

A Qualitative Approach to Effort Judgment for Web Service Composition based SOA Implementations

Zheng Li
NICTA and
ANU School of Computer Science
Canberra, Australia
Zheng.Li@nicta.com.au

Liam O'Brien
CSIRO and
ANU School of Computer Science
Canberra, Australia
Liam.O'Brien@csiro.au

Abstract—With more and more availability of services, Web service composition (WSC) based SOA implementations have increasingly become a significant type of SOA projects in practice. However, effort estimation for such a type of SOA project can still be limited because of the numerous and various approaches to WSC. Through viewing WSC based SOA system from a perspective of mechanistic organization, this paper borrows Divide-and-Conquer (D&C) as the generic strategy to narrow down the problem of effort judgment for the entire SOA implementation to that for individual WSCs. Moreover, benefiting from an effort-oriented classification matrix and a set of effort-related hypotheses, we assign scores to effort factors of WSC assisted by a set of rules. These effort scores are used to facilitate qualitatively judging different effort between different types of WSC approaches, and eventually construct an effort checklist for WSC approaches. Finally, this effort checklist can be used together with D&C algorithm to realize the qualitative effort judgment for WSC based SOA implementations.

Keywords—Service-Oriented Architecture (SOA); Web service composition; Divide-and-Conquer (D&C); effort judgment; effort checklist

I. INTRODUCTION

Effort estimation for implementations of Service-Oriented Architecture (SOA) confronts many challenges due to the diversity of SOA projects [25]. However, we can focus on one particular type of SOA implementation, for example service composition based SOA implementation, to decrease the complexity and difficulty of the corresponding research. As enterprises move to having more and more services, and business application software will increasingly rely on subscribing services [26], the major problem in SOA implementations will be service composition and may be less on development of new services. Meanwhile, it has been identified that the benefits of SOA cannot be fully realized until reaching the level of service composition [27]. Therefore, it is worthwhile to narrow down our concerns from generic SOA implementations to service composition based SOA implementations that do not take into account service migration or new service development. Considering Web service is the de facto representation of service in practice, we can further concentrate on the Web service composition (WSC) based SOA implementations.

In the past decade, numerous and various works about composing Web services have been reported in the literature. As a result, it is nearly impossible to collect development data of all the published composition approaches to estimate effort of WSC based SOA implementations quantitatively. Considering “learning by analogy” is also a feasible way to investigate distributed systems like WSC based SOA system [2], this paper borrows Divide-and-Conquer (D&C) as a generic strategy of effort judgment from the organization theory domain. Through the generic strategy D&C, the work on effort judgment for a whole WSC based SOA implementation can be further narrowed down to that for individual WSCs. After applying a set of qualitative effort-related hypotheses to WSC effort factors that are identified by a classification matrix, we employ several rules to assign effort scores to different factors and different types of WSC approaches. For a certain WSC based SOA implementation, we can use the D&C algorithm to further calculate effort scores of different implementation proposals, and therefore conveniently compare the qualitative effort required by different proposals.

This paper is organized as follows. Section II briefly introduces the generic strategy of effort judgment for WSC based SOA implementations. Section III specifies the methodology and procedure of effort score calculation for different type of WSC approaches. A small part of the resulting effort checklist of WSC approaches is listed in Appendix I. Section IV uses a case study to demonstrate the complete qualitative approach of effort judgment proposed in this paper. The conclusion is drawn in Section V.

II. GENERIC STRATEGY OF EFFORT JUDGMENT: DIVIDE-AND-CONQUER FROM AN ORGANIZATIONAL PERSPECTIVE

As previously mentioned, WSC based SOA implementation is one of the implementation styles of SOA. In general, SOA emerges for building large distributed systems [1], which supposes services are decentralized and may be under the control of different owners. Therefore, WSC based SOA projects can be viewed as concrete instances of distributed systems. When unfolding research into distributed systems, we can generally adopt two different approaches [2]: one is learning by doing (to build real distributed systems), while another is learning by analogy (to draw upon ideas from other research areas). Considering human organizations are usually adopted as

analogues of distributed systems [2, 5], we can also treat a WSC based SOA system as a particular organization to use the knowledge of organization theory to inspire the research in effort judgment. Our previous work has viewed SOA systems from an organizational perspective [3]: services are organizational units of an SOA system, while composite services play integrative roles that have the similar responsibilities of managers in human organizations. Therefore, a WSC based SOA system can be viewed as a mechanistic organization that has a clear hierarchical structure.

Mechanistic organization, studied by Burns and Stalker [4], has a rigid management system. With a strict hierarchy of authority, generally, a mechanistic organization uses multiple levels of management to insure proper and centralized decision making. In daily routine, the information and decisions are propagated through each level of the hierarchy. Meanwhile, the tasks in mechanistic organizations are precisely defined and broken down into separately specialized parts. These separate parts will be allocated to members who are the most suitable for them. The individual members are then coordinated through the management system to achieve global goals. Consequently, the hierarchical structure of a mechanistic organization can be viewed as a global goal tree: the global goal is stepwise decomposed into a set of sub-tasks, while the goal is achieved through recomposing the sub-solutions to those sub-tasks. The notion of decomposition and recombination directly engenders D&C approach that allows the organization to use larger groups of work units more efficiently and address problems on a larger scale [5]. Likewise, WSC based SOA systems can also be thought of a D&C way to achieve business goals. When building WSC based SOA systems, we can therefore follow the D&C way to analyze and judge the effort.

