This is the published version:


Available from Deakin Research Online:

http://hdl.handle.net/10536/DRO/DU:30029405

Reproduced with the kind permission of the copyright owner.

Copyright : 2010, IEEE

©2010 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.
Multi-criteria Content Adaptation Service Selection Broker

Mohd Farhan Md Fudzee, Jemal Abawajy and Mustafa Mat Deris*
School of Engineering and Information Technology, Deakin University, Australia
Faculty of Information Technology and Multimedia, Tun Hussein Onn University of Malaysia*
{mfmd, Jemal}@deakin.edu.au, mmustafa@uthm.edu.my

Abstract
In this paper, we propose a service-oriented content adaptation framework and an approach to the Content Adaptation Service Selection (CASS) problem. In particular, the problem is how to assign adaptation tasks (e.g., transcoding, video summarization, etc) together with respective content segments to appropriate adaptation services. Current systems tend to be mostly centralized suffering from single point failures. The proposed algorithm consists of a greedy and single objective assignment function that is constructed on top of an adaptation path tree. The performance of the proposed service selection framework is studied in terms of efficiency of service selection execution under various conditions. The results indicate that the proposed policy performs substantially better than the baseline approach.

1. Introduction
The rapid growth of digital media technologies, networks and Internet appliances have enabled the emergence of novel media content types for various application domains (e.g., e-Commerce, e-Education, and e-entertainment) available in the form of content and services targeted to diverse users with low-cost and ubiquitous access to Internet. Electronic documents are becoming increasingly rich in content and varied in format and style while at the same time client devices are getting increasingly varied in their capabilities. Moreover, the original content is normally developed for desktop computers and is naturally made-up of media objects of different types with complicated structure and layout [1]. Thus, direct content delivery to handheld devices without layout adjustment and content adaptation often leads to disorganization of information [2]. As a result, end users frequently experience frustration when their devices are unable to handle certain media types or the data takes a long time to download [3].

Thus, in order to increase content accessibility and improve end user’s experience within a heterogeneous network environment, a mechanism for the content to be tailored according to the users’ preferences, network characteristics and client device capabilities need to be resolved. This content customising process is referred to as a content adaptation. What is required is a content adaptation approach that could automatically generate any content version from one single original version such that the content is adapted to the device and the user preferences. Although many content adaptation approaches exist [2], [4], [5], [6], [7], most of them tend to be fully or partially centralized suffering from scalability and single-point failures. Recently, a service-oriented content adaptation (SOCA) framework has been proposed to address the content-device mismatch problem [1], [4], [7]. A service, in service-based content adaptation, abstracts a set of functionalities offered by the content adaptation service providers. As SOCA scheme is essentially distributed in nature, an adaptation task can be performed by multiple services. In this case, selecting appropriate services among the many available services is necessary to increase the overall performance of the system in delivering the adapted content to the user and can increase the path determination processing execution up to 30% [8]. Hence, a mechanism is required to map the adaptation tasks to the appropriate adaptation services [4].

In this paper, we propose a service-oriented content adaptation (SOCA) framework. Based on the proposed SOCA framework, a multi-criteria adaptation service selection broker that enables end users to select the best service among the available content adaptation service candidates. Moreover, the proposed algorithm takes into account the different QoS values’ relation towards the score (e.g., positive, negative relation). The performance of the proposed service selection framework is studied in terms of efficiency of service selection execution under various conditions. The results indicate that the proposed policy performs substantially better than the baseline approach.

The rest of the paper is organized as follows. In Section 2, the background and related work is reviewed. SOCA architecture is presented and the
formulation of the service selection problem is described. Section 3 presents the proposed multi-criteria service selection algorithm. The performance evaluation of the proposed algorithm and discussion of the results are presented in Section 4. Finally, we concluded the paper in Section 5.

2. Background and Related Work

In this section we will give a brief description of the SOCA system model, the background work and then formulate the service selection problem. We also review some related works.

2.1. SOCA System Architecture

Figure 1 shows a layered architecture of the SOCA framework. The aim of the SOCA framework is to provide an enhanced user experience by offering value-added content and also to provide flexible and scalable service-based content delivery mechanism. The framework consists of components that provide access to content servers, formulate user request to source format, manage and provide content description (metadata).

