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

Disclaimer: The ARCADIA project is co-funded by the European Commission under the Horizon 2020 Programme. This document reflects only authors' views. EC is not liable for any use that may be done of the information contained therein.
## ARCADIA Profile

<table>
<thead>
<tr>
<th>Grant Agreement No.:</th>
<th>645372</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Acronym:</strong></td>
<td>ARCADIA</td>
</tr>
<tr>
<td><strong>Title:</strong> A NOVEL RECONFIGURABLE BY DESIGN HIGHLY DISTRIBUTED APPLICATIONS DEVELOPMENT PARADIGM OVER PROGRAMMABLE INFRASTRUCTURE</td>
<td></td>
</tr>
<tr>
<td><strong>URL:</strong></td>
<td><a href="http://www.arcadia-framework.eu/">http://www.arcadia-framework.eu/</a></td>
</tr>
<tr>
<td><strong>Start Date:</strong></td>
<td>01/01/2015</td>
</tr>
<tr>
<td><strong>Duration:</strong></td>
<td>36 months</td>
</tr>
</tbody>
</table>

### Partners

<table>
<thead>
<tr>
<th>Logo</th>
<th>Name</th>
<th>Country</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="NUI Galway" /></td>
<td>Insight Centre for Data Analytics, National University of Ireland, Galway</td>
<td>Ireland</td>
</tr>
<tr>
<td><img src="image" alt="SINTEF" /></td>
<td>Stiftelsen SINTEF</td>
<td>Norway</td>
</tr>
<tr>
<td><img src="image" alt="Technische Universität Berlin" /></td>
<td>Technische Universität Berlin</td>
<td>Germany</td>
</tr>
<tr>
<td><img src="image" alt="Consorzio Nazionale Interuniversitario per le Telecomunicazioni" /></td>
<td>Consorzio Nazionale Interuniversitario per le Telecomunicazioni</td>
<td>Italy</td>
</tr>
<tr>
<td><img src="image" alt="Univerza v Ljubljani" /></td>
<td>Univerza v Ljubljani</td>
<td>Slovenia</td>
</tr>
<tr>
<td><img src="image" alt="UBITECH" /></td>
<td>UBITECH</td>
<td>Greece</td>
</tr>
<tr>
<td><img src="image" alt="WINGS ICT Solutions Information &amp; Communication Technologies EPE" /></td>
<td>WINGS ICT Solutions Information &amp; Communication Technologies EPE</td>
<td>Greece</td>
</tr>
<tr>
<td><img src="image" alt="MAGGIOLI SPA" /></td>
<td>MAGGIOLI SPA</td>
<td>Italy</td>
</tr>
<tr>
<td><img src="image" alt="ADITESS Advanced Integrated Technology Solutions and Services Ltd" /></td>
<td>ADITESS Advanced Integrated Technology Solutions and Services Ltd</td>
<td>Cyprus</td>
</tr>
</tbody>
</table>
## Document History

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

1 Introduction .......................................................................................................................... 8
   1.1 Scope of the Deliverable ................................................................................................. 8
   1.2 Structure of the Deliverable ........................................................................................... 9

2 The role of ARCADIA Context Model in the ARCADIA Operational Environment ......................................................................................................................... 11
   2.1 The Facets of the ARCADIA Context Model and their usage in HDA lifecycle .......... 11
   2.2 Bird’s eye view on ARCADIA Architectural Components that use the Context Model ................................................................................................................................. 12
   2.3 Related Work that affected the formulation of ARCADIA Context Model & Differentiation ......................................................................................................................... 14
      2.3.1 Compatibility of ARCADIA context model with Canonical Juju meta-model.......... 14
      2.3.2 Compatibility of ARCADIA context model with TOSCA NFV specification ......... 15
      2.3.3 Correlation with models that derived from research projects ............................... 16