The principle underlying D&C is to recursively decompose the problem into smaller sub-problems until all the sub-problems are sufficiently simple enough, and then to solve the sub-problems. Resulting solutions are then recomposed to form an overall solution. As revealed in the principle, one of the advantages of applying the D&C approach to suitable problems is its structural simplicity. Profiting from perhaps the simplest structuring technique, D&C has been identified as a high prior strategy to resolve problems not only in the computer science field but also in politics and sociology fields. No matter where the D&C approach is applied the solution structure can be expressed explicitly in a program-like function as:

$$\begin{aligned}
 & \text{Solution}(x) \equiv \text{IF } \text{IsBase}(x) \\
 & \quad \text{THEN } \text{SolveDirectly}(x) \\
 & \quad \text{ELSE } \text{Compose}(\text{Solution}(\text{Decompose}(x))).
 \end{aligned} \tag{1}$$

Where x is the original problem that will be solved through *Solution* procedure. *IsBase* is used to verify whether the problem x is primitive or not, which returns TRUE if x is a basic problem unit, or FALSE otherwise. *SolveDirectly* presents the conquer procedure. *Decompose* is referred to as

the decomposing operation, while *Compose* is referred to as the composing operation.

According to the previous analysis, we can employ this D&C based solution structure as a generic strategy of effort judgment for WSC-based SOA implementations. More precisely, the complete process can be further expressed in the following pseudo code.

TABLE I. ALGORITHM OF D&C BASED EFFORT JUDGMENT FOR WSC BASED SOA IMPLEMENTATION

```

1) //Treat the system as the highest-level service S to analyze.
2) double SoaEffortJudgment(service S) {
3)   double effort = 0;
4)   Determine the decomposability of S according to the design and
   real situations.
5)   if (S is decomposable) {
6)     Divide S into component services at lower level.
7)     foreach component service in S
8)       effort += SoaEffortJudgment(component service);
9)     effort += The effort of service composition for component
       services in S;
10)  }
11)  return effort;
12) }

```

As shown in Table I, the WSC based SOA system itself is treated as the highest-level coarse-grain service, which is also the initial input parameter of *SoaEffortJudgment* function. Within the body of *SoaEffortJudgment* function, unless the input service is already available, the effort of the service implementation will be recursively calculated by analyzing and judging the effort of the compositions of component Web services. Benefiting from this way of D&C, our focus on effort judgment for the whole WSC based SOA implementations can be narrowed down to that for individual WSCs.

III. QUALITATIVE ANALYSIS OF EFFORT OF WEB SERVICE COMPOSITION USING A CLASSIFICATION MATRIX

A. Classification Matrix of Web Service Compositions

Considering that many WSC approaches have been discussed in the literature, it is impossible to examine all of the related work and judge the effort of every WSC instance. However, we can inductively draw some classification of this work, and thereby facilitate the effort judgment for different type of WSC approaches. To overcome the deficiencies of existing classifications for WSC, we have introduced a novel classification matrix aimed at the influence on the effort of WSC [6]. This matrix uses clarified terminology, and differentiates the classifications between the *Context* and *Technology* dimensions. The *Context* dimension includes major effort related contexts that are *Pattern* (orchestration and choreography), *Semiotics* (semantics and syntax), *Mechanism* (SOAP and REST), *Design Time* (manual, semi-auto, and automatic) and *Runtime* (static and dynamic). The *Technology* dimension is divided into well defined *Workflow-based*, *Model-driven*, and *AI Planning* technology categories.

After further investigation, we decided to replace the *Technology* dimension with a *Process* dimension that can better reflect the characteristics of different technology

categories of WSC approaches. Furthermore, the *Process* dimension is divided into three process models, namely *One-Stop*, *Bridge* and *Double-Bridge*. If we refer to *Planning* as the manipulation at design time while *Execution* as the runtime activity in WSC approaches, the One-Stop process model can be as illustrated in Figure 1.

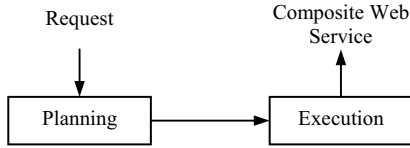


Figure 1. One-Stop Process Model of Web Service Composition.

In the One-Stop process, the *Planning* activity happens just after receiving the composition requirement, and delivers the executable composition specification directly. In most cases of One-Stop based approaches, during the planning stage the user must provide inputs at choice points, decide the interoperation among component Web services, and specify the composition procedure.

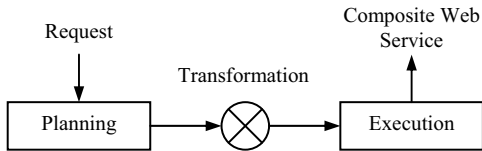


Figure 2. Bridge Process Model of Web Service Composition.

Unlike workflow-based approaches that usually employ the One-Stop process for WSC, most of the existing modeling techniques adopt the Bridge process when composing Web services, as shown in Figure 2. The Bridge process can be viewed as an evolution from the One-Stop process, which describes such approaches that plan WSCs at an abstract level, while the planning results cannot be directly executed and have to be transformed into executable specifications. Therefore, any WSC approach adopting the Bridge process uses a transformation procedure for the mapping between the planning result and executable specification.

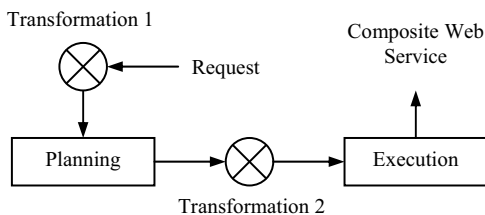


Figure 3. Double-Bridge Process Model of Web Service Composition.

