

THE ESTIMATION OF INVOCATION COST FOR COMPOSITE SERVICES

Moe Thandar Wynn, David Edmond and Stephen Milliner
Research Centre for IT Innovation
Queensland University of Technology
GPO Box 2434, Brisbane QLD 4001, Australia
Phone:(+617)3864 9488, Fax:(+617)3864 9390
{*m.wynn,d.edmond,s.milliner*} @qut.edu.au

ABSTRACT

When a new electronic service is composed from existing ones, the characteristics of the component services will naturally have a major influence on both the functional and the non-functional characteristics of the composite service. When more than one service satisfy the specified functional requirements of a given component for a composite service, these services can be differentiated on the basis of their non-functional properties. To ensure proper evaluation of various composition options, we propose to estimate the corresponding non-functional properties of a composite service during design time by taking into account the characteristics of the candidate services and the composition constructs used to integrate these services. In this paper, we describe the specification and estimation of the invocation cost based on different pricing models and the composition constructs possible in the workflow patterns for composite services.

Keywords: electronic services, web services, service composition, non-functional properties, service planning, cost and pricing models, workflow patterns

I. INTRODUCTION

Achieving economically viable business functionality requires that services are carefully planned and executed. No rational organisation is going to sell a service that does not make a profit and no rational organisation is going to use a service that is not cost effective, robust and timely. However, much of the current research into electronic services ignores crucial aspects such as cost and timeliness. There is an underlying assumption that these services will be free, immediately accessible, instantaneous to execute and globally available always. These unrealistic assumptions will hinder e-service adoption and the corresponding potential for economic benefit.

A composite service relies on other electronic services to perform certain activities in order to fulfil its requirements. The responsibility for coordination and management of these services rests with the composite service to ensure that the overall business goals can be achieved. When a new electronic service is composed from existing ones, the characteristics of the component services will naturally have a major influence on both the functional and

the non-functional characteristics of the composite service. The functionality of the composite service can be seen as the integration of different functionalities from the participating services. For instance, when we combine the functionalities provided by stock quote retrieval, credit checking and online trading services, the composite service is capable of providing on-line stock purchase service. The new service can be initiated with real-time stock retrieval, then checking the intended purchase amount against the credit available to the purchaser is carried out and followed finally by the stock trade execution.

When more than one service satisfies the specified functional requirements of a given component for a composite service, these services can be differentiated on the basis of their non-functional properties. The goal is to find services that can fulfil not only the functional requirements but also the non-functional requirements such as cost and temporal constraints. We are interested in investigating how the non-functional properties can be specified, how their values can be used for differentiating services during service selection for composition, and how they can impact on the non-functional properties of potential composite service. We believe that proper evaluation of non-functional characteristics plays a crucial part in successful development and the use of composite services in business scenarios. In particular, the information on the cost of composition is essential for decision making and also for price setting of the resulting composite service. We propose that certain non-functional characteristics of the composite service such as invocation cost can be estimated from the non-functional characteristics of service components together with their configuration pattern in the composition.

Consider a simple travel-booking scenario where there are three activities, namely, airline ticket booking A, concert ticket booking B and city tour booking C with associated costs per invocation from third-party web services of \$10, \$6 and \$4 respectively (assuming that there is a cost associated with each service). For a composition scenario whereby all three of these services are executed in sequence, the total invocation cost would be estimated as \$20. Consider another possibility whereby a customer can choose to go to a concert, to take a tour of the city or to do both. In this case, the estimated cost of invocation varies

depending on which bookings a customer decides to make. The three possible paths are ABC, AB and AC and the costs for invocation will be \$20, \$16 and \$14 respectively. From this discussion, it can be observed that even though the costs for service components strongly influence the estimation of invocation costs for the resulting service, they are not deterministic when a decision needs to be made. The composition constructs used, i.e. how the component services are configured in the business process influence the cost of composite service.

The organisation of the rest of the paper is as follows: In section 2, we propose the need for complex pricing models for web services, identify some common pricing models and describe how they can be taken into account in cost estimation. In section 3, cost estimation models for various workflow patterns are presented. In section 4, a number of scenarios are introduced to demonstrate the cost estimation process. Section 5 mentions the related work and section 6 summarises our research on cost estimation for composite services.

II. PRICING MODELS

In the travel booking scenario, we have simplified the situation with the cost for web services stated as \$ per invocation pricing model. This price model is set by the service provider and the price for invocation is disclosed to the potential service requestors as part of web service description. The pricing models for the current web services tend to be either free or “per-click” / “per invocation” models [14], [10]. According to [2], the dynamic nature of business-to-business electronic commerce has produced a recent shift away from fixed pricing and towards flexible pricing. Flexible pricing is said to include both differential pricing, in which different buyers may receive different prices based on expected valuations, and dynamic-pricing mechanisms, where prices and conditions are based on bids by market participants. We strongly believe that there is a need for more sophisticated web service pricing models that reflect the different ways in which web services can be charged.

