Modellazione della QoS in ambienti Web-Service con applicazioni di Video Streaming

F. Buccafurri[3], L. Console[4], P. De Meo[3], M.G. Fugini[2], A. Goy[4], G. Lax[3], P. Lops[1], S. Modafferi[2], B. Pernici[2], C. Picardi[4], D. Redavid[1], G. Semeraro[1], and D. Ursino[3]

[1] Università degli Studi di Bari, Dipartimento di Informatica
[2] Politecnico di Milano, Dipartimento di Elettronica e Informazione
[3] Università degli Studi Mediterranea di Reggio Calabria, Dipartimento di Informatica Matematica Elettronica e Trasporti
[4] Università degli Studi di Torino, Dipartimento di Informatica

Sommario Il Progetto PRIN Quadrantis si propone di modellare gli aspetti legati alla cooperazione degli e-Service concentrandosi sulle problematiche legate alla QoS e sulle possibili azioni di recovery che, a valle di una diagnosi per determinare la causa del problema, possano implementare politiche di adattività. La modellazione degli e-service è basata sull'estensione dei linguaggi disponibili in letteratura con la rappresentazione dei parametri di qualità associati ai servizi. Il contributo fornito è innovativo in quanto considera la rappresentazione semantica dei parametri di qualità sia a livello di Web Service sia a livello di "oggetto erogato". In questa accezione l'oggetto dell'e-service diventa particolarmente interessante se esso stesso è dotato di parametri di qualità e se, al di fuori della sua richiesta che viene gestita con il protocollo dei Web-Service, è possibile realizzare politiche adattive che consentano il monitoraggio e la gestione avanzata della QoS durante la sua erogazione. Si è quindi considerato in dettaglio lo specifico caso in cui l'oggetto di uno o più servizi coinvolti nel processo di business sia un video streaming, che è un esempio in cui la QoS riveste particolare importanza e per i quali esistono sistemi adattivi in grado di effettuarne una gestione avanzata. Si è quindi proposto un framework a due livelli in grado di gestire in maniera adattiva la QoS in ambienti Web-Service cooperativi caratterizzati da applicazioni di video streaming.

1 Introduzione

Recentemente un'interessante direzione della ricerca è quella della modellazione e gestione della QoS in ambiente Web-Service. Accanto al criterio funzionale, la QoS rappresenta infatti la guida nella scelta e nella gestione dei servizi consentendo di sfruttare al meglio le caratteristiche di dinamicità, flessibilità e possibilità di composizione, che sono gli aspetti più interessanti della SOA. Il progetto Quadrantis si propone di modellare la QoS dei Web-Service introducendo l'innovativo concetto di "oggetto erogato" dal Web-Service distinguendone i parametri di qualità da quelli propri del componente erogante e, al tempo stesso, di fornire

una piattaforma per la gestione adattiva della cooperazione fra diversi Web-Service. Per adattività, nel contesto del progetto, si intende la capacità del sistema di monitorare un insieme predefinito di parametri di qualità reagendo ad eventuali violazioni di questi parametri attraverso opportune politiche di recovery. La scelta dell'insieme di operazioni di recovery da implementare è fatta a valle di una fase di diagnosi in grado di individuare la sorgente del fault che non necessariamente è nello stesso attore/componente che intercetta il sintomo.

Particolarmente interessante è il caso in cui l'oggetto dell'operazione del Web-Service è l'erogazione di uno video streaming. In questo caso infatti esiste una diffusa letteratura sulla relativa QoS ed esistono sistemi ad agenti in grado di implementare politiche adattive in caso di degrado della QoS. E' quindi possibile descrivere come la QoS del Web-Service e quella dell'oggetto si influenzano. Lo scenario scelto per testare l'approccio proposto prevede quindi un processo di business all'interno del quale siano presenti dei Web-Service che erogano un video streaming.

