www.ModelingConcepts.com

Do not be afraid to ask!

A Quick Chat About SOMF Structural Modeling

For architects, business analysts, system analysts, software developers, modelers, team leaders, and managers

Use the SOMF modeling capabilities for enterprise architecture, application architecture, service-oriented architecture (SOA), and Cloud Computing projects.

SOMF is empowered by Sparx Systems Enterprise Architect modeling platform

About SOMF Structural Modeling

In addition to other capabilities, the Service-Oriented Modeling Framework (SOMF) enables structural modeling of software assets. Remember, structural modeling is all about physical or even logical manipulation of a software component. In other words, this is about transforming software internal building blocks and external (environment) architecture. These modifications to software are pursued to satisfy business and technical requirements.

Again, structural modeling is not about transformation of ideas and abstractions; it is merely about logical or physical software structure modifications.

What is a service?

According to SOMF, the notion of a service is generalized to a higher abstraction level. Therefore, when you are modeling software, regard a service as any software asset that your organization has been constructing, acquiring, or will be building in the future. This definition may include software entities such as an application, a Web service, a database trigger, a library, an enterprise service bus (ESB), a business process, or a .NET or Java class.

Structural Modeling Software Assets

The software assets that are illustrated in a SOMF diagram are services. These services are categorized in three different structural types, as depicted in Figure 1:

1. *Atomic Service*: an indivisible and fine-grained software asset that typically offers limited processes, interfaces, and capabilities. Example: Customer Information Service that provides high-level account information, such as name, address, and phone number

2. *Composite Service*: a coarse-grained software asset that contains internal finer-grained services. Examples: an application that encompasses smaller modules, an ESB that includes internal orchestration and business rules engines, a coarse-grained Web service that offers a large number of trading capabilities, and more

3. *Service Cluster*: a collection of atomic and/or composite services that collaborate to provide a solution. Example: An accounting service cluster that offers accounts receivable, accounts payable, and payroll modules

Figure 1 also illustrates a consumer, a generalized notion of any software entity that may not only provide services, but also calls other services for data and information.

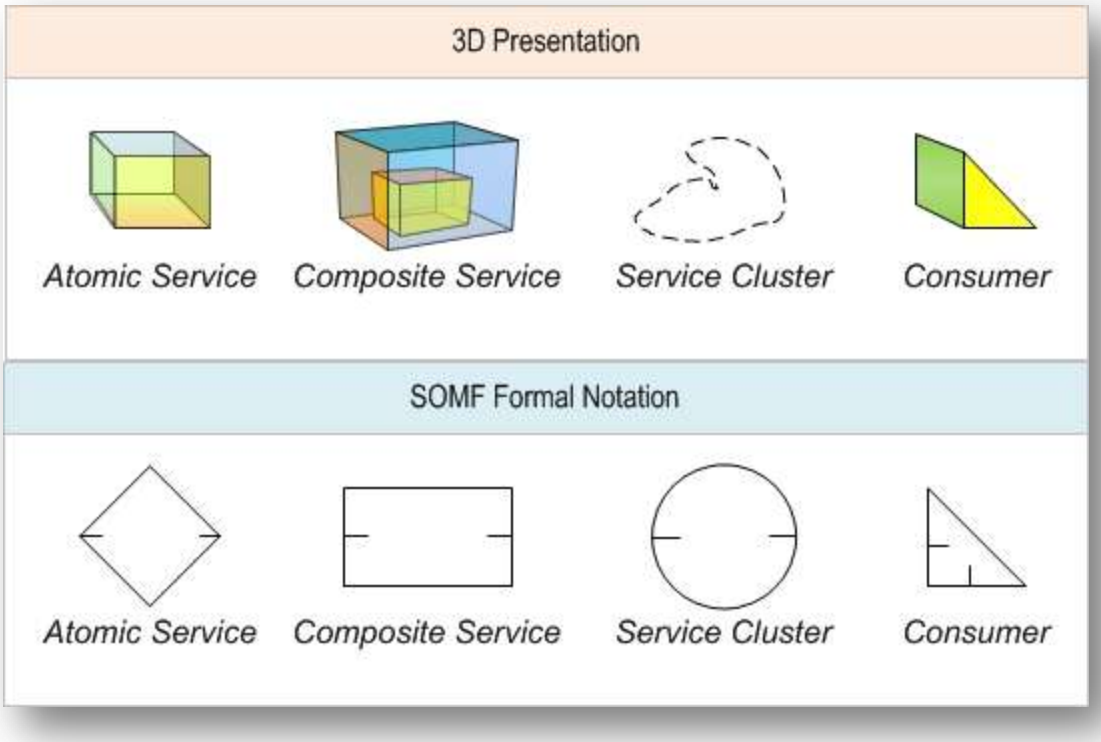

Figure 1: SOMF Software Modeling Assets

SOMF Structural Modeling Notation

Figure 2 illustrates the notation that is used in this tutorial to demonstrate structural modeling in SOMF. The apparent symbols are typically used during the service discovery and analysis phase of a project. These symbols are discussed in details in the sections that follow.

Remember, as discussed earlier, the notation that is depicted in Figure 2 is provided to help practitioners to focus on the structural aspects of service modeling.

