

Design Patterns for Policy-Based Service Engagements

Yathiraj B. Udupi

Department of Computer Science
North Carolina State University
Raleigh, NC 27695-8206, USA
ybudupi@ncsu.edu

Munindar P. Singh

Department of Computer Science
North Carolina State University
Raleigh, NC 27695-8206, USA
singh@ncsu.edu

Abstract

Service engagements arise commonly in business and scientific computing. A service engagement is characterized by autonomous parties coming together in a contractual arrangement to share resources or carry out tasks for one another. Recent work shows how to model service engagements in an interactive manner and at a high level. This work formalizes the atoms of a service engagement as commitments among the participants, to be created and manipulated as the engagement progresses. Further, it scopes the commitments of an engagement in a (virtual) organization, and specifies how the policies of the participants affect their interactions. This paper contributes design patterns for service engagements formulated in terms of roles, commitments, and allied concepts. Each pattern reflects a distinct element of a service engagement from a business perspective and highlights exactly where policies apply. This enables the perspicuous, reusable specification of service engagements.

1 Introduction

The rising prominence of services poses new challenges to computing. Services have a natural match with policies, yet existing literature on policies has not specifically addressed the challenges of services. Service engagements are characterized by autonomous parties coming together in a contractual arrangement to share resources or carry out tasks for one another. The participants exhibit complex behaviors by autonomously applying their own policies.

Service engagements arise commonly in business and scientific computing. Let us consider an example of how even a simple service engagement can turn out to require complex behaviors. Consider a (fictitious, but realistic) scenario involving two organizations, the National Oceanic and Atmospheric Administration (NOAA) with member National Hurricane Center (NHC) and the National Labs Or-

ganization (NL) with members Argonne (ANL) and Oak Ridge (ORNL). A service engagement exists wherein NL provides grid computing services to NOAA for hurricane modeling. This engagement may get delegated, assigned, or otherwise manipulated to result in engagements among their members.

A challenging use case here is that of *preemptive scheduling* of resources in the light of critical events. Consider a scenario when ANL has a service engagement with NHC as well as another organization (say, a Data Mining (DM) organization). Say, certain events analyzed by NHC indicate an emergency hurricane situation causing a surge in NHC's computational resource requirement beyond what is currently being offered by ANL. In this scenario, if NHC makes a request to ANL for additional resources, ANL might deny the request because of the other service engagement. This situation demands for *preemptive scheduling*, where ANL should be preempted from its lower priority service engagement with DM, so that it can satisfy the high priority request from NHC.

The above use case is challenging because it involves more than one autonomous stakeholder. Thus, there is not a unique point where a desired policy can be applied. Further the desired outcome depends upon how different business entities relate to one another. Anticipating the dynamics of such "configurations" when authoring policies is clearly not viable. Further, the challenge of implementing using policies is not about what the policies are, but where they are placed and the vocabulary of interactions and relationships to which they apply. Consequently, current approaches rely on hard-coded solutions or human intervention.

Our recent work provides an agent-based conceptual model [1] and a policy-based governance architecture [2] for addressing the above kinds of challenges. This work is centered on the notion of *agents*: computational entities that model autonomous, heterogeneous, and dynamic entities. A service engagement is captured as a set of *commitments* between the participating agents. A commitment is a directed three-party obligation from a *debtor* to a *credi-*

tor arising within the scope of a *context*. The context provides a means to handle exceptions by revoking or otherwise manipulating the commitments. The enactment of a service engagement proceeds by various commitment manipulations such as *create*, *discharge*, *cancel*, *release*, *delegate*, *assign*, and *escalate* [1]. An organization or *Org* forms the context for the commitments of an engagement. Orgs are multiagent systems, each individual and Org being modeled (recursively) as an agent. Agents participating in a service engagement exhibit rich policies to govern their actions. Based on the observed events and its policies, an agent may choose its actions. Here, the policies include the operational policies (controlling the domain specific operations) as well as the policies that control the manipulations of commitments and other configurations of the agents.

In the above example, NHC and ANL are individual agents, NL and NOAA are Orgs. Org-NOAA-NL is created as a context Org for the commitment from NL to NOAA. Org-NHC-ANL is created for the commitment from ANL to NHC. ANL or NHC may send an escalate to Org-NHC-ANL. If NHC escalates, Org-NHC-ANL will apply its *escalate* handling policy before deciding its action. If the request is legitimate and if ANL cannot grant it, then Org-NHC-ANL can forward the escalate to Org-NOAA-NL. In well-designed organizations, exceptional situations are mostly handled locally, and forwarding of escalations is rare. The higher Orgs may preempt some service agreements in favor of others. However, sometimes an escalation may not be resolved, and the original commitment is violated. Here, Org-NOAA-NL would request National Labs to provide the additional resources to NHC. National Labs may preempt ANL to give up its service engagement with DM to support NHC fully instead.

The policy-based governance approach [2] produces proactive behaviors based on a specification of the Org of the given engagement. However, specifying an engagement directly via commitments is unwieldy. Fortunately, engagements exhibit several key repeating patterns. This paper formalizes such patterns in terms of roles, commitments, policies, and allied concepts to provide a powerful, yet simple vocabulary to specify engagements. Each pattern reflects a distinct aspect of some component of a service engagement.

2 Organizational Design Patterns

Designing a service engagement so that the right interactions ensue is difficult because what is right depends on what the stakeholders want. Designing a service engagement involves specifying an Org in which the engagement will be enacted. This includes specifying the member roles and their capabilities, the relationships among the roles, the policies of each role, the constraints on the roles and their capabilities, the commitments required of the roles. Ser-

vice engagements have elements that fall into repeating patterns, which we can formalize and use for specifying engagements. An *Org (design) pattern* specifies distinct, commonly occurring elements of service engagements.

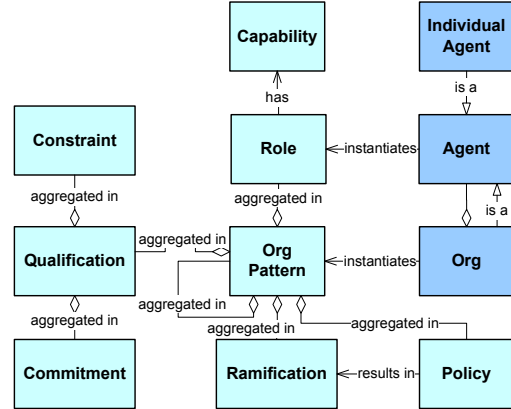


Figure 1. Org pattern model

A service engagement is designed by specifying two or more roles and applying one or more Org patterns on them. Figure 1 describes our Org pattern model. Syntactically, a *service engagement design* is just like an Org pattern, so for brevity we treat them alike. Hence, an Org pattern may recursively include Org patterns to allow for a service engagement to include one or more Org patterns. An Org pattern requires two or more roles, each with an associated capability. An Org pattern describes a set of *qualifications* that are constraints on the roles (their capabilities and commitments) for the pattern to be applied. Each pattern includes a set of *policy points* for each role, indicating where its policies apply. To participate in a service engagement, agents select and instantiate one or more roles specified in the corresponding service engagement design. The actual policies are authored when a service engagement is created. The policies are applied by agents playing the roles at runtime to decide on the actions needed to enact the service engagement. For each policy point, a pattern specifies the *ramifications* such as any resulting commitments when the corresponding decision is made.

2.1 Example Org Patterns

The patterns below use the following notations. A commitment $C_{DCr} : C(D, Cr, \phi, \overline{DCr})$ is a directed obligation from D (a debtor) to Cr (a creditor) to accomplish ϕ (a discharge condition), but arising within the scope of \overline{DCr} , a context Org. For agents A and B , their context Org is written as \overline{AB} . Cap_X is the set of capabilities of role X . $\phi \in Cap_X$ means that role X has the required capability to achieve ϕ . Res_{DCr} is the set of resources relevant to the commitment C_{DCr} .

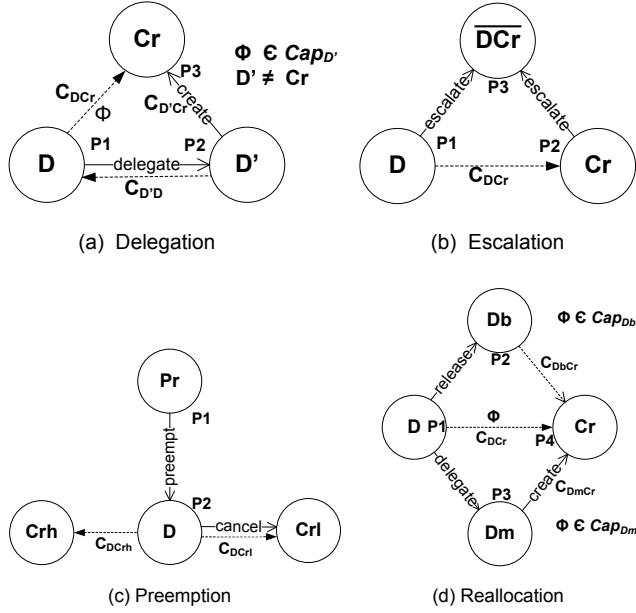


Figure 2. Example Org pattern scenarios

Figure 2 shows example Org pattern scenarios. Here, the circles are the roles; dotted edges with solid arrows indicate existing commitments. The solid edges with arrows indicate the pattern interaction in the given direction. The relevant policy points of the pattern are indicated next to the respective nodes. Other constraints are shown next to the nodes. These reflect *design-time* relationships among various roles as in an engagement specification, and may be enacted in multiple ways. We present the descriptions in English, but each of these can be expressed in XML or Jess (as in our prototype implementation).

Figure 2(a) illustrates the *delegation* pattern. When a delegated commitment fails to discharge and gets canceled, the creditor or the debtor of the commitment may escalate it to the context Org. This scenario is captured by the *escalation* pattern. For brevity we do not show this pattern in detail. Figure 2(b) illustrates the pattern *Escalation* (debtor D , creditor Cr), where D cancels the commitment. Here, D or Cr may send an escalate to \overline{DCr} . \overline{DCr} takes an action based on its escalate handling policy, either locally resolving it, or may forward the escalate up to its parent Org in the Org hierarchy.

Conflicting situations in service engagements require either *preemption* of some agents, or *reallocation* to handle conflicts. The example discussed in Section 1 demonstrates *preemptive scheduling* as a way to resolve conflicting situations. The *preemption* pattern captures this scenario.

For brevity we do not show other patterns in detail. Figure 2(d) illustrates the pattern *Reallocation*((D , Cr , Db , Dm)). Here, a delegatee (Db) that cancels

Name: Delegation (debtor D , creditor Cr , delegatee D')
Purpose and applicability: The debtor of a commitment delegates it to a delegatee who may accept the delegation, thus creating a new commitment with the delegatee as the new debtor
Qualifications: $C_{DCr} : C(D, Cr, \overline{DCr}, \phi)$: the commitment to be delegated exists $C_{D'D} : C(D', D, \overline{DD'}, \phi')$: D' is committed to D to accept (ϕ') the delegations of commitments from D Cn1 : $\phi \in Cap_{D'}$: the delegatee has the required capability to satisfy the commitment Cn2 : $D' \neq Cr$: the delegatee is not the creditor
Scenarios: In Figure 2(a), D sends a <i>delegate request</i> to D' , D' <i>accepts</i> the delegate and <i>creates</i> a new commitment $C_{D'Cr}$ within a newly created context Org $\overline{D'Cr}$
Policies and Ramifications: P1 (D 's policy point): D <i>delegates</i> C_{DCr} to D' <i>Ramification</i> : D grants access of Res_{DCr} to D' P2 (D' 's policy point): D' <i>accepts</i> the <i>delegate</i> from D <i>Ramification</i> : $C_{D'Cr} : C(D', Cr, \overline{D'Cr}, \phi)$ is created P3 (Cr 's policy): Cr <i>accepts</i> the newly created commitment. Otherwise, the new commitment is <i>released</i>
Known uses: A merchant delegates shipping a product to a shipper

a delegated commitment is released and the original commitment is redelegated to another agent (Dm). Here, a cancel is implied on timeouts (when a violation occurs). This applies when an agent is involved in multiple contracts simultaneously. For example, in the hospital scenario, when Dr. Smith scheduled to attend to patient A in a hospital needs to attend to another (critical) patient B , the hospital may reallocate Dr. Jones in place of Dr. Smith to see patient A . **Assignment**(D , Cr , Cr') corresponds to a commitment C_{DCr} being assigned by Cr to Cr' .