![Figure 1: Service-oriented content adaptation](Image)

In the SOCA scheme, the content servers store the content and they are distributed across the Internet. Similarly, there is several content adaptation service providers located in many places in the network. The broker and the service discovery components of the system cooperate to select the best possible service for the query to maximise user satisfaction. The brokers and device capability databases (DCDBs) can be placed in the local proxies in distributed location and should be synchronized timely.

The clients initiate a request for information from content servers via the HTTP protocol [9]. The client may also indicate quality of service (QoS) desired. A broker, on behalf of the client, analyzes the content requested to determine whether a content adaptation is required or not. If adaptation is required, the broker requests the service discovery component to facilitate discovery of a subset of relevant services satisfying client request. The service discovery system returns a set of service handles for service invocation or further communication with the selected services. The broker will then select a set of best content adaptation service providers based on the client’s requirements. Client’s criteria requirements are defined in the service level agreement (SLA) that both client and broker agree to and that the broker refers to when selecting the content adaptation services. Then, the chosen services adapt the content and send it back to the user. Note that a single client query may require multiple content adaptation services. As shown in Figure 1, a client request resulted in 3 services being selected to perform the adaptation.

2.2. Content Adaptation Problem

In the reference system architecture (i.e., Figure 1), there are multiple adaptation service providers with different QoS. As an adaptation task can be achieved by more than one adaptation service providers, choosing appropriate adaptation service providers is an obvious requirement [1]. The multi-criteria content adaptation service selection (CASS) problem of interest can be formulated as follow:

Let \( D = \{d_1, d_2, ..., d_n\} \) be a set of adaptation services and \( C = \{c OBJ_1, c OBJ_2, ..., c OBJ_m\} \) be a set of original contents. Let \( C' = \{c' OBJ_1, c' OBJ_2, ..., c' OBJ_m\} \) represent the content requests from the end-users. Each content request is composed of a series of adaptation tasks \( T = \{t_1, t_2, ..., t_n\} \) that can be performed by multiple adaptation services based on quality of service (QoS) criteria \( Q = \{q_1, q_2, ..., q_n\} \). Example of QoS criteria are time, cost, availability and rating. In this paper, we assume that there is no correlation between the QoS criteria. Client’s QoS requirements are defined in the service level agreement (SLA) that both client and broker agree to and that the broker refers to when selecting the content adaptation services. SLA specification strategy provides a formal method for describing QoS requirements. There are specially-designed service specification languages such as Web Services Agreement Specification [10] and Web Service Level Agreement [11] that can be utilized to interpret and enforce negotiated SLAs.

Given a set of \( D, C, T \) and \( Q \), the CASS problem is how to allocate adaptation tasks (e.g., transcoding, video summarization, etc) together with the content segments to appropriate adaptation services with the aim of achieving user specified QoS. This problem can be generally viewed as the appropriate path selection.
problem (henceforth, we refer to the service selection problem as path determination problem [4]).

To address the adaptation path construction, a directed acyclic graph (DAG) is discussed in [1], [4]. The transformation prescript graph for DAG is organized in serial manner and bounded by the media format. A static path determination criteria (SPDC) policy that suffers from a number of shortcomings is discussed in [1], [4]. In this paper, we propose an algorithm that consists of a greedy and single objective assignment function that is constructed on top of an adaptation path tree. Unlike SPDC, our algorithm uses two relationships for computing service scores. Also, having a single optimal path improves the service selection execution [8]. On the other hand, having multiple optimal paths will require the system to have additional decision rules to choose the best path, with which complicates the determination.

### 3.2. Calculate Aggregate Score

Each path is associated with an aggregate score. A given path score computation is based on QoS rating and the QoS relationship. A service (i.e., node) can be associated with one or more QoS (e.g., rating, reliability, etc.). Also, clients can rate a given QoS as more important than another QoS. For example, if a client prefers to minimize the cost rather than time, cost will have a higher weight factor (value) as compared to time. To represent such client QoS preference, we associate a weight, $0 < w_m < 1$, with each QoS specified by the user.

