City Research Online

City St George's, University of London

Citation: Foster, H. & Spanoudakis, G. (2011). SMaRT: A Workbench for reporting the monitorability of services from SLAs. In: Proceedings of the 3rd International Workshop on Principles of Engineering Service-Oriented Systems. (pp. 36-42). New York; USA: ACM. ISBN 978-1-4503-0591-4 doi: 10.1145/1985394.1985400

This is the accepted version of the paper.

This version of the publication may differ from the final published version. To cite this item please consult the publisher's version.

Permanent repository link: https://openaccess.city.ac.uk/id/eprint/4666/

Link to published version: https://doi.org/10.1145/1985394.1985400

Copyright and Reuse: Copyright and Moral Rights remain with the author(s) and/or copyright holders. Copies of full items can be used for personal research or study, educational, or not-for-profit purposes without prior permission or charge, unless otherwise indicated, provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way. For full details of reuse please refer to City Research Online policy.

City Research Online: http://openaccess.city.ac.uk/ publications@citystgeorges.ac.uk

SMaRT: A Workbench for Reporting the Monitorability of Services from SLAs

Howard Foster and George Spanoudakis
Department of Computing, City University London,
Northampton Square, EC1V 0HB, London,
England, United Kingdom
{howard.foster.1,g.e.spanoudakis}@city.ac.uk

ABSTRACT

Service Level Agreements (SLAs) for Software Services aim to clearly identify the service level commitments established between service requesters and providers. A dynamic configuration for the monitoring of these SLAs provides the opportunity for service monitor providers to offer and release monitoring infrastructures for different types of services. Whilst there has been work on automating this monitor matching and configuration, additional support may be needed in the negotiation and provision of monitors for which the current monitoring infrastructure does not provide suitable SLA term monitors. In this paper we describe an approach to effectively report and assist service monitoring support groups in managing this provision. The approach described is illustrated with mechanical support in the form of a *SMaRT Workbench* Eclipse IDE plug-in for reporting on the monitorability of SLAs for service monitoring infrastructures.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.5 [**Testing and Debugging**]: [Monitors]

General Terms

Design, Experimentation, Management

Keywords

Run-time SLA Monitoring, Monitor- ing Capabilities, Monitorability Assessment

1. INTRODUCTION

As a key part of monitoring and management, systems developed with the Service-Oriented Architecture (SOA) design pattern should utilise negotiated agreements between service providers and requesters. Typically the result of this negotiation is specified in Service-Level Agreements (SLAs), which are then used to monitor key levels of service provided and to optionally specify pre-conditions and actions to take in the event of such levels being violated. The dynamic availability and allocation of responsibility for monitoring SLAs (and often individual parts within them) to different monitoring components is necessary as both SLAs and the components available for monitoring them may change dynamically during the operation of a service based system [2]. The complexity of SLA terms however, means that several monitoring components may need to be selected for a single SLA guaranteed term expression (e.g. availability $> 90\%$) since each part of the expression may be reasoned by a physically different monitor providers. Our existing work [4, 5] focused on the automated configuration of SLA term monitoring with available monitoring infrastructure services, leaving the reporting of unsupported SLA terms for future work. Such reporting is useful in the case of negotiations and identifying new monitoring requirements as SLA requirements change. Our reporting is aimed at those groups who are responsible to support these monitoring infrastructures and capabilities.

In this paper we show how this reporting is achieved from the configuration activities of matching complex service agreement terms, which are decomposed in to manageable monitoring configurations. In addition, our configuration approach also includes a mechanism to support preferred monitoring component selection and reporting. End-user monitorability reporting on the supported or non-supported configurations is provided by a series of components which mechanically parse SLAs, generates a formal SLA Abstract Syntax Tree (AST) and decomposes the terms of the AST in to expressions for monitoring. Each expression is then used to select appropriate service reasoning or sensor monitoring components. If appropriate reasoner or service sensor components cannot be provided for a term or other part of the SLA, then this is flagged in a monitorability report. Note that it is assumed that partial monitoring of the SLA (containing multiple service agreements) is still acceptable (if it is deemed a valid negotiation between service consumer and provider). Therefore, the main contribution of this work is that of producing a clear and easily accessible view of a monitorablility report to aid the negotiation and capability assessment of monitoring infrastructure support. The approach to this reporting is provided mechanically in the form of an Eclipse Integrated Development Environment (IDE) plug-in known as the *SMaRT Workbench*.

The paper is structured as follows. In section 2 we discuss a background scenario to the monitorability reporting,

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
PESOS '11, May 23-24 2011, Honolulu, Hawaii, USA
Copyright 20XX ACM X-XXXXX-XX-X/XX/XX ...$10.00.

whilst in section 3 we briefly outline a monitoring architecture using monitoring configurations. In section 4 we describe the monitorability assessment activities and in section 5 the monitorability workbench. In section 6 we discuss related work and conclude the paper in Section 7 with a discussion of present and future work.

2. BACKGROUND

In our previous work we have focused on providing a mechanism to support dynamic configuration of monitoring infrastructures [4, 5] by way of deciding service providers from their match in providing terms and attributes for service consumer configuration preferences. Mechanical support was provided for this configuration in a wider service-level agreement focused architecture (as used in the EU SLA@SOI project [13]), which included the business modelling of such SLAs and predicting violations of such SLAs. In this work a key assumption was that monitoring negotiation and enactment is automated and requires no human intervention. However, just as with any other business or technical services environment there are scenarios where human interactions for service monitoring (and individual monitors) is also required. For example, in large enterprise organisations, the service monitors will typically be deployed on to hosted environments and *managed* by a separate support group than the service monitoring providers. Therefore there is also high value in alternative service monitoring scenarios, where negotiations for service monitoring will include service support groups, for example, liaising with new and existing service monitor providers for new monitoring requirements.

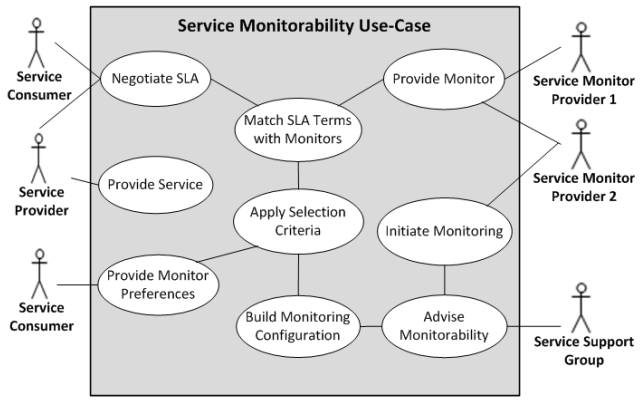

Figure 1: A Monitorability Reporting Scenario

Our scenario for this support group side of service monitoring, focuses on monitorability reporting as illustrated in the scenario of Figure 1. Service consumers and providers initially negotiate a service SLA, where constraints are specified for various functional and non-functional Quality of Service (QoS). Additionally, service monitor providers (of which there may be many for the same type of service) supply monitoring services (as capabilities) offering monitoring components. Note that in this scenario we assume that independent monitor providers are supplying the monitors although it is equally feasible that the service providers themselves are also monitor providers. For each constraint specified in the SLA a set of monitors are selected which match the constraints specified for the service in the SLA. Additionally, the service consumer can also specify monitor and provider appropriateness properties (such as preferred provider or geographical location) which are then used to locate the most appropriate monitors for monitoring. If an SLA can be monitored a Monitoring System Configuration (MSC) is produced which configures and initiates monitoring infrastructure as part of a services architecture.

Reporting on monitoring capability is the focus of this work and therefore we assume that there is some service support group which can advise consumers (or providers) on the monitorability of services in the SLA. We noticed that there are several particularly interesting angles to this scenario from the perspective of offering such support. First, even though an SLA may be negotiated and agreed based upon a set of common SLA terms, the necessary monitoring infrastructure may not be readily available to reason on decomposed term evaluations or on other monitoring preferences. As we described in the introduction, SLAs can become very complex and evaluating and configuring all monitoring components within the infrastructure can become equally or more complex.

Second, given matching reports on particular monitoring requirements from the SLA, and in particular noting elements in the report that are raised for unsupported terms and monitors, a service monitoring support group may wish to engage with new providers that can support such requirements. This then cycles an iterative support process as SLA requirements change and new providers are required to meet the new SLA negotiations. Therefore, negotation in this case is not only agreeing on what can be achieved given the existing monitoring infrastructure capabilities but also what needs to be.

In section 3 we briefly describe a service monitoring infrastructure architecture (from the EC SLA@SOI project [3]) and the Monitoring Features used for matching terms and monitors. The results of the configuration activities involved in matching forms the main input for reporting *monitoring capability* to service support groups.

3. SERVICE MONITORING

An overview architecture for Service Monitoring in the EC SLA@SOA project is illustrated in Figure 2. The Planning and Optimization Component (POC) is a local executive controller for a ServiceManager. It is responsible for assessing and customizing SLA offers, evaluating available service implementations and planning optimal service provisioning and monitoring strategies. The POC generates a suitable execution plan for monitoring (based upon a configuration obtained from the MonitoringManager component) and passes this to the Provisioning and Adjustment Component (PAC). The PAC collects information from the Low Level Monitoring System, analyses the incoming events and decides if a problem has occurred or it is about to occur, identifies the root cause and if possible decides and triggers the best corrective or proactive action. In case the problem cannot be solved at a local level, the PAC escalates the issue to a higher level component, namely the POC. In case of an SLA violation, adjustment can trigger re-planning, re-configuration and/or alerting to higher-level SLA monitoring. These capabilities are considered to be important in order to assure preserving service provision and resource quality.

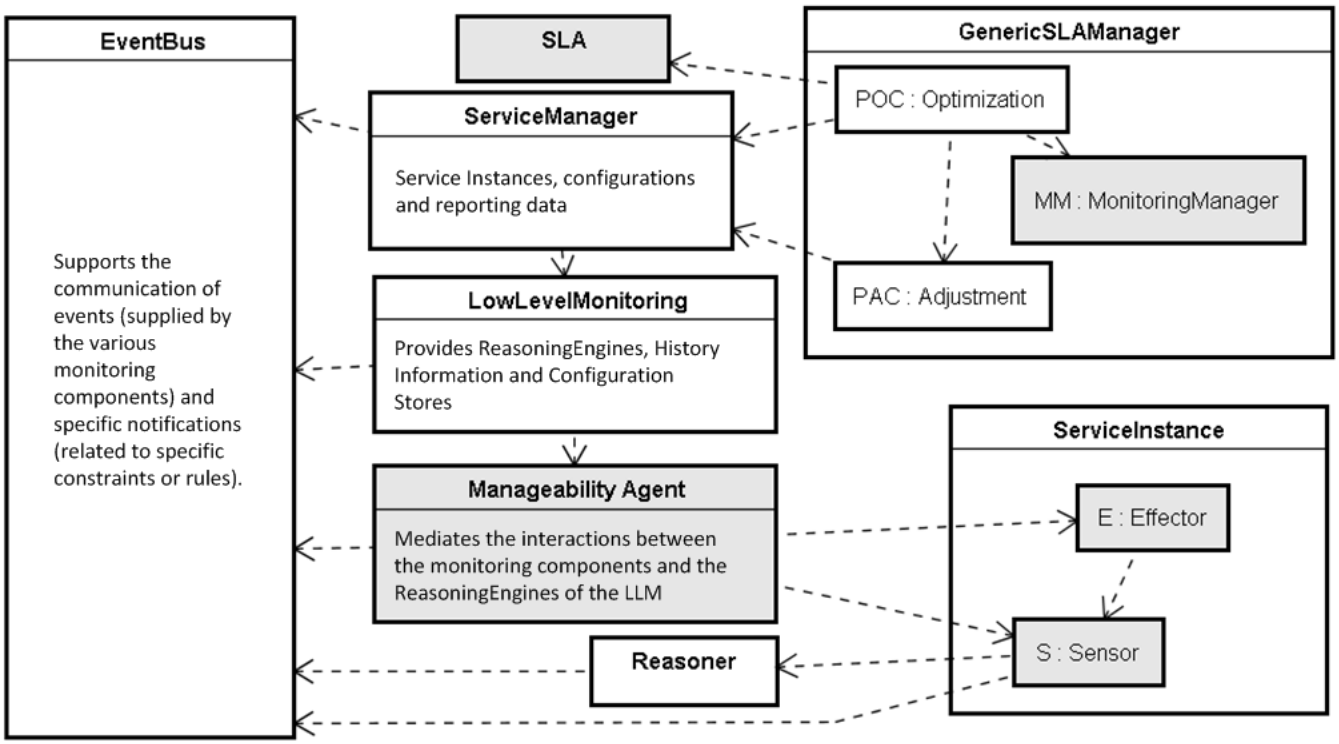

Figure 2: A Service Monitoring Architecture

The MonitoringManager (MM) coordinates the generation of a monitoring configuration of the system. It decides, for an SLA specification instance it receives, which is the most appropriate monitoring configuration according to a configurable selection criteria. A monitoring configuration describes which components to configure and how their configurations can be used to obtain results of monitoring Guarantee States. The Low Level Monitoring Manager is a central entity for storing and processing monitoring data. It collects raw observations, processes them, computes derived metrics, evaluates the rules, stores the history and offers all this data to other components (accessible through the ServiceManager). It implements the monitoring part of a ProvisioningRequest, containing constraint based rules (time and data driven evaluations) and ServiceInstance specific Sensor related configurations. It is general by design, so capable of supporting monitoring of software services, infrastructure services and other resources.

There are three types of Monitoring Capability Features in the monitoring system. First, *Sensors* collect information from a service instance. Their designs and implementations are domain-specific. A sensor can be injected into the service instance (e.g., service instrumentation), or it can be outside the service instance intercepting service operation invocations. A sensor can send the collected information to a communication infrastructure (e.g. an Event Bus) or other components can request (query) information from it. There can be many types of sensors, depending on the type of information they want to collect, but all of them implement a common sensor interface. The interface provides methods for starting, stopping, and configuring a sensor. Second, *Effectors* are components for configuring service instance behaviour. Their designs and implementations are also domain-specific. An effector can also be injected into a service instance or can interface with a service configuration. There can be many types of effectors, depending on the service instance to be controlled, but all of them implement a common effector interface. The interface provides methods for configuring a service. The third type of monitoring feature is a *Reasoning Component Gateway (RCG)*. An RCG provides the interface for accessing a Reasoning Engine. A Reasoning Engine (or short name as Reasoner) performs a computation based upon a series of inputs provided by events or messages sent from sensors or effectors. An example RCG may provide a function to *compute the average completion time* of service requests. In this case the RCG accepts events from sensors detecting both request and responses to a service operation and computes an average over a period of time. RCGs also provide access to generic runtime monitoring frameworks such as EVEREST [14].

4. MONITORABILITY ASSESSMENT

Given an SLA specification and a set of component Monitoring Features, our approach to dynamic assessment of monitoring infrastructures is based on the process illustrated in Figure 3.

The process starts by extracting the Guarantee States from Agreement Terms of the SLA specification. The terms are in turn parsed in to a formal Abstract Syntax Tree (AST) for the expressions of the states and provides a highly efficient and formal basis for analysing the SLA requirements. The AST is then used as input to select each expression of each state (by traversal of the AST) and match each left-hand side (lhs), operator and right-hand side (rhs) of the expression with appropriate component monitoring fea-

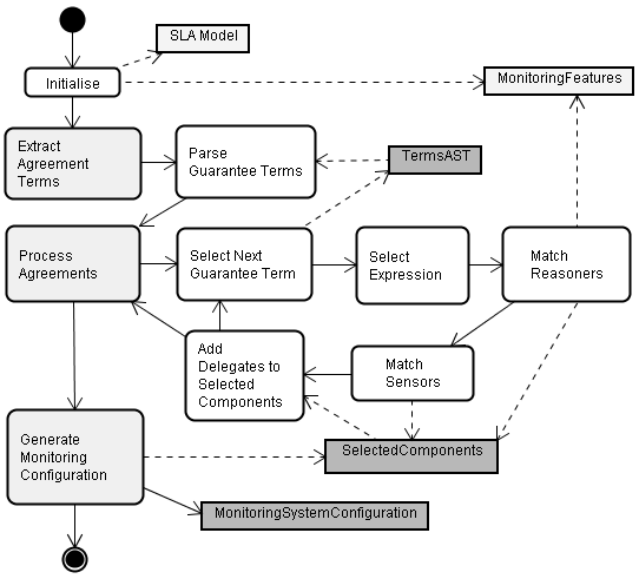

Figure 3: SLA Monitorability Assessment

tures. The monitor matching and selection algorithms are described in detail in early work [4, 5]. Following selection, the delegate components form a SelectedComponents list, which in turn, is used to generate a complete Monitoring System Configuration (MSC) for an SLA. If no suitable monitoring configuration can be formed (i.e. not all monitoring requirements could be matched) then an empty configuration is returned for a particular agreement term. The approach can be used for two perspectives; first, to configure the monitoring system when a new SLA needs to be monitored, and second to perform adjustments to an existing configuration when requirements change or violations are detected. The main focus in this paper is the first, in that we assume that a new SLA is to monitored and therefore do not consider how this would affect the current state of monitoring. The end result of the configuration process is an MSC representing the configuration of selected Monitoring Components which reason or provide events to monitor each Agreement Term of the input SLA. Currently, monitoring assessment and configurations only consider SLA Agreement Terms and Guaranteed States leaving pre-conditions and Guaranteed Actions for future work.

5. THE SMART TOOL SUITE

The primary goal of the Service Monitorability Reporting Tool Suite (SMaRT), which includes a Monitorability Reporting workbench is to provide ease of access to defining, managing and reporting feedback on results for the monitorability of SLAs for service infrastructures. First, we define an architecture for the SMaRT components, their relationships and interactions, and how this is implemented as a workbench. We then describe some validation activities to ensure the approach and techniques are valid for such assessment and reporting.

5.1 Architecture

The architecture (illustrated in Figure 4) is based upon the Model-View-Controller (MVC) software architecture pattern providing management of models (SLA, Monitoring Features and Monitoring System Configuration) and their representing views by way of some controlling components. The views layer provides a component for each of these models and representing their basic state. In addition the use of these views provides input for changes to either SLA, Monitoring Features or results of producing a monitoring system configuration.

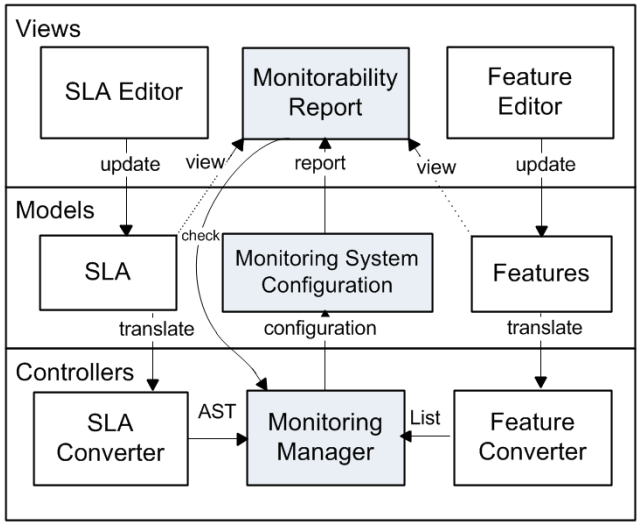

Figure 4: The SMaRT Architecture

Whilst the SLA and Monitoring Feature Editors provide basic input and output operations on their models, the Monitorability Report view provides a much more complex view on the state of monitorability assessment and monitoring system configuration. Use of the workbench is as follows. When both SLA and Monitoring Features are in a *ready* state (i.e. they are complete and updated in their models) the service support analyst can invoke a *check* operation through the report view which starts the monitorability assessment. The MM takes the inputs of SLA and features and performs the monitorability assessment and configuration steps as described in section 4. The results of assessment yields a monitorability assessment report (illustrated in Figure 5).

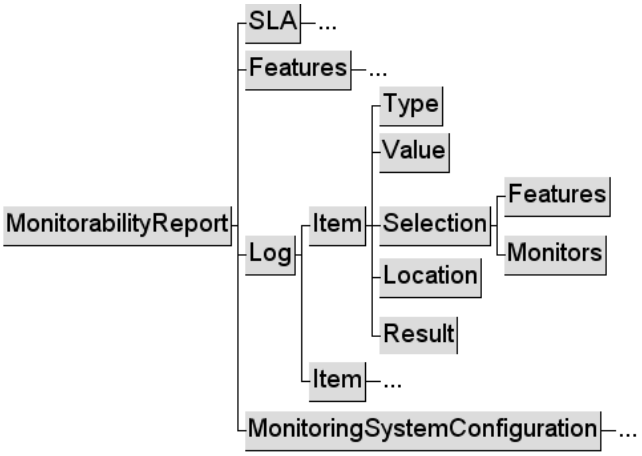

Figure 5: Report Structure as XML Elements

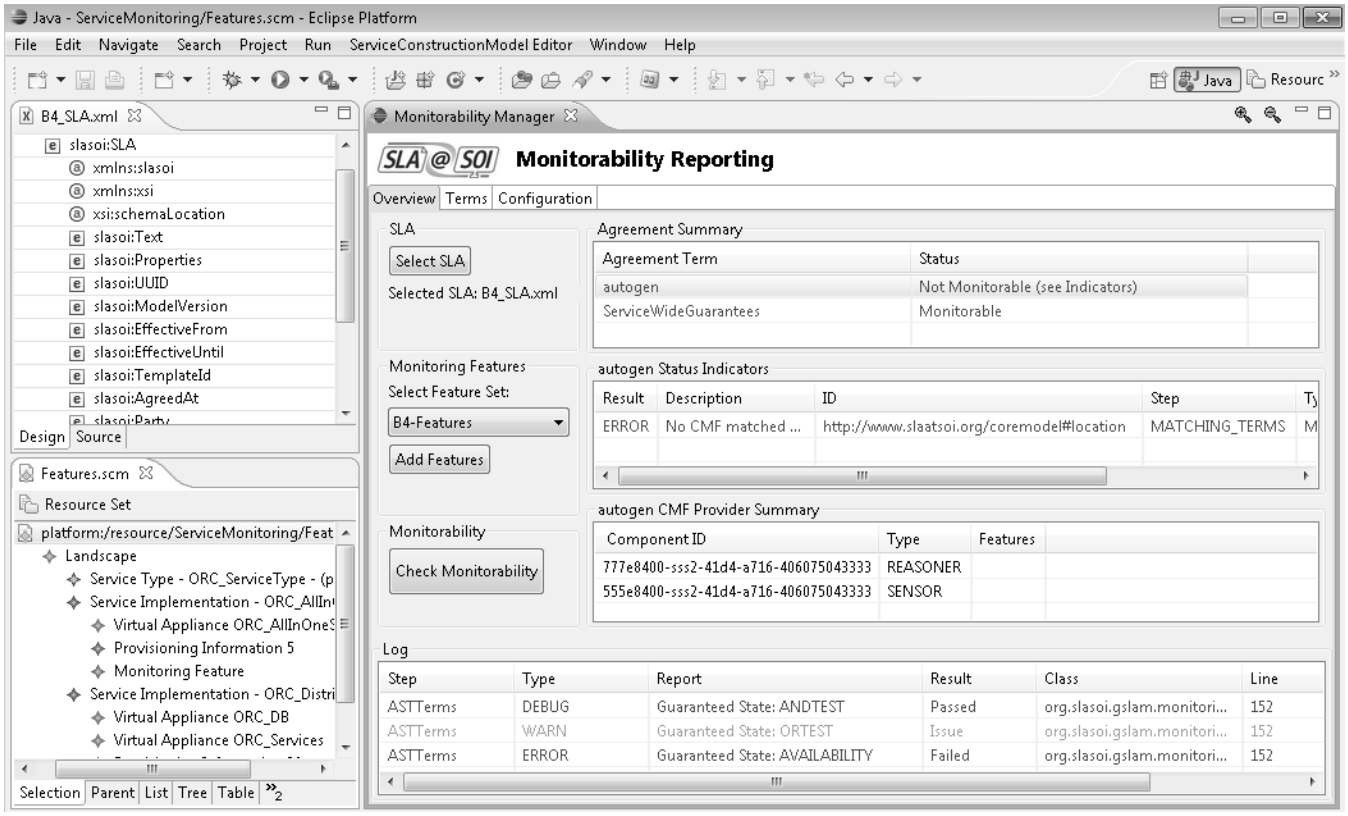

Figure 6: SMaRT: A Service Monitorability Reporting Tool Suite

The report contains for each step of assessment a log item. Each log item has a type (a kind of step, for example matching or selection etc), a result (an indicator for filtering debug, warnings or errors), a value (the value processed depending on the type of step), an optional selection item, a location indicator for the SLA processed item and a complete monitoring configuration as an MSC. The selection item is only included when a successful match and selection has occurred. If the case that a selection is made then features matched and monitors selected are reported. This information is presented back to the user in the form of a table which can be inspected and used to report information back to service consumer and provider. This is particularly useful to assess where further service monitor providers or capabilities are required. As we noted in section 2, even though an assumption in negotiations of SLAs is that the terms agreed can be monitored, additional consumer preference information (such as cost of overall monitoring, geography etc) in provision may not be known in advance. This dynamic provision of monitors may also mean that providers "supply and discontinue" monitors, leading to support issues. Additionally, the MSC is useful for service monitor providers, and service support staff, to reference matched components for particular SLA terms that can be monitored. It may also be inspected, from a technical perspective, to check that assigned reasoner, sensor or effector monitoring components are configured appropriately.

5.2 Workbench

The overall workbench implementation (built upon the Eclipse IDE) is illustrated in Figure 6, showing the three views (SLA Editor, Features List and Monitorability Report). The workbench platform is supported by the Eclipse Modeling Framework (EMF) for underlying models of Features and Monitoring System Configuration. EMF provides native basic editors for these models (as resource views). Additionally, the SLA is represented in a basic XML editor. Future work will explore how these model representations are presented more appropriately to the end-user and linked to the monitoring assessment reports (e.g. bi-directional referencing). The Monitorability Report view provides an accessible interface for service monitoring support staff to check the monitorability of SLAs and examine the assessment results. On the right side of the view in Figure 6, note that the tables listed represent a break-down of SLA Agreement Term Status, Status Indicators for supporting features and Monitoring Component assignment. The table at the bottom of the report view lists each logged entry in the assessment, with good (DEBUG only), warnings (WARN) and errors (ERROR) in assessment matching and selection.

The *SMaRT* Workbench is open source software, available as part the SLA@SOI project platform, examples and related set of tools at **http://sourceforge.net/projects/sla-at-soi/**. Further tools developed as part of the workbench will be released under the same conditions at that address.

5.3 Validation

To thoroughly test the correctness of configurations produced and monitorability reports, we devised an SLA monitorability coverage test based upon each of the model el-

Table 1: Sample Test Cases for SLA Elements, Parsing, Configuration and Monitoring

Test Case	Model	SLA-ID	Events	Parsed	MSC	Client	Monitorable
Core SLA(T)	InterfaceDeclrs	ID1	None	Yes	Yes	Yes	No
Core SLA(T)	AgreementTerms	AT1	Violation	Yes	Yes	Yes	Yes
Core SLA(T)	Guaranteed Actions[a]	GA1	Violation	No	No	No	No
Core SLA(T)	Guaranteed States	GS1	Violation	Yes	Yes	Yes	Yes
Core SLA(T)	VariableDeclrs	VD1	Computation	Yes	Yes	Yes	Yes
Test Case	Terms	SLA-ID	Events	Parsed	MSC	Client	Monitorable
Core SLA(T)	core:and	GS1	Computation	Yes	Yes	Yes	Yes
Core SLA(T)	core:equals	GS1	Computation	Yes	Yes	Yes	Yes
Core SLA(T)	core:sum	GS1	Computation	Yes	Yes	Yes	Yes
Core SLA(T)	core:series	GS2	Computation	Yes	Yes	Yes	Yes
Core SLA(T)	core:availability	GS1	Request-Response	Yes	Yes	Yes	Yes
B4 Use-case	infra:cpu	INF1	Computation	Yes	No	No	No[b]
B6 Use-case	busi:satisfaction	BUS1	Computation	Yes	No	No	No[b]

[a] The element is not currently supported by the MM Component [b] Reported as *NotSupported* to *SMaRT Workbench* user

ements described in the EC SLA@SOI Project SLA Model [13] and the features available by a monitoring engine. Aligned with testing of the overall Monitoring Architecture in section 3 we listed each element along with the specification in a test SLA (SLA-ID), the events required to be monitored (Events), whether the model element expression in the SLA could parsed by the MonitoringManager (Parsed), whether a suitable configuration was produced (MSC), whether the configuraton was accepted by a client monitoring component (Client) and the result of matching and selection items (Monitorable). As we discussed in section 4, the grammar for the SLA parser is currently only based upon the Agreement Term and Guaranteed State expressions. Therefore it is future work to enable Guaranteed Actions to be parsed and monitored. A sample of the results are listed in Table 1. In addition to these, we also tested SLA model metrics (such as units of time) and primitive types (such as BOOL, CONST, TIME etc), mixing them and providing permutations for exhaustive testing.

The other main testing that has been carried out on is with the use-cases featured in the EC SLA@SOI project. The SLA specifications for both B4 (infrastructure quality terms) and B6 (business-level quality terms) use-cases have been fully covered in testing. This includes the elements listed in our initial coverage test. We also expect to continue testing with other monitoring engines, which will illustrate how the generic MonitoringManager features can be used between more than one monitoring engine. For example, the ASTRO Project [10] SLA monitoring tools can be tested with infrastructure monitoring components.

6. RELATED WORK

Related work in this paper falls within two areas. First we consider the approaches for decidability in the monitorability of SLAs and second the reporting on the results of this monitorability assessment. Our work is highly related to automatic monitoring from software requirements such as reported in [1], where the authors describe flexibility in generating monitoring configurations based upon some constraints specified. In this work, the user expresses his/her requirements and assumptions for monitoring in FLEA (a Formal Language for Expressing Assumptions). This language provides a set of composable constructs tailored for the convenient expression of a wide range of monitoring concerns including sequences, combinations and time sensitive events). We base the constraints on those already specified in the SLA, and automatically derive suitable monitoring configurations based upon some monitorability calculation. Several works have focused on deciding monitorability based upon some calculation of the total-cost of monitoring the SLA between provider to provider. For example in [12, 11], the authors describe an approach to determine the monitorability of systems of SLAs (a system of SLAs is closely related to a set of Agreement Terms in the SLA notation used in our work). Analysis of SLAs in their work assumes a set of pre-configured properties (e.g. availability, satisfaction etc) and does not dynamically seek reasoning components as our work provides. Hence our work is different as it aims to report on those terms that are either supported and unsupported, based upon constraints specified in the SLA and also service consumer monitor selection preferences. The TrustCOM project [15] has also produced a reference implementation for SLA establishment and monitoring. This implementation, however, does not involve the dynamic setup of monitoring infrastructures or reporting. The SLA Monitoring and Evaluation architecture presented within the Gridipedia project [6] has several similarities with the approach presented in this paper, such as the need to separate SLA from service management. Their focus of work, however, is on statically binding services and monitors, whilst ours is on dynamically allocating monitors to SLA parts, based upon matching the exact terms that need to be monitored and the monitoring capabilities available in different services.

Second, on monitorability assessment and reporting, the services monitoring infrastructure design and initiation shares many related aspects with broader interacting communications equipment. For example, in [9, 8] the authors describe monitorability assessment for communications infrastructure based upon a specification for monitoring (synonymous for an SLA) and how the capabilities of the monitor-

ing infrastructure can be improved for effective reporting. Our work aligns with this but proposes how this is achieved for service level agreements rather than industrial communication standards. Bridging business and technical monitorability reporting requirements has been discussed in the NextGrid project [7], where the authors propose a human-centric architecture for SLA composition and checking. In particular, they stipulate the importance of business-level objectives such as "utilize my resources a hundred percent" or "get the maximum profit, while spending as little money as possible" alongside the SLA quality of service terms. In a way, these business-level objectives are where the support group discussed in our work liase with consumers and providers to reach such objectives.

The work described in this paper extends an existing approach on dynamic generation of monitoring system configurations [2, 4, 5]. Specifically, we consider individual guarantee terms within an SLA by decomposition of complex guarantee expressions, a wider spectrum of monitoring components (e.g. effectors) and support complex monitoring configurations and reporting that can engage different monitoring components for checking the same SLA term if necessary.

7. CONCLUSIONS

In this paper we have described an approach to reporting on the advanced configuration of service systems, in particular, where monitorability assessment is reported as part of the dynamic configuration of monitoring infrastructures. The work aims to provide support for engineering SLA monitoring components, highlighting where the necessary components for monitoring are not available and therefore are needed from monitoring providers. Our work will be extended to cover further elements of the SLA specification (such as Guaranteed Actions, which are not presently considered) and also including preferential selection of monitoring components. Preferential selection of components is useful where there are multiple monitoring components offered for the same term. Preferences could be based upon monitoring cost (both in computing resource or financially) or non-functional requirements. The existing implementation is already part of the wider SLA@SOI project monitoring platform, providing integration and validation testing, and we are keen to seek other environments to test it within.

8. ACKNOWLEDGMENTS

The research reported in this paper has been supported by the EU Commission as part of the FP7 Integrated Project *SLA@SOI* (grant agreement n. 216556).

9. REFERENCES

[1] D. Cohen, M. Feather.S, K. Narayanaswamy, and S. S. Fickas. Automatic Monitoring of Software Requirements. *Software Engineering, International Conference on*, 0:602, 1997.

[2] M. Comuzzi and G. Spanoudakis. Dynamic Set-up of Monitoring Infrastructures for Service-Based Systems. In *25th Annual ACM Symposium on Applied Computing, Track on Service Oriented Architectures and Programming (SAC 2010)*, Sierre, Switzerland, 2010. ACM.

[3] ECFP7. *SLA@SOI - Grant Agreement N.216556*. European Commission Framework Programme 7, Website: http://www.eu-trustcom.com/, 2007.

[4] H. Foster and G. Spanoudakis. Model-Driven Service Configuration with Formal SLA Decomposition and Selection. In *The 4th International Symposium On Leveraging Applications of Formal Methods, Verification and Validation (ISoLA)*, Crete, Greece, 2010.

[5] H. Foster and G. Spanoudakis. Advanced Service Monitoring Configurations with SLA Decomposition and Selection. In *26th Annual ACM Symposium on Applied Computing (SAC) Track on Service Oriented Architectures and Programming (SOAP)*, TaiChung, Taiwan, 2011. ACM.

[6] Gridipedia. *SLA Monitoring and Evaluation Technology Solution*. Available from: http://www.it-tude.com/?id=gridipedia, 2009.

[7] P. Hasselmeyer, B. Koller, L. Schubert, and P. Wieder. Towards SLA-Supported Resource Management. In *High Performance Computing and Communications*, volume 4208 of *LNCS*, pages 743–752. Springer Berlin / Heidelberg, 2006.

[8] J. May. *Systems and Methods for Intelligent Communicating Storage of Condition Monitorable Replaceable Components*, January 2010.

[9] N. Novikov, O. Korovin, Y. Astapenko, and D. Overchenko. Interaction between Monitorability Indicators and Operational Characteristics of Communications Equipment. *Measurement Techniques*, 39:607–612, 1996. 10.1007/BF02369823.

[10] M. Pistore, F. Barbon, P. Bertoli, D. Shaparau, and P. Traverso. Planning and Monitoring Web Service Composition. In *AIMSA*, pages 106–115, 2004.

[11] F. Raimondi, J. Skene, and W. Emmerich. Efficient Online Monitoring of Web-Service SLAs. In *Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering*, SIGSOFT '08/FSE-16, pages 170–180, New York, NY, USA, 2008. ACM.

[12] J. Skene, A. Skene, J. Crampton, and W. Emmerich. The Monitorability of Service-level Agreements for Application-Service Provision. In *Proceedings of the 6th International Workshop on Software and Performance 2007 (WOSP'07)*, Buenos Aires, Argentina, 2007.

[13] SLA@SOI. *Deliverable D.A1a: Framework Architecture*. Available from: http://sla-at-soi.eu/publications/deliverables, 2009.

[14] G. Spanoudakis, C. Kloukinas, and K. Mahbub. The SERENITY Runtime Monitoring Framework. In *Security and Dependability for Ambient Intelligence,Information Security Series*. Springer, 2009.

[15] TrustCOM. *Deliverable 64: Final TrustCoM Reference Implementation and Associated Tools and User Manual*. Available from: http://www.eu-trustcom.com/, 2007.