Moreover, we can find that the WSC approaches using AI planning techniques normally contain Double-Bridge process, as illustrated in Figure 3. The Double-Bridge process can be treated as further evolution from the Bridge process. The *Planning* activity is settled between two transformation procedures in a Double-Bridge process. In detail, since AI planning systems generally adopt dedicated,

formal, and mathematical techniques, the initial information and composition requirement must be transformed for input into a planning system, and the planning result should be transformed again into an executable specification to build a composite Web service.

Considering the different influences of different contexts and processes on the composition effort, those process models and context types in the classification matrix can be viewed as effort factors when composing Web services. Therefore, we can use this classification matrix to facilitate the effort judgment for different WSC approaches. Since the data we collected here are all based on qualitative descriptions, it is not suitable to do quantitative work for composition effort judgment. Through analyzing these qualitative descriptions, however, we can further use a set of effort-related hypotheses to realize the effort comparison between different WSC approaches within the classification matrix.

B. Qualitative Effort Judgment Hypotheses

In the context of software engineering, effort required for a task is generally accounted by calculating how long and how many workers are needed to finish the task, and the unit can be person-day, person-month, or person-year. In brief, the amount of human activities in a project is proportional to the amount of effort required to finish the project. Therefore, for a certain software project, we can hypothesize:

H1. The increase of human activities in a project will have a proportional impact on the final effort.

Human activities include both physical and mental activities. Since software engineering is a knowledge-intensive domain, the effort of a software project is mainly composed of mental activities. Unfortunately, within a given time span people have limited mental capability to deal with information [7]. For every single person, the increased amount of information beyond a certain point may even defeat his/her mental ability, and hence result in errors [9]. As a result, the more information that exists in a project, the more people and human activities will be required to perform accurate manipulations. Together with H1, therefore, we can hypothesize:

H2. The increase of information in a project will have a proportional impact on the required human activities.

H2'. The increase of information in a project will have a proportional impact on the final effort.

Moreover, complexity has been proved to be a significant and non-negligible factor that influences software development and maintenance [10]. Meanwhile, the more complexity involved in a system, the more difficulty the designers or engineers have to understand the implementation process and thus the system itself [8], and hence the greater mental effort people have to exert to solve the complexity [7]. To summarize, we can further hypothesize:

H3. The increase of complexity in a project will have a proportional impact on the final effort.

When it comes to project complexity, one of the main contributors is the complexity of the methods with regards to achieving the project goals [11]. The methods generally consist of processes, tools, and techniques that are used to complete the corresponding project [12]. In particular, processes and techniques have been viewed as internal environment of a system (organization), while the system's complexity is considered a response to the environmental complexity [13]. Consequently, the complexity of processes and techniques involved in a software project will proportionally influence the complexity of the project. As for the tools, although the adoption of sophisticated tools usually implies a complex project, tools are essentially developed and used to save human activities. For a certain project, the more work the tools can fulfill, the less human activities the project will require. Overall, we can also hypothesize:

- H4. The increase of process complexity in a project will have a proportional impact on the project complexity.
- H4'. The increase of process complexity in a project will have a proportional impact on the final effort.
- H5. The increase of difficulty of techniques in a project will have a proportional impact on the project complexity.
- H5'. The increase of difficulty of techniques in a project will have a proportional impact on the final effort.
- H6. The increase of work that tools can fulfill in a project will have an inversely proportional impact on the human activities.
- H6'. The increase of work that tools can fulfill in a project will have an inversely proportional impact on the final effort.

C. Comparison between Partial Composition Effort Determined by Comparable Effort Factors

As mentioned earlier, we treat process models and context types in the classification matrix as effort factors of WSC approaches. After applying different effort judgment hypotheses to different but comparable factors, the partial composition effort determined by these factors can be qualitatively compared at first. To facilitate the effort comparison, some symbols and rules are also proposed:

For one certain task of WSC, we use E_{F-H} to represent the effort E determined by factor F when applying hypothesis H . Moreover, a score S will be set for E_{F-H} to flag different effort determined by different but comparable factors when applying some hypothesis. For convenience of calculation, the rules of score setting can be proposed as Equation (2). Note that if we use E_F to represent the effort E determined by factor F under all the different but applicable hypotheses, then all the scores for E_F under corresponding hypotheses can be summed up and represented as $S(E_F)$.

$$\begin{cases} S(E_{F1-H}) = 2, S(E_{F2-H}) = 1 & \text{if } E_{F1-H} > E_{F2-H} \\ S(E_{F1-H}) = 1, S(E_{F2-H}) = 1 & \text{if } E_{F1-H} \approx E_{F2-H} \\ S(E_{F1-H}) = 1, S(E_{F2-H}) = 2 & \text{if } E_{F1-H} < E_{F2-H} \end{cases} \quad (2)$$

We can hereby compare the partial composition effort following the sequence of building the classification matrix.

1) *For Orchestration and Choreography*: Orchestration stands for a central coordination, while choreography represents multiparty collaborations. Since distributed processing would be inevitably more complicated than non-distributed processing [1], for the same WSC project choreography requires more effort than orchestration if applying H3. Meanwhile, as the current de facto standard of orchestrating Web services, BPEL stemmed from existing languages and tools and has been widely accepted, whereas the choreography language WS-CDL was developed without any prior implementation and is still far from mature [14]. Considering this technical influence, the implementation of choreography will be more difficult than that of orchestration. By using *For* for representing the effort factor Orchestration and *Fch* for Choreography, the effort comparison and scores can be listed in Table II.

