Plan Analysis for Enabling Service Oriented Computing

Biplav Srivastava

IBM India Research Laboratory
Block 1, IIT Campus,
Hauz Khas, New Delhi, India 110016
sbiplav@in.ibm.com

Abstract

When Services Oriented Computing (SOC) approach is applied in enterprise application domain, the business processes are first modeled and then selected processes are implemented as Services Oriented Architecture (SOA) applications, e.g., web services. Planning techniques have been widely considered suitable for web services, e.g., for composition. Traditionally, the area of planning has looked at plans in the context of how they could be generated from the domain theory and problem specification. While synthesizing plans is useful and important, it puts heavy demands on how the information is modeled and available in the domain. In SOC, the domain is too complex to build explicit models for automated planning. Plan analysis and management looks at issues related to how plans, defined as a set of coordinated activities, are used after they are obtained and how they play a role in subsequent planning, without assuming planning to be automated or manual. We show that plan analysis can be the enabler of automated reasoning in Services Oriented Computing by helping in identifying business process improvement possibilities and then in identifying how web service implementations can be effectively created or assembled.

Introduction

[1]When Services Oriented Computing is applied in an enterprise domain, the business processes representing business functions of interest are first modeled and then a subset of the processes is selected for automation using SOA, e.g., web services. To understand how users work with business processes and web services, and identify the requirements of reasoning in SOC, we introduce the abstraction of ***component***. A component refers to the basic granularity of system building blocks. It can be a physical or conceptual artifact that is aggregated to build systems of interest. As representation of real world activities, business processes are components. As encapsulation of IT system implementations, web services are components too but they are additionally services. A *service* is a component that follows the publish-find-bind pattern to discover and invoke other components.

The abstraction of component is now useful in identifying different dimensions of how users may work with business processes and web services. The dimensions are - the components have to be represented, some of them have to be discovered and aggregated (composed) while building new components, there may be gaps in what capabilities are available and what are desired -- they need to be analyzed and filled, and finally, components' life cycle has to be managed.

Planning techniques, as a means to synthesize action sequences, has been widely considered suitable for web services composition. But we believe that this is an incomplete view of its full potential in SOC. Planning has primarily looked at plans in the context of how they could be generated from the domain theory and problem specification. While synthesizing plans is useful and important, it puts heavy demand on how the information is modeled and available in an application. Even without synthesis, plans may be acquired from domain meta-models or experts, or learnt, and later be customized to new situations using planning techniques. This is especially true for business process models but also valid for web service compositions.

More generally, there is a growing realization that acquiring plans (whether automated or manual) is just one part of the complex process of how a plan is used in any real world application. Users want to generate plans only when needed and they want to know how they are using their plans. Once available, they may be organized in large collections, where they can be grouped along different purposes and are made amenable to the search, inspection, evaluation, and modification by human experts or automated reasoning systems. Eventually, plans will outlast their utility and be replaced by newer, better ones. Plan analysis and management looks at issues of acquiring plans (e.g., eliciting plans, learning them), synthesizing plans when the domain is incomplete, modeling updates to plans (e.g., through instructions), techniques to manage plans, metadata generation, storage of plans and how to retire them. See [6] for details on some of these methods.

Plan analysis can be the enabler of automated reasoning in Services Oriented Computing by helping in identifying business process improvement possibilities and then in identifying how web service implementations can be effectively created or assembled. We do not restrict the plan representation to only Planning Domain Description Language (PDDL) that is used in the planning community. We consider plans to be also workflows like BPEL4WS to stress the fact that beyond the PDDL plan representation, planning should also consider the generalized workflows representation which is not always automatically generated.

The rest of the paper is organized as follows. We start by giving an example of expense processing in the enterprise business domain and see how an SOC approach may be applied to implement an IT system for it. We see that plan analysis techniques can help in identifying bottlenecks and transformation opportunities in the business process model. Then, the web services implementations of a subset of the processes that are selected for automation can also be facilitated by techniques to assess and leverage existing web services. We then discuss the general notion of working with components and how plan analysis may be relevant to it to enable SOC.