In determining the invocation cost of a composite service, we need to take into account the pricing structure of component services. To do that, we need to understand the various pricing models that can be employed by businesses to raise revenue by supplying web services. The original and most basic pricing methodologies are simple definitions of the units provided multiplied by a rate or price to be charged for each unit [5]. The use of services can be measured in a number of ways, for example in terms of resources used (people, equipments) or events handled. Prices can be expressed in many different ways using fixed price per unit, fixed price with guaranteed minimum and maximum utilization, variable price per unit based on consumption ranges or other factors, prices based on cost plus profit margin and prices based on prevailing marketplace rates [5]. Businesses have traditionally differentiated their

prices based on customer groups (group pricing), product features (versioning), sales volume (volume discounts) or customers’ utility (value-based pricing) [8]. Some of the transaction settlement models mentioned in [9] include subscription, metered, facilitated, escrow, and swap.

The cost estimation model takes into account a number of ways in which web services’ price can be given. The simplest ones are either free service (Cost=0) or cost per invocation. When the cost of using a particular web service is given as per invocation cost, these cost figures can be used directly in the estimation. More complex pricing models for web services would be subscription, lease or registration models. These prices could be determined based on volume (eg. per 100 transactions) or based on the time period (eg. monthly, annually). In this case, the price given does not represent per invocation cost and therefore, our planning model will need to derive average per invocation cost before estimating the overall cost of the service per invocation. When periodic prices such as monthly or annual fees are used for web services, we will need to consider the total cost per period as well as average invocation cost per transaction. We also need to take into account the behaviour of marginal cost for different pricing models. Marginal cost is the additional cost incurred to produce one extra unit of goods. We can see that per invocation pricing models result in constant marginal cost and the volume of transactions have no impact on service selection (Figure 1). However, with subscriptions and registration models, there is diminishing marginal cost, that is, the marginal cost decreases as the volume increases. Hence, the cost estimated is the cost for a given volume estimate and can easily fluctuate. The accuracy of average invocation cost calculated depends on the estimated volume of transactions.

The price can also be given as a range with a number of

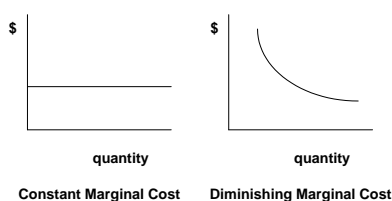


Fig. 1. Marginal cost

discounts either based on the customer type or on volume. Assume that the price for a web service is stated as \$5 per invocation for business customers and \$8 otherwise. Our planning model will need to recognise the discount available and use the appropriate price when estimating the total cost. This time assume that the price for a web service is stated as \$5 per invocation and 10% discount if total transactions per month is at least 100. Our model should query the designer of the composite service regarding the estimated sale volume and consider the discount, if applicable.

III. COST ESTIMATIONS FOR WORKFLOW PATTERNS

We realise that the very use of service components will affect the overall characteristics of the composite service. We are interested in how the individual non-functional characteristic is affected by the use of different composition constructs. Composition logic refers to the way a composite service is constructed in terms of its constituent services. The basic structured activities described in BPEL4WS include sequence, switch, while, pick, flow [6]. The process constructs identified by [1] include sequence, concurrent, split, split and join, unordered, choice, if-then-else, repeat-until and repeat-while. In our work, we focus on the composition constructs described in the workflow patterns [12]. In the workflow patterns research, the requirements for workflow languages are indicated through workflow patterns. A number of workflow patterns are identified and addressed in order to identify comprehensive workflow functionality. The 20 workflow patterns are categorised into 6 main groups: basic control flow patterns, advanced branching and synchronisation patterns, structural patterns, patterns involving multiple instances, state-based patterns and cancellation patterns. [12] have also performed in-depth analysis and comparison of a number of commercially available workflow management systems. As discussed in the paper by [13], the current web services composition languages such as BPEL4WS, XLANG, WSFL are also found to support a subset of these workflow patterns. We believe that workflow patterns represent a rather complete list of constructs possible for business process modelling and that they provide an appropriate starting point in dealing with business processes for our planning model.

The variability in estimated cost for composite service depends on which services are used and how many times they are used in the context of composition. Therefore, the sequence and the parallel split patterns will result in the same cost estimation provided that the same web services components are used in both scenarios. We also found that some workflow patterns such as exclusive choice, multiple choice are more important to our work on estimation of cost due to the nature of selection of services based on conditions or data. When a particular construct contributes to a different cost estimation, we will calculate the cost for these different paths, rather than calculating an average cost figure. We strongly believe that the detailed cost information regarding the various paths are essential in service selection and planning for composition. In the subsections below, the impact on various workflow patterns on cost estimations will be discussed in detail. We have adopted the graphical notation of Yet Another Workflow Language (YAWL) for our discussion on the workflow patterns [11]. Figure 2 shows some of the modelling elements of YAWL that will be used in this paper.

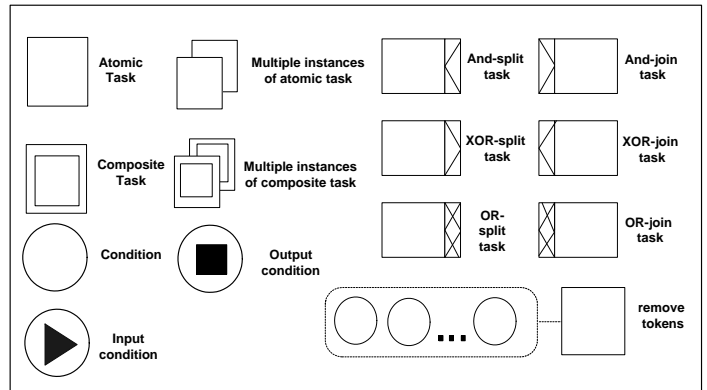


Fig. 2. Symbols in YAWL

A. Basic control flow patterns

These patterns capture elementary aspects of process control and are supported in most standards.