TABLE II. EFFORT COMPARISON BETWEEN ORCHESTRATION AND CHOREOGRAPHY

Applied Hypotheses	Compare	Scores
H3	$E_{For-H3} < E_{Fch-H3}$	$S(E_{For-H3})=1, S(E_{Fch-H3})=2$
H5'	$E_{For-H5'} < E_{Fch-H5'}$	$S(E_{For-H5'})=1, S(E_{Fch-H5'})=2$
Total	$E_{For} < E_{Fch}$	$S(E_{For})=2, S(E_{Fch})=4$

2) *For Syntactic and Semantic Compositions*:

TABLE III. EFFORT COMPARISON BETWEEN SYNTACTIC AND SEMANTIC COMPOSITION APPROACHES

Applied Hypotheses	Compare	Scores
H1	$E_{Fsy-H1} < E_{Fse-H1}$	$S(E_{Fsy-H1})=1, S(E_{Fse-H1})=2$
H5'	$E_{Fsy-H5'} \approx E_{Fse-H5'}$	$S(E_{Fsy-H5'})=1, S(E_{Fse-H5'})=1$
Total	$E_{Fsy} < E_{Fse}$	$S(E_{Fsy})=2, S(E_{Fse})=3$

Since semantic Web and semantic Web services are proposed to automate service discovery, selection, composition and execution by adding the inherent meanings, human activities within semantic compositions will be decreased while the involved information will be increased. Considering the increased information is for machine interpretation rather than human intervention, however, hypothesis H2 is not applicable here. Meanwhile, syntactic and semantic Web services share the unified Web infrastructure and both use markup language based techniques to describe information. It can then be stated that the difficulty levels of techniques adopted in both syntactic and semantic service compositions are similar. Therefore, by using *Fsy* for representing the effort factor Syntax and *Fse* for Semantics, the effort comparison and scores can be listed in Table III.

3) *For SOAP-based and RESTful Compositions*: Compared with RESTful Web service compositions, SOAP-based compositions employ more sophisticated techniques

including heavyweight protocols and a set of WS-* stack, which can satisfy more QoS requirements while also dealing with more information. Therefore, the hypotheses H2' and H5' are both applicable. By using *Fso* for representing the effort factor SOAP and *Fre* for REST, the effort comparison and scores can be listed in Table IV.

TABLE IV. EFFORT COMPARISON BETWEEN SOAP-BASED AND RESTFUL COMPOSITION APPROACHES

Applied Hypotheses	Compare	Scores
H2'	$E_{Fso-H2'} > E_{Fre-H2'}$	$S(E_{Fso-H2'})=2, S(E_{Fre-H2'})=1$
H5'	$E_{Fso-H5'} > E_{Fre-H5'}$	$S(E_{Fso-H5'})=2, S(E_{Fre-H5'})=1$
Total	$E_{Fso} > E_{Fre}$	$S(E_{Fso})=4, S(E_{Fre})=2$

4) *For Manual, Semi-Automatic, and Automatic Compositions*: During the design time of WSC, the more automated the design processes are, the less human activities the compositions will require, and the less detailed information developers should concern. Considering the realization of automation usually requires assistant tools and more techniques, for example the Semantic Matching approach [2], the hypohese H5' and H6' are both applicable together with H1 and H2'. By using *Fma* for representing the effort factor Manual, *Fsa* for Semi-Auto and *Fau* for Auto, the effort comparison and scores can be listed in Table V.

TABLE V. EFFORT COMPARISON BETWEEN MANUAL, SEMI-AUTOMATIC AND AUTOMATIC COMPOSITION APPROACHES

Applied Hypotheses	Compare	Scores
H1	$E_{Fma-H1} > E_{Fsa-H1}$ $E_{Fma-H1} > E_{Fau-H1}$ $E_{Fsa-H1} > E_{Fau-H1}$	$S(E_{Fma-H1})=2+2=4$ $S(E_{Fsa-H1})=1+2=3$ $S(E_{Fau-H1})=1+1=2$
H2'	$E_{Fma-H2'} > E_{Fsa-H2'}$ $E_{Fma-H2'} > E_{Fau-H2'}$ $E_{Fsa-H2'} > E_{Fau-H2'}$	$S(E_{Fma-H2'})=2+2=4$ $S(E_{Fsa-H2'})=1+2=3$ $S(E_{Fau-H2'})=1+1=2$
H5'	$E_{Fma-H5'} < E_{Fsa-H5'}$ $E_{Fma-H5'} < E_{Fau-H5'}$ $E_{Fsa-H5'} < E_{Fau-H5'}$	$S(E_{Fma-H5'})=1+1=2$ $S(E_{Fsa-H5'})=2+1=3$ $S(E_{Fau-H5'})=2+2=4$
H6'	$E_{Fma-H6'} > E_{Fsa-H6'}$ $E_{Fma-H6'} > E_{Fau-H6'}$ $E_{Fsa-H6'} > E_{Fau-H6'}$	$S(E_{Fma-H6'})=2+2=4$ $S(E_{Fsa-H6'})=1+2=3$ $S(E_{Fau-H6'})=1+1=2$
Total	$E_{Fma} > E_{Fsa} > E_{Fau}$	$S(E_{Fma})=14, S(E_{Fsa})=12,$ $S(E_{Fau})=10$

5) *For Static and Dynamic Compositions*: If we emphasize the adaptation in both static and dynamic compositions during runtime, we can draw the same conclusions through the similar analysis as above. Note although dynamic composition intuitively involves more complex concerns than static one does, in practice, dynamic composition is generally supported by existing engines or tools while static composition has to realize adaptation by adjusting Web services manually. Since the complexity of manual adaptation depends on the unpredictable situation in the future, the applicability of H3 cannot be guaranteed here.

Therefore, by using *Fst* for representing the effort factor Static and *Fdy* for Dynamic, the effort comparison and scores can be listed in Table VI.