3 Overview of the ARCADIA Component Model ................................................................... 18
   3.1 Overview of the ARCADIA Component Model ............................................................. 18
   3.2 Elaboration on the ARCADIA Component Model ....................................................... 19

4 Overview of the ARCADIA Service Graph Model ............................................................... 26
   4.1 Overview of the ARCADIA Service Graph Model ....................................................... 26
   4.2 Elaboration on the ARCADIA Service Graph Model ................................................... 27

5 Overview of the ARCADIA Service Deployment Model ....................................................... 30
   5.1 Overview of the ARCADIA Service Deployment Model .............................................. 30
   5.2 Elaboration on the ARCADIA Service Deployment Model ......................................... 31

6 Overview of the ARCADIA Service Runtime Model ............................................................ 34
   6.1 Overview of the ARCADIA Service Runtime Model .................................................... 34

7 Future Work on ARCADIA Context Model ........................................................................ 36

Annex I: References .................................................................................................................. 37
Annex II: ARCADIA Context Model (v1.0) Documentation .................................................... 39
List of Figures

Figure 2-1: The ARCADIA modelling facets and their usage .......................................................... 11
Figure 2-2: The ARCADIA architectural components that utilize the context model ..................... 13
Figure 2-3: Juju GUI [4] .................................................................................................................. 14
Figure 2-4: Network Service Example for NFV [3] ........................................................................... 15
Figure 2-5: TOSCA node, capability and relationship types used in NFV application [1] ............... 16
Figure 2-6: The class diagram of the CAMEL metamodel including packages ............................... 17
Figure 3-1: Overview of ARCADIA Component Model ................................................................. 18
Figure 3-2: Overview of ‘ComponentMetadata’ element ............................................................... 20
Figure 3-3: Overview of ‘ComponentConfiguration’ element ....................................................... 21
Figure 3-4: Overview of ‘Requirements’ element ............................................................................. 22
Figure 3-5: Overview of ‘Distribution’ element ................................................................................. 22
Figure 3-6: Overview of ‘ExposedMicroServices’ element ............................................................ 23
Figure 3-7: Overview of ‘CoreHooks’ element ................................................................................. 24
Figure 3-8: Overview of ‘RelationHooks’ element ............................................................................. 25
Figure 4-1: Overview of ARCADIA Service Graph Model ............................................................... 26
Figure 4-2: Overview of ‘GraphNodeIdentifier’ element ............................................................... 27
Figure 4-3: Overview of ‘VirtualLinkDescriptor’ element ............................................................... 28
Figure 4-4: Overview of ‘GraphMeasurableMetric’ element ......................................................... 29
Figure 5-1: Overview of ARCADIA Service Deployment Model ...................................................... 30
Figure 5-2: Overview of ‘ComponentPlacementAction’ element ................................................... 31
Figure 5-3: Overview of ‘IaaSConnectivity’ element ....................................................................... 32
Figure 5-4: Overview of ‘DeploymentConstraints’ element ........................................................... 33
Figure 6-1: Overview of ARCADIA Service Runtime Model ........................................................... 34
## Acronyms

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

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.

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).

![Juju GUI](image)

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.
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.

Although ARCADIA does not adopt an MDE approach (since it follows a juju-like approach which is closer to the HDA lifecycle) 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.
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).
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.
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).
The next element is 'Distribution' (Figure 3-5) which defines the physical location of the Component.

As it is depicted, there are two options for distributing the executable Component. 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 (‘RequiredMicroServices’) provides descriptive information regarding the interfaces that are exposed/consumed by one MicroService.

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 Juju1 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.

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.

As depicted above, the Service Graph Model contains three main elements; the 'GraphNodeDescriptor' element, the 'VirtualLinkDescriptor' element and the 'GraphMonitoringDescriptor' element. 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.

Figure 4-1: Overview of ARCADIA Service Graph Model
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](image)

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.

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.