- *Sequence*

Sequential routing is when an activity in a workflow process is enabled after the completion of another activity in the same process. When all the activities in the workflow are invoked in sequence, the formula for the total invocation cost would be a sum of the costs of all the service components used in the composition:

$$C = \sum C_i$$

This base formula is applicable when the business process only uses sequential or parallel constructs and when there are no transitional conditions, choice or loop constructs.

- *Parallel (AND split)*

Parallel split describes a point in the workflow process where a single thread of control splits into multiple threads of control which can be executed in parallel, thus allowing activities to be executed simultaneously or in any order. In terms of cost estimation, the total cost is equivalent to the sum of all the service components used, similar to sequence construct. Figure 3 shows that after executing activity A, activities B, C and D must be carried out in any order. The execution traces can be either ABCD, ACBD or ADCB. The cost will be the sum of the costs for these four activities. This pattern does not contribute to different estimations as there is no conditional invocation of activities.

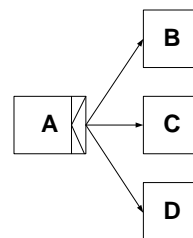


Fig. 3. AND split

- *Synchronization (AND join)*

This is a point in the workflow process where multiple parallel activities converge into one single thread of control, thus synchronizing multiple threads. This type of join only results in one cost estimation as the activity afterwards is executed only once. In the Figure 4, no matter which activities A, B or C are executed, D will be executed only once.

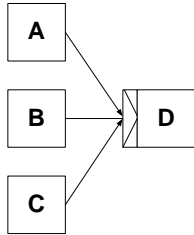


Fig. 4. AND join

- *Exclusive choice (XOR split)*

This is a point in the workflow process where, based on a decision or workflow control data, one of several branches is chosen. For this construct, the number of different cost estimations will depend on the number of possible activities afterwards. Figure 5 describes a scenario whereby after executing activity A, there is an exclusive choice between invoking activities B, C or D. The possible paths are AB, AC and AD and there will be three different cost estimations C_{AB} , C_{AC} , C_{AD} calculated using the base formula. For the path AB, the cost will be $C_{AB} = C_A + C_B$.

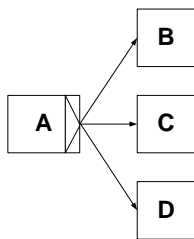


Fig. 5. XOR split

- *Simple merge (XOR join)*

This is a point in the workflow process where two or more alternative branches come together without synchronization. It is an assumption of this pattern that none of the alternative branches is ever executed in parallel. This type of join only results in one estimation as the activity after the XOR join construct is executed only once. In the Figure 6, no matter which activities A, B or C is executed, D will be executed only once.

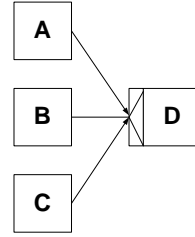


Fig. 6. XOR join

B. Advanced branching and synchronization patterns

The four patterns identified as advanced branching and synchronization patterns include: multi-choice, synchronizing merge, multi-merge and discriminator.

- *Multi-choice (OR split)*

This is a point in the workflow process where, based on a decision or workflow control data, a number of branches are chosen. This pattern indicates the possibility that a number of branches can be chosen. The cost estimation will depend on the combination of activities that can be selected. We can calculate the number of possible paths for a given number of activities by using the formula for combinations. The number of k-combinations of an n-set is the binomial coefficient ${}_n C_k = n!/[k!(n-k)!]$. For instance in Figure 7, there are three paths that can be taken and we first calculate the number of combinations ${}_3 C_{1,3} C_2$ and ${}_3 C_3$ which are 3,3,1 respectively. Hence, there are seven possible paths which can be listed as ABCD, ABC, ACD, ABD, AB, AC and AD. The cost estimations for these seven paths can then be calculated using basic summation formula.

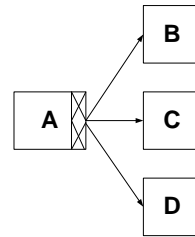


Fig. 7. OR split

- *Synchronizing merge*

This is a point in the workflow process where multiple paths converge into a single thread. If only one path is taken, the alternative branches should reconverge without synchronization. The use of three condition symbols is to indicate that three tokens can be passed from the activities and if more than one path is taken, there will be synchronization. The synchronizing merge pattern indicates that the activities after the synchronizing merge will be invoked exactly once. This pattern is likely to be used together with multiple-choice pattern as shown in Figure 8. The seven possible paths are ABE, ACE, ADE, ABCE, ACDE, ABDE

and ABCDE. It can be seen that no matter which activities (B, C, D) are selected, there is only one execution of E.