TABLE VI. EFFORT COMPARISON BETWEEN STATIC AND DYNAMIC COMPOSITION APPROACHES

Applied Hypotheses	Compare	Scores
H1	$E_{Fst-H1} > E_{Fdy-H1}$	$S(E_{Fst-H1})=2, S(E_{Fdy-H1})=1$
H2'	$E_{Fst-H2'} > E_{Fdy-H2'}$	$S(E_{Fst-H2'})=2, S(E_{Fdy-H2'})=1$
H5'	$E_{Fst-H5'} < E_{Fdy-H5'}$	$S(E_{Fst-H5'})=1, S(E_{Fdy-H5'})=2$
H6'	$E_{Fst-H6'} > E_{Fdy-H6'}$	$S(E_{Fst-H6'})=2, S(E_{Fdy-H6'})=1$
Total	$E_{Fst} > E_{Fdy}$	$S(E_{Fst})=7, S(E_{Fdy})=5$

6) *For One-Stop, Bridge and Double-Bridge Process Models*:

TABLE VII. EFFORT COMPARISON BETWEEN ONE-STOP, BRIDGE AND DOUBLE-BRIDGE PROCESS BASED COMPOSITION APPROACHES

Applied Hypotheses	Compare	Scores
H1	$E_{Fos-H1} > E_{Fbr-H1}$ $E_{Fos-H1} > E_{Fdb-H1}$ $E_{Fbr-H1} > E_{Fdb-H1}$	$S(E_{Fos-H1})=2+2=4$ $S(E_{Fbr-H1})=1+2=3$ $S(E_{Fdb-H1})=1+1=2$
H2'	$E_{Fos-H2'} > E_{Fbr-H2'}$ $E_{Fos-H2'} > E_{Fdb-H2'}$ $E_{Fbr-H2'} > E_{Fdb-H2'}$	$S(E_{Fos-H2'})=2+2=4$ $S(E_{Fbr-H2'})=1+2=3$ $S(E_{Fdb-H2'})=1+1=2$
H4'	$E_{Fos-H4'} < E_{Fbr-H4'}$ $E_{Fos-H4'} < E_{Fdb-H4'}$ $E_{Fbr-H4'} < E_{Fdb-H4'}$	$S(E_{Fos-H4'})=1+1=2$ $S(E_{Fbr-H4'})=2+1=3$ $S(E_{Fdb-H4'})=2+2=4$
H5'	$E_{Fos-H5'} \approx E_{Fbr-H5'}$ $E_{Fos-H5'} \approx E_{Fdb-H5'}$ $E_{Fbr-H5'} \approx E_{Fdb-H5'}$	$S(E_{Fos-H5'})=1+1=2$ $S(E_{Fbr-H5'})=1+1=2$ $S(E_{Fdb-H5'})=1+1=2$
H6'	$E_{Fos-H6'} > E_{Fbr-H6'}$ $E_{Fos-H6'} > E_{Fdb-H6'}$ $E_{Fbr-H6'} > E_{Fdb-H6'}$	$S(E_{Fos-H6'})=2+2=4$ $S(E_{Fbr-H6'})=1+2=3$ $S(E_{Fdb-H6'})=1+1=2$
Total	$E_{Fos} > E_{Fbr} > E_{Fdb}$	$S(E_{Fos})=16, S(E_{Fbr})=14,$ $S(E_{Fdb})=12$

Considering that One-Stop process delivers executable specificaitons, Bridge process focuses on the abstract modeling, and Double-Bridge process focuses on the composition requirement, approaches adopting One-Stop process have to deal with the most information while Double-Bridge process based approaches deal with the least information for one certain task of WSC. Meanwhile, Double-Bridge approaches have the longest processes while One-Stop approaches have the shortest. However, we can imagine that both One-Stop and Bridge processes also contain two transformation procedures as well as Double-Bridge process does. The intangible transformation procedures essentially take place as mental activities, while the tangible ones can be supported by tools. Therefore, it can be found that Double-Bridge approaches require less human activities and use more tools, One-Stop approaches require more human activities and use less tools, while Bridge approaches are in the middle. When it comes to techniques, it is nearly impossible to compare the difficulty levels behind the process models with each other. Consequently, here we

simply treat their difficulties similarly. After applying all the suitable hypotheses and using Fos for representing the effort factor One-Stop model, Fbr for Bridge model and Fdb for Double-Bridge model, the effort comparison and scores can be listed in Table VII.

D. Qualitative Effort Comparison between Different Web Service Composition Approaches

To realize the effort comparison between different WSC approaches, we should find a way to reflect the combined influences of different factors on the composition effort. Considering different types of contexts can be superposed together to influence WSC, we define that the scores for different context types are accumulable in the *Context* dimension. Meanwhile, considering the *Context* and *Process* dimensions are independent of each other, we can further define that the effort scores are multipliable across different dimensions. As such, the effort score of a WSC approach can be expressed through the sum and product of the scores of the relevant effort factors. According to the combination of different effort factors identified in the classification matrix, there are totally 144 different types of WSC approaches ($3 \times 2 \times 2 \times 3 \times 2 = 144$). Therefore, we can fill the applicable hypotheses and scores to the classification matrix to achieve an effort checklist of 144 types of WSC approaches, as demonstrated in Appendix I. Benefiting from this checklist, we can conveniently judge the qualitative effort between different approaches for a certain WSC instance: one composition approach requires more effort than another does if the former's effort score is bigger than the latter's.

However, it should be noted that the scores do NOT indicate any count of the amount of effort. Instead, these quantitative numbers are only used to facilitate qualitatively judging the effort of different composition approaches before implementing a WSC. Meanwhile, some other meaning of the effort scores can be further revealed. By investigating the result and procedure of the calculation of effort scores, as shown in Figure 4, we can find that the amount of applicable hypotheses implies the times of comparisons, while the times of consistent comparisons is proportional to the difference between resulting scores of effort factors.

