Caractérisation des transitions temporisées dans les logs de conversation de services Web

Didier Devaurs, Fabien de Marchi, Mohand-Saïd Hacid

Caractérisation des transitions temporisées dans les logs de conversation de services Web

Didier Devaurs, Fabien De Marchi, Mohand-Saïd Hacid

LIRIS, UMR 5205, CNRS / Université Claude Bernard Lyon 1
Bâtiment Nautibus, 8 boulevard Niels Bohr, F-69622 Villeurbanne, France
didier.devaurs@irisa.fr, {fabien.demarchi, mohand-said.hacid}@liris.cnrs.fr

Résumé. La connaissance du protocole de conversation d'un service Web est importante pour les utilisateurs et les fournisseurs, car il en modélise le comportement externe ; mais, il n'est souvent pas spécifié lors de la conception. Notre travail s'inscrit dans une thématique d'extraction du protocole de conversation d'un service existant à partir de ses données d'exécution. Nous en étudions un sous-problème important qui est la découverte des transitions temporisées (i.e. les changements d'état liés à des contraintes temporelles). Nous proposons un cadre formel aboutissant à la définition des *expirations propres*, qui représentent un équivalent dans les logs des transitions temporisées. A notre connaissance, ceci représente la première contribution à la résolution de ce problème.

1 Introduction

Les services Web constituent la nouvelle génération des technologies du Web pour l'intégration d'applications. Ce sont des composants logiciels mis à disposition par des fournisseurs, invocables sur Internet par des clients (des utilisateurs ou d'autres services), et communiquant de façon asynchrone, par le biais de messages. Ils permettent de réaliser une intégration à faible couplage et à moindre coût, du fait qu'ils utilisent des standards généralistes fortement répandus (XML, HTTP). Toutefois, cette souplesse d'intégration n'est possible que si les utilisateurs d'un service savent comment interagir avec celui-ci. A un service doivent donc être associées des descriptions assez riches pour permettre de comprendre sa sémantique d'exécution.

Le langage WSDL, par exemple, spécifie l'interface d'un service : les opérations, les types de messages, le format des entrées-sorties. Cependant, Benatallah et al. (2004) ont montré que ceci était insuffisant dans l'optique d'une utilisation automatique des services Web, et ont défini le *protocole de conversation*, qui permet de spécifier quelles sont les séquences ordonnées de messages (appelées conversations) qu'un service peut émettre ou recevoir. Benatallah et al. (2005a,b) ont ensuite ajouté des contraintes temporelles à leur modèle, rebaptisé *protocole de conversation temporisé*. Son utilisation offre de nombreuses applications, pour la vérification automatique de bon fonctionnement, de compatibilité, etc. Néanmoins, en pratique, de nombreux services ne possèdent pas une telle spécification. Il est donc légitime de chercher à obtenir le protocole de conversation d'un service s'il n'a pas été défini lors de la conception.

Fournir le protocole d'un service à ses partenaires et clients est bien sûr l'application la plus directe de ce problème de découverte ; mais il possède un intérêt bien plus grand pour l'ingé-

nierie des services Web. Par exemple, un concepteur pourrait connaître le protocole « effectif » (réellement utilisé) d'un service, et savoir s'il correspond bien aux contraintes de conception ; il serait possible de faire évoluer un service plus facilement : l'utilisation d'un modèle visuel de son comportement (plutôt que la simple analyse de code) permettrait de faciliter l'ajout de nouvelles fonctionnalités, de nouvelles contraintes ou règles, etc.

L'extraction du protocole de conversation d'un service englobe de nombreux défis techniques. Le premier réside dans la modélisation du protocole découvert : il est important de prendre en compte l'incertitude du résultat, et de proposer des indices de confiance et des critères de qualité, permettant d'évaluer sa pertinence. Cette incertitude provient principalement du fait que les logs d'un service peuvent contenir des erreurs d'enregistrement (du « bruit »). Des outils sont donc nécessaires, pour analyser et nettoyer les données avant de les traiter. Il est également important de proposer à l'utilisateur des outils lui permettant de modifier et corriger le protocole découvert. Un autre point délicat est la corrélation des messages, i.e. l'identification et la séparation, dans les logs, des différentes conversations (qui peuvent se chevaucher).

Ce problème de découverte constitue en fait un cas particulier d'une problématique beaucoup plus large : la découverte d'un modèle à partir d'instances de celui-ci. De nombreux travaux sont liés à cette thématique ; on peut citer par exemple l'inférence grammaticale (Parekh et Honavar, 2000), la fouille de workflow (Cook et Wolf, 1998; van der Aalst et al., 2004), ou la fouille d'interactions de services Web (Dustdar et al., 2004). En inférence grammaticale, le but consiste à trouver, à partir d'un ensemble de mots appartenant ou pas à un langage donné, une grammaire permettant de générer ce langage. Pour la fouille de workflow (ou découverte de processus), le problème réside dans la construction, à partir des logs d'exécution d'un processus, d'un modèle formel permettant de représenter le fonctionnement de ce processus. En fouille des interactions de services Web, l'objectif est de découvrir, à partir des logs d'un ensemble de services, un workflow modélisant les interactions possibles entre ces services.

Dans le contexte applicatif des services Web, Motahari et al. (2006) ont proposé une méthode d'extraction de protocoles à partir des archives de conversation entre services (fichiers « logs »). Ce travail porte sur plusieurs des enjeux mentionnés plus haut. Cependant, il ne considère pas les aspects temporels du protocole de conversation.

