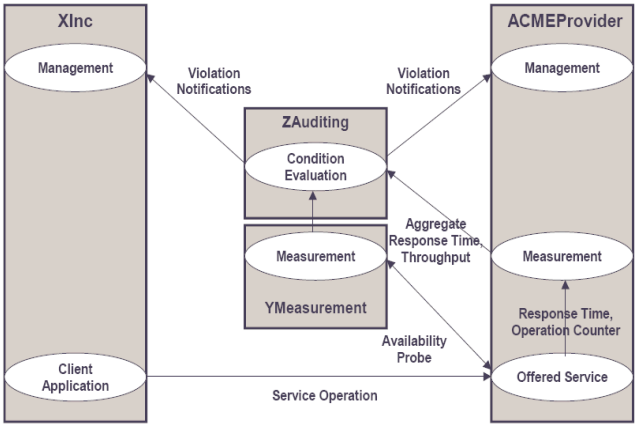

Figura 4. Gestione di fornitori multipli di servizi

L'architettura WSLA è anche in grado di supportare la composizione di servizi da parte di diversi fornitori partendo da un unico SLA. La Figura 4 mostra un esempio di fornitori multipli di servizi, con al centro un servizio di auditing e uno di monitoraggio. In simili situazioni è necessario mostrare ad ogni fornitore solo la parte di agreement che lo vede interessato, secondo quello che viene comunemente chiamato il *need to know principle*; questo principio permette di separare maggiormente tra loro i diversi fornitori dei servizi, che quindi possono essere più facilmente sostituiti, e di rispettare eventualmente la privacy per quanto riguarda lo scambio di informazioni, quando questo possa avere un senso.

3.1.3 Architettura

Analizziamo più in dettaglio l'architettura del framework WSLA. La Figura 5 mostra le interazioni tra i diversi servizi che compongono un WSLA. Seppure il framework possa lavorare con qualsiasi tipo di servizio, si suppone qui che il SLA sia definito per un *web service* descritto da un documento WSDL.
Con riferimento ai numeri indicati in figura, le fasi principali di un sistema di definizione e monitoraggio di un WSLA sono le seguenti:
1. Negoziazione del SLA: un servizio detto di *Establishment* consente a un fornitore e a un consumatore di concordare un accordo sulla qualità di un servizio e di definirne un prezzo. La fase si conclude con la firma dell'accordo da entrambe le parti.
2. Sviluppo del SLA: un servizio detto detto di *Deployment* si occupa di verificare la validità dell'accordo e di distribuirlo ai diversi componenti coinvolti. I componenti devono essere informati dei ruoli dei due firmatari dell'accordo. Lo sviluppo di un SLA comporta la necessità di suddividere l'accordo in diverse parti contenenti solo le informazioni pertinenti (per rispettare il *need to know principle*), definire le interfacce che ogni componente deve esportare e riformulare le specifiche in modo che siano compatibili con le interfacce di configurazione di ciascun componente (che in un sistema eterogeneo possono essere incompatibili tra loro).
3. Misurazione e valutazione dei livelli di servizio: i sistemi di *Misurazione* e di *Valutazione delle condizioni* interagiscono per la raccolta delle misurazioni provenienti dalle risorse e per la verifica del rispetto degli accordi definiti nel SLA. In particolare, il servizio di misurazione tiene traccia della configurazione del sistema e delle informazioni di run-time sulle metriche coinvolte nel SLA; il servizio di valutazione delle condizioni, invece, serve a confrontare i parametri misurati con le soglie definite nel SLA e a informare il servizio di gestione.
4. Azioni correttive: il servizio di valutazione delle condizioni, una volta rilevata la violazione di un qualche obiettivo (SLO), notifica con un messaggio il servizio di *Management*, che applicherà le eventuali azioni correttive descritte nel SLA (risoluzione automatica dei problemi, notifica degli allarmi, ecc.). Prima di applicare una qualsiasi azione correttiva, il servizio di gestione chiede l'approvazione alla *Business Entity*, che è incaricata di verificare tutte le proposte del sistema automatico di gestione per approvarle o modificarle. Nel framework WSLA questa "entità aziendale" è un componente concettuale che modella le conoscenze, gli obiettivi e le politiche aziendali di una delle due parti coinvolte nell'accordo, e che in genere non viene condiviso. Le informazioni che alimentano la business entity possono provenire da fonti molto diverse ed eterogenee (sistemi SAP, dipendenti dell'azienda, ecc.).
I due componenti coinvolti nella fase di correzione costituiscono il punto critico del sistema, e diventano molto complicate man mano che la gestione aziendale si fa sofisticata.
5. Terminazione del SLA: è possible definire alcune condizioni che provocano la terminazione di un agreement, come ad esempio il mancato rispetto di una clausola da parte di una delle due parti o il raggiungimento di un certo timeout.