Given a QoS weight (i.e., $w_m$), a node is computed as a normalized score between $[0, 1]$ and defined as follows:

$$Q = \sum_{i=1}^{n} w_i Q_i$$

where $Q$ represents the number of QoSs for a particular task, and $Q$ represents the total number of QoVs for a particular task.

The sequences of the tasks are arranged based on their dependencies as follows:

$$C \rightarrow t_1, t_2, t_3 \ldots t_n \rightarrow C’.$$  

Starting with task $t_1$, each available service for $t_1$ creates one different node/link to $t_2$, in left to right order along the path score tree. This step is repeated for the next consecutive tasks until $t_n$ is reached (i.e., the desired content version $C’$ is achieved). The combinations of these nodes create a number of the potential adaptation paths. For each task, the available services are created as different nodes. Also, each task is associated with the service’s selection QoSs. For instance, a suitable service for each task can be selected based on time and reputation QoSs. The combinations of these nodes create a number of the potential adaptation paths.

As an example, consider the adaptation case $AP (C, C’, 3, 2, Q)$ such that the initial state $C$ is a full video with Spanish audio and the goal state $C’$ is to have a short animation version of the video with English audio. Further suppose that three adaptation tasks $t = \{t_1, t_2, t_3\}$ and $\{s_{11}, s_{12}, s_{21}, s_{22}, s_{31}, s_{32}\}$ adaptation service providers. An example of $t_1$ is conversion of video to animation, $t_2$ is translation of Spanish to English audio (of the video) and $t_3$ is media summarization of the animation. Further assume that $t_1$ and $t_2$ are independent of each other but both are the predecessors of $t_3$. Further suppose that each task is performed by two different services. For example, $t_1$ is performed by two services $\{s_{11}, s_{12}\}$, $t_2$ is performed by two services $\{s_{21}, s_{22}\}$ and $t_3$ is performed by two services $\{s_{31}, s_{32}\}$. Since there are 3 tasks and each task could be serviced by two service providers, we have 8 possible adaptation paths to select from.

### 3.3. Calculate Aggregate Score

Each path is associated with an aggregate score. A given path score computation is based on QoS rating and the QoS relationship. A service (i.e., node) can be associated with one or more QoS (e.g., rating, reliability, etc.). Also, clients can rate a given QoS as more important than another QoS. For example, if a client prefers to minimize the cost rather than time, cost will have a higher weight factor (value) as compared to time. To represent such client QoS preference, we associate a weight, $0 < w_m < 1$, with each QoS specified by the user.

Given a QoS weight (i.e., $w_m$), a node is computed as a normalized score between $[0, 1]$ and defined as follows:

$$Q = \sum_{i=1}^{n} w_i Q_i$$

where $Q$ represents the number of QoSs for a particular task, and $Q$ represents the total number of QoVs for a particular task.

The sequences of the tasks are arranged based on their dependencies as follows:

$$C \rightarrow t_1, t_2, t_3 \ldots t_n \rightarrow C’.$$  

Starting with task $t_1$, each available service for $t_1$ creates one different node/link to $t_2$, in left to right order along the path score tree. This step is repeated for the next consecutive tasks until $t_n$ is reached (i.e., the desired content version $C’$ is achieved). The combinations of these nodes create a number of the potential adaptation paths. For each task, the available services are created as different nodes. Also, each task is associated with the service’s selection QoSs. For instance, a suitable service for each task can be selected based on time and reputation QoSs. The combinations of these nodes create a number of the potential adaptation paths.

As an example, consider the adaptation case $AP (C, C’, 3, 2, Q)$ such that the initial state $C$ is a full video with Spanish audio and the goal state $C’$ is to have a short animation version of the video with English audio. Further suppose that three adaptation tasks $t = \{t_1, t_2, t_3\}$ and $\{s_{11}, s_{12}, s_{21}, s_{22}, s_{31}, s_{32}\}$ adaptation service providers. An example of $t_1$ is conversion of video to animation, $t_2$ is translation of Spanish to English audio (of the video) and $t_3$ is media summarization of the animation. Further assume that $t_1$ and $t_2$ are independent of each other but both are the predecessors of $t_3$. Further suppose that each task is performed by two different services. For example, $t_1$ is performed by two services $\{s_{11}, s_{12}\}$, $t_2$ is performed by two services $\{s_{21}, s_{22}\}$ and $t_3$ is performed by two services $\{s_{31}, s_{32}\}$. Since there are 3 tasks and each task could be serviced by two service providers, we have 8 possible adaptation paths to select from.