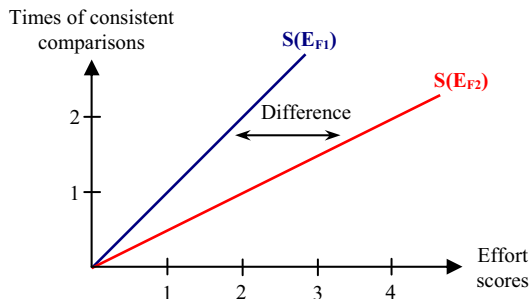


Figure 4. A Sample of Consistent Comparisons for $E_{F2} > E_{F1}$.

Here we define different comparisons are *consistent* when the same conclusion can be drawn in these comparisons by applying different hypotheses. For example, there are two consistent comparisons when applying hypotheses $H3$ and $H5'$ to the compare between

Orchestration and Choreography in Table II. Since the consistent comparisons can help to confirm and reinforce the comparison result, the difference between scores of effort factors reflects the extent of our confidence in the effort judgment. When it comes to composition approaches, the difference between total effort scores is further magnified through the summation and multiplication by scores of effort factors. Therefore, to summarize, the larger difference that exists between two approach effort scores, the more confidence we will have in the judgment result.

IV. A CASE STUDY OF EFFORT JUDGMENT BEFORE IMPLEMENTATION

After analyzing the effort of different types of WSC approaches based on a classification matrix, we can now tackle the problem of effort judgment for WSC based SOA implementations. Here we employ the RailCo Ltd. case study presented in [15] to demonstrate the effort judgment approach proposed in this paper. RailCo Ltd. is a railway parts supplier company specializing in air brakes and related installation tools. To improve the working efficiency of this company, a service-oriented analysis was conducted, which decomposed the business process logic into two business services that are composed of four application services. Therefore, the redesigned automation system can be viewed as a typical WSC based SOA implementation, as illustrated in Figure 5.

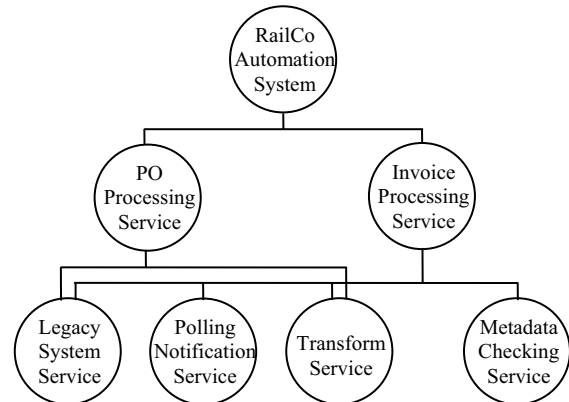


Figure 5. Redesigned Automation System of RailCo Ltd.

By applying the algorithm listed in Table I and using the aforementioned effort checklist, we can conveniently calculate the complete effort score for the WSC based SOA implementation of RailCo Ltd. Considering that Web service uses an abstract interface to conceal the technical details like the implementation language, deployment platform and underlying communication protocol, in practice, different composite Web services in one SOA project can be implemented with different techniques under different contexts by different people. Consequently, we use arbitrary combinations of WSC approaches to represent different implementations of this WSC based SOA system.

Suppose there are three candidate implementation proposals listed in Table VIII, through the implementation effort scores, we can find that the second one requires the

least effort among three proposals. Moreover, compared with the proposal 3, we have more confidence to say that the proposal 1 requires more effort than the proposal 2 does.

TABLE VIII. EFFORT COMPARISON BETWEEN IMPLEMENTATION PROPOSALS FOR THE AUTOMATION SYSTEM OF RAILCO LTD.

Proposal	Implementation Effort Scores
Proposal 1	$S(\text{WSC approach 1}) + S(\text{WSC approach 1}) + S(\text{WSC approach 1}) = 464 + 464 + 464 = 1392$
Proposal 2	$S(\text{WSC approach 7}) + S(\text{WSC approach 7}) + S(\text{WSC approach 7}) = 312 + 312 + 312 = 936$
Proposal 3	$S(\text{WSC approach 2}) + S(\text{WSC approach 5}) + S(\text{WSC approach 9}) = 432 + 406 + 312 = 1150$

V. CONCLUSION

Due to the number of various WSC approaches developed in the literature, it is impossible to collect development data through experiments to do quantitative effort estimation for WSC based SOA implementations. Therefore, we propose a qualitative approach to judge the effort of different proposals before implementing a WSC based SOA project. The contribution of this paper is mainly twofold. First, we employ a generic strategy of effort judgment for WSC based SOA implementations. The strategy can be realized by different approaches in addition to ours. Second, we develop a method of assigning scores to effort factors of WSC, which can be used to facilitate the qualitative effort judgment between different effort factors, between different type of WSC approaches and even between different SOA implementation proposals.