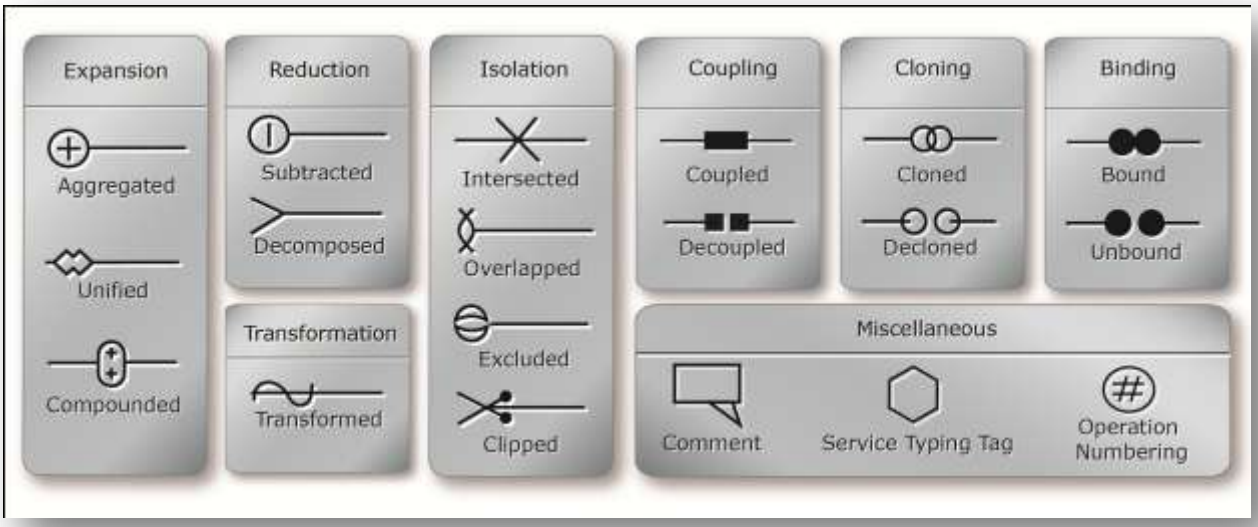

Figure 2: SOMF Structural Modeling Notation for Service Analysis and Discovery

Aggregation

How would you illustrate a composite service that contains internal services? Figure 3 illustrates this scenario. Note in the SOMF Notation panel on the far right, the "Aggregated" symbol (plus sign) is pointed to composite service C. This modeling operation implies that atomic service A is now a part of composite service C.

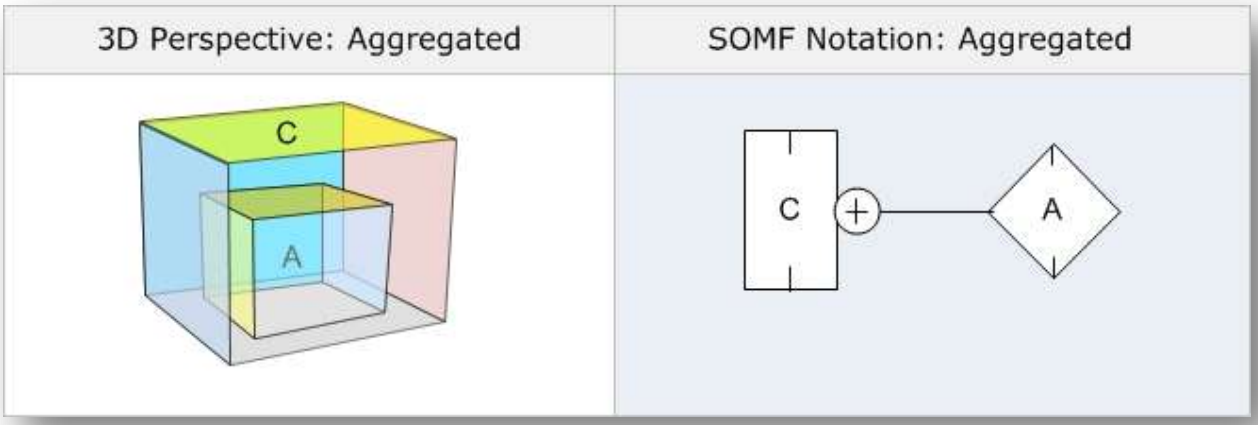

Figure 3: Service Aggregation

Decomposition

And how would you separate a service from its parent service? Figure 4 illustrates this scenario. In the apparent SOMF Notation panel, on the far right, you will notice the usage of the "Decomposed" symbol that signifies the separation of service A (child service) that is contained in its parent service C.