### 3.3. Calculate Aggregate Score

Each path is associated with an aggregate score. A given path score computation is based on QoS rating and the QoS relationship. A service (i.e., node) can be associated with one or more QoS (e.g., rating, reliability, etc.). Also, clients can rate a given QoS as more important than another QoS. For example, if a client prefers to minimize the cost rather than time, cost will have a higher weight factor (value) as compared to time. To represent such client QoS preference, we associate a weight, $0 < w_m < 1$, with each QoS specified by the user.

Given a QoS weight (i.e., $w_m$), a node is computed as a normalized score between $[0, 1]$ and defined as follows:

$$Q = \sum_{i=1}^{n} w_i Q_i$$

where $Q$ represents the number of QoSs for a particular task, and $Q$ represents the total number of QoVs for a particular task.

The sequences of the tasks are arranged based on their dependencies as follows:

$$C \rightarrow t_1, t_2, t_3 \ldots t_n \rightarrow C’.$$  

Starting with task $t_1$, each available service for $t_1$ creates one different node/link to $t_2$, in left to right order along the path score tree. This step is repeated for the next consecutive tasks until $t_n$ is reached (i.e., the desired content version $C’$ is achieved). The combinations of these nodes create a number of the potential adaptation paths. For each task, the available services are created as different nodes. Also, each task is associated with the service’s selection QoSs. For instance, a suitable service for each task can be selected based on time and reputation QoSs. The combinations of these nodes create a number of the potential adaptation paths.

As an example, consider the adaptation case $AP (C, C’, 3, 2, Q)$ such that the initial state $C$ is a full video with Spanish audio and the goal state $C’$ is to have a short animation version of the video with English audio. Further suppose that three adaptation tasks $t = \{t_1, t_2, t_3\}$ and $\{s_{11}, s_{12}, s_{21}, s_{22}, s_{31}, s_{32}\}$ adaptation service providers. An example of $t_1$ is conversion of video to animation, $t_2$ is translation of Spanish to English audio (of the video) and $t_3$ is media summarization of the animation. Further assume that $t_1$ and $t_2$ are independent of each other but both are the predecessors of $t_3$. Further suppose that each task is performed by two different services. For example, $t_1$ is performed by two services $\{s_{11}, s_{12}\}$, $t_2$ is performed by two services $\{s_{21}, s_{22}\}$ and $t_3$ is performed by two services $\{s_{31}, s_{32}\}$. Since there are 3 tasks and each task could be serviced by two service providers, we have 8 possible adaptation paths to select from.
where $q_m$ is the QoS relationship associated with a given path and defined as either positive or negative relationship. Rating, reliability and reputation; service cost, adaptation time and transport time; are some examples of QoS positive relation and negative relation categorization, respectively. Given the value for QoS ($v_i$), the $q_m$ score for the positive relations is determined as follows:

$$q_m = \frac{v_i}{v_{i,\text{max}}}.$$  \hspace{1cm} (3)

In contrast, the $q_m$ score for the negative relationship is defined as follows:

$$q_m = \begin{cases} v_{i,\text{min}} & \text{if } v_{i,\text{min}} > 0, \\ v_i & \text{if } v_{i,\text{min}} = 0. \\ 1 - a & \text{if } v_{i,\text{min}} = 0. \end{cases} \hspace{1cm} (4)$$

Note that the case $v_{i,\text{min}} = 0$ can only occur for the adaptation cost QoS because, it is possible to have free adaptation cost for the time being. However, when adaptation services become commercial, services will definitely charge at least a minimum cost. As such, the probability of using the second formula in (4) is very low. Also note that, it depends on the SLA to determine a QoS relationship of a given path score.