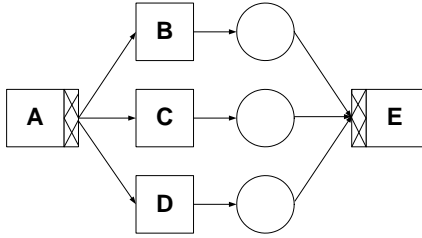


Fig. 8. Synchronizing merge preceded by multi-choice

- *Multi-merge*

This is a point in a workflow process where two or more branches reconverge without synchronization. If more than one branch gets activated, possibly concurrently, the activity following the merge is started for every activation of every incoming branch. This pattern indicates possible multiple invocation for activities after the merge construct. The number of times the activities after multi-merge will be executed is determined by the number of activities in each combination (k). In Figure 9, the cost estimation needs to observe how many paths are possible for the multi-choice pattern using combinations formula and take into account the possible number of times the activity afterwards can be executed. The seven possible paths are ABE, ACE, ADE, ABECE, ACEDE, ABEDE and ABCEDE. In Figure 10, E will be executed three times, ABCEDE. It can be seen that the number of times E is executed depends on the number of activities (B,C,D) invoked for each path.

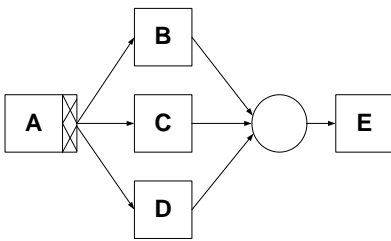


Fig. 9. Multi merge preceded by an OR split

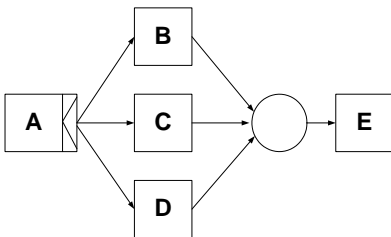


Fig. 10. Multi merge preceded by an AND split

- *Discriminator*

This pattern describes a point in a workflow process that waits for one of the incoming branches to complete before activating the subsequent activity. From that moment on, it cancels other activities within the scope as defined by the box with the dotted lines in the Figure 11. As in the case of simple merge, the activities after the discriminator will only be executed once. However, we need to take into account the fact that it will cancel other activities that has been enabled. Assume that B puts a token in the condition and enables E. That results in activities C and D being cancelled. C and D can be in two states, being invoked and still waiting. In terms of cost estimation, if the other activities are waiting to be enabled, then the cost will be ABE. If the other activities have been completed, even though they are cancelled, there will be a cost associated with it and it will be ABCDE. The question is how much will it cost if you cancel activities C and D midway. We can assume that the cost will be the same as successful invocation (the maximum penalty).

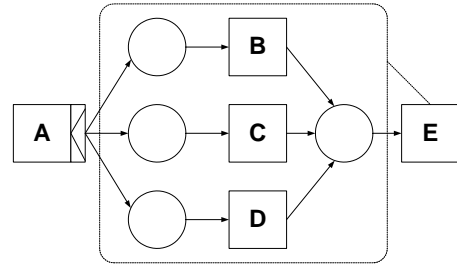


Fig. 11. Discriminator

C. Patterns involving multiple instances

From a theoretical point of view, the concept of multiple instances corresponds to multiple threads of execution referring to a shared definition. From a practical viewpoint, it means that an activity in a workflow graph can have more than one running, active instance at the same time. If the instances need to be synchronized, the number of instances is highly relevant and three patterns with synchronization and one pattern for multiple instances without synchronization are identified. Patterns involving multiple instances will have impact on estimation of invocation cost. The cost estimation for all four patterns involving multiple instances will take into account the number of instances of a given activity.

$$C_{MI} = n C_i$$

where C represents the cost, n is the number of instances. We recognise that in some cases, this number will not be known at design time. In those situations, n could be estimated either using historical data or using projected estimates and probabilities.

- *Multiple instances without synchronization*

Within the context of a single case multiple instances of an activity can be created and there is no need to synchronize these threads. In terms of cost estimation, we need to know how many instances can be created.

- *MI with a priori design time knowledge*

For one process instance an activity is enable multiple times. The number of instances of a given activity for a given process instance is known at design time. Once all instances are completed some other activity needs to be started. This is very simple case for cost estimation as the number of instances will be known at design time.

- *MI with a priori runtime knowledge*

For one case an activity is enabled multiple times. The number of instances of a given activity for a given case varies and may depend on characteristics of the case or availability of resources but is known at some stage during runtime, before the instances of that activity have to be created. Once all instances are completed some other activity needs to be started. this pattern is described in Figure 12. The first two parameters describe the minimum and the maximum number of instances. The third parameter can be used to specify a threshold or inf for no threshold. The fourth parameter describes whether there is a possibility of creating new instances of the fly (static or dynamic).

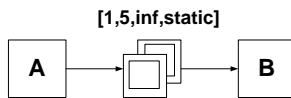


Fig. 12. Multiple Instance pattern with a priori runtime knowledge

- *MI with no a priori runtime knowledge*

For one case an activity is enable multiple times. The number of instances of a given activity for a given case is not known during design time, nor is it known at any stage during runtime, before the instances of that activity have to be created. The difference with the previous pattern is that even while some of the instances are being executed or already completed, new ones can be created. The fourth parameter will be specified as “dynamic”.

D. Cancellation patterns

Two cancellation patterns are described in workflow patterns and they are cancel activity and cancel case patterns. The patterns indicate the possibility of activity or a workflow not being invoked in certain situations. Cancellation is taken to mean both disabling an activity and cancelling the enabled activity.

