Service Identification: BPM and SOA Handshake

Srikanth Inaganti & Gopala Krishna Behara

Abstract

Service identification is one of the first steps in the Service Oriented Development life cycle. This has been challenging to application development teams for several reasons, such as lack of business process documentation or perspective, reuse analysis expertise, and lack of business executive involvement coupled with turbulence in the project teams. This article emphasizes the need to do value-chain analysis – in depth business process analysis coupled with use case analysis across multiple projects identified in enterprise SOA initiative. This article mentions some best practices, wherever appropriate, to point out the vagueness involved in service identification.

Introduction

The process of identifying services can be approached from a number of perspectives, and in combination with a particular project type. Identifying services is most likely one of the first activities in the modeling of a service-oriented solution, and therefore errors made during identification can flow down through detailed design and implementation activities that may necessitate multiple iterations, especially in building composite applications.

The following are the approaches for service identification.

- Top-down
- Bottom-up, exposing existing assets

Top-down approach can be further divided in to two types such as business process driven and use case driven – based on whether the initiative is at the enterprise level, business unit level, or specific to one huge application within one functional domain. If the initiative is at the enterprise level, service identification should be driven by value-chain analysis and in-depth business process analysis. If the SOA initiative is specific to one project falling under a functional domain or enterprise that has just embarked upon SOA adoption with no maturity, most likely it is driven based on use cases for various projects. Note that whether the initiative is at application level or enterprise level, it is always advisable to bring along business process perspective, which is whether it is being automated OR optimized OR supported, into the design and implementation. Without enterprise level or value chain analysis and business process perspective, there is a danger of building service oriented stove-pipes.

Bottom-up approach would enable architects to identify the redundancies at the logic/code and data across enterprise applications portfolio. These redundancies can be potential candidates for the need for services. At the data level, CRUD (Create, Read, Update and Delete) based operations on some entities like customer/user, order, etc. would be probably be appropriate.

Top-Down – Business Process Driven (Value-Chain Analysis)

This is the most preferable approach and best practice to start with for the enterprise transformation to SOA. It is suggested that all consultants and architects have thorough understanding of the enterprise by doing a value-chain analysis both in business and IT perspective. This will enable them to understand what are critical, non-critical, and support functions for the enterprise so that SOA principles and best practices can be applied in the areas where quick benefits will be realized to get necessary executive support and to achieve maneuverability for competitive advantage in the market. And, also, it is advisable to have a clear understanding of interactions between various functions to model the service interfaces that fall within functions in a generic way at the edges of the business processes.

Value chain analysis involves the following activities.

- Understand and capture the broad functional domains and how they interact with each other.
- Detail the business processes and their handshake points (probably in swim lane diagrams that capture each role and its interaction with various activities).
- Detail subprocesses.
- Identify the high-level candidate services (based on process activities).
- As-Is process to IT systems mapping to understand the existing IT systems boundaries. Analysis of IT landscape for business process and application portfolio mapping would help in marking the applications that are affected by SOA. This exercise would lead to better understanding of the SOA projects, and their scope, and would help define the overall integration roadmap.

The following are the advantages of documenting the business process.

- Allows architects to understand the business context
- Bridges the gap between IT and Business teams
- Preliminary list of activities can be treated as potential list of services for further analysis
- Activity/service dependency identifications at a high level
- Business rules identification at a high level from the decision boxes. This will help architects and designers to externalize the volatile parts of the processes.
- Interactions between various actors within the enterprise for workflow requirements
- Better understanding about the complexity within the enterprise in terms of message exchange patterns
- Defining and scoping the SOA project boundaries

Top-Down, Use Case Driven (for composite Applications)

With the customers following the multi-sourcing strategy, coupled with SOA, as a goal, software services vendors would get engaged at different stages. In the case of projects with design-to-deploy scope, SOA vendors will have to start with use cases where the complete business process model is not available or documented as part of either business discovery or plan and define phases. Even if documented business process is available, the tendency is to start with use case documentation in a traditional development process lifecycle. In this scenario, an initial list of services will be determined only after some decent amount of effort on use case analysis for common functional scenarios and sub-system identification. This is usually the case with large projects at a functional domain or department or business unit level.