Contribution. Notre travail se place également dans le contexte de cette problématique. Nous traitons un sous-problème important qui est la découverte des transitions temporisées du protocole de conversation, i.e. des changements d'état liés non pas à l'émission d'un message mais à l'existence d'une contrainte de temps. Bien qu'une telle transition ne figure pas de façon explicite dans les logs du service, il est possible d'identifier les conséquences de son existence. Ce sont donc ces « traces » que nous formalisons et que nous extrayons des données. Pour cela, nous introduisons la notion d'*expiration propre*, qui représente dans les logs ce que la notion de transition temporisée représente dans le protocole de conversation.

Organisation de l'article. Tout d'abord, nous explicitons ce qu'est le protocole de conversation temporisé d'un service Web, et ce que contiennent des logs de conversation. Puis, nous définissons précisément le problème, et présentons nos hypothèses de travail. Nous décrivons ensuite les différents concepts que nous avons définis pour le résoudre. Enfin, nous donnons les conclusions et perspectives de notre travail.

2 Préliminaires

2.1 Protocole de conversation temporisé

Le *protocole de conversation* a été proposé par Benatallah et al. (2004) afin de modéliser la sémantique d'exécution d'un service Web. Il est représenté par un automate à états fini déterministe, où les états constituent les différentes phases dans lesquelles le service peut se trouver lors de son exécution. Les transitions sont déclenchées lorsque le service émet ou reçoit des messages ; chacune est étiquetée, soit par un message entrant (signe $+$), soit par un message sortant (signe $-$). Un *message* correspond à l'invocation d'une opération du service, ou à l'envoi de son résultat. Une *conversation* est une séquence de messages échangés entre un service et un client. Le protocole de conversation permet de spécifier l'ensemble des conversations supportées par un service (i.e. l'ensemble des séquences d'échanges de messages valides).

Certains changements d'état du service ne sont pas liés à l'envoi de messages explicites, mais à des contraintes temporelles (durée de validité, échéance, etc.). Le modèle de base a donc été enrichi de transitions temporisées, et rebaptisé de ce fait *protocole de conversation temporisé*. Une *transition temporisée* se produit de façon automatique, après qu'un certain laps de temps se soit écoulé à partir du moment où elle a été permise (i.e. où l'état source de la transition est devenu l'état courant), ou bien après qu'une certaine date soit atteinte ; elle est étiquetée par la contrainte de temps associée. Notons que, puisque le modèle est déterministe, un état ne peut admettre plusieurs transitions temporisées comme transitions sortantes.

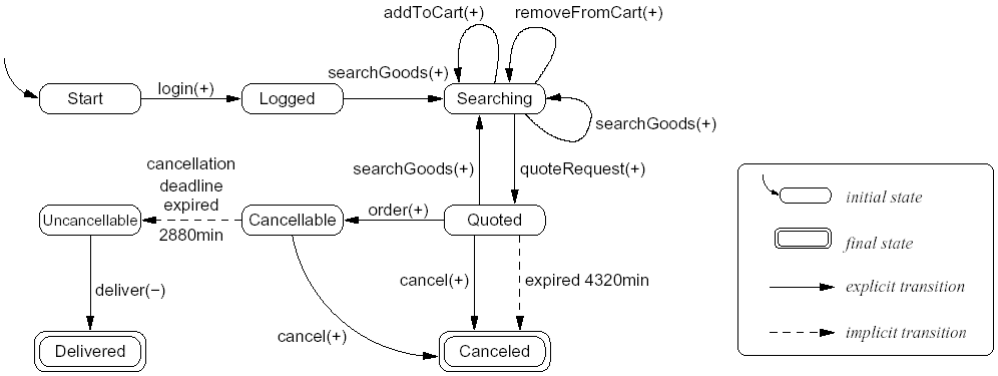


FIG. 1 – *Exemple de protocole de conversation temporisé (Benatallah et al., 2005b).*

Exemple 1 La figure 1 représente un protocole de conversation temporisé décrivant le comportement externe d'un service de commande de marchandises. Les transitions explicites sont représentées en trait plein, et les transitions temporisées en pointillés. Ce protocole spécifie que le client du service doit d'abord se connecter (opération *login*), puis chercher des produits (*searchGoods*). Il peut alors ajouter ou enlever des produits de son caddie (*addToCart*, *removeFromCart*), chercher d'autres marchandises (*searchGoods*), ou demander un devis (*quoteRequest*) qui sera valide seulement pendant 3 jours (i.e. 4320 minutes). Il peut alors commander les marchandises (*order*). S'il ne le fait pas, au bout des 3 jours la conversation se termine (par le biais de la transition temporisée sortant de l'état *Quoted*), et la commande est annulée.

2.2 Logs de conversation

Les différentes façons de collecter les logs d'interaction d'un service ont été décrites par Dustdar et al. (2004). En fonction de la façon dont les services sont implémentés et du type d'outils utilisés pour gérer leur exécution, différents types d'informations peuvent être présents dans les logs. Dans un scénario réaliste, les informations collectées sont en général, en plus du contenu du message, l'émetteur, le receveur et la date.

Ces informations peuvent ne pas suffire pour identifier une conversation de façon unique, dans le cas où l'on ne dispose pas d'un identifiant de chaque conversation enregistrée dans les logs. Il est mis en avant par Motahari et al. (2006) que le fait de fournir automatiquement un tel identifiant (s'il n'est pas présent par défaut) est un véritable problème en soi. Aussi, ont-ils supposé, pour mener à bien leur tâche de découverte du protocole de conversation, que cette information était présente dans les logs. Nous en ferons de même.