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](image)

The ‘ServiceProviderDescriptor’ element consists of an ‘IaaSConnectivity’ (see Figure 5-3) element and a ‘DeploymentConstraints’ element (see Figure 5-4).
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 encapsulates 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](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.


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


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 Simple types and navigate to the respective part of the documentation.

<table>
<thead>
<tr>
<th>Elements</th>
<th>Complex types</th>
<th>Simple types</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArcadiaComponentModel</td>
<td>ActionType</td>
<td>ConfigurationValueType</td>
</tr>
<tr>
<td>ArcadiaServiceDeploymentModel</td>
<td>ArcadiaComponentModelType</td>
<td>ExecutionLanguageType</td>
</tr>
<tr>
<td>ArcadiaServiceGraphModel</td>
<td>ArcadiaServiceDeploymentModelType</td>
<td>MeasurementUnit</td>
</tr>
<tr>
<td>ArcadiaServiceRuntimeModel</td>
<td>ArcadiaServiceGraphModelType</td>
<td>ProcessorArchitectureType</td>
</tr>
<tr>
<td>ComponentIdentifier</td>
<td>ArcadiaServiceRuntimeModelType</td>
<td>ServiceCategoryType</td>
</tr>
<tr>
<td>ConfigurationElementIdentifier</td>
<td>ArcadiaServiceRuntimeModel</td>
<td></td>
</tr>
<tr>
<td>GraphNodeIdentifier</td>
<td>ComponentConfigurationType</td>
<td></td>
</tr>
<tr>
<td>MetricIdentifier</td>
<td>ComponentMetadataType</td>
<td></td>
</tr>
<tr>
<td>MicroServiceIdentifier</td>
<td>ExecutionEnvironmentType</td>
<td></td>
</tr>
<tr>
<td>VirtualLinkIdentifier</td>
<td>GraphNodeIdentifierType</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HookType</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MaintainerType</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MeasureableMetricType</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MicroServiceType</td>
<td></td>
</tr>
</tbody>
</table>
**element ArcadiaComponentModel**

**ArcadiaComponentModelType**

**ComponentMetadata**
The Component Metadata element encapsulates descriptive information of the Component Model instance.

**ComponentConfiguration**
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.

**Requirements**
This element encapsulates the set of requirements that have to be met for the smooth execution of the Component Model instance.

**Distribution**
This element includes information regarding the ‘physical’ distribution of the executable Component Model instance.

**ExposedMicroServices**
The element describes the set of exposed MicroServices that are the most granular expressible functions of the Component Model instance.

**RequiredMicroServices**
The element describes the set of required MicroServices which are the most granular consumable functions required by the Component Model instance.

**CoreHooks**
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.

**type** ArcadiaComponentModelType

**properties**
- content complex

**children**
- ComponentMetadata
- ComponentConfiguration
- Requirements
- Distribution
- ExposedMicroServices
- RequiredMicroServices
- CoreHooks
element **ArcadiaServiceDeploymentModel**

- **Diagram**
  - `ArcadiaServiceDeploymentModel`
  - `ArcadiaServiceDeploymentModelType`
    - `ArcadiaServiceGraphModelReference`
    - `ConfigurationDescriptor`
    - `ComponentPlacementDescriptor`

- **Type** `ArcadiaServiceDeploymentModelType`
- **Properties** `content` (complex)
- **Children** `ArcadiaServiceGraphModelReference`, `ConfigurationDescriptor`, `ComponentPlacementDescriptor`

- **Source**
  ```xml
  <xs:element name="ArcadiaComponentModel" type="ArcadiaComponentModelType"/>
  ```

---

element **ArcadiaServiceGraphModel**

- **Diagram**
  - `ArcadiaServiceGraphModel`
  - `ArcadiaServiceGraphModelType` (extension)
    - `GraphNodeDescriptor`
    - `VirtualLinkDescriptor`
    - `GraphMonitoringDescriptor`

- **Type** `ArcadiaServiceGraphModelType`
- **Properties**
  - `GraphNodeDescriptor`
  - `VirtualLinkDescriptor`
  - `GraphMonitoringDescriptor`

- **Source**
  ```xml
  <xs:element name="ArcadiaServiceGraphModel" type="ArcadiaServiceGraphModelType"/>
  ```
D2.2 – Definition of the ARCADIA Context Model

**Type** extension of `ArcadiaServiceGraphModelType`

**Properties** content complex

**Children** `GraphNodeDescriptor` `VirtualLinkDescriptor` `GraphMonitoringDescriptor`

**Source**

```xml
<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**

```xml
<xs:element name="ArcadiaServiceRuntimeModel" type="ArcadiaServiceRuntimeModelType"/>
```

**Element** `ComponentIdentifier`

**Diagram**

- **Type** `xs:string`
- **Properties** content simple
- **Used by** complexTypes `ComponentMetadataType` `GraphNodeIdentifierType`
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.

```xml
<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>
```

This element represents an identifier that differentiates Configuration Element instances. This identifier is used in the frame of an ARCADIA Service DeploymentModel instance.

```xml
<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>
```

This element represents an identifier that differentiates Graph Node instances. This identifier is used in the frame of one Graph. Please bare in mind that one Component MAY participate as multiple Graph Nodes in the same graph.

```xml
<xs:element name="GraphNodeIdentifier">
  <xs:annotation>
    <xs:documentation>
      This element represents an identifier that differentiates Graph Node instances. This identifier is used 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>
```
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.

```xml
<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**

This element represents an identifier that differentiates Metrics.

```xml
<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**

This element represents an identifier that differentiates Micro Service instances.

```xml
<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:**
  - This element represents the identifier of a specific link. The identifier SHALL be unique in the frame of one graph.

- **Type:** xs:string

- **Properties:** content simple

- **Used by:**
  - 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:**
```xml
<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:**
  - This element represents one action that can be performed by one Micro Service.

- **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:**
```xml
<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 ActionName

**Diagram:**

```
<xs:element name="ActionName" type="xs:string">
  <xs:annotation>
    <xs:documentation>This element represents the action name.</xs:documentation>
  </xs:annotation>
</xs:element>
```

**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 ActionDescription

**Diagram:**

```
<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>
```

**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>
```
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.

### complexType ArcadiaComponentModelType

<table>
<thead>
<tr>
<th>children</th>
<th>ComponentMetadata</th>
<th>ComponentConfiguration</th>
<th>Requirements</th>
<th>Distribution</th>
<th>ExposedMicroServices</th>
<th>RequiredMicroServices</th>
<th>CoreHooks</th>
</tr>
</thead>
<tbody>
<tr>
<td>used by</td>
<td></td>
<td>ArcadiaComponentModel</td>
<td>ArcadiaServiceGraphModelType/GraphNodeDescriptor/GraphNodeComponentDescriptor/ComponentDescriptor</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>annotation</td>
<td>documentation</td>
<td>A Component Model</td>
<td>represents</td>
<td>the most</td>
<td>granular deployment</td>
<td>unit of an</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>application.</td>
<td>Component</td>
<td>Model instances</td>
<td>ARCADIA</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Model</td>
<td>instances can</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>be combined</td>
<td>towards</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>the specification</td>
<td>of a Service Graph.</td>
<td></td>
</tr>
</tbody>
</table>
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.

The Component Metadata element encapsulates descriptive information of the Component Model instance.

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.

This element encapsulates the set of requirements that have to be met for the smooth execution of the Component Model instance.

This element encapsulates the Compute, Memory and Storage Resource requirements that SHALL be met during the Resource allocation process by the ARCADIA Smart Controller.

This element represents the Compute Requirements.

This element represents the minimum CPU speed in MHz (e.g., 1200).

This element represents the maximum CPU speed in MHz (e.g., 2400).
<table>
<thead>
<tr>
<th>Element</th>
<th>Name</th>
<th>Type</th>
<th>minOccurs</th>
</tr>
</thead>
<tbody>
<tr>
<td>minCores</td>
<td>xs:int</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>maxCores</td>
<td>xs:int</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Memory</td>
<td>xs:int</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Storage</td>
<td>xs:int</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

This element represents the minimum number of Cores required for the execution (e.g. 2).

This element represents the maximum number of Cores required for the execution (e.g. 10).

This element represents the Memory Requirements.

This element represents the maximum size of RAM (in MB) required for the execution (e.g. 2048)

This element represents the minimum size of RAM (in MB) required for the execution (e.g. 512)

This element represents the Storage Requirements.

This element represents the minimum size of data storage (in MB) required for the execution (e.g. 1000).

This element represents the maximum size of data storage (in MB) required for the execution (e.g. 4000).
<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: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:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</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:element>
D2.2 – Definition of the ARCADIA Context Model

<x:simpleType>
  <xs:restriction base="xs:string">  
    <xs:enumeration value="Linux"/>
    <xs:enumeration value="Windows"/>
  </xs:restriction>
</xs:simpleType>

This element provides information about the required execution environment.

This element includes information regarding the 'physical' distribution of the executable Component Model instance.

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 redundant since it is already available in a mandatory install hook (see CoreHooks elements).

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.

This element describes the set of exposed MicroServices that are the...
most granular exposable functions of the Component Model instance.

This element represents the most granular exposable function of a Component Model instance.

This element describes the set of required MicroServices which are the most granular consumable functions required by the Component Model instance.

This element represents the most granular function that can be consumed by a Component Model instance.

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.

This element encapsulates the installation actions. Installation runs before any other hook. It should be used to perform one-time setup operations only.

This element encapsulates 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 executed at least once when the ARCADIA agent is restarted. Finally, these actions CAN be executed after a configuration change.
### StartHook

This element encapsulates 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.

```xml
<xs:element name="StartHook" type="HookType">
  <xs:annotation>
    <xs:documentation>This element encapsulates 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>
```

### UpgradeHook

This element encapsulates the actions that have to be executed after any upgrade operation that does not itself interrupt an existing error state.

```xml
<xs:element name="UpgradeHook" type="HookType">
  <xs:annotation>
    <xs:documentation>This element encapsulates 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>
```

### StopHook

This element encapsulates 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.

```xml
<xs:element name="StopHook" type="HookType">
  <xs:annotation>
    <xs:documentation>This element encapsulates 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>
```
element ArcadiaComponentModelType/ComponentMetadata
diagram

ComponentMetadataType

This element represents an identifier that differentiates Component Model instances. It COULD be created based on a hash value of ComponentName and Version. The identifier is used in the frame of an ARCADIA Service Graph Model instance.

ComponentName

The element represents the descriptive name of the component. The ComponentName SHALL be unique when combined with the Version element.

Version

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.

Description

A detailed description of the Component Model instance. Description provides information regarding the functionality of the Component Model instance.

Icon

Vector image that symbolizes the Component Model instance.

Maintainer

It represents the Person that is responsible for maintaining the Component Model instance that is published in the ARCADIA Component Repository.

Tags

By convention tags SHOULD contain comma separated Service Categories.

type ComponentMetadataType

properties content complex

children ComponentIdentifier ComponentName Version Description Icon Maintainer Tags
The Component Metadata element encapsulates descriptive information of the Component Model instance.

```xml
<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>
```

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.

```xml
<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

- Requirements
  - This element encapsulates the set of requirements that have to be met for the smooth execution of the Component Model instance.

- ResourceRequirements
  - This element encapsulates the Compute, Memory and Storage Resource requirements that SHALL be met during the Resource allocation process by the ARCADIA Smart Controller.

- HostingRequirements
  - This element encapsulates the Hosting requirements that SHALL be met during the placement of the Component by the ARCADIA Smart Controller.