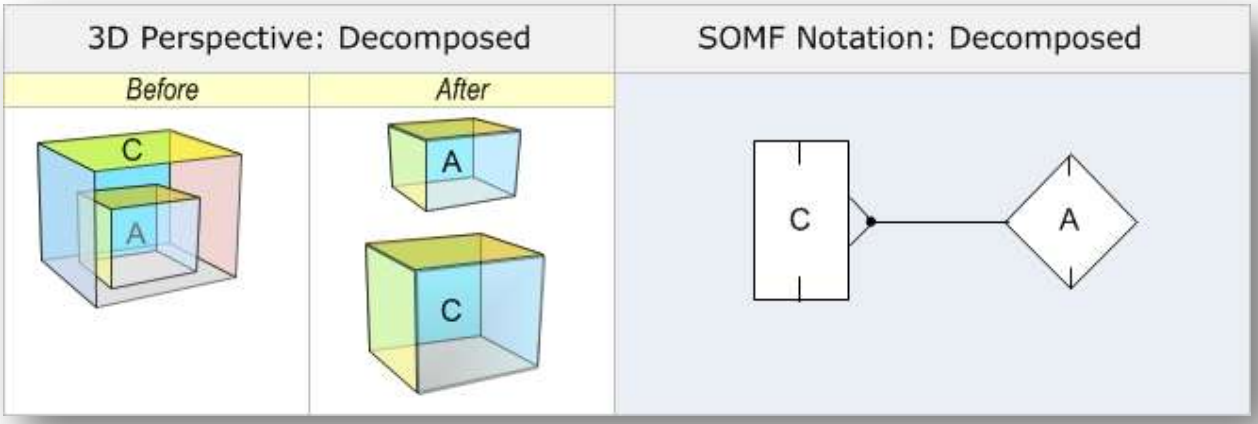

Figure 4: Decomposing a Service

Subtraction

Service subtraction is a modeling operation that indicates retirement. The term "retirement" pertains to an impractical service that should be permanently removed from a design blue print or even from production. If service capabilities are not needed any loner, use the "Subtract" modeling operation to signify the termination of the service life cycle. Figure 5 depicts retirement. Here the "Subtract" symbol points out that service A will be retired and eliminated from service C.

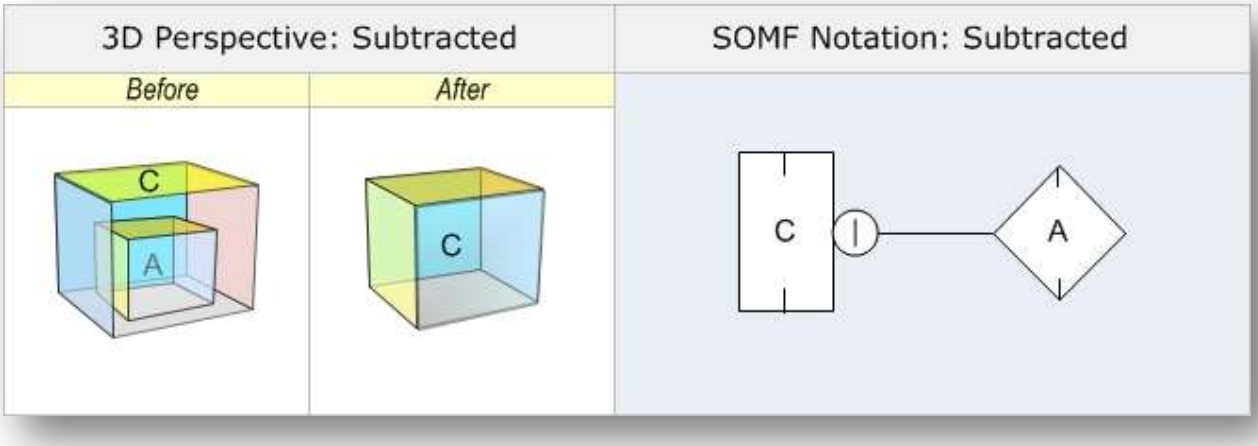

Figure 5: Decomposing a Service

Coupling and Decoupling

Too often we are required to provide a diagram in which a service is affiliated to another service. This brings us to the coupling modeling operation which is devised to indicate an association between a service and its peers. This relationship may imply message exchange between two or more services, an informal contract that will materialize in the future, and more.
To depict a connection between services, employ that "Coupled" symbol, as illustrated in Figure 6. Note the indicated affiliation between atomic service A1 and atomic service A2.

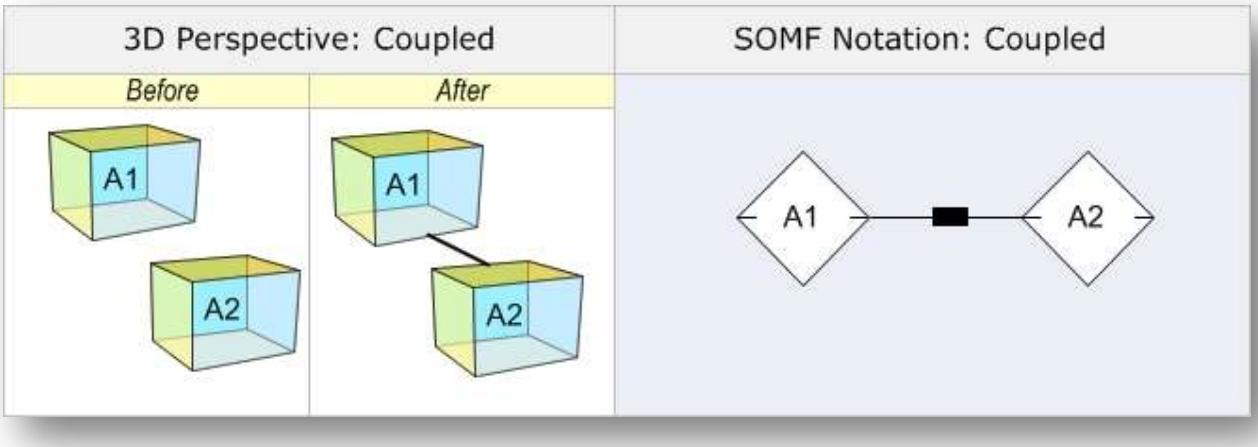

Figure 6: Service Coupling

To signify a decoupling modeling activity, use the symbol "Decoupled" as illustrated in Figure 7. Here atomic service A1 is decoupled from atomic service A2. This operation indicates past relationship between services that are no longer recognized. Ceasing associations between services typically signifies discontinuation of message exchange activities between these software entities or termination of binding contracts.