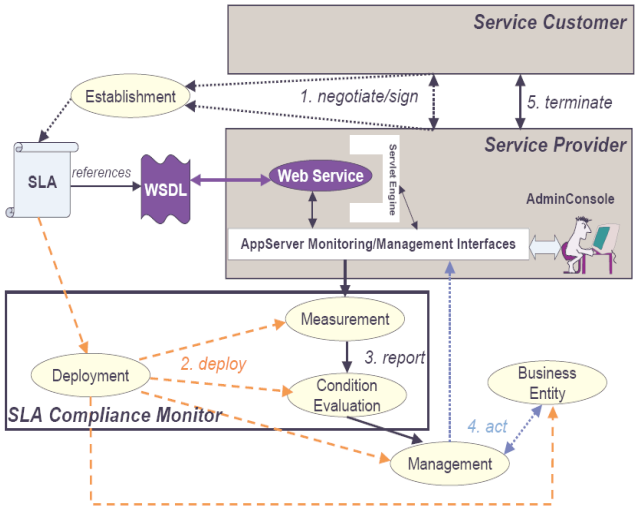

Figura 5. Interazioni tra i servizi di un WSLA

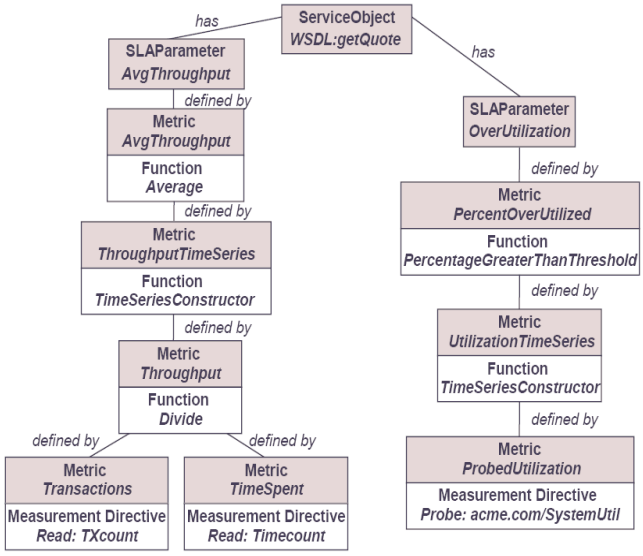

Figura 6. Esempio di descrizione di un servizio

3.2 Il linguaggio WSLA

Segue ora una descrizione concisa delle parti più rilevanti del linguaggio WSLA. È possibile vedere la specifica completa del linguaggio in [11].

Il linguaggio WSLA è definito da uno schema XML Schema; nelle sezioni seguenti analizzeremo i suoi elementi principali.

3.2.1 Soggetti

La prima sezione di un documento WSLA definisce tutte le parti coinvolte nell'accordo. I *firmatari* (fornitore e consumatore del servizio) sono descritti dai rispettivi identificatori e dalle interfacce pubbliche. I soggetti di *supporto* sono descritti anche dalla descrizione degli sponsor.

3.2.2 Servizio

La seconda sezione del documento descrive il servizio in termini di parametri osservabili. Il componente di misurazione dei livelli accede a questa parte per effettuare le misurazioni.

Le informazioni che compongono questa parte servono a descrivere i *parametri* del SLA, ad associarli ad un preciso *servizio*, a definirne una *metrica* (misurata o calcolata) e a determinare le modalità di *accesso* alla misura ottenuta.