For both business process and use case driven approaches, it is recommended from our experiences that the project team be organized into services and application teams so that the services team will try to identify/mark the core service requirements for each service, and, at the same time, the application team, organized as per the number of sub-systems/modules, will understand the requirements for each sub-system and will convey their expectations from each service with regard to each of their modules to the services team. The communication from the application team would help the services team in making sure that the interfaces specified would cater to all requirements. The result of this collaboration will be a service specification document for each service. Please refer to Appendix for a service specification template.

The following are the activities involved in the process of developing the traditional software systems:

- Sub-system analysis/Use case modeling
    - Identify the sub-systems
    - Identify the components
    - Map sub-systems back to business processes
    - Identify the related business use cases that directly map on to activities or processes or subprocesses
    - Set of related use cases may become part of a service
- Component specification development
    - Interface definition
    - Methods
    - Tentative method signatures
    - High level attributes
    - Inputs
    - Outputs
    - Dependencies

The following are the additional activities required with respect to services elaboration.

- Service-to-Component Mapping
    - Identify the service responsibilities
    - Service responsibilities distribution to components

- Service Specification Development
    - Interface details
    - Invocation methodology
    - All business use case interactions – to make the service design generic and long-lasting

The services identified with this approach would need to be correlated with the list of services from the top-down business process driven approach to further refine the services and their responsibilities.

Bottom-up – Exposing Existing Assets

The bottom-up approach involves taking the enterprise level application portfolio analysis, assessment for reuse, redundancy, and rationalization effort. In general, the targets for bottom - up approach typically will be redundant business logic, multiple copies of business data entities, or implementing same business logic with multiple products, resulting in huge licensing, operational and maintenance costs, etc. An example for implementing same business logic with multiple products could be implementing search with different product suites in different business units using Autonomy, Verity, Google, etc., within an enterprise!!

Services identified using this approach may or may not be composed to build the business processes. But still the contribution of these services to ROI under the reuse line item could be higher depending on the scenario. Examples of these are user profile, search etc. These services will become part of the basic services, such as the infrastructure services or technical services in SOA service layers.

The following diagram represents the overall process involved in service identification methodology, depicting various enterprise level activities.