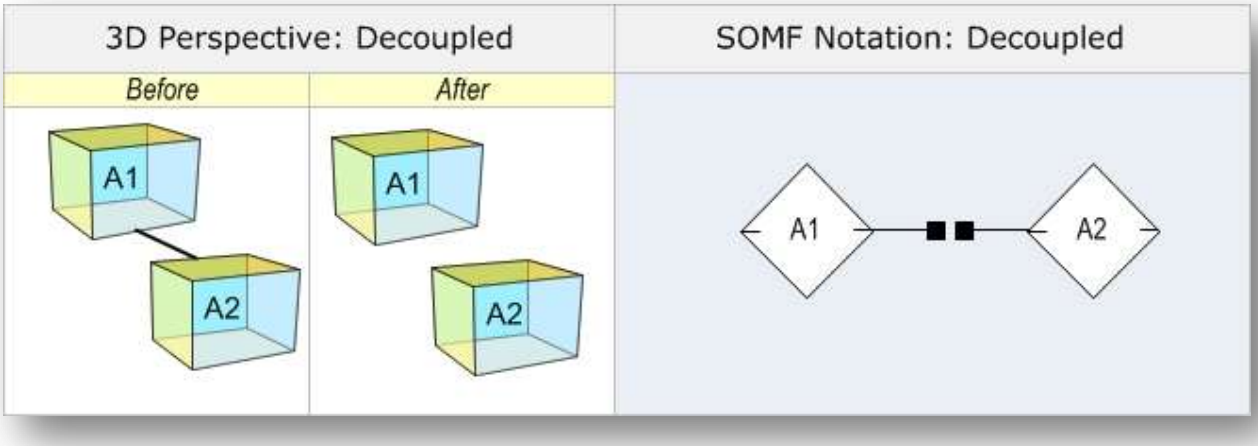

Figure 7: Service Coupling

Compounding

Imagine a group of services that complement each other's capabilities, assembled to perform business or technical activities. This modeling operation is named compounding because of the limited goals and functionalities that are assigned to the group. Thus, compounding differs from service clustering because of the smaller operation scope. Employ the "Compounded" symbol, similar to the illustrated scenario in Figure 8, to indicate gathering a small group of services that work together to provide a solution. Note that composite service C, atomic service A1, and atomic service A2 are compounded.

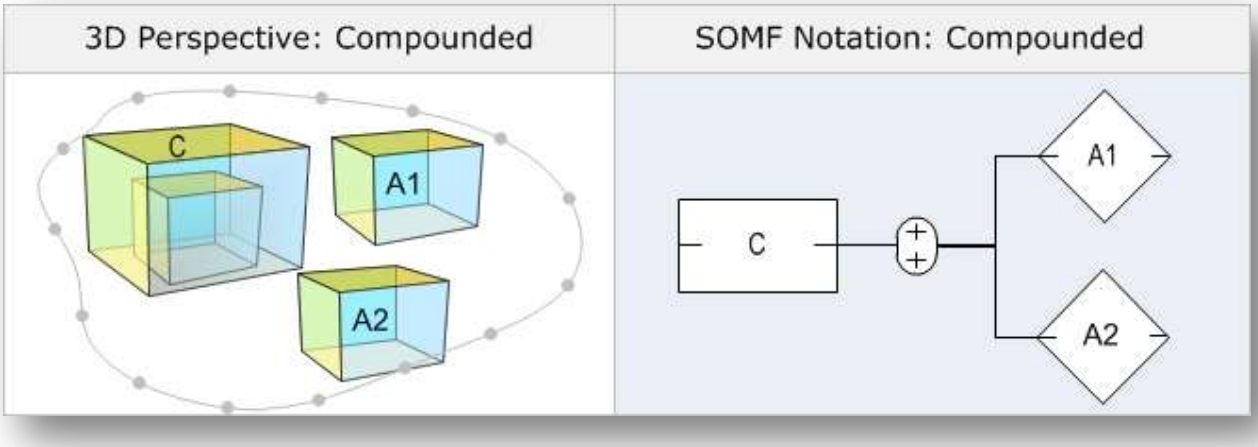

Figure 8: Service Compounding

Unification

In cases where there is justification for merging two or more services to form a newly coarse-grained service, you may want to use the "Unified" symbol to model this operation. Unification of services is typically pursued when a service is too fine-grained, and offers limited capabilities and functionality. In this case, you may want to unify two or more services that offer similar reduced functionality. Figure 9 depicts a service unification scenario in which two services: atomic service A1 and atomic service A2 are merged, forming atomic service A1+A2.