L'articolo è strutturato nel seguente modo: nella Sezione 2 viene discusso il problema della qualità nei Web-Service, la Sezione 3 è dedicata alla presentazione dell'ontologia sviluppata mentre nella Sezione 4 viene presentato il framework per la gestione contestuale della QoS nei Web-Service e nello streaming. La Sezione 5 è dedicata alle conclusione e agli sviluppi futuri.

2 La qualità nei Web-Service

Una delle sfide più significative per rendere la Service Oriented Architecture (SOA) realmente efficace nei moderni sistemi di business è la gestione della Quality of Service (QoS). Come già accennato, infatti, la QoS rappresenta, accanto al criterio funzionale, la guida nella scelta e nella gestione dei Web-Service, e consente di sfruttare al meglio le caratteristiche di dinamicità, flessibilità e possibilita di composizione della SOA. In generale la QoS viene gestita attraverso contratti stipulati fra un provider ed un client. Attraverso un contratto il provider si impegna a garantire una certa QoS ed il client a pagare un certo prezzo P.

All'interno di un processo di business una violazione della QoS può avere svariate origini e sistemi adattivi che supportano il *Self-Healing* sono in grado di capire come e perchè si sia generata tale violazione e di reagire conseguentemente. In questa accezione le violazioni di QoS possono essere interpretate come fault di una specifica istanza di un processo; è quindi parte integrante delle capacità adattive del sistema la possibilità di reagire correttamente a queste situazioni.

In letteratura esistono diversi approcci per la modellazione e per la gestione della QoS. Un interessante confronto fra alcuni modelli di qualità è presentata in [16]. In generale si sta sempre di più affermando l'uso di ontologie per la rappresentazione della QoS perchè supportano: i) un vocabolario condiviso utilizzabile dalle macchine; ii) una rappresentazione della conoscenza interpretabile dalle macchine; iii) la possibilità di fare reasoning automatico sulle descrizione della QoS.

Esistono dei sistemi il cui focus è l'utilizzo di ontologie per la gestione della QoS, ad esempio [5]. Questa attenzione alle ontologie porta ad avere modelli ricchi, ma sistemi reali piuttosto semplici. E' il caso del framework presentato in [11] che è basato su agenti che utilizzano l'ontologia per selezionare i servizi richiesti dall'utente; un interessante tentativo di integrare una rappresentazione ontologica all'interno di un sistema per la gestione della QoS è presentato in [14].

Altri modelli si focalizzano invece sulla gestione della QoS assumendone una rappresentazione piuttosto semplice ed essenzialmente basata su modelli scritti in XML, ma non necessariamente ontologici. In [2] il focus è la selezione dinamica delle risorse per la massimizzazione della QoS per l'utente finale. Le dimensioni di qualità considerate sono poche e semplici e il modello per la loro rappresentazione è un semplice file XML. In [6] vengono rappresentate la QoS offerta da un servizio e quella richiesta dall'utente e, attraverso l'uso del linguaggio Ws-Policy, vengono implementate delle politiche di matching che tendono a massimizzare la qualità come percepita dall'utente. In [10] gli autori propongono un framework per supportare l'esecuzione di servizi composti con una logica goal-oriented. Il lavoro [17] presenta una piattaforma middleware per la selezione e l'esecuzione di Web-Service composti in grado di supportare l'ottimizzazione sia a livello locale che a livello globale.

I modelli basati su ontologie, sopra presentati, sono generici ed hanno una naturale estensibilità che li rende capaci di supportare l'approccio proposto in questo articolo. Non esistono però modelli di gestione che esplicitamente distinguano tra qualità del servizio e qualità dell'oggetto, che è invece molto importante sia oggettivamente, che per la percezione globale della qualità che ha l'utente.