Les logs que nous traitons sont les enregistrements des messages émis ou envoyés par un service. Ces enregistrements sont effectués par le serveur hébergeant le service. Nous ne considérons pas les logs « internes » du service, qui peuvent être ajoutés au code par le concepteur. Les informations dont nous disposons sont les intitulés des messages, et leurs dates d'émission ; la connaissance de l'émetteur ou du receveur ne nous est d'aucune utilité. Précisons également que nous ne tenons pas compte de la polarité des messages.

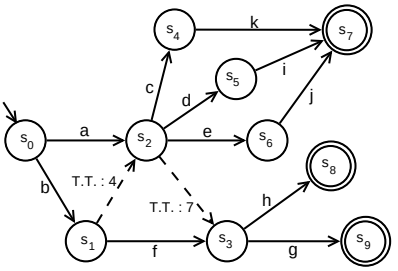
Exemple 2 Pour un service vérifiant le protocole de conversation représenté par la figure 1, on peut par exemple obtenir les conversations suivantes :
(login, 9:18) (searchGoods, 9:20) (addToCart, 9:21) (quoteRequest, 9:22) (cancel, 9:51) ;
(login, 11:03) (searchGoods, 11:04) (addToCart, 11:08) (quoteRequest, 11:12).

3 Spécification du problème

L'ensemble des intitulés des messages appartenant au protocole de conversation temporisé sera noté Msg. On notera L les logs de conversation dont on dispose. Formellement, L sera un multi-ensemble de conversations. Une conversation sera notée : $C = \langle M_1, M_2, ..., M_{n_C} \rangle$, où $\forall\ 1 \leq i \leq n_C$ (avec $n_C \in \mathbb{N}^*$), $M_i = (m_i, t_i)$, avec $m_i \in Msg$ (intitulé du message) et $t_i \in \mathbb{R}_+$ (instant d'enregistrement du message), et $t_1 < t_2 < ... < t_{n_C}$; elle représentera une séquence d'occurrences de messages. On notera de plus : $\forall\ 1 \leq i \leq n_C,\ m_i = M_i.nom$ et $t_i = M_i.date$.

3.1 Tâche de découverte

Considérons le cas où l'on ne connaît pas le protocole de conversation qui a permis de générer les logs L. Le problème consiste à exhiber le fait que ces données traduisent la présence de transitions temporisées dans le protocole. Notre méthode vise à examiner les couples de messages dans les logs, afin de déterminer si, entre deux transitions explicites, a été déclenchée une transition implicite ou pas. Pour ce faire, nous allons calculer, pour chaque conversation, le laps de temps écoulé entre l'émission de deux messages consécutifs.



$C_1 = \langle (a,0),(c,1),(k,4) \rangle$
$C_2 = \langle (a,0),(c,3),(k,5) \rangle$
$C_3 = \langle (a,0),(d,2),(i,6) \rangle$
$C_4 = \langle (a,0),(d,5),(i,8) \rangle$
$C_5 = \langle (a,0),(e,4),(j,5) \rangle$
$C_6 = \langle (a,0),(e,6),(j,8) \rangle$
$C_7 = \langle (a,0),(h,8) \rangle$
$C_8 = \langle (a,0),(h,10) \rangle$
$C_9 = \langle (a,0),(g,15) \rangle$
$C_{10} = \langle (a,0),(g,16) \rangle$
$C_{11} = \langle (b,0),(f,1),(h,3) \rangle$
$C_{12} = \langle (b,0),(f,2),(g,4) \rangle$
$C_{13} = \langle (b,0),(f,3),(h,4) \rangle$
$C_{14} = \langle (b,0),(f,1),(g,5) \rangle$
$C_{15} = \langle (b,0),(c,6),(k,8) \rangle$
$C_{16} = \langle (b,0),(c,8),(k,9) \rangle$
$C_{17} = \langle (b,0),(d,7),(i,8) \rangle$
$C_{18} = \langle (b,0),(d,10),(i,11) \rangle$
$C_{19} = \langle (b,0),(e,8),(j,10) \rangle$
$C_{20} = \langle (b,0),(e,9),(j,10) \rangle$
$C_{21} = \langle (b,0),(h,13) \rangle$
$C_{22} = \langle (b,0),(h,15) \rangle$
$C_{23} = \langle (b,0),(g,14) \rangle$
$C_{24} = \langle (b,0),(g,15) \rangle$

FIG. 2 – *Protocole P_1 et logs associés L_1.*

Exemple 3 Les logs L_1, associés au protocole P_1 représenté dans la figure 2, constituerons l'exemple que nous développerons dans la suite de cet article. Ici, nous devons par exemple mettre en évidence le fait, qu'après l'émission du message a, les messages c, d et e peuvent être émis uniquement avant un certain laps de temps, et que les messages g et h peuvent être émis uniquement après ce laps de temps. Il est important de noter que les dates d'enregistrement des messages sont les dates relatives au début de chaque conversation.

3.2 Hypothèses de travail

Nous supposerons dans la suite que les protocoles associés aux logs que nous traiterons, comme dans l'exemple précédent, ne possèdent pas de transition temporisée menant dans un état final, bien que cela puisse se produire dans les cas réels. Le fait de se ramener à ce cas particulier - qui est déjà relativement complexe en soi - nous aidera par la suite à mieux appréhender le cas général. Nous supposerons également que les transitions du protocole de conversation sont étiquetées de façon unique, même si certaines correspondent à un même message. Si nous ne sommes pas dans un tel cas de figure, nous pourrons envisager de nous y ramener en effectuant un pré-traitement sur les données.