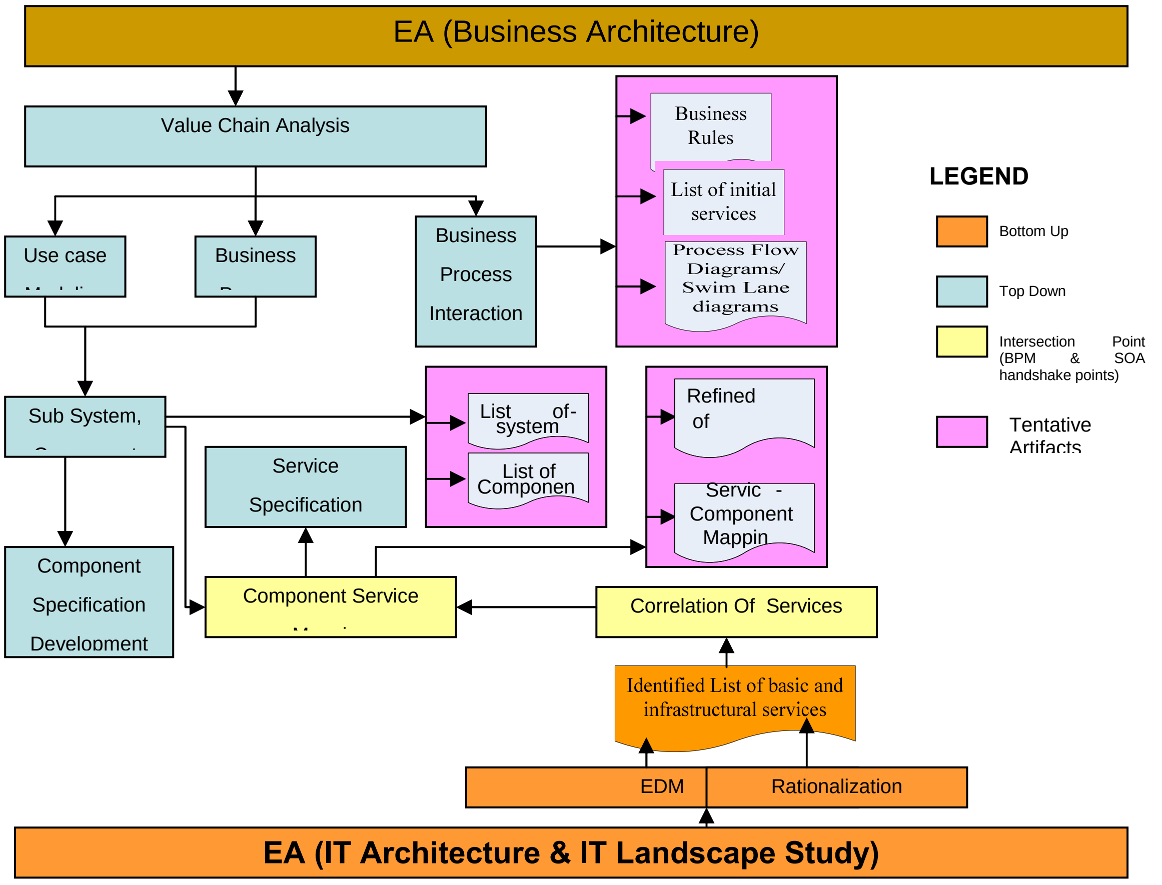
EA (Business Architecture)
Value Chain Analysis
Use case
Business
Business Process Interaction
Business Rules
List of initial services
Process Flow Diagrams/ Swim Lane diagrams
Sub System,
List of- system
List of Componen
Refined of
Servic - Component Mappin
Service Specification
Component Specification Development
Component Service
Correlation Of Services
Identified List of basic and infrastructural services
EDM
Rationalization
EA (IT Architecture & IT Landscape Study)
LEGEND
Bottom Up
Top Down
Intersection Point (BPM & SOA handshake points)
Tentative Artifacts

Figure 1. Service Identification Methodology

BPM and SOA Handshake

This section clearly describes step-by-step account of service identification involving both top-down and bottom-up approaches.

The first step in this process would be to lay out the core business process, and identify the various business activities within it. The following Activity diagram is a template that could be followed to list the key business activities within a business process.

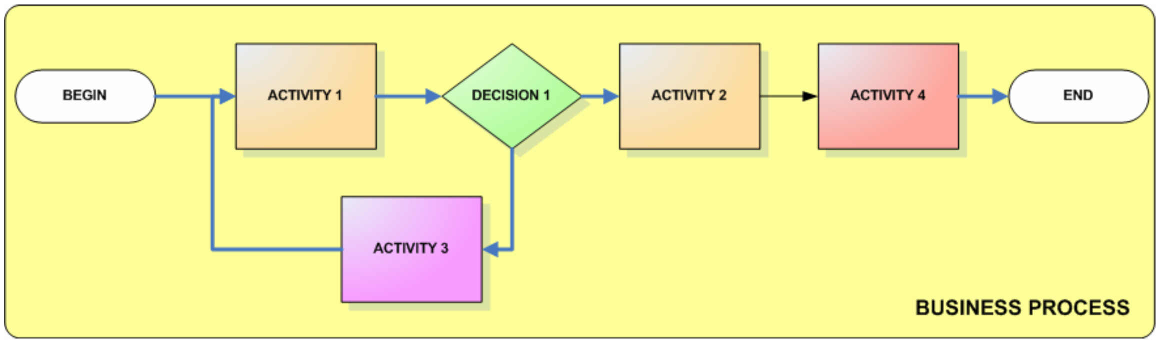

Figure 2. Top-Down Approach Step 1

The second step would be to identify the points of functionality in the existing systems. The functionality offered by applications is exposed as services, and these would act as windows of interaction with the existing applications. The figure below shows the various existing applications in the landscape, and the functionalities offered by them. These points of functionality are then defined and exposed as services in the service layer, as shown in the following figure:

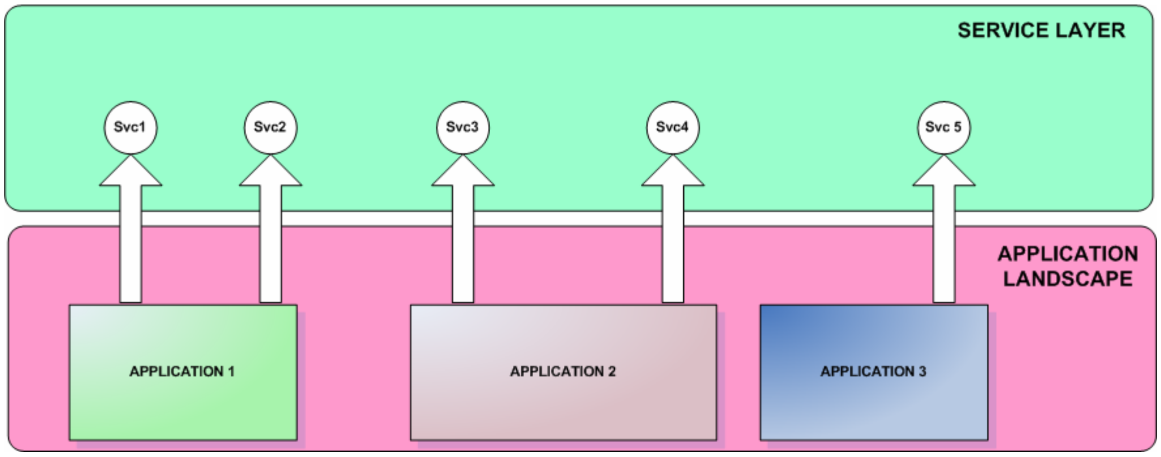

Figure 3. Bottom-Up Approach Step 2

Typically existing assets would be turned into services in any of the following ways.

(i) Wrap an existing function as a service

(ii) Wrap and replace an existing function with a service

(iii) Create a service adapter to make the application work with services

(iv) Integrate the function into a service

The third step is to map the identified services to an activity within the business process. A direct correlation between business activities and the identified services within the application is not likely at the outset. The mappings could be one-to-one, or one-to-many. One service could also be mapped to only part of a business activity. Finally, there could also be certain services that cannot be mapped to any activity within the business process. Please note, at this point we are only trying to correlate the services exposed at the application level with the activities required in the business level.

The following figure shows the preliminary mapping activity between business activities and services exposed at the application level:

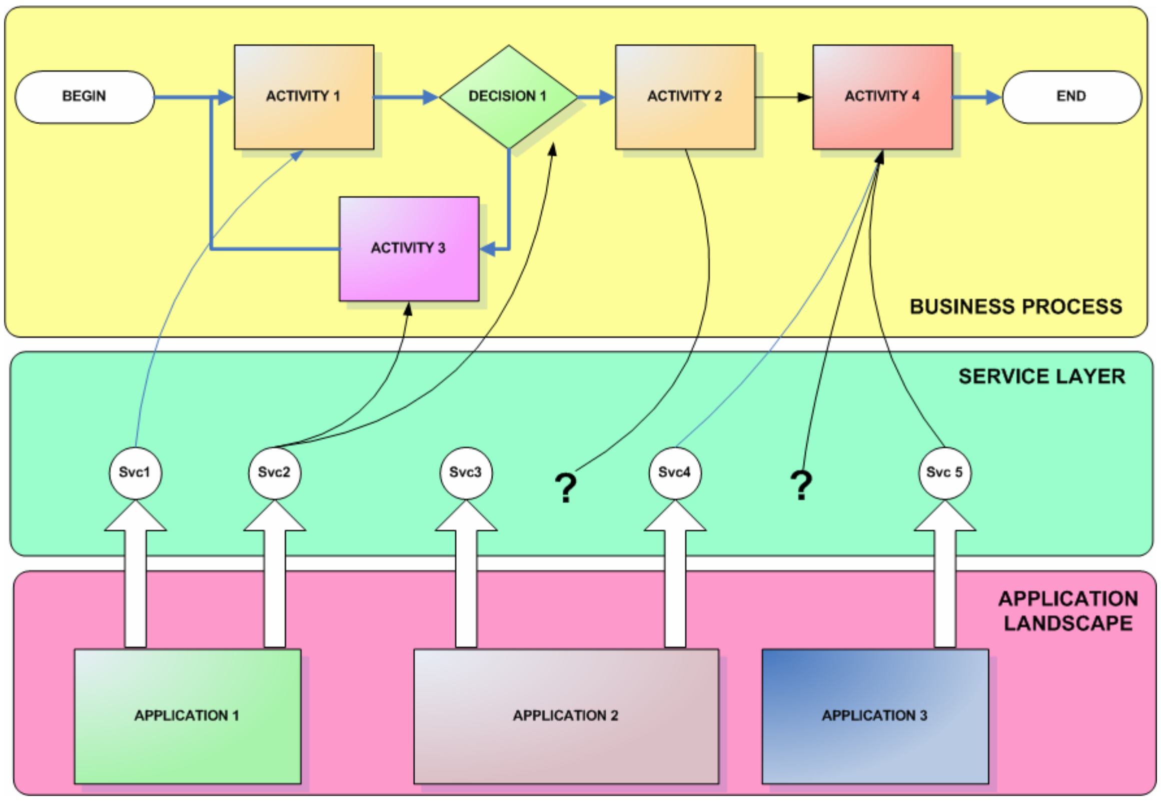

Figure 4. Correlation of Services and Activities Step 3

The final step would be to refine the process model to include more subprocesses in order to achieve a one-one mapping between process and service. However, it is also desirable to achieve a mapping of several decomposed activities to one service in order to maximize reusability of service components. New services need to be created where a business process does not have an equivalent service. The final service mapping should ideally have a one-to-one mapping between business activities and services exposed in the application layer, with a view to reducing maintenance overheads. However, aggregation of two or more services to map to a certain business activity is also allowed.