- *Cancel activity*

An enabled activity is disabled, i.e. a thread waiting for the execution of an activity is removed. In the Figure 13, it can be seen that when there is a token enabling Cancel A activity, then activity A would not be instantiated. This is important for estimation, to indicate the cost when an

activity is cancelled. When we encounter cancel activity construct, we need to consider the possible paths with activities that can be cancelled. In general, there will be three possible estimation paths, one where the cancel activity is not executed and two where the cancel activity is executed (one where the activities have been started i.e. the invocation cost have been incurred and one where the activities have not been enabled i.e. no cost has been incurred).

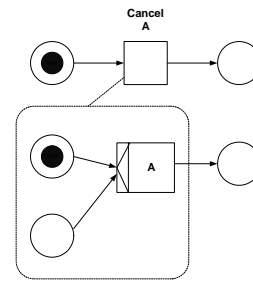


Fig. 13. Cancel activity

- *Cancel case*

Cancel case describes a situation where a workflow instance is removed completely (i.e., even if parts of the process are instantiated multiple times, all descendants are removed). In Figure 14, when there is a token to instantiate activity E, this will remove all the tokens from the other activities and finish the workflow instance by cancelling. This is important for estimation to indicate the cost when the business process is cancelled. We need to consider the cost for web services that have already been invoked at the time the workflow case is cancelled. There will be two estimations: one which provides the cost information for all activities up to and including the activity that results in cancellation. This represents the cost incurred before the activities are cancelled. The alternative cost estimation can be calculated for the situation where the activity enabling cancel case is not invoked. This will represent the cost for all the activities without taking into account the activity for cancellation.

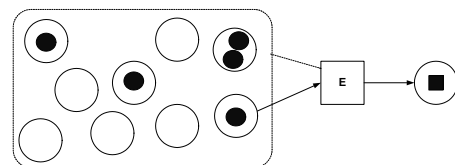


Fig. 14. Cancel case

E. State-based patterns

In real workflows, most workflow instances are in a state awaiting processing rather than being processed and these patterns can be used where an explicit notation of state is required.

- *Deferred choice*

A point in the workflow process where one of several branches is chosen. The choice is delayed until the processing in one of the alternative branches is actually started, i.e., the moment of choice is as late as possible. In contrast to XOR-split, the choice is not made explicitly (based on data or a decision) but several alternatives are offered to the environment. However, in contrast to the AND-split, only one of the alternatives is executed. This means that once the environment activates one of the branches the other alternative branches are withdrawn. Even though the moment of choice is at runtime, this does not affect the estimation of cost for different paths. Therefore, the number of possible estimations is the same as the number of alternatives. The cost estimation will be calculated for all the possible paths using summation formula.

- *Interleaved parallel routing*

A set of activities is executed in an arbitrary order. Each activity in the set is executed, the order is decided at run-time and no two activities are executed at the same moment. Even though the order is determined at runtime, we can estimate the cost because we know which activities will be instantiated at design time. The basic summation formula is applied.

- *Milestone*

The enabling of an activity depends on the case being in a specified state, i.e. the activity is only enabled if a certain milestone has been reached which did not expire yet. We need to recognise the possibility of not invoking an activity because of a milestone. The possible paths for the workflow need to be calculated.

F. Structural patterns

The two structural patterns are used to illustrate typical restrictions imposed on workflow specifications and their consequences. As they are not represented as constructs in the workflow, we are not able to determine cost estimations for structural patterns.

- *Arbitrary cycles*

This is a point in a workflow process where one or more activities can be done repeatedly. Most workflow engines provide support only for structured cycles either through decomposition construct or through a special loop construct.

- *Implicit termination*

A given subprocess should be terminated when there is nothing else to be done. In other words, there are no active activities in the workflow and no other activity can be made active (and at the same time the workflow is not in deadlock).

G. Findings

From the above discussion, it can be seen that the cost estimation formula for sequence and parallel split patterns is the sum of the cost of all the components in the workflow. We found that the order of execution is not a major issue in cost estimation. When a process model consists of constructs that allow the process to decide an execution path from many possible paths at runtime either using explicit constructs or transitional conditions, we need to estimate the invocation costs for all the possible paths for a business process. A general rule would be to work out all the possible paths from the model and associate costs for the activities invoked in each path. The exclusive choice and deferred choice patterns will result in different cost estimations based on the number of possible activities that can be invoked.

The multi-choice pattern is very significant for cost estimation, as there are a number of possible alternative paths and we need to determine the number of paths as well as the activities associated with each path. In terms of merge constructs, the synchronization, simple merge and synchronizing merge and the discriminator patterns have similar effect on the estimation of cost with the activities afterwards are only executed once. The multi-merge pattern is more significant in the sense that the number of times the activities afterwards will be invoked is equal to the number of times the possible alternatives which are invoked. The four multiple instances patterns described share one cost formula with the only difference being the point at which the number of instances will be known (design time or runtime knowledge). The cancellation patterns result in two estimations, one for cost estimation without cancelling and one for cost estimation when activity or case is cancelled. We also found that it is not possible to provide cost estimations for certain workflow patterns such as arbitrary cycles, implicit termination due to the limitation that estimation is done during design time.