REFERENCES

- [1] N. M. Josuttis, *SOA in Practice: The Art of Distributed System Design*, Sebastopol: O'Reilly Media, Inc., 2007.
- [2] M. S. Fox, "An Organizational View of Distributed Systems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-11, no. 1, Jan. 1981, pp. 70–80.
- [3] Z. Li, L. O'Brien, H. Zhang, and J. Keung, "Towards SOA Implementation Complexity Measurement Enlightened by Organization Theory," Volume 2 of Proc. 11th International Conference on Product Focused Software Development and Process Improvement (PROFES 2010), ACM Press, June 2010, pp. 16–19.
- [4] D. Campbell and T. Craig, *Organisations and the Business Environment*, 2nd ed., Burlington, MA: Butter-worth-Heinemann, 2005.
- [5] O. Yadgar, S. Kraus, and C. Ortiz, "Scaling up Distributed Sensor Networks: Cooperative Large-Scale Mobile-Agent Organizations," in *Distributed Sensor Networks: A Multiagent Perspective*. Boston Hardbound: Kluwer Academic Publishers, May 2003, pp. 185–218.
- [6] Z. Li, L. O'Brien, J. Keung, and X. Xu, "Effort-Oriented Classification Matrix of Web Service Composition," Proc. the Fifth International Conference on Internet and Web Applications and Services (ICIW 2010), IEEE Press, June 2010, pp. 357-362.
- [7] T. Globerson, "Mental Capacity, Mental Effort, and Cognitive Style," *Developmental Review*, vol. 3, no. 3, Sept. 1983, pp. 292-302.
- [8] J. Cardoso, "How to Measure the Control-Flow Complexity of Web Processes and Workflows," *Workflow Handbook 2005*, Lighthouse Point: Layna Fischer, Apr. 2005, pp. 199-212.
- [9] G. A. Miller, "The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information," *Psychological Review*, vol. 63, no. 2, Mar. 1956, pp. 81-97.
- [10] C. Francalanci and F. Merlo, "The Impact of Complexity on Software Design Quality and Costs: An Exploratory Empirical Analysis of Open Source Applications," Proc. 16th European Conference on Information Systems (ECIS 2008), June 2008, pp. 1442-1453, Galway, Ireland.
- [11] J. R. Turner and R. A. Cochrane, "Goals-and-Methods Matrix: Coping with Projects with Ill-defined Goals and/or Methods of Achieving them," *International Journal of Project Management*, vol. 11, no. 2, May 1993, pp. 93-102.
- [12] A. Camci and T. Kotnour, "Technology Complexity in Projects: Does Classical Project Management Work?," Proc. Technology Management for the Global Future (PICMET 2006), IEEE Press, vol. 5, July 2006, pp. 2181-2186.
- [13] K. Dooley, "Organizational Complexity," *International Encyclopedia of Business and Management*, M. Warner (ed.), London: Thompson Learning, Oct. 2001, pp. 5013-5022.
- [14] A. Barros, M. Dumas, and P. Oaks, "Standards for Web Service Choreography and Orchestration: Status and Perspectives," Proc. Business Process Management Workshops, Sept. 2005, pp. 61-74.
- [15] T. Erl, *Service-Oriented Architecture: Concepts, Technology, and Design*. Crawfordsville: Prentice Hall PTR, Aug. 2005.
- [16] F. Casati, S. Ilnicki, L. Jin, V. Krishnamoorthy, and M. Shan, "Adaptive and Dynamic Service Composition in EFlow," Proc. 12th International Conference on Advanced Information Systems Engineering (CaiSE*00), Springer, Jun. 2000, pp. 13-31.
- [17] J. Latham, K. Gomadam, and A. P. Sheth, "SA-REST and (S)mashups : Adding Semantics to RESTful Services," Proc. First IEEE International Conference on Semantic Computing (ICSC 2007), IEEE Press, Sept. 2007, pp. 469-476.
- [18] D. Skogan, R. Groenmo, and I. Solheim, "Web Service Composition in UML," Proc. Eighth IEEE International Enterprise Distributed Object Computing Conference (EDOC 2004), IEEE Press, Sept. 2004, pp. 47-57.
- [19] J. T. E. Timm and G. C. Gannod, "Specifying Semantic Web Service Compositions using UML and OCL," Proc. 2007 IEEE International Conference on Web Services (ICWS 2007), IEEE Press, Jul. 2007, pp. 521-528.
- [20] R. Anzboeck and S. Dustdar, "Semi-Automatic Generation of Web Services and BPEL Processes - A Model-Driven Approach," *Lecture Notes in Computer Science*, vol. 3649/2005, Sept. 2005, pp. 64-79.
- [21] E. Sirin, B. Parsia, D. Wu, J. Hendler, and D. Nau, "HTN Planning for Web Service Composition using SHOP2," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 1, Oct. 2004, pp. 377-396.
- [22] V. Valero, M. E. Cambronero, G. Díaz, and H. Macià, "A Petri Net Approach for the Design and Analysis of Web Services Choreographies," *Journal of Logic and Algebraic Programming*, vol. 78, May-Jun. 2009, pp. 359-380.
- [23] S. G. H. Tabatabaei, W. M. N. Kadir, and S. Ibrahim, "Semantic Web Service Discovery and Composition Based on AI Planning and Web Service Modeling Ontology," Proc. IEEE Asia-Pacific Services Computing Conference (APSCC '08), IEEE Press, Dec. 2008, pp. 397-403.
- [24] H. Zhao and P. Doshi, "Towards Automated RESTful Web Service Composition," Proc. IEEE International Conference on Web Services (ICWS 2009), IEEE Press, Jul. 2009, pp. 189-196.
- [25] L. O'Brien, "A Framework for Scope, Cost and Effort Estimation for Service Oriented Architecture (SOA) Projects," Proc. 20th Australian Software Engineering Conference (ASWEC'09), IEEE Press, Apr. 2009, pp. 101-110.
- [26] H. Demirkan, R. J. Kauffman, J. A. Vayghan, H. G. Fill, D. Karagiannis, and P. P. Maglio, "Service-Oriented Technology and Management: Perspectives on Research and Practice for the Coming Decade," *Electronic Commerce Research and Applications*, vol. 7, no. 4, 2008, pp.356-376.
- [27] P. Sarang, F. Jennings, M. Juric, and R. Loganathan, *SOA Approach to Integration: XML, Web services, ESB, and BPEL in Real-World SOA projects*, Birmingham: Packt Publishing, 2007.