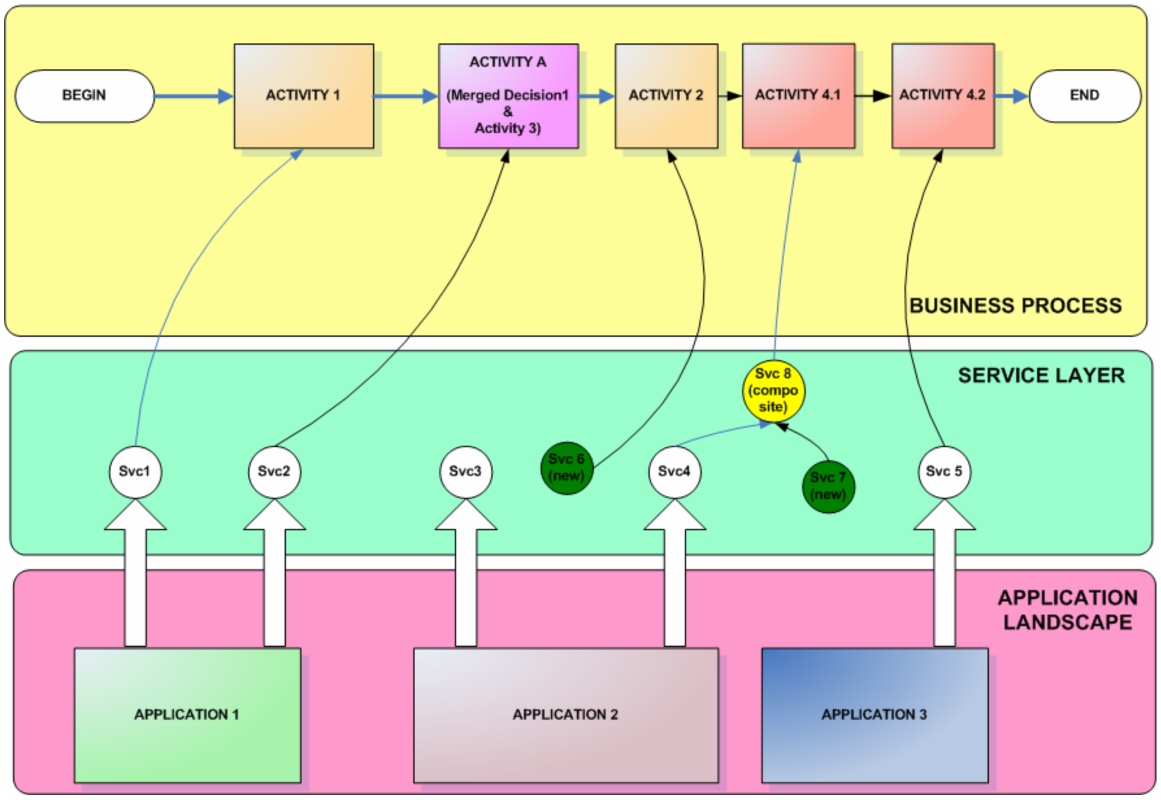

Figure 5. Correlation of Services and Activities (Contd.) – Step 4

After analysis services are identified for an initial list, the next challenge will be to scope the responsibilities of each service. The following diagram depicts the complexity and subjectivity involved in a service boundary definition due to its relationships with business and IT elements like business processes, sub-systems, and business use cases. Please refer to next section for more details to be considered for service granularity.