IV. SCENARIOS

In this section, we demonstrate how cost estimation can be carried out using the six scenarios.

A. Travel booking scenario

This scenario describes holiday travel booking involving three activities, namely, airline ticket booking (A), concert ticket booking (C) and city tour booking (T) as shown in Figure 15. The OR split construct is used to model the fact that a customer can choose to go to a concert, to take a tour of the city or to do both. In this case, the estimated cost of invocation varies depending on which bookings a customer decides to make. Assume that associated costs per invocation are $A=10$, $C=6$ and $T=4$. The cost estimation for the scenario will be calculated as follows:

1. Identify the possible estimation paths for the scenario.
The scenario uses OR split with two possible activities (T, C) after the split. The combinations ${}_2C_1$ and ${}_2C_2$ give a total of three possible estimations. The three paths are AT, AC and ATC.
2. Use the appropriate cost formula.
The cost estimations are

$$C_{ATC} = 10 + 6 + 4 = 20,$$

$$C_{AT} = 10 + 4 = 14 \text{ and}$$

$$C_{AC} = 10 + 6 = 16.$$

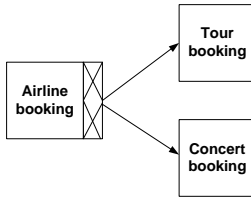


Fig. 15. Travel booking

B. Teaching scenario

This scenario is modelled as shown in Figure 16 and describes teaching three topics namely, SQL, ER and Workflow and then carrying out assessments. After the Teach SQL activity is carried out, Teach ER activity and Assess SQL activity can be performed. Teach ER is followed by Teach Workflow activity. Assess ER activity will only be performed after both Assess SQL and Teach ER have been completed (AND join). Examine activity is carried out after both Teach Workflow and Assess ER have been completed. Assume that “Teach” activities have a cost of \$3 per invocation and assessment and examinations have a cost of \$2 per invocation. The cost estimation can be calculated as follows:

1. Identify the possible estimation paths for the scenario.
It can be seen that this is a fairly complex workflow including sequential, and split and join constructs to model the business process. In terms of cost estimation of this process, it only results in one cost estimate, as there are no conditional transitions and no loops or multi merge pattern.
2. Use the appropriate cost formula.
The formula would be $C = \sum C_i$. The cost estimation for the teaching scenario would then be $3+2+3+2+3+2 = 15$.

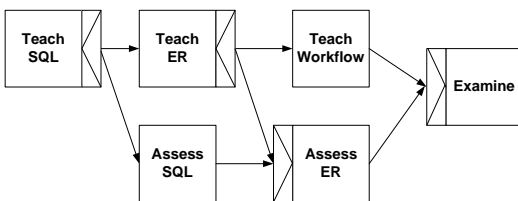


Fig. 16. Teaching and assessment

C. E-commerce scenario

To make an online purchase, the customer begin by browsing (B) for the items in the electronic catalog. When the customer finds an item that he/she would like to buy, the item is added into the shopping cart (S). The customer will keep on adding items into the shopping cart, until he/she has all the items for purchase. Next, the customer initiates the checkout (C) procedure by filling in personal and delivery details. The customer can then decide to provide either the credit card details (V) or use PayPal account (P) to complete the transaction. This scenario can be represented using YAWL notation as shown in Figure 17. Assume that the following costs are associated with invocation of each activity: B=2, S=0.5, C=2, V= 3, P=1. The cost estimation will be calculated as follows:

1. Identify the possible estimation paths for the scenario.
The scenario is mostly sequential with the XOR split to represent the decision between two payment methods. Hence, the two estimation paths are $BSCV$ and $BSCP$.
2. Take into account multiple-instance atomic task (S).
The activity of adding items into the shopping cart will be repeated a number of times (n). The customer will buy at least one item and hence, minimum value of n will be 1.
3. Use the appropriate cost formula.

In this case, C_{MI} will be nC_s . The cost estimations are

$$C_{BSCV} = 2 + 2 + 3 + 0.5n = 7 + 0.5n \text{ and}$$

$$C_{BSCP} = 2 + 2 + 1 + 0.5n = 5 + 0.5n$$
where n represents the number of items purchased. For instance, C_{BSCV} will be 7.5 and C_{BSCP} will be 5.5 for on-line sales involving one item. The value of n is obtained from past sale data and estimates of prospective sales.

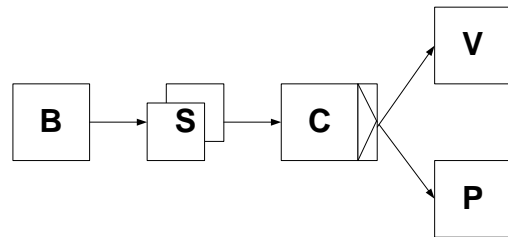


Fig. 17. Online shopping

D. Computer purchase scenario

When a purchase order (O) is received by a computer store, it places orders with a number of suppliers (S) to fulfil various hardware and software requirements. When all the parts have arrived, the computer assembly (A) can be carried out. Delivery (D) is an optional service provided by the store. When everything is completed, the customer is then billed (B). The scenario is described in the Figure 18. Assume that the following cost has been associated with the activities: O=3, S=2, A=5, D=3, B=2. The cost can be estimated as follows:

1. Identify the possible estimation paths for the scenario.
The scenario involves multiple instances of activity S and also uses XOR split with two possibilities. The paths are OSADB (with delivery) and OSAB (without delivery). Assume that the store normally gets their supplies from 3 different suppliers (n=3).
2. Use the appropriate cost formula.
As activity S is represented as multiple instance of atomic task $C_{MI} = nC_S = 2n$ The cost estimations are
 $C_{OSADB} = 3 + 2n + 5 + 3 + 2 = 13 + 2n = 19$ and
 $C_{OSAB} = 3 + 2n + 5 + 2 = 10 + 2n = 16$.

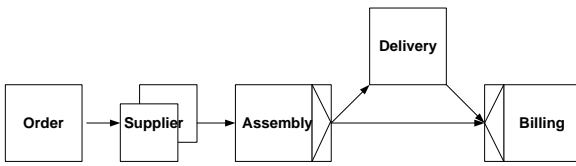


Fig. 18. Computer purchase

E. Procurement negotiation scenario

This scenario 19 describes negotiation with three different suppliers to purchase materials. When an request for materials is received (O), the negotiations can be entered into with suppliers A,B and C. If a particular negotiation which either one of these suppliers is successful, the purchase order (P) is raised and the transaction will start. This also cancels other negotiations taking place with other suppliers. This is a typical scenario for discriminator construct in Workflow patterns. Assume that the following costs are associated with each activity: O=2, A=1, B=1, C=1, P=2.

1. Identify the possible estimation paths for the scenario.
The possible paths are OAP, OBP and OCP. We also need to take into account whether other activities have been invoked at the time. The question is what will be the cost if you cancel your negotiations midway. Here, we assume that the maximum penalty for cancelling would be the same as the invocation cost stated.
2. Use the appropriate cost formula.
 C_{cancel} will be in the range \$0 to \$2 (cost of cancelling 2 out of 3 negotiation activities.) The total cost for the path OAP would be $C_{OAP} + C_{cancel} = 2 + 1 + 2 + 2 = 7$. As the cost for negotiation with three suppliers is given as the same value, the total cost is the same for all three paths.

F. Hotel booking scenario

This scenario captures the process of booking a hotel room from the initial request to reconfirmation. When an initial booking request(I) comes in, a temporary booking(B) is made for the specified dates. Reconfirmation(R) is required a few days prior to arrival and depending on the reconfirmation, the booking is either cancelled(C) or a room

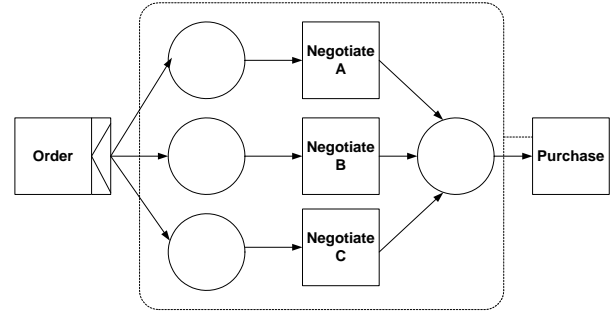


Fig. 19. Procurement scenario

is allocated for the customer(A) as shown in Figure 20. Assume that the following costs are associated with the activities: I=1, B=2, R=1, C=0.5 and A=2.

1. Identify the possible estimation paths for the scenario.
This scenario uses cancel-case workflow pattern and there will be two estimations, one where the activity cancel-booking is invoked and the other where there is no cancellation.

2. Use the appropriate cost formula.

The cost estimations are

$$C_{IBRC} = 1 + 2 + 1 + 0.5 = 4.5 \text{ and}$$

$C_{IBRA} = 1 + 2 + 1 + 2 = 6$. This demonstrates the fact that there is still cost associated with cancelling the processes and that this is an expense not recovered by the revenue from the customer.

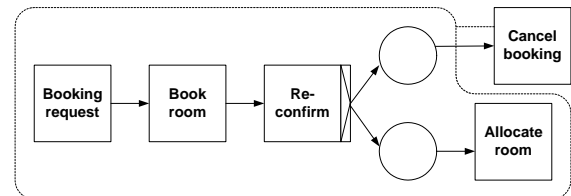


Fig. 20. Hotel booking scenario

G. Analysing cost estimations

The result of cost estimation would be one estimation for the overall workflow or a number of estimations for workflows with certain types of splits or merge constructs. We believe that the developer needs to be aware of the invocation cost for each path in the business process. The estimation figures can be displayed using bar graphs for visual comparison. We can also provide min, avg, max and range information for multiple execution paths. If probability ratios for each path is provided ($0 \leq p \leq 1$), then we can calculate the weighted average cost for the business process.

$$C_{weightedAvg} = \sum p_j C_j$$

where p is the probability and j is a possible path. This detailed information could be used in selecting web services and planning for composition. The estimated cost figures

also enable the business to determine their floor price for the resulting service. We can also observe the effect of pricing models used in composition to find the suitable pricing model for the composite service. It might be desirable for the businesses to use the same or similar pricing model as those used by the service providers.