Let $AgS (P)$ be a function that computes the aggregate score for a given adaptation path $P$. $AgS (P)$ is defined as follows:

$$AgS (P) = \sum_{i=1}^{k} n_i.$$

(5)

From (5), the aggregate score for path $P_i$ is computed by adding the nodes’ scores (i.e., $n_i$) along the path $P_i$ and $k$ is the maximum number of nodes in path $P_i$.

### 3.3. Adaptation Service Selection

Each path is associated with an aggregate score. For each adaptation task, we select the best services to perform the tasks from the generated optimal path and assign the tasks to the selected services. The highest aggregate score from the start to the end path will be selected as the optimal path. For example, consider the adaptation case $AP \{C, C', 3, 2, Q \}$. Let us further assume that a suitable service for an adaptation task is selected based on two QoS $Q = \{\text{time, reputation}\}$ such that time ($s_{11} = 0.6s, s_{12} = 1.0s, s_{21} = 0.8s, s_{22} = 1.0s, s_{31} = 0.8s, s_{32} = 1.6s$), reputation ($s_{11} = 4, s_{12} = 5, s_{21} = 5, s_{22} = 3, s_{31} = 4, s_{32} = 4$) and both QoSs are to be equally rate (i.e., $w_{n} = 0.5$ each). Time and reputation QoSs have negative and positive relationship respectively. Using calculate aggregate score function, $P1$ is selected as the optimal path based on the highest aggregate score. The services along $P1$ are chosen to perform the three tasks, as follows:

$$n_i = \sum_{m=1}^{j} q_m \times w_m.$$  \hspace{1cm} (2)

Then, the chosen services adapt the content and send it back to the user. Note that a single client query may require multiple content adaptation services.

### 4. Performance Evaluation

We use simulation to study the efficiency of the service selection execution of the proposed policy. We followed the simulation and verification methodology described in [12]. We used two different workloads: workload 1 (W1) and workload 2 (W2). In W1, each QoS has the same weight while W2 imposes QoS with different weight. These workloads are important to represent the user preference towards the selection of QoS [1], [4]. Data to represent the QoS values are generated based on skew distribution provided by [13]. The skew distribution is useful to fit observed data with “normal-like” shape of the empirical distribution but with lack of symmetry [13] and is practical to represent the QoS values between services [14].

At each run, we generated the number of adaptation tasks ($T$) to be in the range of 1 to 5. The total number of QoS ($C$) for a particular task is set in the range of 1 to 4. We set the number of available service providers ($D$) for a particular task in the range of 2 to 5. The values we used for each parameter are in line with the current literature and also reflect the actual environment. The number of tasks and the QoS for each task used in the experiments are in line with the work of [1], [4]. The number of service providers is chosen based on [8].

We used the static path determination criteria (SPDC) policy [1], [4] as the baseline policy for the purpose of performance comparison. We chose SPDC policy as it is widely accepted and is the closest policy to our policy. SPDC assigns score to each node, which is accumulated to generate aggregate score for each path. We modified the score computation to represent the positive relation’s score for the SPDC as follows:

$$Q_{ij} = \begin{cases} \frac{Q_{ij}^\text{max} - Q_{ij}^\text{min}}{Q_{ij}^\text{max} - Q_{ij}^\text{min}} & \text{if } Q_{ij}^\text{max} - Q_{ij}^\text{min} \neq 0 \\ 1 & \text{if } Q_{ij}^\text{max} - Q_{ij}^\text{min} = 0 \end{cases}.$$  \hspace{1cm} (6)

For comparing the efficiency of the service selection execution of the two policies, first, we study the single optimal path generation which is based on the adaptation path aggregate score, $sop$ defined as follows:

$$sop = \frac{\text{single optimal path generated}}{\text{number of runs}}.$$  \hspace{1cm} (7)

Then, we analyse APDC’s improvement on service selection execution by comparing the $sop$ between the
two policies. This analysis is adopted from [15] and considers the improvement, \( im \) as the ratio between the baseline policy and the proposed policy, defined as follows:

\[
im = \frac{APDC \, sop}{SPDC \, sop} . \tag{8}
\]

We then compute the average service selection execution improvement ratio, defined as follows:

\[
ar = \frac{\sum_{c} (im)}{counter} . \tag{9}
\]

4.1. Results and discussions

Extensive simulation analysis of the proposed policy has been carried out. In this section, we present some results and for complete results, interested readers can refer to [8].