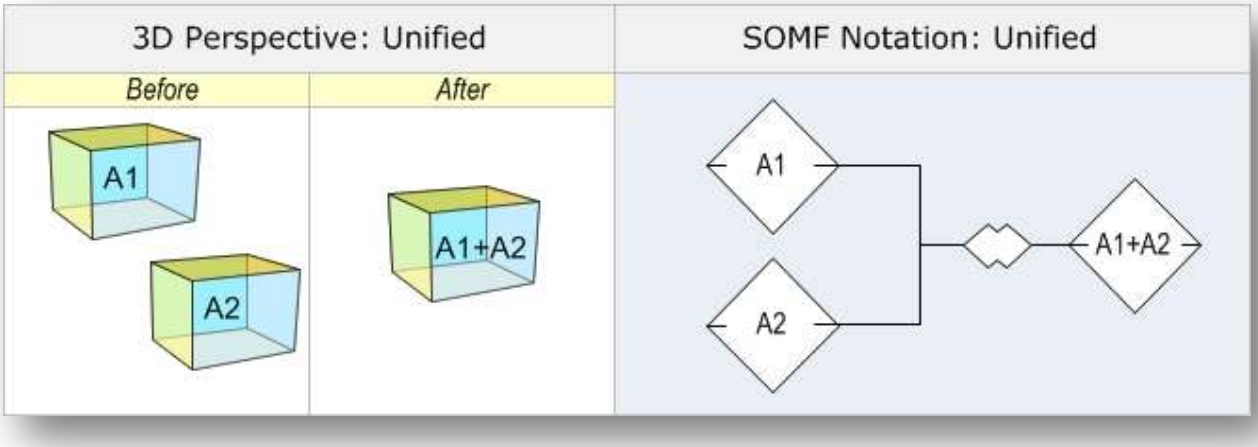

Figure 9: Service Unification

Transformation

Should a decomposed composite service structure that does not contain any more internal services transform into an atomic formation? The answer is "yes". Should an atomic service that aggregates smaller services transform to a composite service? Again, the answer is "yes". Therefore, employ the "Transformed" symbol to indicate a modeling transformation operation similar to the example that is shown in Figure 10. Note that composite service C is transformed into atomic service CA.

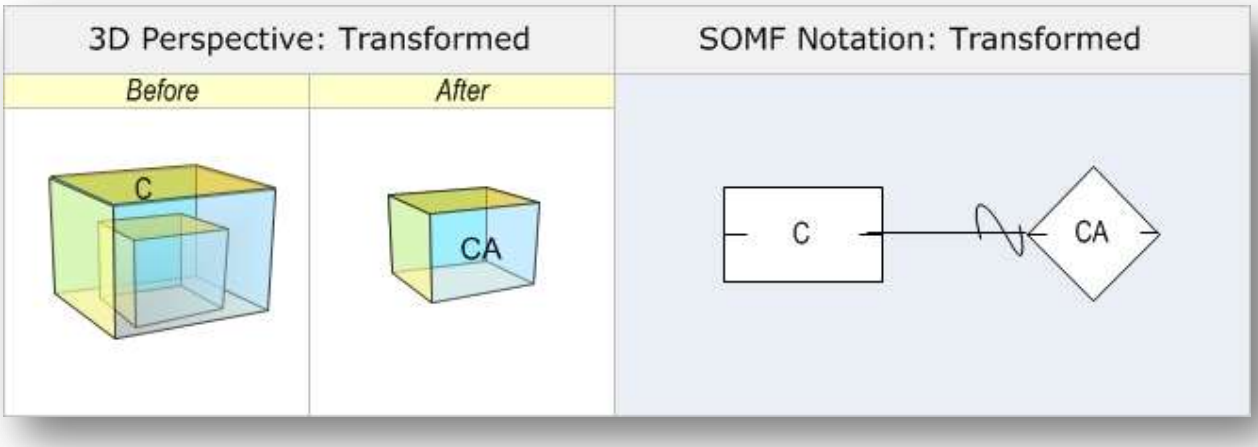

Figure 10: Service Transformation

Intersection and Overlapping

The intersection modeling operation should be applied on two or more service clusters. This modeling activity is employed to discover new services, identify common functionality in two or more service clusters, or even analyze service capabilities that may be excluded. Figure 11 illustrates two intersected service clusters by employing the "Intersected" modeling symbol. Note that the intersection of service clusters CLS1 and CLS2 creates an overlapping region that contains an atomic service.

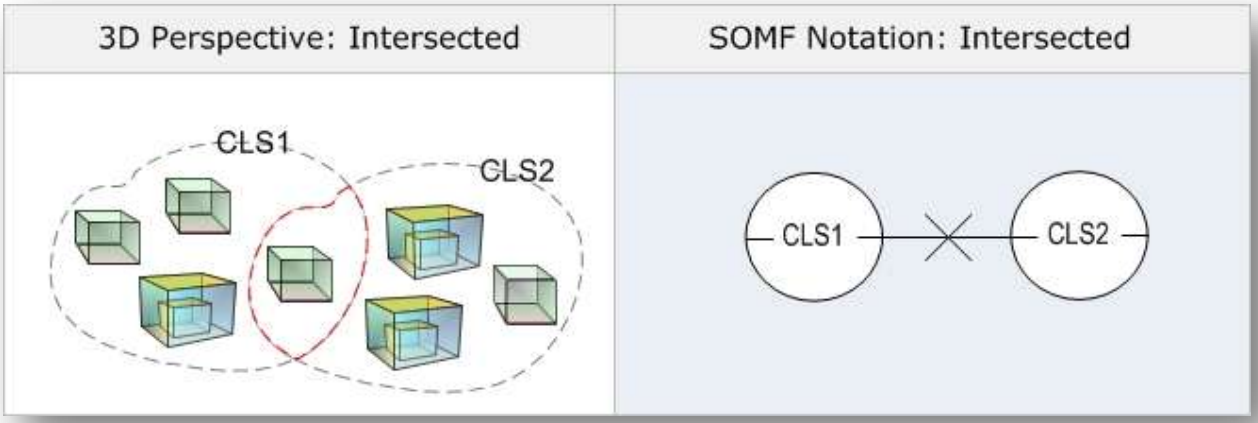

Figure 11: Service Intersection

To illustrate an overlapping region, use the "Overlapped" symbol as depicted in Figure 12. It is apparent that this intersection between service cluster CLS1 and service cluster CLS2 resulted in an overlapping region that now contains atomic services A1 and A2.

