SOA Governance: Exploring Challenges & Benefits from an Autonomic Perspective

J. A. Parejo, Pablo Fernandez, and Antonio Ruiz-Cortés

University of Seville, Spain
japarejo@us.es

Abstract. Both Academy and Industry agree in the importance of having an adequate management of the Service Oriented Architecture (SOA) to adapt and scale to meet the evolving needs of the organization. In order to face this problem, SOA Governance is defined as the set of policies and principles that manage the operations related with the SOA and allow an appropriate evolution aligned with the business goals of the organization. These are both human-oriented and infrastructure-oriented and can be applied within the overall life-cycle of the service: from design-time to run-time. Currently, some seminal works have been proposed to create a reference model for SOA Governance and some infrastructures of Governance have been proposed; current approaches rely on human-oriented tasks in governance without advanced autonomic behaviors. This paper shows SOA Governance automation as a challenge in the autonomic computing area, and analyzes how the different self-* properties of autonomic systems could be applied to this context, identifying desirable capabilities and open issues.

1 Introduction

Service Oriented Architecture (SOA) adoption by organizations is growing fast [1]. However, it inherently brings a various orders of magnitude increase on the number components of the architecture and their dependences, and consequently a dramatic growth of the effort needed to properly manage and control it [2]. In this context, SOA Governance is defined as the management process to deliver the SOA promise of effective reuse, business goals support and change responsiveness [3] [4]. Consequently, both industry and academia have identified recently SOA Governance as a hot spot [4][5].

Autonomic Computing is defined as system self-management to overcome the growing complexity of current systems as they evolve and scale [6]. In this context, the essence of Autonomic Computing is imbricated on the SOA Governance framework and its unavoidable need of automation [7]. Additionally, in a SOA, complexity and management needs come not only from computing systems, but also from the organizational structure, business processes and goals. As a consequence, a proper SOA Governance Model should seamlessly incorporate people management, organizational management, processes management and IT management.

In doing so, the vision of services as business entities, would drive to a business dimension the meaning of autonomic properties.

In this paper we explore the relation between SOA Governance and Autonomic Computing, showing how principles and properties developed for autonomic systems can support SOA Governance. This leads to the creation of Autonomic SOA Governance Infrastructures (ASGI). In particular, we present a set of governance requirements, and target different self-* properties showing how they would support those requirements for effective SOA Governance.

This paper contributions are:

1. The identification of set of governance requirements (desirable behaviors) that could be fullfilled by an ASGI (governance infrastructure with autonomic capabilities).
2. A conceptual model of SOA Governance extending previous ones where we highlight elements that could implement autonomic properties.

The rest of the paper is structured as follows: In the next section we describe a general SOA Governance Model for Organizations in order to clarify responsibilities and elements that an ASGI should face, highlighting the elements that could be target of autonomic computing techniques and properties. In section 3, we apply the autonomic properties to SOA Governance focusing on policies, and providing insights on the capabilities and challenges they represent. In section 4 we describe the related work. Finally, in the last section we summarize and enumerate the open issues and future work of our research.

2 SOA Governance Model

SOA Governance is a subset of the whole organization governance framework that accounts for the specific characteristics of Service Oriented Architectures, such as increased number of IT components and dependencies, and focus on business alignment. In this context, SOA Governance is defined as the management process to deliver the SOA promise of effective reuse, business goals support and change responsiveness [3] [4].