3 L'ontologia per la descrizione della QoS

L'evoluzione del Web è legato in modo imprescindibile alla rappresentazione della semantica delle informazioni in esso presenti. Tale rappresentazione permetterà lo scambio di informazioni tra applicazioni eterogenee allo scopo di abilitare le applicazioni ad automatizzare le operazioni che oggi possono essere eseguite solo manualmente [4]. Tale evoluzione deve avvenire necessariamente in modo graduale, per cui è auspicabile che le comunità di ricerca che lavorano nell'ambito del Web si confrontino con le metodologie e tecnologie sviluppate per il Semantic Web. In quest'ottica nell'ambito del progetto Quadrantis sono stati adottati costrutti ontologici per la rappresentazione della semantica associata ai parametri di qualità descritti in Tabella 1. In generale, l'ontologia costruita descrive attributi di qualità arbitrari, la natura delle associazioni che esistono tra di loro e le modalità con cui è possibile misurarle.

Le principali classi ontologiche, riportate anche in Figura 1[1], utilizzate per descrivere i parametri in esame sono le seguenti:

[1] Realizzata con il plugin per protegè Jambalaya, disponibile al sito: http://www.thechiselgroup.org/jambalaya

Nome	Definizione
Tempo di risposta	Ritardo tra l'istante di tempo in cui una richiesta è mandata e quello quando il risultato è ottenuto.
Qualità dei dati	Parametro relativo ai dati e ai database a cui il servizio fa riferimento, è tipicamente suddiviso in tre sottocategorie:
Timeliness	Grado di aggiornamento dei dati.
Accuracy	Valutazione della corrispondenza del dato rispetto a un dato di riferimento considerato corretto.
Completeness	Copertura dei dati scambiati rispetto alla totalità delle informazioni.
Prezzo	Importo che chi richiede un servizio deve pagare al Provider dello stesso per l'invocazione.
Availability	Probabilità che una data operazione sia realmente accessibile al momento della richiesta.
Reputation	Grandezza definita come il rapporto fra il numero di invocazioni del servizio che rispettano la QoS e il totale delle invocazioni del servizio.

Tabella 1. Parametri di qualità del livello Web-Service

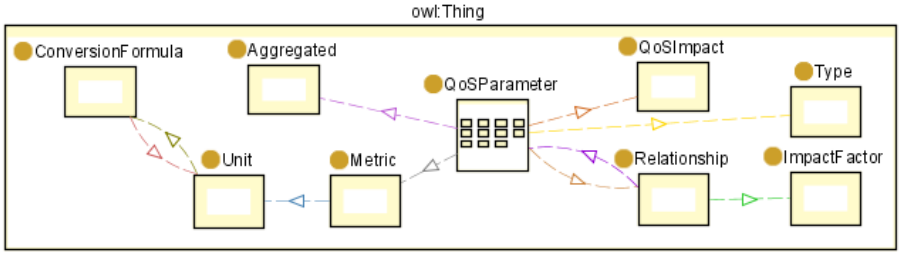

Figura 1. Le classi dell'ontologia per la rappresentazione dei parametri di qualità

- QoSParameter - tutti i parametri in esame sono sottoclassi di QoSParameter. Le relazioni di proprietà di questa classe possono essere misurabili o non misurabili e possono avere relazioni tra di loro.
- Metric - questa classe definisce la modalità con cui è possibile assegnare un valore al parametro di qualità, ed è associata alla classe QoSParameter attraverso la object property hasMetric. Ogni individual di questa classe ha due datatype property, MetricType e Value. La relazione di proprietà hasUnit lega la classe Metric alla descrizione dell'unità di misura associata per misurare le quantità del QoSParameter.
- QoSImpact - questa classe abilita alla rappresentazione di indicatori sulla qualità percepita dall'utilizzatore del servizio sulla base del valore del QoSParameter. La proprietà ad esso correlata (hasQoSImpact) abiliterebbe un sistema alla stima automatica del grado di soddisfazione dell'utente inerente un dato valore di QoSParameter.