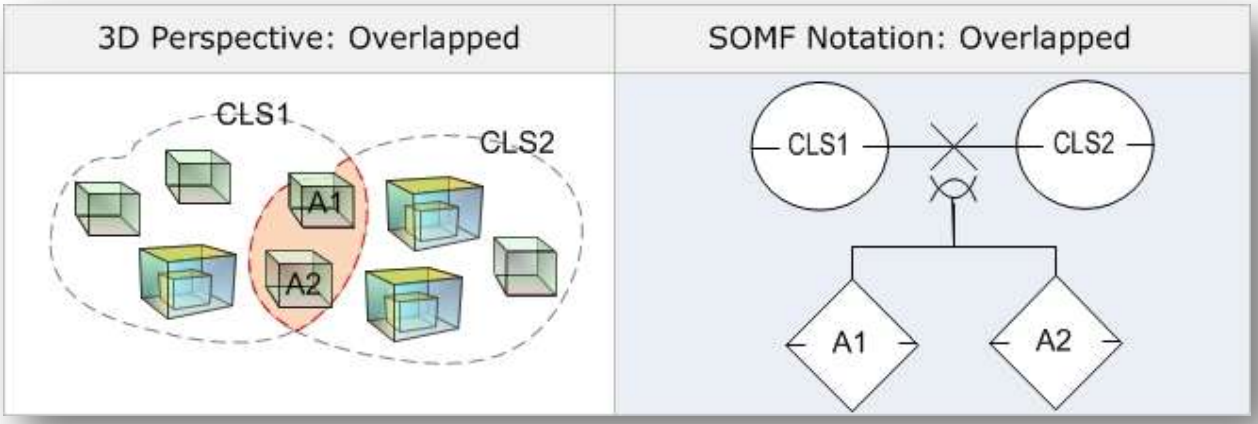

Figure 12: Overlapping Region

Exclusion

How should we exclude services from two or more clusters? Figure 13 depicts service cluster CLS1 and service cluster CLS2 with their mutual intersected region that contains atomic services A1 and A2. This exclusion modeling operation may suggest that these services are not needed in a production environment, a design and architecture effort should not focus on the excluded area, or the excluded services offer redundant functionality. Note the "Excluded" symbol that is used to signify exclusion of service capabilities.

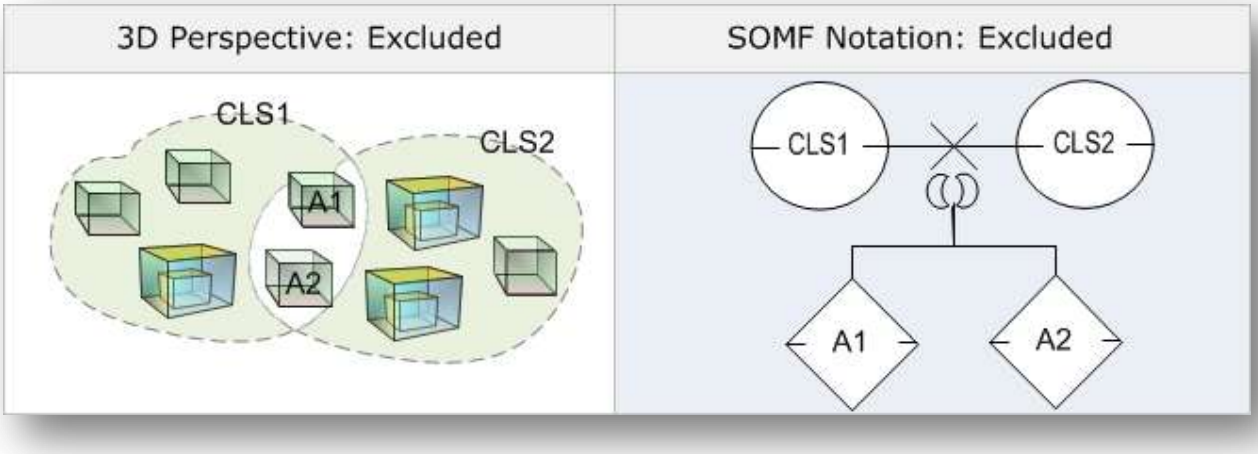

Figure 13: Service Exclusion

Clipping

If your service ecosystem contains not only clusters, but also standalone atomic and composite services, use the "Clipped" symbol to identify the selected services that are singled out for a particular design effort, or for continuous analysis. This clipping modeling operation can be applied to any service structure despite any architecture complexity. Figure 14 illustrates the usage of the "Clipped" symbol to isolate three atomic services: A1, A2, and A3 from the environment they operate in. Note that A1 is a child service that is aggregated in composite service C; A2 is a standalone service; and Service A3 is a part of cluster CLS1.