V. RELATED WORK

A predictive QoS model that makes it possible to compute the quality of service for workflows automatically based on atomic task QoS attributes has been discussed in [4]. Their QoS model takes into account the four dimensions: time, cost, reliability and fidelity and applies an algorithm, which includes a set of reduction rules, to a workflow to compute the overall QoS of a workflow. The six reduction rules used include sequential, parallel, conditional, fault-tolerant, loop and network [3]. The similarities are that we both identify the importance of QoS in web services and attempt to estimate the overall QoS for composite service. The difference lies in the use of pricing models and also a more complete investigation of composition constructs as described in the work on workflow patterns. On the other hand, [15] propose the selection of service components during the execution of a composite service rather than at design-time, using multiple criteria such as price, duration, reliability as well as global constraints and preferences set by the user. We believe that the estimation of prices is not as simple as it is discussed in their proposal (eg. execution price of a plan is given as a sum of every service's execution price.) They propose a global planning approach to optimally select services during the execution of a composite service.

VI. CONCLUSION

There is a strong need for accurate cost estimation of business processes during the planning stage for web service composition. This information can be used to analyse the viability of a proposed composition, as well as its profitability. To ensure proper evaluation of various composition options, we propose to take into account whether the services will be invoked (choice / decision point) and/or how many times the service will be invoked (loops and multiple instances) in the context of composition. We first need to identify all the possible paths and then estimate cost for individual path. The basic summation formula can be applied to the estimation for each path. We expect the service descriptions for electronic services to mention their pricing models in one way or another. A general model for describing pricing in electronic catalogs can be found in [7]. The relevant factors for price such as order unit, territory, customer, valid time period, agreements, price types as well as allowances and charges can be described in this pricing model. In this paper, we have presented some useful pricing models for web services as well as cost estimations for various workflow patterns using scenarios. We believe

that proper evaluation and analysis of non-functional properties during service planning forms the basis for determining profitable e-services for organisations.

REFERENCES

- [1] A. Ankolekar, M. Burstein, J. R. Hobbs, O. Lassila, D. Martin, D. McDermott, S. A. McIlraith, S. Narayanan, M. Paolucci, T. Payne, and K. Sycara, "DAML-S: Web Service Description for the Semantic Web," in *Proceedings of the 1st International Semantic Web Conference (ISWC 2002), June 9-12 2002, Sardinia, Italy, 2002*.
- [2] M. Bichler, J. Kalagnanam, K. Katircioglu, A. J. King, R. D. Lawrence, H. Lee, G. Y. Lin, and Y. Lu, "Applications of flexible pricing in business-to-business electronic commerce," *IBM Systems Journal*, vol. 41, no. 2, pp. 287–302, 2002.
- [3] J. Cardoso, "Quality of service and semantic composition of workflows," PhD Thesis, University of Georgia, 2002.
- [4] J. Cardoso, J. Miller, A. Sheth, and J. Arnold, "Modeling Quality of Service for Workflows and Web Service Processes," 2002, <http://chief.cs.uga.edu/~jam/webwork/geneflow/papers/wspapers.html> accessed on 25 Nov 2002.
- [5] M. F. Corbett, "Moving Forward: Pricing Models that Share Gains," 1 June 2001, <http://www.firmbuilder.com/articles/5/32/571/> downloaded on 8 Nov 2002.
- [6] F. Curbera, Y. Goland, J. Klein, F. Leymann, D. Roller, S. Thatte, and S. Weerawarana, "Business process execution language for web services, version 1.0," 31 July 2002, <http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/> accessed on 13 Nov 2002.
- [7] O. Kelkar, J. Leukel, and V. Schmitz, "Price Modeling in Standards for Electronic Product Catalogs Based on XML," in *Proceedings of the World Wide Web (WWW) Conference, May 7-1, 2002*. Honolulu, Hawaii, USA: ACM, 2002.
- [8] S. Klein and C. Loebbecke, "The Transformation of Pricing Models on the Web: Examples from the Airline Industry," in *Proceedings of the 13th International Bled Electronic Commerce Conference, June 19-21, 2000*, Bled, Slovenia, 2000.
- [9] J. O'Sullivan, D. Edmond, and A. ter Hofstede, "What's in a service? Towards accurate description of non-functional service properties," *Distributed and Parallel Databases Journal - Special Issue on E-Services*, vol. 12, no. 2-3, pp. 117–133, 2002.
- [10] Salcentral.com, "Salcentral web services brokerage," 2002, <http://www.salcentral.com/salnet/webserviceswsdl.asp>.
- [11] W. M. P. van der Aalst and A. ter Hofstede, "YAWL: Yet Another Workflow Language," Queensland University of Technology, FIT Technical report FIT-TR-2002-06, 2002.
- [12] W. M. P. van der Aalst, A. ter Hofstede, B. Kiepuszewski, and A. Barros, "Workflow Patterns," Tech. Rep., 2002, <http://tmitwww.tn.tue.nl/research/patterns/> accessed on 10 Dec 2002.
- [13] P. Wohed, W. M. P. van der Aalst, M. Dumas, and A. ter Hofstede, "Pattern Based Analysis of BPEL4WS," Queensland University of Technology, FIT Technical report FIT-TR-2002-04, 2002.
- [14] xMethods.net, 2002, <http://www.xmethods.com/>.
- [15] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q. Z. Sheng, "Quality driven web service composition," in *Proceedings of the World Wide Web (WWW) Conference, May 20-24, 2003*. Budapest, Hungary: ACM, 2003.