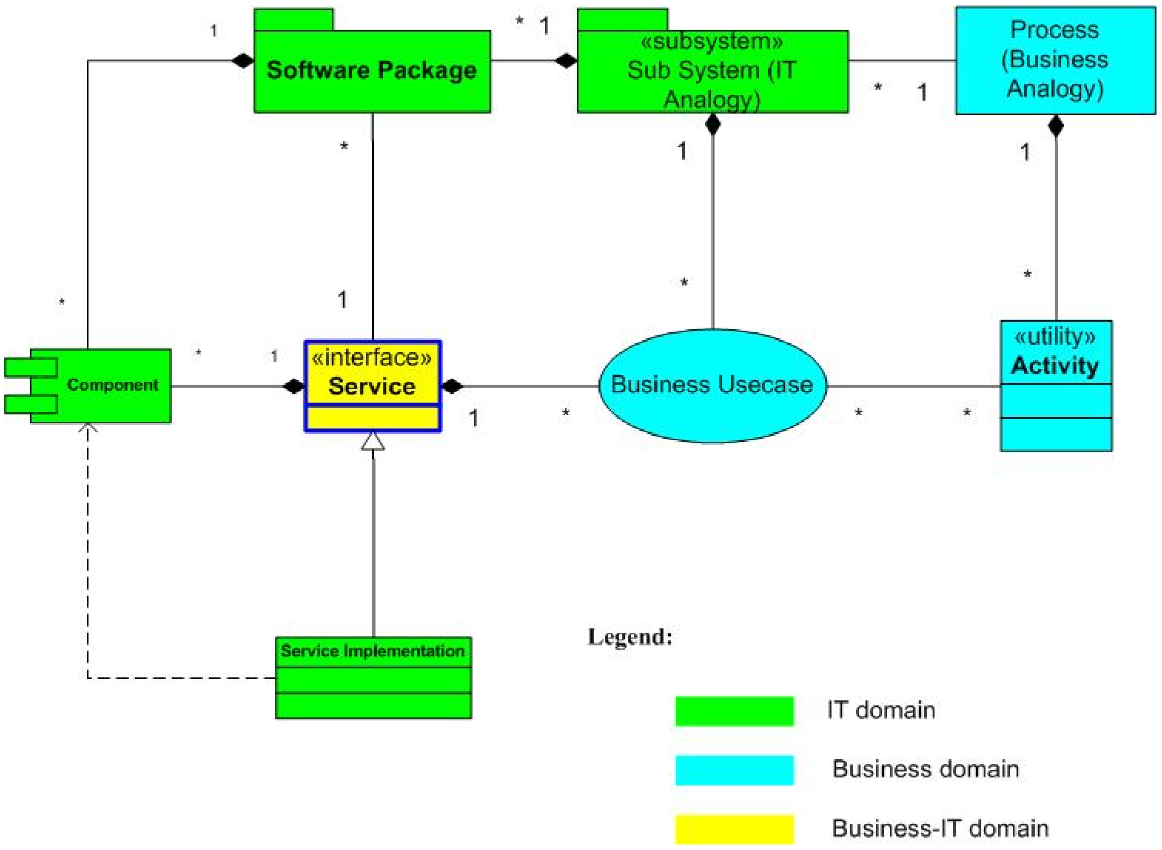

Figure 6. Service Boundary Subjectivity

Some of the points to be considered while deciding on service granularity are

- Reusable
  - Trade off between fine-grained and coarse-grained
  - Flexible and composable in other function or domain areas
- Considerable or reasonable business value
- Payloads of service operations should not be so high that it causes the performance and network overheads.

Need for Effective Governance

Note that there is a danger of service duplication from multiple projects running under the SOA transformation program unless the following important elements are made mandatory as part of SOA Governance structure.

(i) Business executive involvement in defining the service responsibilities, promoting/selling it across the enterprise

(ii) Reuse promotion

- Enterprise must have SDLC review toll gates with EARB (or necessary IT processes)
- Single enterprise wide repository to host service related artifacts

(iii) Enterprise Architecture Review Board (reporting to CIO) should act like of task force to make sure that every project requirement is assessed for reuse opportunities right in the business discovery phase or idea generation phase itself:

- in using the existing services
- to identify the potential future services
- to provide the guidelines to the application team in both the above scenarios

Having the above-mentioned high level business and IT executive support would take the enterprise onto agility highway via RAMP (Reuse Across Multiple Projects). Without the above support, SOA efforts would result in Service Oriented Stovepipes – however good your services and service identification methodology may be.

Conclusions

This article has brought out the ways to follow the identification of services and best practices while applying them during an enterprise level SOA transformation. This article also specified the required supporting IT processes that should be in place for successful reuse promotion and to prevent duplication or service proliferation. This article has clarified the need for business and IT team collaboration for identification of enterprise level services. Service identification is one of the core elements of the BPM and SOA handshake that reinforces the current mantra that "BPM should be service oriented, SOA should be business process focused, and SOA takes over where BPM leaves the enterprise in a path towards agility."

Appendix

Service Specification Template

The following table depicts the content coverage within a service specification document. This will be useful to architects, designers, and project managers.