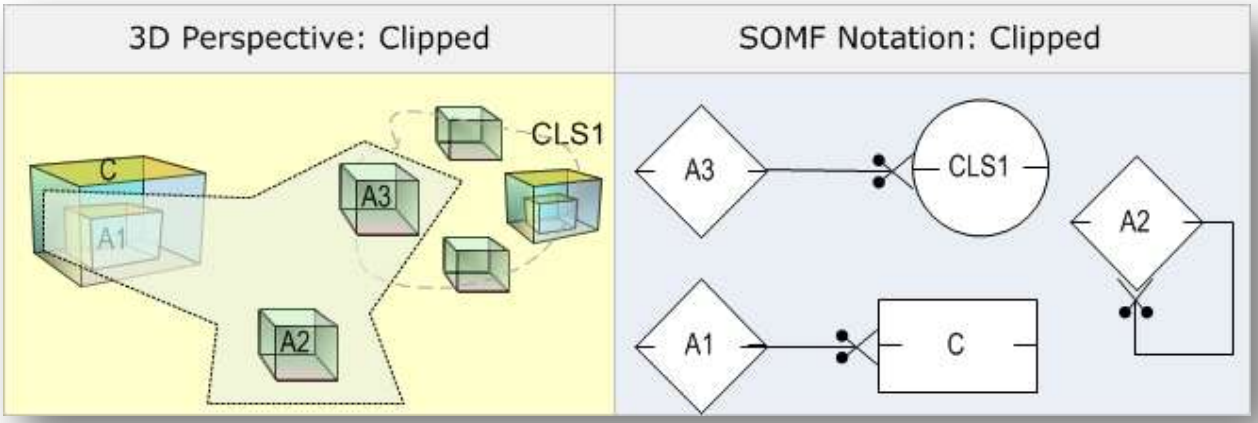

Figure 14: Clipping Region

Binding and Unbinding

Service binding is a contract that is formed between two services, a consumer and its corresponding service, or a service and its related consumers. The "Bound" modeling symbol, therefore, identifies a strong link between two entities that will be, or are currently exchanging messages. Figure 15 illustrates a scenario in which atomic service A1 is bound to a consumer. In this case a consumer may be another service or a consuming application.

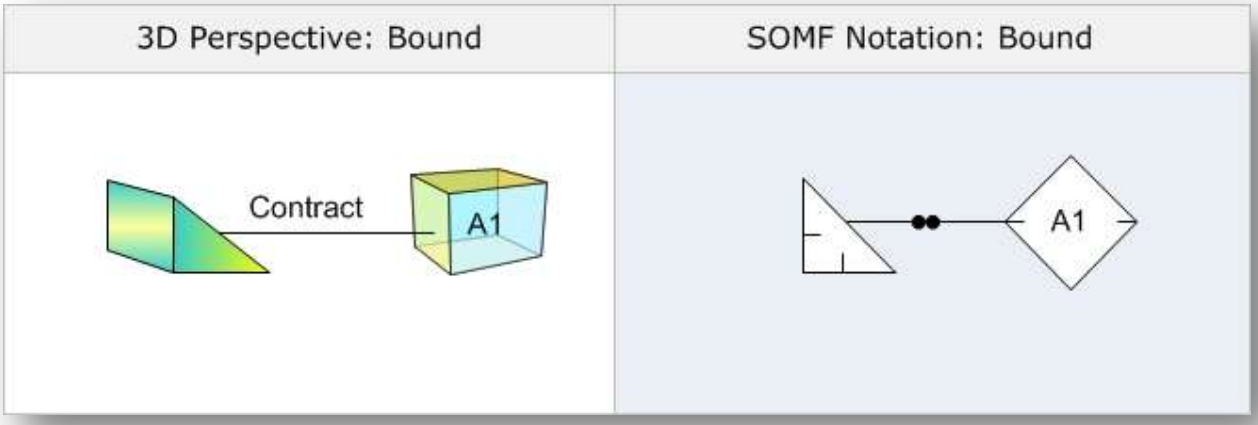

Figure 15: Service Binding

Service unbinding is illustrated in Figure 16. Note the usage of the "Unbound" symbol that indicates discontinuation of a contract that took place between atomic service A1 and a consumer.