Le relazioni di proprietà più significative per queste classi sono le seguenti:

- hasNature - indica la natura del parametro, statica o dinamica. Parametri di qualità definiti a priori che non cambiano durante l'intera sessione del servizio sono detti statici, quelli che possono variare durante l'esecuzione del servizio sono detti dinamici.
- isAggregated - esprime il legame tra due o più parametri di qualità.
- hasRelationship - lega il parametro di qualità alla classe Relationship, che descrive il modo in cui il parametro è correlato con gli altri parametri.

L'analisi eseguita sui parametri di qualità selezionati non ha evidenziato relazioni di proprietà caratterizzanti i singoli parametri, ma la rappresentazione ontologica, descritta utilizzando OWL [12], permette di specializzare le loro definizioni

non appena se ne evidenzia il bisogno. L'utilizzo delle ontologie nel contesto del progetto Quadrantis è previsto solo per la rappresentazione della semantica associata ai parametri di qualità scelti. Nonostante non è previsto l'impiego di strumenti di inferenza automatica che sfruttino il potere espressivo dei formalismi ontologici, l'utilizzo di formalismi propri del Semantic Web predispone i risultati della ricerca corrente a sviluppi futuri nella direzione proposta per l'evoluzione del Web. Nel proseguimento del progetto le ontologie saranno sfruttate anche per la rappresentazione della semantica dell'oggetto erogato dal servizio che, nel caso particolare, è lo streaming video.

4 Il framework

L'architettura proposta è formata da una serie di nodi che cooperano fra di loro per la realizzazione di un business process. Per supportare strumenti di diagnosi e recovery è necessario che i nodi abbiano determinate caratteristiche: quelle minime richieste perché siano considerati nodi attivi nelle azioni di diagnosi e recovery è che supportino il monitoring ed esportino un'interfaccia di management. In realtà non è richiesto che tutti i nodi abbiano tutte le caratteristiche: pur partecipando a tutti gli effetti alla realizzazione della parte di business, alcuni nodi possono essere coinvolti marginalmente o del tutto non coinvolti nella parte di diagnosi e recovery. Il modello generale di riferimento proposto considera uno scenario di business in cui alcuni servizi sono provider di video streaming. Per la gestione il modello prevede due livelli di comunicazione distinti: uno per l'interazione fra Web-Service ed uno per l'erogazione dei contenuti di streaming. Tale scelta è dettata dalla volontà di integrare in un unico framework le potenzialità offerte dall'approccio Service Oriented con l'erogazione di servizi Real Time. L'eterogeneità dei protocolli utilizzati e l'inadeguatezza del protocollo SOAP che sta alla base delle architetture SOA per l'erogazione di servizi real time, ha portato a separare i due canali. Tale separazione è netta nei protocolli di comunicazione ma restano interessanti interazioni fra i due livelli soprattutto per la gestione della QoS e quindi per la reazione ad eventuali fault.

La Figura 2.**a** mostra un esempio di framework dove viene evidenziato che esistono due distinti livelli e che i nodi hanno caratteristiche diverse. Si ipotizza che possano esistere nodi passivi, che cioè non mostrano nulla della propria struttura interna e/o forniscono informazioni sulla QoS. Tutti gli altri nodi devono come minimo avere una struttura che supporti il monitoring e un'interfaccia di management che consenta di chiedere/attivare specifiche azioni di recovery su quel nodo. Particolarmente interessanti sono i cosidetti *Complete Streaming Node* che sono nodi coinvolti sia al livello Web-Service che al livello Streaming. Hanno infatti l'interfaccia di business e di management ad entrambi i livelli ed in un certo senso costituiscono, da un punto di vista logico, il gateway fra i due livelli. In Figura 2.**b** viene mostrata una possibile istanza dell'architettura. Su input dell'utente, a livello Web-Service viene costruita e gestita l'orchestrazione e/o la coreografia dei servizi all'interno della quale trovano posto anche servizi di streaming. A livello streaming è gestita l'erogazione dei contenuti di streaming.

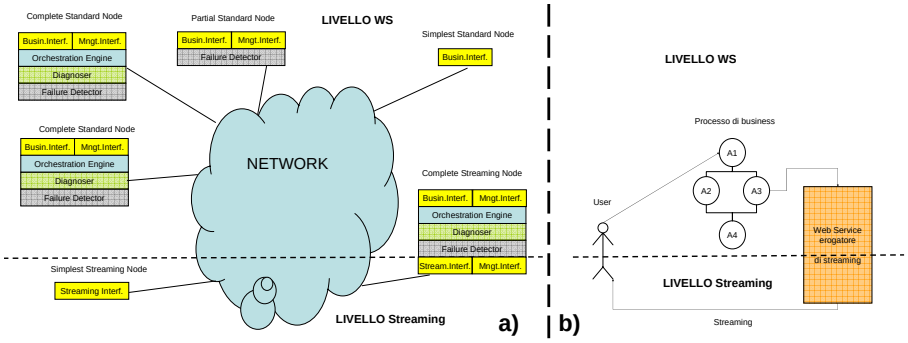

Figura 2. Architettura generale ed esempio di istanza

Si crea un canale diretto fra l'erogatore e l'utente finale. Eventuali fault o modifiche significative della QoS che vengono registrate a questo livello possono essere gestite in prima istanza sempre a tale livello, ma possono anche influenzare e quindi essere gestite al livello Web-Service. Il server gestore dello streaming funge quindi da gateway fra i due livelli. Supporta l'infrastruttura per il monitoring della QoS ad entrambi i livelli ed è legata alla sua struttura interna la propagazione di sintomi di fault da un livello ad un altro.

4.1 Livello Web-Service

A questo livello viene gestito il business process all'interno del quale vengono erogati contenuti di streaming. Il livello Web-Service è costituito da più nodi che collaborano all'interno di una architettura SOA. Nel caso più semplice esiste un solo nodo completo e quindi il processo è totalmente orchestrato. In casi più interessanti esistono più nodi completi e quindi all'interno del framework si realizza una coreografia.

Il monitoring. Se il servizio composto è descritto mediante una coreografia, i Web-Service che cooperano hanno solo viste parziali sul servizio complessivo e devono coordinarsi tra loro attraverso una conversazione complessa che coinvolge più partecipanti. In questa prospettiva, è possibile dotare il servizio coreografato di una funzionalità di monitoraggio del progredire della sua esecuzione, che permetta l'individuazione precoce degli errori e la notifica dei problemi ai Web-Service coinvolti, attraverso il tracciamento del comportamento conversazionale dei Web-Service che cooperano [3].

Durante l'esecuzione del servizio coreografato è presente un Monitor che viene informato relativamente ai messaggi spediti o ricevuti dai Web-Service coinvolti e al loro stato di esecuzione, attraverso l'interfaccia di management. Il Monitor utilizza queste informazioni per monitorare i parametri della QoS del livello Web-Service (per esempio, se il Monitor, interagendo con l'interfaccia di management del servizio che - secondo la coreografia - dovrebbe inviare un messaggio, si accorge che si trova in uno stato tale da non poter effettuare l'invio, può rilevare

- prima del timeout del ricevente - un ritardo che può causare una violazione di QoS) e per controllare se il servizio progredisce correttamente, cioè se il flusso dei messaggi tra i Web-Service rispetta la coreografia (accorgendosi, per esempio, se i messaggi non vengono inviati nell'ordine previsto). Se si verifica un errore, il Monitor valuta se il servizio coreografato è in grado di proseguire e informa i Web-Service partecipanti, al fine di metterli in grado di reagire al problema che si è verificato. Per mantenere il framework il più generale possibile, assumiamo che il Monitor non abbia alcuna informazione relativa all'implementazione interna dei Web-Service. I log dei messaggi scambiati durante l'esecuzione del servizio composto vengono anche utilizzati per popolare l'ontologia che rappresenta la QoS. I valori monitorati costituiscono inoltre l'input del predittore che sarà in grado di prevedere il verificarsi di fault. Poichè attualmente le informazioni individuate inerenti i parametri di qualità sono numeriche saranno impiegate tecniche classiche di machine learning supervisionato che, sulla base delle osservazioni provenienti dai file di log, forniranno indicazioni su possibili fault di qualità.

La comunicazione fra i nodi. L'interfaccia che i nodi espongono a questo livello è costituita dal WSDL per la parte di business e da WSDM per la parte di management. Attraverso tale interfaccia è possibile ottenere delle informazioni sullo stato del processo e specificamente delle sue variabili pubbliche, ma è anche possibile chiedere lo stop momentaneo dell'esecuzione di una specifica istanza o ancora specifiche azioni di recovery come la sostituzione di un servizio invocato. Ciascun nodo dichiara quindi quali azioni di recovery supporta che possono quindi essergli richieste da altri nodi. Se un nodo non espone un'interfaccia di management o le operazioni che offre sono inadeguate a risolvere il problema verificatosi nel sistema, esso può essere totalmente sostituito da un altro nodo che abbia caratteristiche compatibili.

La diagnosi. In sistemi complessi e composti da diversi attori, per individuare la causa di un fault è necessaria una attività di diagnosi. Nel progetto Quadrantis si è deciso di utilizzare l'approccio *model-based* (MBD, *Model-Based Diagnosis*) [8]. La maggior parte degli approcci *model-based* sono basati su un modello orientato ai componenti del sistema da diagnosticare. Il modello del sistema complessivo risultante è in grado di predire, o quanto meno di vincolare, l'effetto del comportamento corretto o scorretto di un componente, anche su variabili che non sono direttamente in relazione con il componente in esame. Il ragionamento diagnostico identifica le diagnosi come assegnamenti di modalità di comportamento ai componenti, che spieghino un certo insieme di osservazioni (valori per le variabili osservabili). Un motore diagnostico, in generale, esplora lo spazio delle diagnosi candidate ed effettua una discriminazione tra i candidati, eventualmente suggerendo l'acquisizione di ulteriori informazioni utili allo scopo. La discriminazione può essere effettuata esclusivamente in vista di un obiettivo diagnostico, per esempio la selezione di un'azione di riparazione appropriata.

Nella diagnosi *consistency-based* [15] utilizzata nel nostro approccio, una diagnosi è un assegnamento di modalità di comportamento ai componenti tale da essere consistente con le osservazioni. Per i modelli statici, questo significa che la

diagnosi candidata predice, per le variabili osservabili, un insieme di possibili valori che includono quello osservato. L'approccio alla diagnosi che viene proposto è *decentralizzato*. In particolare: i) ad ogni servizio è associato un *diagnosticatore locale* che deve fornire al diagnosticatore globale (vedi punto seguente) le informazioni necessarie per identificare le cause del malfunzionamento (del servizio globale); ii) esiste un *diagnosticatore globale* in grado di invocare i diagnosticatori locali e di mettere in relazione le informazioni ricevute da questi.

Si è scelto di adottare questo approccio in quanto esso consente di raggruppare ricorsivamente i Web-Service in aggregati di sotto-servizi, nascondendo i dettagli dell'aggregazione ai servizi dei livelli più alti. Il diagnosticatore globale deve conoscere esclusivamente le interfacce dei diagnosticatori locali e quindi condividere un protocollo di comunicazione con essi. Questo significa che il modello di un servizio resta privato al suo diagnosticatore locale e non deve essere reso visibile agli altri diagnosticatori locali o al diagnosticatore globale stesso.

Le politiche adattive. Sono al momento in corso di definizione le politiche di recovery che vengono messe in atto a fronte di un fault di QoS. Bisogna ricordare che il monitoring delle QoS del livello Web-Service e del livello streaming è totalmente disaccoppiato mentre le azioni di recovery possono influenzarsi.

Se la QoS a livello Web-Service si degrada in relazione ad operazioni non legate allo streaming, verranno attuate delle politiche di recovery per provare a risolvere il problema. Tali politiche possono o meno coinvolgere il task del processo che prevede l'erogazione dello streaming. Nel secondo caso l'erogazione continua perchè vuol dire che nulla osta a che prosegua lo streaming; nel primo caso invece l'operazione di erogazione dello streaming viene considerata a tutti gli effetti una operazione di un servizio composto e può quindi subire operazioni di recovery come ad esempio quelle definite in [7,13]. I Complete Streaming nodes offrono a livello Web-Service anche azioni specifiche legate allo streaming.

4.2 Livello Streaming

Il server gestore dello streaming funge da gateway fra i due livelli. A livello di streaming viene creato un canale apposito che utilizza i protocolli propri del video streaming. La comunicazione fra il lato client e quello server avviene attraverso una struttura ad agenti il cui compito è anche quello di implementare azioni di adattamento dinamico della QoS basate sulle preferenze dell'utente.

Il monitoring. Il problema di monitorare la banda gioca un ruolo chiave nei processi di gestione e regolazione della Qualità del Servizio di un'applicazione di streaming. Una metodologia per la stima della banda in un ambiente multimediale si basa sul concetto di *probing packet*. In particolare, si consideri una rete di calcolatori e si supponga di avere un sender S e un receiver R; per misurare la banda disponibile per trasmettere dati da S a R supponiamo che il sender invii dei pacchetti (detti *probing packet*) al receiver. In particolare, diciamo R_p la velocità con cui il sender emette dei pacchetti e con B la banda da stimare. Supponiamo di indicare con ΔD il ritardo associato alla ricezione di due pacchetti consecutivi. Si può dimostrare [9] che se il probing rate è inferiore alla banda da

stimare, allora il ritardo ΔD è nullo. Questo risultato permette di progettare un algoritmo iterativo per la stima della banda B [1,9].

La comunicazione fra i nodi. Le interfacce dei nodi interessati allo streaming, sia dal lato server che da quello client sono gestite tramite un'architettura ad agenti ed il linguaggio utilizzato per le comunicazioni è basato sullo standard ACML. I Complete Streaming Nodes forniscono inoltre all'interno dell'interfaccia di management del Web-Service delle primitive specifiche per l'interazione con lo streaming. Questa caratteristica garantisce agli attori del livello Web-Service un ulteriore strumento per le loro politiche adattative e di recovery. Ad esempio semplicemente interrompere momentaneamente lo streaming perchè è necessaria un'interazione con l'utente e poi riprenderla subito dopo è una situazione realizzabile solo con un'interfaccia come quella appena proposta. .

Le politiche adattive. Se la QoS a livello di streaming si degrada, gli agenti intelligenti preposti alle politiche adattive proveranno a modificare i parametri per provare a fare rientrare i valori in un range accettabile. Se tali politiche falliscono il fault verrà inoltrato dal Server di streaming al livello Web-Service. Il fault diventa quindi il fault di una operazione definita nell'interfaccia WSDL del server di streaming. Verranno applicate le consuete tecniche per le recovery a livello Web-Service come ad esempio quelle definite in [13].

5 Risultati preliminari e ricerca in corso

Nel primo anno di attività del progetto si è definita l'Ontologia per i parametri di qualità di un Web-Service ed è in corso di definizione quella per i parametri di qualità di un video streaming. Si è definito il framework generale presentato nella Sezione 4 e si sta studiando il problema di learning che sarà alla base del funzionamento del classificatore utilizzato per la predizioni di possibili fault di QoS. E' infine allo studio una definizione precisa dell'interazione fra le QoS dei due livelli. I punti dove avviene il contatto fra le due QoS sono l'utente e il Server di streaming. La QoS dello streaming viene rappresentata nel sistema come QoS dell'oggetto del Web-Service. E' possibile che dal livello Web-Service vengano richieste misurazioni della QoS dell'altro livello, ma in generale ciascun livello in fase di analisi considera la QoS dell'altro come un valore binario. E' possibile invece ipotizzare un'interazione più sofisticata durante l'attuazione di politiche adattive. E' importante notare che il rapporto fra violazioni di QoS non è simmetrico: è infatti presumibile che anche se il video streaming fallisce l'intero processo di business possa ripararsi, al limite sostituendo il Web-Service erogatore; non è invece plausibile che, a fronte di un fallimento dell'intero business process a causa di un errore critico, il Server di streaming continui ad erogare i suoi contenuti.

Ringraziamenti

Questo articolo è parzialmente supportato dal Ministero per l'Università e la Ricerca Scientifica nell'ambito del progetto PRIN Quadrantis.

Riferimenti bibliografici

1. M. Allman. Measuring end-to-end bulk transfer capacity. In *Proc. of ACM SIGCOMM Workshop on Internet Measurement*, pages 139–143. ACM Press, 2001.
2. D. Ardagna and B. Pernici. Dynamic web service composition with QoS constraints. *Int. J. Business Process Integration and Management*, 1(4), 2006.
3. L. Ardissono, R. Furnari, A. Goy, G. Petrone, and M. Segnan. Monitoring choreographed services. In *Proc. of Int. Joint Conferences on Computer, Information, System Sciences, and Engineering CISSE 2006*, 2006.
4. T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American.*, 284(5), 2001.
5. C. Cappiello, B. Pernici, and P. Plebani. Quality-agnostic or quality-aware semantic service descriptions? In *W3C Workhsop on Semantic Web Service Framework*, Innsbruck (Austria), 2005.
6. M.G. Fugini, P. Plebani, and F. Ramoni. A user driven policy selection model. In *Proc. of Int. Conference on Service Oriented Computing ICSOC*, Chicago, IL, USA, 2006.
7. R. Hamadi and B. Benatallah. Recovery nets: Towards self-adaptive workflow systems. In *Proc. of Int. Conf. on Web Information Systems Engineering (WISE)*, pages 439–453, Brisbane, Australia, 2004. Springer.
8. W. Hamscher, L. Console, and J. de Kleer, editors. *Readings in Model-Based Diagnosis*. Morgan Kaufmann, 1992.
9. M. Jain and C. Dovrolis. End-to-end available bandwidth: measurement methodology, dynamics, and relation with tcp throughput. *IEEE/ACM Transactions on Networking*, 11(4):537–549, 2003.
10. A. Lazovik, M. Aiello, and M.P. Papazoglou. Planning and monitoring the execution of web service requests. In *Proc. of Int. Conf. on Service-Oriented Computing (ICSOC*, pages 335–350, Trento, Italy, 2003.
11. E. M. Maximilien and M. P. Singh. A Framework and Ontology for Dynamic Web Services Selection. *IEEE Internet Computing*, 08(5):84–93, 2004.
12. D.L. McGuinness and F. van Harmelen. OWL Web Ontology Language - Overview. W3C Recommendation, February 2004. http://www.w3.org/TR/owl-features/.
13. S. Modafferi and E. Conforti. Methods for enabling recovery actions in WS-BPEL. In *Proc. of Int. Conf. on Cooperative Information Systems (COOPIS)*, Montpellier, France, 2006.
14. I.V. Papaioannou, D.T. Tsesmetzis, I.G. Roussaki, and M.E. Anagnostou. A QoS ontology language for web-services. In *Proc. of IEEE Int. Conf. on Advanced Information Networking and Applications AINA*, pages 101–106, Los Alamitos, CA, USA, 2006. IEEE Computer Society.
15. R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–95, 1987.
16. J. Ruckert and B. Paech. Web service quality descriptions for web service consumers. In *Proc. of Int. Conference on Quality Engineering in Software Technology CONQUEST*, 2006.
17. L. Zeng, B. Benatallah, A.H.H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang. Qos-aware middleware for web services composition. *IEEE Trans. Software Eng.*, 30(5):311–327, 2004.