Gli *oggetti* per i quali si definiscono i parametri sono le unità atomiche con le quali il nostro servizio è in grado di lavorare. Nel contesto dei web services, ad esempio, tali oggetti possono corrispondere alle singole *operazioni* di un WSDL. I *parametri* definiti su un oggetto sono definiti da un nome, un tipo e una unità di misura. Esempi di parametri possono essere il tempo di risposta, il throughput o la disponibilità di un servizio. I parametri possono essere definiti da un insieme di *metriche*, che possono essere composte come si è discusso nel capitolo 3.1.1.

La Figura 6 mostra un esempio di descrizione di servizio: l'oggetto (ServiceObject) a cui siamo interessati è l'operazione "getQuote" definita in un WSDL. Definiamo per questo oggetto due diversi parametri, ognuno definito da una metrica composta.

3.2.3 Obblighi

La terza sezione del documento WSLA descrive i vincoli e le garanzie che devono essere rispettati dai parametri del servizio. Il componente di valutazione delle condizioni accede a questa parte per effettuare le verifiche. Esistono due diversi tipi di obbligazione:

1. Service Level Objective

Gli obiettivi a livello di servizio rappresentano una sorta di "promessa" circa lo stato di un parametro. Solitamente è il fornitore del servizio a farsi carico del mantenimento di un SLO, anche se esistono casi in cui è il consumatore a dover garantire il mantenimento di un certo obiettivo.

Un SLO è definito dall'insieme di *parte* coinvolta, *periodi* di validità ed una *espressione* che descrive l'obiettivo da mantenere (tipicamente, un predicato di logica del prim'ordine). È possibile completare un SLO precisando un *trigger* che attivi la valutazione dell'espressione solo all'occorrenza di un particolare evento.

2. Action Guarantees

Le "garanzie di azione" costituiscono l'impegno al compimento di una data attività a fronte del verificarsi di una certa precondizione. Può essere coinvolto in questo tipo di obbligo qualsiasi soggetto, firmatario o di supporto.

Le AG sono formate dalla descrizione del *soggetto* coinvolto, da una espressione logica che descrive la *precondizione* da verificare, da un evento di *trigger o* da uno *schedule* per governare la valutazione della precondizione e dalla descrizione *dell'azione* da compiere. Tra le precondizioni è possibile includere anche la *violazione* di un altro obbligo.

Le azioni possono essere eseguite dalla parte interessata o da una terza parte. È possibile anche definire una *modalità* di esecuzione relativa al valore della precondizione (sempre con precondizione soddisfatta, solo nell'istante in cui la precondizione diventa soddisfatta, solo nell'istante in cui la precondizione cessa di essere soddisfatta).

3.3 Implementazione

Il primo prototipo del framework WSLA è stato introdotto nel *Web Services ToolKit*, un ambiente integrato di progetto, sviluppo e gestione di web services realizzato dalla IBM stessa. Recentemente, il toolkit si è evoluto in una nuova versione, ed è diventato parte di un sistema più ampio denominato *Emerging Technologies ToolKit*, di cui costituisce il componente ETTK-WS [12]. Il progetto ETTK vuole riunire in un unico ambiente tutte le tecnologie emergenti in fase di sviluppo, e il tema dei SLA rientra in questa definizione.

La versione più recente di ETTK-WS (versione 2.3) vede l'abbandono del formato WSLA a favore di WS-Agreement (capitolo 4), uno standard che rappresenta l'evoluzione di WSLA e che oltre all'IBM vede coinvolti importanti soggetti come il W3C, la Hewlett-Packard e altri. Il riconoscimento da parte delle grandi società della necessità di cooperare verso il raggiungimento di un accordo comune in temi nuovi e importanti come quello dei SLA fa ben sperare per il futuro di queste tecnologie, che nascono anche per superare i limiti introdotti dall'incompatibilità tra sistemi di produttori diversi.

4. WS-Agreement

WS-Agreement è la proposta da Global Grid Forum per la creazione e il monitoraggio d'accordi fra il fornitore e il consumatore di un servizio. Permette (a) ai clienti di esprimere i loro requisiti sui QoS in un formato formale, comprensibile dalla macchina, e (b) ai fornitori di valutare le proprie risorse e in base alle proprie potenzialità di accettare o rifiutare la richiesta per un accordo.