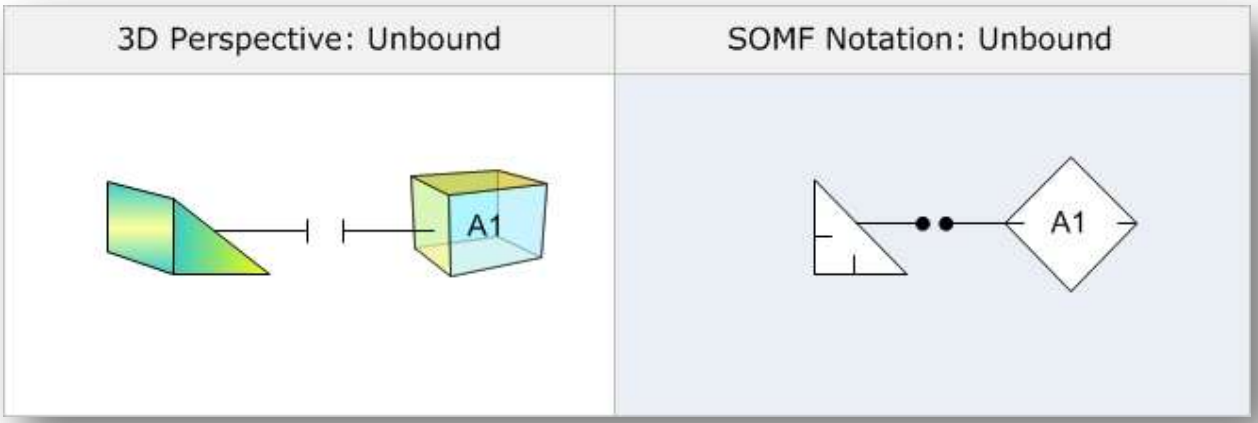

Figure 16: Service Unbinding

Cloning and Decloning

Service cloning is a modeling operation that signifies duplication of a service. This duplication activity produces an identical service that will operate, or have already been offering business or technical functionality in a production environment. Moreover, the modeling cloning operation can be employed to illustrate federation of services or even disaster recovery installations. Figure 17 depicts cloning of composite service C1. Note that the C1 is actually duplicated by using he "Cloned" symbol.

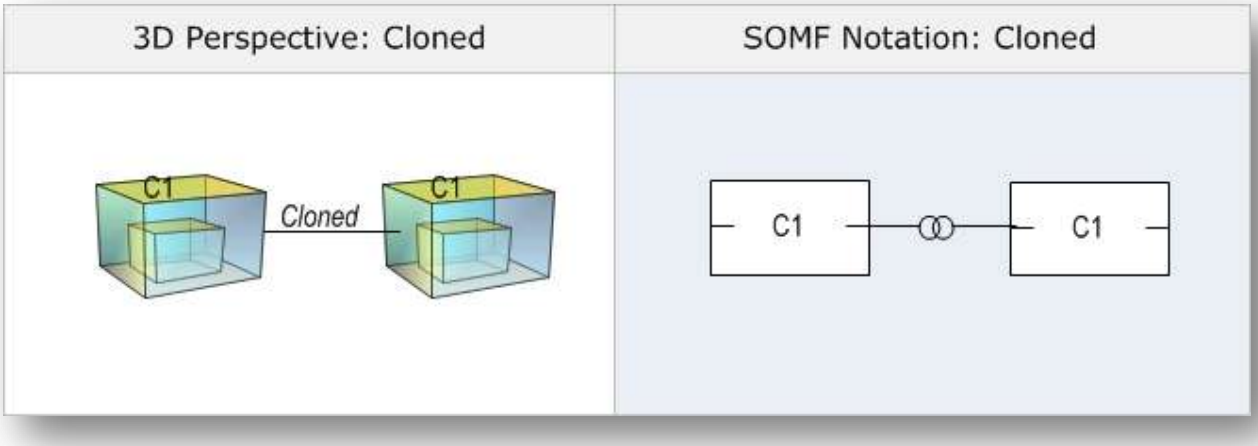

Figure 17: Service Cloning

Finally, the decloning modeling activity is depicted in Figure 18, in which service C1 is disassociated from its cloned composite service C1.1. In this scenario, the decloning effort produced two distinct services that may offer different variations of capabilities.

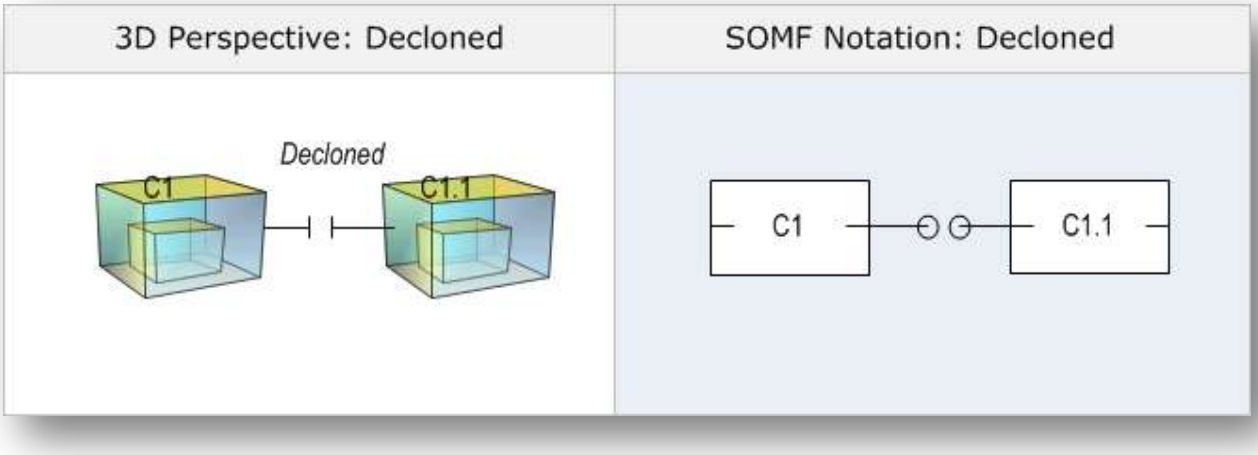

Figure 18: Service Decloning

SOMF Literature

To learn more about service structural modeling and other modeling topics refer to these three books on service-oriented modeling: