A novel reconfigurable by design highly distributed applications development paradigm over programmable infrastructure

D2.2 - Definition of the ARCADIA context model

Editors:	Panagiotis Gouvas (UBITECH), Alessandro Rossini (SINTEF)
Contributors:	Franck Chauvel (SINTEF), Anastasios Zafeiropoulos (UBITECH), Eleni Fotopoulou (UBITECH), Constantinos Vassilakis (UBITECH), Matteo Repetto (CNIT), Kostas Tsagkaris (WINGS), Nikos Koutsouris (WINGS), Katerina Demesticha (WINGS), Stefan Kovaci(TUB), Giovanni Carella(TUB)
Date:	31/7/2015
Version:	1.00
Status:	Final
Workpackage:	WP2 –ARCADIA Framework Specifications
Classification:	Public

ARCADIA Profile

Grant Agreement No.:	645372
Acronym:	ARCADIA
Title:	A NOVEL RECONFIGURABLE BY DESIGN HIGHLY DISTRIBUTED APPLICATIONS DEVELOPMENT PARADIGM OVER PROGRAMMABLE INFRASTRUCTURE
URL:	http://www.arcadia-framework.eu/
Start Date:	01/01/2015
Duration:	36 months

Partners

NUI Galway OÉ Gaillimh	**Insight Centre for Data Analytics, National University of Ireland, Galway**	**Ireland**
SINTEF	**Stiftelsen SINTEF**	**Norway**
TU berlin	**Technische Universität Berlin**	**Germany**
cnit	**Consorzio Nazionale Interuniversitario per le Telecomunicazioni**	**Italy**
Univerza v Ljubljani	**Univerza v Ljubljani**	**Slovenia**
UBITECH ubiquitous solutions	**UBITECH**	**Greece**
WINGS ICT Solutions	**WINGS ICT Solutions Information & Communication Technologies EPE**	**Greece**
GRUPPO Maggioli	**MAGGIOLI SPA**	**Italy**
aditess Advanced Integrated Technology Solutions & Services	**ADITESS Advanced Integrated Technology Solutions and Services Ltd**	**Cyprus**

Document History

Version	Date	Author (Partner)	Remarks
0.1	5/4/2015	Alessandro Rossini (SINTEF)	ToC formulated and chapter assignment performed
0.2	19/4/2015	Eleni Fotopoulou (UBITECH), Franck Chauvel (SINTEF)	Introduction, Additions to Chapter 2
0.3	26/4/2015	Panagiotis Gouvas (UBITECH), Matteo Repetto (CNIT), Nikos Koutsouris (WINGS), Katerina Demesticha (WINGS), Kostas Tsagkaris (WINGS)	Additions to Chapter 3
0.4	3/5/2015	Panagiotis Gouvas (UBITECH), Alessandro Rossini (SINTEF), Franck Chauvel (SINTEF), Stefan Kovaci (TUB), Giovanni Carella (TUB)	Additions to Chapter 2, 4
0.5	17/5/2015	Panagiotis Gouvas (UBITECH), Eleni Fotopoulou (UBITECH), Constantinos Vassilakis (UBITECH)	Additions to Chapters 5, 6
0.6	31/5/2015	Alessandro Rossini (SINTEF), Franck Chauvel (SINTEF)	Finalization of all Chapters
0.7	8/6/2015	Anastasios Zafeiropoulos (UBITECH)	Conclusions & Future Work Finalized
0.8	22/6/2015	Panagiotis Gouvas (UBITECH)	Final Model v1.0 appended and respective figures updated
0.9	29/6/2015	Alessandro Rossini (SINTEF)	Document submitted for internal review cycle
1.0	13/7/2015	Panagiotis Gouvas (UBITECH)	Comments of internal reviewers addressed

Executive Summary

The scope of the ARCADIA project is to provide a novel reconfigurable-by-design Highly Distributed Applications (hereinafter HDAs) development paradigm over programmable infrastructure. Given the inability of Highly Distributed-Application-Developers to foresee the changes as well as the heterogeneity on the underlying infrastructure, it is considerable crucial the design and development of novel software paradigms that facilitate application developers to take advantage of the emerging programmability of the underlying infrastructure and therefore develop Reconfigurable-by-Design applications. In parallel, it is crucial to design solutions that are scalable, support high performance, are resilient-to failure and take into account the conditions of their runtime environment.

As a result, the development, configuration and operation of Highly Distributed Applications entail many and multi-level challenges. These challenges relate to the entire life-cycle of a highly distributed application. More specifically, this lifecycle includes software engineering, optimization of deployment, infrastructural on-boarding, execution and the deprovisioning phases.

In order to cope with challenges, ARCADIA Framework will rely on an extensible Context Model which will be used during the entire lifecycle of an HDA. The purpose of this deliverable is to document the first version of the ARCADIA Context Model. It should be clarified that this model is 'multi-faceted' since it consists of complementary models that are conceptually grouped according to their functional purpose. To this end, for the sake of comprehension, this deliverable elaborates in each facet separately.

Towards the above lines, the four main facets of the existing model are thoroughly described, namely; a) the ARCADIA Component Model, b) the ARCADIA Service Graph Model, c) ARCADIA Service Deployment Model and d) the ARCADIA Service Runtime Model. Beyond the description of the models per se, the scope of this deliverable is to provide a bird's eye view of the ARCADIA architecture and elaborate on existing models that affected the formulation of ARCADIA Context Model. On the one hand, the description of the high level view of the architecture is necessary for the comprehension of the provided modeling artifacts while on the other hand it should be clarified that during the modeling process re-usable parts of other models have been used.

Moreover, it should be clarified that the development of the ARCADIA Context Model is a continuous and iterative procedure. That is the reason why this deliverable reports only on the first version. The documented model will be evolved and extended accordingly based on the feedback of the final architecture and the release of the early prototype. The extensions that are already in progress are discussed in the final chapter. Finally, the ARCADIA Context Model is modeled using XML schema notation. The updated version of the model will be published in the projects' web site (http://www.arcadia-framework.eu). For the sake of completeness, the documentation of the existing normative format is provided in Annex II.

Table of Contents

List of Figures

Acronyms

Acronym	*Explanation*
CAMEL	Cloud Application Modelling and Execution Language
CSOP	Constraint Satisfaction Optimization Problem
DG	Directed Graph
DoW	Description of Work
HDA	Highly Distributed Application
IaaS	Infrastructure as a Service
MDE	Model Driven Engineering
NFV	Network Function Virtualization
PoP	Point of Presence
SDN	Software Defined Networking
WP	Work Package
XSD	XML Schema Definition

1 Introduction

The development, configuration and operation of highly distributed applications entail many and multi-level challenges. These challenges relate to the entire life-cycle of a highly distributed application (hereinafter HDA). More specifically, the lifecycle includes the Software Engineering phase, the Infrastructural On-boarding phase, the Execution and the Optimization phase. Under this perspective, the vision of ARCADIA is to provide a novel reconfigurable by design HDAs' development paradigm over programmable infrastructure. To do so, the ARCADIA Framework will rely on the development of an extensible context model that will be used during the entire lifecycle of an HDA. The purpose of this deliverable is to provide the first version of the ARCADIA Context Model.

1.1 Scope of the Deliverable

Since the ARCADIA Context Model is used in the entire lifecycle of an HDA, it is crucial to understand that the model per se is not 'monolithic'; on the contrary it consists of multiple sub-models that are used in one phase of the lifecycle. For the sake of consistency, we will address, hereinafter, these sub-models as 'facets'. Therefore, it is essential to clarify how many facets comprise the ARCADIA Context Model and elaborate on each of these separately. The scope of this deliverable is to introduce the first version of the facets and elaborate on their usage. The existing deliverable is released before the architecture deliverable (D2.3). This fact generates some comprehension problems since many modeling facets are bound to architectural components. To this end, in the frame of this deliverable, we present a standalone version of the ARCADIA Context Model by providing all appropriate information needed by the end-user prior to the explanation of the modeling artifacts. That is the reason why a high level view of the architecture is provided without delving into many technical details.

Beyond the architectural overview, which is briefly discussed, the clarification of the HDA lifecycle phases is also provided. The scope of this deliverable is to analyze the modeling facets that are associated with each lifecycle-phase and elaborate of what is the role of each facet. This is essential since the ARCADIA Context Model is used in a diverse way by different architectural components in order to achieve specific technical or non-technical goals. The demystification of all these is a major goal.

Furthermore, it should be clarified that the ARCADIA Context Model is a normative model and not a conceptual/descriptive one. As such, it allows strict validation of all model instances that can be produced based on this model. However, in the domain of HDAs and Cloud Computing there are a lot of descriptive models that have been proposed. Some of these models provide fundamental concepts that should be re-used in ARCADIA. Since the aim of ARCADIA is to avoid re-inventing the wheel, wherever this is possible, the correlation of the ARCADIA Context Model and the existing models is also briefly discussed. In addition, since the modeling facets are normative the XSD serialization format has been chosen based on the mature libraries that already exist for XSD handling. Under this perspective a fully documented XSD model has been developed. For the sake of completeness, the formal XSD along with its documentation is released in the project web-site: http://www.arcadia-framework.eu.

Finally, the development of the ARCADIA Context Model is a continuous and iterative procedure. That is the reason why this deliverable reports only on the first version. During the course of the project, three scheduled releases will be produced. The expected delta between the existing version and the

scheduled ones will be clearly defined. In a nutshell, the scheduled upgrades will extend the 'width' and the 'depth' of the ARCADIA Context Model. 'Width' refers to the creation of the facets that have not been released in the first version; while 'depth' refers to the refinement or completion of an existing facet.

1.2 Structure of the Deliverable

Taking under consideration the scope of the current deliverable, this report has been structured as follows; Chapter 2 provides a high level view of the ARCADIA architecture which is required in order for the reader to comprehend the usage of the various modeling artifacts. In parallel, the modeling artifacts per se are described along with their correlation with the architectural components and the HDA lifecycle that they correspond to. Special emphasis is given to the description of the facets that are finalized in the frame of the first version. Moreover, since the ARCADIA Context Model is practically a new model it is essential to examine the existing models which attempt to conceptualize part of the domains considered relevant in HDA applications such as elasticity, scalability etc. Towards these lines, the final section of this chapter is devoted to the analysis of these models. It is essential to perform an assessment on the reusability of the existing models as well as to infer the complementary aspects.

Chapter 3 is devoted in the analysis of the ARCADIA Component Model. This modeling facet represents the most granular executable unit of an ARCADIA application. Several Component Model instances can be combined towards the realization of a Service Graph. An HDA consists of several components that can be physically and logically distributed; yet they collaborate in order to provide a complex service. Since these components have dependencies among them, it could be argued that a complex service is practically a logical graph of dependencies between executable components. This logical graph is a directed graph which will be addressed as ARCADIA Service Graph. Chapter 3 analyzes the elements of the Component Model. These elements relate to Components' metadata, execution requirements, required interfaces, exposed interfaces etc.

Chapter 4 provides an overview of the ARCADIA Service Graph Model as defined above. It has to be clarified that this model reuses entirely the ARCADIA Component Model since a graph consists practically of interconnected Component instances. Special emphasis is given to the additional elements that relate mainly to the definition of the graph and to the monitoring metrics of the entire service graph. Moving one step forward, one Service Graph can be deployed in a different way taking under consideration the announced infrastructural resources. Irrelevant to the algorithm that performs the deployment plan (i.e. the placement of one component in an execution container) the deployment plan of a Service Graph is represented by the ARCADIA Service Deployment Model. This model is described in Chapter 5.

After the formulation of the deployment plan and the actual placement of a Service Graph Components in respective execution containers the entire service is considered to be in the operational state. The operational state of the service (addressed also as runtime state) is described by the ARCADIA Service Runtime Model in Chapter 6. For the sake of completeness, chapter 7 analyses the concrete extensions that will be provided to the model in the frame of the next scheduled releases. The actual formal normative model is provided in Annex II.

Finally, the deliverable is generated following some writing conventions which are explained below:

- Intra-sentence capitalization of words: Some phrases in the deliverable are deliberately capitalized (e.g. ARCADIA Service Graph Model). These phrases denote either ARCADIA modeling artifacts or ARCADIA architectural components (e.g. ARCADIA Smart Controller).
- Usage of Abbreviations: All abbreviations that are used(e.g. HDA) are summarized in the abbreviations' table above
- Usage of SHALL, SHOULD, MAY according to RFC2119: Whenever these terms are used in the deliverable or in the model's annotations they follow the RFC2119 convention.
- Usage of '*Quoted-Italic*' style: XSD elements are denoted as quoted italic terms.

2 The role of ARCADIA Context Model in the ARCADIA Operational Environment

2.1 The Facets of the ARCADIA Context Model and their usage in HDA lifecycle

As already discussed, the ARCADIA project aims to address challenges that relate to the development and operation of HDAs in reconfigurable environments. The ARCADIA Context Model is used in all service lifecycle phases (i.e. development, composition, deployment planning, execution) in order to conceptualize specific aspects of HDA services that are essential by the ARCADIA architectural components. Figure 2-1 summarizes the six facets of the ARCADIA Context Model.

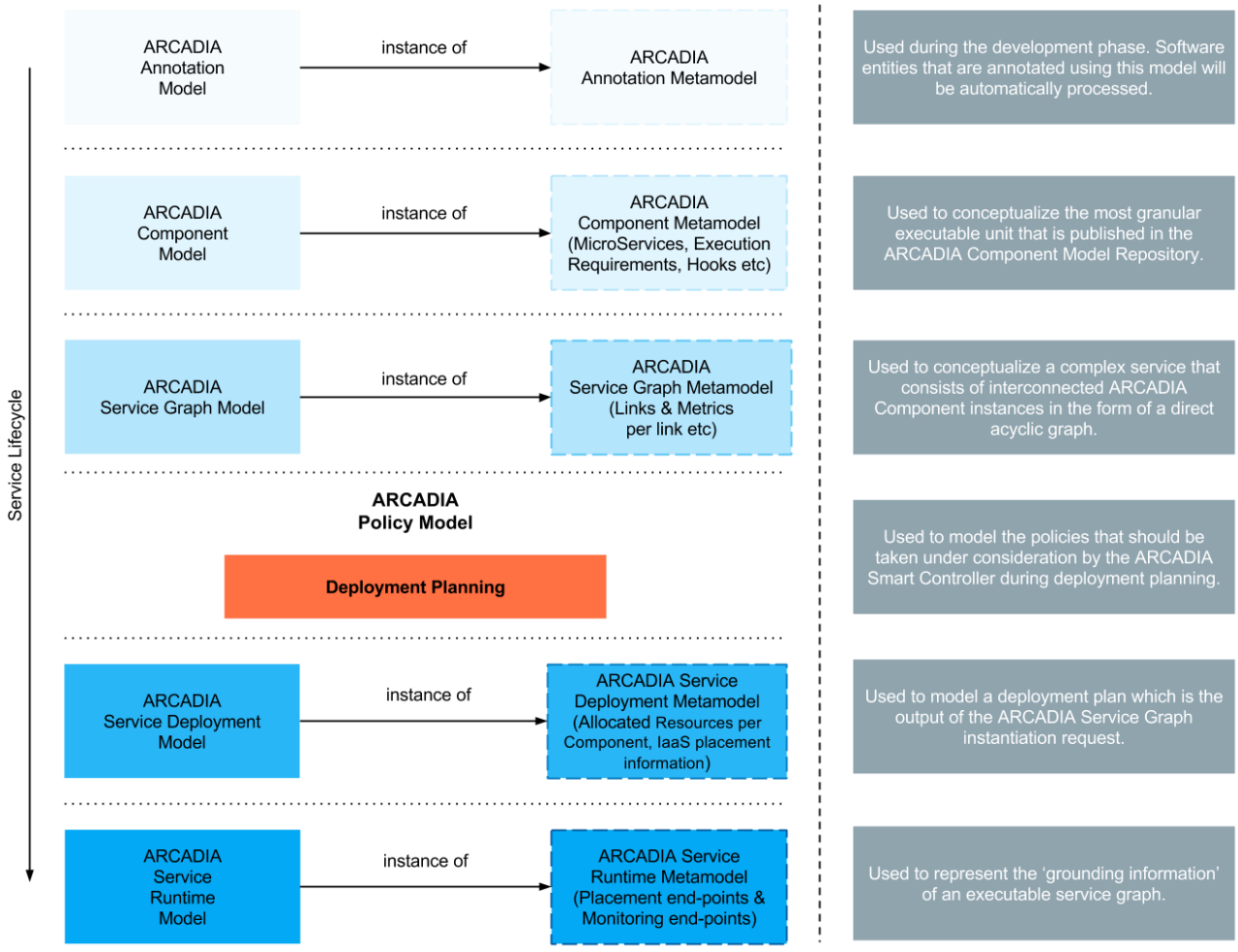


Figure 2-1: The ARCADIA modelling facets and their usage

In a nutshell, the Annotation facet is used to conceptualize code-level annotations that can be used by developers during the implementation of components. These annotations will be interpreted by a specific interpreter in order to create an instance of a Component Model. A Component Model instance represents the most granular executable unit that can be hosted in an execution container of an IaaS provider. As we will see in chapter 3, a Component Model is the most granular executable unit of the ARCADIA ecosystem. A design decision that has been drawn early in the project is that an ARCADIA

Component model may be auto-generated based on code-level annotations or it may be manually created by a DevOps user.

The Component facet is complemented by the Service Graph facet. The Service Graph represents a complex service that consists of several components that have interdependencies to each other. From a logical point of view, the components and the virtual links constitute a directed graph. The scope of this facet is to represent the graph along with several monitoring metrics that relate to the evaluation of the entire graph.

Moving to the next facet, a complex service (that is represented by a service graph) can be instantiated in many ways taking under consideration the underlying programmable resources. Let us elaborate on a simple example according to which one service consists of two components and one virtual link between them. If a service provider owns an account to two IaaS providers s/he has the ability to instantiate the two components on the same IaaS or on different IaaS providers. The decision is driven by a specific policy.

Therefore, the scope of the next facet (Policy facet) is to define the policies that relate to one service graph. The definition of a policy is essential since the decision of which component has to be instantiated in which IaaS resource (a.k.a. deployment planning) is practically an optimization problem that has to take specific constraints under consideration and produce the optimal deployment plan (minimizing the costs and maximizing the benefits). This optimization problem will be solved by a sub-component of the ARCADIA Smart Controller.

Assuming that a deployment plan has been produced (manually or through the ARCADIA Smart Controller) there should be a way to represent the service graph instantiation on top of the IaaS resources. The ARCADIA Service Deployment facet undertakes this responsibility. More specifically, the placement information per each component is modelled along with valuable information which will guide the actual deployment process. When a deployment plan is performed, each component and each link contains some 'grounding' information e.g. routable IP-address, reserved port etc. The runtime aspects of a deployment plan is represented by the Runtime facet.

It should be clarified that in the first version of the model which is documented in this deliverable, four out of the six facets have been elaborated. More specifically, the Annotation facet and the Policy facet will be documented in the next scheduled release. Furthermore, specific types of enhancements are already under development for the existing facets. A holistic view of planned extensions and enhancements is provided in chapter 7.

2.2 Bird's eye view on ARCADIA Architectural Components that use the Context Model

The modeling facets that have been presented above are created or consumed by different 'business' roles or architectural components. Before we delve into the details of the ARCADIA Context Model we will provide a high level view of the architectural components that interact with the model instances. Figure 2-2 depicts the aforementioned architectural components. Initially, an ARCADIA Component Model can be created either manually or using code-level annotations. In case of the usage of the annotations, the Annotations Interpreter component is used by an HDA developer. While in the case of manual creation a DevOps user creates a Component instance and publishes it directly in the Component Model repository. As depicted there are two separate repositories; one that stores the instances of the Component Model and one that stores the Service Graphs.

A Service Graph is created through the combination of multiple Component Model instances. The responsibility of creating a complex service is appointed to a Service Provider. To do so, s/he uses the ARCADIA Service Graph Editor. Any Service Graph can be instantiated on top of a multi-IaaS environment. As already explained, this is a constraint satisfaction optimization problem (a.k.a. CSOP) that is solved by the Smart Controller. The constraints per se are provided by the ARCADIA Policy Manager.

After the Smart Controller proposes an optimal solution, the solution will be serialized in the form of a Deployment Model. The Deployment Model will be 'executed' by a Deployment Manager (it is part of the Smart Controller) which practically means that each component and each virtual-link will be realized in the allocated execution container that is appointed. After the Deployment Model instantiation the Deployment Manager produces the respective Service Runtime Model which is used during the operation of the service.

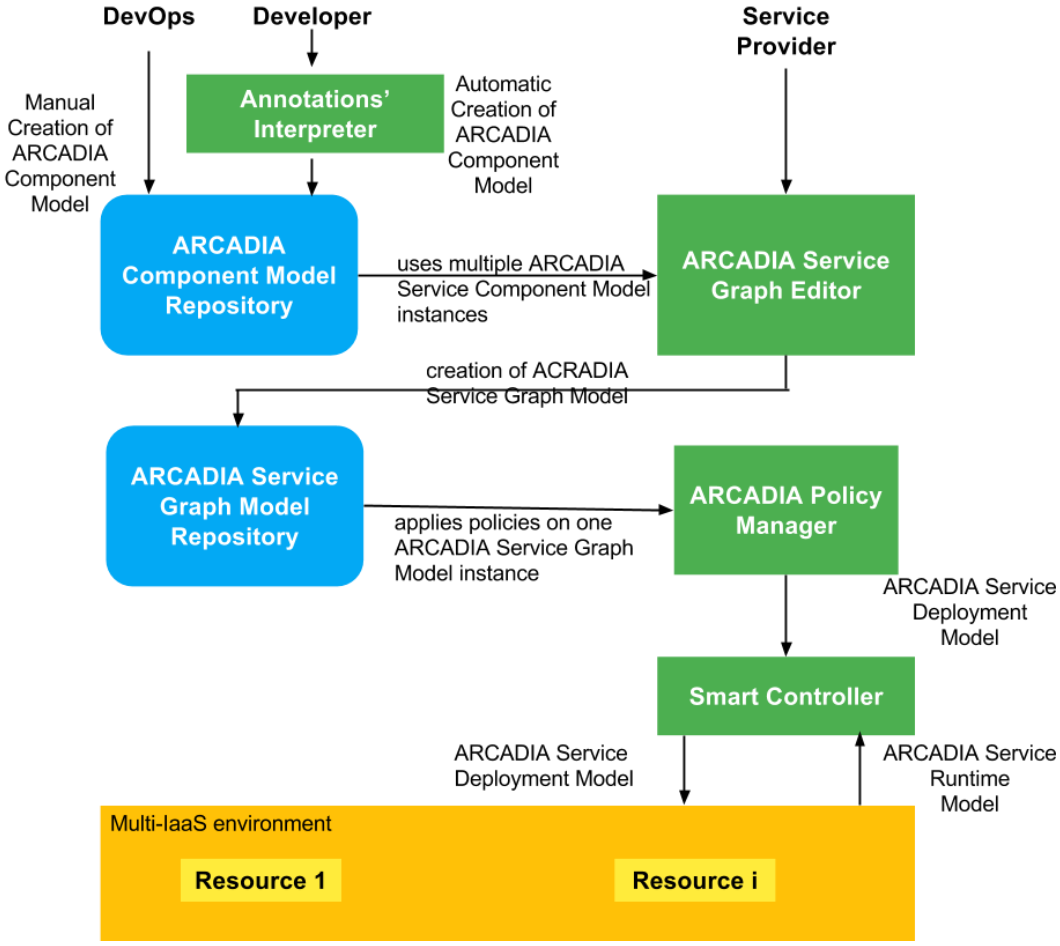


Figure 2-2: The ARCADIA architectural components that utilize the context model

It should be clarified that Figure 2-2 provides only a high-level view of some architectural components that relate directly to the models discussed in the current deliverable. More details will be provided in the respective architectural deliverable (D2.3) where all components are thoroughly elaborated.

2.3 Related Work that affected the formulation of ARCADIA Context Model & Differentiation

2.3.1 Compatibility of ARCADIA context model with Canonical Juju meta-model

The ARCADIA Context Model is used in all service lifecycle phases in order to address several issues of HDA applications. Two critical aspects of the ARCADIA Context Model are the conceptualization of the 'executable unit' that can be hosted in a virtualized execution container and the conceptualization of the dependencies among the 'executable units'. Irrelevant to the nature of the application (if it is highly distributed or not) there are existing approaches that try to address the two aforementioned aspects.

One of the most prominent is Juju[4] which is developed by Canonical Ltd. Juju is a framework that can be used in order to model services in the cloud and orchestrate their deployment and management. It contains several interaction modalities such as a command line and a graphical user interface. Through these modalities a DevOps user can instantiate a service on top of multiple IaaS providers (e.g. OpenStack) (see Figure 2-3).

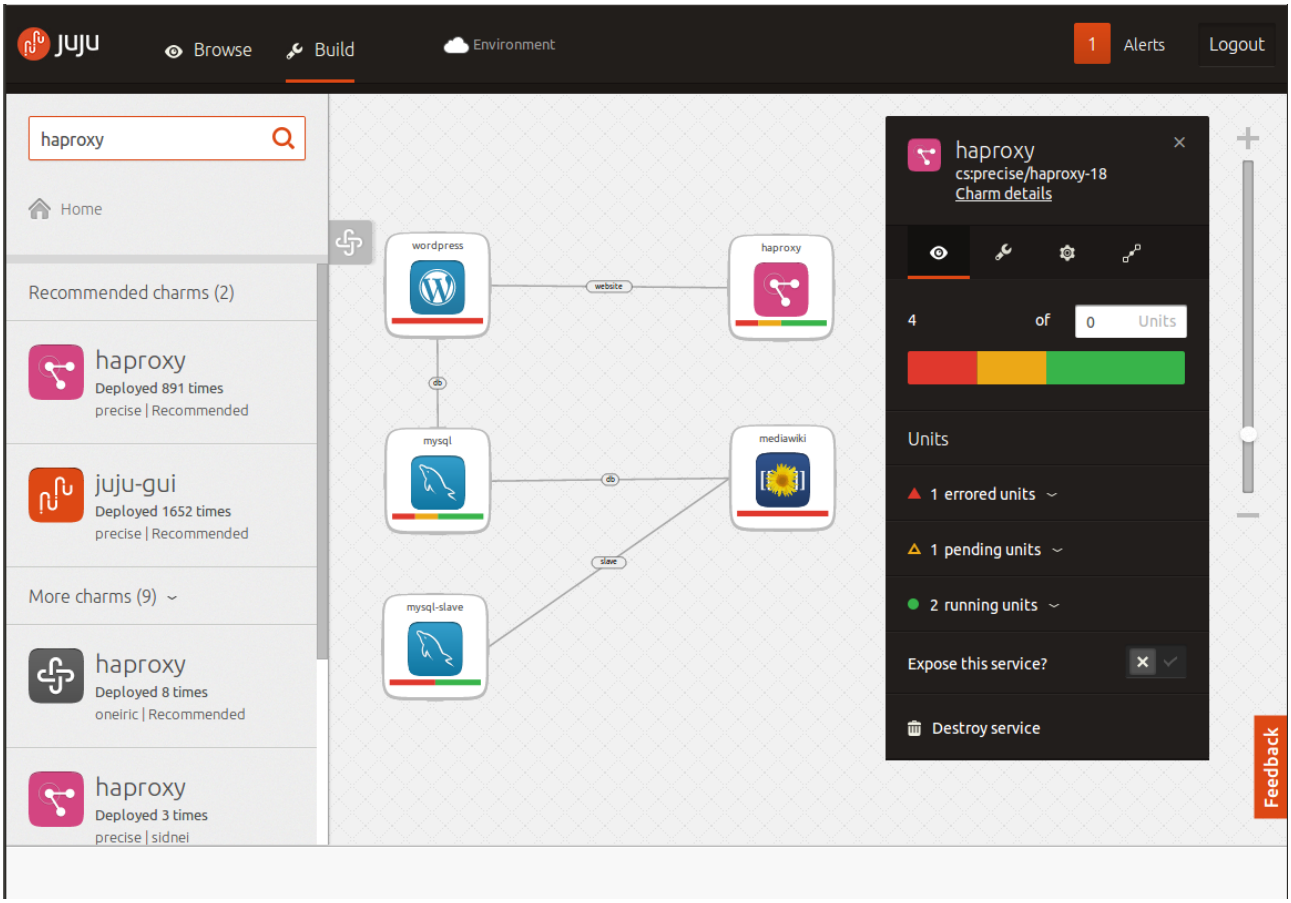


Figure 2-3: Juju GUI [4]

One of the most valuable assets of the Juju platform is the service-metamodel. The service-metamodel is addressed as 'charm' [5] and contains specific elements that are required in order for a specific service to be composable and orchestratable. *Although, the Juju platform is not built in order to address HDA requirements (e.g. it does not support multi-IaaS service deployment) we consider that the 'charm'*

model is well-defined in terms of concepts and their relationships; therefore we put special emphasis so as the ARCADIA Component Model to be fully backward compatible with it.

2.3.2 Compatibility of ARCADIA context model with TOSCA NFV specification

Towards the specification of the ARCADIA Context Model, focus has been given on supporting backward compatibility with relevant work that is in progress within standardization forums as well as other research projects. Given that each ARCADIA application is represented in the form of a service graph consisted by a set of software components along with the dependencies among them –as defined in Deliverable 2.1 [2]-, part of which may be related with specific Virtual Network Functions (VNFs) –as described and deployed in the Network Function Virtualization (NFV) initiatives- it is considered important the support of compatibility with the TOSCA NFV specification.

Actually, the TOSCA NFV profile specifies a NFV specific data model using TOSCA language [1].The deployment and operational behavior requirements of each Network Service in NFV is captured in a deployment template, and stored during the Network Service on-boarding process in a catalogue, for future selection for instantiation. This profile uses TOSCA as the deployment template in NFV, and defines the NFV specific types to fulfill the NFV requirements. By being compatible with this profile specification, TOSCA NFV deployment scripts can be mapped to ARCADIA deployment scripts where the software components that constitute the service graph regard a set of VNFs.

Specifically, in the NFV world, the Network Service Descriptor (NSD) describes the attributes and requirements necessary to realize a network service[3]. A network service can be viewed architecturally as a forwarding graph of Network Functions (NFs) interconnected by supporting network infrastructure. Such network functions may regard to Virtual Network Functions (VNFs) or Physical Network Functions (PNFs), while their interconnection is realized via Virtual Links (VLs) (see Figure 2-4). A VL describes the basic topology of the connectivity between one or more VNFs connected to this VL and other required parameters (e.g. bandwidth and QoS class).NFV introduces Connection Points (CPs) that represent the virtual and/or physical interfaces of the VNFs and their associated properties and other metadata. In TOSCA, the modeling of the NFV applications is realized by using the TOSCA *node*, *capability* and *relationship* types, while also using the *virtualLinkTo* relationship between VNF and virtual link, as shown in Figure 2-5.

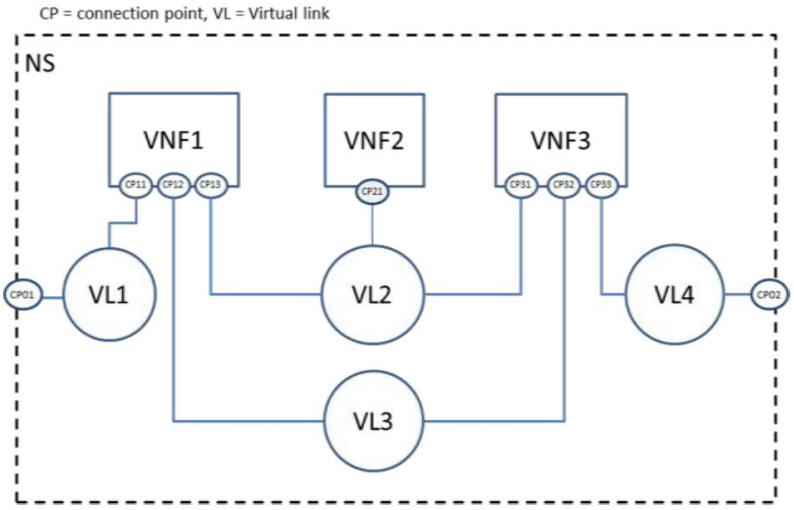


Figure 2-4: Network Service Example for NFV [3]

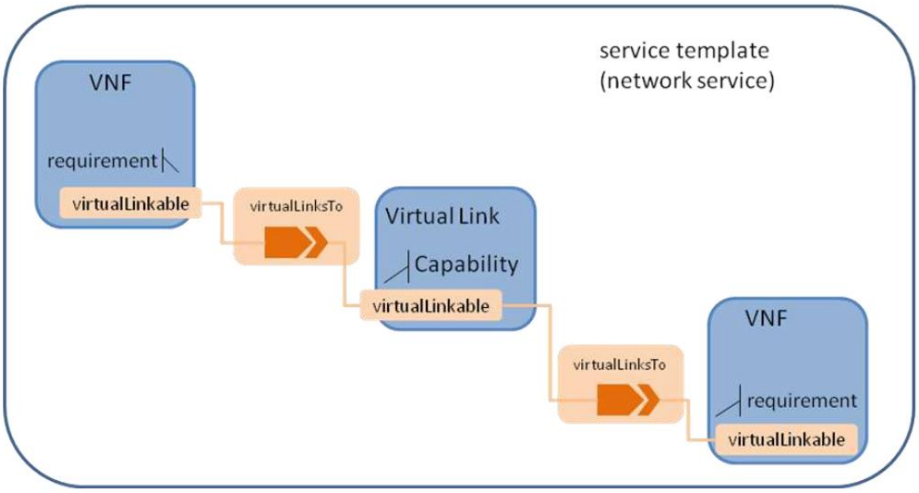


Figure 2-5: TOSCA node, capability and relationship types used in NFV application [1]

In ARCADIA, compatibility with the TOSCA NFV profile is going to be supported, since, as it is described in the following sections, the proposed context model includes the notion of service graphs (similar to the VNF forwarding graphs in NFV), software components (similar to VNFs in TOSCA NFV, however not limited only to VNFs), virtual links (similar to VLs in TOSCA NFV), as well as representation of a set of concepts related with a set of monitoring hooks, infrastructural resources and policies imposed by the services providers. Contribution to the evolution of the work realized in TOSCA NFV specification on behalf of the project is also going to be examined.

2.3.3 Correlation with models that derived from research projects

Beyond industrial approaches (such as Juju) and standardization efforts (such as TOSCA NFV) there are several research projects in the area of cloud computing that attempt to formulate re-usable models. Although these models do not target HDAs their produced models have been taken under consideration during the ARCADIA Context Model engineering. One of the most prominent and active project is PaaSage [6].

PaaSage aims to facilitate the modelling and execution of cloud-based applications by leveraging upon model-driven engineering (a.k.a. MDE) techniques and methods, and by exploiting multiple cloud infrastructures. MDE is a branch of software engineering that aims at improving the productivity, quality, and cost-effectiveness of software development by shifting the paradigm from code-centric to model-centric. Models enable the abstraction from the implementation details of heterogeneous cloud services, while model transformations facilitate the automatic generation of the source code that exploits these services. This approach, which is commonly summarised as “model once, generate anywhere”, is particularly relevant when it comes to the modeling and execution of multi-cloud applications (i.e., applications that can be deployed across multiple private, public, or hybrid cloud infrastructures), which allow exploiting the peculiarities of each cloud service and hence optimizing performance, availability, and cost of the applications [17].

Models can be specified using general-purpose languages like the Unified Modeling Language (UML) [7]. However, to fully unfold the potential of MDE, models are frequently specified using domain-specific languages (DSLs), which are tailored to a specific domain of concern. In order to cover the necessary aspects of the modelling and execution of multi-cloud applications, PaaSage adopts the Cloud Application Modelling and Execution Language (a.k.a. CAMEL).

CAMEL integrates and extends existing DSLs, namely Cloud Modelling Language (CloudML) [8, 9, 10], Saloon [11, 12, 13], and the Organisation part of CERIF [14]. In addition, CAMEL integrates new DSLs developed within the project, such as the Scalability Rule Language (SRL) [15, 16]. CAMEL enables PaaSage users to specify multiple aspects of multi-cloud applications, such as provisioning and deployment topology, provisioning and deployment requirements, service-level objectives, metrics, scalability rules, providers, organisations, users, roles, security controls, execution contexts, execution histories, etc.

The abstract syntax of a language describes the set of concepts, their attributes, and their relations, as well as the rules for combining these concepts to specify valid statements that conform to this abstract syntax. The concrete syntax of a language describes the textual or graphical notation that renders these concepts, their attributes, and their relations.

Unlike the ARCADIA Context Model, CAMEL has been designed as a single metamodel organised into packages, whereby each package reflects the aspect (or domain) covered by the package. Figure 2-6 shows the top-level camel package of the CAMEL metamodel. A CamelModel is a collection of sub-models as follows: DeploymentModels, RequirementModels, LocationModels, MetricModels, ScalabilityModels, ProviderModels, OrganisationModels, SecurityModels, ExecutionModels, and TypeModels.

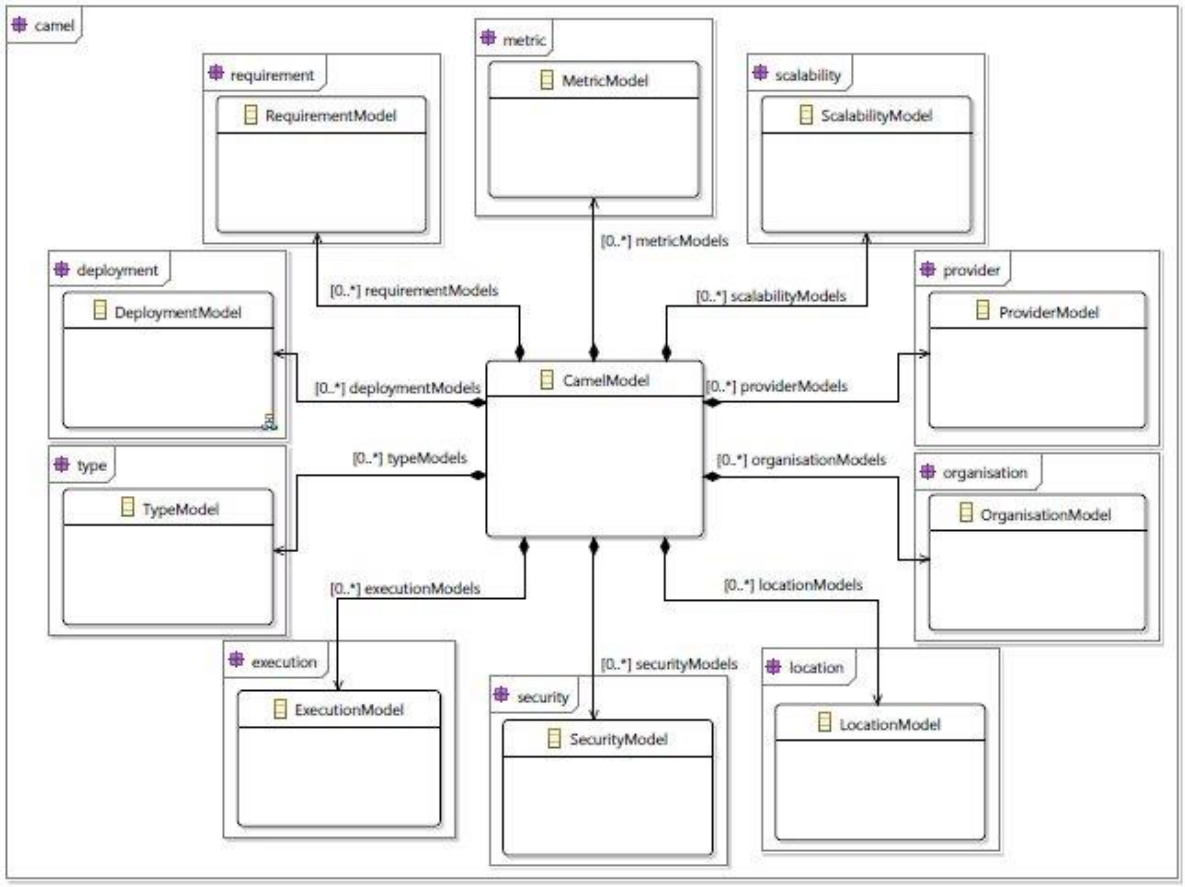


Figure 2-6: The class diagram of the CAMEL metamodel including packages [17]

Although ARCADIA does not adopt an MDE approach (since it follows a juju-like approach which is closer to the HDA lifecyle) many concepts of CAMEL (such as DeploymentModel, MetricModel and ProviderModel) have been ported in XSD notation and reused.

3 Overview of the ARCADIA Component Model

3.1 Overview of the ARCADIA Component Model

The ARCADIA Component Model represents the most granular executable unit of an ARCADIA application. Several Component Model instances can be combined towards the specification of a Service Graph. As already explained, HDAs are practically an instantiation of a complex service graph. Therefore, Components can be considered as the building blocks of these graphs. Figure 3-1 provides the first level of the Component Model schema.

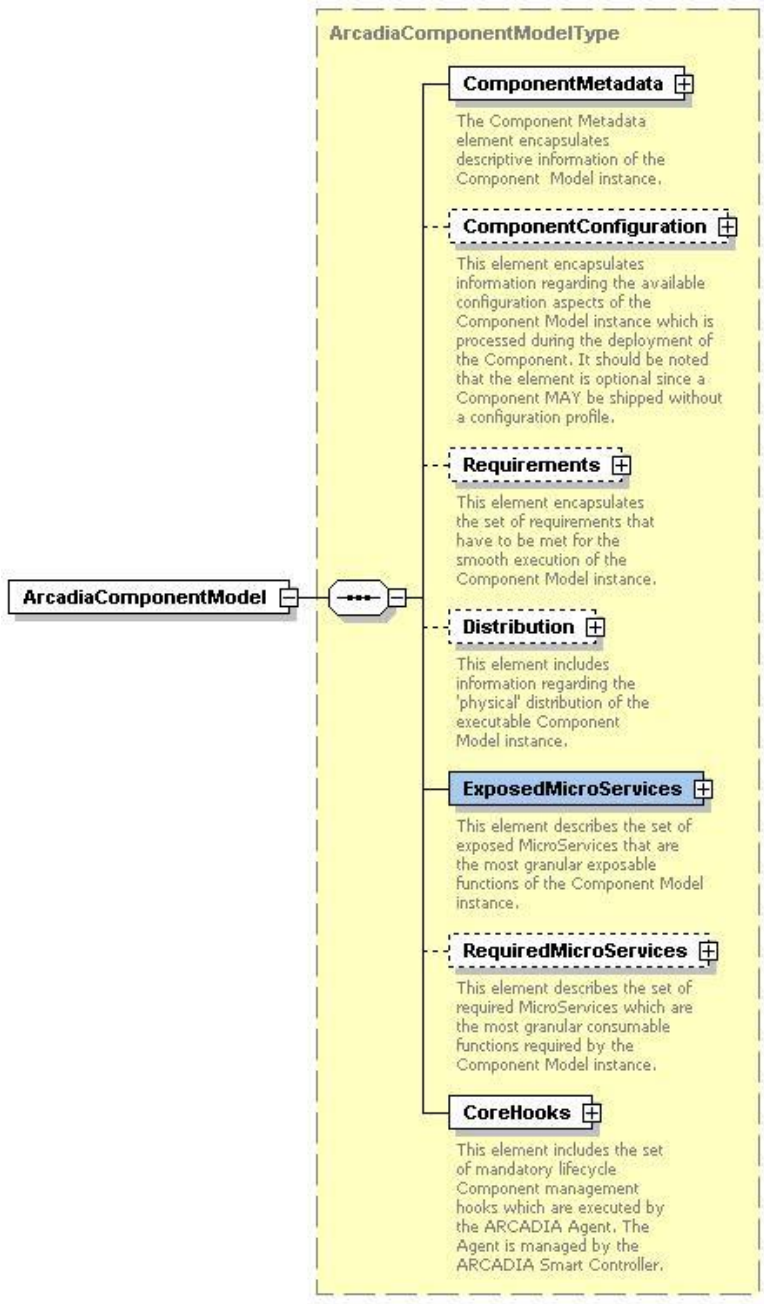


Figure 3-1: Overview of ARCADIA Component Model

As it is depicted, the Component Model schema consists of seven elements. The '*ComponentMetadata*' element encapsulates descriptive information of the Component Model instance. The '*ComponentConfiguration*' element encapsulates information regarding the available configuration aspects of the Component Model instance which is processed during the deployment of the Component. It should be noted that the element is optional since a Component may be shipped without a specific configuration profile.

Moreover, the '*Requirements*' element encapsulates the set of requirements that have to be met for the smooth execution of the Component Model instance. The '*Distribution*' element includes information regarding the 'physical' distribution of the executable Component Model instance. The '*ExposedMicroServices*' element describes the set of exposed MicroServices that are the most granular exposable functions of the Component Model instance. The '*RequiredMicroServices*' element describes the set of required MicroServices which are the most granular consumable functions required by the Component Model instance. The '*CoreHooks*' element includes the set of mandatory lifecycle Component management hooks which are executed by the ARCADIA Agent. The Agent is managed by the ARCADIA Smart Controller.

3.2 Elaboration on the ARCADIA Component Model

The scope of this section is to provide a bird's eye view of the elements that belong to the first level. The first '*ComponentMetadata*' element consists of several sub-elements that aim to characterize any Component Model instance (see Figure 3-2).

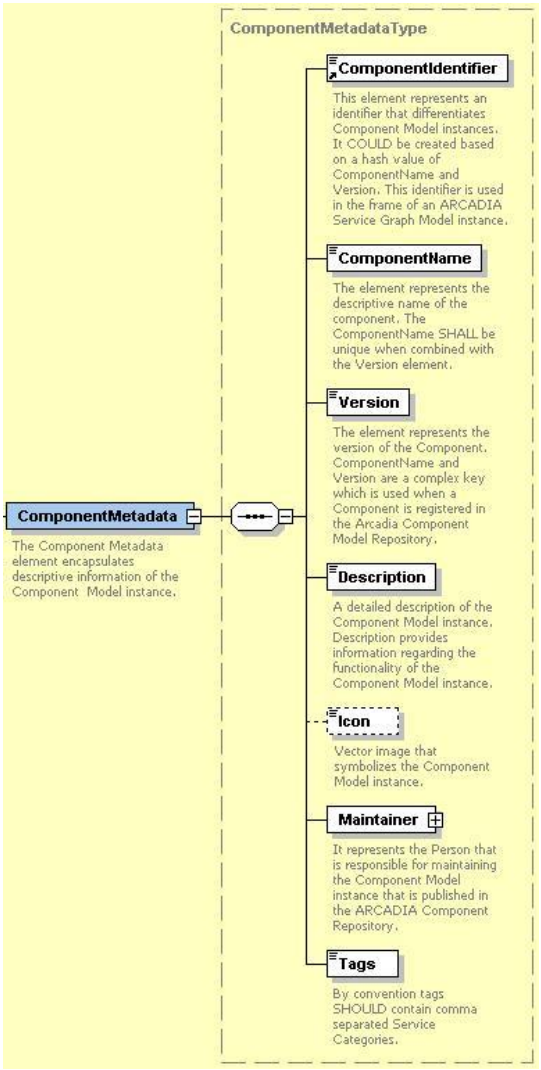


Figure 3-2: Overview of 'ComponentMetadata' element

The unique identification of a Component Model instance is a crucial functionality since in the frame of the ARCADIA operational environment a dedicated repository (the 'Component Repository') will host the Component Model instances. The metadata section of a Component is complemented by elements that represent the Component's name, its version, its description, its maintainer (natural person) and some indexing tags which will be used for searching. Search functionality will be provided by the 'Service Graph Composer' which is the component used in order to create Directed Graphs that represent one complex service.

Furthermore, a crucial aspect of the Component Model relates to its configuration layer (see Figure 3-3). Each component may have one or more configuration parameters. Each parameter is represented by one identifier ('*ConfigurationElementIdentifier*'), one descriptive label, one default value, a declarative description of the parameter and an optional enumeration of available values that may be used. It should be clarified that the 'logical' validation of the configuration layer should be performed by the business logic that parses the XML file that corresponds to a Component Model.

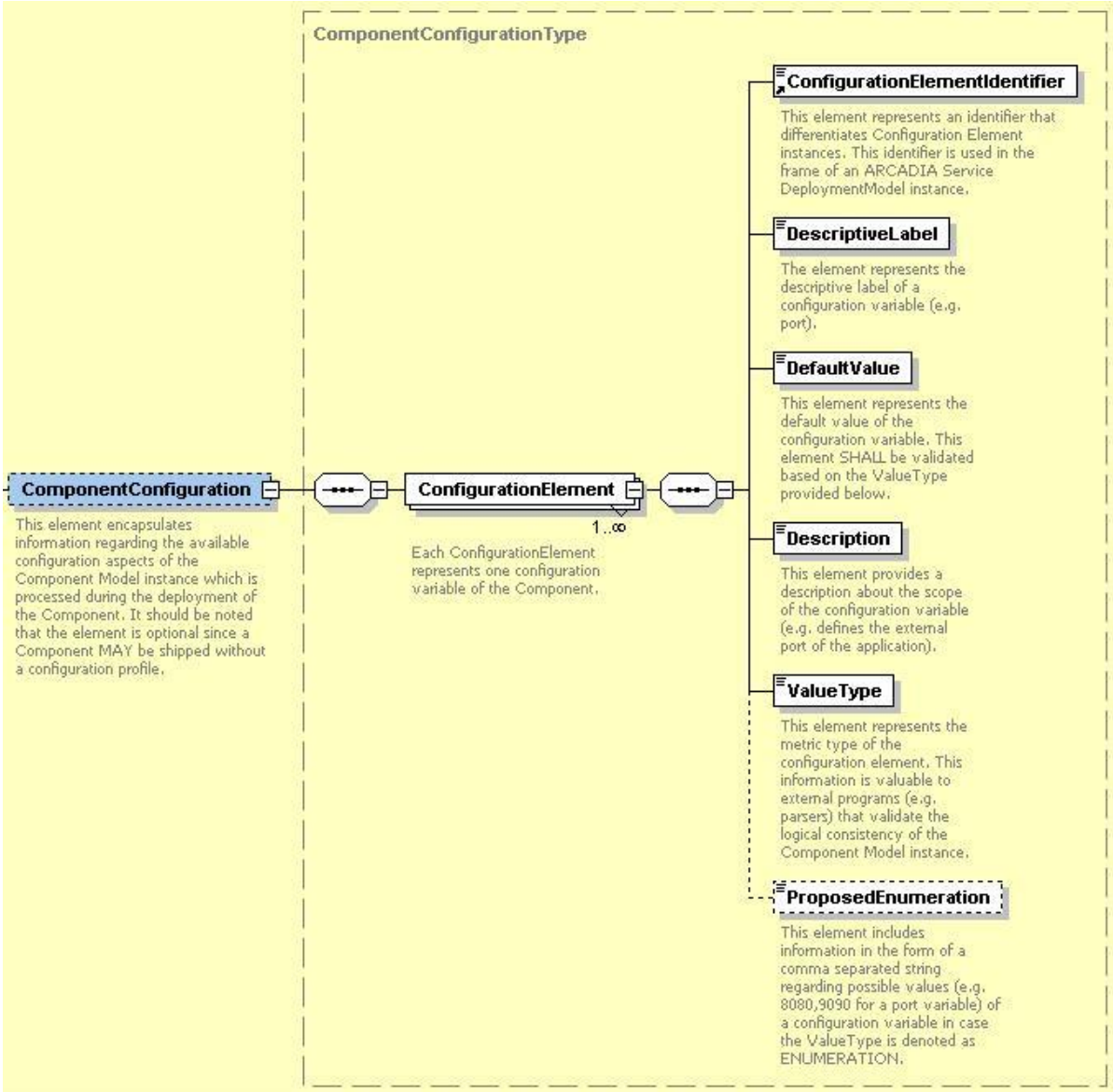


Figure 3-3: Overview of 'ComponentConfiguration' element

The next element that should be clarified is '*Requirements*'. Requirements represent the parameters that should be interpreted as constraints when the "ARCADIA Smart Controller" selects the IaaS resources that should be used per Component in the frame of one Service Graph deployment. As depicted in Figure 3-4 requirements are mainly distinguished to resource-related requirements (e.g. CPU speed, number of Cores) and hosting-related requirements (e.g. Operating system of the VM).

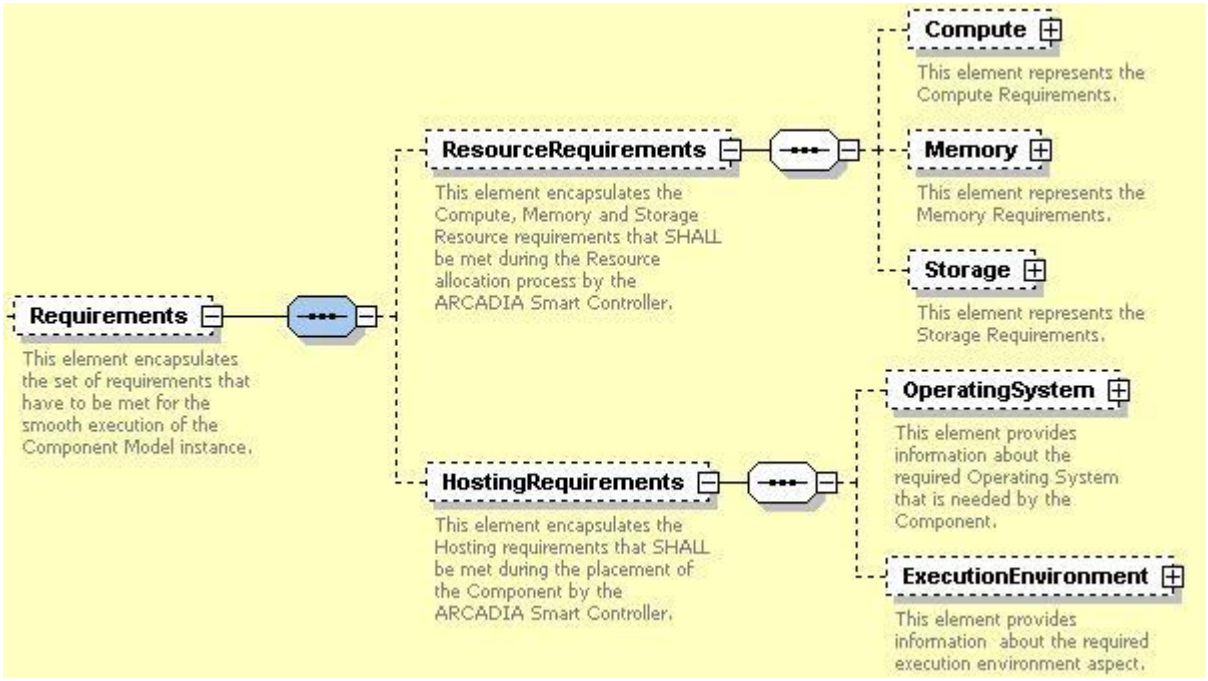


Figure 3-4: Overview of 'Requirements' element

The next element is 'Distribution' (Figure 3-5) which defines the physical location of the Component.

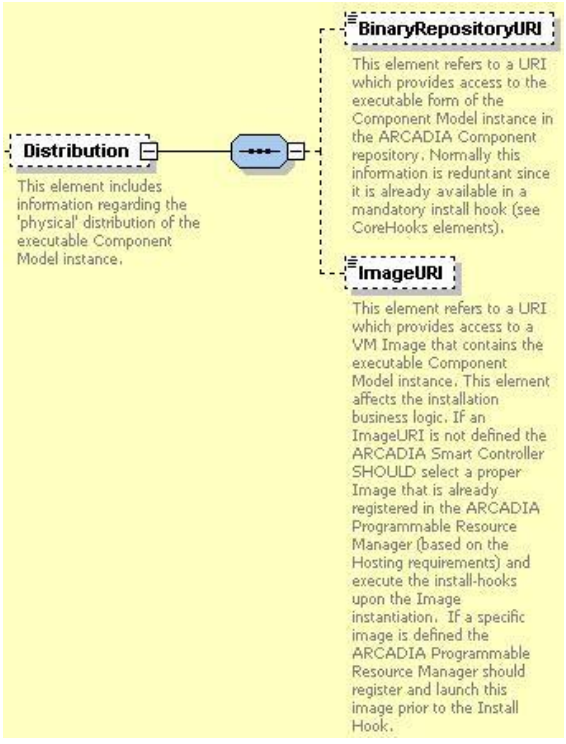


Figure 3-5: Overview of 'Distribution' element

As it is depicted, there are two options for distributing the executable Compoment. The one is through an Image repository which contains a pre-bundled Image VM for the component and the other is

through a specific application repository. In the latter case the protocol for accessing the repository should be defined in the frame of the BinaryRepositoryURI.

It could be argued that the most crucial element of the Component Model is the '*ExposedMicroServices*' element (see Figure 3-6). This element along with its reciprocal one ('*RequiredMicroServicces*') provides descriptive information regarding the interfaces that are exposed/consumed by one MicroService.

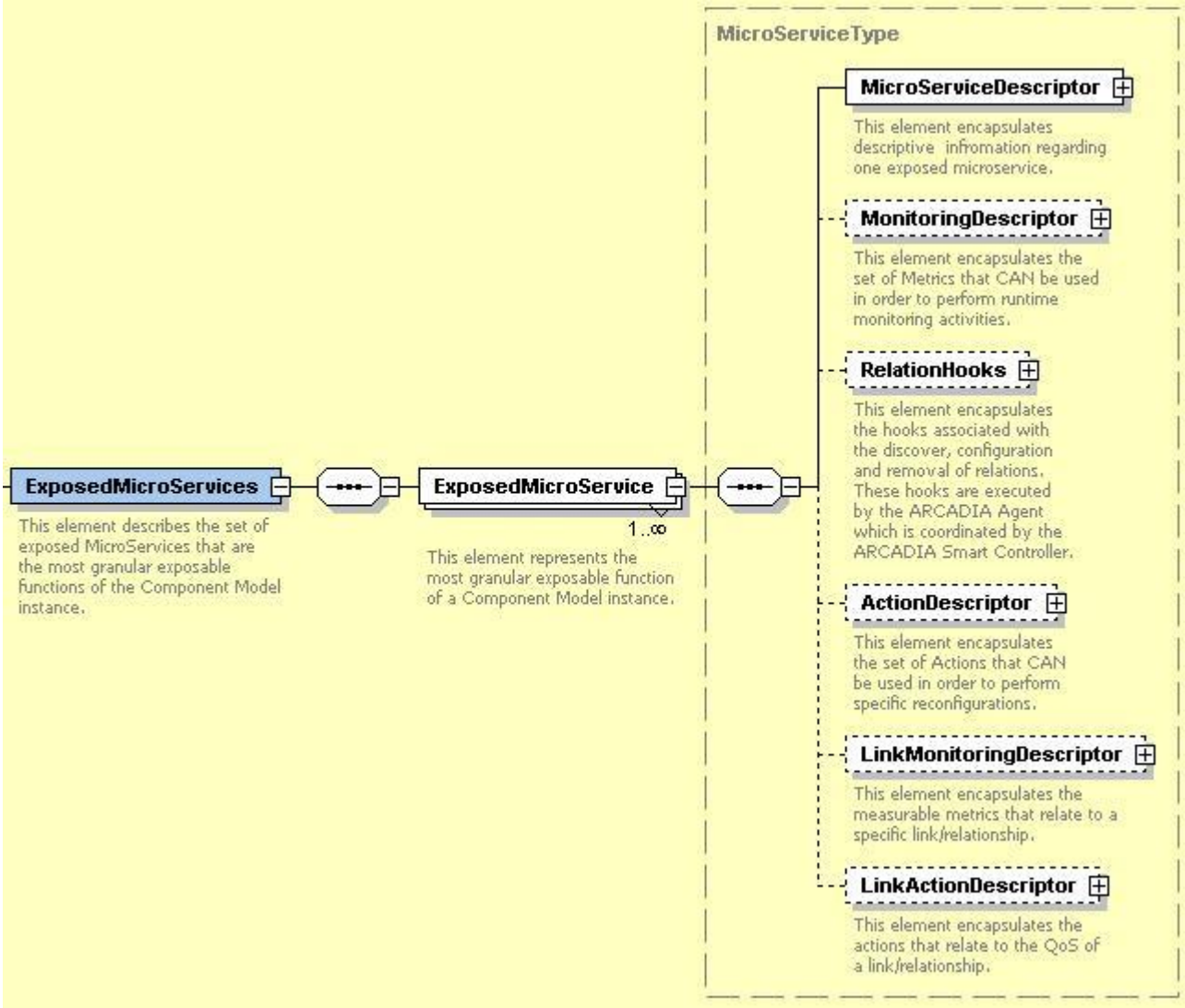


Figure 3-6: Overview of 'ExposedMicroServices' element

As depicted, one Component may expose many MicroServices. It should be noted that MicroServices and interfaces are conceptually equivalent. Each MicroService contains one descriptor ('*MicroServiceDescriptor*' element) which contains information that uniquely identifies the MicroService within a Component (intra-Component) and among several Components (inter-Component). Uniqueness is a crucial aspect since only compatible interfaces can be combined in the frame of one Service Graph. As a result, there should be no ambiguity regarding the identification of one MicroService.

Beyond identification, another crucial aspect is monitoring. Each MicroService can be accompanied by a set of monitoring metrics that are indicative of the MicroService's performance. The element '*MonitoringDescriptor*' encapsulates these metrics since they are used by the ARCADIA Smart Controller. The final element which will be discussed is 'CoreHooks' (Figure 3-7).

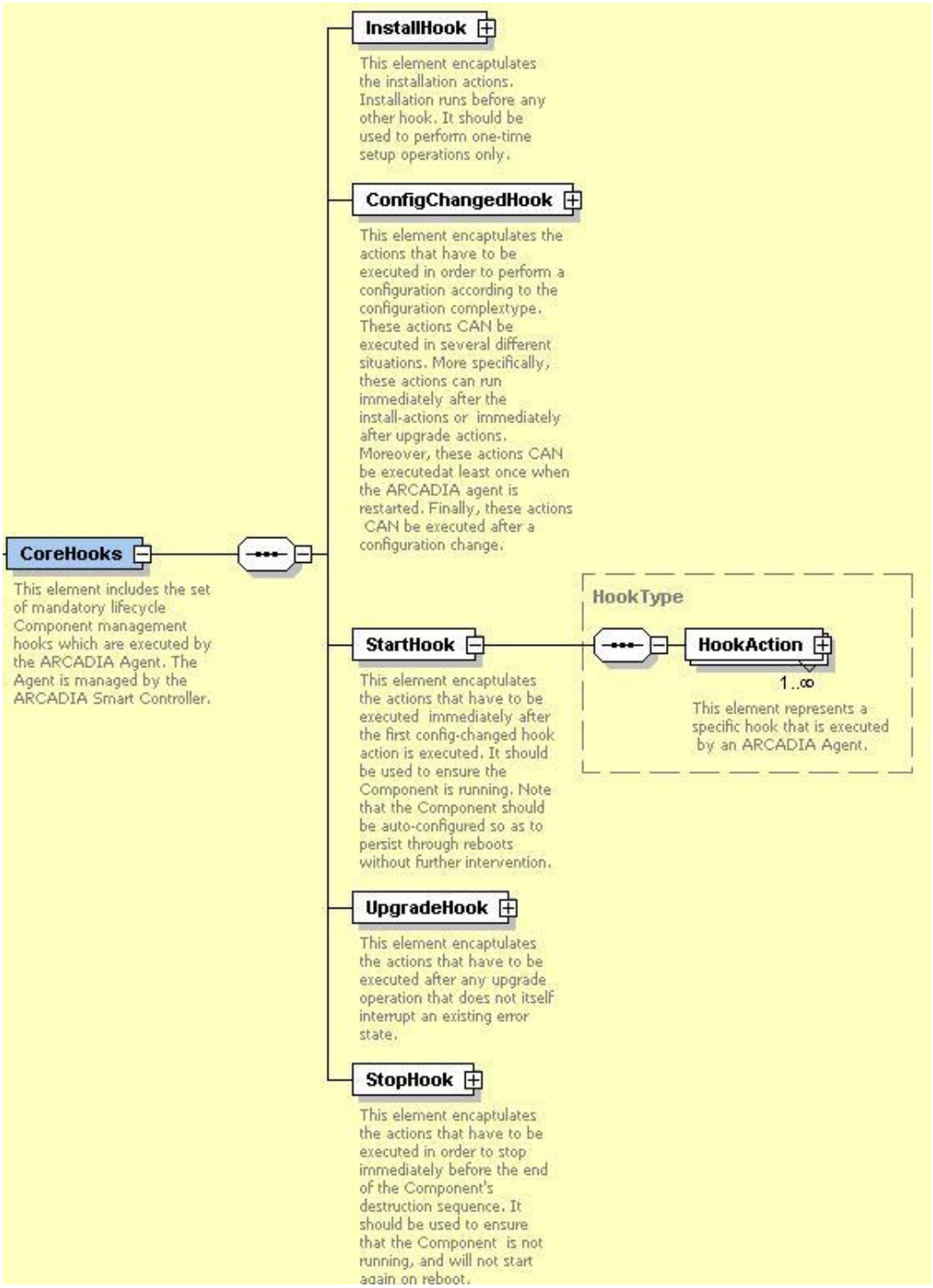


Figure 3-7: Overview of 'CoreHooks' element

'CoreHooks' element encapsulates a set of actions that must be supported by any Component Model instance. These actions are executed by a specific ARCADIA architectural component which is addressed as the ARCADIA Deployment Agent. This Agent implements a specific signaling protocol that is coordinated by a sub-component of the Smart Controller. Any Component Model instance provides a set of actions for a) installation (see 'InstallHook'), b) change of configuration (see 'ConfigChangedHook'), c) initiation of the service (see 'startHook'), d) update of the service (see 'updateHook') and e) termination of the service (see 'stopHook'). This type of modeling provides full backward compatibility with Canonical Juju[1] components (a.k.a. Charms).

At this point, it should be clarified that ARCADIA component model supports two types of hooks; i) the 'coreHooks' that relate to the component lifecycle and ii) the 'relationHooks' that are attached to the Component's dependencies (see Figure 3-8). The 'relationHooks' element encapsulates the hooks associated with the a) discovery (see 'RelationJoinedHook' element), b) configuration(see 'RelationChangedHook' element) and c) removal of relations(see 'RelationDepartedHook' and 'RelationBrokenHook' elements). These hooks are executed by the ARCADIA Agent which is coordinated by the ARCADIA Smart Controller.

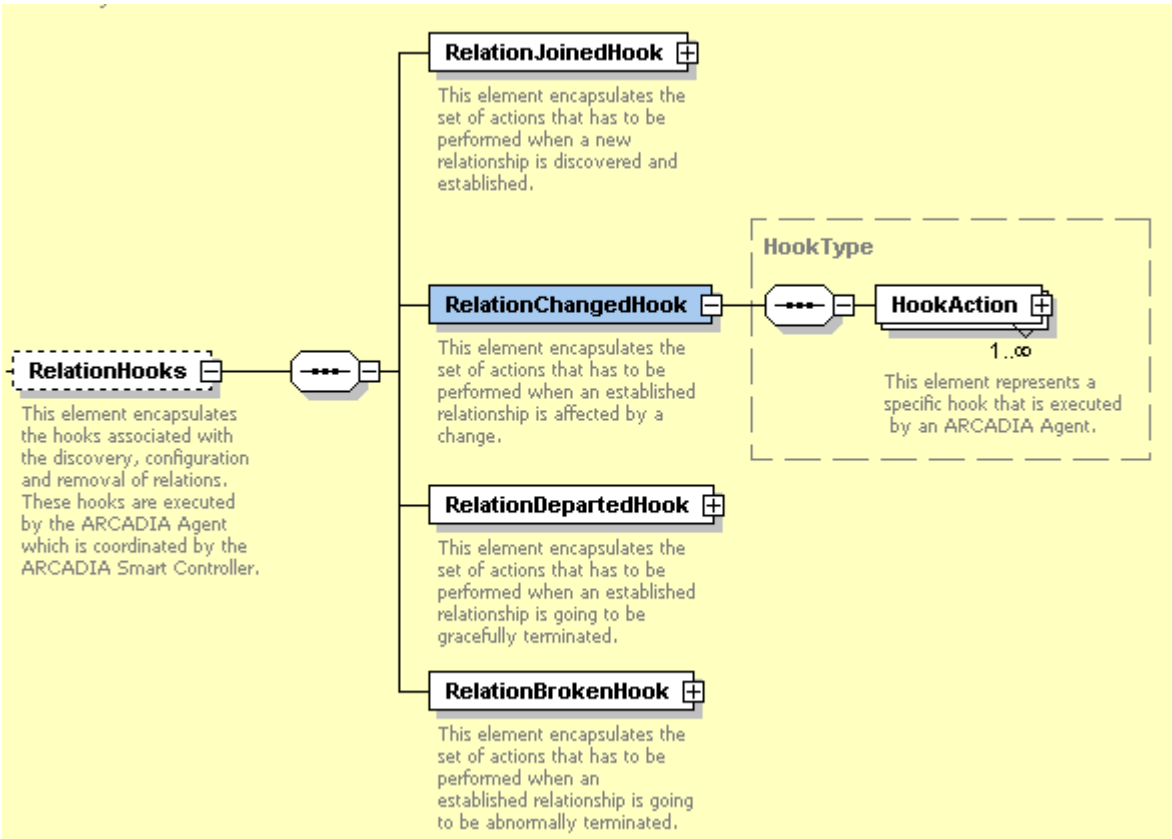


Figure 3-8: Overview of 'RelationHooks' element

A detailed analysis of the ARCADIA Component Model is provided in Annex II.

[1] https://jujucharms.com/

4 Overview of the ARCADIA Service Graph Model

4.1 Overview of the ARCADIA Service Graph Model

As already explained, many ARCADIA Component Models can be combined in order to create one ARCADIA Service Graph Model. A Service Graph Model is practically a directed graph (a.k.a. DG). Figure 4-1 provides an overview of the Service Graph Model schema.

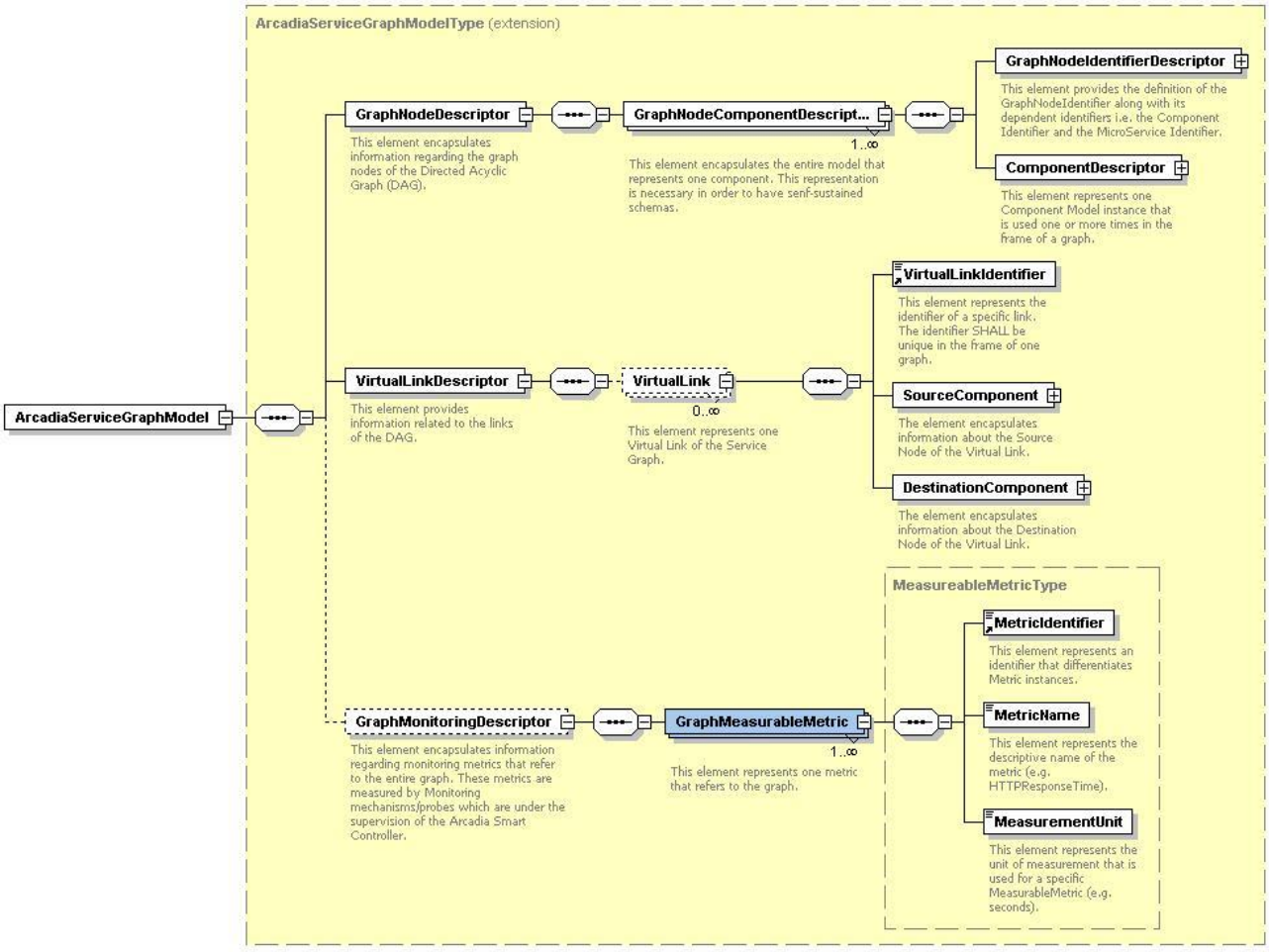


Figure 4-1: Overview of ARCADIA Service Graph Model

As depicted above, the Service Graph Model contains three main elements; the '*GraphNodeDescriptor*' element, the '*VirtualLinkDescriptor*' element and the '*GraphMonitoringDescriptor*' element. The the '*GraphNodeDescriptor*' element encapsulates information regarding the graph nodes of the Directed Graph. Additionally, the '*VirtualLinkDescriptor*' element provides information related to the links of the DG. Furthermore, the '*GraphMonitoringDescriptor*' element encapsulates information regarding monitoring metrics that refer to the entire graph. These metrics are measured by Monitoring mechanisms/probes which are under the supervision of the Arcadia Smart Controller. These elements will be further discussed in the next section.

4.2 Elaboration on the ARCADIA Service Graph Model

The scope of this section is to provide a bird's eye view of the elements that belong to the first level. Initially, the '*GraphNodeComponentDescriptor*' (see Figure 4-2) encapsulates the characteristics that represent one component in a graph. This representation is necessary in order to have self-sustained schemas. Each Component Model instance has one unique identifier (a.k.a. '*ComponentIdentifier*') in order to be distinguishable in the ARCADIA Component Repository. A specific Component Model instance may participate in one or more virtual links in the frame of a Service Graph. Each time a Component is used as a graph-node it is appointed a unique identifier (the '*GraphNodeIdentifier*') within the scope of the graph. This identifier provides the flexibility to perform many virtual links from the same Component instance or from different Component instances.

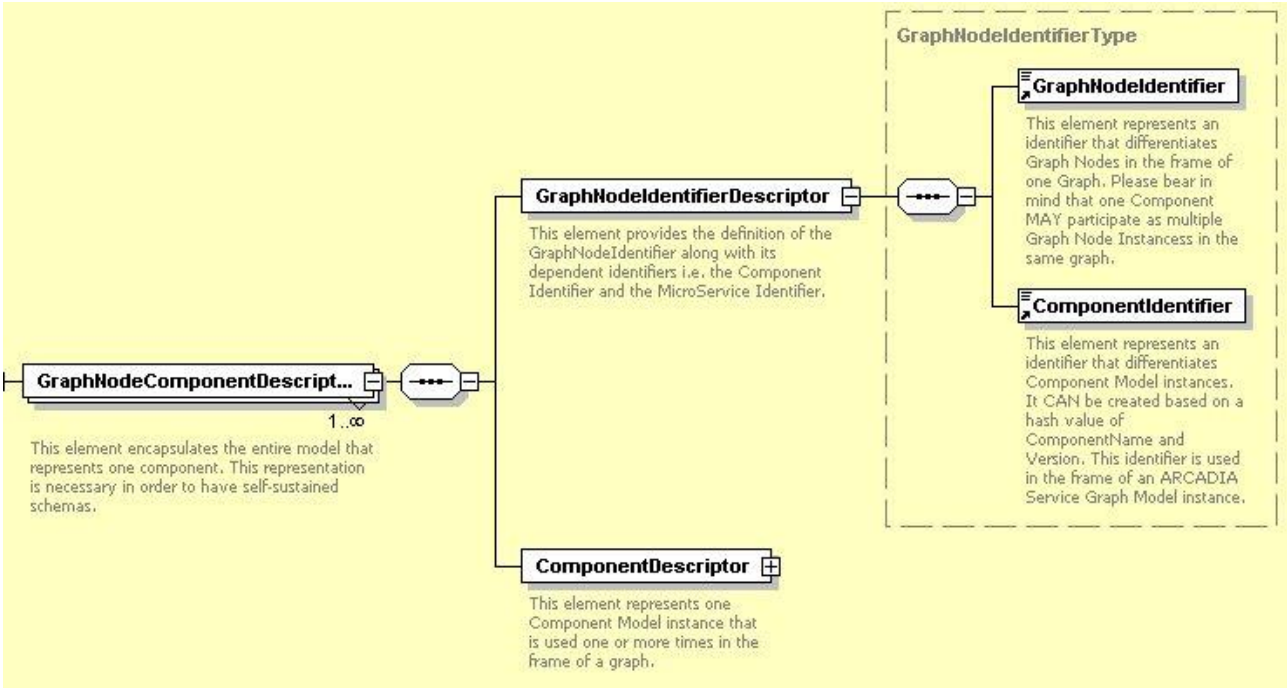


Figure 4-2: Overview of 'GraphNodeIdentifier' element

The '*ComponentDescriptor*' element is used in order to represent the set of the Component Models that have been used as graph nodes. It should be clarified that, although this is a real serialization overhead, it makes the Service Graph Model totally independent. In other words, one model-validator can assess the logical-correctness of the graph model without querying any repository.

On the other hand, one virtual link is modeled under the '*VirtualLink*' element (see Figure 4-3) and combines two graph nodes based on the identifier that has been discussed above. Since the graph is always directed, the source and the destination should be clearly distinguished. This is achieved through the respective '*SourceComponent*' and '*DestinationComponent*' elements. Both of these elements encapsulate one '*GraphNodeIdentifier*' and one '*MicroServiceIdentifier*'. The '*GraphNodeIdentifier*' element provides the identification of the Component instance that should be created while the '*MicroServiceIdentifier*' identifier provides the proper interface that should be chained in the frame of the virtual link. It should be clarified that the responsibility of validating the compatibility of the interfaces is delegated to a specific ARCADIA component which will assess the compatibility during design time.

Design-time validation of the entire graph is a highly critical aspect. In general, the ARCADIA Smart Controller should perform a service graph deployment only when a service graph is logically valid. The logical validation of a service graph super-exceeds the scope of this deliverable.

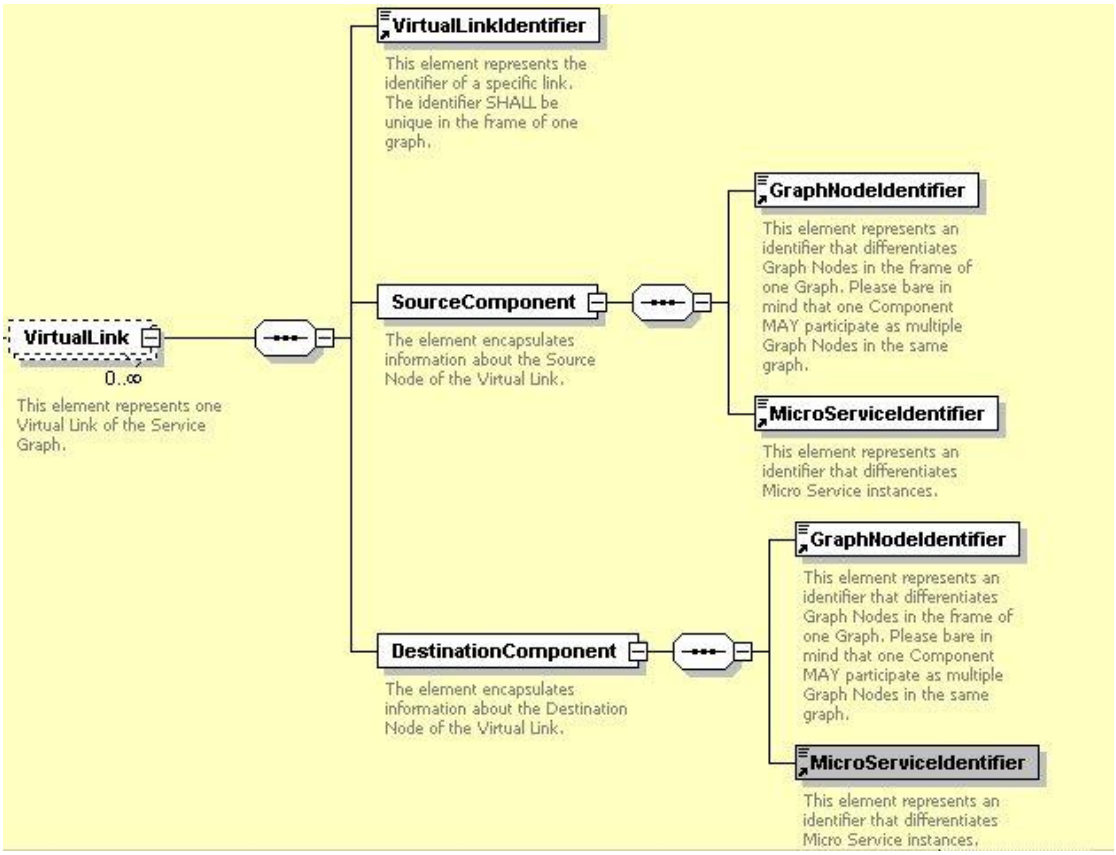


Figure 4-3: Overview of 'VirtualLinkDescriptor' element

Finally, in parallel with the measurable metrics that accompany a Component Model, such metrics exist in the Service Graph Model (see Figure 4-4). The substantial difference is that a measurable metric at the service graph level characterizes the entire graph (i.e. complex service) and not one node or virtual link.

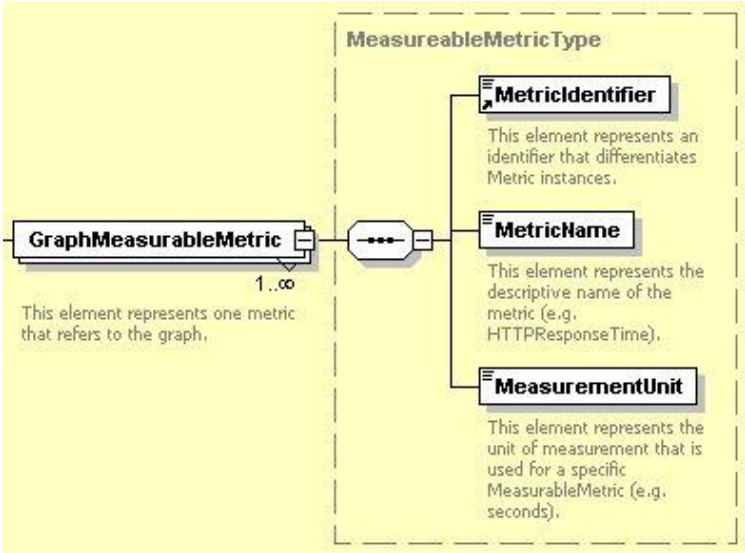


Figure 4-4: Overview of 'GraphMeasurableMetric' element

A detailed analysis of the ARCADIA Service Graph Model is provided in Annex II.

5 Overview of the ARCADIA Service Deployment Model

5.1 Overview of the ARCADIA Service Deployment Model

As already discussed, once a Service Graph instance is created any potential service provider can use ARCADIA in order to instantiate this graph (i.e. the complex service that is represented by this graph). Instantiation practically refers to the selection/allocation of a specific resource that has to be decided according to a specific policy. The definition of the policy model will be provided in the second version of the model. When the desired placement of each component is decided, a specific schema should track the association of the components' placement to the IaaS resources. This is the role of the Service Deployment Model. It should be clarified that the Service Deployment Model is agnostic to the placement process. In other words, it is irrelevant how a placement decision was taken (it could have been a manual process). Figure 5-1 provides an overview of the Service Deployment Model.

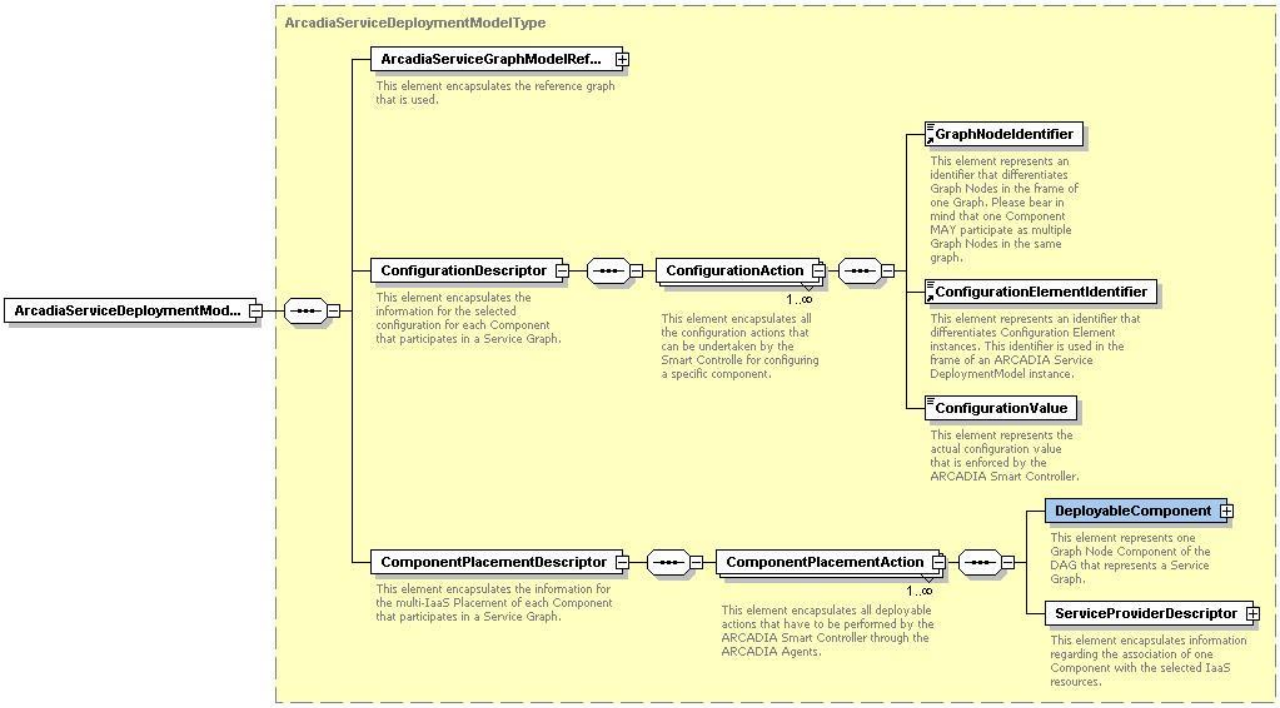


Figure 5-1: Overview of ARCADIA Service Deployment Model

As depicted above, an Arcadia Service Deployment Model consists of a '*ServiceGraphModelReference*' element that encapsulates the reference graph that is used for a specific deployment, a '*ConfigurationDescriptor*' element that encapsulates the information for the selected configuration for each Component that participates in a Service Graph and the '*ComponentPlacementDescriptor*' that encapsulates the information for the multi-IaaS Placement of each Component that participates in a Service Graph. These elements are further discussed in the next section.

5.2 Elaboration on the ARCADIA Service Deployment Model

The scope of this section is to provide a bird's eye view of the elements that belong to the first level. At first, the *ComponentPlacementDescriptor*' element consists of multiple 'ComponentPlacementAction' elements (see Figure 5-2). This element encapsulates all deployable actions that have to be performed by the ARCADIA Smart Controller by the ARCADIA Agents. Each action consists of one '*DeployableComponent*' element and one '*ServiceProviderDescriptor*' element. The '*DeployableComponent*' represents one graph node of the directed graph that represents a complex service. Therefore, it encapsulates a '*GraphNodeIdentifier*' element. As discussed in the previous chapter, the '*GraphNodeIdentifier*' is the only unique identifier of a component within the scope of one service graph.

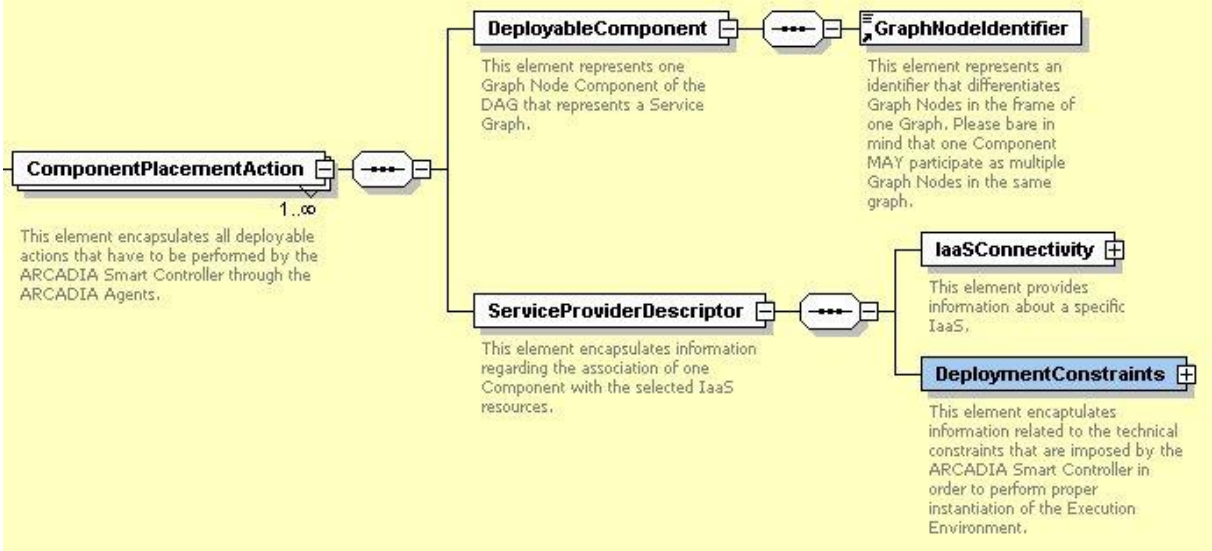


Figure 5-2: Overview of 'ComponentPlacementAction' element

The '*ServiceProviderDescriptor*' element consists of an '*IaaSConnectivity*' (see Figure 5-3) element and a '*DeploymentContraints*' element (see Figure 5-4).

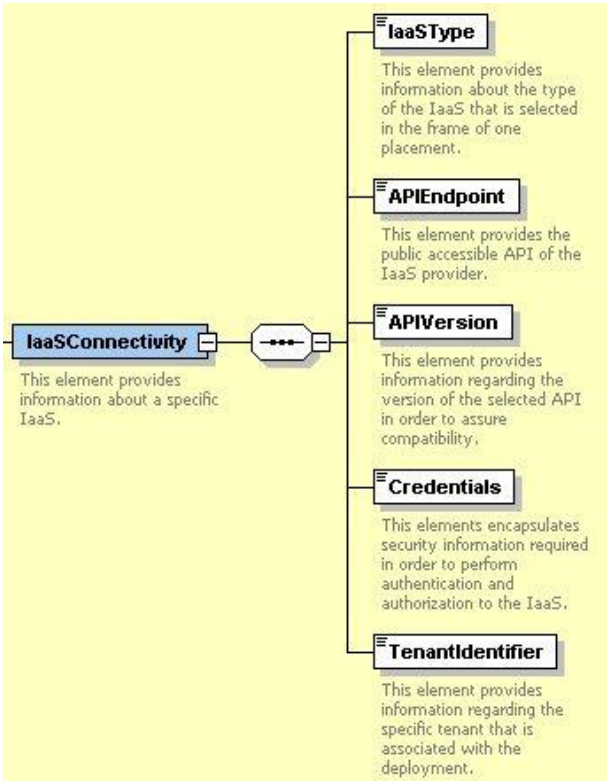


Figure 5-3: Overview of 'IaaSConnectivity' element

The '*IaaSConnectivity*' element provides information about a specific IaaS such as the type of the IaaS (e.g. OpenStack), the URI of the endpoint, the API version that will be used for interaction, the credentials that will be used and the tenant identifier that belongs to the service provider.

On the other hand, the '*DeploymentConstraints*' element encaptulates information related to the technical constraints that are imposed by the ARCADIA Smart Controller in order to perform proper instantiation of the Execution Environment.

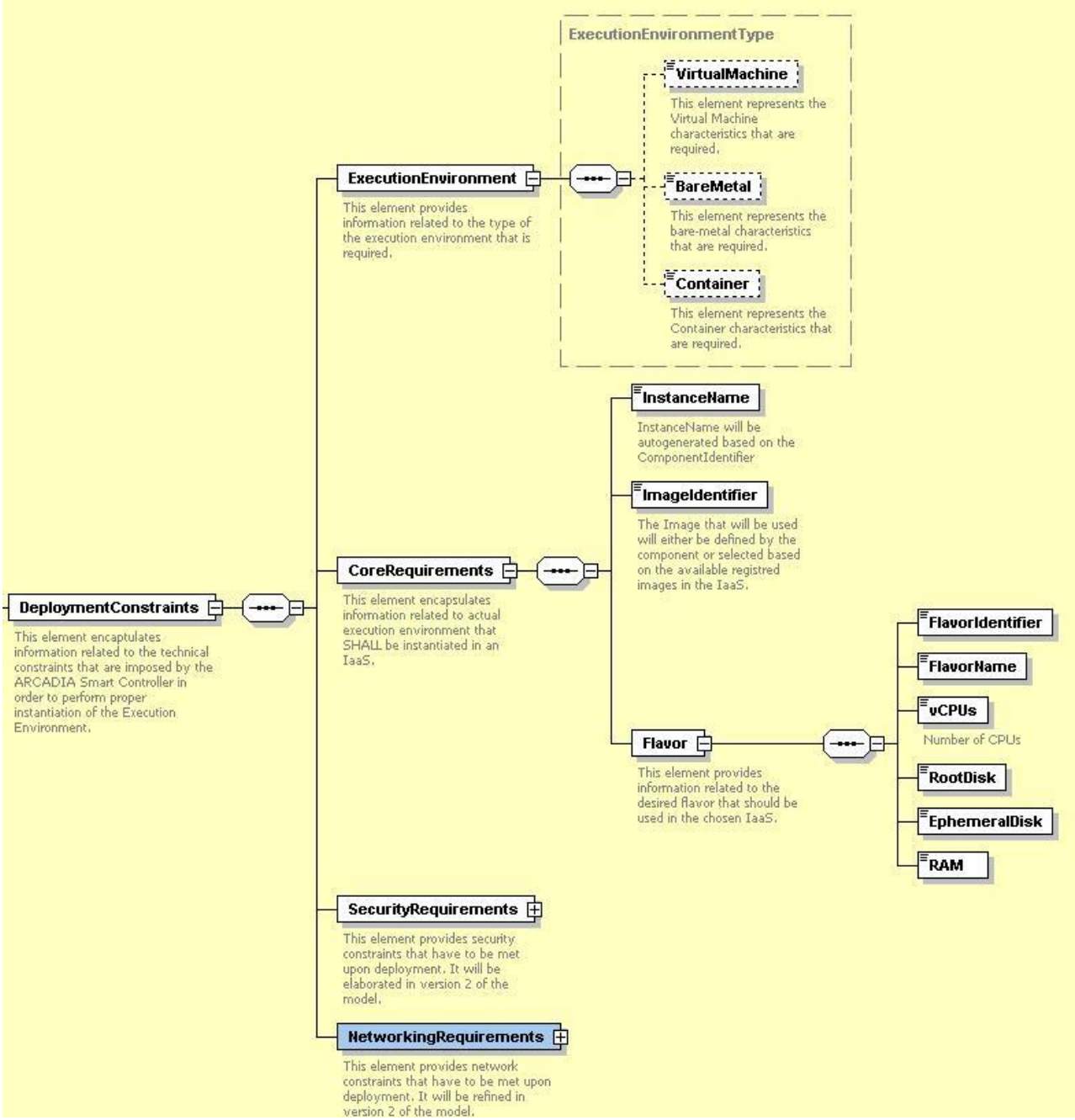


Figure 5-4: Overview of 'DeploymentConstraints' element

The '*DeploymentConstraints*' consists of four main elements. These include the '*ExecutionEnvironment*' element, the '*CoreRequirements*' elements, the '*SecurityRequirements*' elements and the '*NetworkingRequirements*' element. A detailed analysis of the ARCADIA Service Deployment Model is provided in Annex II.

6 Overview of the ARCADIA Service Runtime Model

6.1 Overview of the ARCADIA Service Runtime Model

An ARCADIA Service Runtime Model represents, conceptually, an instance of a deployed ARCADIA Service Graph which follows the 'rules' that are imposed by the ARCADIA Smart Controller. These rules are 'serialized' in the ARCADIA Service Deployment Model. Therefore, in a non-conceptual way, an ARCADIA Service Runtime Model is an extension of a Deployment Model. Figure 6-1 provides an overview of the model.

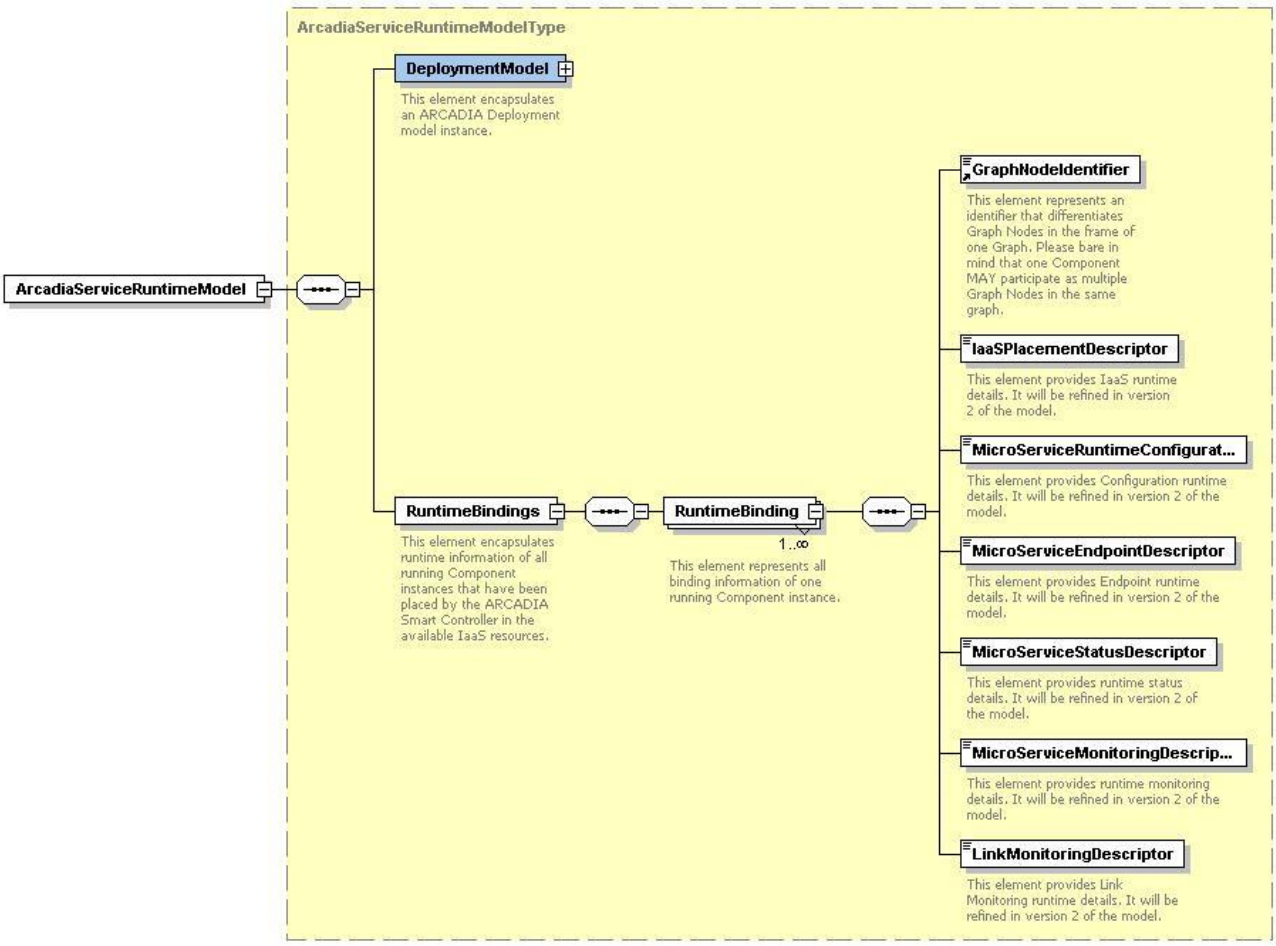


Figure 6-1: Overview of ARCADIA Service Runtime Model

The ARCADIA Service Runtime Model consists mainly of two elements; the '*DeploymentModel*' element and the '*RuntimeBindings*' element. The '*DeploymentModel*' encapsulates an entire ARCADIA Deployment Model instance while the '*RuntimeBindings*' element encapsulates runtime information of all running Component instances that have been placed by the ARCADIA Smart Controller in the available IaaS resources. The '*RuntimeBindings*' element consists of multiple '*RuntimeBinding*' elements.

Each '*RuntimeBinding*' element consists of seven elements; the '*GraphNodeIdentifier*' element that represents a graph node in a service graph, the '*IaaSPlacementDescriptor*' element provides IaaS

runtime details, the '*MicroServiceRuntimeConfiguration'* element that exposes the existing configuration of the component, the '*MicroServiceEndpointDescriptor*' element that exposes the running endpoint of a component interface, the '*MicroServiceStatusDescriptor*' element that provides the runtime status of a component, the '*MicroServiceMonitoringDescriptor*' element that provides information about the component instrumentation and the '*LinkMonitoringDescriptor*' element that provides virtual link monitoring runtime details.

The Service Runtime Model will be subjected to enhancements in the frame of the next release as it will be explained below. A detailed analysis of the ARCADIA Service Runtime Model is provided in Annex II.

7 Future Work on ARCADIA Context Model

The current deliverable introduced the first version of the ARCADIA Context Model. This model is a multi-faceted and multi-purpose model. Modeling artifacts are conceptually grouped in facets based on the HDA lifecycle phase that they support. In a nutshell, the documented facets include: a)the ARCADIA Component Model which conceptualizes the most granular executable unit that can be hosted in an execution container; b) the ARCADIA Service Graph Model which conceptualizes a directed graph that represents a complex service; c) the ARCADIA Service Deployment Model that represents a deployment plan of a specific service graph instance (that is generated by a Smart Controller) and finally d) the ARCADIA Service Runtime Model that represents the state of an entire service graph. The finalization of the ARCADIA Context Model is an evolutionary and iterative procedure. To this end, specific extensions and possible modifications will be performed in the forthcoming scheduled releases.

The concrete extensions that will be delivered include:

a) **The creation of the Source Code Annotation facet**: The existing facets cover specific HDA lifecycle phases (i.e. service instantiation, service execution etc.) yet they do not cover the service creation phase. In the frame of the ARCADIA project, specific set of annotations will be used in order to automate the creation of the Service Component Model. This is a crucial functionality since the added-value of the ARCADIA components will be leveraged by this automation. To this end, the metadata element, the exposed/required MicroService element and the monitoring metric element will be auto-generated.

b) **The elaboration on the IaaS resource advertisement**: The existing modeling artifacts deliberatively do not cover the various types of resources offered by IaaS middleware (e.g. Openstack, Docker). This will be performed in the next release.

c) **The creation of the scalability profiling facet**: One of the most critical issues of an ARCADIA Component Model is the conceptualization of its behavior when the execution container parameters change (e.g. more CPUs, disk or bandwidth is provided). This behavior affects the scalability capabilities (vertical or horizontal) of the ARCADIA Component Model instance. The conceptualization of the behavior will be performed using a normative profiling model.

d) **The creation of the ARCADIA Policy Modeling facet**: As already explained the ARCADIA Service Deployment Model is practically a deployment plan of an ARCADIA Service Graph Model that is generated by the ARCADIA Smart Controller. In order for the deployment plan to be produced specific optimization parameters have to be taken under consideration. These parameters will be expressed based on a normative policy model that will be created.

e) **The evolution of the ARCADIA Service Runtime Model**: Each IaaS provider provides diverse runtime parameters of a deployed application. These parameters may include execution, security, networking and monitoring aspects. The ARCADIA Service Runtime Model will be regularly updated based on these runtime parameters that are supported during the development process.

Finally, it should be clarified that, during the finalization of the architecture and during the development phase, the existing model may be subjected to minor changes based on additional requirements that may arise. In any case the up-to-date model will be published in the official website of the project: http://www.arcadia-framework.eu.

Annex I: References

[1] TOSCA Simple Profile for NFV Version 1.0 Committee Specification Draft 03 has been published by the TC.

[2] Deliverable 2.1, Description of Highly Distributed Applications and Programmable Infrastructure Requirements, ARCADIA project, http://www.arcadia-framework.eu/

[3] ETSI GS NFV-MAN001 v1.1.1, Network Functions Virtualization (NFV); Management and Orchestration

[4] Juju Orchestrator by Canonical Ltd. https://jujucharms.com/about

[5] Juju Charms https://jujucharms.com/

[6] PaaSage project http://www.paasage.eu/

[7] Object Management Group. Unified Modeling Language Specification. 2.4.1. http://www.omg.org/spec/UML/2.4.1/. Aug. 2011.

[8] Nicolas Ferry, Hui Song, Alessandro Rossini, Franck Chauvel and Arnor Solberg. "CloudMF: Applying MDE to Tame the Complexity of Managing Multi-Cloud Applications". In: UCC 2014: 7th IEEE/ACM International Conference on Utility and Cloud Computing. Ed. by Randall Bilof. IEEE Computer Society, 2014, pp. 269–277. doi: 10.1109/UCC.2014.36.

[9] Nicolas Ferry, Franck Chauvel, Alessandro Rossini, Brice Morin and Arnor Solberg. "Managing multi-cloud systems with CloudMF". In: NordiCloud 2013: 2nd Nordic Symposium on Cloud Computing and Internet Technologies. Ed. by Arnor Solberg, Muhammad Ali Babar, Marlon Dumas and Carlos E. Cuesta. ACM, 2013, pp. 38–45. isbn: 978-1-4503-2307-9. doi: 10.1145/2513534.2513542.

[10] Nicolas Ferry, Alessandro Rossini, Franck Chauvel, Brice Morin and Arnor Solberg. "Towards model-driven provisioning, deployment, monitoring, and adaptation of multi-cloud systems". In: CLOUD 2013: 6th IEEE International Conference on Cloud Computing. Ed. by Lisa O'Conner. IEEE Computer Society, 2013, pp. 887–894. isbn: 978-0-7695-5028-2. doi: 10. 1109/CLOUD.2013.133.

[11] Clément Quinton, Daniel Romero and Laurence Duchien. "Cardinalitybased feature models with constraints: a pragmatic approach". In: SPLC 2013: 17th International Software Product Line Conference. Ed. by Tomoji Kishi, Stan Jarzabek and Stefania Gnesi. ACM, 2013, pp. 162–166. isbn: 978-1-4503-1968-3. doi: 10.1145/2491627.2491638.

[12] Clément Quinton, Nicolas Haderer, Romain Rouvoy and Laurence Duchien. "Towards multi-cloud configurations using feature models and ontologies". In: MultiCloud 2013: International Workshop on Multi-cloud Applications and Federated Clouds. ACM, 2013, pp. 21–26. isbn: 978-1- 4503-2050-4. doi: 10.1145/2462326.2462332.

[13] Clément Quinton, Romain Rouvoy and Laurence Duchien. "Leveraging Feature Models to Configure Virtual Appliances". In: CloudCP 2012: 2nd International Workshop on Cloud Computing Platforms. ACM, 2012, 2:1–2:6. isbn: 978-1-4503-1161-8. doi: 10.1145/2168697.2168699.

[14] Keith Jeffery, Nikos Houssos, Brigitte Jörg and Anne Asserson. "Research information management: the CERIF approach". In: IJMSO 9.1 (2014),

pp. 5–14. doi: 10.1504/IJMSO.2014.059142.

[15] Kyriakos Kritikos, Jörg Domaschka and Alessandro Rossini. "SRL: A Scalability Rule Language for Multi-Cloud Environments". In: Cloud-Com 2014: 6th IEEE International Conference on Cloud Computing Technology and Science. Ed. by Juan E. Guerrero. IEEE Computer Society,

2014, pp. 1–9. isbn: 978-1-4799-4093-6. doi: 10 . 1109 / CloudCom .2014.170.

[16] Jörg Domaschka, Kyriakos Kritikos and Alessandro Rossini. “Towards a Generic Language for Scalability Rules”. In: Advances in Service-Oriented and Cloud Computing - Workshops of ESOCC 2014. Ed. by Guadalupe Ortiz and Cuong Tran. Vol. 508. Communications in Computer and Information Science. Springer, 2015, pp. 206–220. isbn: 978-3-319-14885- 4. doi: 10.1007/978-3-319-14886-1_19.

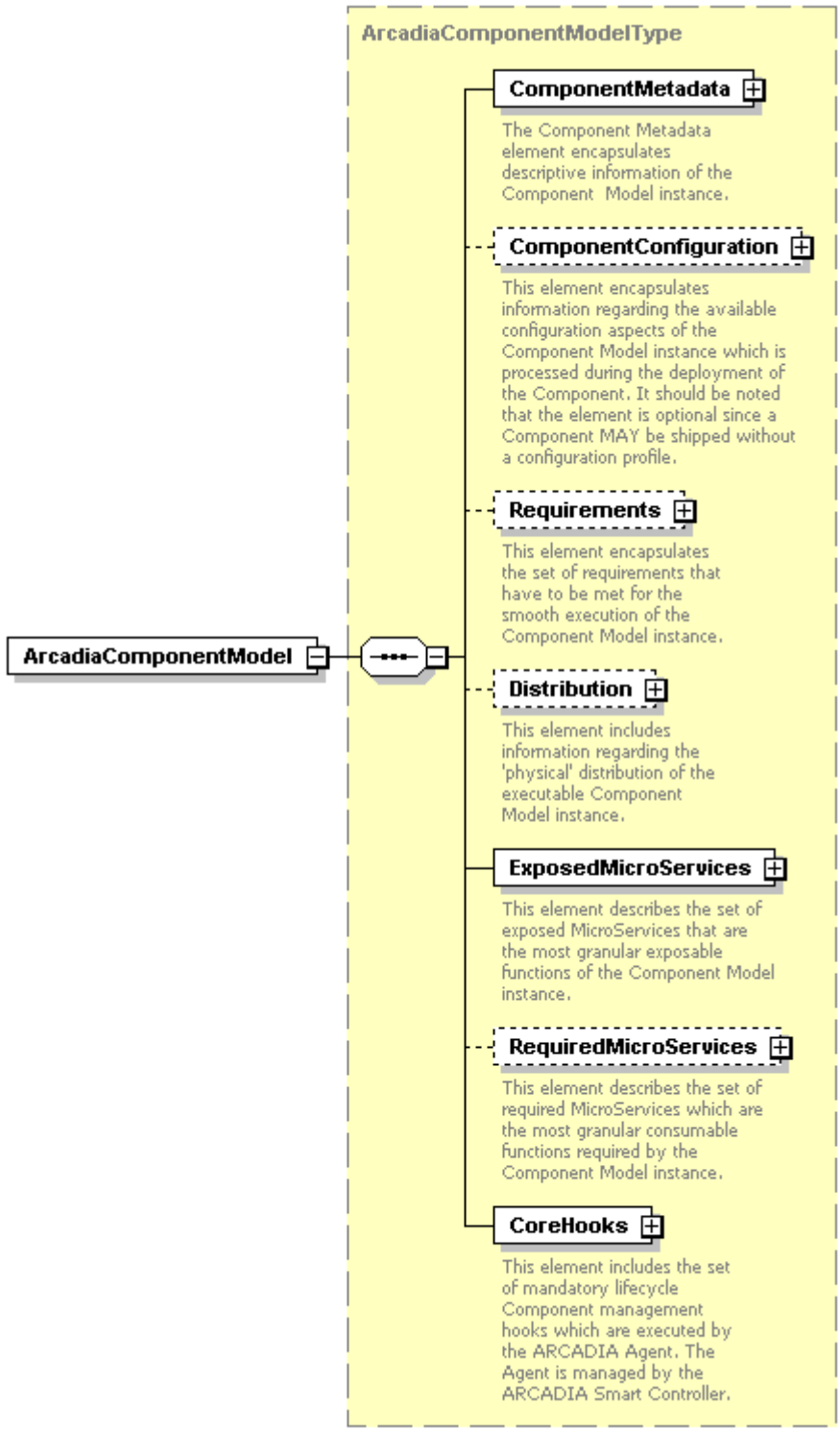
[17] PaaSage Project, Deliverable 2.1.1 http://www.paasage.eu/deliverables/31-deliverable-d2-1-1

Annex II: ARCADIA Context Model (v1.0) Documentation

This annex provides a complete guide to the first version of the ARCADIA Context Model. You can click on XSD Elements, Complex types and Simpletypes and navigate to the respective part of the documentation.

element **ArcadiaComponentModel**

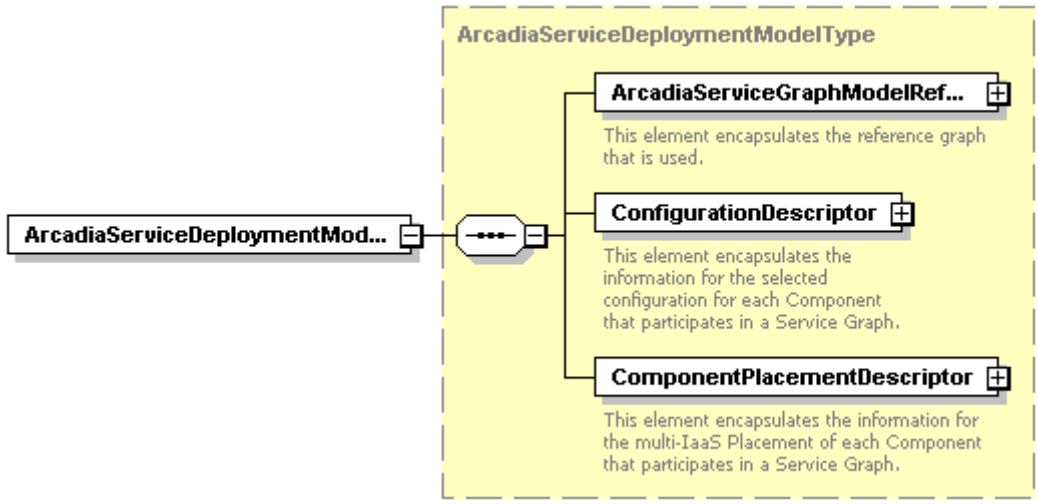
diagram



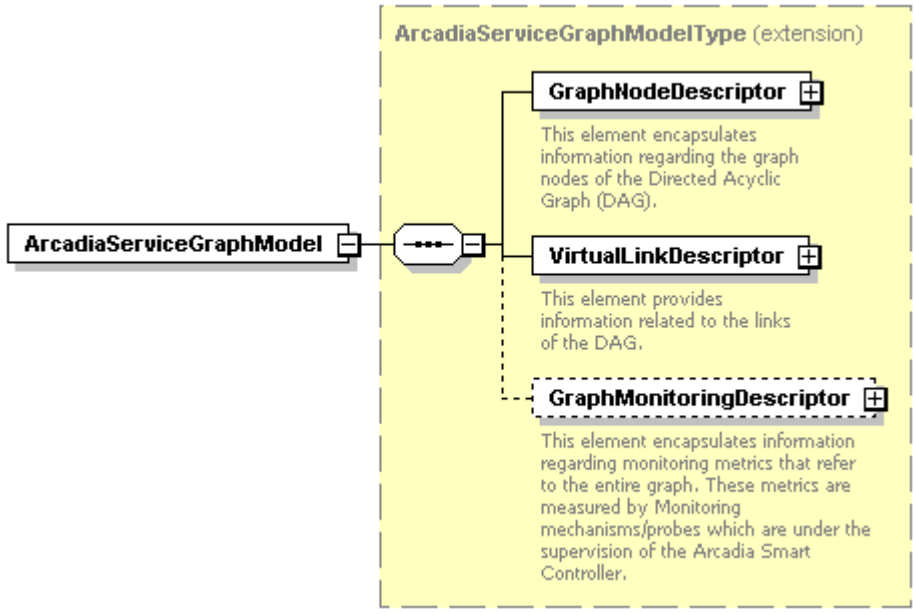
type	ArcadiaComponentModelType
properties	content complex
children	ComponentMetadata ComponentConfiguration Requirements Distribution ExposedMicroServices RequiredMicroServices CoreHooks

source	<xs:element name="ArcadiaComponentModel" type="ArcadiaComponentModelType"/>

element **ArcadiaServiceDeploymentModel**

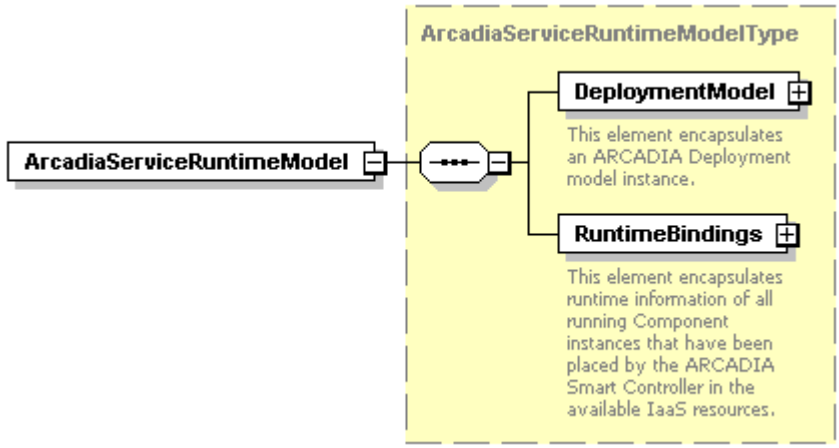
diagram	
type	ArcadiaServiceDeploymentModelType
properties	content complex
children	ArcadiaServiceGraphModelReference ConfigurationDescriptor ComponentPlacementDescriptor
source	<xs:element name="ArcadiaServiceDeploymentModel" type="ArcadiaServiceDeploymentModelType"/>

element **ArcadiaServiceGraphModel**

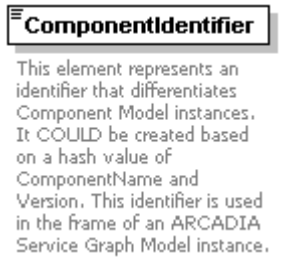
diagram	

type	extension of ArcadiaServiceGraphModelType
properties	content complex
children	GraphNodeDescriptor VirtualLinkDescriptor GraphMonitoringDescriptor
source	<xs:element name="ArcadiaServiceGraphModel"> <xs:complexType> <xs:complexContent> <xs:extension base="ArcadiaServiceGraphModelType"/> </xs:complexContent> </xs:complexType> </xs:element>

element **ArcadiaServiceRuntimeModel**

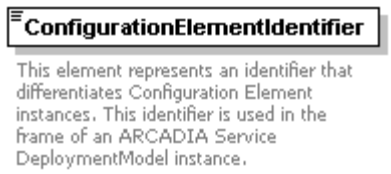
diagram	
type	ArcadiaServiceRuntimeModelType
properties	content complex
children	DeploymentModel RuntimeBindings
source	<xs:element name="ArcadiaServiceRuntimeModel" type="ArcadiaServiceRuntimeModelType"/>

element **ComponentIdentifier**

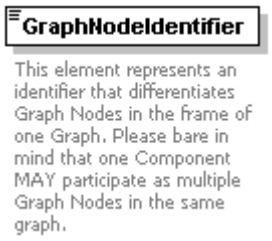
diagram	
type	**xs:string**
properties	content simple
used by	complexTypes ComponentMetadataType GraphNodeIdentifierType

annotation	documentation This element represents an identifier that differentiates Component Model instances. It COULD be created based on a hash value of ComponentName and Version. This identifier is used in the frame of an ARCADIA Service Graph Model instance.
source	<xs:element name="ComponentIdentifier" type="xs:string"> <xs:annotation> <xs:documentation>This element represents an identifier that differentiates Component Model instances. It COULD be created based on a hash value of ComponentName and Version. This identifier is used in the frame of an ARCADIA Service Graph Model instance.</xs:documentation> </xs:annotation> </xs:element>

element **ConfigurationElementIdentifier**

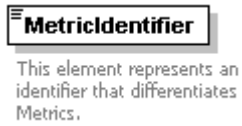
diagram	
used by	elements ArcadiaServiceDeploymentModelType/ConfigurationDescriptor/ConfigurationAction ComponentConfigurationType/ConfigurationElement
annotation	documentation This element represents an identifier that differentiates Configuration Element instances. This identifier is used in the frame of an ARCADIA Service DeploymentModel instance.
source	<xs:element name="ConfigurationElementIdentifier"> <xs:annotation> <xs:documentation>This element represents an identifier that differentiates Configuration Element instances. This identifier is used in the frame of an ARCADIA Service DeploymentModel instance.</xs:documentation> </xs:annotation> </xs:element>

element **GraphNodeIdentifier**

diagram	
used by	elements ArcadiaServiceDeploymentModelType/ConfigurationDescriptor/ConfigurationAction ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction/DeployableComponent ArcadiaServiceGraphModelType/VirtualLinkDescriptor/VirtualLink/DestinationComponent ArcadiaServiceRuntimeModelType/RuntimeBindings/RuntimeBinding ArcadiaServiceGraphModelType/VirtualLinkDescriptor/VirtualLink/SourceComponent complexType GraphNodeIdentifierType

annotation	documentation This element represents an identifier that differentiates Graph Nodes in the frame of one Graph. Please bare in mind that one Component MAY participate as multiple Graph Nodes in the same graph.
source	<xs:element name="GraphNodeIdentifier"> <xs:annotation> <xs:documentation>This element represents an identifier that differentiates Graph Nodes in the frame of one Graph. Please bare in mind that one Component MAY participate as multiple Graph Nodes in the same graph.</xs:documentation> </xs:annotation> </xs:element>

element **MetricIdentifier**

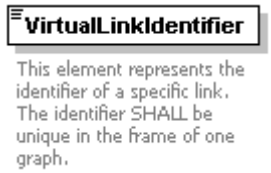
diagram	
type	**xs:string**
properties	content simple
used by	complexType **MeasureableMetricType**
annotation	documentation This element represents an identifier that differentiates Metrics.
source	<xs:element name="MetricIdentifier" type="xs:string"> <xs:annotation> <xs:documentation>This element represents an identifier that differentiates Metrics.</xs:documentation> </xs:annotation> </xs:element>

element **MicroServiceIdentifier**

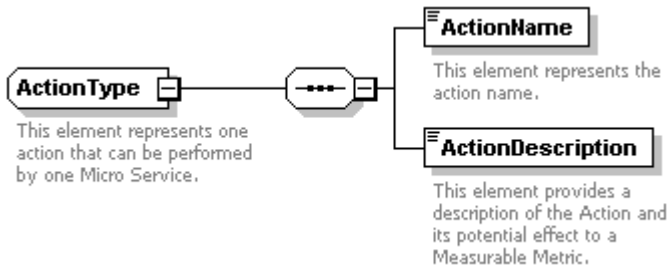
diagram	
type	**xs:string**
properties	content simple
used by	elements **ArcadiaServiceGraphModelType/VirtualLinkDescriptor/VirtualLink/DestinationComponent** **MicroServiceType/MicroServiceDescriptor** **ArcadiaServiceGraphModelType/VirtualLinkDescriptor/VirtualLink/SourceComponent**
annotation	documentation This element represents an identifier that differentiates Micro Service instances.
source	<xs:element name="MicroServiceIdentifier" type="xs:string"> <xs:annotation> <xs:documentation>This element represents an identifier that differentiates Micro Service instances.</xs:documentation> </xs:annotation>

	</xs:element>

element **VirtualLinkIdentifier**

diagram	
type	**xs:string**
properties	content simple
used by	element ArcadiaServiceGraphModelType/VirtualLinkDescriptor/VirtualLink
annotation	documentation This element represents the identifier of a specific link. The identifier SHALL be unique in the frame of one graph.
source	<xs:element name="VirtualLinkIdentifier" type="xs:string"> <xs:annotation> <xs:documentation>This element represents the identifier of a specific link. The identifier SHALL be unique in the frame of one graph.</xs:documentation> </xs:annotation> </xs:element>

complexType **ActionType**

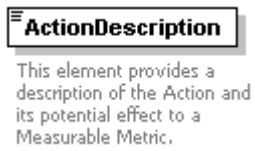
diagram	
children	ActionName ActionDescription
used by	elements MicroServiceType/ActionDescriptor/CustomActions/Action MicroServiceType/ActionDescriptor/QoSActions/QoSAction MicroServiceType/LinkActionDescriptor/QoSActions/QoSAction
annotation	documentation This element represents one action that can be performed by one Micro Service.
source	<xs:complexType name="ActionType"> <xs:annotation> <xs:documentation>This element represents one action that can be performed by one Micro Service.</xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="ActionName" type="xs:string"> <xs:annotation> <xs:documentation>This element represents the action name.</xs:documentation> </xs:annotation>

```
    </xs:element>
    <xs:element name="ActionDescription" type="xs:string">
     <xs:annotation>
      <xs:documentation>This element provides a description of the Action and its potential effect to a Measurable Metric.</xs:documentation>
     </xs:annotation>
    </xs:element>
   </xs:sequence>
</xs:complexType>
```

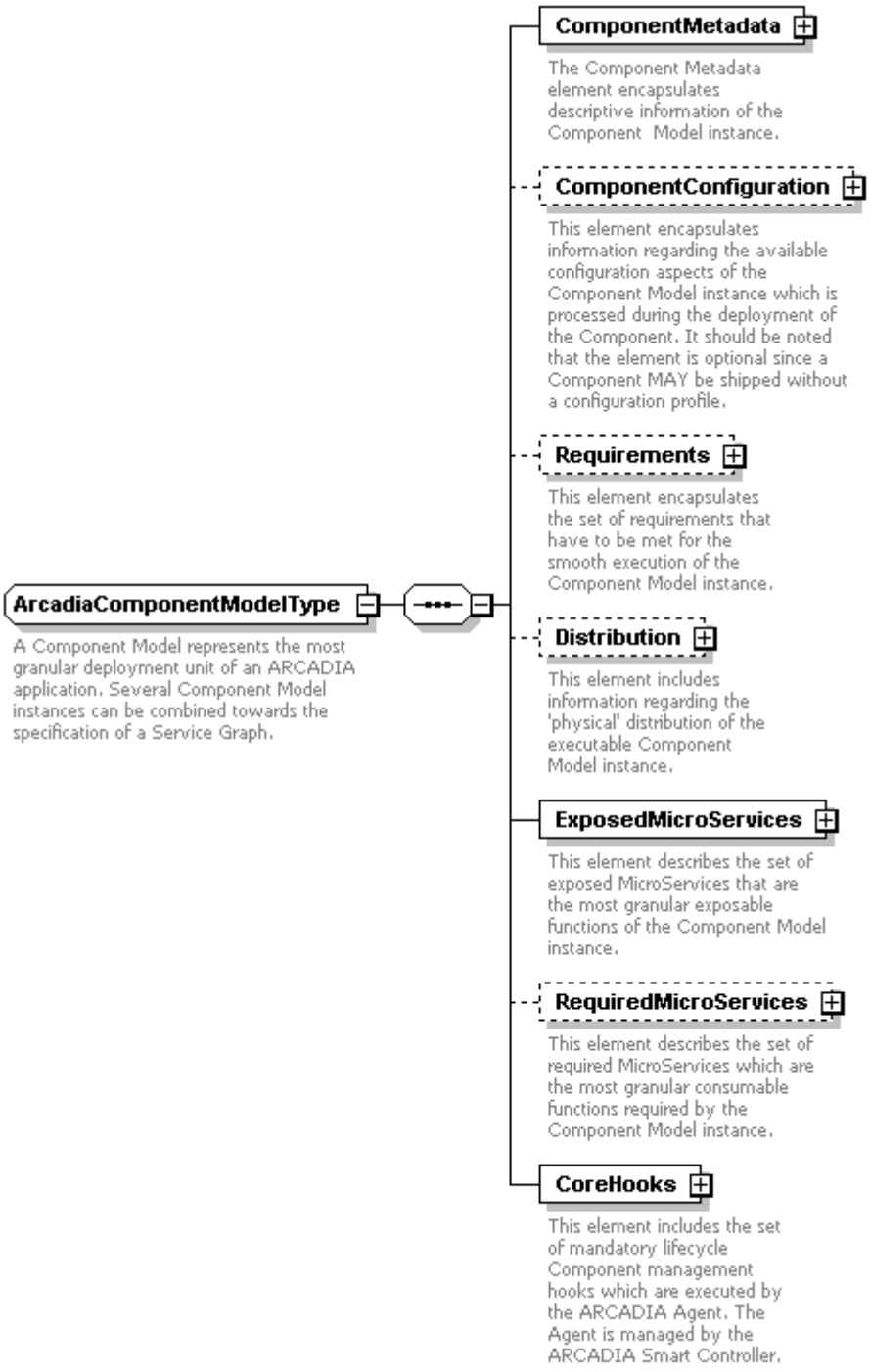
element **ActionType/ActionName**

diagram	
type	**xs:string**
properties	content simple
annotation	documentation This element represents the action name.
source	<xs:element name="ActionName" type="xs:string"> <xs:annotation> <xs:documentation>This element represents the action name.</xs:documentation> </xs:annotation> </xs:element>

element **ActionType/ActionDescription**

diagram	
type	**xs:string**
properties	content simple
annotation	documentation This element provides a description of the Action and its potential effect to a Measurable Metric.
source	<xs:element name="ActionDescription" type="xs:string"> <xs:annotation> <xs:documentation>This element provides a description of the Action and its potential effect to a Measurable Metric.</xs:documentation> </xs:annotation> </xs:element>

complexType **ArcadiaComponentModelType**

diagram	
children	ComponentMetadata ComponentConfiguration Requirements Distribution ExposedMicroServices RequiredMicroServices CoreHooks
used by	elements ArcadiaComponentModel ArcadiaServiceGraphModelType/GraphNodeDescriptor/GraphNodeComponentDescriptor/ComponentDescriptor
annotation	documentation A Component Model represents the most granular deployment unit of an ARCADIA application. Several Component Model instances can be combined towards the specification of a Service Graph.

source

```
<xs:complexType name="ArcadiaComponentModelType">
 <xs:annotation>
  <xs:documentation>A Component Model represents the most granular deployment unit of an ARCADIA application. Several Component Model instances can be combined towards the specification of a Service Graph.</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element name="ComponentMetadata" type="ComponentMetadataType">
   <xs:annotation>
    <xs:documentation>The Component Metadata element encapsulates descriptive information of the Component Model instance.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="ComponentConfiguration" type="ComponentConfigurationType" minOccurs="0">
   <xs:annotation>
    <xs:documentation>This element encapsulates information regarding the available configuration aspects of the Component Model instance which is processed during the deployment of the Component. It should be noted that the element is optional since a Component MAY be shipped without a configuration profile.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="Requirements" minOccurs="0">
   <xs:annotation>
    <xs:documentation>This element encapsulates the set of requirements that have to be met for the smooth execution of the Component Model instance.</xs:documentation>
   </xs:annotation>
   <xs:complexType>
    <xs:sequence>
     <xs:element name="ResourceRequirements" minOccurs="0">
      <xs:annotation>
       <xs:documentation>This element encapsulates the Compute, Memory and Storage Resource requirements that SHALL be met during the Resource allocation process by the ARCADIA Smart Controller.</xs:documentation>
      </xs:annotation>
      <xs:complexType>
       <xs:sequence>
        <xs:element name="Compute" minOccurs="0">
         <xs:annotation>
          <xs:documentation>This element represents the Compute Requirements.</xs:documentation>
         </xs:annotation>
         <xs:complexType>
          <xs:sequence>
           <xs:element name="minCPUSpeed" type="xs:double" minOccurs="0">
            <xs:annotation>
             <xs:documentation>This element represents the minimum CPU speed in MHz (e.g. 1200).</xs:documentation>
            </xs:annotation>
           </xs:element>
           <xs:element name="maxCPUSpeed" type="xs:double" minOccurs="0">
            <xs:annotation>
             <xs:documentation>This element represents the maximum CPU speed in MHz (e.g. 2400).</xs:documentation>
            </xs:annotation>
           </xs:element>
```

```
<xs:element name="minCores" type="xs:int" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the minimum number of Cores required for the execution (e.g. 2).</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="maxCores" type="xs:int" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the maximum number of Cores required for the execution (e.g. 10).</xs:documentation>
</xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="Memory" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the Memory Requirements.</xs:documentation>
</xs:annotation>
<xs:complexType>
<xs:sequence>
<xs:element name="minRAM" type="xs:int" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the minimum size of RAM (in MB) required for the execution (e.g. 512)</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="maxRAM" type="xs:int" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the maximum size of RAM (in MB) required for the execution (e.g. 2048)</xs:documentation>
</xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="Storage" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the Storage Requirements.</xs:documentation>
</xs:annotation>
<xs:complexType>
<xs:sequence>
<xs:element name="minStorage" type="xs:int" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the minimum size of data storage (in MB) required for the execution (e.g. 1000).</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="maxStorage" type="xs:int" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the maximum size of data storage (in MB) required for the execution (e.g. 4000).</xs:documentation>
</xs:annotation>
</xs:element>
```

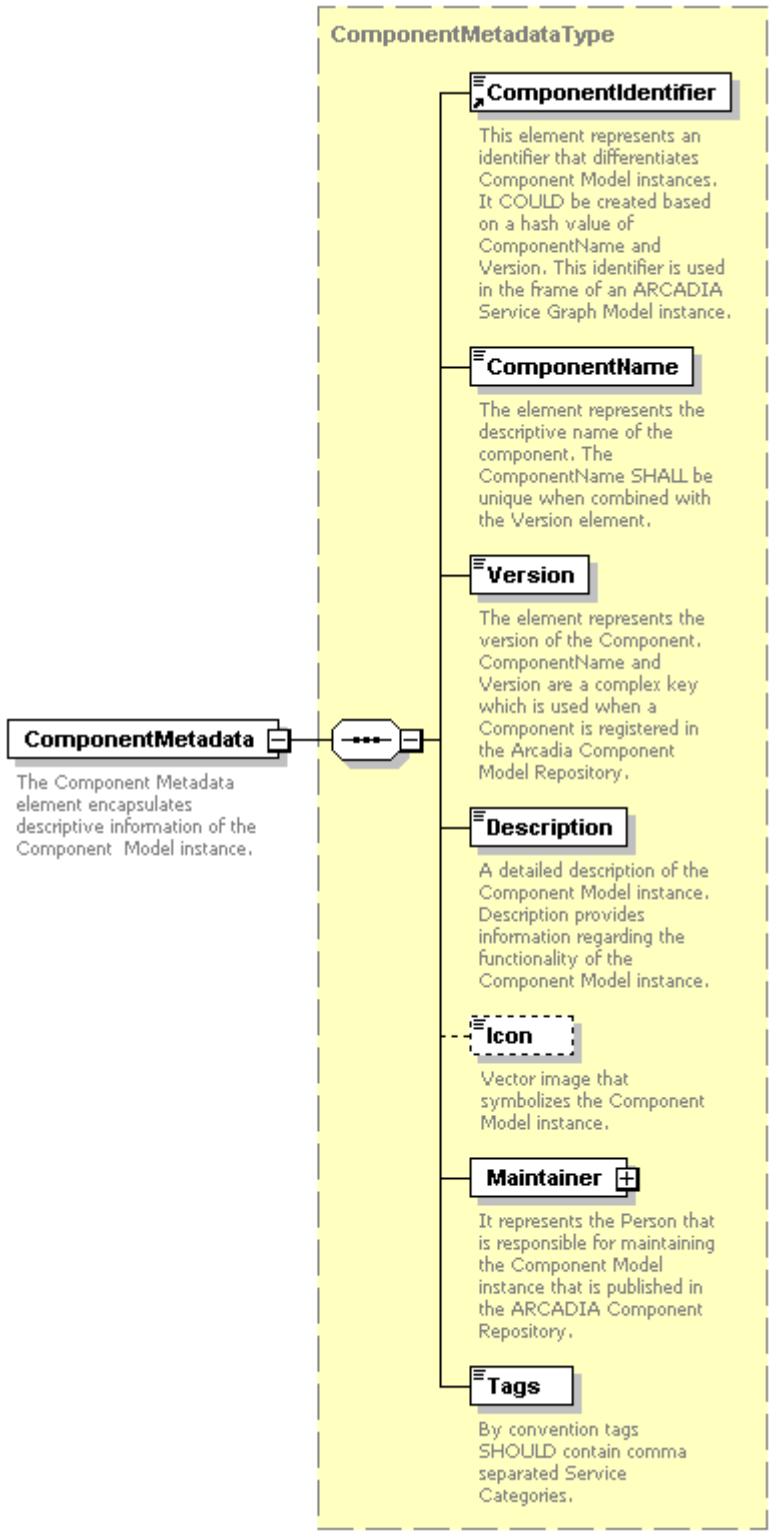
```
<xs:element name="StorageType" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the type of the storage element that is required.</xs:documentation>
</xs:annotation>
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:enumeration value="SSD"/>
<xs:enumeration value="AmazonS3"/>
<xs:enumeration value="HDD"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="HostingRequirements" minOccurs="0">
<xs:annotation>
<xs:documentation>This element encapsulates the Hosting requirements that SHALL be met during the placement of the Component by the ARCADIA Smart Controller.</xs:documentation>
</xs:annotation>
<xs:complexType>
<xs:sequence>
<xs:element name="OperatingSystem" minOccurs="0">
<xs:annotation>
<xs:documentation>This element provides information about the required Operating System that is needed by the Component.</xs:documentation>
</xs:annotation>
<xs:complexType>
<xs:sequence>
<xs:element name="Name" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the descriptive name of the Operating System.</xs:documentation>
</xs:annotation>
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:enumeration value="Ubuntu14.04LTSamd64"/>
<xs:enumeration value="Ubuntu14.04LTSi386"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
<xs:element name="ProcessorArchitecture" type="ProcessorArchitectureType">
<xs:annotation>
<xs:documentation>This element provides information regarding the type of the processor that is required.</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="OSType" minOccurs="0">
<xs:annotation>
<xs:documentation>This element provides information regarding the type of the Operating System that is required.</xs:documentation>
```

```
                </xs:annotation>
                <xs:simpleType>
                  <xs:restriction base="xs:string">
                    <xs:enumeration value="Linux"/>
                    <xs:enumeration value="Windows"/>
                  </xs:restriction>
                </xs:simpleType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="ExecutionEnvironment" type="ExecutionEnvironmentType" minOccurs="0">
          <xs:annotation>
            <xs:documentation>This element provides information about the required execution environment aspect.</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="Distribution" minOccurs="0">
 <xs:annotation>
  <xs:documentation>This element includes information regarding the 'physical' distribution of the executable Component Model instance.</xs:documentation>
 </xs:annotation>
 <xs:complexType>
  <xs:sequence>
   <xs:element name="BinaryRepositoryURI" type="xs:string" minOccurs="0">
    <xs:annotation>
     <xs:documentation>This element refers to a URI which provides access to the executable form of the Component Model instance in the ARCADIA Component repository. Normally this information is reduntant since it is already available in a mandatory install hook (see CoreHooks elements).</xs:documentation>
    </xs:annotation>
   </xs:element>
   <xs:element name="ImageURI" type="xs:string" minOccurs="0">
    <xs:annotation>
     <xs:documentation>This element refers to a URI which provides access to a VM Image that contains the executable Component Model instance. This element affects the installation business logic. If an ImageURI is not defined the ARCADIA Smart Controller SHOULD select a proper Image that is already registered in the ARCADIA Programmable Resource Manager (based on the Hosting requirements) and execute the install-hooks upon the Image instantiation. If a specific image is defined the ARCADIA Programmable Resource Manager should register and launch this image prior to the Install Hook.</xs:documentation>
    </xs:annotation>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="ExposedMicroServices">
 <xs:annotation>
  <xs:documentation>This element describes the set of exposed MicroServices that are the
```

```
most granular exposable functions of the Component Model instance.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
     <xs:sequence>
      <xs:element name="ExposedMicroService" type="MicroServiceType"
maxOccurs="unbounded">
       <xs:annotation>
        <xs:documentation>This element represents the most granular exposable function of a
Component Model instance.</xs:documentation>
       </xs:annotation>
      </xs:element>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="RequiredMicroServices" minOccurs="0">
    <xs:annotation>
     <xs:documentation>This element describes the set of required MicroServices which are the
most granular consumable functions required by the Component Model
instance.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
     <xs:sequence>
      <xs:element name="RequiredMicroService" type="MicroServiceType" minOccurs="0"
maxOccurs="unbounded">
       <xs:annotation>
        <xs:documentation>This element represents the most granular function that can be
consumed by a Component Model instance.</xs:documentation>
       </xs:annotation>
      </xs:element>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="CoreHooks">
    <xs:annotation>
     <xs:documentation>This element includes the set of mandatory lifecycle Component
management hooks which are executed by the ARCADIA Agent. The Agent is managed by the
ARCADIA Smart Controller.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
     <xs:sequence>
      <xs:element name="InstallHook" type="HookType">
       <xs:annotation>
        <xs:documentation>This element encaptulates the installation actions. Installation runs
before any other hook. It should be used to perform one-time setup operations
only.</xs:documentation>
       </xs:annotation>
      </xs:element>
      <xs:element name="ConfigChangedHook" type="HookType">
       <xs:annotation>
        <xs:documentation>This element encaptulates the actions that have to be executed in
order to perform a configuration according to the configuration complextype. These actions CAN
be executed in several different situations. More specifically, these actions can run immediately
after the install-actions or  immediately after upgrade actions. Moreover, these actions CAN be
executedat least once when the ARCADIA agent is restarted. Finally, these actions  CAN be
executed after a configuration change. </xs:documentation>
       </xs:annotation>
```

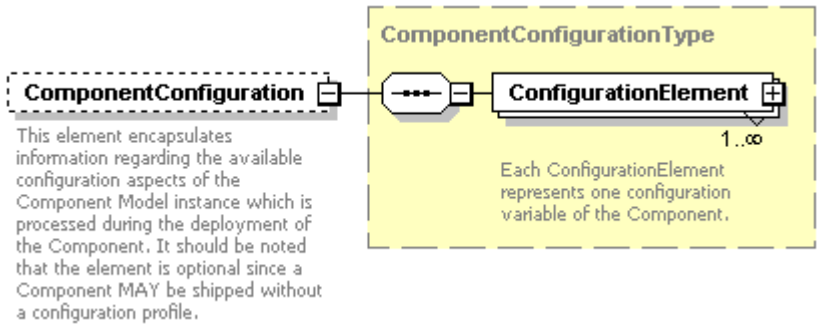
```
        </xs:element>
        <xs:element name="StartHook" type="HookType">
          <xs:annotation>
            <xs:documentation>This element encaptulates the actions that have to be executed immediately after the first config-changed hook action is executed. It should be used to ensure the Component is running. Note that the Component should be auto-configured so as to persist through reboots without further intervention.</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="UpgradeHook" type="HookType">
          <xs:annotation>
            <xs:documentation>This element encaptulates the actions that have to be executed after any upgrade operation that does not itself interrupt an existing error state.</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="StopHook" type="HookType">
          <xs:annotation>
            <xs:documentation>This element encaptulates the actions that have to be executed in order to stop immediately before the end of the Component's destruction sequence. It should be used to ensure that the Component is not running, and will not start again on reboot.</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
 </xs:sequence>
</xs:complexType>
```

element **ArcadiaComponentModelType/ComponentMetadata**

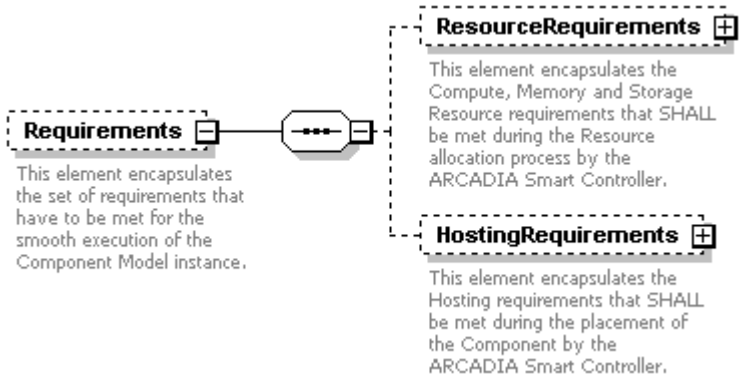
diagram	
type	ComponentMetadataType
properties	content complex
children	ComponentIdentifier ComponentName Version Description Icon Maintainer Tags

annotation	documentation The Component Metadata element encapsulates descriptive information of the Component Model instance.
source	<xs:element name="ComponentMetadata" type="ComponentMetadataType"> <xs:annotation> <xs:documentation>The Component Metadata element encapsulates descriptive information of the Component Model instance.</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaComponentModelType/ComponentConfiguration**

diagram	
type	**ComponentConfigurationType**
properties	minOcc 0 maxOcc 1 content complex
children	**ConfigurationElement**
annotation	documentation This element encapsulates information regarding the available configuration aspects of the Component Model instance which is processed during the deployment of the Component. It should be noted that the element is optional since a Component MAY be shipped without a configuration profile.
source	<xs:element name="ComponentConfiguration" type="ComponentConfigurationType" minOccurs="0"> <xs:annotation> <xs:documentation>This element encapsulates information regarding the available configuration aspects of the Component Model instance which is processed during the deployment of the Component. It should be noted that the element is optional since a Component MAY be shipped without a configuration profile.</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaComponentModelType/Requirements**

diagram	
properties	minOcc 0 maxOcc 1 content complex
children	ResourceRequirements HostingRequirements
annotation	documentation This element encapsulates the set of requirements that have to be met for the smooth execution of the Component Model instance.

source

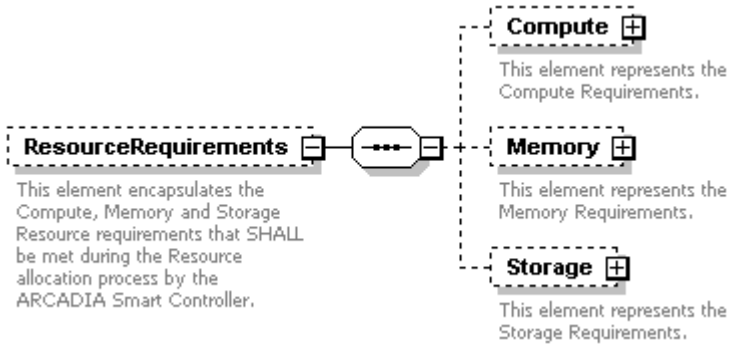
```
<xs:element name="Requirements" minOccurs="0">
 <xs:annotation>
  <xs:documentation>This element encapsulates the set of requirements that have to be met for the smooth execution of the Component Model instance.</xs:documentation>
 </xs:annotation>
 <xs:complexType>
  <xs:sequence>
   <xs:element name="ResourceRequirements" minOccurs="0">
    <xs:annotation>
     <xs:documentation>This element encapsulates the Compute, Memory and Storage Resource requirements that SHALL be met during the Resource allocation process by the ARCADIA Smart Controller.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
     <xs:sequence>
      <xs:element name="Compute" minOccurs="0">
       <xs:annotation>
        <xs:documentation>This element represents the Compute Requirements.</xs:documentation>
       </xs:annotation>
       <xs:complexType>
        <xs:sequence>
         <xs:element name="minCPUSpeed" type="xs:double" minOccurs="0">
          <xs:annotation>
           <xs:documentation>This element represents the minimum CPU speed in MHz (e.g. 1200).</xs:documentation>
          </xs:annotation>
         </xs:element>
         <xs:element name="maxCPUSpeed" type="xs:double" minOccurs="0">
          <xs:annotation>
           <xs:documentation>This element represents the maximum CPU speed in MHz (e.g. 2400).</xs:documentation>
          </xs:annotation>
         </xs:element>
```

```
<xs:element name="minCores" type="xs:int" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the minimum number of Cores required for the execution (e.g. 2).</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="maxCores" type="xs:int" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the maximum number of Cores required for the execution (e.g. 10).</xs:documentation>
</xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="Memory" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the Memory Requirements.</xs:documentation>
</xs:annotation>
<xs:complexType>
<xs:sequence>
<xs:element name="minRAM" type="xs:int" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the minimum size of RAM (in MB) required for the execution (e.g. 512)</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="maxRAM" type="xs:int" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the maximum size of RAM (in MB) required for the execution (e.g. 2048)</xs:documentation>
</xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="Storage" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the Storage Requirements.</xs:documentation>
</xs:annotation>
<xs:complexType>
<xs:sequence>
<xs:element name="minStorage" type="xs:int" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the minimum size of data storage (in MB) required for the execution (e.g. 1000).</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="maxStorage" type="xs:int" minOccurs="0">
<xs:annotation>
<xs:documentation>This element represents the maximum size of data storage (in MB) required for the execution (e.g. 4000).</xs:documentation>
</xs:annotation>
</xs:element>
```

```
            <xs:element name="StorageType" minOccurs="0">
              <xs:annotation>
                <xs:documentation>This element represents the type of the storage element that is required.</xs:documentation>
              </xs:annotation>
              <xs:simpleType>
                <xs:restriction base="xs:string">
                  <xs:enumeration value="SSD"/>
                  <xs:enumeration value="AmazonS3"/>
                  <xs:enumeration value="HDD"/>
                </xs:restriction>
              </xs:simpleType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="HostingRequirements" minOccurs="0">
  <xs:annotation>
    <xs:documentation>This element encapsulates the Hosting requirements that SHALL be met during the placement of the Component by the ARCADIA Smart Controller.</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element name="OperatingSystem" minOccurs="0">
        <xs:annotation>
          <xs:documentation>This element provides information about the required Operating System that is needed by the Component.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Name" minOccurs="0">
              <xs:annotation>
                <xs:documentation>This element represents the descriptive name of the Operating System.</xs:documentation>
              </xs:annotation>
              <xs:simpleType>
                <xs:restriction base="xs:string">
                  <xs:enumeration value="Ubuntu14.04LTSamd64"/>
                  <xs:enumeration value="Ubuntu14.04LTSi386"/>
                </xs:restriction>
              </xs:simpleType>
            </xs:element>
            <xs:element name="ProcessorArchitecture" type="ProcessorArchitectureType">
              <xs:annotation>
                <xs:documentation>This element provides information regarding the type of the processor that is required.</xs:documentation>
              </xs:annotation>
            </xs:element>
            <xs:element name="OSType" minOccurs="0">
              <xs:annotation>
                <xs:documentation>This element provides information regarding the type of the Operating System that is required.</xs:documentation>
```

```
                </xs:annotation>
                <xs:simpleType>
                  <xs:restriction                                   base="xs:string">
                    <xs:enumeration                                 value="Linux"/>
                    <xs:enumeration                                 value="Windows"/>
                  </xs:restriction>
                </xs:simpleType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element      name="ExecutionEnvironment"      type="ExecutionEnvironmentType"
minOccurs="0">
          <xs:annotation>
            <xs:documentation>This element provides information  about the required execution
environment                                                     aspect.</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
 </xs:sequence>
</xs:complexType>
</xs:element>
```

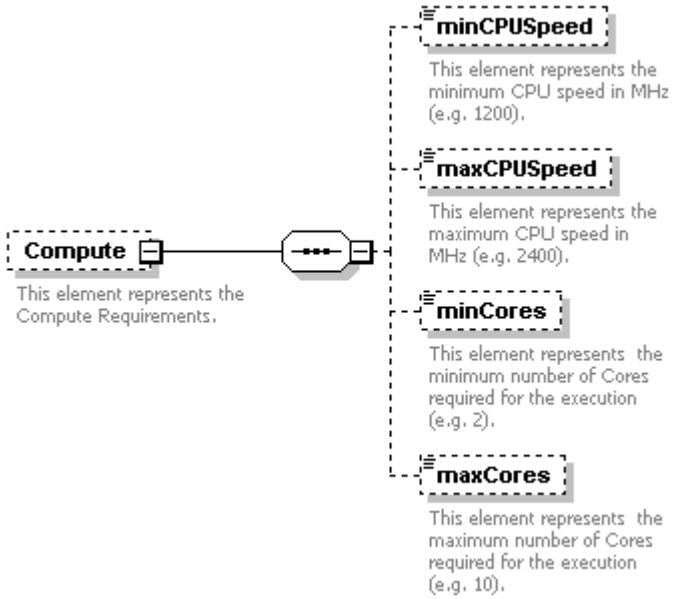
element **ArcadiaComponentModelType/Requirements/ResourceRequirements**

diagram	
properties	minOcc 0 maxOcc 1 content complex
children	**Compute Memory Storage**
annotation	documentation This element encapsulates the Compute, Memory and Storage Resource requirements that SHALL be met during the Resource allocation process by the ARCADIA Smart Controller.
source	<xs:element name="ResourceRequirements" minOccurs="0"> <xs:annotation> <xs:documentation>This element encapsulates the Compute, Memory and Storage Resource requirements that SHALL be met during the Resource allocation process by the ARCADIA Smart Controller.</xs:documentation> </xs:annotation> <xs:complexType>

```
<xs:sequence>
 <xs:element name="Compute" minOccurs="0">
  <xs:annotation>
   <xs:documentation>This element represents the Compute Requirements.</xs:documentation>
  </xs:annotation>
  <xs:complexType>
   <xs:sequence>
    <xs:element name="minCPUSpeed" type="xs:double" minOccurs="0">
     <xs:annotation>
      <xs:documentation>This element represents the minimum CPU speed in MHz (e.g. 1200).</xs:documentation>
     </xs:annotation>
    </xs:element>
    <xs:element name="maxCPUSpeed" type="xs:double" minOccurs="0">
     <xs:annotation>
      <xs:documentation>This element represents the maximum CPU speed in MHz (e.g. 2400).</xs:documentation>
     </xs:annotation>
    </xs:element>
    <xs:element name="minCores" type="xs:int" minOccurs="0">
     <xs:annotation>
      <xs:documentation>This element represents the minimum number of Cores required for the execution (e.g. 2).</xs:documentation>
     </xs:annotation>
    </xs:element>
    <xs:element name="maxCores" type="xs:int" minOccurs="0">
     <xs:annotation>
      <xs:documentation>This element represents the maximum number of Cores required for the execution (e.g. 10).</xs:documentation>
     </xs:annotation>
    </xs:element>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="Memory" minOccurs="0">
  <xs:annotation>
   <xs:documentation>This element represents the Memory Requirements.</xs:documentation>
  </xs:annotation>
  <xs:complexType>
   <xs:sequence>
    <xs:element name="minRAM" type="xs:int" minOccurs="0">
     <xs:annotation>
      <xs:documentation>This element represents the minimum size of RAM (in MB) required for the execution (e.g. 512)</xs:documentation>
     </xs:annotation>
    </xs:element>
    <xs:element name="maxRAM" type="xs:int" minOccurs="0">
     <xs:annotation>
      <xs:documentation>This element represents the maximum size of RAM (in MB) required for the execution (e.g. 2048)</xs:documentation>
     </xs:annotation>
    </xs:element>
   </xs:sequence>
  </xs:complexType>
```

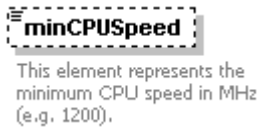
```
    </xs:element>
    <xs:element name="Storage" minOccurs="0">
     <xs:annotation>
      <xs:documentation>This element represents the Storage Requirements.</xs:documentation>
     </xs:annotation>
     <xs:complexType>
      <xs:sequence>
       <xs:element name="minStorage" type="xs:int" minOccurs="0">
        <xs:annotation>
         <xs:documentation>This element represents the minimum size of data storage (in MB) required for the execution (e.g. 1000).</xs:documentation>
        </xs:annotation>
       </xs:element>
       <xs:element name="maxStorage" type="xs:int" minOccurs="0">
        <xs:annotation>
         <xs:documentation>This element represents the maximum size of data storage (in MB) required for the execution (e.g. 4000).</xs:documentation>
        </xs:annotation>
       </xs:element>
       <xs:element name="StorageType" minOccurs="0">
        <xs:annotation>
         <xs:documentation>This element represents the type of the storage element that is required.</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
         <xs:restriction base="xs:string">
          <xs:enumeration value="SSD"/>
          <xs:enumeration value="AmazonS3"/>
          <xs:enumeration value="HDD"/>
         </xs:restriction>
        </xs:simpleType>
       </xs:element>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
   </xs:sequence>
  </xs:complexType>
</xs:element>
```

element **ArcadiaComponentModelType/Requirements/ResourceRequirements/Compute**

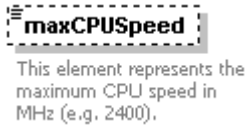
diagram	
properties	minOcc 0 maxOcc 1 content complex
children	**minCPUSpeed maxCPUSpeed minCores maxCores**
annotation	documentation This element represents the Compute Requirements.
source	<xs:element name="Compute" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the Compute Requirements.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="minCPUSpeed" type="xs:double" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the minimum CPU speed in MHz (e.g. 1200).</xs:documentation> </xs:annotation> </xs:element> <xs:element name="maxCPUSpeed" type="xs:double" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the maximum CPU speed in MHz (e.g. 2400).</xs:documentation> </xs:annotation> </xs:element> <xs:element name="minCores" type="xs:int" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the minimum number of Cores required for the execution (e.g. 2).</xs:documentation> </xs:annotation> </xs:element> <xs:element name="maxCores" type="xs:int" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the maximum number of Cores required for

	the execution (e.g. 10).</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType> </xs:element>

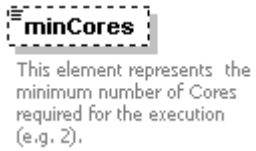
element
ArcadiaComponentModelType/Requirements/ResourceRequirements/Compute/minCPUSpeed

diagram	
type	**xs:double**
properties	minOcc 0 maxOcc 1 content simple
annotation	documentation This element represents the minimum CPU speed in MHz (e.g. 1200).
source	<xs:element name="minCPUSpeed" type="xs:double" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the minimum CPU speed in MHz (e.g. 1200).</xs:documentation> </xs:annotation> </xs:element>

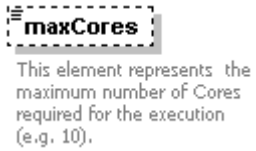
element
ArcadiaComponentModelType/Requirements/ResourceRequirements/Compute/maxCPUSpeed

diagram	
type	**xs:double**
properties	minOcc 0 maxOcc 1 content simple
annotation	documentation This element represents the maximum CPU speed in MHz (e.g. 2400).
source	<xs:element name="maxCPUSpeed" type="xs:double" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the maximum CPU speed in MHz (e.g. 2400).</xs:documentation> </xs:annotation> </xs:element>

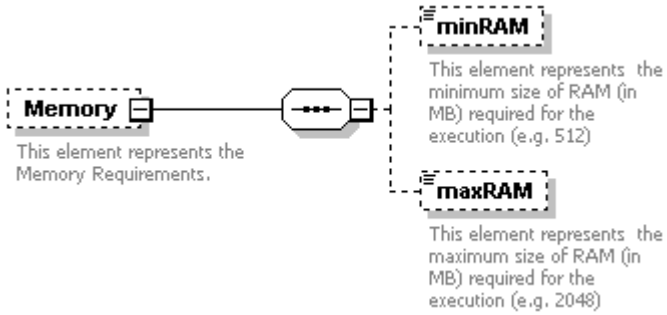
element **ArcadiaComponentModelType/Requirements/ResourceRequirements/Compute/minCores**

diagram	
type	**xs:int**
properties	minOcc 0 maxOcc 1 content simple
annotation	documentation This element represents the minimum number of Cores required for the execution (e.g. 2).
source	<xs:element name="minCores" type="xs:int" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the minimum number of Cores required for the execution (e.g. 2).</xs:documentation> </xs:annotation> </xs:element>

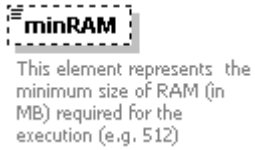
element **ArcadiaComponentModelType/Requirements/ResourceRequirements/Compute/maxCores**

diagram	
type	**xs:int**
properties	minOcc 0 maxOcc 1 content simple
annotation	documentation This element represents the maximum number of Cores required for the execution (e.g. 10).
source	<xs:element name="maxCores" type="xs:int" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the maximum number of Cores required for the execution (e.g. 10).</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaComponentModelType/Requirements/ResourceRequirements/Memory**

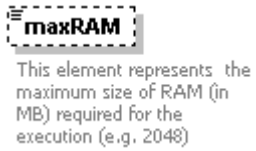
diagram	
properties	minOcc 0 maxOcc 1 content complex
children	**minRAM** **maxRAM**
annotation	documentation This element represents the Memory Requirements.
source	<xs:element name="Memory" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the Memory Requirements.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="minRAM" type="xs:int" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the minimum size of RAM (in MB) required for the execution (e.g. 512)</xs:documentation> </xs:annotation> </xs:element> <xs:element name="maxRAM" type="xs:int" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the maximum size of RAM (in MB) required for the execution (e.g. 2048)</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType> </xs:element>

element **ArcadiaComponentModelType/Requirements/ResourceRequirements/Memory/minRAM**

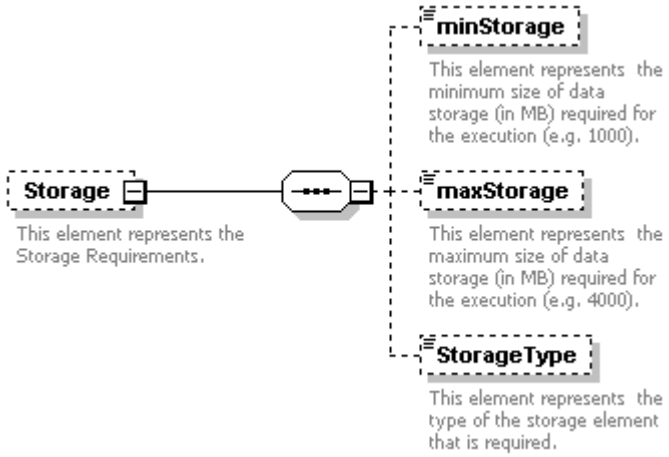
diagram	
type	**xs:int**
properties	minOcc 0 maxOcc 1 content simple

annotation	documentation This element represents the minimum size of RAM (in MB) required for the execution (e.g. 512)
source	<xs:element name="minRAM" type="xs:int" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the minimum size of RAM (in MB) required for the execution (e.g. 512)</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaComponentModelType/Requirements/ResourceRequirements/Memory/maxRAM**

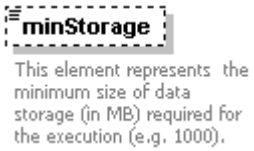
diagram	
type	**xs:int**
properties	minOcc 0 maxOcc 1 content simple
annotation	documentation This element represents the maximum size of RAM (in MB) required for the execution (e.g. 2048)
source	<xs:element name="maxRAM" type="xs:int" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the maximum size of RAM (in MB) required for the execution (e.g. 2048)</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaComponentModelType/Requirements/ResourceRequirements/Storage**

diagram	
properties	minOcc 0 maxOcc 1 content complex

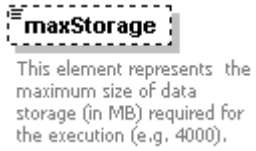
children	**minStorage maxStorage StorageType**
annotation	documentation This element represents the Storage Requirements.
source	<xs:element name="Storage" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the Storage Requirements.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="minStorage" type="xs:int" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the minimum size of data storage (in MB) required for the execution (e.g. 1000).</xs:documentation> </xs:annotation> </xs:element> <xs:element name="maxStorage" type="xs:int" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the maximum size of data storage (in MB) required for the execution (e.g. 4000).</xs:documentation> </xs:annotation> </xs:element> <xs:element name="StorageType" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the type of the storage element that is required.</xs:documentation> </xs:annotation> <xs:simpleType> <xs:restriction base="xs:string"> <xs:enumeration value="SSD"/> <xs:enumeration value="AmazonS3"/> <xs:enumeration value="HDD"/> </xs:restriction> </xs:simpleType> </xs:element> </xs:sequence> </xs:complexType> </xs:element>

element **ArcadiaComponentModelType/Requirements/ResourceRequirements/Storage/minStorage**

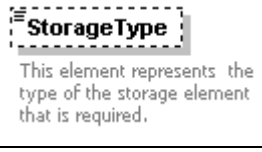
diagram	
type	**xs:int**
properties	minOcc 0 maxOcc 1 content simple
annotation	documentation This element represents the minimum size of data storage (in MB) required for the execution (e.g. 1000).

source	<xs:element name="minStorage" type="xs:int" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the minimum size of data storage (in MB) required for the execution (e.g. 1000).</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaComponentModelType/Requirements/ResourceRequirements/Storage/maxStorage**

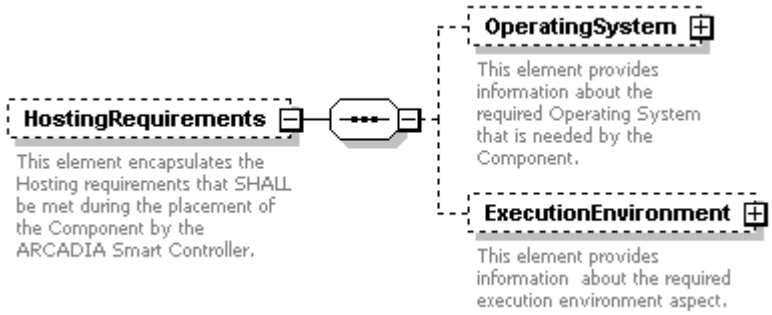
diagram	
type	**xs:int**
properties	minOcc 0 maxOcc 1 content simple
annotation	documentation This element represents the maximum size of data storage (in MB) required for the execution (e.g. 4000).
source	<xs:element name="maxStorage" type="xs:int" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the maximum size of data storage (in MB) required for the execution (e.g. 4000).</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaComponentModelType/Requirements/ResourceRequirements/Storage/StorageType**

diagram	
type	restriction of **xs:string**
properties	minOcc 0 maxOcc 1 content simple
facets	Kind Value Annotation enumeration SSD enumeration AmazonS3 enumeration HDD
annotation	documentation This element represents the type of the storage element that is required.
source	<xs:element name="StorageType" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the type of the storage element that is required.</xs:documentation>

```
  </xs:annotation>
  <xs:simpleType>
   <xs:restriction                                              base="xs:string">
    <xs:enumeration                                             value="SSD"/>
    <xs:enumeration                                             value="AmazonS3"/>
    <xs:enumeration                                             value="HDD"/>
   </xs:restriction>
  </xs:simpleType>
</xs:element>
```

element **ArcadiaComponentModelType/Requirements/HostingRequirements**

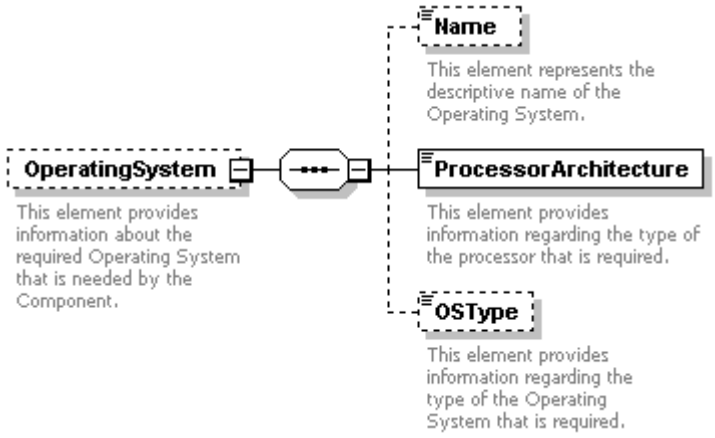
diagram	
properties	minOcc 0 maxOcc 1 content complex
children	**OperatingSystem ExecutionEnvironment**
annotation	documentation This element encapsulates the Hosting requirements that SHALL be met during the placement of the Component by the ARCADIA Smart Controller.

source

```
<xs:element                    name="HostingRequirements"                    minOccurs="0">
 <xs:annotation>
  <xs:documentation>This element encapsulates the Hosting requirements that SHALL be met during the placement of the Component by the ARCADIA Smart Controller.</xs:documentation>
 </xs:annotation>
 <xs:complexType>
  <xs:sequence>
   <xs:element                    name="OperatingSystem"                    minOccurs="0">
    <xs:annotation>
     <xs:documentation>This element provides information about the required Operating System that is needed by the Component.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
     <xs:sequence>
      <xs:element                    name="Name"                    minOccurs="0">
       <xs:annotation>
        <xs:documentation>This element represents the descriptive name of the Operating System.</xs:documentation>
       </xs:annotation>
       <xs:simpleType>
        <xs:restriction                                              base="xs:string">
         <xs:enumeration                                             value="Ubuntu14.04LTSamd64"/>
         <xs:enumeration                                             value="Ubuntu14.04LTSi386"/>
```

```
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
        <xs:element name="ProcessorArchitecture" type="ProcessorArchitectureType">
          <xs:annotation>
            <xs:documentation>This element provides information regarding the type of the processor that is required.</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="OSType" minOccurs="0">
          <xs:annotation>
            <xs:documentation>This element provides information regarding the type of the Operating System that is required.</xs:documentation>
          </xs:annotation>
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:enumeration value="Linux"/>
              <xs:enumeration value="Windows"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="ExecutionEnvironment" type="ExecutionEnvironmentType" minOccurs="0">
    <xs:annotation>
      <xs:documentation>This element provides information about the required execution environment aspect.</xs:documentation>
    </xs:annotation>
  </xs:element>
 </xs:sequence>
</xs:complexType>
</xs:element>
```

element **ArcadiaComponentModelType/Requirements/HostingRequirements/OperatingSystem**

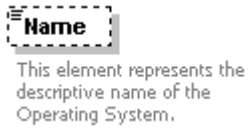
diagram	
properties	minOcc 0 maxOcc 1

	content complex
children	**Name** **ProcessorArchitecture** **OSType**
annotation	documentation This element provides information about the required Operating System that is needed by the Component.

source

```
<xs:element name="OperatingSystem" minOccurs="0">
 <xs:annotation>
  <xs:documentation>This element provides information about the required Operating System that is needed by the Component.</xs:documentation>
 </xs:annotation>
 <xs:complexType>
  <xs:sequence>
   <xs:element name="Name" minOccurs="0">
    <xs:annotation>
     <xs:documentation>This element represents the descriptive name of the Operating System.</xs:documentation>
    </xs:annotation>
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:enumeration value="Ubuntu14.04LTSamd64"/>
      <xs:enumeration value="Ubuntu14.04LTSi386"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:element name="ProcessorArchitecture" type="ProcessorArchitectureType">
    <xs:annotation>
     <xs:documentation>This element provides information regarding the type of the processor that is required.</xs:documentation>
    </xs:annotation>
   </xs:element>
   <xs:element name="OSType" minOccurs="0">
    <xs:annotation>
     <xs:documentation>This element provides information regarding the type of the Operating System that is required.</xs:documentation>
    </xs:annotation>
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:enumeration value="Linux"/>
      <xs:enumeration value="Windows"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

element **ArcadiaComponentModelType/Requirements/HostingRequirements/OperatingSystem/Name**

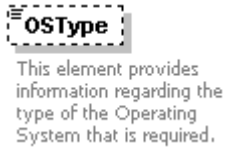
diagram	
type	restriction of **xs:string**

properties	minOcc 0 maxOcc 1 content simple
facets	Kind Value Annotation enumeration Ubuntu14.04LTSamd64 enumeration Ubuntu14.04LTSi386
annotation	documentation This element represents the descriptive name of the Operating System.
source	<xs:element name="Name" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the descriptive name of the Operating System.</xs:documentation> </xs:annotation> <xs:simpleType> <xs:restriction base="xs:string"> <xs:enumeration value="Ubuntu14.04LTSamd64"/> <xs:enumeration value="Ubuntu14.04LTSi386"/> </xs:restriction> </xs:simpleType> </xs:element>

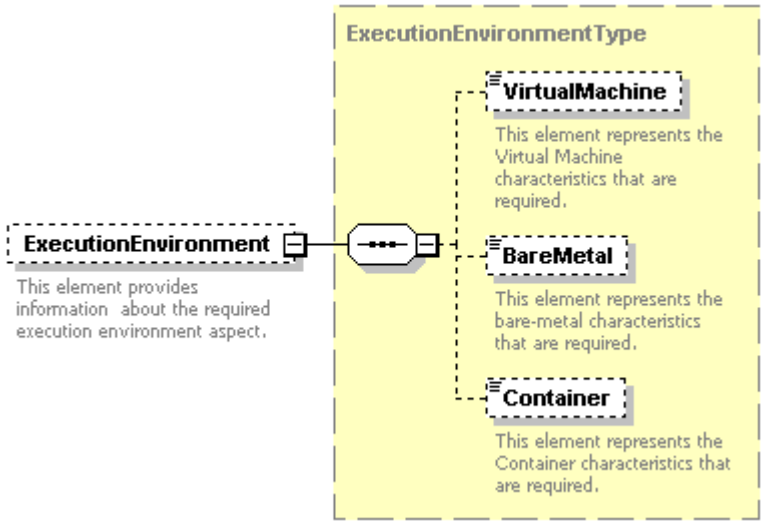
element
ArcadiaComponentModelType/Requirements/HostingRequirements/OperatingSystem/ProcessorArchitecture

diagram	
type	ProcessorArchitectureType
properties	content simple
facets	Kind Value Annotation enumeration amd64 enumeration i386
annotation	documentation This element provides information regarding the type of the processor that is required.
source	<xs:element name="ProcessorArchitecture" type="ProcessorArchitectureType"> <xs:annotation> <xs:documentation>This element provides information regarding the type of the processor that is required.</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaComponentModelType/Requirements/HostingRequirements/OperatingSystem/OSType**

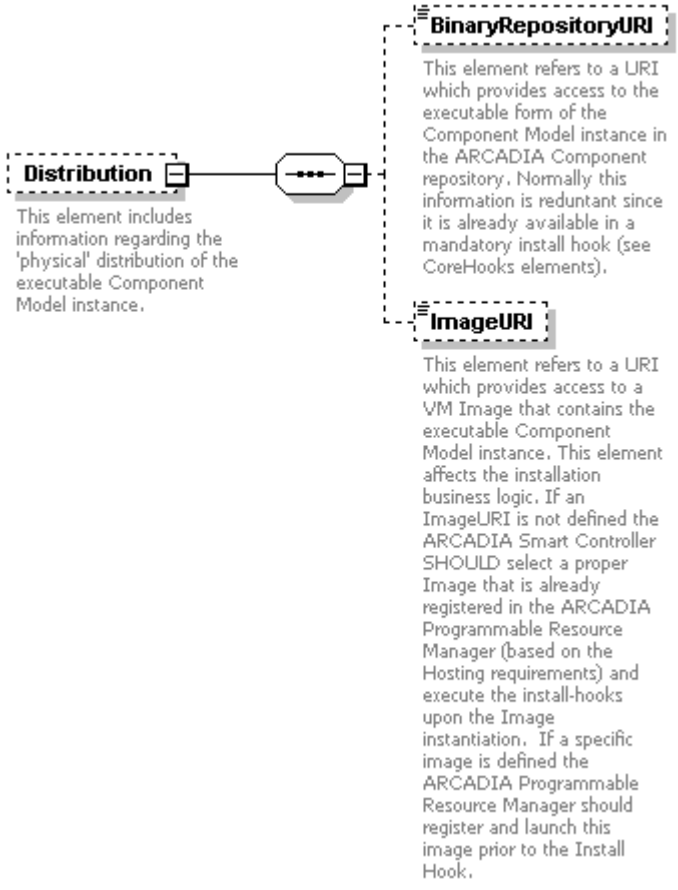
diagram	
type	restriction of **xs:string**
properties	minOcc 0 maxOcc 1 content simple
facets	Kind Value Annotation enumeration Linux enumeration Windows
annotation	documentation This element provides information regarding the type of the Operating System that is required.
source	<xs:element name="OSType" minOccurs="0"> <xs:annotation> <xs:documentation>This element provides information regarding the type of the Operating System that is required.</xs:documentation> </xs:annotation> <xs:simpleType> <xs:restriction base="xs:string"> <xs:enumeration value="Linux"/> <xs:enumeration value="Windows"/> </xs:restriction> </xs:simpleType> </xs:element>

element **ArcadiaComponentModelType/Requirements/HostingRequirements/ExecutionEnvironment**

diagram	
type	**ExecutionEnvironmentType**

properties	minOcc 0 maxOcc 1 content complex
children	**VirtualMachine BareMetal Container**
annotation	documentation This element provides information about the required execution environment aspect.
source	<xs:element name="ExecutionEnvironment" type="ExecutionEnvironmentType" minOccurs="0"> <xs:annotation> <xs:documentation>This element provides information about the required execution environment aspect.</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaComponentModelType/Distribution**

diagram	
properties	minOcc 0 maxOcc 1 content complex
children	**BinaryRepositoryURI ImageURI**
annotation	documentation This element includes information regarding the 'physical' distribution of the executable Component Model instance.
source	<xs:element name="Distribution" minOccurs="0">

```
<xs:annotation>
 <xs:documentation>This element includes information regarding the 'physical' distribution of the executable Component Model instance.</xs:documentation>
</xs:annotation>
<xs:complexType>
 <xs:sequence>
  <xs:element name="BinaryRepositoryURI" type="xs:string" minOccurs="0">
   <xs:annotation>
    <xs:documentation>This element refers to a URI which provides access to the executable form of the Component Model instance in the ARCADIA Component repository. Normally this information is reduntant since it is already available in a mandatory install hook (see CoreHooks elements).</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="ImageURI" type="xs:string" minOccurs="0">
   <xs:annotation>
    <xs:documentation>This element refers to a URI which provides access to a VM Image that contains the executable Component Model instance. This element affects the installation business logic. If an ImageURI is not defined the ARCADIA Smart Controller SHOULD select a proper Image that is already registered in the ARCADIA Programmable Resource Manager (based on the Hosting requirements) and execute the install-hooks upon the Image instantiation. If a specific image is defined the ARCADIA Programmable Resource Manager should register and launch this image prior to the Install Hook.</xs:documentation>
   </xs:annotation>
  </xs:element>
 </xs:sequence>
</xs:complexType>
</xs:element>
```

element **ArcadiaComponentModelType/Distribution/BinaryRepositoryURI**

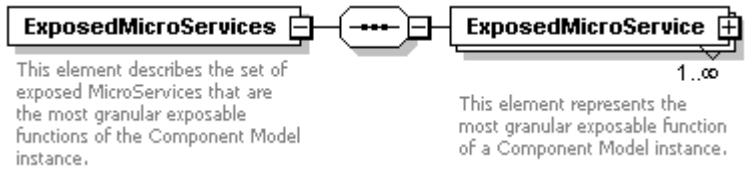
diagram	
type	**xs:string**
properties	minOcc 0 maxOcc 1 content simple
annotation	documentation This element refers to a URI which provides access to the executable form of the Component Model instance in the ARCADIA Component repository. Normally this information is reduntant since it is already available in a mandatory install hook (see CoreHooks elements).
source	<xs:element name="BinaryRepositoryURI" type="xs:string" minOccurs="0"> <xs:annotation> <xs:documentation>This element refers to a URI which provides access to the executable form of the Component Model instance in the ARCADIA Component repository. Normally this information is reduntant since it is already available in a mandatory install hook (see CoreHooks

	elements).</xs:documentation> </xs:annotation> </xs:element>

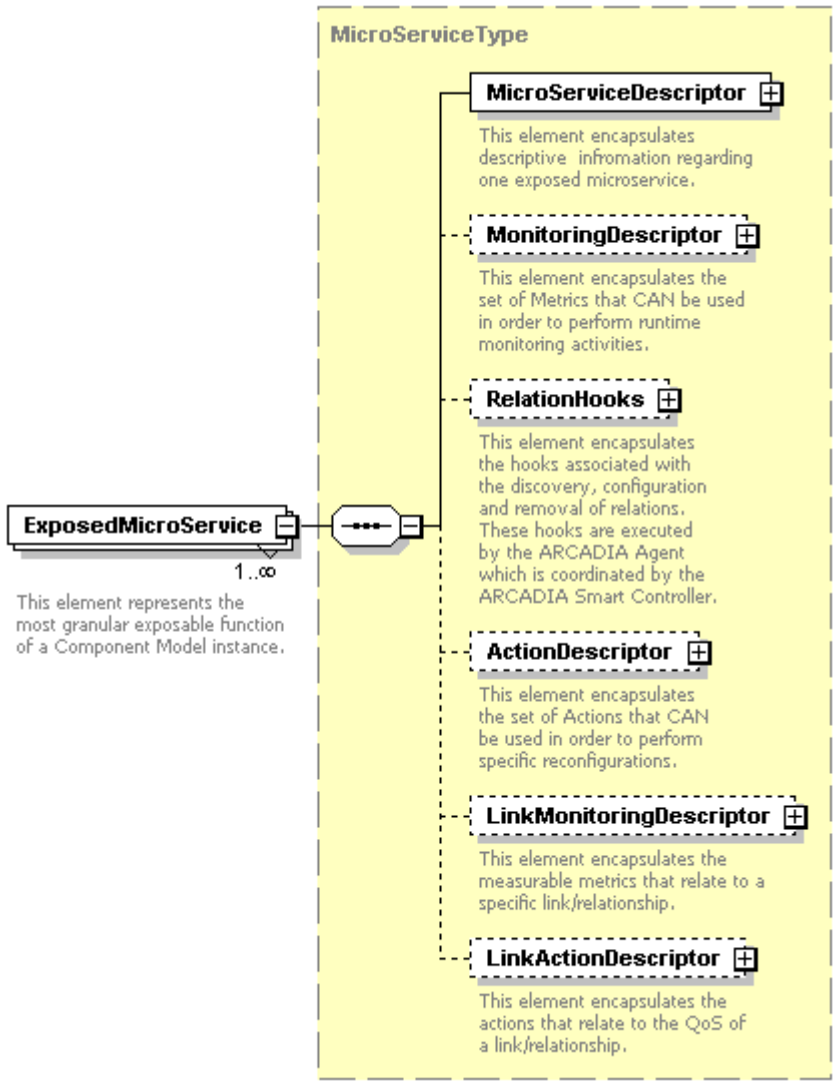
element **ArcadiaComponentModelType/Distribution/ImageURI**

diagram	
type	**xs:string**
properties	minOcc 0 maxOcc 1 content simple
annotation	documentation This element refers to a URI which provides access to a VM Image that contains the executable Component Model instance. This element affects the installation business logic. If an ImageURI is not defined the ARCADIA Smart Controller SHOULD select a proper Image that is already registered in the ARCADIA Programmable Resource Manager (based on the Hosting requirements) and execute the install-hooks upon the Image instantiation. If a specific image is defined the ARCADIA Programmable Resource Manager should register and launch this image prior to the Install Hook.
source	<xs:element name="ImageURI" type="xs:string" minOccurs="0"> <xs:annotation> <xs:documentation>This element refers to a URI which provides access to a VM Image that contains the executable Component Model instance. This element affects the installation business logic. If an ImageURI is not defined the ARCADIA Smart Controller SHOULD select a proper Image that is already registered in the ARCADIA Programmable Resource Manager (based on the Hosting requirements) and execute the install-hooks upon the Image instantiation. If a specific image is defined the ARCADIA Programmable Resource Manager should register and launch this image prior to the Install Hook.</xs:documentation> </xs:annotation> </xs:element>

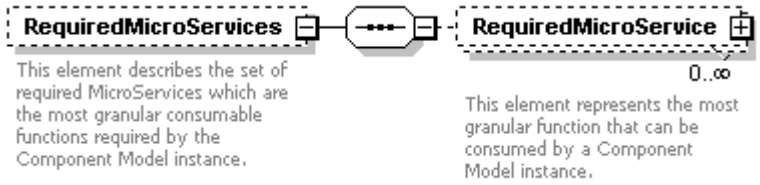
element **ArcadiaComponentModelType/ExposedMicroServices**

diagram	
properties	content complex
children	**ExposedMicroService**
annotation	documentation This element describes the set of exposed MicroServices that are the most granular exposable functions of the Component Model instance.
source	<xs:element name="ExposedMicroServices"> <xs:annotation> <xs:documentation>This element describes the set of exposed MicroServices that are the most granular exposable functions of the Component Model instance.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="ExposedMicroService" type="MicroServiceType" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents the most granular exposable function of a Component Model instance.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType> </xs:element>

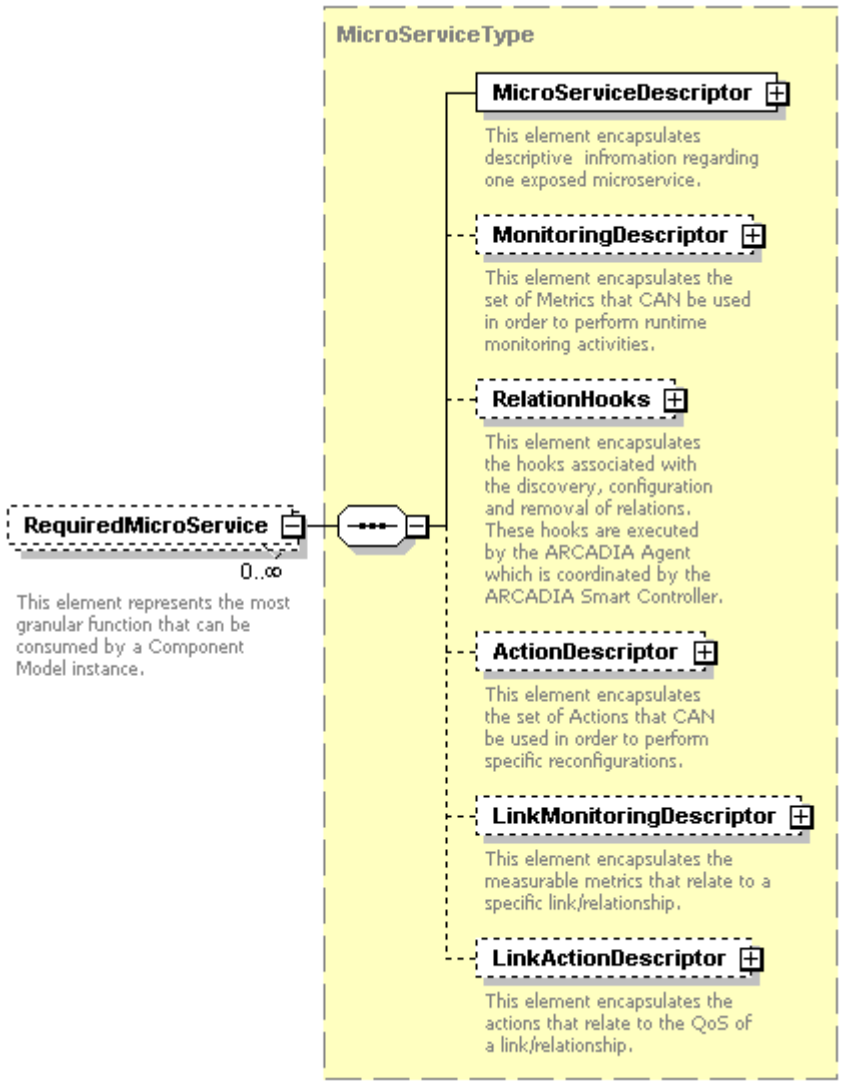
element **ArcadiaComponentModelType/ExposedMicroServices/ExposedMicroService**

diagram	
type	MicroServiceType
properties	minOcc 1 maxOcc unbounded content complex
children	MicroServiceDescriptor MonitoringDescriptor RelationHooks ActionDescriptor LinkMonitoringDescriptor LinkActionDescriptor
annotation	documentation This element represents the most granular exposable function of a Component Model instance.
source	<xs:element name="ExposedMicroService" type="MicroServiceType" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents the most granular exposable function of a Component Model instance.</xs:documentation> </xs:annotation> </xs:element>

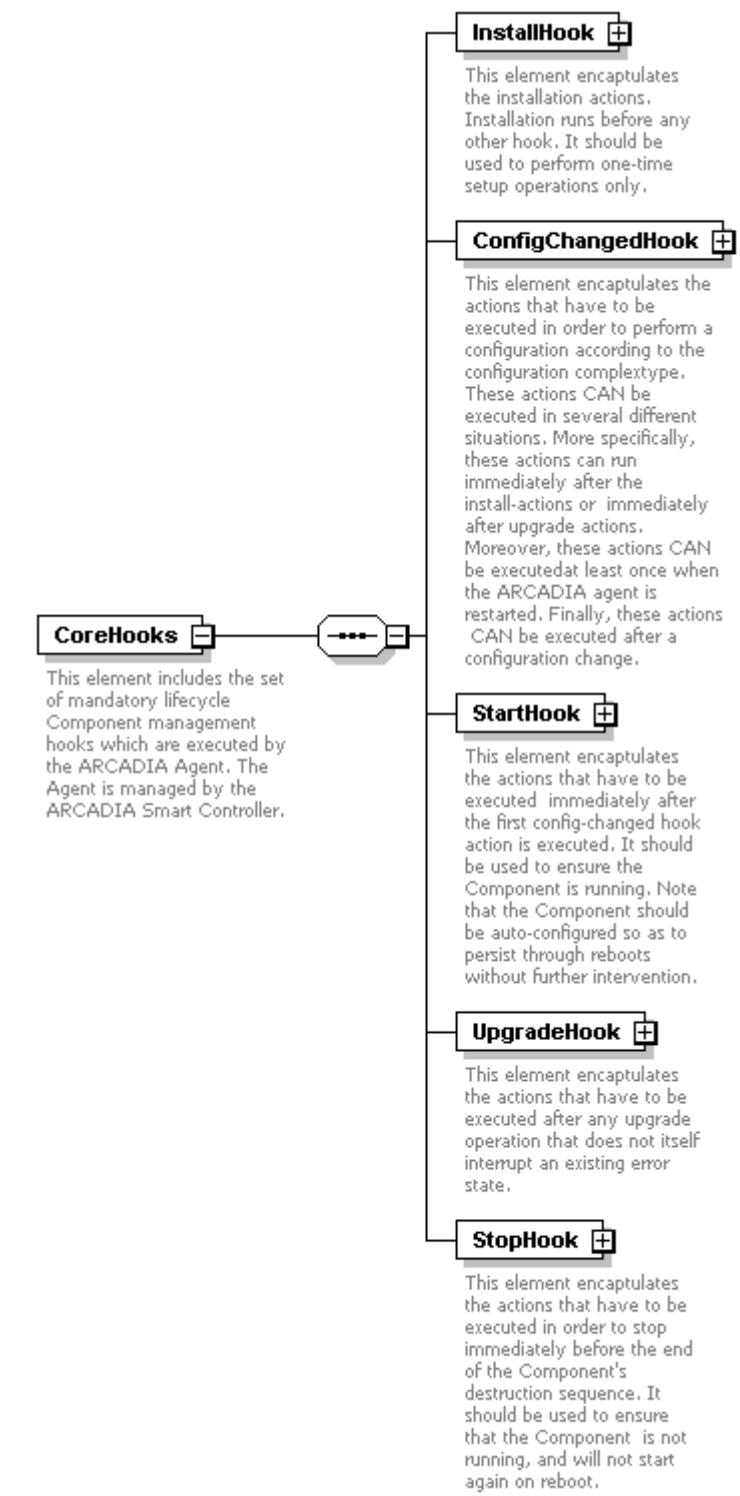
element **ArcadiaComponentModelType/RequiredMicroServices**

diagram	
properties	minOcc 0 maxOcc 1 content complex
children	**RequiredMicroService**
annotation	documentation This element describes the set of required MicroServices which are the most granular consumable functions required by the Component Model instance.
source	<xs:element name="RequiredMicroServices" minOccurs="0"> <xs:annotation> <xs:documentation>This element describes the set of required MicroServices which are the most granular consumable functions required by the Component Model instance.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="RequiredMicroService" type="MicroServiceType" minOccurs="0" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents the most granular function that can be consumed by a Component Model instance.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType> </xs:element>

element **ArcadiaComponentModelType/RequiredMicroServices/RequiredMicroService**

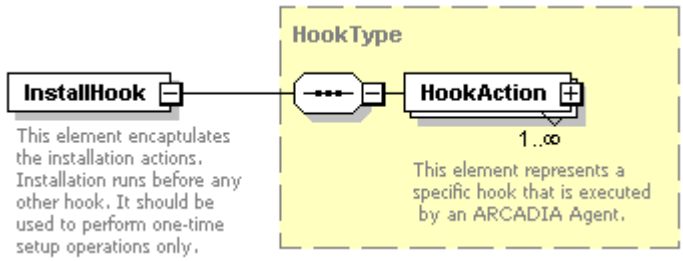
diagram	
type	MicroServiceType
properties	minOcc 0 maxOcc unbounded content complex
children	MicroServiceDescriptor MonitoringDescriptor RelationHooks ActionDescriptor LinkMonitoringDescriptor LinkActionDescriptor
annotation	documentation This element represents the most granular function that can be consumed by a Component Model instance.
source	<xs:element name="RequiredMicroService" type="MicroServiceType" minOccurs="0" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents the most granular function that can be consumed by a Component Model instance.</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaComponentModelType/CoreHooks**

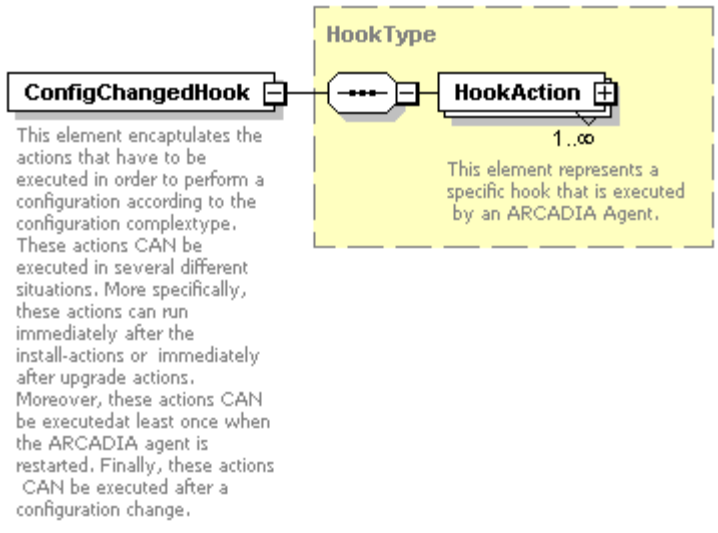
diagram	
properties	content complex
children	InstallHook ConfigChangedHook StartHook UpgradeHook StopHook
annotation	documentation

	This element includes the set of mandatory lifecycle Component management hooks which are executed by the ARCADIA Agent. The Agent is managed by the ARCADIA Smart Controller.
source	<xs:element name="CoreHooks"> <xs:annotation> <xs:documentation>This element includes the set of mandatory lifecycle Component management hooks which are executed by the ARCADIA Agent. The Agent is managed by the ARCADIA Smart Controller.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="InstallHook" type="HookType"> <xs:annotation> <xs:documentation>This element encaptulates the installation actions. Installation runs before any other hook. It should be used to perform one-time setup operations only.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="ConfigChangedHook" type="HookType"> <xs:annotation> <xs:documentation>This element encaptulates the actions that have to be executed in order to perform a configuration according to the configuration complextype. These actions CAN be executed in several different situations. More specifically, these actions can run immediately after the install-actions or immediately after upgrade actions. Moreover, these actions CAN be executedat least once when the ARCADIA agent is restarted. Finally, these actions CAN be executed after a configuration change. </xs:documentation> </xs:annotation> </xs:element> <xs:element name="StartHook" type="HookType"> <xs:annotation> <xs:documentation>This element encaptulates the actions that have to be executed immediately after the first config-changed hook action is executed. It should be used to ensure the Component is running. Note that the Component should be auto-configured so as to persist through reboots without further intervention.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="UpgradeHook" type="HookType"> <xs:annotation> <xs:documentation>This element encaptulates the actions that have to be executed after any upgrade operation that does not itself interrupt an existing error state.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="StopHook" type="HookType"> <xs:annotation> <xs:documentation>This element encaptulates the actions that have to be executed in order to stop immediately before the end of the Component's destruction sequence. It should be used to ensure that the Component is not running, and will not start again on reboot.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType> </xs:element>

element **ArcadiaComponentModelType/CoreHooks/InstallHook**

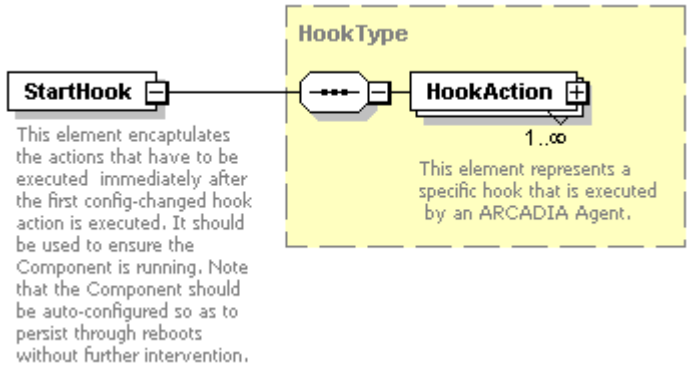
diagram	
type	**HookType**
properties	content complex
children	**HookAction**
annotation	documentation This element encaptulates the installation actions. Installation runs before any other hook. It should be used to perform one-time setup operations only.
source	<xs:element name="InstallHook" type="HookType"> <xs:annotation> <xs:documentation>This element encaptulates the installation actions. Installation runs before any other hook. It should be used to perform one-time setup operations only.</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaComponentModelType/CoreHooks/ConfigChangedHook**

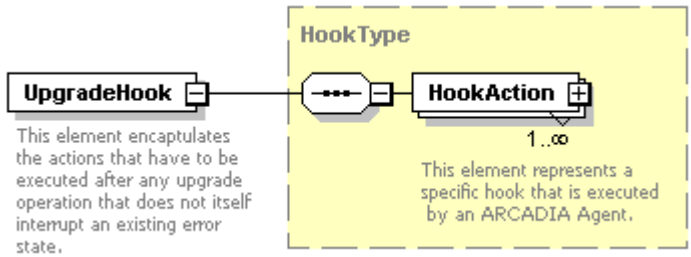
diagram	
type	**HookType**
properties	content complex
children	**HookAction**
annotation	documentation This element encaptulates the actions that have to be executed in order to perform a configuration according to the configuration complextype. These actions CAN be executed in several different situations. More specifically, these actions can run immediately after the install-actions or immediately after upgrade actions. Moreover, these actions CAN be executedat least once when the ARCADIA agent is restarted. Finally, these actions CAN be executed after a

	configuration change.
source	<xs:element name="ConfigChangedHook" type="HookType"> <xs:annotation> <xs:documentation>This element encaptulates the actions that have to be executed in order to perform a configuration according to the configuration complextype. These actions CAN be executed in several different situations. More specifically, these actions can run immediately after the install-actions or immediately after upgrade actions. Moreover, these actions CAN be executedat least once when the ARCADIA agent is restarted. Finally, these actions CAN be executed after a configuration change. </xs:documentation> </xs:annotation> </xs:element>

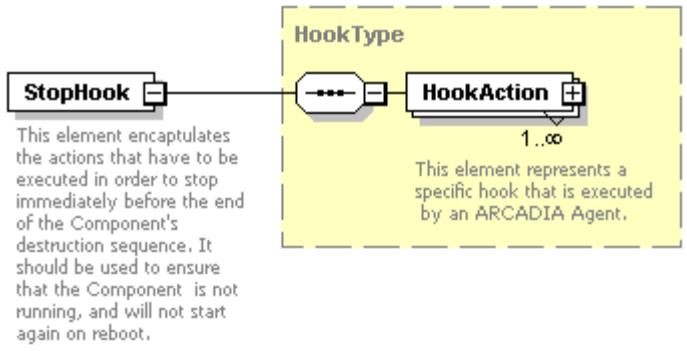
element **ArcadiaComponentModelType/CoreHooks/StartHook**

diagram	
type	**HookType**
properties	content complex
children	**HookAction**
annotation	documentation This element encaptulates the actions that have to be executed immediately after the first config-changed hook action is executed. It should be used to ensure the Component is running. Note that the Component should be auto-configured so as to persist through reboots without further intervention.
source	<xs:element name="StartHook" type="HookType"> <xs:annotation> <xs:documentation>This element encaptulates the actions that have to be executed immediately after the first config-changed hook action is executed. It should be used to ensure the Component is running. Note that the Component should be auto-configured so as to persist through reboots without further intervention.</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaComponentModelType/CoreHooks/UpgradeHook**

diagram	
type	**HookType**
properties	content complex
children	**HookAction**
annotation	documentation This element encaptulates the actions that have to be executed after any upgrade operation that does not itself interrupt an existing error state.
source	<xs:element name="UpgradeHook" type="HookType"> <xs:annotation> <xs:documentation>This element encaptulates the actions that have to be executed after any upgrade operation that does not itself interrupt an existing error state.</xs:documentation> </xs:annotation> </xs:element>

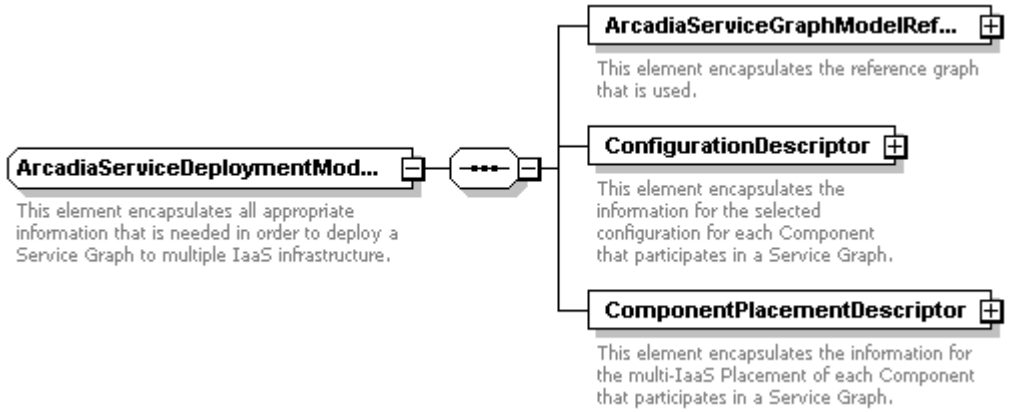
element **ArcadiaComponentModelType/CoreHooks/StopHook**

diagram	
type	**HookType**
properties	content complex
children	**HookAction**
annotation	documentation This element encaptulates the actions that have to be executed in order to stop immediately before the end of the Component's destruction sequence. It should be used to ensure that the Component is not running, and will not start again on reboot.
source	<xs:element name="StopHook" type="HookType"> <xs:annotation> <xs:documentation>This element encaptulates the actions that have to be executed in order to stop immediately before the end of the Component's destruction sequence. It should be used to ensure that the Component is not running, and will not start again on reboot.</xs:documentation>

```
  </xs:annotation>
</xs:element>
```

complexType **ArcadiaServiceDeploymentModelType**

diagram



children: ArcadiaServiceGraphModelReference ConfigurationDescriptor ComponentPlacementDescriptor

used by: elements ArcadiaServiceDeploymentModel ArcadiaServiceRuntimeModelType/DeploymentModel

annotation: documentation

This element encapsulates all appropriate information that is needed in order to deploy a Service Graph to multiple IaaS infrastructure.

source

```
<xs:complexType name="ArcadiaServiceDeploymentModelType">
 <xs:annotation>
  <xs:documentation>This element encapsulates all appropriate information that is needed in order to deploy a Service Graph to multiple IaaS infrastructure.</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element name="ArcadiaServiceGraphModelReference" type="ArcadiaServiceGraphModelType">
   <xs:annotation>
    <xs:documentation>This element encapsulates the reference graph that is used.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="ConfigurationDescriptor">
   <xs:annotation>
    <xs:documentation>This element encapsulates the information for the selected configuration for each Component that participates in a Service Graph.</xs:documentation>
   </xs:annotation>
   <xs:complexType>
    <xs:sequence>
     <xs:element name="ConfigurationAction" maxOccurs="unbounded">
      <xs:annotation>
       <xs:documentation>This element encapsulates all the configuration actions that can be undertaken by the Smart Controlle for configuring a specific component.</xs:documentation>
      </xs:annotation>
      <xs:complexType>
       <xs:sequence>
        <xs:element ref="GraphNodeIdentifier">
         <xs:annotation>
```

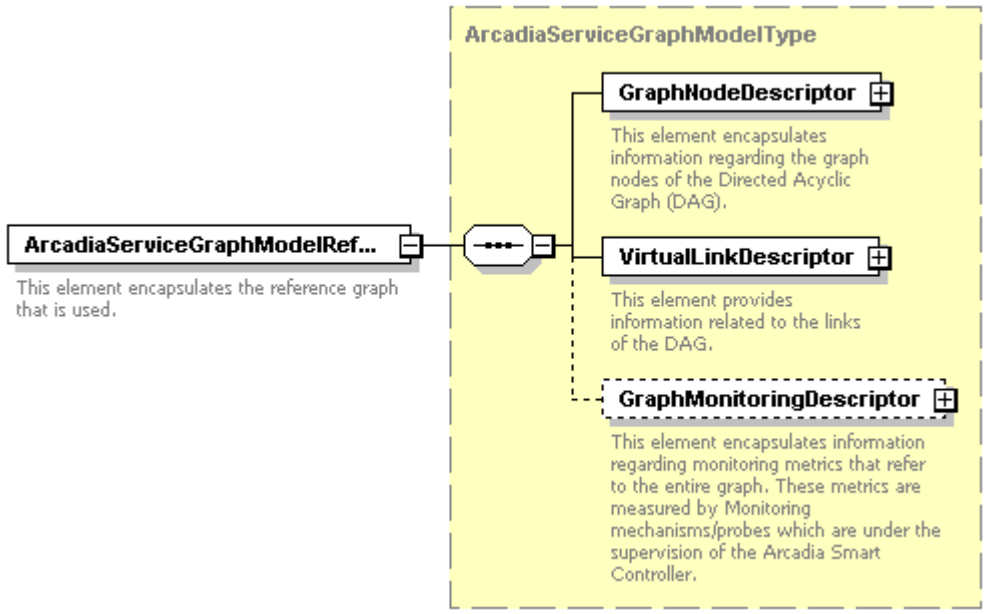
```
                <xs:documentation>This element represents an identifier that differentiates Graph Nodes in the frame of one Graph. Please bear in mind that one Component MAY participate as multiple Graph Nodes in the same graph.</xs:documentation>
              </xs:annotation>
            </xs:element>
            <xs:element ref="ConfigurationElementIdentifier"/>
            <xs:element name="ConfigurationValue" type="xs:string">
              <xs:annotation>
                <xs:documentation>This element represents the actual configuration value that is enforced by the ARCADIA Smart Controller.</xs:documentation>
              </xs:annotation>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="ComponentPlacementDescriptor">
  <xs:annotation>
   <xs:documentation>This element encapsulates the information for the multi-IaaS Placement of each Component that participates in a Service Graph.</xs:documentation>
  </xs:annotation>
  <xs:complexType>
   <xs:sequence>
    <xs:element name="ComponentPlacementAction" maxOccurs="unbounded">
     <xs:annotation>
      <xs:documentation>This element encapsulates all deployable actions that have to be performed by the ARCADIA Smart Controller through the ARCADIA Agents. </xs:documentation>
     </xs:annotation>
     <xs:complexType>
      <xs:sequence>
       <xs:element name="DeployableComponent">
        <xs:annotation>
         <xs:documentation>This element represents one Graph Node Component of the DG that represents a Service Graph.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
         <xs:sequence>
          <xs:element ref="GraphNodeIdentifier"/>
         </xs:sequence>
        </xs:complexType>
       </xs:element>
       <xs:element name="ServiceProviderDescriptor">
        <xs:annotation>
         <xs:documentation>This element encapsulates information regarding the association of one Component with the selected IaaS resources.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
         <xs:sequence>
          <xs:element name="IaaSConnectivity">
           <xs:annotation>
            <xs:documentation>This element provides information about a specific IaaS.</xs:documentation>
           </xs:annotation>
           <xs:complexType>
```

```
                    <xs:sequence>
                      <xs:element name="IaaSType">
                        <xs:annotation>
                          <xs:documentation>This element provides information about the type of the IaaS that is selected in the frame of one placement.</xs:documentation>
                        </xs:annotation>
                        <xs:simpleType>
                          <xs:restriction base="xs:string">
                            <xs:pattern value="OpenStack"/>
                          </xs:restriction>
                        </xs:simpleType>
                      </xs:element>
                      <xs:element name="APIEndpoint" type="xs:string">
                        <xs:annotation>
                          <xs:documentation>This element provides the public accessible API of the IaaS provider.</xs:documentation>
                        </xs:annotation>
                      </xs:element>
                      <xs:element name="APIVersion">
                        <xs:annotation>
                          <xs:documentation>This element provides information regarding the version of the selected API in order to assure compatibility.</xs:documentation>
                        </xs:annotation>
                      </xs:element>
                      <xs:element name="Credentials" type="xs:string">
                        <xs:annotation>
                          <xs:documentation>This elements encapsulates security information required in order to perform authentication and authorization to the IaaS.</xs:documentation>
                        </xs:annotation>
                      </xs:element>
                      <xs:element name="TenantIdentifier" type="xs:string">
                        <xs:annotation>
                          <xs:documentation>This element provides information regarding the specific tenant that is associated with the deployment.</xs:documentation>
                        </xs:annotation>
                      </xs:element>
                    </xs:sequence>
                  </xs:complexType>
                </xs:element>
                <xs:element name="DeploymentConstraints">
                  <xs:annotation>
                    <xs:documentation>This element encaptulates information related to the technical constraints that are imposed by the ARCADIA Smart Controller in order to perform proper instantiation of the Execution Environment.</xs:documentation>
                  </xs:annotation>
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element name="ExecutionEnvironment" type="ExecutionEnvironmentType">
                        <xs:annotation>
                          <xs:documentation>This element provides information related to the type of the execution environment that is required.</xs:documentation>
                        </xs:annotation>
                      </xs:element>
                      <xs:element name="CoreRequirements">
                        <xs:annotation>
```

```
                <xs:documentation>This element encapsulates information related to actual execution environment that SHALL be instantiated in an IaaS.</xs:documentation>
              </xs:annotation>
              <xs:complexType>
               <xs:sequence>
                <xs:element name="InstanceName" type="xs:string">
                 <xs:annotation>
                  <xs:documentation>InstanceName will be autogenerated based on the ComponentIdentifier</xs:documentation>
                 </xs:annotation>
                </xs:element>
                <xs:element name="ImageIdentifier" type="xs:string">
                 <xs:annotation>
                  <xs:documentation>The Image that will be used will either be defined by the component or selected based on the available registred images in the IaaS.</xs:documentation>
                 </xs:annotation>
                </xs:element>
                <xs:element name="Flavor">
                 <xs:annotation>
                  <xs:documentation>This element provides information related to the desired flavor that should be used in the chosen IaaS.</xs:documentation>
                 </xs:annotation>
                 <xs:complexType>
                  <xs:sequence>
                   <xs:element name="FlavorIdentifier"/>
                   <xs:element name="FlavorName" type="xs:string"/>
                   <xs:element name="vCPUs" type="xs:int">
                    <xs:annotation>
                     <xs:documentation>Number of CPUs</xs:documentation>
                    </xs:annotation>
                   </xs:element>
                   <xs:element name="RootDisk" type="xs:int"/>
                   <xs:element name="EphemeralDisk" type="xs:int"/>
                   <xs:element name="RAM" type="xs:int"/>
                  </xs:sequence>
                 </xs:complexType>
                </xs:element>
               </xs:sequence>
              </xs:complexType>
             </xs:element>
             <xs:element name="SecurityRequirements">
              <xs:annotation>
               <xs:documentation>This element provides security constraints that have to be met upon deployment. It will be elaborated in version 2 of the model.</xs:documentation>
              </xs:annotation>
              <xs:complexType>
               <xs:sequence>
                <xs:element name="SecurityGroupIdentifier"/>
               </xs:sequence>
              </xs:complexType>
             </xs:element>
             <xs:element name="NetworkingRequirements">
              <xs:annotation>
               <xs:documentation>This element provides network constraints that have to be met upon deployment. It will be refined in version 2 of the model.</xs:documentation>
```

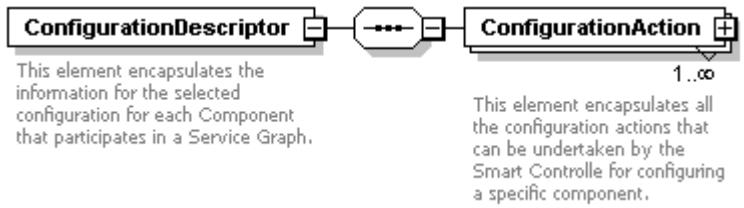
```
                    </xs:annotation>
                    <xs:complexType>
                      <xs:sequence>
                        <xs:element                                    name="SubNetIdentifier"/>
                      </xs:sequence>
                    </xs:complexType>
                  </xs:element>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
```

element **ArcadiaServiceDeploymentModelType/ArcadiaServiceGraphModelReference**

diagram	
type	ArcadiaServiceGraphModelType
properties	content complex
children	GraphNodeDescriptor VirtualLinkDescriptor GraphMonitoringDescriptor
annotation	documentation This element encapsulates the reference graph that is used.
source	<xs:element name="ArcadiaServiceGraphModelReference" type="ArcadiaServiceGraphModelType"> <xs:annotation>

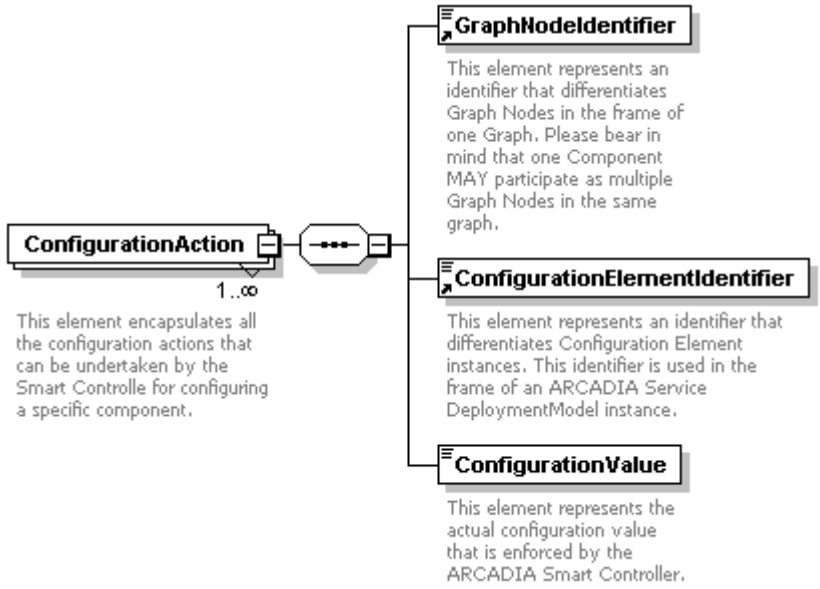
	<xs:documentation>This element encapsulates the reference graph that is used.</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaServiceDeploymentModelType/ConfigurationDescriptor**

diagram	
properties	content complex
children	**ConfigurationAction**
annotation	documentation This element encapsulates the information for the selected configuration for each Component that participates in a Service Graph.
source	<xs:element name="ConfigurationDescriptor"> <xs:annotation> <xs:documentation>This element encapsulates the information for the selected configuration for each Component that participates in a Service Graph.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="ConfigurationAction" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element encapsulates all the configuration actions that can be undertaken by the Smart Controlle for configuring a specific component.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element ref="GraphNodeIdentifier"> <xs:annotation> <xs:documentation>This element represents an identifier that differentiates Graph Nodes in the frame of one Graph. Please bear in mind that one Component MAY participate as multiple Graph Nodes in the same graph.</xs:documentation> </xs:annotation> </xs:element> <xs:element ref="ConfigurationElementIdentifier"/> <xs:element name="ConfigurationValue" type="xs:string"> <xs:annotation> <xs:documentation>This element represents the actual configuration value that is enforced by the ARCADIA Smart Controller.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType> </xs:element> </xs:sequence> </xs:complexType>

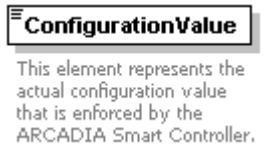
	</xs:element>

element **ArcadiaServiceDeploymentModelType/ConfigurationDescriptor/ConfigurationAction**

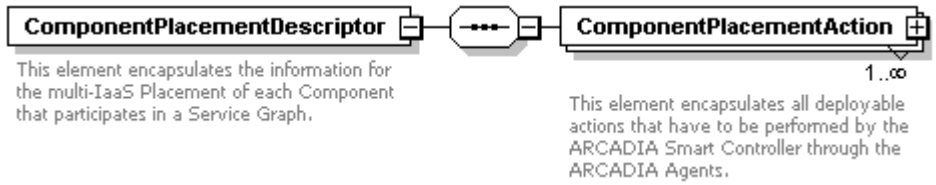
diagram	
properties	minOcc 1 maxOcc unbounded content complex
children	GraphNodeIdentifier ConfigurationElementIdentifier ConfigurationValue
annotation	documentation This element encapsulates all the configuration actions that can be undertaken by the Smart Controlle for configuring a specific component.
source	<xs:element name="ConfigurationAction" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element encapsulates all the configuration actions that can be undertaken by the Smart Controlle for configuring a specific component.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element ref="GraphNodeIdentifier"> <xs:annotation> <xs:documentation>This element represents an identifier that differentiates Graph Nodes in the frame of one Graph. Please bear in mind that one Component MAY participate as multiple Graph Nodes in the same graph.</xs:documentation> </xs:annotation> </xs:element> <xs:element ref="ConfigurationElementIdentifier"/> <xs:element name="ConfigurationValue" type="xs:string"> <xs:annotation> <xs:documentation>This element represents the actual configuration value that is enforced by the ARCADIA Smart Controller.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence>

```
  </xs:complexType>
</xs:element>
```

element **ArcadiaServiceDeploymentModelType/ConfigurationDescriptor/ConfigurationAction/ConfigurationValue**

diagram	
type	**xs:string**
properties	content simple
annotation	documentation This element represents the actual configuration value that is enforced by the ARCADIA Smart Controller.
source	<xs:element name="ConfigurationValue" type="xs:string"> <xs:annotation> <xs:documentation>This element represents the actual configuration value that is enforced by the ARCADIA Smart Controller.</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor**

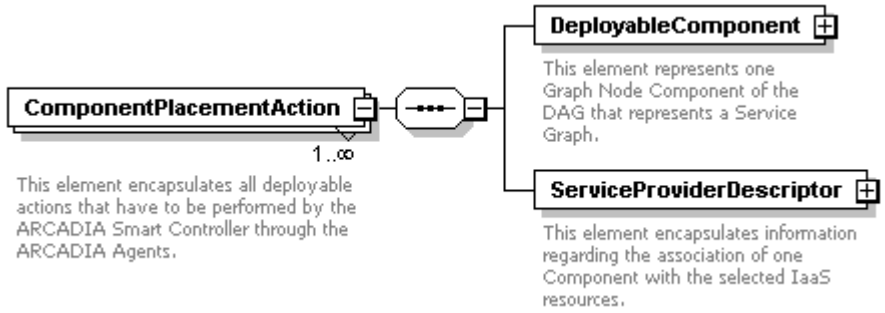
diagram	
properties	content complex
children	**ComponentPlacementAction**
annotation	documentation This element encapsulates the information for the multi-IaaS Placement of each Component that participates in a Service Graph.
source	<xs:element name="ComponentPlacementDescriptor"> <xs:annotation> <xs:documentation>This element encapsulates the information for the multi-IaaS Placement of each Component that participates in a Service Graph.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="ComponentPlacementAction" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element encapsulates all deployable actions that have to be performed by the ARCADIA Smart Controller through the ARCADIA Agents. </xs:documentation> </xs:annotation>

```
      <xs:complexType>
       <xs:sequence>
        <xs:element                                    name="DeployableComponent">
         <xs:annotation>
          <xs:documentation>This element represents one Graph Node Component of the DG
that          represents          a          Service          Graph.</xs:documentation>
         </xs:annotation>
         <xs:complexType>
          <xs:sequence>
           <xs:element                                    ref="GraphNodeIdentifier"/>
          </xs:sequence>
         </xs:complexType>
        </xs:element>
        <xs:element                                    name="ServiceProviderDescriptor">
         <xs:annotation>
          <xs:documentation>This element encapsulates information regarding the association of
one      Component      with      the      selected      IaaS      resources.</xs:documentation>
         </xs:annotation>
         <xs:complexType>
          <xs:sequence>
           <xs:element                                    name="IaaSConnectivity">
            <xs:annotation>
             <xs:documentation>This element provides information about a specific
IaaS.</xs:documentation>
            </xs:annotation>
            <xs:complexType>
             <xs:sequence>
              <xs:element                                    name="IaaSType">
               <xs:annotation>
                <xs:documentation>This element provides information about the type of the
IaaS that is selected in the frame of one placement.</xs:documentation>
               </xs:annotation>
               <xs:simpleType>
                <xs:restriction                                    base="xs:string">
                 <xs:pattern                                    value="OpenStack"/>
                </xs:restriction>
               </xs:simpleType>
              </xs:element>
              <xs:element                    name="APIEndpoint"                    type="xs:string">
               <xs:annotation>
                <xs:documentation>This element provides the public accessible API of the IaaS
provider.</xs:documentation>
               </xs:annotation>
              </xs:element>
              <xs:element                                    name="APIVersion">
               <xs:annotation>
                <xs:documentation>This element provides information regarding the version of
the selected API in order to assure compatibility.</xs:documentation>
               </xs:annotation>
              </xs:element>
              <xs:element                    name="Credentials"                    type="xs:string">
               <xs:annotation>
                <xs:documentation>This elements encapsulates security information required in
order to perform authentication and authorization to the IaaS.</xs:documentation>
               </xs:annotation>
              </xs:element>
```

```
                <xs:element name="TenantIdentifier" type="xs:string">
                  <xs:annotation>
                    <xs:documentation>This element provides information regarding the specific tenant that is associated with the deployment.</xs:documentation>
                  </xs:annotation>
                </xs:element>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
          <xs:element name="DeploymentConstraints">
            <xs:annotation>
              <xs:documentation>This element encaptulates information related to the technical constraints that are imposed by the ARCADIA Smart Controller in order to perform proper instantiation of the Execution Environment.</xs:documentation>
            </xs:annotation>
            <xs:complexType>
              <xs:sequence>
                <xs:element name="ExecutionEnvironment" type="ExecutionEnvironmentType">
                  <xs:annotation>
                    <xs:documentation>This element provides information related to the type of the execution environment that is required.</xs:documentation>
                  </xs:annotation>
                </xs:element>
                <xs:element name="CoreRequirements">
                  <xs:annotation>
                    <xs:documentation>This element encapsulates information related to actual execution environment that SHALL be instantiated in an IaaS.</xs:documentation>
                  </xs:annotation>
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element name="InstanceName" type="xs:string">
                        <xs:annotation>
                          <xs:documentation>InstanceName will be autogenerated based on the ComponentIdentifier</xs:documentation>
                        </xs:annotation>
                      </xs:element>
                      <xs:element name="ImageIdentifier" type="xs:string">
                        <xs:annotation>
                          <xs:documentation>The Image that will be used will either be defined by the component or selected based on the available registred images in the IaaS.</xs:documentation>
                        </xs:annotation>
                      </xs:element>
                      <xs:element name="Flavor">
                        <xs:annotation>
                          <xs:documentation>This element provides information related to the desired flavor that should be used in the chosen IaaS.</xs:documentation>
                        </xs:annotation>
                        <xs:complexType>
                          <xs:sequence>
                            <xs:element name="FlavorIdentifier"/>
                            <xs:element name="FlavorName" type="xs:string"/>
                            <xs:element name="vCPUs" type="xs:int">
                              <xs:annotation>
                                <xs:documentation>Number of CPUs</xs:documentation>
                              </xs:annotation>
                            </xs:element>
```

```
                    <xs:element name="RootDisk" type="xs:int"/>
                    <xs:element name="EphemeralDisk" type="xs:int"/>
                    <xs:element name="RAM" type="xs:int"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="SecurityRequirements">
          <xs:annotation>
            <xs:documentation>This element provides security constraints that have to be met upon deployment. It will be elaborated in version 2 of the model.</xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:sequence>
              <xs:element name="SecurityGroupIdentifier"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="NetworkingRequirements">
          <xs:annotation>
            <xs:documentation>This element provides network constraints that have to be met upon deployment. It will be refined in version 2 of the model.</xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:sequence>
              <xs:element name="SubNetIdentifier"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
```

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction

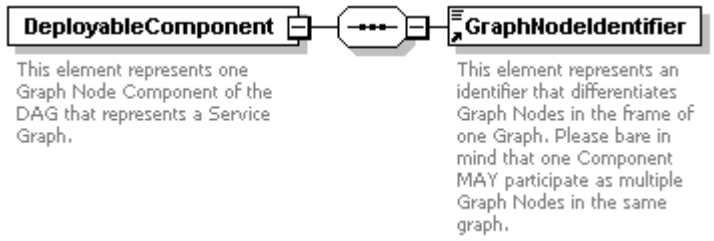
diagram	
properties	minOcc 1 maxOcc unbounded content complex
children	DeployableComponent ServiceProviderDescriptor
annotation	documentation This element encapsulates all deployable actions that have to be performed by the ARCADIA Smart Controller through the ARCADIA Agents.
source	<xs:element name="ComponentPlacementAction" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element encapsulates all deployable actions that have to be performed by the ARCADIA Smart Controller through the ARCADIA Agents. </xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="DeployableComponent"> <xs:annotation> <xs:documentation>This element represents one Graph Node Component of the DG that represents a Service Graph.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element ref="GraphNodeIdentifier"/> </xs:sequence> </xs:complexType> </xs:element> <xs:element name="ServiceProviderDescriptor"> <xs:annotation> <xs:documentation>This element encapsulates information regarding the association of one Component with the selected IaaS resources.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="IaaSConnectivity"> <xs:annotation> <xs:documentation>This element provides information about a specific IaaS.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="IaaSType"> <xs:annotation> <xs:documentation>This element provides information about the type of the IaaS

```
that is selected in the frame of one placement.</xs:documentation>
                </xs:annotation>
                <xs:simpleType>
                  <xs:restriction base="xs:string">
                    <xs:pattern value="OpenStack"/>
                  </xs:restriction>
                </xs:simpleType>
              </xs:element>
              <xs:element name="APIEndpoint" type="xs:string">
                <xs:annotation>
                  <xs:documentation>This element provides the public accessible API of the IaaS
provider.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="APIVersion">
                <xs:annotation>
                  <xs:documentation>This element provides information regarding the version of the
selected API in order to assure compatibility.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="Credentials" type="xs:string">
                <xs:annotation>
                  <xs:documentation>This elements encapsulates security information required in
order to perform authentication and authorization to the IaaS.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="TenantIdentifier" type="xs:string">
                <xs:annotation>
                  <xs:documentation>This element provides information regarding the specific tenant
that is associated with the deployment.</xs:documentation>
                </xs:annotation>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="DeploymentConstraints">
          <xs:annotation>
            <xs:documentation>This element encaptulates information related to the technical
constraints that are imposed by the ARCADIA Smart Controller in order to perform proper
instantiation of the Execution Environment.</xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:sequence>
              <xs:element name="ExecutionEnvironment" type="ExecutionEnvironmentType">
                <xs:annotation>
                  <xs:documentation>This element provides information related to the type of the
execution environment that is required.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="CoreRequirements">
                <xs:annotation>
                  <xs:documentation>This element encapsulates information related to actual
execution environment that SHALL be instantiated in an IaaS.</xs:documentation>
                </xs:annotation>
                <xs:complexType>
                  <xs:sequence>
```

```
<xs:element name="InstanceName" type="xs:string">
<xs:annotation>
<xs:documentation>InstanceName will be autogenerated based on the ComponentIdentifier</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="ImageIdentifier" type="xs:string">
<xs:annotation>
<xs:documentation>The Image that will be used will either be defined by the component or selected based on the available registred images in the IaaS.</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="Flavor">
<xs:annotation>
<xs:documentation>This element provides information related to the desired flavor that should be used in the chosen IaaS.</xs:documentation>
</xs:annotation>
<xs:complexType>
<xs:sequence>
<xs:element name="FlavorIdentifier"/>
<xs:element name="FlavorName" type="xs:string"/>
<xs:element name="vCPUs" type="xs:int">
<xs:annotation>
<xs:documentation>Number of CPUs</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="RootDisk" type="xs:int"/>
<xs:element name="EphemeralDisk" type="xs:int"/>
<xs:element name="RAM" type="xs:int"/>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="SecurityRequirements">
<xs:annotation>
<xs:documentation>This element provides security constraints that have to be met upon deployment. It will be elaborated in version 2 of the model.</xs:documentation>
</xs:annotation>
<xs:complexType>
<xs:sequence>
<xs:element name="SecurityGroupIdentifier"/>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="NetworkingRequirements">
<xs:annotation>
<xs:documentation>This element provides network constraints that have to be met upon deployment. It will be refined in version 2 of the model.</xs:documentation>
</xs:annotation>
<xs:complexType>
<xs:sequence>
<xs:element name="SubNetIdentifier"/>
</xs:sequence>
</xs:complexType>
```

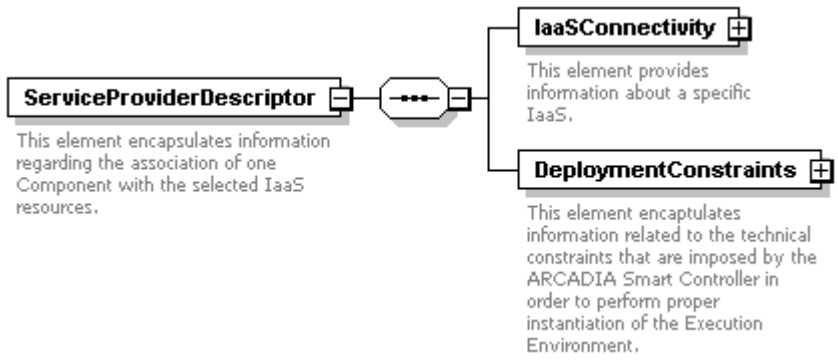
```
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction/DeployableComponent

diagram	
properties	content complex
children	**GraphNodeIdentifier**
annotation	documentation This element represents one Graph Node Component of the DG that represents a Service Graph.

source	`<xs:element name="DeployableComponent">`
	`<xs:annotation>`
	`<xs:documentation>`This element represents one Graph Node Component of the DG that represents a Service Graph.`</xs:documentation>`
	`</xs:annotation>`
	`<xs:complexType>`
	`<xs:sequence>`
	`<xs:element ref="GraphNodeIdentifier"/>`
	`</xs:sequence>`
	`</xs:complexType>`
	`</xs:element>`

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction/ServiceProviderDescriptor

diagram	
properties	content complex
children	IaaSConnectivity DeploymentConstraints
annotation	documentation This element encapsulates information regarding the association of one Component with the selected IaaS resources.

source

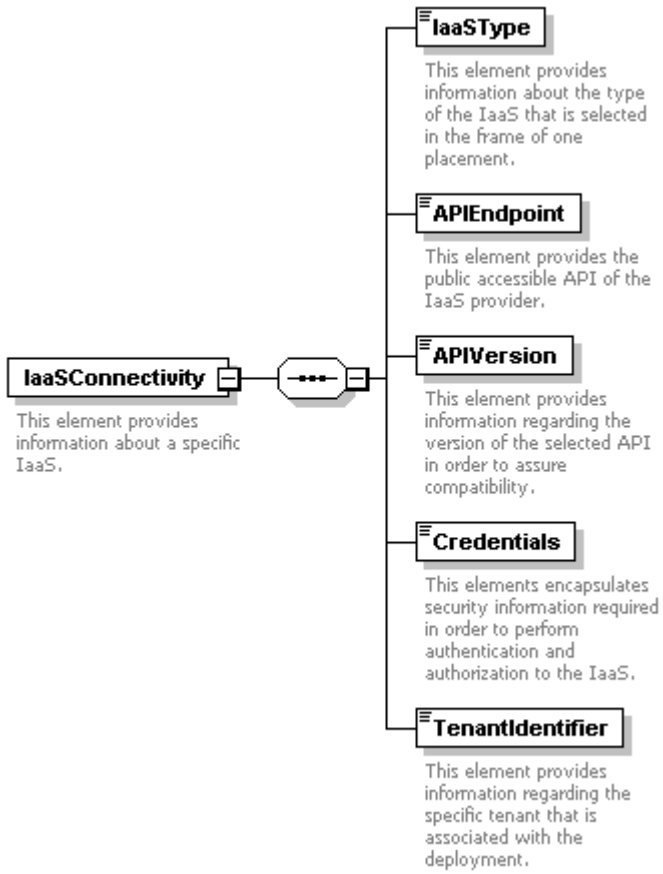
```
<xs:element name="ServiceProviderDescriptor">
 <xs:annotation>
  <xs:documentation>This element encapsulates information regarding the association of one Component with the selected IaaS resources.</xs:documentation>
 </xs:annotation>
 <xs:complexType>
  <xs:sequence>
   <xs:element name="IaaSConnectivity">
    <xs:annotation>
     <xs:documentation>This element provides information about a specific IaaS.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
     <xs:sequence>
      <xs:element name="IaaSType">
       <xs:annotation>
        <xs:documentation>This element provides information about the type of the IaaS that is selected in the frame of one placement.</xs:documentation>
       </xs:annotation>
       <xs:simpleType>
        <xs:restriction base="xs:string">
         <xs:pattern value="OpenStack"/>
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
      <xs:element name="APIEndpoint" type="xs:string">
       <xs:annotation>
        <xs:documentation>This element provides the public accessible API of the IaaS provider.</xs:documentation>
       </xs:annotation>
      </xs:element>
      <xs:element name="APIVersion">
       <xs:annotation>
        <xs:documentation>This element provides information regarding the version of the selected API in order to assure compatibility.</xs:documentation>
```

```
            </xs:annotation>
          </xs:element>
          <xs:element name="Credentials" type="xs:string">
            <xs:annotation>
              <xs:documentation>This elements encapsulates security information required in order to perform authentication and authorization to the IaaS.</xs:documentation>
            </xs:annotation>
          </xs:element>
          <xs:element name="TenantIdentifier" type="xs:string">
            <xs:annotation>
              <xs:documentation>This element provides information regarding the specific tenant that is associated with the deployment.</xs:documentation>
            </xs:annotation>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="DeploymentConstraints">
      <xs:annotation>
        <xs:documentation>This element encaptulates information related to the technical constraints that are imposed by the ARCADIA Smart Controller in order to perform proper instantiation of the Execution Environment.</xs:documentation>
      </xs:annotation>
      <xs:complexType>
        <xs:sequence>
          <xs:element name="ExecutionEnvironment" type="ExecutionEnvironmentType">
            <xs:annotation>
              <xs:documentation>This element provides information related to the type of the execution environment that is required.</xs:documentation>
            </xs:annotation>
          </xs:element>
          <xs:element name="CoreRequirements">
            <xs:annotation>
              <xs:documentation>This element encapsulates information related to actual execution environment that SHALL be instantiated in an IaaS.</xs:documentation>
            </xs:annotation>
            <xs:complexType>
              <xs:sequence>
                <xs:element name="InstanceName" type="xs:string">
                  <xs:annotation>
                    <xs:documentation>InstanceName will be autogenerated based on the ComponentIdentifier</xs:documentation>
                  </xs:annotation>
                </xs:element>
                <xs:element name="ImageIdentifier" type="xs:string">
                  <xs:annotation>
                    <xs:documentation>The Image that will be used will either be defined by the component or selected based on the available registred images in the IaaS.</xs:documentation>
                  </xs:annotation>
                </xs:element>
                <xs:element name="Flavor">
                  <xs:annotation>
                    <xs:documentation>This element provides information related to the desired flavor that should be used in the chosen IaaS.</xs:documentation>
                  </xs:annotation>
                  <xs:complexType>
```

```
                <xs:sequence>
                  <xs:element                                    name="FlavorIdentifier"/>
                  <xs:element                name="FlavorName"        type="xs:string"/>
                  <xs:element                   name="vCPUs"          type="xs:int">
                    <xs:annotation>
                      <xs:documentation>Number of CPUs</xs:documentation>
                    </xs:annotation>
                  </xs:element>
                  <xs:element                  name="RootDisk"        type="xs:int"/>
                  <xs:element                name="EphemeralDisk"     type="xs:int"/>
                  <xs:element                    name="RAM"           type="xs:int"/>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element                                          name="SecurityRequirements">
        <xs:annotation>
          <xs:documentation>This element provides security constraints that have to be met upon deployment. It will be elaborated in version 2 of the model.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element                                    name="SecurityGroupIdentifier"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element                                        name="NetworkingRequirements">
        <xs:annotation>
          <xs:documentation>This element provides network constraints that have to be met upon deployment. It will be refined in version 2 of the model.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element                                    name="SubNetIdentifier"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/IaaSConnectivity

diagram



properties — content: complex

children — IaaSType APIEndpoint APIVersion Credentials TenantIdentifier

annotation — documentation

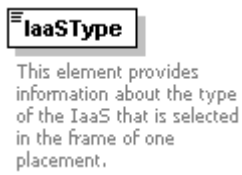
This element provides information about a specific IaaS.

source

```
<xs:element name="IaaSConnectivity">
 <xs:annotation>
  <xs:documentation>This element provides information about a specific IaaS.</xs:documentation>
 </xs:annotation>
 <xs:complexType>
  <xs:sequence>
   <xs:element name="IaaSType">
    <xs:annotation>
     <xs:documentation>This element provides information about the type of the IaaS that is selected in the frame of one placement.</xs:documentation>
    </xs:annotation>
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:pattern value="OpenStack"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
```

<table>
<tr><td></td><td>
<xs:element name="APIEndpoint" type="xs:string">
<xs:annotation>
<xs:documentation>This element provides the public accessible API of the IaaS provider.</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="APIVersion">
<xs:annotation>
<xs:documentation>This element provides information regarding the version of the selected API in order to assure compatibility.</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="Credentials" type="xs:string">
<xs:annotation>
<xs:documentation>This elements encapsulates security information required in order to perform authentication and authorization to the IaaS.</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="TenantIdentifier" type="xs:string">
<xs:annotation>
<xs:documentation>This element provides information regarding the specific tenant that is associated with the deployment.</xs:documentation>
</xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</td></tr>
</table>

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/IaaSConnectivity/IaaSType

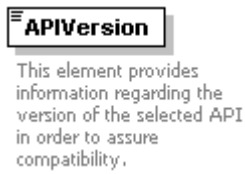
<table>
<tr><td>diagram</td><td> 
</td></tr>
<tr><td>type</td><td>restriction of xs:string</td></tr>
<tr><td>properties</td><td>content simple</td></tr>
<tr><td>facets</td><td>Kind Value Annotation
pattern OpenStack</td></tr>
<tr><td>annotation</td><td>documentation
This element provides information about the type of the IaaS that is selected in the frame of one placement.</td></tr>
<tr><td>source</td><td>
<xs:element name="IaaSType">
<xs:annotation>
<xs:documentation>This element provides information about the type of the IaaS that is selected in the frame of one placement.</xs:documentation>
</xs:annotation>
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:pattern value="OpenStack"/>
</td></tr>
</table>

	</xs:restriction> </xs:simpleType> </xs:element>

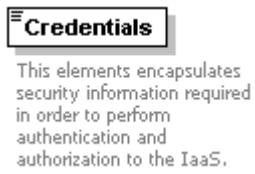
element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction/ServiceProviderDescriptor/IaaSConnectivity/APIEndpoint

diagram	
type	**xs:string**
properties	content simple
annotation	documentation This element provides the public accessible API of the IaaS provider.
source	<xs:element name="APIEndpoint" type="xs:string"> <xs:annotation> <xs:documentation>This element provides the public accessible API of the IaaS provider.</xs:documentation> </xs:annotation> </xs:element>

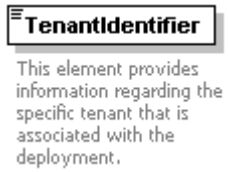
element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction/ServiceProviderDescriptor/IaaSConnectivity/APIVersion

diagram	
annotation	documentation This element provides information regarding the version of the selected API in order to assure compatibility.
source	<xs:element name="APIVersion"> <xs:annotation> <xs:documentation>This element provides information regarding the version of the selected API in order to assure compatibility.</xs:documentation> </xs:annotation> </xs:element>

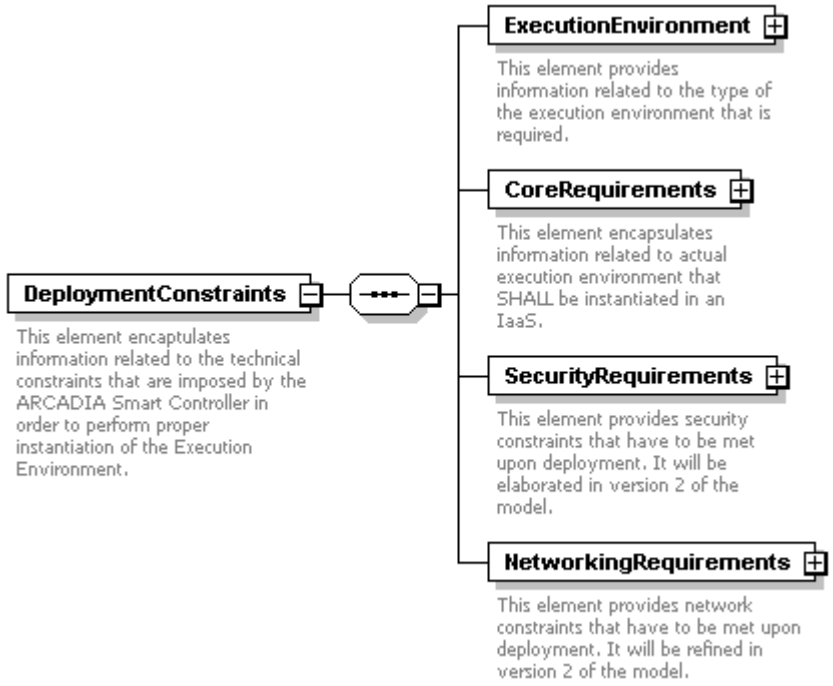
element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/IaaSConnectivity/Credentials

diagram	
type	**xs:string**
properties	content simple
annotation	documentation This elements encapsulates security information required in order to perform authentication and authorization to the IaaS.
source	<xs:element name="Credentials" type="xs:string"> <xs:annotation> <xs:documentation>This elements encapsulates security information required in order to perform authentication and authorization to the IaaS.</xs:documentation> </xs:annotation> </xs:element>

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/IaaSConnectivity/TenantIdentifier

diagram	
type	**xs:string**
properties	content simple
annotation	documentation This element provides information regarding the specific tenant that is associated with the deployment.
source	<xs:element name="TenantIdentifier" type="xs:string"> <xs:annotation> <xs:documentation>This element provides information regarding the specific tenant that is associated with the deployment.</xs:documentation> </xs:annotation> </xs:element>

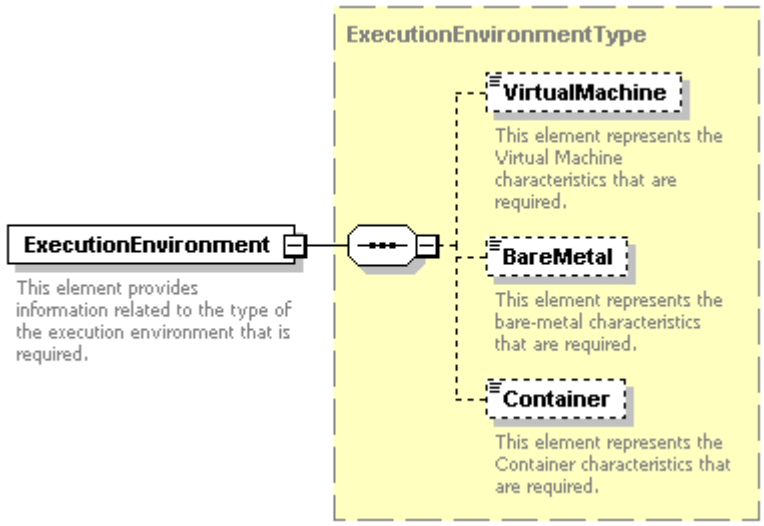
element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/DeploymentConstraints

diagram	
properties	content complex
children	ExecutionEnvironment CoreRequirements SecurityRequirements NetworkingRequirements
annotation	documentation This element encaptulates information related to the technical constraints that are imposed by the ARCADIA Smart Controller in order to perform proper instantiation of the Execution Environment.
source	<xs:element name="DeploymentConstraints"> <xs:annotation> <xs:documentation>This element encaptulates information related to the technical constraints that are imposed by the ARCADIA Smart Controller in order to perform proper instantiation of the Execution Environment.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="ExecutionEnvironment" type="ExecutionEnvironmentType"> <xs:annotation> <xs:documentation>This element provides information related to the type of the execution environment that is required.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="CoreRequirements"> <xs:annotation> <xs:documentation>This element encapsulates information related to actual execution environment that SHALL be instantiated in an IaaS.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="InstanceName" type="xs:string"> <xs:annotation> <xs:documentation>InstanceName will be autogenerated based on the

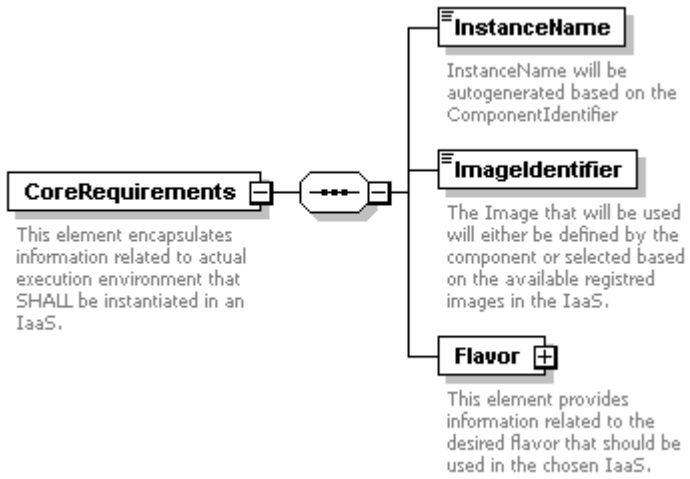
```
ComponentIdentifier</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="ImageIdentifier" type="xs:string">
          <xs:annotation>
            <xs:documentation>The Image that will be used will either be defined by the component or selected based on the available registred images in the IaaS.</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="Flavor">
          <xs:annotation>
            <xs:documentation>This element provides information related to the desired flavor that should be used in the chosen IaaS.</xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:sequence>
              <xs:element name="FlavorIdentifier"/>
              <xs:element name="FlavorName" type="xs:string"/>
              <xs:element name="vCPUs" type="xs:int">
                <xs:annotation>
                  <xs:documentation>Number of CPUs</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="RootDisk" type="xs:int"/>
              <xs:element name="EphemeralDisk" type="xs:int"/>
              <xs:element name="RAM" type="xs:int"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="SecurityRequirements">
    <xs:annotation>
      <xs:documentation>This element provides security constraints that have to be met upon deployment. It will be elaborated in version 2 of the model.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element name="SecurityGroupIdentifier"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="NetworkingRequirements">
    <xs:annotation>
      <xs:documentation>This element provides network constraints that have to be met upon deployment. It will be refined in version 2 of the model.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element name="SubNetIdentifier"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
 </xs:sequence>
</xs:complexType>
```

	</xs:element>

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/DeploymentConstraints/ExecutionEnvironment

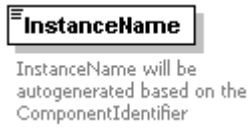
diagram	
type	**ExecutionEnvironmentType**
properties	content complex
children	**VirtualMachine BareMetal Container**
annotation	documentation This element provides information related to the type of the execution environment that is required.
source	<xs:element name="ExecutionEnvironment" type="ExecutionEnvironmentType"> <xs:annotation> <xs:documentation>This element provides information related to the type of the execution environment that is required.</xs:documentation> </xs:annotation> </xs:element>

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/DeploymentConstraints/CoreRequirements

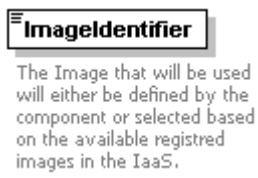
diagram	
properties	content complex
children	**InstanceName ImageIdentifier Flavor**
annotation	documentation This element encapsulates information related to actual execution environment that SHALL be instantiated in an IaaS.
source	<xs:element name="CoreRequirements"> <xs:annotation> <xs:documentation>This element encapsulates information related to actual execution environment that SHALL be instantiated in an IaaS.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="InstanceName" type="xs:string"> <xs:annotation> <xs:documentation>InstanceName will be autogenerated based on the ComponentIdentifier</xs:documentation> </xs:annotation> </xs:element> <xs:element name="ImageIdentifier" type="xs:string"> <xs:annotation> <xs:documentation>The Image that will be used will either be defined by the component or selected based on the available registred images in the IaaS.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="Flavor"> <xs:annotation> <xs:documentation>This element provides information related to the desired flavor that should be used in the chosen IaaS.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="FlavorIdentifier"/> <xs:element name="FlavorName" type="xs:string"/> <xs:element name="vCPUs" type="xs:int"> <xs:annotation> <xs:documentation>Number of CPUs</xs:documentation>

```
        </xs:annotation>
      </xs:element>
      <xs:element name="RootDisk" type="xs:int"/>
      <xs:element name="EphemeralDisk" type="xs:int"/>
      <xs:element name="RAM" type="xs:int"/>
    </xs:sequence>
  </xs:complexType>
 </xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
```

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/DeploymentConstraints/CoreRequirements/InstanceName

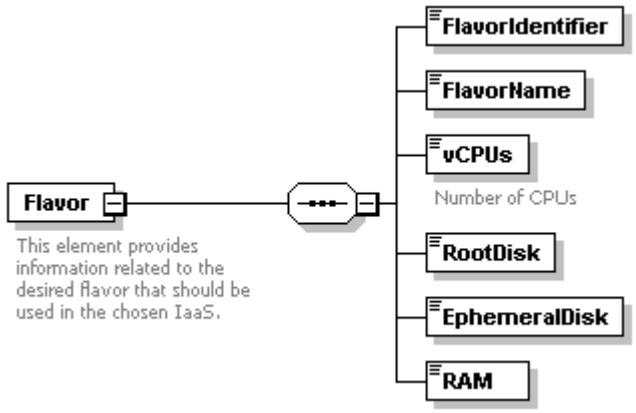
diagram	
type	**xs:string**
properties	content simple
annotation	documentation InstanceName will be autogenerated based on the ComponentIdentifier
source	<xs:element name="InstanceName" type="xs:string"> <xs:annotation> <xs:documentation>InstanceName will be autogenerated based on the ComponentIdentifier</xs:documentation> </xs:annotation> </xs:element>

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/DeploymentConstraints/CoreRequirements/ImageIdentifier

diagram	
type	**xs:string**
properties	content simple
annotation	documentation The Image that will be used will either be defined by the component or selected based on the available registred images in the IaaS.
source	<xs:element name="ImageIdentifier" type="xs:string"> <xs:annotation>

	<xs:documentation>The Image that will be used will either be defined by the component or selected based on the available registred images in the IaaS.</xs:documentation> </xs:annotation> </xs:element>

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/DeploymentConstraints/CoreRequirements/Flavor

diagram	
properties	content complex
children	FlavorIdentifier FlavorName vCPUs RootDisk EphemeralDisk RAM
annotation	documentation This element provides information related to the desired flavor that should be used in the chosen IaaS.
source	<xs:element name="Flavor"> <xs:annotation> <xs:documentation>This element provides information related to the desired flavor that should be used in the chosen IaaS.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="FlavorIdentifier"/> <xs:element name="FlavorName" type="xs:string"/> <xs:element name="vCPUs" type="xs:int"> <xs:annotation> <xs:documentation>Number of CPUs</xs:documentation> </xs:annotation> </xs:element> <xs:element name="RootDisk" type="xs:int"/> <xs:element name="EphemeralDisk" type="xs:int"/> <xs:element name="RAM" type="xs:int"/> </xs:sequence> </xs:complexType> </xs:element>

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction

/ServiceProviderDescriptor/DeploymentConstraints/CoreRequirements/Flavor/FlavorIdentifier

diagram	
source	<xs:element name="FlavorIdentifier"/>

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/DeploymentConstraints/CoreRequirements/Flavor/FlavorName

diagram	
type	**xs:string**
properties	content simple
source	<xs:element name="FlavorName" type="xs:string"/>

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/DeploymentConstraints/CoreRequirements/Flavor/vCPUs

diagram	
type	**xs:int**
properties	content simple
annotation	documentation Number of CPUs
source	<xs:element name="vCPUs" type="xs:int"> <xs:annotation> <xs:documentation>Number of CPUs</xs:documentation> </xs:annotation> </xs:element>

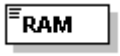
element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/DeploymentConstraints/CoreRequirements/Flavor/RootDisk

diagram	
type	**xs:int**
properties	content simple
source	<xs:element name="RootDisk" type="xs:int"/>

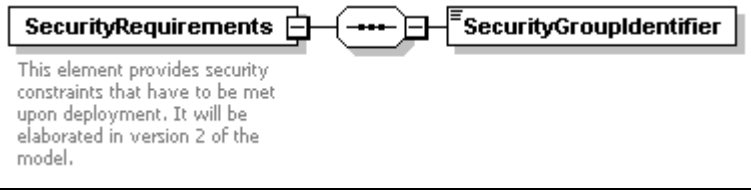
element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/DeploymentConstraints/CoreRequirements/Flavor/EphemeralDisk

diagram	
type	**xs:int**
properties	content simple
source	<xs:element name="EphemeralDisk" type="xs:int"/>

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/DeploymentConstraints/CoreRequirements/Flavor/RAM

diagram	
type	**xs:int**
properties	content simple
source	<xs:element name="RAM" type="xs:int"/>

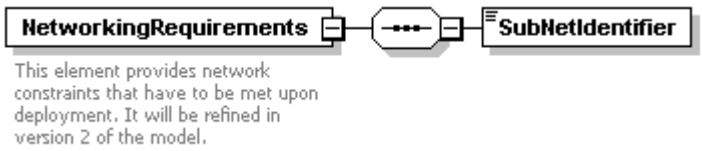
element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/DeploymentConstraints/SecurityRequirements

diagram	
properties	content complex
children	**SecurityGroupIdentifier**
annotation	documentation This element provides security constraints that have to be met upon deployment. It will be elaborated in version 2 of the model.
source	<xs:element name="SecurityRequirements"> <xs:annotation> <xs:documentation>This element provides security constraints that have to be met upon deployment. It will be elaborated in version 2 of the model.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="SecurityGroupIdentifier"/> </xs:sequence> </xs:complexType> </xs:element>

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/DeploymentConstraints/SecurityRequirements/SecurityGroupIdentifier

diagram	
source	<xs:element name="SecurityGroupIdentifier"/>

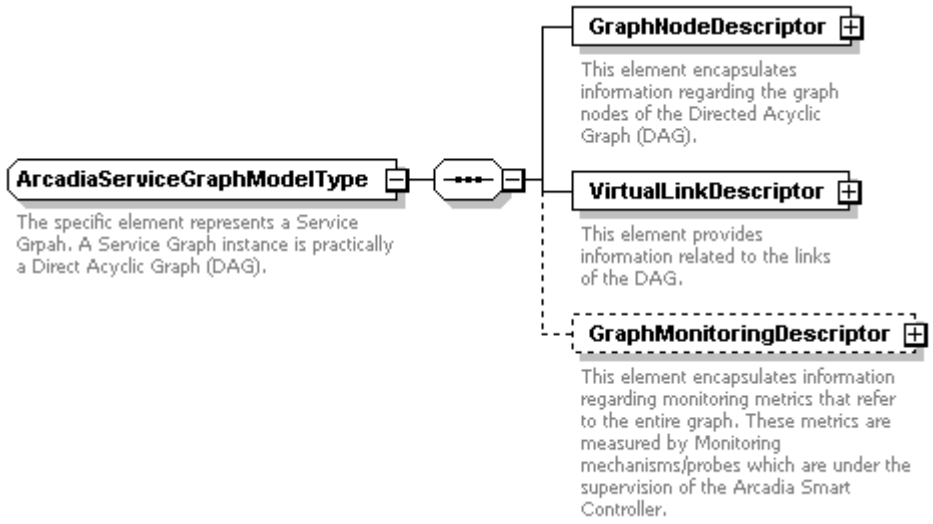
element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/DeploymentConstraints/NetworkingRequirements

diagram	
properties	content complex
children	SubNetIdentifier
annotation	documentation This element provides network constraints that have to be met upon deployment. It will be refined in version 2 of the model.
source	<xs:element name="NetworkingRequirements"> <xs:annotation> <xs:documentation>This element provides network constraints that have to be met upon deployment. It will be refined in version 2 of the model.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="SubNetIdentifier"/> </xs:sequence> </xs:complexType> </xs:element>

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction /ServiceProviderDescriptor/DeploymentConstraints/NetworkingRequirements/SubNetIdentifier

diagram	
source	<xs:element name="SubNetIdentifier"/>

complexType **ArcadiaServiceGraphModelType**

diagram	
children	GraphNodeDescriptor VirtualLinkDescriptor GraphMonitoringDescriptor
used by	elements ArcadiaServiceGraphModel ArcadiaServiceDeploymentModelType/ArcadiaServiceGraphModelReference
annotation	documentation The specific element represents a Service Grpah. A Service Graph instance is practically a Directed Graph (DG).

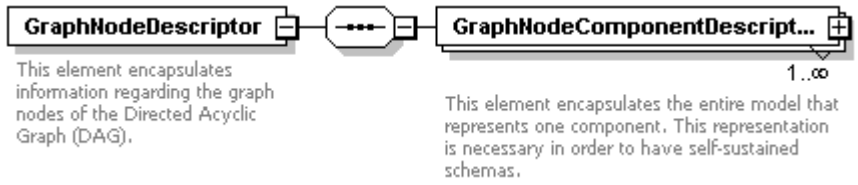
source

```
<xs:complexType name="ArcadiaServiceGraphModelType">
 <xs:annotation>
  <xs:documentation>The specific element represents a Service Grpah. A Service Graph instance is practically a Directed Graph (DG).</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element name="GraphNodeDescriptor">
   <xs:annotation>
    <xs:documentation>This element encapsulates information regarding the graph nodes of the Directed Graph (DG).</xs:documentation>
   </xs:annotation>
   <xs:complexType>
    <xs:sequence>
     <xs:element name="GraphNodeComponentDescriptor" maxOccurs="unbounded">
      <xs:annotation>
       <xs:documentation>This element encapsulates the entire model that represents one component. This representation is necessary in order to have self-sustained schemas.</xs:documentation>
      </xs:annotation>
      <xs:complexType>
       <xs:sequence>
        <xs:element name="GraphNodeIdentifierDescriptor" type="GraphNodeIdentifierType">
         <xs:annotation>
          <xs:documentation>This element provides the definition of the GraphNodeIdentifier along with its dependent identifiers i.e. the Component Identifier and the MicroService Identifier.</xs:documentation>
         </xs:annotation>
        </xs:element>
        <xs:element name="ComponentDescriptor" type="ArcadiaComponentModelType">
         <xs:annotation>
          <xs:documentation>This element represents one Component Model instance that is
```

```
used one or more times in the frame of a graph.</xs:documentation>
            </xs:annotation>
           </xs:element>
          </xs:sequence>
         </xs:complexType>
        </xs:element>
       </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="VirtualLinkDescriptor">
     <xs:annotation>
      <xs:documentation>This element provides information related to the links of the
DG.</xs:documentation>
     </xs:annotation>
     <xs:complexType>
      <xs:sequence>
       <xs:element name="VirtualLink" minOccurs="0" maxOccurs="unbounded">
        <xs:annotation>
         <xs:documentation>This element represents one Virtual Link of the Service
Graph.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
         <xs:sequence>
          <xs:element ref="VirtualLinkIdentifier"/>
          <xs:element name="SourceComponent">
           <xs:annotation>
            <xs:documentation>The element encapsulates information about the Source Node of
the Virtual Link.</xs:documentation>
           </xs:annotation>
           <xs:complexType>
            <xs:sequence>
             <xs:element ref="GraphNodeIdentifier"/>
             <xs:element ref="MicroServiceIdentifier"/>
            </xs:sequence>
           </xs:complexType>
          </xs:element>
          <xs:element name="DestinationComponent">
           <xs:annotation>
            <xs:documentation>The element encapsulates information about the Destination
Node of the Virtual Link.</xs:documentation>
           </xs:annotation>
           <xs:complexType>
            <xs:sequence>
             <xs:element ref="GraphNodeIdentifier"/>
             <xs:element ref="MicroServiceIdentifier"/>
            </xs:sequence>
           </xs:complexType>
          </xs:element>
         </xs:sequence>
        </xs:complexType>
       </xs:element>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
    <xs:element name="GraphMonitoringDescriptor" minOccurs="0">
     <xs:annotation>
```

```
      <xs:documentation>This element encapsulates information regarding monitoring metrics that refer to the entire graph. These metrics are measured by Monitoring mechanisms/probes which are under the supervision of the Arcadia Smart Controller.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
     <xs:sequence>
      <xs:element name="GraphMeasurableMetric" type="MeasureableMetricType" maxOccurs="unbounded">
       <xs:annotation>
        <xs:documentation>This element represents one metric that refers to the graph.</xs:documentation>
       </xs:annotation>
      </xs:element>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
</xs:complexType>
```

element **ArcadiaServiceGraphModelType/GraphNodeDescriptor**

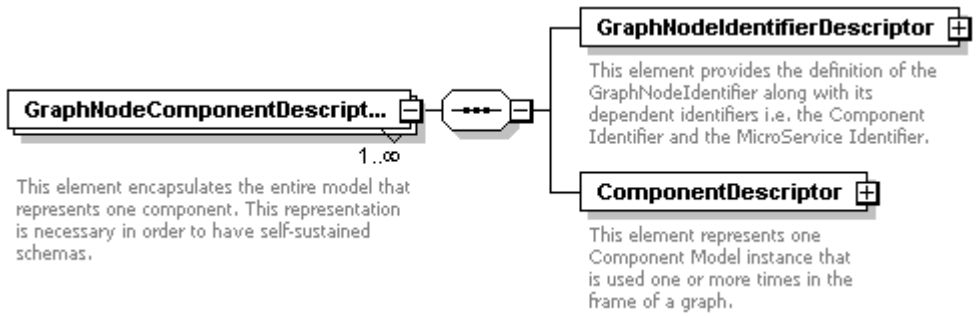
diagram	
properties	content complex
children	**GraphNodeComponentDescriptor**
annotation	documentation This element encapsulates information regarding the graph nodes of the Directed Graph (DG).

source

```
<xs:element name="GraphNodeDescriptor">
 <xs:annotation>
  <xs:documentation>This element encapsulates information regarding the graph nodes of the Directed Graph (DG).</xs:documentation>
 </xs:annotation>
 <xs:complexType>
  <xs:sequence>
   <xs:element name="GraphNodeComponentDescriptor" maxOccurs="unbounded">
    <xs:annotation>
     <xs:documentation>This element encapsulates the entire model that represents one component. This representation is necessary in order to have self-sustained schemas.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
     <xs:sequence>
      <xs:element name="GraphNodeIdentifierDescriptor" type="GraphNodeIdentifierType">
       <xs:annotation>
        <xs:documentation>This element provides the definition of the GraphNodeIdentifier along with its dependent identifiers i.e. the Component Identifier and the MicroService Identifier.</xs:documentation>
       </xs:annotation>
```

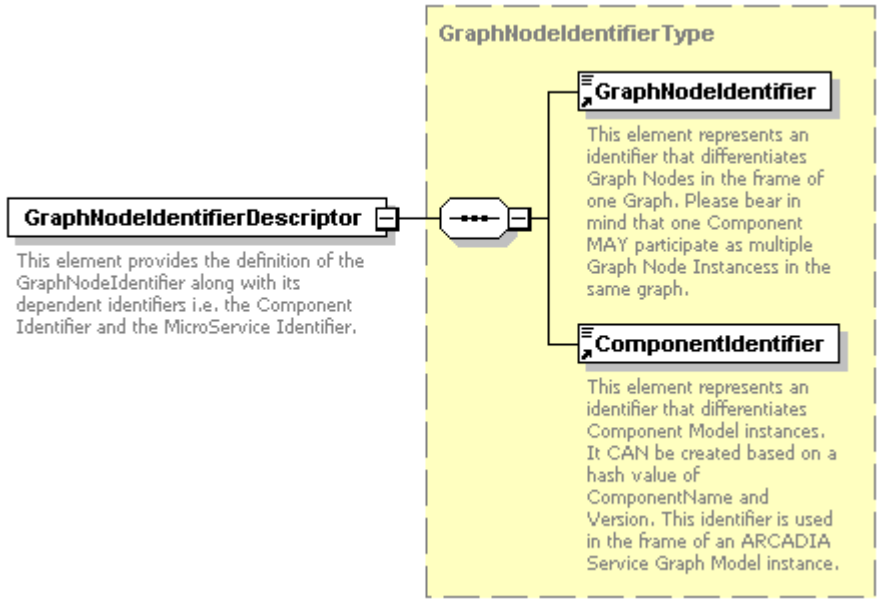
```
        </xs:element>
        <xs:element name="ComponentDescriptor" type="ArcadiaComponentModelType">
         <xs:annotation>
          <xs:documentation>This element represents one Component Model instance that is used one or more times in the frame of a graph.</xs:documentation>
         </xs:annotation>
        </xs:element>
       </xs:sequence>
      </xs:complexType>
     </xs:element>
    </xs:sequence>
   </xs:complexType>
</xs:element>
```

element **ArcadiaServiceGraphModelType/GraphNodeDescriptor/GraphNodeComponentDescriptor**

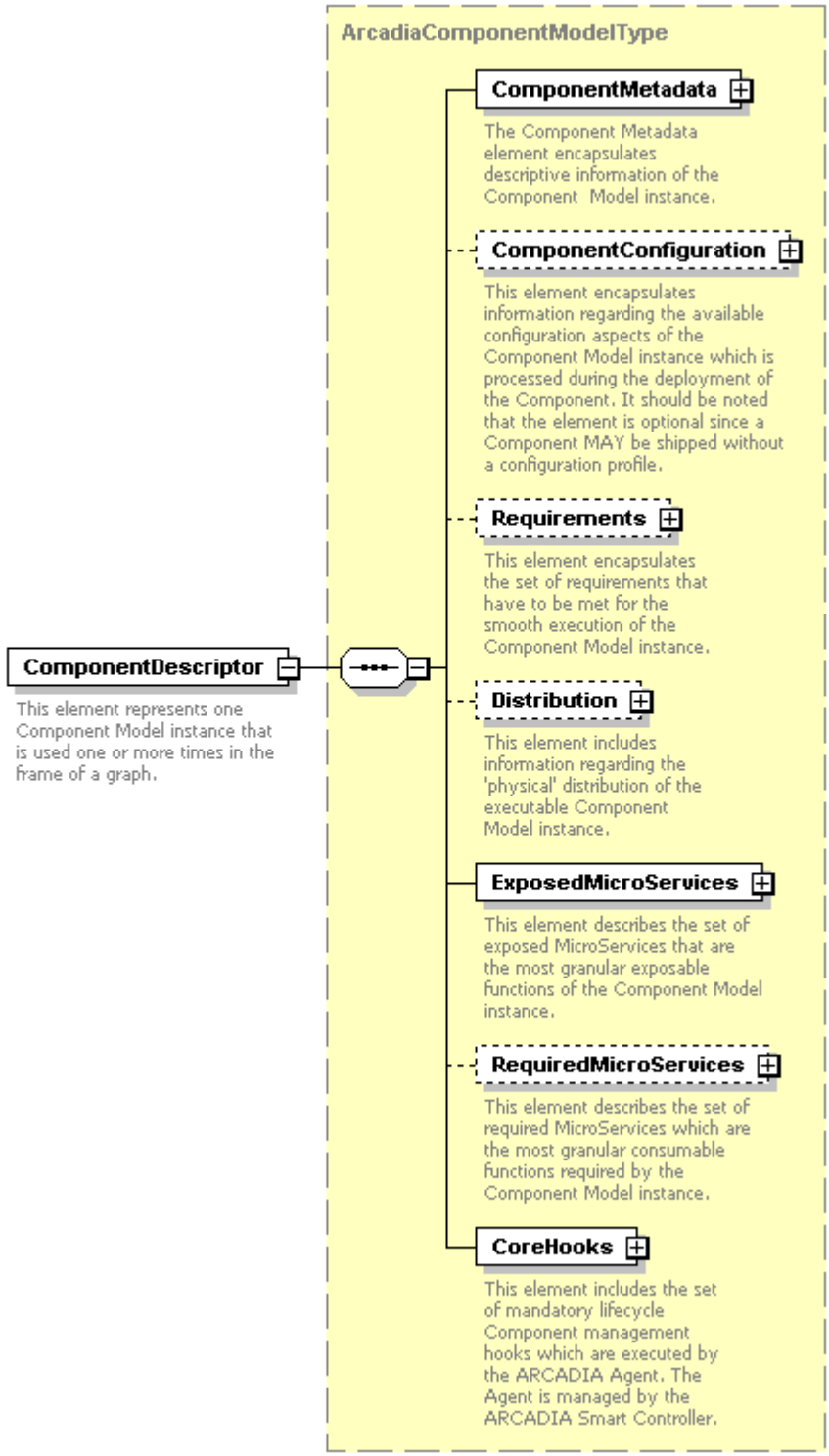
diagram	
properties	minOcc 1 maxOcc unbounded content complex
children	GraphNodeIdentifierDescriptor ComponentDescriptor
annotation	documentation This element encapsulates the entire model that represents one component. This representation is necessary in order to have self-sustained schemas.
source	<xs:element name="GraphNodeComponentDescriptor" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element encapsulates the entire model that represents one component. This representation is necessary in order to have self-sustained schemas.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="GraphNodeIdentifierDescriptor" type="GraphNodeIdentifierType"> <xs:annotation> <xs:documentation>This element provides the definition of the GraphNodeIdentifier along with its dependent identifiers i.e. the Component Identifier and the MicroService Identifier.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="ComponentDescriptor" type="ArcadiaComponentModelType"> <xs:annotation> <xs:documentation>This element represents one Component Model instance that is used one or more times in the frame of a graph.</xs:documentation>

```
      </xs:annotation>
    </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

element **ArcadiaServiceGraphModelType/GraphNodeDescriptor/GraphNodeComponentDescriptor/GraphNodeIdentifierDescriptor**

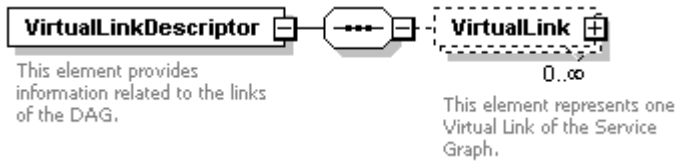
diagram	
type	GraphNodeIdentifierType
properties	content complex
children	GraphNodeIdentifier ComponentIdentifier
annotation	documentation This element provides the definition of the GraphNodeIdentifier along with its dependent identifiers i.e. the Component Identifier and the MicroService Identifier.
source	<xs:element name="GraphNodeIdentifierDescriptor" type="GraphNodeIdentifierType"> <xs:annotation> <xs:documentation>This element provides the definition of the GraphNodeIdentifier along with its dependent identifiers i.e. the Component Identifier and the MicroService Identifier.</xs:documentation> </xs:annotation> </xs:element>

element
ArcadiaServiceGraphModelType/GraphNodeDescriptor/GraphNodeComponentDescriptor/ComponentDescriptor

diagram	
type	ArcadiaComponentModelType
properties	content complex

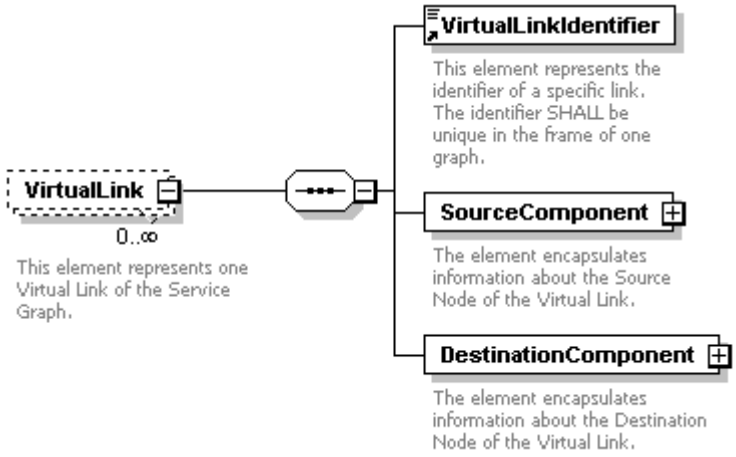
children	ComponentMetadata ComponentConfiguration Requirements Distribution ExposedMicroServices RequiredMicroServices CoreHooks
annotation	documentation This element represents one Component Model instance that is used one or more times in the frame of a graph.
source	<xs:element name="ComponentDescriptor" type="ArcadiaComponentModelType"> <xs:annotation> <xs:documentation>This element represents one Component Model instance that is used one or more times in the frame of a graph.</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaServiceGraphModelType/VirtualLinkDescriptor**

diagram	
properties	content complex
children	VirtualLink
annotation	documentation This element provides information related to the links of the DG.
source	<xs:element name="VirtualLinkDescriptor"> <xs:annotation> <xs:documentation>This element provides information related to the links of the DG.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="VirtualLink" minOccurs="0" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents one Virtual Link of the Service Graph.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element ref="VirtualLinkIdentifier"/> <xs:element name="SourceComponent"> <xs:annotation> <xs:documentation>The element encapsulates information about the Source Node of the Virtual Link.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element ref="GraphNodeIdentifier"/> <xs:element ref="MicroServiceIdentifier"/> </xs:sequence> </xs:complexType> </xs:element> <xs:element name="DestinationComponent"> <xs:annotation>

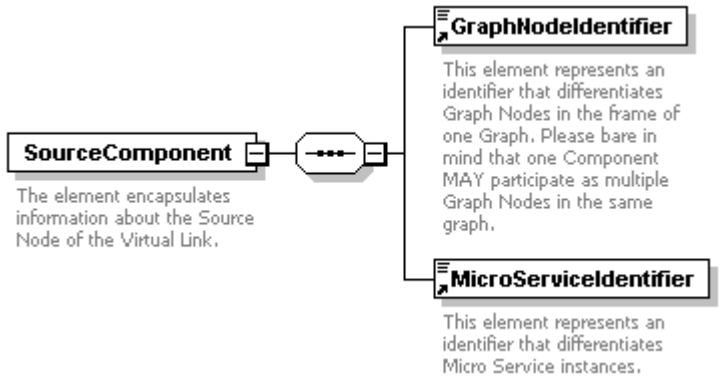
	<xs:documentation>The element encapsulates information about the Destination Node of the Virtual Link.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element ref="GraphNodeIdentifier"/> <xs:element ref="MicroServiceIdentifier"/> </xs:sequence> </xs:complexType> </xs:element> </xs:sequence> </xs:complexType> </xs:element> </xs:sequence> </xs:complexType> </xs:element>

element **ArcadiaServiceGraphModelType/VirtualLinkDescriptor/VirtualLink**

diagram	
properties	minOcc 0 maxOcc unbounded content complex
children	VirtualLinkIdentifier SourceComponent DestinationComponent
annotation	documentation This element represents one Virtual Link of the Service Graph.
source	<xs:element name="VirtualLink" minOccurs="0" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents one Virtual Link of the Service Graph.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element ref="VirtualLinkIdentifier"/> <xs:element name="SourceComponent"> <xs:annotation> <xs:documentation>The element encapsulates information about the Source Node of the Virtual Link.</xs:documentation>

```
      </xs:annotation>
      <xs:complexType>
       <xs:sequence>
        <xs:element                                   ref="GraphNodeIdentifier"/>
        <xs:element                                   ref="MicroServiceIdentifier"/>
       </xs:sequence>
      </xs:complexType>
     </xs:element>
     <xs:element                                      name="DestinationComponent">
      <xs:annotation>
       <xs:documentation>The element encapsulates information about the Destination Node of
the                                  Virtual                                  Link.</xs:documentation>
      </xs:annotation>
      <xs:complexType>
       <xs:sequence>
        <xs:element                                   ref="GraphNodeIdentifier"/>
        <xs:element                                   ref="MicroServiceIdentifier"/>
       </xs:sequence>
      </xs:complexType>
     </xs:element>
    </xs:sequence>
   </xs:complexType>
</xs:element>
```

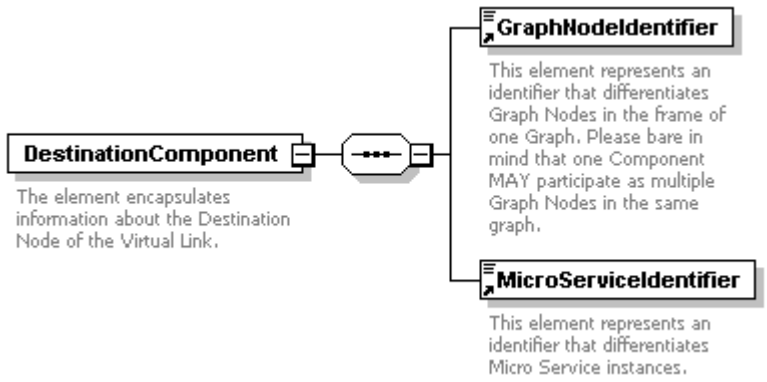
element **ArcadiaServiceGraphModelType/VirtualLinkDescriptor/VirtualLink/SourceComponent**

diagram	
properties	content complex
children	**GraphNodeIdentifier** **MicroServiceIdentifier**
annotation	documentation The element encapsulates information about the Source Node of the Virtual Link.
source	<xs:element name="SourceComponent"> <xs:annotation> <xs:documentation>The element encapsulates information about the Source Node of the Virtual Link.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element ref="GraphNodeIdentifier"/> <xs:element ref="MicroServiceIdentifier"/> </xs:sequence>

```
  </xs:complexType>
</xs:element>
```

element **ArcadiaServiceGraphModelType/VirtualLinkDescriptor/VirtualLink/DestinationComponent**

diagram



properties: content complex

children: **GraphNodeIdentifier** **MicroServiceIdentifier**

annotation: documentation

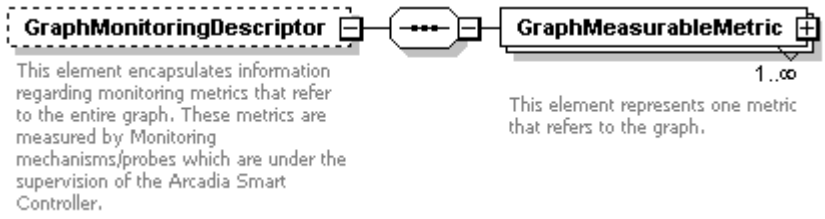
The element encapsulates information about the Destination Node of the Virtual Link.

source

```
<xs:element name="DestinationComponent">
  <xs:annotation>
    <xs:documentation>The element encapsulates information about the Destination Node of the Virtual Link.</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="GraphNodeIdentifier"/>
      <xs:element ref="MicroServiceIdentifier"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

element **ArcadiaServiceGraphModelType/GraphMonitoringDescriptor**

diagram



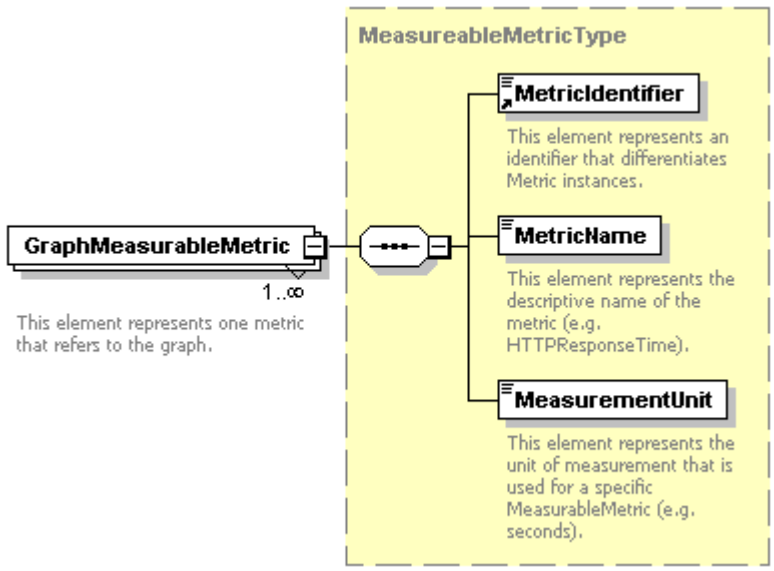
properties: minOcc 0; maxOcc 1; content complex

children: **GraphMeasurableMetric**

annotation: documentation

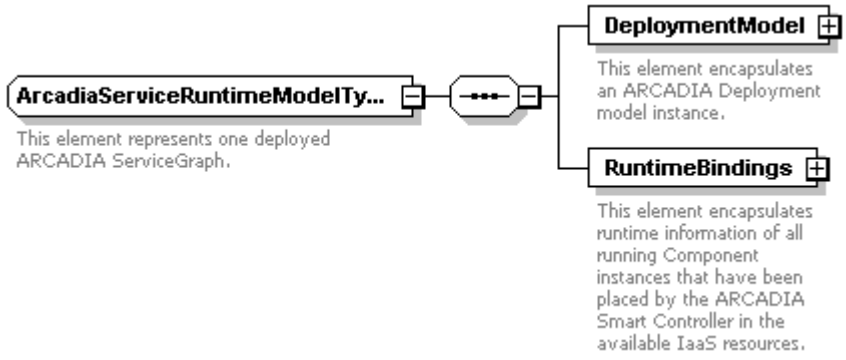
	This element encapsulates information regarding monitoring metrics that refer to the entire graph. These metrics are measured by Monitoring mechanisms/probes which are under the supervision of the Arcadia Smart Controller.
source	<xs:element name="GraphMonitoringDescriptor" minOccurs="0"> <xs:annotation> <xs:documentation>This element encapsulates information regarding monitoring metrics that refer to the entire graph. These metrics are measured by Monitoring mechanisms/probes which are under the supervision of the Arcadia Smart Controller.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="GraphMeasurableMetric" type="MeasureableMetricType" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents one metric that refers to the graph.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType> </xs:element>

element **ArcadiaServiceGraphModelType/GraphMonitoringDescriptor/GraphMeasurableMetric**

diagram	
type	MeasureableMetricType
properties	minOcc 1 maxOcc unbounded content complex
children	MetricIdentifier MetricName MeasurementUnit
annotation	documentation This element represents one metric that refers to the graph.
source	<xs:element name="GraphMeasurableMetric" type="MeasureableMetricType" maxOccurs="unbounded"> <xs:annotation>

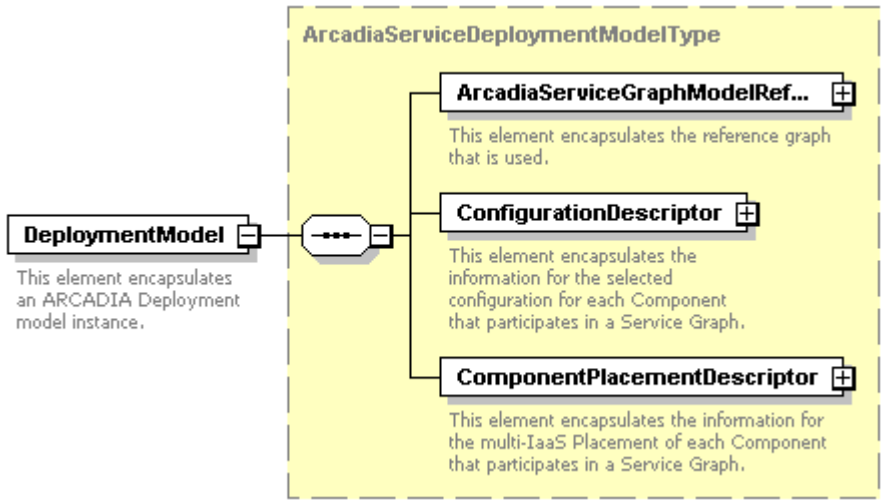
	<xs:documentation>This element represents one metric that refers to the graph.</xs:documentation> </xs:annotation> </xs:element>

complexType **ArcadiaServiceRuntimeModelType**

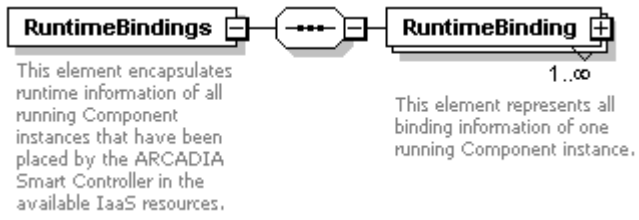
diagram	
children	DeploymentModel RuntimeBindings
used by	element ArcadiaServiceRuntimeModel
annotation	documentation This element represents one deployed ARCADIA ServiceGraph.
source	<xs:complexType name="ArcadiaServiceRuntimeModelType"> <xs:annotation> <xs:documentation>This element represents one deployed ARCADIA ServiceGraph.</xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="DeploymentModel" type="ArcadiaServiceDeploymentModelType"> <xs:annotation> <xs:documentation>This element encapsulates an ARCADIA Deployment model instance.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="RuntimeBindings"> <xs:annotation> <xs:documentation>This element encapsulates runtime information of all running Component instances that have been placed by the ARCADIA Smart Controller in the available IaaS resources.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="RuntimeBinding" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents all binding information of one running Component instance.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element ref="GraphNodeIdentifier"/> <xs:element name="IaaSPlacementDescriptor">

```
                <xs:annotation>
                  <xs:documentation>This element provides IaaS runtime details. It will be refined in version 2 of the model.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="MicroServiceRuntimeConfiguration">
                <xs:annotation>
                  <xs:documentation>This element provides Configuration runtime details. It will be refined in version 2 of the model.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="MicroServiceEndpointDescriptor">
                <xs:annotation>
                  <xs:documentation>This element provides Endpoint runtime details. It will be refined in version 2 of the model.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="MicroServiceStatusDescriptor">
                <xs:annotation>
                  <xs:documentation>This element provides runtime status details. It will be refined in version 2 of the model.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="MicroServiceMonitoringDescriptor">
                <xs:annotation>
                  <xs:documentation>This element provides runtime monitoring details. It will be refined in version 2 of the model.</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="LinkMonitoringDescriptor">
                <xs:annotation>
                  <xs:documentation>This element provides Link Monitoring runtime details. It will be refined in version 2 of the model.</xs:documentation>
                </xs:annotation>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
 </xs:sequence>
</xs:complexType>
```

element **ArcadiaServiceRuntimeModelType/DeploymentModel**

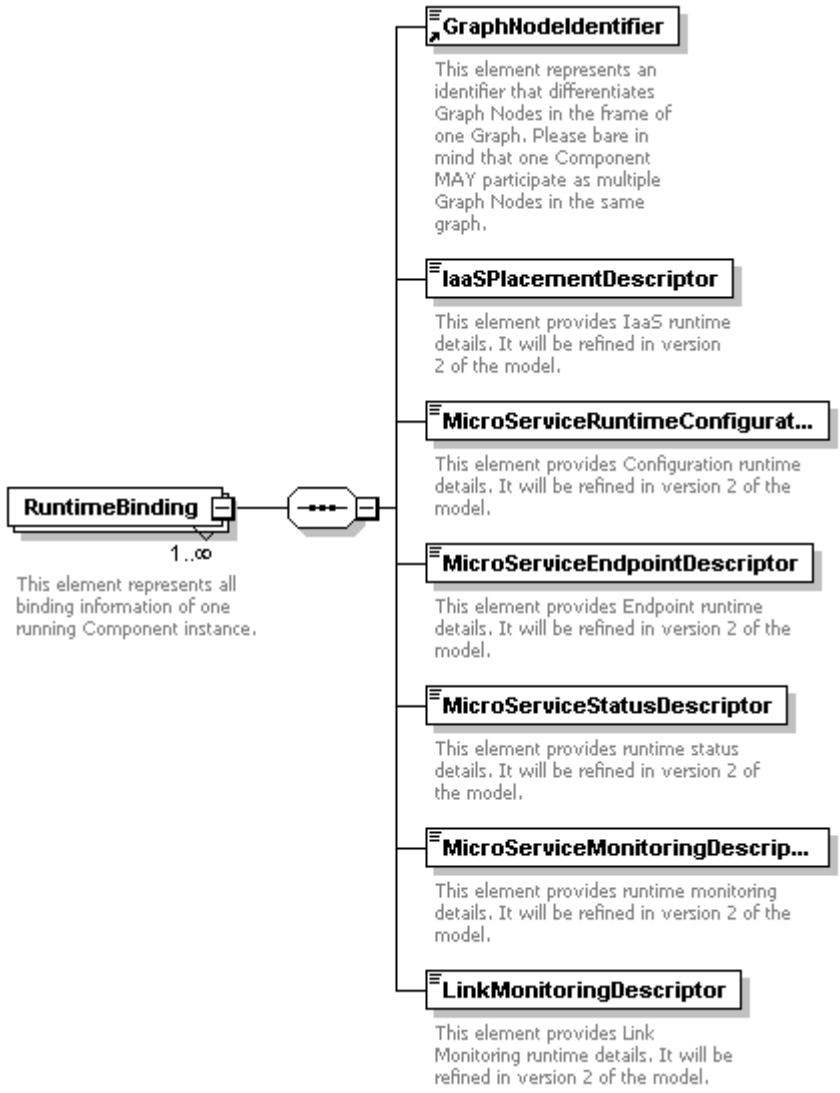
diagram	
type	ArcadiaServiceDeploymentModelType
properties	content complex
children	ArcadiaServiceGraphModelReference ConfigurationDescriptor ComponentPlacementDescriptor
annotation	documentation This element encapsulates an ARCADIA Deployment model instance.
source	<xs:element name="DeploymentModel" type="ArcadiaServiceDeploymentModelType"> <xs:annotation> <xs:documentation>This element encapsulates an ARCADIA Deployment model instance.</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaServiceRuntimeModelType/RuntimeBindings**

diagram	
properties	content complex
children	RuntimeBinding
annotation	documentation This element encapsulates runtime information of all running Component instances that have been placed by the ARCADIA Smart Controller in the available IaaS resources.
source	<xs:element name="RuntimeBindings"> <xs:annotation> <xs:documentation>This element encapsulates runtime information of all running Component instances that have been placed by the ARCADIA Smart Controller in the available IaaS resources.</xs:documentation> </xs:annotation>

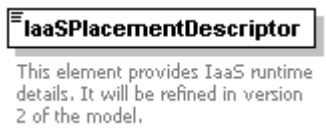
```
<xs:complexType>
 <xs:sequence>
  <xs:element name="RuntimeBinding" maxOccurs="unbounded">
   <xs:annotation>
    <xs:documentation>This element represents all binding information of one running Component instance.</xs:documentation>
   </xs:annotation>
   <xs:complexType>
    <xs:sequence>
     <xs:element ref="GraphNodeIdentifier"/>
     <xs:element name="IaaSPlacementDescriptor">
      <xs:annotation>
       <xs:documentation>This element provides IaaS runtime details. It will be refined in version 2 of the model.</xs:documentation>
      </xs:annotation>
     </xs:element>
     <xs:element name="MicroServiceRuntimeConfiguration">
      <xs:annotation>
       <xs:documentation>This element provides Configuration runtime details. It will be refined in version 2 of the model.</xs:documentation>
      </xs:annotation>
     </xs:element>
     <xs:element name="MicroServiceEndpointDescriptor">
      <xs:annotation>
       <xs:documentation>This element provides Endpoint runtime details. It will be refined in version 2 of the model.</xs:documentation>
      </xs:annotation>
     </xs:element>
     <xs:element name="MicroServiceStatusDescriptor">
      <xs:annotation>
       <xs:documentation>This element provides runtime status details. It will be refined in version 2 of the model.</xs:documentation>
      </xs:annotation>
     </xs:element>
     <xs:element name="MicroServiceMonitoringDescriptor">
      <xs:annotation>
       <xs:documentation>This element provides runtime monitoring details. It will be refined in version 2 of the model.</xs:documentation>
      </xs:annotation>
     </xs:element>
     <xs:element name="LinkMonitoringDescriptor">
      <xs:annotation>
       <xs:documentation>This element provides Link Monitoring runtime details. It will be refined in version 2 of the model.</xs:documentation>
      </xs:annotation>
     </xs:element>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
 </xs:sequence>
</xs:complexType>
</xs:element>
```

element **ArcadiaServiceRuntimeModelType/RuntimeBindings/RuntimeBinding**

diagram	
properties	minOcc 1 maxOcc unbounded content complex
children	GraphNodeIdentifier IaaSPlacementDescriptor MicroServiceRuntimeConfiguration MicroServiceEndpointDescriptor MicroServiceStatusDescriptor MicroServiceMonitoringDescriptor LinkMonitoringDescriptor
annotation	documentation This element represents all binding information of one running Component instance.
source	<xs:element name="RuntimeBinding" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents all binding information of one running Component instance.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element ref="GraphNodeIdentifier"/> <xs:element name="IaaSPlacementDescriptor"> <xs:annotation>

<table>
<tr><td></td><td>
<xs:documentation>This element provides IaaS runtime details. It will be refined in version 2 of the model.</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="MicroServiceRuntimeConfiguration">
<xs:annotation>
<xs:documentation>This element provides Configuration runtime details. It will be refined in version 2 of the model.</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="MicroServiceEndpointDescriptor">
<xs:annotation>
<xs:documentation>This element provides Endpoint runtime details. It will be refined in version 2 of the model.</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="MicroServiceStatusDescriptor">
<xs:annotation>
<xs:documentation>This element provides runtime status details. It will be refined in version 2 of the model.</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="MicroServiceMonitoringDescriptor">
<xs:annotation>
<xs:documentation>This element provides runtime monitoring details. It will be refined in version 2 of the model.</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="LinkMonitoringDescriptor">
<xs:annotation>
<xs:documentation>This element provides Link Monitoring runtime details. It will be refined in version 2 of the model.</xs:documentation>
</xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</td></tr>
</table>

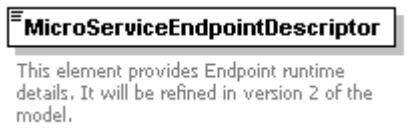
element
ArcadiaServiceRuntimeModelType/RuntimeBindings/RuntimeBinding/IaaSPlacementDescriptor

<table>
<tr><td>diagram</td><td> 
</td></tr>
<tr><td>annotation</td><td>documentation
This element provides IaaS runtime details. It will be refined in version 2 of the model.</td></tr>
<tr><td>source</td><td>
<xs:element name="IaaSPlacementDescriptor">
<xs:annotation>
<xs:documentation>This element provides IaaS runtime details. It will be refined in version 2 of the model.</xs:documentation>
</xs:annotation>
</xs:element>
</td></tr>
</table>

element
ArcadiaServiceRuntimeModelType/RuntimeBindings/RuntimeBinding/MicroServiceRuntimeConfiguration

diagram	
annotation	documentation This element provides Configuration runtime details. It will be refined in version 2 of the model.
source	<xs:element name="MicroServiceRuntimeConfiguration"> <xs:annotation> <xs:documentation>This element provides Configuration runtime details. It will be refined in version 2 of the model.</xs:documentation> </xs:annotation> </xs:element>

element
ArcadiaServiceRuntimeModelType/RuntimeBindings/RuntimeBinding/MicroServiceEndpointDescriptor

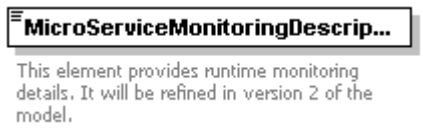
diagram	
annotation	documentation This element provides Endpoint runtime details. It will be refined in version 2 of the model.
source	<xs:element name="MicroServiceEndpointDescriptor"> <xs:annotation> <xs:documentation>This element provides Endpoint runtime details. It will be refined in version 2 of the model.</xs:documentation> </xs:annotation> </xs:element>

element
ArcadiaServiceRuntimeModelType/RuntimeBindings/RuntimeBinding/MicroServiceStatusDescriptor

diagram	
annotation	documentation This element provides runtime status details. It will be refined in version 2 of the model.
source	<xs:element name="MicroServiceStatusDescriptor"> <xs:annotation> <xs:documentation>This element provides runtime status details. It will be refined in version 2

	of the model.</xs:documentation> </xs:annotation> </xs:element>

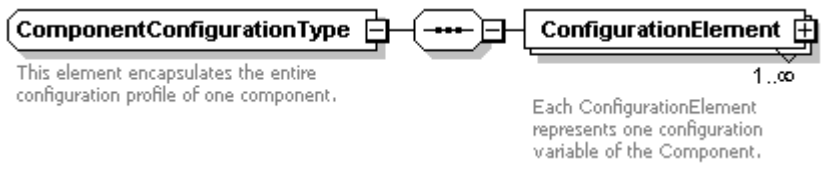
element **ArcadiaServiceRuntimeModelType/RuntimeBindings/RuntimeBinding/MicroServiceMonitoringDescriptor**

diagram	
annotation	documentation This element provides runtime monitoring details. It will be refined in version 2 of the model.
source	<xs:element name="MicroServiceMonitoringDescriptor"> <xs:annotation> <xs:documentation>This element provides runtime monitoring details. It will be refined in version 2 of the model.</xs:documentation> </xs:annotation> </xs:element>

element **ArcadiaServiceRuntimeModelType/RuntimeBindings/RuntimeBinding/LinkMonitoringDescriptor**

diagram	
annotation	documentation This element provides Link Monitoring runtime details. It will be refined in version 2 of the model.
source	<xs:element name="LinkMonitoringDescriptor"> <xs:annotation> <xs:documentation>This element provides Link Monitoring runtime details. It will be refined in version 2 of the model.</xs:documentation> </xs:annotation> </xs:element>

complexType **ComponentConfigurationType**

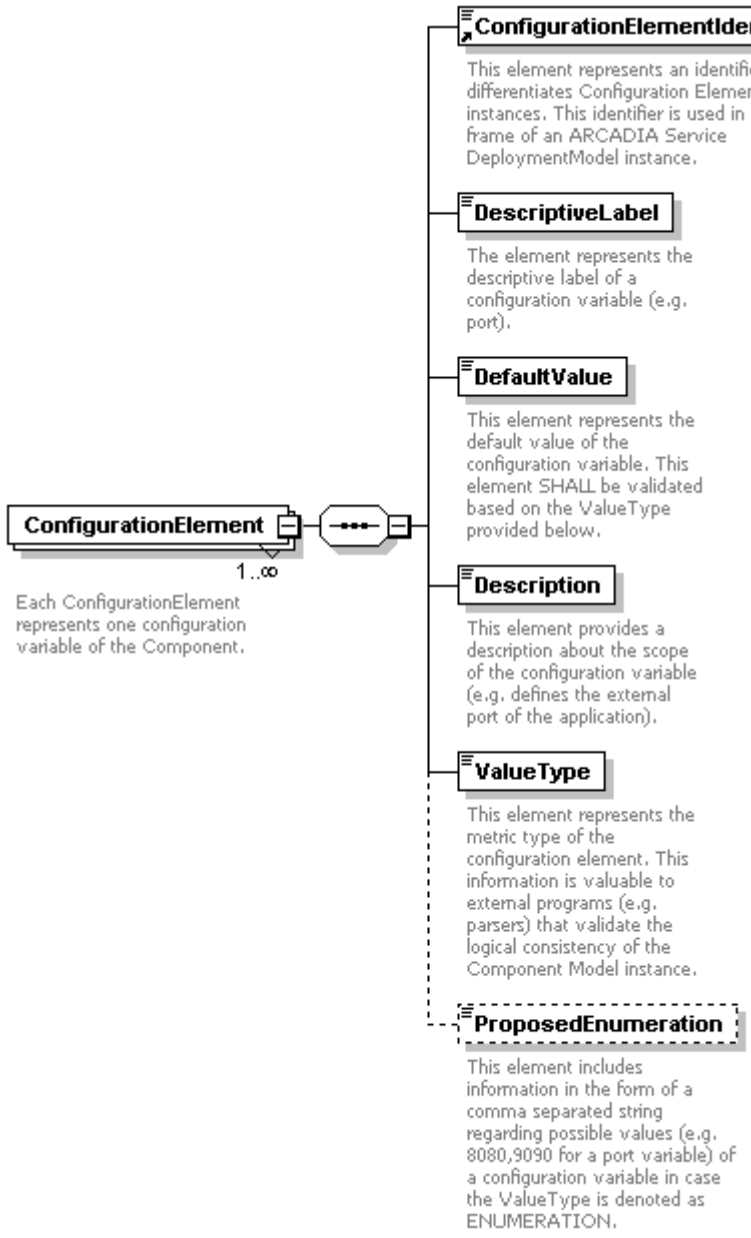
diagram	
children	**ConfigurationElement**
used by	element **ArcadiaComponentModelType/ComponentConfiguration**

annotation	documentation This element encapsulates the entire configuration profile of one component.

source

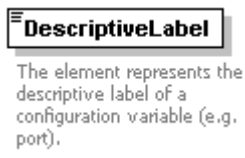
```
<xs:complexType name="ComponentConfigurationType">
 <xs:annotation>
  <xs:documentation>This element encapsulates the entire configuration profile of one component.</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element name="ConfigurationElement" maxOccurs="unbounded">
   <xs:annotation>
    <xs:documentation>Each ConfigurationElement represents one configuration variable of the Component.</xs:documentation>
   </xs:annotation>
   <xs:complexType>
    <xs:sequence>
     <xs:element ref="ConfigurationElementIdentifier"/>
     <xs:element name="DescriptiveLabel" type="xs:string">
      <xs:annotation>
       <xs:documentation>The element represents the descriptive label of a configuration variable (e.g. port).</xs:documentation>
      </xs:annotation>
     </xs:element>
     <xs:element name="DefaultValue" type="xs:string">
      <xs:annotation>
       <xs:documentation>This element represents the default value of the configuration variable. This element SHALL be validated based on the ValueType provided below.</xs:documentation>
      </xs:annotation>
     </xs:element>
     <xs:element name="Description" type="xs:string">
      <xs:annotation>
       <xs:documentation>This element provides a description about the scope of the configuration variable (e.g. defines the external port of the application).</xs:documentation>
      </xs:annotation>
     </xs:element>
     <xs:element name="ValueType" type="ConfigurationValueType">
      <xs:annotation>
       <xs:documentation>This element represents the metric type of the configuration element. This information is valuable to external programs (e.g. parsers) that validate the logical consistency of the Component Model instance.</xs:documentation>
      </xs:annotation>
     </xs:element>
     <xs:element name="ProposedEnumeration" type="xs:string" minOccurs="0">
      <xs:annotation>
       <xs:documentation>This element includes information in the form of a comma separated string regarding possible values (e.g. 8080,9090 for a port variable) of a configuration variable in case the ValueType is denoted as ENUMERATION.</xs:documentation>
      </xs:annotation>
     </xs:element>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
 </xs:sequence>
</xs:complexType>
```

element **ComponentConfigurationType/ConfigurationElement**

diagram	
properties	minOcc 1 maxOcc unbounded content complex
children	ConfigurationElementIdentifier DescriptiveLabel DefaultValue Description ValueType ProposedEnumeration
annotation	documentation Each ConfigurationElement represents one configuration variable of the Component.
source	<xs:element name="ConfigurationElement" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>Each ConfigurationElement represents one configuration variable of the Component.</xs:documentation> </xs:annotation>

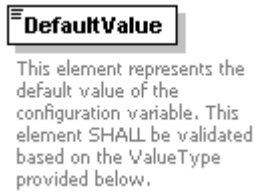
```
<xs:complexType>
 <xs:sequence>
  <xs:element ref="ConfigurationElementIdentifier"/>
  <xs:element name="DescriptiveLabel" type="xs:string">
   <xs:annotation>
    <xs:documentation>The element represents the descriptive label of a configuration variable (e.g. port).</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="DefaultValue" type="xs:string">
   <xs:annotation>
    <xs:documentation>This element represents the default value of the configuration variable. This element SHALL be validated based on the ValueType provided below.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="Description" type="xs:string">
   <xs:annotation>
    <xs:documentation>This element provides a description about the scope of the configuration variable (e.g. defines the external port of the application).</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="ValueType" type="ConfigurationValueType">
   <xs:annotation>
    <xs:documentation>This element represents the metric type of the configuration element. This information is valuable to external programs (e.g. parsers) that validate the logical consistency of the Component Model instance.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="ProposedEnumeration" type="xs:string" minOccurs="0">
   <xs:annotation>
    <xs:documentation>This element includes information in the form of a comma separated string regarding possible values (e.g. 8080,9090 for a port variable) of a configuration variable in case the ValueType is denoted as ENUMERATION.</xs:documentation>
   </xs:annotation>
  </xs:element>
 </xs:sequence>
</xs:complexType>
</xs:element>
```

element **ComponentConfigurationType/ConfigurationElement/DescriptiveLabel**

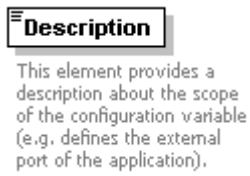
diagram	
type	**xs:string**
properties	content simple
annotation	documentation The element represents the descriptive label of a configuration variable (e.g. port).
source	<xs:element name="DescriptiveLabel" type="xs:string"> <xs:annotation>

	<xs:documentation>The element represents the descriptive label of a configuration variable (e.g. port).</xs:documentation> </xs:annotation> </xs:element>

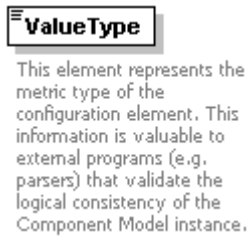
element **ComponentConfigurationType/ConfigurationElement/DefaultValue**

diagram	
type	**xs:string**
properties	content simple
annotation	documentation This element represents the default value of the configuration variable. This element SHALL be validated based on the ValueType provided below.
source	<xs:element name="DefaultValue" type="xs:string"> <xs:annotation> <xs:documentation>This element represents the default value of the configuration variable. This element SHALL be validated based on the ValueType provided below.</xs:documentation> </xs:annotation> </xs:element>

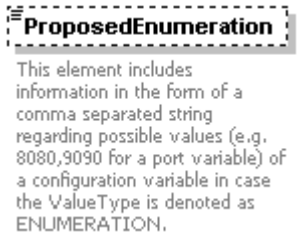
element **ComponentConfigurationType/ConfigurationElement/Description**

diagram	
type	**xs:string**
properties	content simple
annotation	documentation This element provides a description about the scope of the configuration variable (e.g. defines the external port of the application).
source	<xs:element name="Description" type="xs:string"> <xs:annotation> <xs:documentation>This element provides a description about the scope of the configuration variable (e.g. defines the external port of the application).</xs:documentation> </xs:annotation> </xs:element>

element **ComponentConfigurationType/ConfigurationElement/ValueType**

diagram	
type	ConfigurationValueType
properties	content simple
facets	Kind Value Annotation enumeration INTEGER enumeration STRING enumeration BOOLEAN enumeration SINGLE_SELECTION_ENUMERATION enumeration MULTI_SELECTION_ENUMERATION
annotation	documentation This element represents the metric type of the configuration element. This information is valuable to external programs (e.g. parsers) that validate the logical consistency of the Component Model instance.
source	<xs:element name="ValueType" type="ConfigurationValueType"> <xs:annotation> <xs:documentation>This element represents the metric type of the configuration element. This information is valuable to external programs (e.g. parsers) that validate the logical consistency of the Component Model instance.</xs:documentation> </xs:annotation> </xs:element>

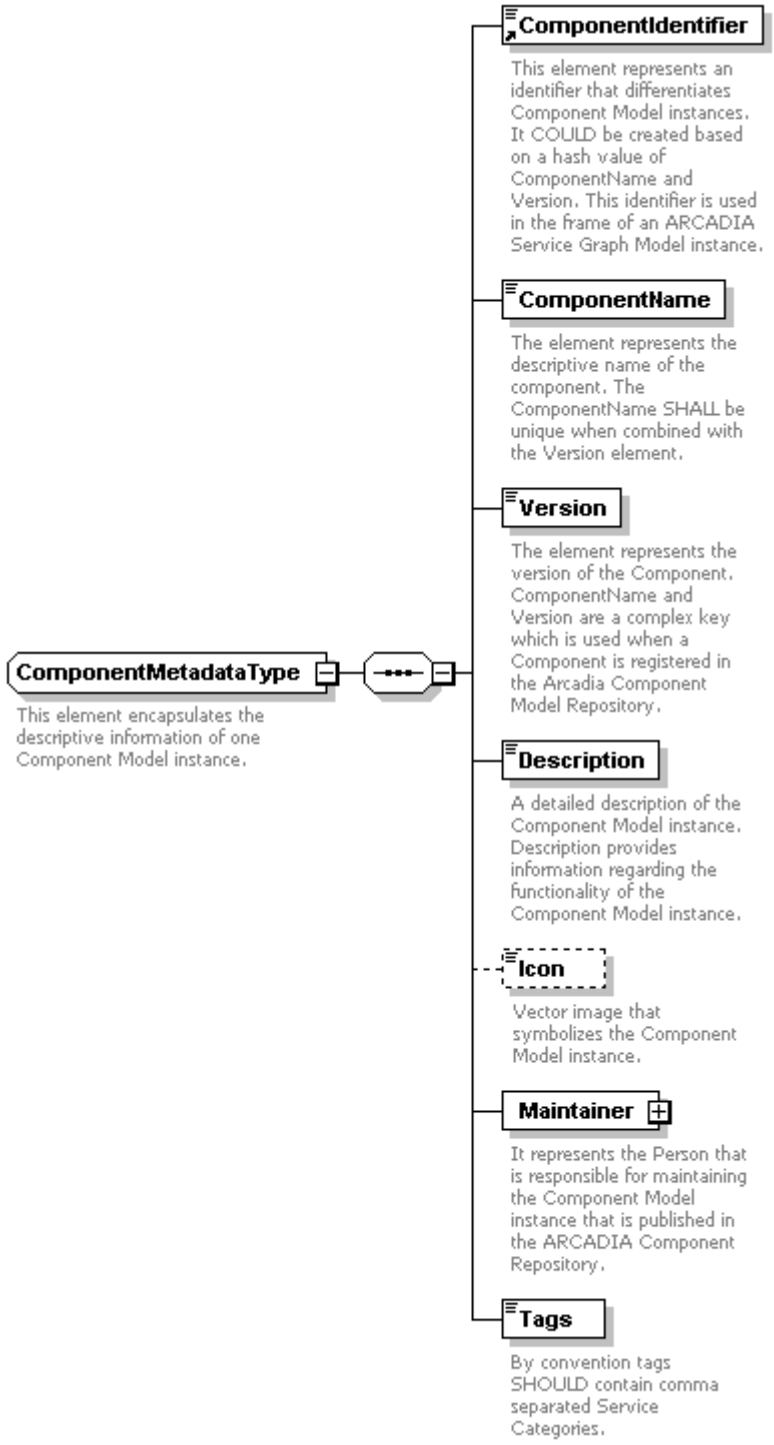
element **ComponentConfigurationType/ConfigurationElement/ProposedEnumeration**

diagram	
type	**xs:string**
properties	minOcc 0 maxOcc 1 content simple
annotation	documentation This element includes information in the form of a comma separated string regarding possible values (e.g. 8080,9090 for a port variable) of a configuration variable in case the ValueType is denoted as ENUMERATION.
source	<xs:element name="ProposedEnumeration" type="xs:string" minOccurs="0"> <xs:annotation> <xs:documentation>This element includes information in the form of a comma separated string

	regarding possible values (e.g. 8080,9090 for a port variable) of a configuration variable in case the ValueType is denoted as ENUMERATION.</xs:documentation> </xs:annotation> </xs:element>

complexType **ComponentMetadataType**

diagram

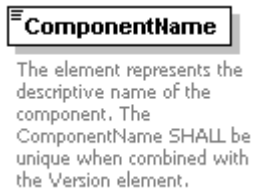


children	**ComponentIdentifier ComponentName Version Description Icon Maintainer Tags**
used by	element **ArcadiaComponentModelType/ComponentMetadata**
annotation	documentation This element encapsulates the descriptive information of one Component Model instance.

source

```
<xs:complexType name="ComponentMetadataType">
 <xs:annotation>
  <xs:documentation>This element encapsulates the descriptive information of one Component Model instance.</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element ref="ComponentIdentifier"/>
  <xs:element name="ComponentName" type="xs:string">
   <xs:annotation>
    <xs:documentation>The element represents the descriptive name of the component. The ComponentName SHALL be unique when combined with the Version element.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="Version" type="xs:string">
   <xs:annotation>
    <xs:documentation>The element represents the version of the Component. ComponentName and Version are a complex key which is used when a Component is registered in the Arcadia Component Model Repository.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="Description" type="xs:string">
   <xs:annotation>
    <xs:documentation>A detailed description of the Component Model instance. Description provides information regarding the functionality of the Component Model instance.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="Icon" type="xs:base64Binary" minOccurs="0">
   <xs:annotation>
    <xs:documentation>Vector image that symbolizes the Component Model instance.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="Maintainer" type="MaintainerType">
   <xs:annotation>
    <xs:documentation>It represents the Person that is responsible for maintaining the Component Model instance that is published in the ARCADIA Component Repository.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="Tags" type="xs:string">
   <xs:annotation>
    <xs:documentation>By convention tags SHOULD contain comma separated Service Categories.</xs:documentation>
   </xs:annotation>
  </xs:element>
 </xs:sequence>
</xs:complexType>
```

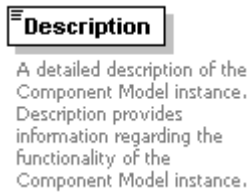
element **ComponentMetadataType/ComponentName**

diagram	
type	**xs:string**
properties	content simple
annotation	documentation The element represents the descriptive name of the component. The ComponentName SHALL be unique when combined with the Version element.
source	<xs:element name="ComponentName" type="xs:string"> <xs:annotation> <xs:documentation>The element represents the descriptive name of the component. The ComponentName SHALL be unique when combined with the Version element.</xs:documentation> </xs:annotation> </xs:element>

element **ComponentMetadataType/Version**

diagram	
type	**xs:string**
properties	content simple
annotation	documentation The element represents the version of the Component. ComponentName and Version are a complex key which is used when a Component is registered in the Arcadia Component Model Repository.
source	<xs:element name="Version" type="xs:string"> <xs:annotation> <xs:documentation>The element represents the version of the Component. ComponentName and Version are a complex key which is used when a Component is registered in the Arcadia Component Model Repository.</xs:documentation> </xs:annotation> </xs:element>

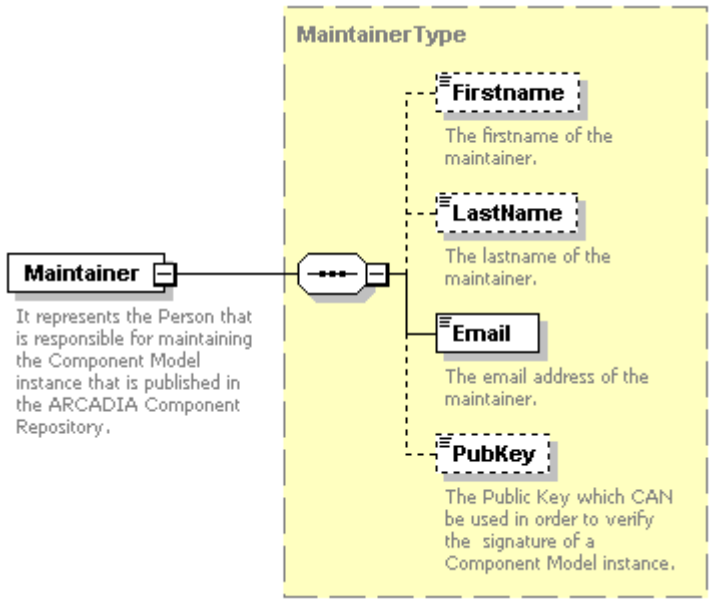
element **ComponentMetadataType/Description**

diagram	
type	**xs:string**
properties	content simple
annotation	documentation A detailed description of the Component Model instance. Description provides information regarding the functionality of the Component Model instance.
source	<xs:element name="Description" type="xs:string"> <xs:annotation> <xs:documentation>A detailed description of the Component Model instance. Description provides information regarding the functionality of the Component Model instance.</xs:documentation> </xs:annotation> </xs:element>

element **ComponentMetadataType/Icon**

diagram	
type	**xs:base64Binary**
properties	minOcc 0 maxOcc 1 content simple
annotation	documentation Vector image that symbolizes the Component Model instance.
source	<xs:element name="Icon" type="xs:base64Binary" minOccurs="0"> <xs:annotation> <xs:documentation>Vector image that symbolizes the Component Model instance.</xs:documentation> </xs:annotation> </xs:element>

element **ComponentMetadataType/Maintainer**

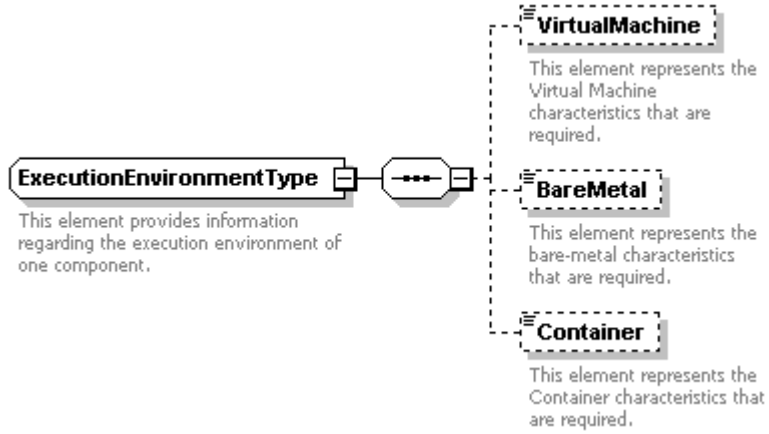
diagram	
type	**MaintainerType**
properties	content complex
children	**Firstname** **LastName** **Email** **PubKey**
annotation	documentation It represents the Person that is responsible for maintaining the Component Model instance that is published in the ARCADIA Component Repository.
source	<xs:element name="Maintainer" type="MaintainerType"> <xs:annotation> <xs:documentation>It represents the Person that is responsible for maintaining the Component Model instance that is published in the ARCADIA Component Repository.</xs:documentation> </xs:annotation> </xs:element>

element **ComponentMetadataType/Tags**

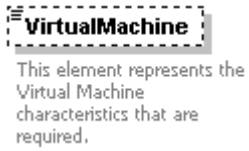
diagram	
type	**xs:string**
properties	content simple
annotation	documentation By convention tags SHOULD contain comma separated Service Categories.
source	<xs:element name="Tags" type="xs:string"> <xs:annotation> <xs:documentation>By convention tags SHOULD contain comma separated Service Categories.</xs:documentation> </xs:annotation>

	</xs:element>

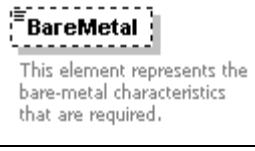
complexType **ExecutionEnvironmentType**

diagram	
children	VirtualMachine BareMetal Container
used by	elements ArcadiaComponentModelType/Requirements/HostingRequirements/ExecutionEnvironment ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction/ServiceProviderDescriptor/DeploymentConstraints/ExecutionEnvironment
annotation	documentation This element provides information regarding the execution environment of one component.
source	<xs:complexType name="ExecutionEnvironmentType"> <xs:annotation> <xs:documentation>This element provides information regarding the execution environment of one component.</xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="VirtualMachine" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the Virtual Machine characteristics that are required.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="BareMetal" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the bare-metal characteristics that are required.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="Container" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the Container characteristics that are required.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType>

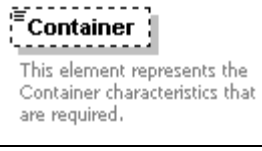
element **ExecutionEnvironmentType/VirtualMachine**

diagram	
properties	minOcc 0 maxOcc 1
annotation	documentation This element represents the Virtual Machine characteristics that are required.
source	<xs:element name="VirtualMachine" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the Virtual Machine characteristics that are required.</xs:documentation> </xs:annotation> </xs:element>

element **ExecutionEnvironmentType/BareMetal**

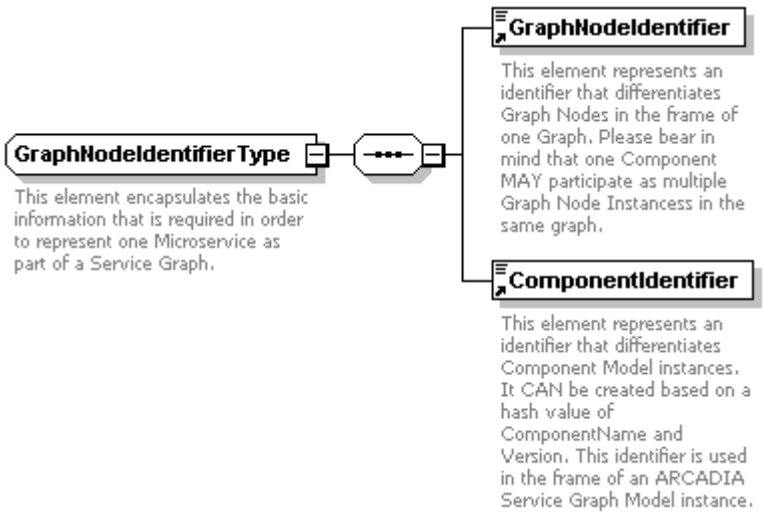
diagram	
properties	minOcc 0 maxOcc 1
annotation	documentation This element represents the bare-metal characteristics that are required.
source	<xs:element name="BareMetal" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the bare-metal characteristics that are required.</xs:documentation> </xs:annotation> </xs:element>

element **ExecutionEnvironmentType/Container**

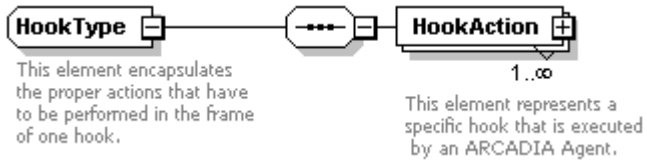
diagram	
properties	minOcc 0 maxOcc 1
annotation	documentation This element represents the Container characteristics that are required.
source	<xs:element name="Container" minOccurs="0"> <xs:annotation>

	<xs:documentation>This element represents the Container characteristics that are required.</xs:documentation> </xs:annotation> </xs:element>

complexType **GraphNodeIdentifierType**

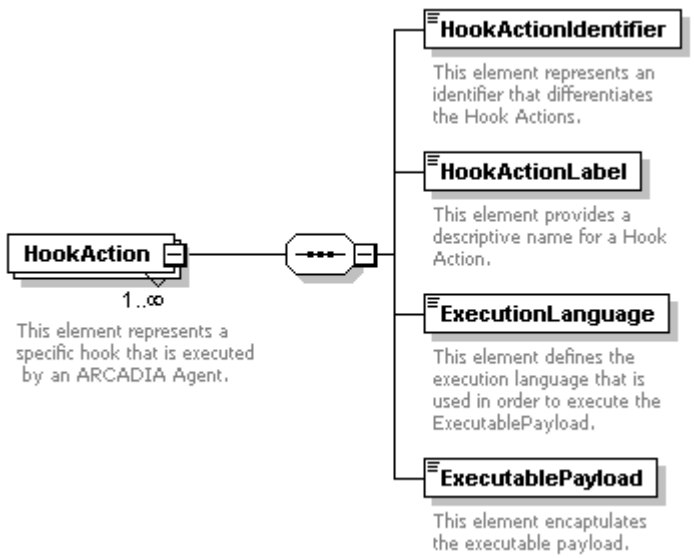
diagram	
children	GraphNodeIdentifier ComponentIdentifier
used by	element ArcadiaServiceGraphModelType/GraphNodeDescriptor/GraphNodeComponentDescriptor/GraphNodeIdentifierDescriptor
annotation	documentation This element encapsulates the basic information that is required in order to represent one Microservice as part of a Service Graph.
source	<xs:complexType name="GraphNodeIdentifierType"> <xs:annotation> <xs:documentation>This element encapsulates the basic information that is required in order to represent one Microservice as part of a Service Graph.</xs:documentation> </xs:annotation> <xs:sequence> <xs:element ref="GraphNodeIdentifier"> <xs:annotation> <xs:documentation>This element represents an identifier that differentiates Graph Nodes in the frame of one Graph. Please bear in mind that one Component MAY participate as multiple Graph Node Instancess in the same graph.</xs:documentation> </xs:annotation> </xs:element> <xs:element ref="ComponentIdentifier"> <xs:annotation> <xs:documentation>This element represents an identifier that differentiates Component Model instances. It CAN be created based on a hash value of ComponentName and Version. This identifier is used in the frame of an ARCADIA Service Graph Model instance.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType>

complexType **HookType**

diagram	
children	HookAction
used by	elements ArcadiaComponentModelType/CoreHooks/ConfigChangedHook ArcadiaComponentModelType/CoreHooks/InstallHook MicroServiceType/RelationHooks/RelationBrokenHook MicroServiceType/RelationHooks/RelationChangedHook MicroServiceType/RelationHooks/RelationDepartedHook MicroServiceType/RelationHooks/RelationJoinedHook ArcadiaComponentModelType/CoreHooks/StartHook ArcadiaComponentModelType/CoreHooks/StopHook ArcadiaComponentModelType/CoreHooks/UpgradeHook
annotation	documentation This element encapsulates the proper actions that have to be performed in the frame of one hook.
source	<xs:complexType name="HookType"> <xs:annotation> <xs:documentation>This element encapsulates the proper actions that have to be performed in the frame of one hook.</xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="HookAction" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents a specific hook that is executed by an ARCADIA Agent.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="HookActionIdentifier"> <xs:annotation> <xs:documentation>This element represents an identifier that differentiates the Hook Actions.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="HookActionLabel"> <xs:annotation> <xs:documentation>This element provides a descriptive name for a Hook Action.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="ExecutionLanguage" type="ExecutionLanguageType"> <xs:annotation> <xs:documentation>This element defines the execution language that is used in order to execute the ExecutablePayload.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="ExecutablePayload"> <xs:annotation> <xs:documentation>This element encaptulates the executable

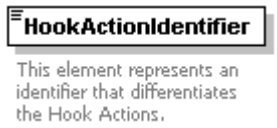
	payload.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType> </xs:element> </xs:sequence> </xs:complexType>

element **HookType/HookAction**

diagram	
properties	minOcc 1 maxOcc unbounded content complex
children	**HookActionIdentifier HookActionLabel ExecutionLanguage ExecutablePayload**
annotation	documentation This element represents a specific hook that is executed by an ARCADIA Agent.
source	<xs:element name="HookAction" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents a specific hook that is executed by an ARCADIA Agent.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="HookActionIdentifier"> <xs:annotation> <xs:documentation>This element represents an identifier that differentiates the Hook Actions.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="HookActionLabel"> <xs:annotation> <xs:documentation>This element provides a descriptive name for a Hook Action.</xs:documentation>

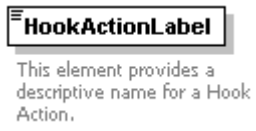
```
      </xs:annotation>
    </xs:element>
    <xs:element name="ExecutionLanguage" type="ExecutionLanguageType">
      <xs:annotation>
        <xs:documentation>This element defines the execution language that is used in order to execute the ExecutablePayload.</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="ExecutablePayload">
      <xs:annotation>
        <xs:documentation>This element encaptulates the executable payload.</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

element **HookType/HookAction/HookActionIdentifier**

diagram	
annotation	documentation This element represents an identifier that differentiates the Hook Actions.

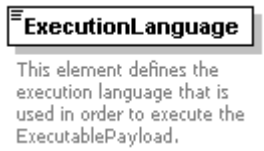
source	`<xs:element name="HookActionIdentifier">`
	`<xs:annotation>`
	`<xs:documentation>This element represents an identifier that differentiates the Hook Actions.</xs:documentation>`
	`</xs:annotation>`
	`</xs:element>`

element **HookType/HookAction/HookActionLabel**

diagram	
annotation	documentation This element provides a descriptive name for a Hook Action.

source	`<xs:element name="HookActionLabel">`
	`<xs:annotation>`
	`<xs:documentation>This element provides a descriptive name for a Hook Action.</xs:documentation>`
	`</xs:annotation>`
	`</xs:element>`

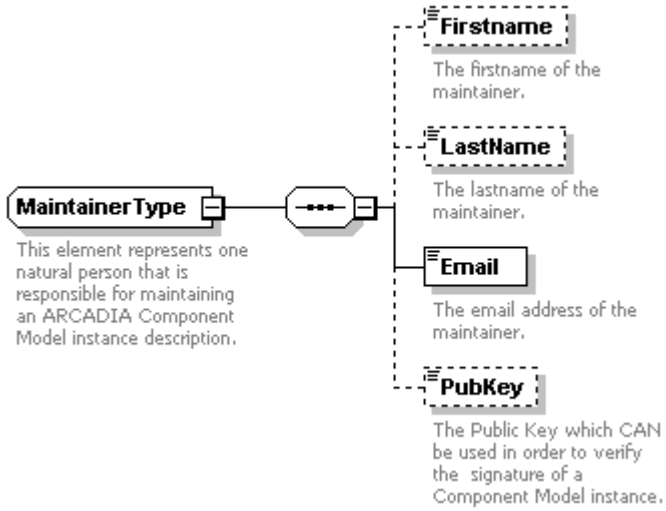
element **HookType/HookAction/ExecutionLanguage**

diagram	
type	ExecutionLanguageType
properties	content simple
facets	Kind Value Annotation enumeration BASH enumeration JAVA enumeration PYTHON enumeration RUBY
annotation	documentation This element defines the execution language that is used in order to execute the ExecutablePayload.
source	<xs:element name="ExecutionLanguage" type="ExecutionLanguageType"> <xs:annotation> <xs:documentation>This element defines the execution language that is used in order to execute the ExecutablePayload.</xs:documentation> </xs:annotation> </xs:element>

element **HookType/HookAction/ExecutablePayload**

diagram	
annotation	documentation This element encaptulates the executable payload.
source	<xs:element name="ExecutablePayload"> <xs:annotation> <xs:documentation>This element encaptulates the executable payload.</xs:documentation> </xs:annotation> </xs:element>

complexType **MaintainerType**

diagram	
children	Firstname LastName Email PubKey
used by	element ComponentMetadataType/Maintainer
annotation	documentation This element represents one natural person that is responsible for maintaining an ARCADIA Component Model instance description.

source

```
<xs:complexType name="MaintainerType">
 <xs:annotation>
  <xs:documentation>This element represents one natural person that is responsible for maintaining an ARCADIA Component Model instance description.</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element name="Firstname" type="xs:string" minOccurs="0">
   <xs:annotation>
    <xs:documentation>The firstname of the maintainer.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="LastName" type="xs:string" minOccurs="0">
   <xs:annotation>
    <xs:documentation>The lastname of the maintainer.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="Email" type="xs:string">
   <xs:annotation>
    <xs:documentation>The email address of the maintainer.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="PubKey" type="xs:string" minOccurs="0">
   <xs:annotation>
    <xs:documentation>The Public Key which CAN be used in order to verify the signature of a Component Model instance.</xs:documentation>
   </xs:annotation>
  </xs:element>
 </xs:sequence>
</xs:complexType>
```

element **MaintainerType/Firstname**

diagram	
type	**xs:string**
properties	minOcc 0 maxOcc 1 content simple
annotation	documentation The firstname of the maintainer.
source	<xs:element name="Firstname" type="xs:string" minOccurs="0"> <xs:annotation> <xs:documentation>The firstname of the maintainer.</xs:documentation> </xs:annotation> </xs:element>

element **MaintainerType/LastName**

diagram	
type	**xs:string**
properties	minOcc 0 maxOcc 1 content simple
annotation	documentation The lastname of the maintainer.
source	<xs:element name="LastName" type="xs:string" minOccurs="0"> <xs:annotation> <xs:documentation>The lastname of the maintainer.</xs:documentation> </xs:annotation> </xs:element>

element **MaintainerType/Email**

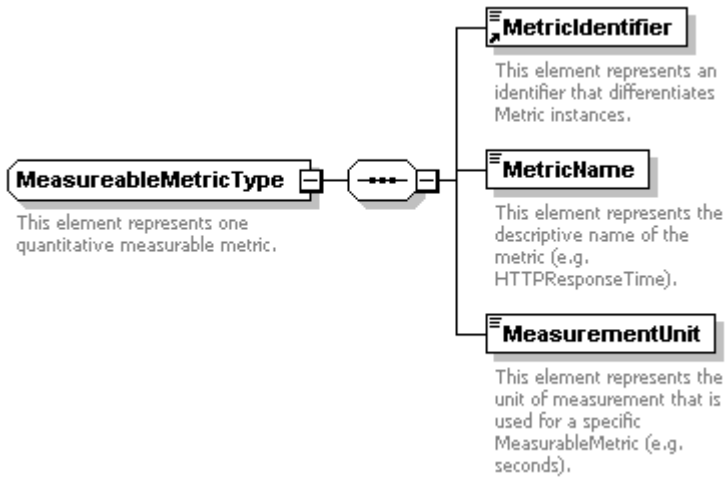
diagram	
type	**xs:string**
properties	content simple
annotation	documentation The email address of the maintainer.
source	<xs:element name="Email" type="xs:string">

	<xs:annotation> <xs:documentation>The email address of the maintainer.</xs:documentation> </xs:annotation> </xs:element>

element **MaintainerType/PubKey**

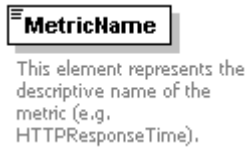
diagram	
type	**xs:string**
properties	minOcc 0 maxOcc 1 content simple
annotation	documentation The Public Key which CAN be used in order to verify the signature of a Component Model instance.
source	<xs:element name="PubKey" type="xs:string" minOccurs="0"> <xs:annotation> <xs:documentation>The Public Key which CAN be used in order to verify the signature of a Component Model instance.</xs:documentation> </xs:annotation> </xs:element>

complexType **MeasureableMetricType**

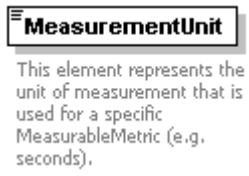
diagram	
children	MetricIdentifier MetricName MeasurementUnit
used by	elements MicroServiceType/ActionDescriptor/QoSActions/QoSAction/AffectedByMetric MicroServiceType/LinkActionDescriptor/QoSActions/QoSAction/AffectedByMetric ArcadiaServiceGraphModelType/GraphMonitoringDescriptor/GraphMeasurableMetric MicroServiceType/LinkMonitoringDescriptor/MeasurableMetric MicroServiceType/MonitoringDescriptor/MeasureableMetric
annotation	documentation

	This element represents one quantitative measurable metric.
source	<xs:complexType name="MeasureableMetricType"> <xs:annotation> <xs:documentation>This element represents one quantitative measurable metric.</xs:documentation> </xs:annotation> <xs:sequence> <xs:element ref="MetricIdentifier"> <xs:annotation> <xs:documentation>This element represents an identifier that differentiates Metric instances.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="MetricName"> <xs:annotation> <xs:documentation>This element represents the descriptive name of the metric (e.g. HTTPResponseTime).</xs:documentation> </xs:annotation> </xs:element> <xs:element name="MeasurementUnit" type="MeasurementUnit"> <xs:annotation> <xs:documentation>This element represents the unit of measurement that is used for a specific MeasurableMetric (e.g. seconds).</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType>

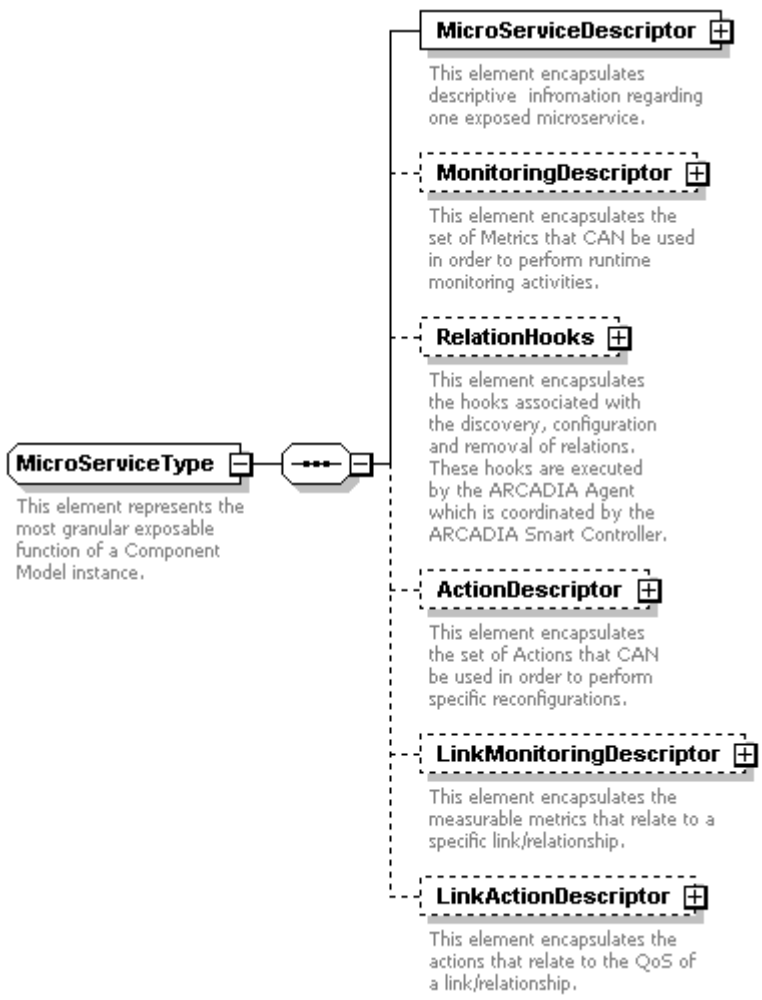
element **MeasureableMetricType/MetricName**

diagram	
annotation	documentation This element represents the descriptive name of the metric (e.g. HTTPResponseTime).
source	<xs:element name="MetricName"> <xs:annotation> <xs:documentation>This element represents the descriptive name of the metric (e.g. HTTPResponseTime).</xs:documentation> </xs:annotation> </xs:element>

element **MeasureableMetricType/MeasurementUnit**

diagram	
type	**MeasurementUnit**
properties	content simple
facets	Kind Value Annotation enumeration HTTP Requests / second enumeration ResponseTime (ms)
annotation	documentation This element represents the unit of measurement that is used for a specific MeasurableMetric (e.g. seconds).
source	<xs:element name="MeasurementUnit" type="MeasurementUnit"> <xs:annotation> <xs:documentation>This element represents the unit of measurement that is used for a specific MeasurableMetric (e.g. seconds).</xs:documentation> </xs:annotation> </xs:element>

complexType **MicroServiceType**

diagram	
children	MicroServiceDescriptor MonitoringDescriptor RelationHooks ActionDescriptor LinkMonitoringDescriptor LinkActionDescriptor
used by	elements ArcadiaComponentModelType/ExposedMicroServices/ExposedMicroService ArcadiaComponentModelType/RequiredMicroServices/RequiredMicroService
annotation	documentation This element represents the most granular exposable function of a Component Model instance.
source	<xs:complexType name="MicroServiceType"> <xs:annotation> <xs:documentation>This element represents the most granular exposable function of a Component Model instance.</xs:documentation> </xs:annotation> <xs:sequence> <xs:element name="MicroServiceDescriptor"> <xs:annotation> <xs:documentation>This element encapsulates descriptive infromation regarding one exposed microservice.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element ref="MicroServiceIdentifier"/> <xs:element name="MicroServiceName">

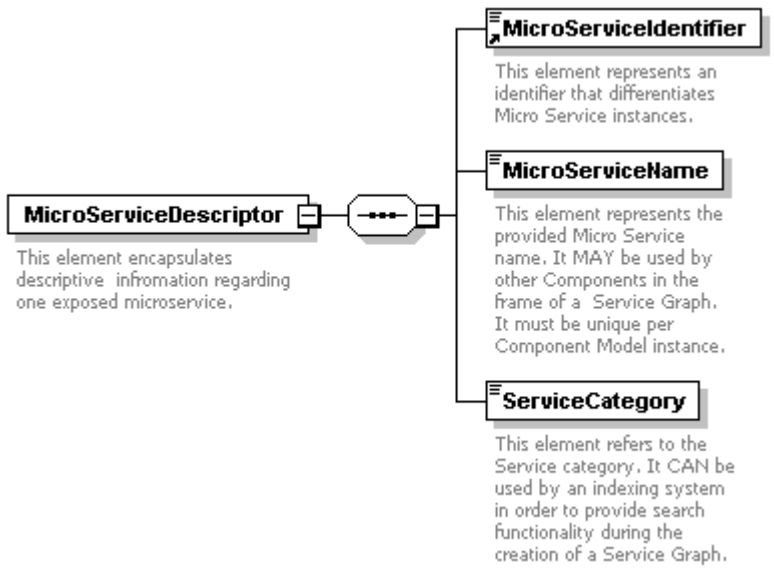
```
        <xs:annotation>
          <xs:documentation>This element represents the provided Micro Service name. It MAY be used by other Components in the frame of a  Service Graph. It must be unique per Component Model instance.</xs:documentation>
        </xs:annotation>
       </xs:element>
       <xs:element name="ServiceCategory">
        <xs:annotation>
          <xs:documentation>This element refers to the Service category. It CAN be used by an indexing system in order to provide search functionality during the creation of a Service Graph.</xs:documentation>
        </xs:annotation>
       </xs:element>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
    <xs:element name="MonitoringDescriptor" minOccurs="0">
     <xs:annotation>
      <xs:documentation>This element encapsulates the set of Metrics that CAN be used in order to perform runtime monitoring activities.</xs:documentation>
     </xs:annotation>
     <xs:complexType>
      <xs:sequence>
       <xs:element name="MeasureableMetric" type="MeasureableMetricType" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation>This element refers to a specific quantitative monitoring metric which is a performance indicator for the specific MicroService.</xs:documentation>
        </xs:annotation>
       </xs:element>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
    <xs:element name="RelationHooks" minOccurs="0">
     <xs:annotation>
      <xs:documentation>This element encapsulates the hooks associated with the discovery, configuration and removal of relations.  These hooks are executed by the ARCADIA Agent which is coordinated by the ARCADIA Smart Controller.</xs:documentation>
     </xs:annotation>
     <xs:complexType>
      <xs:sequence>
       <xs:element name="RelationJoinedHook" type="HookType">
        <xs:annotation>
          <xs:documentation>This element encapsulates the set of actions that has to be performed when a new relationship is discovered and established.</xs:documentation>
        </xs:annotation>
       </xs:element>
       <xs:element name="RelationChangedHook" type="HookType">
        <xs:annotation>
          <xs:documentation>This element encapsulates the set of actions that has to be performed when an established relationship is affected by a change.</xs:documentation>
        </xs:annotation>
       </xs:element>
       <xs:element name="RelationDepartedHook" type="HookType">
        <xs:annotation>
          <xs:documentation>This element encapsulates the set of actions that has to be
```

```
performed when an established relationship is going to be gracefully terminated.</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="RelationBrokenHook" type="HookType">
          <xs:annotation>
            <xs:documentation>This element encapsulates the set of actions that has to be performed when an established relationship is going to be abnormally terminated.</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="ActionDescriptor" minOccurs="0">
    <xs:annotation>
      <xs:documentation>This element encapsulates the set of Actions that CAN be used in order to perform specific reconfigurations.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element name="QoSActions" minOccurs="0">
          <xs:annotation>
            <xs:documentation>This element represents the set of actions that are binded to specific Measurable Metrics.</xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:sequence>
              <xs:element name="QoSAction" maxOccurs="unbounded">
                <xs:annotation>
                  <xs:documentation>This element represents a specific QoS action.</xs:documentation>
                </xs:annotation>
                <xs:complexType>
                  <xs:complexContent>
                    <xs:extension base="ActionType">
                      <xs:sequence>
                        <xs:element name="AffectedByMetric">
                          <xs:annotation>
                            <xs:documentation>This element defines the Measurable Metric that will provide the Triggering Event for this QoS Action. Please bear in mind that the threshold is not defined in this model; yet it SHALL be provided in the frame of the ARCADIA Policy Model.</xs:documentation>
                          </xs:annotation>
                          <xs:complexType>
                            <xs:complexContent>
                              <xs:extension base="MeasureableMetricType"/>
                            </xs:complexContent>
                          </xs:complexType>
                        </xs:element>
                      </xs:sequence>
                    </xs:extension>
                  </xs:complexContent>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
```

```
        </xs:complexType>
      </xs:element>
      <xs:element name="CustomActions" minOccurs="0">
        <xs:annotation>
          <xs:documentation>This element encapsulates a set of custom actions that can be invoked by the ARCADIA Smart Controller.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Action" type="ActionType" maxOccurs="unbounded">
              <xs:annotation>
                <xs:documentation>This element represents one atomic action that can be invoked.</xs:documentation>
              </xs:annotation>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="LinkMonitoringDescriptor" minOccurs="0">
  <xs:annotation>
   <xs:documentation>This element encapsulates the measurable metrics that relate to a specific link/relationship.</xs:documentation>
  </xs:annotation>
  <xs:complexType>
   <xs:sequence>
    <xs:element name="MeasurableMetric" type="MeasureableMetricType" maxOccurs="unbounded">
     <xs:annotation>
      <xs:documentation>This element refers to a specific quantitative monitoring metric which is a performance indicator for a specific relationship/link.</xs:documentation>
     </xs:annotation>
    </xs:element>
   </xs:sequence>
  </xs:complexType>
 </xs:element>
 <xs:element name="LinkActionDescriptor" minOccurs="0">
  <xs:annotation>
   <xs:documentation>This element encapsulates the actions that relate to the QoS of a link/relationship.</xs:documentation>
  </xs:annotation>
  <xs:complexType>
   <xs:sequence>
    <xs:element name="QoSActions" minOccurs="0">
     <xs:annotation>
      <xs:documentation>This element represents the set of actions that are binded to specific Measurable Metrics of the link/relationship.</xs:documentation>
     </xs:annotation>
     <xs:complexType>
      <xs:sequence>
       <xs:element name="QoSAction" maxOccurs="unbounded">
        <xs:annotation>
         <xs:documentation>This element represents one atomic action that can be invoked.</xs:documentation>
```

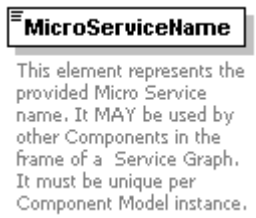
```
            </xs:annotation>
            <xs:complexType>
              <xs:complexContent>
                <xs:extension                                   base="ActionType">
                  <xs:sequence>
                    <xs:element                                 name="AffectedByMetric">
                      <xs:annotation>
                        <xs:documentation>Will define the Triggering Event when
implemented</xs:documentation>
                      </xs:annotation>
                      <xs:complexType>
                        <xs:complexContent>
                          <xs:extension                         base="MeasureableMetricType"/>
                        </xs:complexContent>
                      </xs:complexType>
                    </xs:element>
                  </xs:sequence>
                </xs:extension>
              </xs:complexContent>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

element **MicroServiceType/MicroServiceDescriptor**

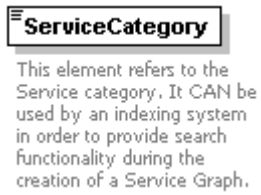
diagram	
properties	content complex
children	MicroServiceIdentifier MicroServiceName ServiceCategory

annotation	documentation This element encapsulates descriptive infromation regarding one exposed microservice.
source	<xs:element name="MicroServiceDescriptor"> <xs:annotation> <xs:documentation>This element encapsulates descriptive infromation regarding one exposed microservice.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element ref="MicroServiceIdentifier"/> <xs:element name="MicroServiceName"> <xs:annotation> <xs:documentation>This element represents the provided Micro Service name. It MAY be used by other Components in the frame of a Service Graph. It must be unique per Component Model instance.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="ServiceCategory"> <xs:annotation> <xs:documentation>This element refers to the Service category. It CAN be used by an indexing system in order to provide search functionality during the creation of a Service Graph.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType> </xs:element>

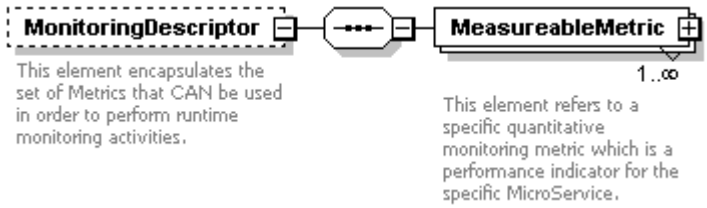
element **MicroServiceType/MicroServiceDescriptor/MicroServiceName**

diagram	
annotation	documentation This element represents the provided Micro Service name. It MAY be used by other Components in the frame of a Service Graph. It must be unique per Component Model instance.
source	<xs:element name="MicroServiceName"> <xs:annotation> <xs:documentation>This element represents the provided Micro Service name. It MAY be used by other Components in the frame of a Service Graph. It must be unique per Component Model instance.</xs:documentation> </xs:annotation> </xs:element>

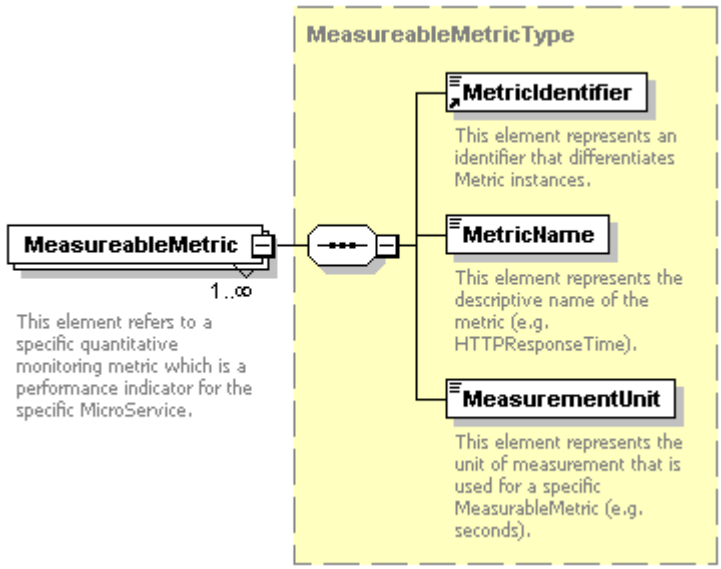
element **MicroServiceType/MicroServiceDescriptor/ServiceCategory**

diagram	
annotation	documentation This element refers to the Service category. It CAN be used by an indexing system in order to provide search functionality during the creation of a Service Graph.
source	<xs:element name="ServiceCategory"> <xs:annotation> <xs:documentation>This element refers to the Service category. It CAN be used by an indexing system in order to provide search functionality during the creation of a Service Graph.</xs:documentation> </xs:annotation> </xs:element>

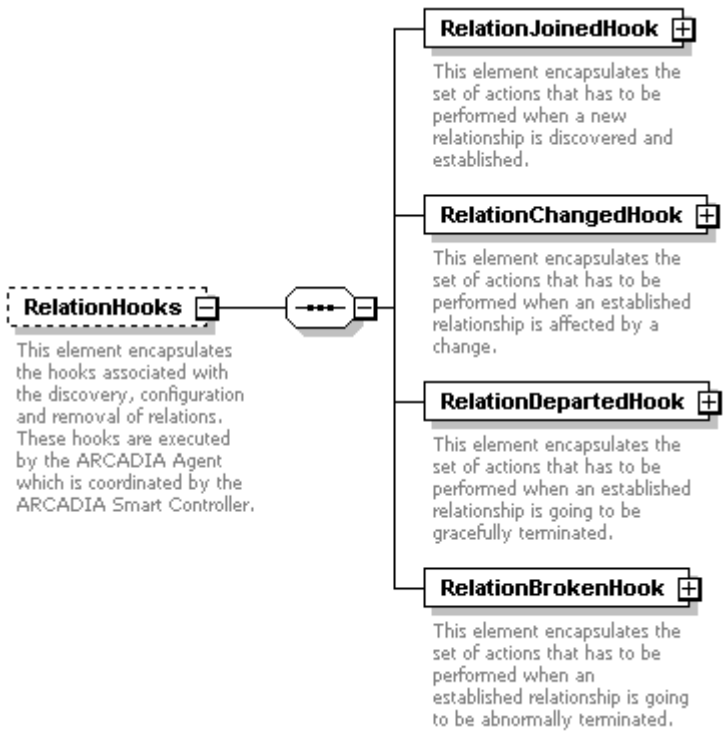
element **MicroServiceType/MonitoringDescriptor**

diagram	
properties	minOcc 0 maxOcc 1 content complex
children	**MeasureableMetric**
annotation	documentation This element encapsulates the set of Metrics that CAN be used in order to perform runtime monitoring activities.
source	<xs:element name="MonitoringDescriptor" minOccurs="0"> <xs:annotation> <xs:documentation>This element encapsulates the set of Metrics that CAN be used in order to perform runtime monitoring activities.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="MeasureableMetric" type="MeasureableMetricType" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element refers to a specific quantitative monitoring metric which is a performance indicator for the specific MicroService.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType> </xs:element>

element **MicroServiceType/MonitoringDescriptor/MeasureableMetric**

diagram	
type	MeasureableMetricType
properties	minOcc 1 maxOcc unbounded content complex
children	MetricIdentifier MetricName MeasurementUnit
annotation	documentation This element refers to a specific quantitative monitoring metric which is a performance indicator for the specific MicroService.
source	<xs:element name="MeasureableMetric" type="MeasureableMetricType" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element refers to a specific quantitative monitoring metric which is a performance indicator for the specific MicroService.</xs:documentation> </xs:annotation> </xs:element>

element **MicroServiceType/RelationHooks**

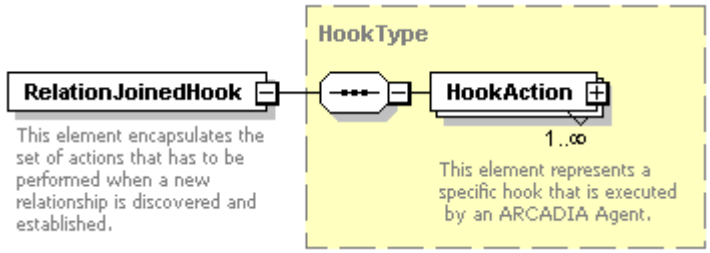
diagram	
properties	minOcc 0 maxOcc 1 content complex
children	RelationJoinedHook RelationChangedHook RelationDepartedHook RelationBrokenHook
annotation	documentation This element encapsulates the hooks associated with the discovery, configuration and removal of relations. These hooks are executed by the ARCADIA Agent which is coordinated by the ARCADIA Smart Controller.

source

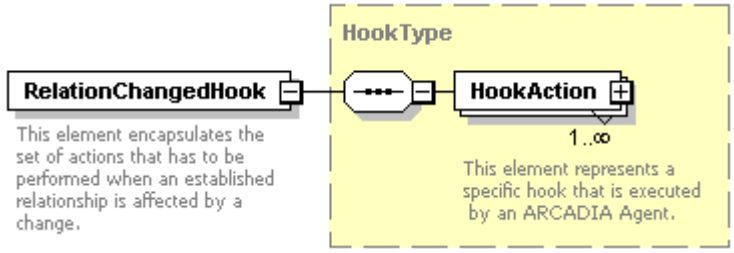
```
<xs:element name="RelationHooks" minOccurs="0">
 <xs:annotation>
  <xs:documentation>This element encapsulates the hooks associated with the discovery, configuration and removal of relations. These hooks are executed by the ARCADIA Agent which is coordinated by the ARCADIA Smart Controller.</xs:documentation>
 </xs:annotation>
 <xs:complexType>
  <xs:sequence>
   <xs:element name="RelationJoinedHook" type="HookType">
    <xs:annotation>
     <xs:documentation>This element encapsulates the set of actions that has to be performed when a new relationship is discovered and established.</xs:documentation>
    </xs:annotation>
   </xs:element>
   <xs:element name="RelationChangedHook" type="HookType">
    <xs:annotation>
     <xs:documentation>This element encapsulates the set of actions that has to be performed when an established relationship is affected by a change.</xs:documentation>
    </xs:annotation>
   </xs:element>
   <xs:element name="RelationDepartedHook" type="HookType">
    <xs:annotation>
```

	<xs:documentation>This element encapsulates the set of actions that has to be performed when an established relationship is going to be gracefully terminated.</xs:documentation> </xs:annotation> </xs:element> <xs:element name="RelationBrokenHook" type="HookType"> <xs:annotation> <xs:documentation>This element encapsulates the set of actions that has to be performed when an established relationship is going to be abnormally terminated.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType> </xs:element>

element **MicroServiceType/RelationHooks/RelationJoinedHook**

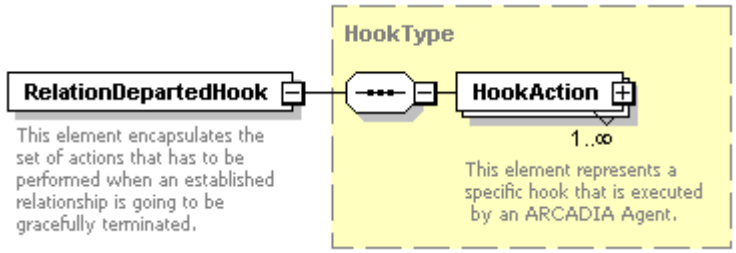
diagram	
type	**HookType**
properties	content complex
children	**HookAction**
annotation	documentation This element encapsulates the set of actions that has to be performed when a new relationship is discovered and established.
source	<xs:element name="RelationJoinedHook" type="HookType"> <xs:annotation> <xs:documentation>This element encapsulates the set of actions that has to be performed when a new relationship is discovered and established.</xs:documentation> </xs:annotation> </xs:element>

element **MicroServiceType/RelationHooks/RelationChangedHook**

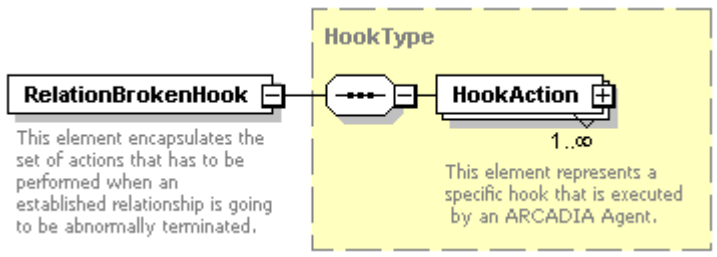
diagram	
type	**HookType**

properties	content complex
children	**HookAction**
annotation	documentation This element encapsulates the set of actions that has to be performed when an established relationship is affected by a change.
source	<xs:element name="RelationChangedHook" type="HookType"> <xs:annotation> <xs:documentation>This element encapsulates the set of actions that has to be performed when an established relationship is affected by a change.</xs:documentation> </xs:annotation> </xs:element>

element **MicroServiceType/RelationHooks/RelationDepartedHook**

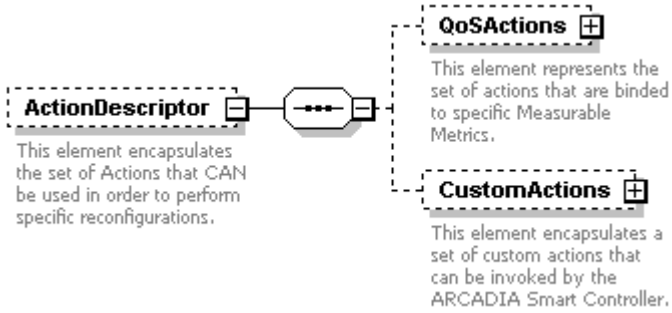
diagram	
type	**HookType**
properties	content complex
children	**HookAction**
annotation	documentation This element encapsulates the set of actions that has to be performed when an established relationship is going to be gracefully terminated.
source	<xs:element name="RelationDepartedHook" type="HookType"> <xs:annotation> <xs:documentation>This element encapsulates the set of actions that has to be performed when an established relationship is going to be gracefully terminated.</xs:documentation> </xs:annotation> </xs:element>

element **MicroServiceType/RelationHooks/RelationBrokenHook**

diagram	
type	**HookType**
properties	content complex

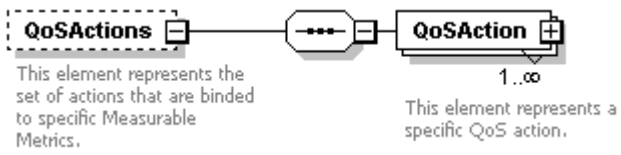
children	**HookAction**
annotation	documentation This element encapsulates the set of actions that has to be performed when an established relationship is going to be abnormally terminated.
source	<xs:element name="RelationBrokenHook" type="HookType"> <xs:annotation> <xs:documentation>This element encapsulates the set of actions that has to be performed when an established relationship is going to be abnormally terminated.</xs:documentation> </xs:annotation> </xs:element>

element **MicroServiceType/ActionDescriptor**

diagram	
properties	minOcc 0 maxOcc 1 content complex
children	**QoSActions** **CustomActions**
annotation	documentation This element encapsulates the set of Actions that CAN be used in order to perform specific reconfigurations.
source	<xs:element name="ActionDescriptor" minOccurs="0"> <xs:annotation> <xs:documentation>This element encapsulates the set of Actions that CAN be used in order to perform specific reconfigurations.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="QoSActions" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the set of actions that are binded to specific Measurable Metrics.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="QoSAction" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents a specific QoS action.</xs:documentation> </xs:annotation> <xs:complexType> <xs:complexContent> <xs:extension base="ActionType">

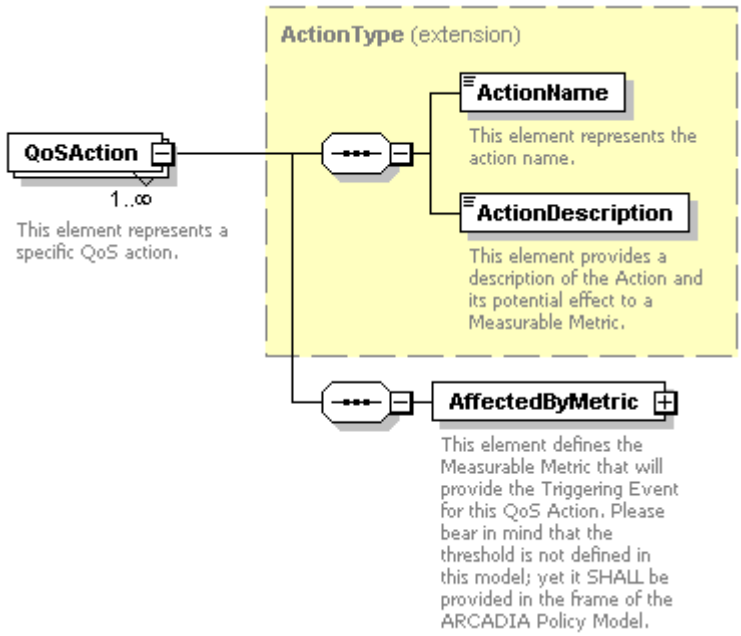
```
                <xs:sequence>
                  <xs:element                                    name="AffectedByMetric">
                    <xs:annotation>
                      <xs:documentation>This element defines the Measurable Metric that will provide the Triggering Event for this QoS Action. Please bear in mind that the threshold is not defined in this model; yet it SHALL be provided in the frame of the ARCADIA Policy Model.</xs:documentation>
                    </xs:annotation>
                    <xs:complexType>
                      <xs:complexContent>
                        <xs:extension                                base="MeasureableMetricType"/>
                      </xs:complexContent>
                    </xs:complexType>
                  </xs:element>
                </xs:sequence>
              </xs:extension>
            </xs:complexContent>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element                    name="CustomActions"                    minOccurs="0">
    <xs:annotation>
      <xs:documentation>This element encapsulates a set of custom actions that can be invoked by the ARCADIA Smart Controller.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element      name="Action"      type="ActionType"      maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>This element represents one atomic action that can be invoked.</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
 </xs:sequence>
</xs:complexType>
</xs:element>
```

element **MicroServiceType/ActionDescriptor/QoSActions**

diagram	
properties	minOcc 0 maxOcc 1 content complex
children	QoSAction

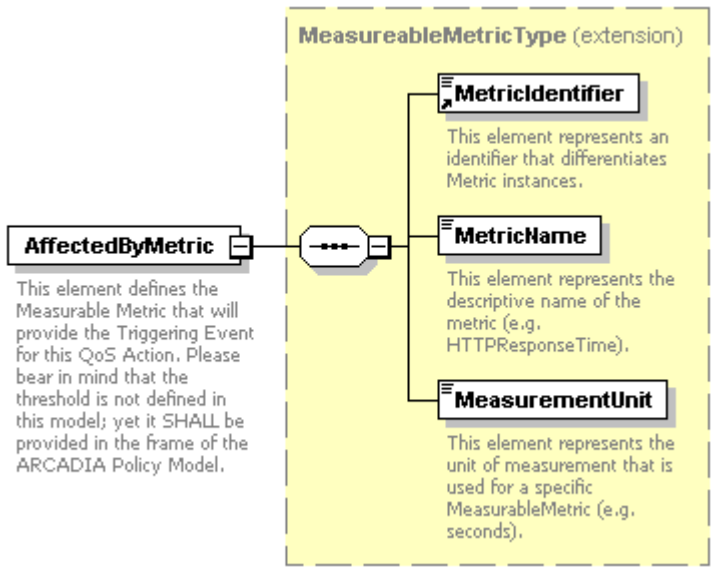
annotation	documentation This element represents the set of actions that are binded to specific Measurable Metrics.
source	<xs:element name="QoSActions" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the set of actions that are binded to specific Measurable Metrics.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="QoSAction" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents a specific QoS action.</xs:documentation> </xs:annotation> <xs:complexType> <xs:complexContent> <xs:extension base="ActionType"> <xs:sequence> <xs:element name="AffectedByMetric"> <xs:annotation> <xs:documentation>This element defines the Measurable Metric that will provide the Triggering Event for this QoS Action. Please bear in mind that the threshold is not defined in this model; yet it SHALL be provided in the frame of the ARCADIA Policy Model.</xs:documentation> </xs:annotation> <xs:complexType> <xs:complexContent> <xs:extension base="MeasureableMetricType"/> </xs:complexContent> </xs:complexType> </xs:element> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </xs:element> </xs:sequence> </xs:complexType> </xs:element>

element **MicroServiceType/ActionDescriptor/QoSActions/QoSAction**

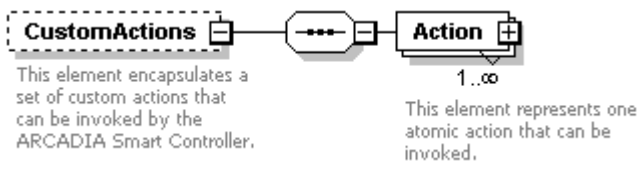
diagram	
type	extension of **ActionType**
properties	minOcc 1 maxOcc unbounded content complex
children	**ActionName** **ActionDescription** **AffectedByMetric**
annotation	documentation This element represents a specific QoS action.
source	<xs:element name="QoSAction" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents a specific QoS action.</xs:documentation> </xs:annotation> <xs:complexType> <xs:complexContent> <xs:extension base="ActionType"> <xs:sequence> <xs:element name="AffectedByMetric"> <xs:annotation> <xs:documentation>This element defines the Measurable Metric that will provide the Triggering Event for this QoS Action. Please bear in mind that the threshold is not defined in this model; yet it SHALL be provided in the frame of the ARCADIA Policy Model.</xs:documentation> </xs:annotation> <xs:complexType> <xs:complexContent> <xs:extension base="MeasureableMetricType"/> </xs:complexContent> </xs:complexType> </xs:element> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType>

	</xs:element>

element **MicroServiceType/ActionDescriptor/QoSActions/QoSAction/AffectedByMetric**

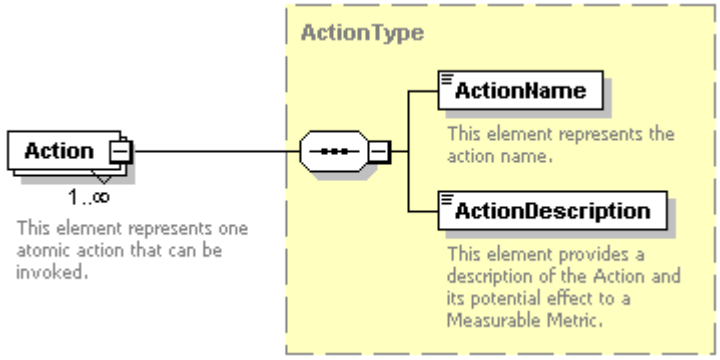
diagram	
type	extension of **MeasureableMetricType**
properties	content complex
children	**MetricIdentifier** **MetricName** **MeasurementUnit**
annotation	documentation This element defines the Measurable Metric that will provide the Triggering Event for this QoS Action. Please bear in mind that the threshold is not defined in this model; yet it SHALL be provided in the frame of the ARCADIA Policy Model.
source	<xs:element name="AffectedByMetric"> <xs:annotation> <xs:documentation>This element defines the Measurable Metric that will provide the Triggering Event for this QoS Action. Please bear in mind that the threshold is not defined in this model; yet it SHALL be provided in the frame of the ARCADIA Policy Model.</xs:documentation> </xs:annotation> <xs:complexType> <xs:complexContent> <xs:extension base="MeasureableMetricType"/> </xs:complexContent> </xs:complexType> </xs:element>

element **MicroServiceType/ActionDescriptor/CustomActions**

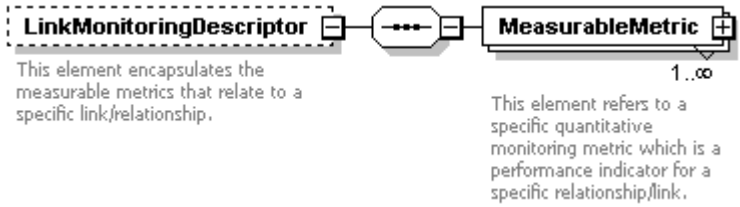
diagram	

properties	minOcc 0 maxOcc 1 content complex
children	**Action**
annotation	documentation This element encapsulates a set of custom actions that can be invoked by the ARCADIA Smart Controller.
source	<xs:element name="CustomActions" minOccurs="0"> <xs:annotation> <xs:documentation>This element encapsulates a set of custom actions that can be invoked by the ARCADIA Smart Controller.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="Action" type="ActionType" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents one atomic action that can be invoked.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType> </xs:element>

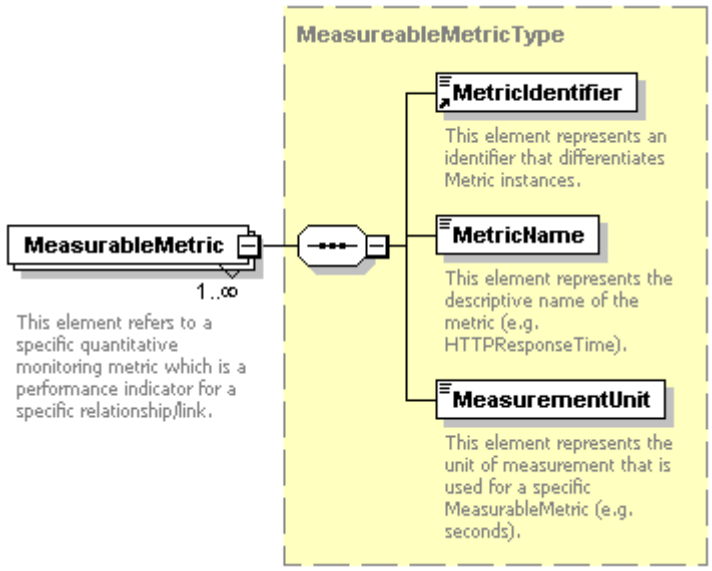
element **MicroServiceType/ActionDescriptor/CustomActions/Action**

diagram	
type	**ActionType**
properties	minOcc 1 maxOcc unbounded content complex
children	**ActionName** **ActionDescription**
annotation	documentation This element represents one atomic action that can be invoked.
source	<xs:element name="Action" type="ActionType" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents one atomic action that can be invoked.</xs:documentation> </xs:annotation> </xs:element>

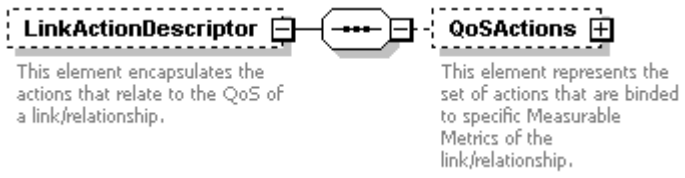
element **MicroServiceType/LinkMonitoringDescriptor**

diagram	
properties	minOcc 0 maxOcc 1 content complex
children	**MeasurableMetric**
annotation	documentation This element encapsulates the measurable metrics that relate to a specific link/relationship.
source	<xs:element name="LinkMonitoringDescriptor" minOccurs="0"> <xs:annotation> <xs:documentation>This element encapsulates the measurable metrics that relate to a specific link/relationship.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="MeasurableMetric" type="MeasureableMetricType" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element refers to a specific quantitative monitoring metric which is a performance indicator for a specific relationship/link.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> </xs:complexType> </xs:element>

element **MicroServiceType/LinkMonitoringDescriptor/MeasurableMetric**

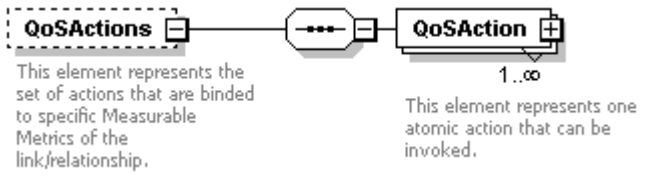
diagram	
type	MeasureableMetricType
properties	minOcc 1 maxOcc unbounded content complex
children	MetricIdentifier MetricName MeasurementUnit
annotation	documentation This element refers to a specific quantitative monitoring metric which is a performance indicator for a specific relationship/link.
source	<xs:element name="MeasurableMetric" type="MeasureableMetricType" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element refers to a specific quantitative monitoring metric which is a performance indicator for a specific relationship/link.</xs:documentation> </xs:annotation> </xs:element>

element **MicroServiceType/LinkActionDescriptor**

diagram	
properties	minOcc 0 maxOcc 1 content complex
children	QoSActions
annotation	documentation This element encapsulates the actions that relate to the QoS of a link/relationship.
source	<xs:element name="LinkActionDescriptor" minOccurs="0">

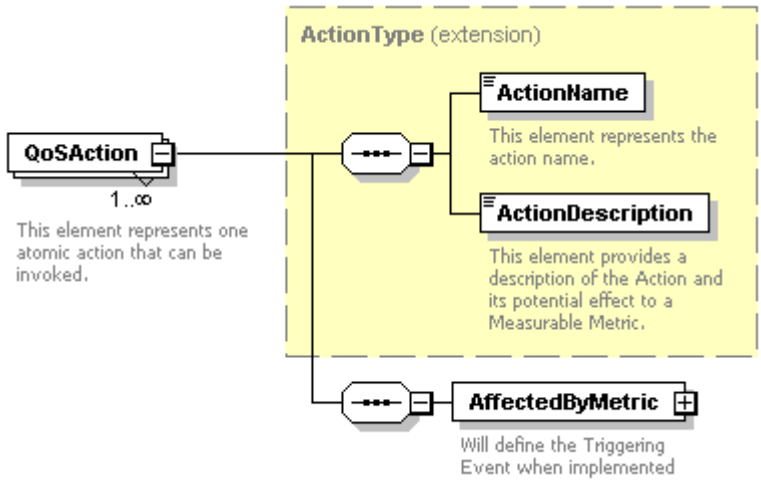
```
<xs:annotation>
  <xs:documentation>This element encapsulates the actions that relate to the QoS of a link/relationship.</xs:documentation>
 </xs:annotation>
 <xs:complexType>
  <xs:sequence>
   <xs:element name="QoSActions" minOccurs="0">
    <xs:annotation>
     <xs:documentation>This element represents the set of actions that are binded to specific Measurable Metrics of the link/relationship.</xs:documentation>
    </xs:annotation>
    <xs:complexType>
     <xs:sequence>
      <xs:element name="QoSAction" maxOccurs="unbounded">
       <xs:annotation>
        <xs:documentation>This element represents one atomic action that can be invoked.</xs:documentation>
       </xs:annotation>
       <xs:complexType>
        <xs:complexContent>
         <xs:extension base="ActionType">
          <xs:sequence>
           <xs:element name="AffectedByMetric">
            <xs:annotation>
             <xs:documentation>Will define the Triggering Event when implemented</xs:documentation>
            </xs:annotation>
            <xs:complexType>
             <xs:complexContent>
              <xs:extension base="MeasureableMetricType"/>
             </xs:complexContent>
            </xs:complexType>
           </xs:element>
          </xs:sequence>
         </xs:extension>
        </xs:complexContent>
       </xs:complexType>
      </xs:element>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

element **MicroServiceType/LinkActionDescriptor/QoSActions**

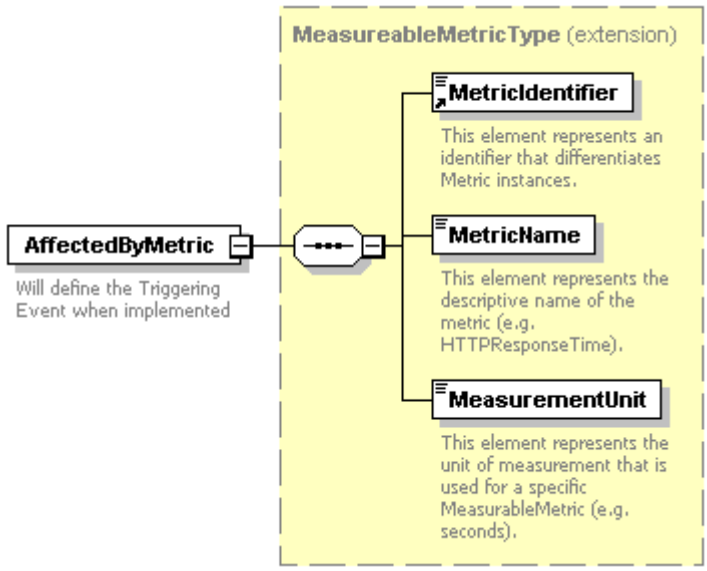
diagram	
properties	minOcc 0

	maxOcc 1 content complex
children	**QoSAction**
annotation	documentation This element represents the set of actions that are binded to specific Measurable Metrics of the link/relationship.
source	<xs:element name="QoSActions" minOccurs="0"> <xs:annotation> <xs:documentation>This element represents the set of actions that are binded to specific Measurable Metrics of the link/relationship.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:element name="QoSAction" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents one atomic action that can be invoked.</xs:documentation> </xs:annotation> <xs:complexType> <xs:complexContent> <xs:extension base="ActionType"> <xs:sequence> <xs:element name="AffectedByMetric"> <xs:annotation> <xs:documentation>Will define the Triggering Event when implemented</xs:documentation> </xs:annotation> <xs:complexType> <xs:complexContent> <xs:extension base="MeasureableMetricType"/> </xs:complexContent> </xs:complexType> </xs:element> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </xs:element> </xs:sequence> </xs:complexType> </xs:element>

element **MicroServiceType/LinkActionDescriptor/QoSActions/QoSAction**

diagram	
type	extension of **ActionType**
properties	minOcc 1 maxOcc unbounded content complex
children	**ActionName** **ActionDescription** **AffectedByMetric**
annotation	documentation This element represents one atomic action that can be invoked.
source	<xs:element name="QoSAction" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>This element represents one atomic action that can be invoked.</xs:documentation> </xs:annotation> <xs:complexType> <xs:complexContent> <xs:extension base="ActionType"> <xs:sequence> <xs:element name="AffectedByMetric"> <xs:annotation> <xs:documentation>Will define the Triggering Event when implemented</xs:documentation> </xs:annotation> <xs:complexType> <xs:complexContent> <xs:extension base="MeasureableMetricType"/> </xs:complexContent> </xs:complexType> </xs:element> </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </xs:element>

element **MicroServiceType/LinkActionDescriptor/QoSActions/QoSAction/AffectedByMetric**

diagram	
type	extension of **MeasureableMetricType**
properties	content complex
children	**MetricIdentifier** **MetricName** **MeasurementUnit**
annotation	documentation Will define the Triggering Event when implemented
source	<xs:element name="AffectedByMetric"> <xs:annotation> <xs:documentation>Will define the Triggering Event when implemented</xs:documentation> </xs:annotation> <xs:complexType> <xs:complexContent> <xs:extension base="MeasureableMetricType"/> </xs:complexContent> </xs:complexType> </xs:element>

simpleType **ConfigurationValueType**

type	restriction of **xs:string**		
properties	base xs:string		
used by	element **ComponentConfigurationType/ConfigurationElement/ValueType**		
facets	Kind	Value	Annotation
	enumeration	INTEGER	
	enumeration	STRING	
	enumeration	BOOLEAN	
	enumeration	SINGLE_SELECTION_ENUMERATION	
	enumeration	MULTI_SELECTION_ENUMERATION	
annotation	documentation		

	This element represents the enumeration of the supported configuration value types.
source	<xs:simpleType name="ConfigurationValueType"> <xs:annotation> <xs:documentation>This element represents the enumeration of the supported configuration value types.</xs:documentation> </xs:annotation> <xs:restriction base="xs:string"> <xs:enumeration value="INTEGER"/> <xs:enumeration value="STRING"/> <xs:enumeration value="BOOLEAN"/> <xs:enumeration value="SINGLE_SELECTION_ENUMERATION"/> <xs:enumeration value="MULTI_SELECTION_ENUMERATION"/> </xs:restriction> </xs:simpleType>

simpleType **ExecutionLanguageType**

type	restriction of **xs:string**
properties	base xs:string
used by	element **HookType/HookAction/ExecutionLanguage**
facets	Kind Value Annotation enumeration BASH enumeration JAVA enumeration PYTHON enumeration RUBY
annotation	documentation This element represents the enumeration of the supported execution languages.
source	<xs:simpleType name="ExecutionLanguageType"> <xs:annotation> <xs:documentation>This element represents the enumeration of the supported execution languages.</xs:documentation> </xs:annotation> <xs:restriction base="xs:string"> <xs:enumeration value="BASH"/> <xs:enumeration value="JAVA"/> <xs:enumeration value="PYTHON"/> <xs:enumeration value="RUBY"/> </xs:restriction> </xs:simpleType>

simpleType **MeasurementUnit**

type	restriction of **xs:string**
properties	base xs:string
used by	element **MeasureableMetricType/MeasurementUnit**
facets	Kind Value Annotation enumeration HTTP Requests / second

	enumeration ResponseTime (ms)
annotation	documentation This element represents the measurable unit of one quantitative metric.
source	<xs:simpleType name="MeasurementUnit"> <xs:annotation> <xs:documentation>This element represents the measurable unit of one quantitative metric.</xs:documentation> </xs:annotation> <xs:restriction base="xs:string"> <xs:enumeration value="HTTP Requests / second"/> <xs:enumeration value="ResponseTime (ms)"/> </xs:restriction> </xs:simpleType>

simpleType **ProcessorArchitectureType**

type	restriction of **xs:string**
properties	base xs:string
used by	element **ArcadiaComponentModelType/Requirements/HostingRequirements/OperatingSystem/ProcessorArchitecture**
facets	Kind Value Annotation enumeration amd64 enumeration i386
annotation	documentation This element represents the enumeration of the supported processor architectures.
source	<xs:simpleType name="ProcessorArchitectureType"> <xs:annotation> <xs:documentation>This element represents the enumeration of the supported processor architectures.</xs:documentation> </xs:annotation> <xs:restriction base="xs:string"> <xs:enumeration value="amd64"/> <xs:enumeration value="i386"/> </xs:restriction> </xs:simpleType>

simpleType **ServiceCategoryType**

type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value Annotation pattern Database
annotation	documentation This element represents the enumeration of the predefined MicroService categories.
source	<xs:simpleType name="ServiceCategoryType"> <xs:annotation> <xs:documentation>This element represents the enumeration of the predefined MicroService

	categories.</xs:documentation> </xs:annotation> <xs:restriction base="xs:string"> <xs:pattern value="Database"/> </xs:restriction> </xs:simpleType>