In this subsection, we examine the relative performance of the APDC and SPDC policy with respect to the generated single optimal path. For each simulation, 100 runs were performed. Figure 2 shows the reduction ration (y axis) as a function of the QoS (x-axis). In the simulation, we varied the number of QoS from 1 to 4. The number of tasks and services remain constant (AP < C, C’, 1, 2, 1-4>). Time, cost, rating and reputation QoS are the commonly used QoS in this simulation.

![Figure 2. Single Optimal Path Generation towards QoS](image)

As can be seen from Figure 2, APDC generated higher percentages for single path generation for both W1 and W2 compared to the SPDC. The percentage increases steadily along x-axis for both policies. APDC constantly produces around 90% for both workloads along x-axis. There is a significant different between APDC-W1 and W2 due to the score-based approach implemented in APDC. This figure implies that having different QoS weighting does not much affect single optimal path generation in APDC. In the other hand, QoS weighting do affect the single optimal path generation in SPDC. This also indicates that APDC policy is more stable towards QoS variation compared to SPDC.

Figure 3 shows the reduction ration (y axis) as a function of the tasks (x-axis). In the simulation, we varied the number of tasks from 1 to 5. The number of QoS and services remain constant (AP < C, C’, 1-5, 2, 2>). As can be seen from Figure 3, there is a slight increment of single optimal path generation for both policies and workloads along x-axis. APDC constantly produces 80% for W1 and 90% for W2, along x-axis. Meanwhile, SPDC produces around 70% for W1 and 80% for W2. There is a significant different between (1) APDC-W1 and SPDC-W1 around 30 %, and (2) APDC-W2 and SPDC-W2 around 20%. A 5% margin is observed between APDC W1 and W2 and 10% margin for SPDC W1 and W2. This implies that number of tasks has a considerable impact on single optimal path generation for SPDC and a small impact for APDC. This also indicates that APDC policy is more stable towards tasks variation compared to SPDC.

![Figure 3. Single Optimal Path Generation towards Tasks](image)

Figure 4 shows the reduction ration (y axis) as a function of the services (x-axis). In the simulation, we varied the number of services from 2 to 5. The number of tasks and services remain constant (AP < C, C’, 2, 2-5, 2>). As can be seen from Figure 4, there is a very slight increment of single optimal path generation for both policies and workloads along x-axis. APDC constantly produces 96% for both workloads along x-axis. Meanwhile, SPDC produces around 65% for W1
and 83% for W2. There is a significant difference between (1) APDC-W1 and SPDC-W1 around 25%, and (2) APDC-W2 and SPDC-W2 around 28%. A 6% margin is observed between APDC W1 and W2 and 10% margin for SPDC W1 and W2. This implies that number of services has a considerable impact on single optimal path generation for SPDC and a small impact for APDC. The score-based approach implemented is SPDC contributes to the low percentage and this aligned with discussion in [8].

Figure 4. Single Optimal Path Generation towards Services

Taken as a whole in these three simulations, the single optimal path generation percentage increases for both policies and workloads along x-axis. The proposed APDC policy is notably better in every variation (QoS, tasks and services) of the simulations. In addition, we found that, applying different workload (W1 and W2) increases the percentage within a particular policy.

5. Conclusion and Future Direction

In this paper, we have shown that content adaptation service selection is one of the fundamental problems that need to be resolved in the service-oriented content adaptation systems. We then proposed a multi-criteria adaptation service selection broker that enables end users to select the best service among the available content adaptation service candidates. The proposed algorithm was studied experimentally and the results showed that the proposed algorithm is proved to be substantially efficient in terms of generating single optimal path and improving service selection execution. We also have shown that applying different QoS weighting for services improves service selection execution. In future, we plan to study on how to dynamically generate service composition and practically provide recovery and fault tolerance mechanism based on the proposed model. Our future work includes how to dynamically handle new joining nodes during adaptation process. We also working on how to handle the case when there exist correlation between the QoSs used and to analysis the effect of APDC on space (memory) complexity.

6. References