According to this definition, SOA Governance sustains and extends business strategy and goals, supporting the alignment of IT and business that SOA proposes. In so doing, the SOA Governance framework must be lead by a SOA governance strategy, that comprise governance goals (aligned with organization strategy and supporting business goals); e.g. governance goal: "*service reuse on new applications should be at least 30%*" that supports the general business goals of "*IT cost reduction*" and "*time to market reduction of new products*". Those governance goals must be defined in quantifiable form, allowing us to measure to what extent we are achieving them. Furthermore, SOA governance goals generate governance principles (general rules to achieve governance goals)[3] [8], that will be expressed and enacted as a set of governance policies. Those elements bridge the gap between the conceptual governance framework, that describes what should be done and descriptions of how to do it at a conceptual level (comprising governance strategy, goals and principles) and the effective governance system, that describes in detail how do things and how we control that things are done accordingly (comprising policies and processes) [3]. The governance model, then, specifies the processes, polices, and mechanisms required to monitor the SOA throughout the SOA lifecycle [9, 3]. The governance model also provides the organizational structure that defines the roles and responsibilities needed to operate the governance model and to ensure that the SOA is successful [8]. In figure 1 a global view of our proposed global governance model is shown. This model is based on the OASIS Reference Model for SOA [10] and on those proposed in [3], [9] and [8].

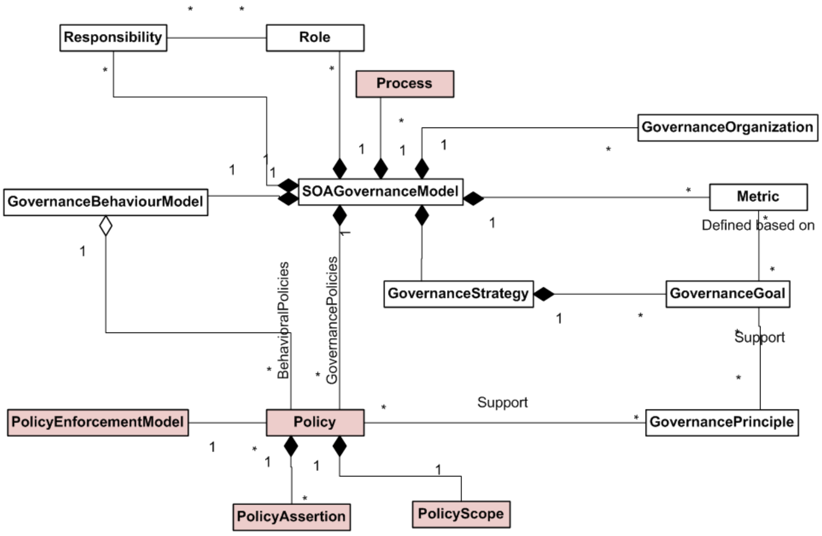

Fig. 1. Conceptual SOA Governance Model - Global View

Effective SOA Governance comprises definition and enforcement of policies and processes. Consequently, in figure 1 processes and policies are highlighted as the "active elements" that support autonomic properties on SOA Governance Infrastructures. Based on service life-cycle, we can subdivide those policies and processes in two groups. On the one hand, design time governance involves service identification, definition, creation and reuse along the organization. On the other hand, runtime governance which comprises management of the rest of the service life-cycle (i.e. deployment, consumption, versioning & change and retirement). SOA Governance automation and specifically automated policy enforcement is a challenge [7] [9] that would bring significant benefits, such as cost reduction and improved responsiveness. Furthermore automation is a must when policies express requirements for services that should be enforced -or provisionedin each invocation or when the issue is critical for the organization -e.g. service security policies-.

A governance policy represents a capability, requirement or behavior that allows the SOA to achieve their goals, and whose meeting can be evaluated. Policies have a *scope* s, and a finite set $a \neq \oslash$ of policy assertions $\{a_1, a_2, \ldots, a_n\}$, that are predicates to be met (by all the policy subjects specified by the scope).

Policy scope defines the elements for which the policy will be applied, i. e. *policy subjects*; examples of policy scopes can be: *All services*, *services provided by department X* or *services used by Applicacion/bussiness process Y*. Governance policies specify rules that SOA parties must adhere to, consequently there are different types of policies depending on the purpose of the rule and the nature of the scope: business and corporate policies,(people) behavioral policies, process policies, technical policies, quality of service (QoS) policies, testing policies, etc.

Governance processes are specifications of actions that when executed contribute to achieve the goals of SOA governance.

In figure 2 a more detailed view of the structure and relations of processes and policies is shown. Policy structure is based on the WS-Policy standard elements [11] (incorporating attachments), but with some added elements to support a richer runtime behavior, such as the*PolicyEnforcementModel* (described in [3]), responsible of assuring that policy assertions met. Enforcement models could be manual, assisted by technology or fully automated. One of the most potentially fruitful areas of governance automation is specifically the creation of autonomic infrastructures that implement automatic enforcement and monitoring models. The heterogeneity of policies to enforce (as described above) makes this automation one difficult challenge.

One interesting point of figure 2 is that policy is a subtype of *PolicySubject*, in this way we can create meta-policies [12], [13] (i.e. policies about policies; e.g. "*authorized business managers may change a policy threshold value up to 10% in either direction without requiring IT approval, but any changes greater than 10% must go through a particular IT manager first*" [14]).

It is important at this point to highlight that design-time activities for policies is associated with runtime activities of meta-policies (i.e. meta-policies should be enforced at policy design time).Consequently, autonomic activities associated with policy enforcement (like monitoring, analysis, etc.) that are usually executed at runtime for usual policies, are executed at design time for meta-policies leading to autonomic governance infrastructures that handle design-time governance.

2.1 Achieving autonomic behaviors on the SOA

The management process that support SOA governance drives the behavior of the elements to assure conformance to the governance model, specifically with policies meeting. In this sense, the whole governance process is a self-protection process, to assure SOA success through management.

In the context of Autonomic Computing, policies have been identified as key elements for human-to-autonomic-system interaction [6]. Consequently, they are

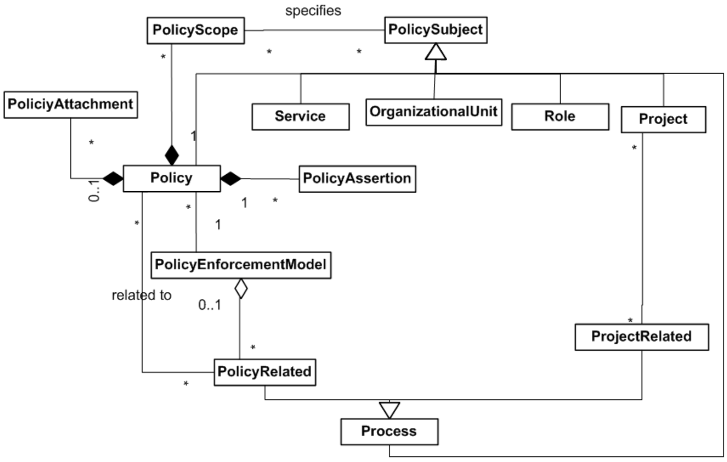

Fig. 2. Conceptual SOA Governance Model - Policies & Processes

used in some architectural descriptions as inputs of the higher level of autonomic management system [15], where different autonomic controllers are orchestrated to support an adequate management and balance of the different autonomic-goals and associated actions.

Accordingly, most SOA Governance policies are declarative expressions of an autonomic goal, that guide the behavior of the underlying infrastructure (and humans interacting on the SOA) to self-protect, self-configure, self-optimize and self-heal the different elements of their scope. In table 2.1 we provide some governance policies that encode autonomic objectives.

Table 1. Governance policies that promote and drive autonomic behaviors

Example governance policy	Autonomic Property/Objective
”*Different versions of a service will have different endpoints and namespace* (automatically assigned and managed based on the version)”	Self-Configuration
”*Prior to create a new service request users should search service registry for services with similar functional properties*”	Self-Optimization
”*If a critical stateless service endpoint is failing, the service should be automatically re-deployed on a different server and a routing rule should be created on the Enterprise Service Bus (ESB) to drive invocations to the new endpoint*”	Self-Healing
”*External service invocations must be authenticated and encrypted using an X.509 certificate*”	Self-Protection

Furthermore, an proper management of governance policies needs by itself an infrastructure that exhibit autonomic properties as we show in the next section.

3 Self-* Properties applied to SOA Governance Infrastructures

The general capabilities of an autonomic (self-managing) system can be summarized as four objectives: self-configuring, self-optimizing, self-healing and self protecting; and four properties: self-awareness, environment-awareness, self-monitoring and self-adjusting [6]. However those properties and objectives are means to give -an autonomic- response to specific needs and usually complex tasks that must be performed on a system daily operation and supporting activities. In this section we link self-* properties and goals to the SOA Governance activities. In doing so, we try to give a response to the following question: Where should an ASGI exhibit those features? or more specifically: Which SOA Governance tasks should be automated?.

Giving a response to this question from a business perspective is fairly simple, all activities that could be automated should be automated to provide maximum benefit and performance. However, providing a coherent approach to the automation of SOA Governance tasks is difficult since the high heterogeneity of tasks to be performed and elements to manage (see [3, 8] for a detailed discussion). Furthermore, from a realistic -more IT related- perspective we should select a set of features that will provide the biggest business and operational benefits with the minimum development and integration effort.

We will focus on SOA Governance policies activities, since its support of the effective governance and wider potential application range.

Concerning to Self-Configuration, we consider governance policies scope updating a design time autonomic property; i.e. the policy design system (usually and Integrated Development Environment or web administration application) should connect to the service registry, and provide help to specify the scope of the currently editing policy.

We consider general governance policies enforcement a runtime autonomic property because of the need of monitoring, and possibly managing and configure the underlying infrastructure of the elements of the scope. Finally, service redeployment and enactment is considered because of the evident need of self-configuration to implement it automatically.

Concerning to Self-Optimization, we consider governance policies simplification and redundance detection a design time self-optimization activity, given that it allows the SOA governance infrastructure perform better, having to manage a smaller number of simpler policies. Moreover, we consider run-time service redeployment and enactment a self-optimization and self-healing activity. On the one hand, when service re-deployment is performed at run-time on a different infrastructure in order to obtain better performance, we consider it a self optimization. On the other hand, it is considered a self-healing activity when is performed on response to a service failure as a form of micro-rebooting [16].

Table 2. Autonomic objectives/properties applied to SOA governance policies management

Autonomic Property/Objective	Design-Time Governance	Run-Time Governance
Self-Configuration	Governance Policies Scope updating	Governance Policies Enforcement Service Redeployment and enactment
Self-Optimization	Governance Policies Simplification and Redundance Detection	Service Redeployment and enactment
Self-Healing	Governance Policies Consistency Checking	Governance Policies Conformance & Violation treatment Service Redeployment and autonomic enactment
Self-Protection	Autonomic Meta-Policies Generation	Governance Policies Monitoring

Concerning to Self-Healing, we consider governance policies consistency checking a self-healing activity, given that it allows to detect and disable inconsistent policies fixing the policies database (it could be considered also a self-protection activity). Moreover, we runtime governance policy violation treatment a self-healing activity, given that they try to compensate the effects of policy violation; e.g. the execution of compensation actions specified in QoS policies such as "*If service X response time becomes bigger than Y, deploy a new instance on a different server and balance invocations between corresponding endpoints*".

Concerning to Self-Protection, we consider autonomic meta-policies generation (based on SOA governance policies monitoring) a self-protection activity, given that meta-policies regulate the design and changes on actual policies; e.g. "*authorized business managers may change a policy threshold value up to 10% in either direction without requiring IT approval, but any changes greater than 10% must go through a particular IT manager first*". Moreover, we consider governance policies monitoring a self-protection activity, given that they provide means to detect (and potentially avoid through provisioning and enforcement) violation of policies.

It is important to note that our table is not complete nor exhaustive. Consequently, policy related activities can be added as were identified as autonomic activities, and our answer to the original question is only a partial one.

Finally, focusing on governance policies life-cycle the desirable behavior of an ASGI would be:

- At design-time: The set of defined meta-policies are enforced (usually performing *self-protection* tasks, such as consistency checking of the policy; e.g. "$a \wedge \neg a$" is not a consistent policy assertion). The system assist on the

specification of the policy scope and assertions (achieving a minimal *self-configuration*).
- At deployment-time: Consistency checking of the assertion on deployment with the whole set of previously stated policies is performed (*self-protection*) and the policy is analyzed and simplified to assure that it is not redundant (*self-optimization*). Moreover, scope is evaluated warning if there are not currently elements on it (*self-healing*).
- At run-time: Policy meeting is monitored and enforced (possibly performing *self-configuration*, *self-optimization* or *self-healing* activities depending on the scope and kind of the policy).

4 Related Work

Both Autonomic Computing and SOA are both flourishing areas in the computer science research landscape. However, combination of both research lines has been a subject of little research effort. Proposals in this context mainly provide ways to implement services [17] or processes [18] with some autonomic capabilities like self-healing [19] and self-configuration [20].

Furthermore, Autonomic SOA Governance is a topic nearly unexplored - with exceptions such as [7]-. In [7] the application of autonomic architectural patterns for managed resources [21, 15, 22] is proposed to support web services management. However, this proposal is strictly focused on web services and life-cycle management, without taking into account policies and the rest of elements of a complete governance model.

Policy-based management has been subject of extensive research in its own right. In the context of autonomic computing, policies have been proposed as high level guidance tools for humans to allow self-management [23], becoming relevant elements in this context [6]. Concerning to autonomic policy enforcement, in [24] policies are presented as means to specify general goals and desired behaviors to the autonomic system that are translated to different levels in the system in order to achieve self-management, and an agent based approach for implementation is proposed. In [25] an SLA based QoS enforcement architecture is proposed, and in [26] this proposal and a SLA management infrastructure are used to support SOA governance policies enforcement for some kinds of policies. This approach provides a mechanism to support some of the capabilities of table 3, specifically governance policies scope updating, governance policies consistency checking and governance policies enforcement.

5 Conclusions

In this paper we propose a conceptual model for SOA Governance. Additionally, we identify important requirements and challenges, such as governance policy enforcement and conformance testing respect to the governance model. Finally, we have shown how different governance policy related activities to automate relate to autonomic goals.

Currently our research group is working on different ways to support autonomic SOA governance as presented in this paper: (i) A governance policy enforcement mechanism based on QoS and SLA management infrastructures is under development with some initial results [25][26]. (ii) An automatic service protocol adaption engine with QoS-aware capabilities for an open source JBI compliant ESB [27] is on design phase. (iii) A policies management and enforcement model and infrastructure supported by (i) that extends the approach presented in [24] based on multi-agent systems is conceptualized.

Acknowledgments This work has been partially supported by the European Commission (FEDER) and Spanish Government under CICYT projects Web-Factories (TIN2006-00472) and SETI (TIN2009-07366), and Andalusian Government project ISABEL (TIC-2533).

References

1. Heffner, R., Leganza, G., Ranade, K.: Soa adoption: Many firms got started in 2007. Technical report, Forrester Research (2008)
2. Simmons, S.: Soa governance and the prevention of service-oriented anarchy. IBM WebSphere Developer Technical Journal. IBM DeveloperWorks (September 2006)
3. Marks, E.A.: Service-Oriented Architecture Governance for the Services Driven Enterprise. John Wiley & Sons (2008)
4. Papazoglou, M.P., Traverso, P., Dustdar, S., Leymann, F.: Service-oriented computing: State of the art and research challenges. IEEE Computer **40**(11) (November 2007) 38–45
5. Kenney, L.F., Plummer, D.C.: Magic quadrant for integrated soa governance technology sets. Technical report, Gartner RAS Core Research (March 2009) Available at http://mediaproducts.gartner.com/reprints/oracle/article65/article65.html.
6. Sterritt, R.: Autonomic computing. Innovations in Systems and Software Engineering **1**(1) (April 2005) 79–88
7. Papazoglou, M., van den Heuvel, W.J.: Web services management: a survey. Internet Computing, IEEE **9**(6) (Nov.-Dec. 2005) 58–64
8. Brown, W.A., Laird, R.G., Gee, C., Mitra, T.: SOA Governance: Achieving and Sustaining Business and IT Agility. IBM Press (December 2008)
9. Bernhardt, J., Seese, D.: A conceptual framework for the governance of service-oriented architectures. In: Service-Oriented Computing ICSOC 2008 Workshops. Springer (2009) 327–338
10. MacKenzie, C.M., Laskey, K., McCabe, F., Brown, P.F., Metz, R.: Reference model for service oriented architecture 1.0 (August 2006)
11. Vedamuthu, A.S., Orchard, D., Hirsch, F., Hondo, M., Yendluri, P., Boubez, T., mit Yalinalp: Web services policy 1.5. W3C Recommendation (September 2007) Available online at http://www.w3.org/TR/2007/REC-ws-policy-20070904/.
12. Hosmer, H.H.: Metapolicies i. SIGSAC Rev. **10**(2-3) (1992) 18–43
13. Quinn, K., Lewis, D., O'Sullivan, D., Wade, V.: Trust meta-policies for flexible and dynamic policy based trust management. In: Policies for Distributed Systems and Networks, 2006. Policy 2006. Seventh IEEE International Workshop on. (June 2006) 4 pp.–148

14. Bloomberg, J.: Soa governance and the butterfly effect. Electronic article (2006) Available online at: http://zapthink.com/report.html?id=ZAPFLASH-2006124.
15. White, S., Hanson, J., Whalley, I., Chess, D., Kephart, J.: An architectural approach to autonomic computing. In: Autonomic Computing, 2004. Proceedings. International Conference on. (May 2004) 2–9
16. Candea, G., Kawamoto, S., Fujiki, Y., Friedman, G., Fox, A.: Microreboot - a technique for cheap recovery. In: OSDI. (2004) 31–44
17. Zeid, A., Gurguis, S.: Towards autonomic web services. In: Computer Systems and Applications, 2005. The 3rd ACS/IEEE International Conference on. (2005) 69–
18. Verma, K., Sheth, A.: Autonomic web processes (2005)
19. Gurguis, S.A., Zeid, A.: Towards autonomic web services: achieving self-healing using web services. SIGSOFT Softw. Eng. Notes **30**(4) (2005) 1–5
20. Haas, R., Droz, P., Stiller, B.: Autonomic service deployment in networks. IBM Syst. J. **42**(1) (2003) 150–164
21. Computing, I.A.: An architectural blueprint for autonomic computing. White paper, IBM (2004)
22. Fuad, M.M., Oudshoorn, M.J.: System architecture of an autonomic element. In: Engineering of Autonomic and Autonomous Systems, 2007. EASe '07. Fourth IEEE International Workshop on. (March 2007) 89–93
23. Kephart, J., Walsh, W.: An artificial intelligence perspective on autonomic computing policies. In: Policies for Distributed Systems and Networks, 2004. POLICY 2004. Proceedings. Fifth IEEE International Workshop on. (June 2004) 3–12
24. Peña, J., Hinchey, M.G., Sterritt, R., Cortés, A.R.: Building and implementing policies in autonomous and autonomic systems using macmas. ISSE **3**(1) (2007) 17–31
25. Parejo, J.A., Fernández, P., Ruiz-Cortés, A., García, J.M.: Slaws: Towards a conceptual architecture for sla enforcement. In: 2008 IEEE Congress on Services - Part I (International Workshop on Methodologies for Non-Functional Properties in Services Computing, MNPSC), Honolulu, HI, IEEE Computer Society Press (2008) 322–328
26. Parejo, J.A., Fernandez, P., Ruiz-Cortés, A.: Towards automated sla-based governance policy enforcement. In: Submitted to the 7th International Joint Conference on Service Oriented Computing (ICSOC). (2009)
27. Ten-Hove, R., Walker, P.: Java business integration (jbi) 1.0 specification (August 2005)