En ce qui concerne les logs, nous allons supposer qu'ils ne sont pas bruités, c'est-à-dire que les messages sont correctement enregistrés, et dans une séquence correcte. Ceci nous permettra, dans un premier temps, de proposer une méthode «complète». Une approche probabiliste pourra ensuite être envisagée pour pallier le fait que les logs puissent être bruités dans les cas réels. Nous supposerons également que les logs sont suffisamment «complets» pour retrouver les transitions temporisées, c'est-à-dire que tous les «chemins» du protocole de conversation ont été parcourus.

4 Formalisation du problème

4.1 Episode

Afin de formaliser le fait que deux messages sont consécutifs dans le temps, nous introduisons la notion d'épisode, qui est directement inspirée de celle définie par Mannila et al. (1997), ainsi que la notion d'occurrence d'un épisode.

Définition 1 (Episode) Un épisode constitue une séquence de deux intitulés de messages : $\alpha = \langle m, m' \rangle$, avec $m, m' \in Msg$.

Définition 2 (Occurrence d'un épisode) Une occurrence de l'épisode $\alpha = \langle m, m' \rangle$ dans L est une séquence $\langle M, M' \rangle$ telle que : $\exists\, C \in L$ vérifiant

$$\begin{cases} M, M' \in C \\ M.nom = m \text{ et } M'.nom = m' \\ M.date < M'.date \\ \forall\, M'' \in C,\ M''.date < M.date \text{ ou } M''.date > M'.date \end{cases}$$

Si une telle séquence existe, on dira que l'épisode α se produit dans la conversation C.

On notera $Occ(\alpha)$ l'ensemble des occurrences de l'épisode α dans L. On dira que l'épisode α se produit dans les logs L si α se produit dans au moins une conversation C de L, i.e. si $Occ(\alpha) \neq \phi$. On notera Ep l'ensemble des épisodes se produisant dans les logs L.

Proposition 1 *Soit, $\forall\, m \in Msg$, l'ensemble $P_m = \{\alpha \in Ep \mid \exists\, m' \in Msg,\ \alpha = \langle m, m' \rangle\}$. Alors, $\{P_m \mid m \in Msg\}$ est une partition de Ep.*

Cette proposition exprime le fait que l'on peut construire une partition de l'ensemble des épisodes, où chaque partie est constituée de l'ensemble des épisodes dont le premier élément est un message m donné. Nous ne donnons pas de preuve pour ce résultat, qui est relativement trivial. Cette propriété va nous permettre de décomposer notre tâche de découverte. Au lieu d'examiner les épisodes dans leur ensemble, nous allons traiter chaque élément de cette partition séparément. Pour ce faire, nous allons définir la notion de durée d'occurrence.

Intuitivement, la durée d'une occurrence d'un épisode est la différence des dates des messages de l'épisode dans l'occurrence. A partir de ceci, on définit la durée d'occurrence minimale (respect. maximale) d'un épisode comme étant la plus petite (respect. la plus grande) durée de toutes les occurrences de cet épisode. L'intervalle de durée d'occurrence d'un épisode est l'intervalle qui englobe l'ensemble des durées d'occurrence de cet épisode.

Définition 3 (Durée d'occurrence d'un épisode)
- La durée de l'occurrence $\langle M, M' \rangle$ de l'épisode α est le réel : $M'.date - M.date$.
- La durée d'occurrence minimale de α dans L est donnée par la fonction $d_{min} : Ep \longrightarrow \mathbb{R}_+$ telle que : $\forall\, \alpha \in Ep$, $d_{min}(\alpha) = min\{M'.date - M.date \mid \langle M, M' \rangle \in Occ(\alpha)\}$.
- La durée d'occurrence maximale de α dans L est donnée par la fonction $d_{max} : Ep \longrightarrow \mathbb{R}_+$ telle que : $\forall\, \alpha \in Ep$, $d_{max}(\alpha) = max\{M'.date - M.date \mid \langle M, M' \rangle \in Occ(\alpha)\}$.
- On appelle intervalle de durée d'occurrence de α dans L, l'intervalle $[d_{min}(\alpha); d_{max}(\alpha)]$.

De la même façon, on définit la durée d'occurrence minimale (respect. maximale) d'un ensemble d'épisodes comme étant le minimum (respect. maximum) des durées d'occurrences minimales (respect. maximales) des épisodes appartenant à cet ensemble. L'intervalle de durée d'occurrence d'un ensemble d'épisodes est l'intervalle qui englobe l'ensemble des durées d'occurrence de tous les épisodes de cet ensemble.

Définition 4 (Durée d'occurrence d'un ensemble d'épisodes)
– La durée d'occurrence minimale d'un ensemble d'épisodes dans L est donnée par la fonction $D_{min} : \mathcal{P}(Ep) \setminus \{\phi\} \longrightarrow \mathbb{R}_+$ telle que : $\forall\, A \subseteq Ep$, $D_{min}(A) = min\{d_{min}(\alpha) \mid \alpha \in A\}$.
– La durée d'occurrence maximale d'un ensemble d'épisodes est donnée par l'application $D_{max} : \mathcal{P}(Ep) \setminus \{\phi\} \longrightarrow \mathbb{R}_+$ telle que : $\forall\, A \subseteq Ep$, $D_{max}(A) = max\{d_{max}(\alpha) \mid \alpha \in A\}$.
– On appelle intervalle de durée d'occurrence d'un ensemble d'épisodes A dans L, l'intervalle $[D_{min}(A); D_{max}(A)]$.