The intent of service specification is to convey the high level design idea behind the component to design and build to the IT team. It will also help the business team to understand internal details such as key features, business rules implemented, constraints, and limitations. Key elements of Service in SOA are interface, contract, and implementation. As part of the high level design, these need to be captured to help the designers and build a team in the later phases.	
Enterprise SOA Layer	The layer in which this particular service fits in with regard to enterprise SOA layers.
Service Interface	This depicts overall functions exposed to the clients of service. These will be arrived at by carefully analyzing the various use cases for commonality, special requirements, and reuse. Different methods and functions of the service may encompass a major end-to-end functionality OR provide the same logical function but with different inputs and outputs, depending on the requirements. This should also cover high level inputs and outputs of every method call within the service interface.
Service Contract	In this section are inputs and outputs to be captured in terms of high level entities, value objects, etc. This should also highlight any of the important attributes that may have different values and help in changing the service flow functionality.
Service Configuration	All necessary configuration details, both on service and client side, have to be captured.
Service Implementation	This should specify the implementation details of each function in terms of either a real or a pseudo code. Documenting the pseudo code and scenario diagrams may be possible only at the end of the design phase. Hence, it is advisable to mention how different use cases would use this particular service to realize the functionality at a very high level in terms of specifying the methods. ▪ Components used back-end ▪ Policies to be enforced ▪ References to Use Cases/Requirements
Type of Service Invocation	Synchronous/Asynchronous
Service Invocation	This section is intended to communicate how application specific layers would be communicating to the service being discussed. This section describes how the responsibilities division across the application and component influences the deployment architecture. In other words, there are various deployment options available for consideration with some tradeoffs between whether patch application to component/service is acceptable whenever a new application gets added to the service client portfolio and whether to keep completely the application specific logic on the application side rather than with component/service. The service invocation section explains how to build a service with regard to both the options.

Service Dependency	This should document the various services (in different SOA layers) which are required for this layer to complete end-to-end functionality.
Data Dependency	Master data requirements, high level service data schema
Service State	Stateless/Stateful
Local/Network	Connected/Disconnected/Both
Hosting Location	Client/Server/External System
Code Packaging	Describes the classes to be packaged together in different modes
Designed Limitations	Mention all the constraints or business requirements leading to the current design, for future reference.
Known Defects	This should explain both technical and business defects.
References	Articles/Patterns/Standards to be referred

References

1. Service oriented modeling and architecture - How to identify, specify, and realize services for your SOA by Ali Arsanjani, 09 Nov 2004, IBM
2. Elements of Service Oriented Analysis and Design – An interdisciplinary modeling approach for SOA projects, Olaf Zimmermann & Co., June 2004, IBM
3. TOGAF v8.1: Architecture Board’s role, responsibilities, operation
4. EARB Reference Guide for a big retail customer – Wipro proprietary documentation
5. Research paper. ”Service Oriented Architecture and Process,” 10 February, 2006 IBM
6. Research Paper. ”Positions 2005: Service Oriented Architecture Adds Flexibility to Business Processes,” 16 February, 2005 Gartner

Glossary of Terms

Acronym/Abbreviation	Definition
BPM	Business Process Management
CRUD	Create, Read, Update, Delete
EA	Enterprise Architecture
EARB	Enterprise Architecture Review Board
EDM	Enterprise Data Model
RAMP	Reuse Across Multiple Projects
ROI	Return On Investment
SDLC	Software Development Life Cycle
SIM	Service Identification Methodology
SOA	Service Oriented Architecture

Acknowledgements

We would like to thank Dr. Udaya Bhaskar Vemulapati, General Manager, Wipro Technologies for giving us the required time and support in many ways in creating this article as part of an SOA Center Of Excellence efforts. We want to thank our colleagues, Kapalee Balakrishnan and Sriram Aravamudan, for sharing their experiences and thoughts as we prepared this article. We also want to thank our colleague Prasad Palli for providing review feedback on the content.

Authors

Srikanth Inaganti is a Senior Enterprise Architect in the Enterprise Consulting and Architecture Practice division of Wipro Technologies. Srikanth Inaganti, PMP, E-Architect, ECAP Group, Wipro Technologies srikanth.inaganti@wipro.com, Office # 91 40 3079 5110; Mobile # 91 9849058064

Dr. Gopala Krishna Behara is a Senior Enterprise Architect in the Enterprise Consulting and Architecture Practice division of Wipro. Dr Gopala Krishna Behara, E-Architect, ECAP Group,Wipro Technologies gopalkrishna.behra@wipro.com, Office # 91 40 3079 5110; Mobile # 91 9949997724