2.2 An Example Service Engagement

A service engagement is specified using the above patterns. Each engagement specification follows the template of an Org pattern. Here, the qualifications are assertions corresponding to the patterns applied on appropriate roles. All policy points and ramifications are inherited from the patterns. A service engagement specification not only states the patterns but also specifies a set of *key services* captured as commitments and other constraints. A service engagement is instantiated when agents adopt the relevant roles and provided all qualifications are met. The agents supply

Name: Preemption (D, Crl, Crh, Pr) Here, Crl is a lower priority creditor than Crh , and Pr is the preempting role
Purpose and applicability: To cancel a commitment based on conflicting demands
Qualifications: $C_{DCrl} : C(D, Crl, \overline{DCrl}, \phi_1)$: the lower priority commitment of D exists $C_{DCrh} : C(D, Crh, \overline{DCrh}, \phi_2)$: the higher priority commitment of D exists Cn1 : Pr has set a higher priority for C_{DCrh} than C_{DCrl}
Scenarios: In Figure 2(c), D is unable to discharge both the commitments C_{DCrl} and C_{DCrh} simultaneously. D is preempted by Pr from the commitment C_{DCrl} . Pr would have either directly received an escalation from D , or Pr would be a member of a context Org who received the escalation. Sometimes, Pr would be the delegator, who preempts the delegatee D from a failing commitment.
Policies and Ramifications: P1 (Pr 's policy point): Pr sends a <i>preemption</i> request to D to cancel C_{DCrl} P2 (D 's policy point): D receives a <i>preemption</i> request Ramification : P1 and P2 result in C_{DCrl} being canceled by D . This may in turn lead to new escalations of C_{DCrl} , which will require a new debtor
Known Uses: A physician is preempted from his consultation hours to attend to a medical emergency

policies for each policy point and are subject to the stated ramifications.

The following formalize the NOAA–NL service engagement using the above approach.

Roles and Role Instantiations: *Service-Providers-Org* (SPO), *Service-Provider* (SP), *Service-Consumer-Org* (SCO), *Service-Consumer* (SC). These roles are instantiated by the agents as follows: National Labs (SPO), ANL (SP), NOAA (SCO), NHC (SC).

Key Services as Commitments: Commitment C_3 : $C(SP, SC, Org-SP-SC, \phi)$, where ϕ means the compute service is provisioned successfully. Commitment C_1 is first created that captures the engagement between SPO and SCO within Org $Org-SPO-SCO$. SCO assigns C_1 to SC to create an assigned commitment C_2 . C_2 is now delegated by SPO to SP to create C_3 .

Qualifications: These include the delegation, assign, escalation, and the preemption patterns described below:

Assign (SPO, SCO, SC) : SCO assigns C_1 to SC . After the assign, SPO makes a new commitment C_2 to SC within

$Org-SPO-SC$.

Delegation (SPO, SC, SP) : Commitment C_2 from SPO to SC is delegated to SP . A new commitment C_3 is created with SP as the new debtor.

Escalation ($SP, SC, Org-SP-SC$) : Commitment C_3 from SP to SC exists within $Org-SP-SC$. SP or SC can send escalates to the context, $Org-SP-SC$.

Preemption (SP, SCL, SC, SPO) : Here, SCL is the service consumer, who is the creditor of the lower priority commitment C_4 , and SC is the creditor of the higher-priority commitment C_3 . SPO is the preempting Org that preempts SP from the lower priority commitment.

3 Discussion

The services sector has long been dominant in developed economies. As services technologies and business models spread in IT, an increasing number of IT applications are taking on a services flavor. Witness the expansion of utility or autonomic computing and software as a service. Services have a natural match with policy techniques. Besides the usual challenges of policy languages and engines, services also throw up the important challenge of setting up interactions among multiple stakeholders so they can carry out a service engagement in a manner that is easy to design and configure, and yet respects their autonomy.

With this motivation, we developed a set of design patterns for specifying service engagements. This set is far from complete. Nor do we believe that a small complete set of patterns exists. A programming language with only a few constructs can be Turing complete, yet books have been filled with patterns of programming. In the same vein, because service engagements exhibit enormous diversity, we expect that more and more design patterns for engagements will be identified.

We have implemented a prototype using a policy engine based on Jess and messaging middleware that demonstrates the policy-based enactment, including the above scenarios. This paper opens up important future directions in the field of distributed policies and service computing. Specifying design patterns for handling more subtle conflicts among service agreements is left to future work.

References

- [1] Y. B. Udupi and M. P. Singh. Contract enactment in virtual organizations: A commitment-based approach. In *Proc. of 21st Natnl. Conf. on Artifl. Intel. (AAAI)*, pages 722–727. 2006.
- [2] Y. B. Udupi and M. P. Singh. Governance of cross-organizational service agreements: A policy-based approach. In *Proc. of the IEEE Intl. Conf. on Services Comp. (SCC)*, pages 36–43, July 2007.