Exemple 4 Considérons les logs L_1 (cf. figure 2). Les intervalles de durée d'occurrence de $\{\langle a, c\rangle, \langle a, d\rangle\}$ et $\{\langle a, h\rangle\}$ sont $[1; 5]$ et $[8; 10]$. Ils sont disjoints et vérifient une relation de précédence sur l'échelle du temps. Bien évidemment, ceci est dû au fait qu'il existe une transition temporisée, entre les états s_2 et s_3, déclenchée automatiquement au temps 7 (après l'arrivée dans l'état s_2) si aucun des messages c, d ou e n'est émis avant.

Du fait qu'elle soit la conséquence de la présence d'une transition temporisée, la relation de précédence présentée dans cet exemple nous sera utile dans la suite. En effet, si l'on inverse le raisonnement, trouver qu'une telle relation est vérifiée par les données pourrait nous conduire à la découverte d'une transition temporisée. Nous formalisons donc cette relation.

4.2 Relation d'ordre sur les ensembles d'épisodes

Définition 5 (relation $\prec$) Soient $A, B \subset Ep$ $(A, B \neq \phi)$.
– On dira que A est **avant** B, ce que l'on notera $A \prec B$, si : $D_{max}(A) < D_{min}(B)$.
– On dira que A et B sont incomparables, ce que l'on notera $A \parallel B$, si : $A \not\prec B$ et $B \not\prec A$.

Intuitivement, l'expression $A \prec B$ traduit le fait que l'intervalle de durée d'occurrence de A est disjoint de celui de B et qu'il le précède sur l'échelle du temps. On dira également que B est après A. On peut montrer que la relation $\prec$ est une relation d'ordre strict sur $\mathcal{P}(Ep) \setminus \{\phi\}$, c'est-à-dire qu'elle est irréflexive, asymétrique et transitive.

Exemple 5 Considérons les logs L_1 (présentés dans la figure 2). On obtient :
$\{\langle a, c\rangle\} \prec \{\langle a, g\rangle\}$, car $D_{max}(\{\langle a, c\rangle\}) = 3 < 15 = D_{min}(\{\langle a, g\rangle\})$;
$\{\langle a, c\rangle, \langle a, d\rangle\} \prec \{\langle a, g\rangle\}$, car $D_{max}(\{\langle a, c\rangle, \langle a, d\rangle\}) = 5 < 15 = D_{min}(\{\langle a, g\rangle\})$;
$\{\langle a, c\rangle, \langle a, d\rangle, \langle a, e\rangle\} \prec \{\langle a, g\rangle, \langle a, h\rangle\}$; etc.

Proposition 2 *Soient $m \in Msg$, et $A, B \subset P_m$ $(A, B \neq \phi)$. S'il existe une transition temporisée, dans le protocole de conversation, entre l'état d'où sortent les transitions correspondant aux éléments de A et celui d'où sortent les transitions correspondant aux éléments de B, alors $A \prec B$.* [1]

Cette proposition constitue une condition nécessaire à l'existence d'une transition temporisée. Notre problème de découverte serait résolu si cette condition était également suffisante (nous disposerions d'un objet équivalent à une transition temporisée), mais ce n'est pas le cas.

[1] L'ensemble des démonstrations des résultats que nous allons énoncer sont présentées dans la version longue de cet article (Devaurs et al., 2006). https://liris.cnrs.fr/publis?id=2604

Remarque : La réciproque de la proposition 2 est fausse.

Contre-exemple On a $\{\langle a, c\rangle\} \prec \{\langle a, e\rangle\}$ (car $D_{max}(\{\langle a, c\rangle\}) = 3 < 4 = D_{min}(\{\langle a, e\rangle\})$), dans les logs L_1, alors que les transitions étiquetées par c et e sortent du même état (cf. fig. 2).

La proposition 2 et l'hypothèse de complétude des logs nous assurent que l'ensemble des expressions de la forme $A \prec B$ vérifiées par les logs englobe l'ensemble des transitions temporisées. Toutefois, ces expressions peuvent aussi nous donner, en plus, de fausses informations, sur des transitions inexistantes. Ceci vient du fait que la relation $\prec$ ne prend pas en compte l'ensemble des informations induites par la présence d'une transition temporisée. C'est pourquoi nous définissons dans la suite une relation plus riche sur les ensembles d'épisodes.

4.3 Expiration

Définition 6 (Expiration) Soient $m \in Msg$, et $A, B \subset P_m$ ($A, B \neq \phi$). On dira que les logs L satisfont l'expiration $E(m, A, B)$, ce que l'on notera $L \vDash E(m, A, B)$, si :

$$\begin{cases} A \prec B \\ \forall\, \alpha \in P_m,\ \{\alpha\} \nparallel A \text{ ou } \{\alpha\} \nparallel B \end{cases}$$

Si $A = \{\langle m, m'_1\rangle, \langle m, m'_2\rangle, \ldots, \langle m, m'_p\rangle\}$, si $B = \{\langle m, m''_1\rangle, \langle m, m''_2\rangle, \ldots, \langle m, m''_q\rangle\}$, et si $L \vDash E(m, A, B)$, par abus de notation, on écrira : $L \vDash E(m, \{m'_1, \ldots, m'_p\}, \{m''_1, \ldots, m''_q\})$.

Intuitivement, l'assertion $L \vDash E(m, A, B)$ traduit le fait que, d'après les logs L, les durées d'occurrence de tous les épisodes de A sont strictement inférieures aux durées d'occurrence de tous les épisodes de B, et qu'il n'existe pas d'épisode dont certaines occurrences auraient une durée appartenant à l'intervalle de durée d'occurrence de A et dont d'autres occurrences auraient une durée appartenant à l'intervalle de durée d'occurrence de B. A cette expiration peut être associé un délai, supérieur à la durée d'occurrence maximale de tous les épisodes de A, et inférieur à la durée d'occurrence minimale de tous les épisodes de B. Il existe donc une infinité de valeurs possibles pour un tel délai d'expiration : ce sont tous les réels de l'intervalle $]D_{max}(A); D_{min}(B)[$, que l'on appellera intervalle d'expiration de $E(m, A, B)$.

Exemple 6 Les logs L_1 satisfont les expirations, $E(a, \{c, d, e\}, \{g\})$, $E(a, \{c, d\}, \{g\})$, $E(b, \{f\}, \{c, d, e\})$, $E(b, \{c, d, e\}, \{g, h\})$, $E(b, \{f\}, \{g, h\})$, $E(b, \{f\}, \{c, d, e, g, h\})$, etc. Par contre, $L_1 \nvDash E(a, \{c\}, \{e\})$, car $\{\langle a, d\rangle\} \parallel \{\langle a, c\rangle\}$ et $\{\langle a, d\rangle\} \parallel \{\langle a, e\rangle\}$.

Proposition 3 *Soient $m \in Msg$, et $A, B \subset P_m$ ($A, B \neq \phi$). S'il existe une transition temporisée, dans le protocole de conversation, entre l'état d'où sortent les transitions correspondant aux éléments de A et celui d'où sortent les transitions correspondant aux éléments de B, alors $L \vDash E(m, A, B)$.*

Remarque : La réciproque de la proposition 3 est fausse.

Contre-exemple L'expiration $E(b, \{f\}, \{g, h\})$ est satisfaite par les logs L_1, bien qu'il n'y ait pas de transition temporisée entre les états s_1 et s_3 du protocole P_1 (cf. figure 2). Par contre, il existe une chaîne formée de deux transitions temporisées, reliant ces deux états.

La proposition 3 et l'hypothèse de complétude des logs nous assurent que chaque transition temporisée du protocole de conversation peut être retrouvée par l'intermédiaire d'une certaine expiration satisfaite par les logs. Cependant, une expiration est satisfaite entre deux ensembles d'épisodes, aussi bien en présence d'une transition temporisée entre les états correspondant à ces ensembles d'épisodes, que d'une chaîne de transitions temporisées. Nous allons donc définir une classe d'expirations plus restreinte, afin d'éviter cette ambiguïté. Un autre problème lié aux expirations est qu'elles sont beaucoup plus nombreuses que les transitions temporisées.

Exemple 7 Dans le cas du protocole P_1 (cf. figure 2), la transition temporisée présente entre les états s_1 et s_2 entraîne la satisfaction, par les logs L_1, de l'expiration $E(b, \{f\}, \{c, d, e\})$, mais aussi de $E(b, \{f\}, \{c, d, e, g, h\})$; celle qui relie les états s_2 et s_3 entraîne la satisfaction de l'expiration $E(a, \{c, d, e\}, \{g\})$, mais aussi de $E(a, \{c, d\}, \{g\})$.

Cet exemple illustre le fait que plusieurs formes de «redondance» apparaissent. Or, nous voulons apporter le minimum d'informations nécessaires à l'utilisateur pour retrouver les transitions temporisées. C'est donc dans ce sens que nous définissons les *expirations propres*.

4.4 Expiration propre

Définition 7 (Expiration propre) Soient $m \in Msg$, et $A, B \subset P_m$ ($A, B \neq \phi$). On dira que les logs L satisfont l'expiration propre $EP(m, A, B)$, ce que l'on notera $L \vDash EP(m, A, B)$, si :

$$\begin{cases} A \prec B \\ \forall\, \alpha \in P_m \setminus (A \cup B),\ \{\alpha\} \nVdash A \cup B \\ \forall\, X, Y \subset A\ (X, Y \neq \phi) \text{ tels que } X \cup Y = A, \text{ on a : } X \not\prec Y \\ \forall\, X, Y \subset B\ (X, Y \neq \phi) \text{ tels que } X \cup Y = B, \text{ on a : } X \not\prec Y \end{cases}$$

Si $A = \{\langle m, m'_1\rangle, \ldots, \langle m, m'_p\rangle\}$, si $B = \{\langle m, m''_1\rangle, \ldots, \langle m, m''_q\rangle\}$, et si $L \vDash EP(m, A, B)$, par abus de notation on écrira : $L \vDash EP(m, \{m'_1, \ldots, m'_p\}, \{m''_1, \ldots, m''_q\})$.

Intuitivement, l'assertion $L \vDash EP(m, A, B)$ traduit le fait que, d'après les logs L, les durées d'occurrence de tous les épisodes de A sont strictement inférieures aux durées d'occurrence de tous les épisodes de B, qu'il n'existe pas d'épisode, en dehors de ceux de A et de B, ayant des occurrences dont la durée appartienne à l'intervalle de durée d'occurrence de $A \cup B$ (intervalle englobant les intervalles de durée d'occurrence de A et de B), et que, ni A, ni B n'est partitionnable en deux sous-ensembles ordonnés par la relation $\prec$. Il est clair qu'une expiration propre est une expiration, bien que la réciproque soit fausse (cf. exemple 8).

expiration propre	intervalle d'expiration
$EP(a, \{c, d, e\}, \{h\})$	$]6; 8[$
$EP(a, \{h\}, \{g\})$	$]10; 15[$
$EP(b, \{f\}, \{c, d, e\})$	$]3; 6[$
$EP(b, \{c, d, e\}, \{g, h\})$	$]10; 13[$

TAB. 1 – *Expirations propres satisfaites par les logs L_1.*

Exemple 8 Les expirations propres satisfaites par les logs L_1 (cf. figure 2) sont reportées dans le tableau 1 (rappelons que les ensembles P_a et P_b sont traités indépendamment l'un de l'autre). On vérifie également que :
– $L_1 \nvDash EP(b, \{f\}, \{g, h\})$, car $\{\langle b, c\rangle\} \parallel \{\langle b, f\rangle, \langle b, g\rangle, \langle b, h\rangle\}$;
– $L_1 \nvDash EP(b, \{f\}, \{c, d, e, g, h\})$, car $\{\langle b, c\rangle, \langle b, d\rangle, \langle b, e\rangle\} \prec \{\langle b, g\rangle, \langle b, h\rangle\}$;
– $L_1 \nvDash EP(a, \{c, d\}, \{g\})$, car $\{\langle a, e\rangle\} \parallel \{\langle a, c\rangle, \langle a, d\rangle, \langle a, g\rangle\}$.

Proposition 4 *Soient $m \in Msg$, et $A, B \subset P_m$ ($A, B \neq \phi$). S'il existe une transition temporisée, dans le protocole de conversation, entre deux états s_1 et s_2 tels que les ensembles A et B correspondent respectivement aux ensembles de transitions sortant de s_1 et s_2, alors il existe $A' \subseteq A$ et $B' \subseteq B$ ($A', B' \neq \phi$) tels que $L \vDash EP(m, A', B')$.*

Remarque : La réciproque de la proposition 4 est fausse.

Contre-exemple On a $L_1 \vDash EP(a, \{h\}, \{g\})$, alors que les transitions étiquetées par h et g sortent du même état (cf. figure 2). Ceci s'explique par le fait que, dans les logs L_1, après que le message a ait été émis, le message g est toujours plus long à émettre que le message h.

D'après la proposition 4 et l'hypothèse de complétude des logs, puisque chaque transition temporisée engendre la satisfaction d'une expiration propre dans les logs, il est possible de toutes les retrouver. Toutefois, on peut découvrir plus d'expirations propres qu'il n'y a de transitions temporisées, dans le cas où certains messages sont toujours plus long à envoyer (ou à recevoir) que tous les messages associés aux autres transitions du même état. Le théorème suivant exprime le fait que ceci constitue le seul cas d'erreur possible.

Théorème 1 *Soient $m \in Msg$, et $A, B \subset P_m$ ($A, B \neq \phi$). Si $L \vDash EP(m, A, B)$, alors il existe dans le protocole de conversation :*
– *ou bien deux états s_1 et s_2 tels que s_2 soit relié à s_1 par une transition temporisée, A corresponde à un sous-ensemble des transitions sortant de s_1, et B corresponde à un sous-ensemble des transitions sortant de s_2,*
– *ou bien un état s tel que $A \cup B$ corresponde à un sous-ensemble des transitions sortant de s, et les messages de B soient toujours plus longs à émettre que les messages de A.*

Bien que l'on ne puisse établir une correspondance totale entre ces objets, les expirations propres représentent, en pratique, le meilleur équivalent possible des transitions temporisées. En effet, le théorème 1 nous assure que, si l'on découvre une expiration propre dans les logs, alors il existe une transition temporisée dans le protocole de conversation, ou alors on est en présence de messages plus longs à émettre que d'autres, sachant que les logs seuls ne permettent pas de déceler si l'on se trouve dans un tel cas de figure. Ce résultat justifie la pertinence de la mise en place d'une méthode de découverte des transitions temporisées basée sur la recherche des expirations propres satisfaites par les logs.

La méthode de découverte «naïve» consiste à générer toutes les expirations propres possibles, et à tester pour chacune d'entre elles si les conditions de la définition 7 sont vérifiées. Cette méthode est cependant doublement exponentielle car, d'une part le nombre d'expirations propres possibles est exponentiel, et d'autre part pour chaque couple (A, B) de sous-ensembles de P_m comparables grâce à la relation $\prec$, il est nécessaire de vérifier que tous les

sous-ensembles de A et de B sont incomparables. Aussi, travaillons-nous actuellement à la définition d'une caractérisation des expirations propres conduisant à un algorithme de découverte polynomial. Cette caractérisation est basée sur la construction d'une partition de chaque sous-ensemble P_m d'épisodes. Elle est actuellement en cours de démonstration.

Rappelons que les éléments de la partition $\{P_m \mid m \in Msg\}$ de Ep sont traités séparément. A chaque partie P_m va donc correspondre un ensemble d'expirations propres qui lui sont associées. Ce procédé peut sembler redondant, dans le sens où, si deux transitions étiquetées respectivement par les messages a et b arrivent sur un même état, d'où sort une transition temporisée, nous allons trouver que deux expirations propres différentes sont satisfaites dans les logs (une pour P_a et une autre pour P_b), et les interpréter comme étant deux transitions temporisées différentes. Il est possible de résoudre ce problème en faisant des recoupements entre les différents ensembles d'expirations propres. Ceci permettra également de rejeter certaines expirations propres qui ne peuvent correspondre à des transitions temporisées.