In questo capitolo introduciamo il linguaggio WS-Agreement (Sezione 4.1), presentiamo la descrizione dell'architettura del WS-Agreement framework denominata Cremona (sezione 4.2), e concludiamo con un breve riferimento al toolkit ETTK che realizza Cremona (sezione 4.3).

4.1 Descrizione del Linguaggio

In questa sezione, presentiamo il linguaggio di WS-Agreement. In particolare, presentiamo (a) le astrazioni per definire accordi e prototipi di accordi, (b) un protocollo per la creazione di un accordo basato su un prototipo, e (c) il protocollo per il monitoraggio della conformità di accordi.

4.1.1 Struttura di Accordi e Template

Gli accordi fra il cliente e il fornitore di un servizio si specificano in un linguaggio basato su XML e si conformano alla struttura presentata in Figura 7. Un *template* viene generato dal fornitore e si usa come guida di riferimento per il cliente per costruire un nuovo accordo. Un *accordo* ha una struttura simile a quella del *template*. La specifica WS-Agreement fornisce solo la struttura di accordi e template, mentre l'implementazione dipende dal dominio del fornitore.

Con riferimento alla figura:

- La sezione *Context* contiene informazioni (metadati) sull'accordo stesso: i partecipanti coinvolti, il template di riferimento per creare un nuovo accordo, il tempo di validità dell'accordo e riferimenti ad altri accordi associati.
- La sezione *Service Description Terms (SDT)* specifica le operazioni del servizio che saranno eseguite se la conformità all'accordo sarà verificata. La definizione delle operazioni all'interno di questa sezione e la loro corrispondenza ad un servizio reso disponibile dal fornitore può essere espressa tramite riferimenti (a) formali in qualsiasi linguaggio accettato (ad esempio EPR), (b) non formali alle proprietà o alla descrizione del servizio. La specifica non definisce il linguaggio per descrivere i riferimenti finali al servizio, né il meccanismo per tracciare un servizio in base all'identificazione delle sue proprietà; essa fornisce solo la rappresentazione astratta che può essere implementata con elementi standard o specifici del dominio.
- La sezione *Guarantee Terms (GT)* definisce la modalità di fornitura delle operazioni del servizio. In questa sezione sono riportati i *Service Level Objectives (SLO)* (ad esempio il tempo medio di risposta per i servizi descritti nelle sezioni SDTs), le condizioni che devono essere rispettate perché un dato SLO possa essere fornito (ad esempio, il tasso di richiesta deve essere inferiore ad una soglia predefinita) e un valore aziendale che rappresenta l'importanza di raggiungere l'obiettivo descritto dal SLO nell'accordo (ad esempio, un valore aziendale alto implica che il fornitore usi più risorse per eseguire le operazioni descritte nella sezione SDT, nel caso in cui le risorse disponibili non possano soddisfare il SLO definito).
- La sezione *Constraint* si presenta solo nella struttura di un template e ha lo scopo di imporre vincoli ai valori accettati per riempire i campi dei termini presentati sopra. Per esempio, il valore accettato per il numero di CPUs usate per soddisfare i SLO (e.g. tempo medio di risposta) definiti per la fornitura delle operazioni del servizio, non può avere valore più di 7. Offerte di accordi che richiedono un numero di CPUs più di 7 non saranno accettate. Questa sezione si usa per verificare la conformità di una richiesta per la creazione di un accordo con un dato template.