APPENDIX I: A SAMPLE OF EFFORT CHECKLIST FOR WEB SERVICE COMPOSITION APPROACHES

Process	Category		Context											
			Pattern		Semiotics		Mechanism		Design Time			Runtime		
			Orchestration	Choreography	Syntax	Semantics	SOAP	REST	Manual	Semi-Auto	Auto	Static	Dynamic	
	<i>Applied Hypotheses</i>		H3, H5'	H3, H5'	H1, H5'	H1, H5'	H2', H5'	H2', H5'	H1, H2', H5', H6'	H1, H2', H5', H6'	H1, H2', H5', H6'	H1, H2', H5', H6'	H1, H2', H5', H6'	
	<i>Score</i>		S(E _{For})=2	S(E _{Fch})=4	S(E _{Fsy})=2	S(E _{Fse})=3	S(E _{Fso})=4	S(E _{Frc})=2	S(E _{Fma})=14	S(E _{Fsa})=12	S(E _{Fau})=10	S(E _{Fst})=7	S(E _{Fdy})=5	
<i>One-Stop</i>	H1, H2', H4', H5', H6'	S(E _{Fos})=16	WSC Approach 1 (e.g. BPEL programming)	√		√		√		√		√		
			S(WSC Approach 1) = S(E _{Fos}) × (S(E _{For}) + S(E _{Fsy}) + S(E _{Fso}) + S(E _{Fma}) + S(E _{Fst})) = 16 × (2 + 2 + 4 + 14 + 7) = 16 × 29 = 464											
			WSC Approach 2 (e.g. eFlow [16])	√		√		√		√				√
			S(WSC Approach 2) = S(E _{Fos}) × (S(E _{For}) + S(E _{Fsy}) + S(E _{Fso}) + S(E _{Fma}) + S(E _{Fdy})) = 16 × (2 + 2 + 4 + 14 + 5) = 16 × 27 = 432											
<i>Bridge</i>	H1, H2', H4', H5', H6'	S(E _{Fbr})=14	WSC Approach 3 (e.g. SA-REST + Smashup [17])	√			√		√		√		√	
			S(WSC Approach 3) = S(E _{Fos}) × (S(E _{For}) + S(E _{Fse}) + S(E _{Frc}) + S(E _{Fsa}) + S(E _{Fst})) = 16 × (2 + 3 + 2 + 12 + 7) = 16 × 26 = 416											
			WSC Approach 4 (e.g. UML + MDA [18])	√		√		√		√			√	
			S(WSC Approach 4) = S(E _{Fbr}) × (S(E _{For}) + S(E _{Fsy}) + S(E _{Fso}) + S(E _{Fma}) + S(E _{Fst})) = 14 × (2 + 2 + 4 + 14 + 7) = 14 × 29 = 406											
<i>Double-Bridge</i>	H1, H2', H4', H5', H6'	S(E _{Fdb})=12	WSC Approach 5 (e.g. UML + OCL [19])	√			√	√		√			√	
			S(WSC Approach 5) = S(E _{Fbr}) × (S(E _{For}) + S(E _{Fse}) + S(E _{Fso}) + S(E _{Fma}) + S(E _{Fdy})) = 14 × (2 + 3 + 4 + 14 + 5) = 14 × 28 = 392											
			WSC Approach 6 (e.g. UML + IHE framework [20])	√		√		√		√				√
			S(WSC Approach 6) = S(E _{Fbr}) × (S(E _{For}) + S(E _{Fsy}) + S(E _{Fso}) + S(E _{Fma}) + S(E _{Fdy})) = 14 × (2 + 2 + 4 + 14 + 5) = 14 × 27 = 378											
<i>Double-Bridge</i>	H1, H2', H4', H5', H6'	S(E _{Fdb})=12	WSC Approach 7 (e.g. SHOP2 [21])	√			√	√				√	√	
			S(WSC Approach 7) = S(E _{Fdb}) × (S(E _{For}) + S(E _{Fse}) + S(E _{Fso}) + S(E _{Fau}) + S(E _{Fst})) = 12 × (2 + 3 + 4 + 10 + 7) = 12 × 26 = 312											
			WSC Approach 8 (e.g. Petri Net [22])		√		√	√				√	√	
			S(WSC Approach 8) = S(E _{Fdb}) × (S(E _{Fch}) + S(E _{Fse}) + S(E _{Fso}) + S(E _{Fau}) + S(E _{Fst})) = 12 × (4 + 3 + 4 + 10 + 7) = 12 × 28 = 336											
<i>Double-Bridge</i>	H1, H2', H4', H5', H6'	S(E _{Fdb})=12	WSC Approach 9 (e.g. AIMO [23])		√		√	√				√	√	
			S(WSC Approach 9) = S(E _{Fdb}) × (S(E _{Fch}) + S(E _{Fse}) + S(E _{Fso}) + S(E _{Fau}) + S(E _{Fdy})) = 12 × (4 + 3 + 4 + 10 + 5) = 12 × 26 = 312											
			WSC Approach 10 (e.g. Situation Calculus for REST [24])	√			√		√		√		√	
			S(WSC Approach 10) = S(E _{Fdb}) × (S(E _{For}) + S(E _{Fse}) + S(E _{Frc}) + S(E _{Fsa}) + S(E _{Fst})) = 12 × (2 + 3 + 2 + 12 + 7) = 12 × 26 = 312											