Exemple 9 Considérons les expirations propres satisfaites par les logs L_1 (cf. tableau 1). L'expiration propre $EP(a, \{h\}, \{g\})$ (associée à P_a) ne peut correspondre à une transition temporisée, car h et g interviennent ensemble dans l'expiration propre $EP(b, \{c, d, e\}, \{g, h\})$ (associée à P_b). D'après le théorème 1, on sait que h et g correspondent à des transitions sortant du même état. Finalement, on trouve deux transitions temporisées : l'une correspondant à $EP(a, \{c, d, e\}, \{h\})$ et $EP(b, \{c, d, e\}, \{g, h\})$, et l'autre à $EP(b, \{f\}, \{c, d, e\})$.

5 Conclusions et perspectives

Notre travail se situe dans le contexte de l'extraction du protocole de conversation temporisé d'un service Web à partir de ses logs d'exécution. Il traite de la découverte des transitions temporisées, et constitue, à notre connaissance, la première contribution apportée à la résolution de ce problème. Notre apport consiste en un cadre formel aboutissant à la définition de la notion d'expiration. Nous avons montré que l'ensemble des expirations propres satisfaites par les logs constitue une caractérisation de l'ensemble des transitions temporisées présentes dans le protocole de conversation d'un service.

Du fait qu'il concerne les aspects temporels du protocole de conversation, notre résultat s'inscrit en complément des travaux existants. Nous envisageons d'intégrer l'algorithme de découverte des transitions temporisées sur lequel nous travaillons à la méthode d'extraction du protocole de conversation (non temporisé) proposée par Motahari et al. (2006), au sein d'une plateforme commune de gestion de services Web. Ceci nous permettra de pouvoir effectuer des tests à grande échelle de notre méthode.

Signalons également que nous avons comme objectif d'élargir le cadre formel présenté ici. Nous envisageons pour cela d'essayer de relâcher les contraintes fixées au départ (par exemple le fait que les transitions du protocole de conversation soient étiquetées de façon unique, ou qu'il n'existe pas de transition temporisée menant dans un état final). La solution permettant de pallier ces limites pourrait être d'effectuer un pré-traitement sur les données, afin de se ramener au cas particulier défini par nos hypothèses de travail. Il serait également intéressant de prendre en compte le « bruit » présent dans les données.

Ce travail s'inscrit dans le cadre du projet *ServiceMosaic*[2] regroupant plusieurs équipes de recherche. ServiceMosaic est un prototype de plateforme visant à permettre la modélisation, l'analyse et la gestion de services Web. Les principaux objectifs du projet sont la définition de modèles pour la description, l'orchestration et la composition de services, la spécification d'une algèbre pour une analyse de haut niveau, ainsi que la création d'outils de développement et de gestion de services.

Références

Benatallah, B., F. Casati, J. Ponge, et F. Toumani (2005a). Compatibility and replaceability analysis for timed web service protocols. In *Proceedings of BDA 2005*, Saint-Malo, France.

Benatallah, B., F. Casati, J. Ponge, et F. Toumani (2005b). On temporal abstractions of web services protocols. In *Proceedings of CAiSE Forum 2005*, Porto, Portugal. Springer.

Benatallah, B., F. Casati, et F. Toumani (2004). Analysis and management of web service protocols. In *Conceptual Modeling - ER 2004*, Shanghai, China, pp. 524–541. Springer.

Cook, J. E. et A. L. Wolf (1998). Discovering models of software processes from event-based data. *ACM Transactions on Software Engineering and Methodology 7*(3), 215–249.

Devaurs, D., F. De Marchi, et M.-S. Hacid (2006). Caractérisation des transitions temporisées dans les logs de services web. Technical Report RR-LIRIS-2006-020, LIRIS, Lyon, France.

Dustdar, S., R. Gombotz, et K. Baïna (2004). Web services interaction mining. Technical Report TUV-1841-2004-16, Technical University of Vienna, Vienna, Austria.

Mannila, H., H. Toivonen, et A. I. Verkamo (1997). Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery 1*(3), 259–289.

Motahari, H., B. Benatallah, et R. Saint-Paul (2006). Protocol discovery from imperfect service interaction data. In *Proceedings of the VLDB 2006 Ph.D. workshop*, Seoul, Korea.

Parekh, R. et V. Honavar (2000). Grammar inference, automata induction, and language acquisition. In *Handbook of natural language processing*, pp. 727–764. Marcel Dekker.

van der Aalst, W., T. Weijters, et L. Maruster (2004). Workflow mining: Discovering process models from event logs. *IEEE - TKDE 16*(9), 1128–1142.

Summary

The knowledge of the business protocol of a Web service is very important, for both clients and providers, because it represents a model of its external behaviour. However, it is often not specified during the design phase. The context of our work is the discovery of the timed business protocol of an existing service from its execution data. We consider an important subproblem: the discovery of the timed transitions (i.e. the state changes related to temporal constraints). We present a formal framework where we define the *proper timeouts*, which represent in the logs an equivalent of what the timed transitions are in the business protocol. To the best of our knowledge, it is the first contribution to this problem.

[2]Partiellement financé par le programme «Jeunes Chercheuses et Jeunes Chercheurs» de l'ANR n° JCJC06_134393, 2007–2010. http://servicemosaic.isima.fr/