Example: Plan Analysis in Business Processes

Consider the process of how an employee in a fictitious company undertakes business expense using the corporate American Express (Amex) charge card and is reimbursed by the employer. The employee determines a business need (step 1), consults his manager (step 2) and makes the expense (step 3) which results in his Amex card getting charged (step 5). Later, he does expensing (step 4) which needs approval by his manager (step 6), finance (step 7) and then payment is released (step 8) to either the employee (step 9) or Amex (step 10). The expense is closed with the release of the payment (step 11). Amex needs payment (step 12) within 3 weeks of charging (Step 5).

Given such a process, suppose there are a lot of complaints with the process. How can one identify the problems with this process and recommend transformations? Is the model good enough or more information is needed? How can the IT systems (expensing and approval tools here) be modified to remove the problems.

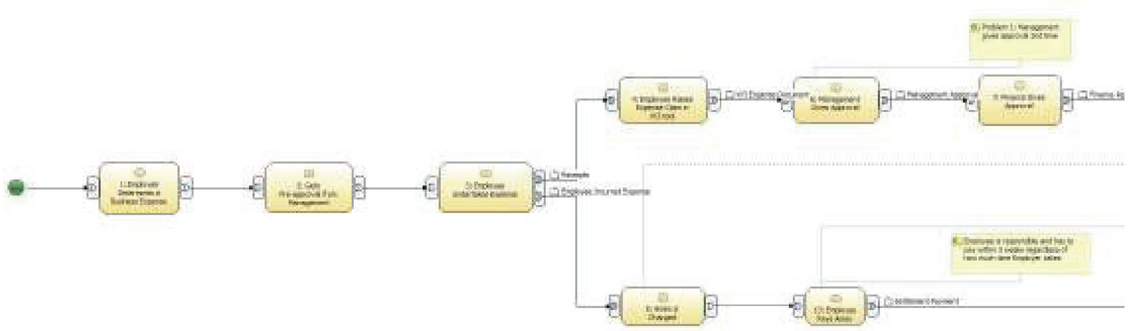
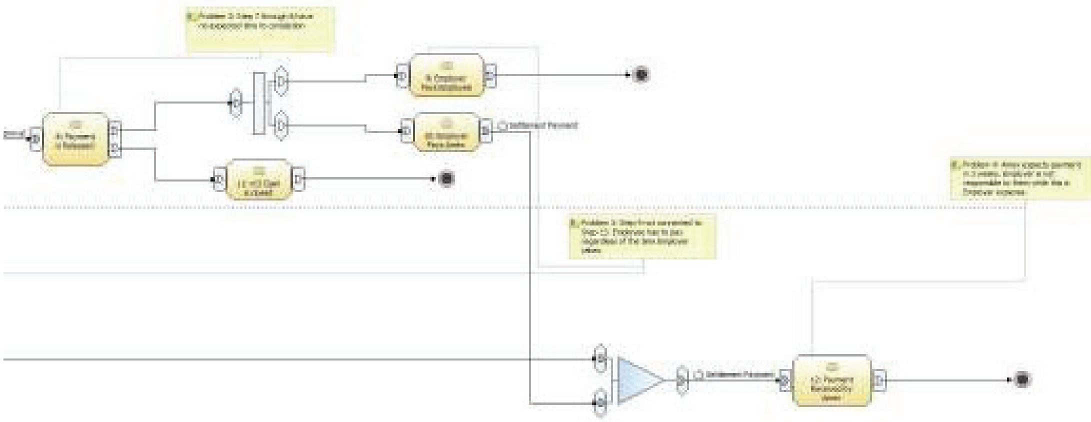


Figure 1: The Business Process of Expensing at a fictitious company. Plan analysis can help in identifying problems in this example of componentization of IT services.



Consultants today observe processes and recommend modifications. At least four problems may be identified in the example based on experience with transforming business processes. They are:

Problem 1: Management approval is needed twice in the process.

Problem 2: Step 7 through 8 (Finance approval and payment release) need to have an expected time to completion.

Problem 3: Step 9 (Employer pays Employee) is not connected to Step 13 (Employee pays Amex). Employee has to pay Amex regardless of the time Employer takes.

Problem 4: Amex expects payment in 3 weeks but the employer is never monitoring this deadline. The employer is effectively passing business (employer) expense responsibility onto the employee.

Plan analysis can help in making identification of problems a science from an art that it is today. The specific techniques that are needed are how one can assess the information requirements of a step, the complexity of the activity, the skill of the actor, and the relationship between the analysis of a step and that of the whole process.

Example: Plan Analysis in Web Service Implementation of Selected Processes

Suppose the decision makers of the company saw the problems and decided to address Problem 3 first as they considered it to be the most critical of them all. Suppose the solution that is required is an Expense Escalation tool that will escalate and eventually process the expense with sufficient time to spare. Now how will it be built as a composite web service? How can the employer's current IT Expensing and Approval tools be integrated along with the Amex's billing system? If the solution will be deployed at a world-wide scale, what specific constituent web services be used in different geographies for best performance?

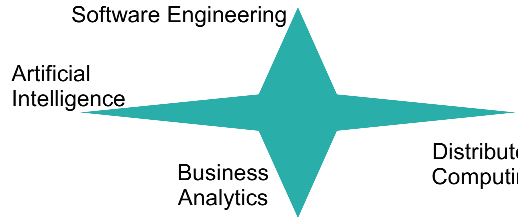
Plan analysis can help in identifying issues like what existing web services are relevant, what new functionality is needed, how the solution would behave in changing external environment – say as the system is applied to a growing region of the company, and how much will the solution cost. Only some of this can be handled by current automated planning techniques after the domain has been fully modeled.

Component Engineering for SOC and Plan Analysis

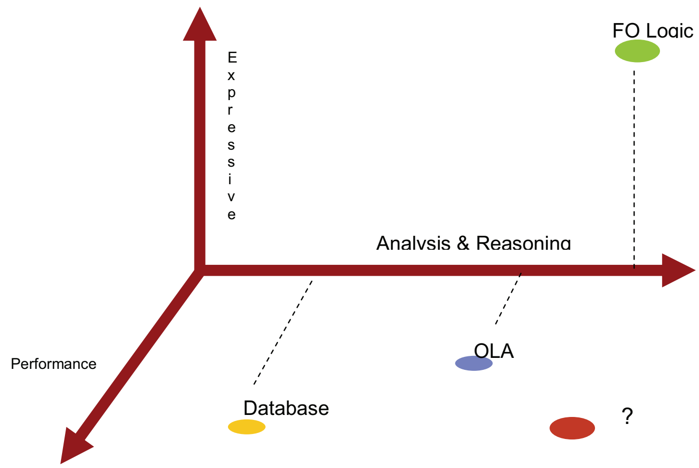
In Figure 2, we consider the tradeoffs involved in defining and working with components. This is a multi-disciplinary area at the intersection of Software Engineering, Artificial Intelligence, Distributed Computing and Business Process Analytics. Ideally, one would want to work with expressively represented components, reason with them in depth and work at very high performance transactional rates. However, as has

Figure 2: Components engineering is multi-disciplinary and requires trade-off between component expressivity, level of reasoning supported and performance needed.

Represent component as detailed as needed *for relevant reasoning of that component (e.g., web services, business process)* and no more detailed



Component Size/ Axis	Web Services	Business Processes
Expressivity	Medium	High
Analysis & Reasoning	Medium	Medium
Performance	High	Low



been found in other Computer Science fields, a tradeoff has to be made among them based on the usage scenario for tractability. The figure also shows the usage scenarios for web services and business processes on the left in terms of these axes.

The traditional operational model for building systems is to:

a) Capture Requirements
b) Develop solution from scratch
c) Deploy solution

If previously built components have to be used effectively in building new components, this model has to change to:

a) Capture Requirements
b) Find components for reuse
c) Develop fragments of missing functionality
d) Deploy solution

The step (b) becomes crucial because it tries to (1) discover matches to existing component that can be directly used, or (2) composed from existing components, or (3) perform gap analysis so that existing components are used in such a manner that the efforts to develop missing functionality is minimized. Hence (b) tries to either make step (c) unnecessary or minimal in the context of existing components.

We now consider componentization through web services (SOA) and business processes and discuss some efforts to address the new operational model. We categorize efforts along 4 dimensions in table below: how components are represented and new requirements will be specified (corresponding to (a) in the operation model, how components are discovered and aggregated/ composed while building new components (corresponding to (b)), how gaps are analyzed and filled (corresponding to (c)), and finally, how components' life cycle is managed (corresponding to (d)).

Component Size/ Operation Phase	**Web Services**	**Business Processes**
Representation	Domain (OWL), Functional (OWL-S), Non- functional (Policy, SLAs)[5], Web services (BPEL, WSDL), QoS engineering	Business Process Modeling of key business functions, roles and data flow through Visio, WBI Modeler
Find - Discovery and Composition	Semantic matching[4], AI planning, Non Functional Requirement optimization[2]	Identify sub-processes, variations, plan templates
Find - Analysis of Gaps	Gaps with software components[1]	Transformation choices
Life Cycle Management	Automatic Workflow Metadata Generation[3]	

The table helps us understand where more research efforts have happened than other areas. For example, we see that there is much to be done in gap analysis for both web services and business processes.

An important thing to note is the connection between components as we have introduced and the general design principles of Object Oriented Analysis and Design (OO). In OO, the classes, objects and interfaces of the system are modeled along with inheritance and polymorphic behavior. In web services, the distinction between service classes and service instances was not originally made. We recently introduced it as part of our work with web services composition [2]. It is not clear if dynamic OO feature like inheritance and polymorphism are crucial in open computing environment of SOA even in the presence of a security and trust model. For business processes, since they are not executable, event the semantics of OO features is not clear. But componentization is a reasonable aim for web services and business processes to make SOC a commercial reality.

SOC is enabled with the vision of component engineering to support the new operation model with technologies that makes the full matrix production ready. And plan analysis from AI, we believe, will contribute deeply to it.

Conclusion

In the paper, we explored how planning techniques can enable SOC in enterprise application. Conventionally, planning techniques meant only automated planning and they were seen applicable to only web service implementations. We introduced the abstraction of components to cover business processes and web services, and illustrated that plan analysis and management techniques can be very relevant in working with them (components).

References

1. B. Srivastava, A Decision-support Framework for Component Reuse and Maintenance in Software Project Management, In IEEE 8th European Conference on Software Maintenance and Reengineering (CSMR 2004), Tampere, Finland, March 24-26, 2004.
2. V. Agarwal, K. Dasgupta, N. Karnik, A. Kumar, A. Kundu, S. Mittal, B. Srivastava, A Service Creation Environment based on End to End Composition of Web Services, Proceedings of the 14th. WWW Conference (WWW 2005), Japan
3. B. Srivastava, J. Vanhatalo and J. Koehler, Managing the Life Cycle of Plans, Proceedings of the 17th Annual Conference on Innovative Applications of Artificial Intelligence (IAAI-05), Pittsburgh, USA
4. R. Akkiraju, B. Srivastava, A. Ivan, R. Goodwin, T. Mahmood, SEMAPLAN: Combining Planning with Semantic Matching to Achieve Web Service Composition, In Proc. International Conference on Automated Planning & Scheduling (ICAPS 2006), The Lake District, UK, 2006
5. A. Kumar, B. Srivastava and S. Mittal, Information Modeling for End to End Composition of Web Services, Proceedings of the 4th International Semantic Web Conference (ISWC 2005), Galway, Ireland.
6. ICAPS 2006 Workshop on Plan Analysis and Management, The Lake District, U.K.