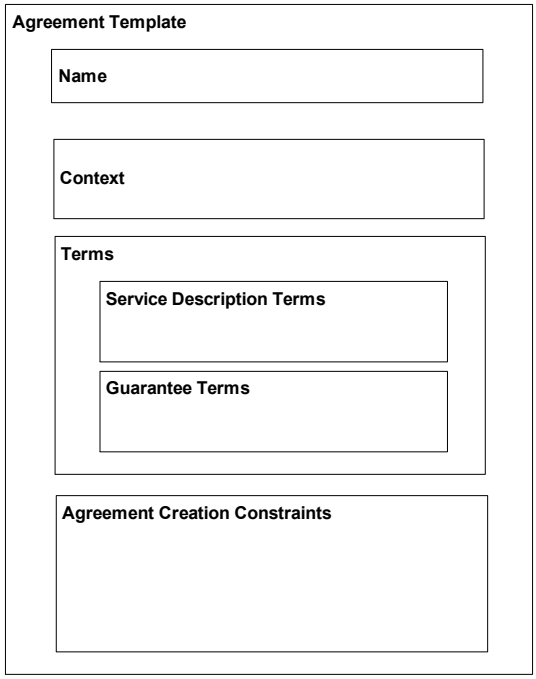

Figura 7. Struttura di WS-Agreement Template

WS-Agreement specifica la struttura degli accordi senza fornire la definizione di SLO, condizioni di qualifica e valori aziendali. La loro definizione si realizza dal fornitore usando linguaggi e algoritmi scelti da lui. Una volta definiti questi elementi vengono messi nella struttura di WS-Agreement per costruire i termini dell'accordo (Guarantee Terms). La loro combinazione si presenta all'iniziatore dell'accordo con la forma di un template. L'iniziatore deve solo riempire questi termini con valori validi per creare un nuovo accordo.

4.1.2 Modello di Interazione fra Accordi

WS-Agreement fornisce un protocollo di trattative necessarie per creare nuovi accordi. Le parti coinvolte sono due. I partecipanti hanno il ruolo dell'iniziatore o del fornitore dell'accordo, indipendentemente dai ruoli che hanno nelle interazioni fra servizi. Quindi l'iniziatore dell'accordo può essere sia il fornitore sia il cliente del servizio.

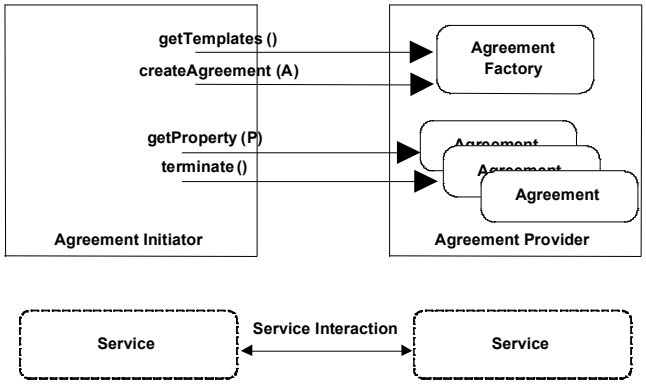

Figura 8. WS-Agreement Interaction Model

Il modello di interazione (Figura 8) specifica solo la creazione di accordi in base a un template. Il fornitore dell'accordo espone le operazioni createAgreement(A) e getTemplates(). Il componente denominato *Agreement Factory* risponde all'invocazione di queste operazioni, rispettivamente, creando nuovi accordi e recuperando i template disponibili. L'ordine dell'invocazione di queste operazioni è la seguente: l'iniziatore dell'accordo recupera i template proposti dal fornitore (chiama l'operazione getTemplates()), e crea un'offerta riempiendo i campi del template scelto dalla lista precedente (createAgreement(A)).

Se l'offerta viene accettata dal fornitore, un nuovo accordo si crea. Il modello definisce i *port type* per il monitoraggio del servizio e dello stato delle garanzie del nuovo accordo. Il monitoraggio di questi termini viene fatto mediante l'operazione getProperty(P). In particolare, i servizi descritti nella sezione SDTs dell'accordo possono essere in stato *NotReady* (l'esecuzione del servizio non è stata iniziata), *Ready* (il servizio è pronto per essere eseguito), *Processing* (il servizio è in esecuzione) e *Completed* (l'interazione con il servizio è stata completata). Invece, le garanzie fornite dall'accordo possono essere in stato *Fulfilled*, *Violated* e *NotDetermined*. Questi valori indicano se una garanzia è stata mantenuta, disattesa o la sua verifica è in corso. In questo modo, si verifica l'adempimento del contratto durante l'esecuzione del servizio.

WS-Agreement specifica il modello astratto di interazione fra l'iniziatore e il fornitore, con le seguenti limitazioni:

1. Non specifica l'implementazione delle parti coinvolte. Nell'ambito di servizi Web, spesso il fornitore del servizio è anche il fornitore dell'accordo. Invece, nell'ambito di servizi Grid il cliente di un servizio sceglie il fornitore adatto alle sue esigenze. In questo caso, il cliente definisce gli accordi e accetta l'offerta più conveniente.
2. Non è un protocollo completo di trattative. Deve essere esteso per fornire operazioni più complesse, e.g. l'integrazione di informazioni SLA nella pubblicazione di un servizio, le trattative delle condizioni di termini per creare un accordo, e l'identificazione di un accordo nei messaggi scambiati per eseguire un servizio.
3. Non definisce se il monitoraggio viene fatto da parti di supporto o direttamente dall'iniziatore. Il monitoraggio delle risorse per verificare l'adempimento del contratto può essere fornito da un partecipante esterno ma l'iniziatore si notifica solo tramite i componenti esposti dal fornitore dell'accordo.

4.2 Cremona: Architettura di WS-Agreement

Nello scenario di interazioni fra servizi Web basate su accordi, tanti argomenti rimangono in sospeso. La specifica WS-Agreements definisce la struttura e il modello necessario per descrivere e scambiare i requisiti del cliente e la potenzialità del fornitore, per ottenere un accordo tra essi; tuttavia, non fornisce l'implementazione dei termini dell'accordo (SDT e GT). L'architettura Cremona definisce questi componenti, implementa le funzionalità per il supporto del protocollo WS-Agreement e per il monitoraggio delle interazioni basate su accordi.

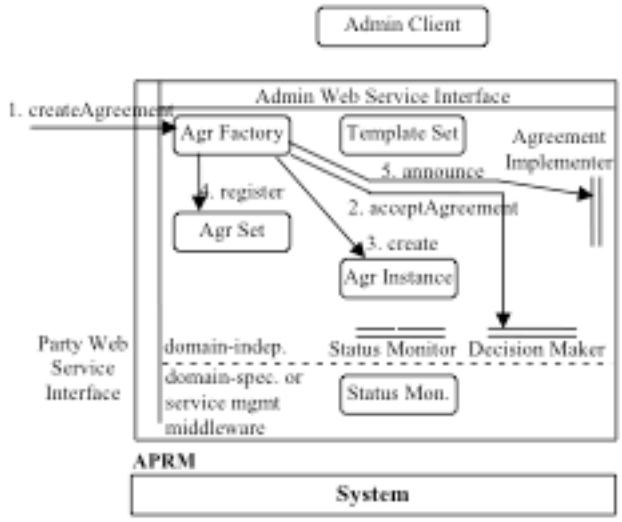

Figura 9. APRM struttura del fornitore dell'accordo

L'architettura si compone di tre livelli. Al primo livello, un insieme di funzioni chiamato *Agreement Protocol Role Management (APRM)* implementa il protocollo WS-Agreement. Di conseguenza, tutte le operazioni e i port type del protocollo vengono definiti per entrambi i partecipanti: per l'iniziatore e per il fornitore dell'accordo. L'obiettivo di APRM è lo scambio dei requisiti dell'iniziatore e delle potenzialità del fornitore finché tutti i due sono d'accordo sui termini del contratto. Per raggiungere questo obiettivo, dalla parte del fornitore sono presenti i componenti necessari per la creazione di contratti in base a un template: *Agreement Factory, Template Set, Agreement Set* e *Agreement Instance* (Figura 9). Inoltre, l'accettazione o il rifiuto di un'offerta per un accordo richiede il monitoraggio dei

servizi corrispondenti. Il fornitore deve essere in grado di analizzare le potenzialità delle risorse del sistema e valutare l'esecuzione dei servizi richiesti. Questo si realizza tramite il componente *Status Monitor Implementation*. Dalla parte dell'iniziatore dell'accordo, l'obiettivo del componente APRM è il recupero dei template dal fornitore, la creazione di un'offerta per un accordo, la richiesta per un'offerta al fornitore, e l'archivio degli accordi accettati. La realizzazione di queste operazioni si raggiunge con i componenti *Agreement Initiator, Factory Set, Template Processor, e Agreement Set*.

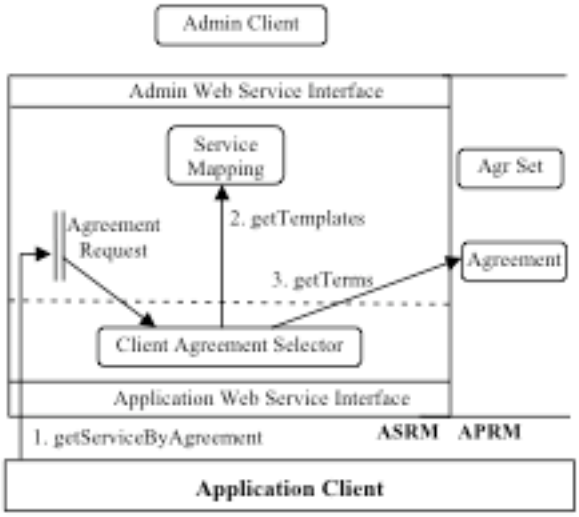

Figura 10. ASRM struttura dell'iniziatore dell'accordo

Il secondo livello si chiama *Agreement Service Role Management (ASRM)* ed è composto da funzionalità per collegare i termini di un accordo al sistema di servizi. Questo livello fornisce tutti i meccanismi necessari al fornitore per creare template e valutare offerte di accordi in base alle capacità di servizi. Permette il monitoraggio delle proprietà di servizi durante la loro esecuzione e azioni da intraprendere per assicurare i SLO in un accordo. Dalla parte dell'iniziatore (Figura 10), ASRM implementa i componenti necessari per il recupero di servizi che soddisfano i suoi requisiti, per ricavare i template dei servizi, e per creare un'offerta per un accordo da un template scelto dalla lista.

L'ultimo insieme di funzioni si chiama *Strategic Agreement Management (SAM)*; esso fornisce le informazioni per implementare i livelli APRM e ASRM. Specificamente, i suoi componenti osservano e misurano l'ambiente dei servizi, e dai risultati ricavati stabiliscono quali template debbano essere modificati e quale sia l'entità delle loro modifiche. L'architettura di Cremona attualmente non fornisce questo set di funzionalità.

4.3 Implementazione di WS-Agreement

L'architettura di WS-Agreement è stata implementata da IBM e appartiene alla collezione *Emerging technologies Toolkit* (ETTK) [13]. ETTK è un'iniziativa dall'IBM con l'obiettivo di fornire nuove tecnologie su Web da vari gruppi di ricerca. Mediante questo strumento, i SLA vengono definiti tramite il linguaggio di WS-Agreement, e vengono inseriti in progetti reali dall'ambito di servizi Web e di tecnologie Grid.

Come già accennato nella sezione 3.3, solo l'ultima versione di ETTK supporta WS-Agreement, che sostituisce il sistema WSLA. Nella sezione seguente mettiamo in risalto le differenze tra le due architetture.

4.4 Confronto tra WSLA e WS – Agreement.

I sistemi WSLA e WS-Agreement risolvono in modo molto simile le problematiche legate ai Service Agreement.

Lo schema dei documenti che descrivono gli accordi è molto simile: entrambi prevedono uno spazio per la descrizione dei soggetti coinvolti e dei servizi forniti, insieme agli indici di qualità per ogni servizio. Così come WSLA, anche WS-Agreement permette di coinvolgere terze parti nella fornitura dei servizi e soprattutto nelle fasi di monitoraggio dei parametri dell'accordo. A differenza di WSLA, tuttavia, in WS-Agreement le terze parti sono mascherate da un'unica interfaccia verso il fornitore del servizio.

WSLA è orientato alla definizione di garanzie che devono essere fornite per l'adempimento di un accordo. Il linguaggio specifica queste garanzie sugli attributi di servizi e propone la loro modellazione in modo uniforme, indipendente dal dominio del fornitore. Dall'altra parte, WS-Agreement definisce accordi ad un livello distinto dal livello del servizio ed è orientato alla gestione di accordi (creazione, monitoraggio). Infatti, gli accordi in WS-Agreement vengono definiti indipendentemente dalla struttura dei servizi esposti dal fornitore. Il vantaggio di questo approccio sta nel fatto che in WS-Agreement è possibile definire accordi su servizi esistenti (servizi generici e non solo di natura Web) senza cambiare la loro implementazione.

Ciò che differenzia maggiormente WS-Agreement da WSLA è la definizione di *template* e di *Agreement Factory*; queste due entità permettono di definire meglio le interazioni tra le parti per la stipula di un contratto, operazione che non era formalmente specificata nella descrizione del framework WSLA.

Inoltre, WS-Agreement definisce solo le interfacce degli accordi, lasciando totale libertà ai fornitori dei servizi per quanto concerne la loro implementazione. WSLA invece definiva più formalmente la struttura delle metriche per la misurazione dei parametri.

Una caratteristica di WSLA non riportata in WS-Agreement invece è la distinzione tra *Service Level Objectives* e *Action Guarantees*, che permetteva una più definizione degli accordi. I SLO di WS-Agreement, inoltre, possono contenere un indice di priorità che dà una stima del valore aziendale di un dato obiettivo; questa scelta è in un certo senso contraria al principio di isolamento delle metriche di Business perseguito dal sistema WSLA, che isolava nel modulo chiamato "*Business Entity*" tutte quelle variabili nascoste alla controparte che costituiscono il valore di un'azienda. Naturalmente, entrambi gli approcci (rendere noto il valore relativo di un obiettivo o nascondere completamente i fini ultimi) presentano vantaggi e svantaggi. Banalmente, conoscere le diverse priorità degli obiettivi che compongono un accordo permette di reagire con più prontezza alle violazioni degli obiettivi più critici, mentre nascondere tutte le politiche aziendali dietro un modulo opaco può essere la scelta migliore nel caso di aziende con una identità molto forte da preservare. Del resto, l'indice prioritario previsto per gli SLO in WS-Agreement è solo opzionale, e ciò permette di venire incontro ad entrambe le esigenze.

5. Conclusioni

Abbiamo visto come i Service Level Agreement possano essere integrati nel contesto dei servizi web per il commercio elettronico,

in maniera flessibile e indipendente dal contesto specifico. Abbiamo presentato due diverse architetture – WSLA e WS-Agreement – che affrontano questo problema con approcci in realtà molto simili tra loro, e ne abbiamo descritto l'implementazione.

Abbiamo sottolineato anche l'importanza di raggiungere uno standard condiviso tra le aziende coinvolte nella definizione e fruizione dei servizi; questa considerazione ha portato gli sviluppatori di WSLA ad abbandonare il progetto per supportare invece WS-Agreement.

6. REFERENCES

[1] BPEL4WS site http://www-128.ibm.com/developerworks/library/specification/ws-bpel/

[2] Ludwig, H., Dan, A., Kearney, R. *Cremona: An Architecture and Library for Creation and Monitoring of WS-Agreements*. ICSOC, New York, USA, 2004.

[3] Global Grid Forum (GGF) site http://www.gridforum.org/.

[4] RosettaNet site http://www.rosettanet.org/.

[5] Simple Object Access Protocol (SOAP) 1.1 http://www.w3.org/TR/SOAP/

[6] Universal Description Discovery and Integration (UDDI) http://www.uddi.org

[7] Web Services Definition Language (WSDL) 1.1, http://www.w3.org/TR/wsdl.

[8] WS-Agreement Specification site http://www.gridforum.org/Meetings/GGF11/Documents7draft-ggf-graap-agreement.pdf

[9] xCBL site http://www.xcbl.org/.

[10] Keller, A and Ludwig, H. The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services. *IBM research report, July 2002*.

[11] H. Ludwig, A. Keller, A. Dan, R. Franck, and R.P. King. *Web Service Level Agreement (WSLA) Language Specification*. IBM Corporation, July 2002.

[12] WSTK, *http://www.alphaworks.ibm.com/tech/webservicestoolkit*, IBM Corporation, July 2000.

[13] ETTK-WS, *http://www.alphaworks.ibm.com/tech/ettkws*, IBM Corporation, May 2005