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
```xml
<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 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:sequence>
                </xs:complexType>
              </xs:element>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:element>
```
<xs:element name="minCores" type="xs:int" minOccurs="0"> required for the execution (e.g. 2).</xs:documentation>
</xs:element>
<xs:element name="maxCores" type="xs:int" minOccurs="0"> required for the execution (e.g. 10).</xs:documentation>
</xs:element>
<xs:element name="Memory" minOccurs="0"> This element represents the Memory Requirements. </xs:documentation>
</xs:element>
<xs:element name="minRAM" type="xs:int" minOccurs="0"> required for the execution (e.g. 512).</xs:documentation>
</xs:element>
<xs:element name="maxRAM" type="xs:int" minOccurs="0"> required for the execution (e.g. 2048).</xs:documentation>
</xs:element>
<xs:element name="Storage" minOccurs="0"> This element represents the Storage Requirements. </xs:documentation>
</xs:element>
<xs:element name="minStorage" type="xs:int" minOccurs="0"> required for the execution (e.g. 1000).</xs:documentation>
</xs:element>
<xs:element name="maxStorage" type="xs:int" minOccurs="0"> required for the execution (e.g. 4000).</xs:documentation>
</xs:element>

This element represents the minimum number of Cores required for the execution (e.g. 2).
This element represents the maximum number of Cores required for the execution (e.g. 10).
This element represents the maximum size of RAM (in MB) required for the execution (e.g. 2048).
This element represents the maximum size of data storage (in MB) required for the execution (e.g. 4000).
This element represents the minimum size of RAM (in MB) required for the execution (e.g. 512).
This element represents the minimum size of data storage (in MB) required for the execution (e.g. 1000).
<table>
<thead>
<tr>
<th>&lt;xs:element name=&quot;StorageType&quot; minOccurs=&quot;0&quot;&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;xs:annotation&gt;</td>
</tr>
<tr>
<td>&lt;xs:documentation&gt;This element represents the type of the storage element that is required.&lt;/xs:documentation&gt;</td>
</tr>
<tr>
<td>&lt;/xs:annotation&gt;</td>
</tr>
<tr>
<td>&lt;xs:simpleType base=&quot;xs:string&quot;&gt;</td>
</tr>
<tr>
<td>&lt;xs:restriction value=&quot;SSD&quot;/&gt;</td>
</tr>
<tr>
<td>&lt;xs:enumeration value=&quot;AmazonS3&quot;/&gt;</td>
</tr>
<tr>
<td>&lt;xs:enumeration value=&quot;HDD&quot;/&gt;</td>
</tr>
<tr>
<td>&lt;/xs:restriction&gt;</td>
</tr>
<tr>
<td>&lt;/xs:simpleType&gt;</td>
</tr>
<tr>
<td>&lt;/xs:element&gt;</td>
</tr>
<tr>
<td>&lt;/xs:complexType&gt;</td>
</tr>
<tr>
<td>&lt;/xs:sequence&gt;</td>
</tr>
<tr>
<td>&lt;/xs:complexType&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>&lt;xs:element name=&quot;HostingRequirements&quot; minOccurs=&quot;0&quot;&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;xs:annotation&gt;</td>
</tr>
<tr>
<td>&lt;xs:documentation&gt;This element encapsulates the Hosting requirements that SHALL be met during the placement of the Component by the ARCADIA Smart Controller.&lt;/xs:documentation&gt;</td>
</tr>
<tr>
<td>&lt;/xs:annotation&gt;</td>
</tr>
<tr>
<td>&lt;xs:complexType&gt;</td>
</tr>
<tr>
<td>&lt;/xs:sequence&gt;</td>
</tr>
<tr>
<td>&lt;/xs:complexType&gt;</td>
</tr>
<tr>
<td>&lt;/xs:element&gt;</td>
</tr>
<tr>
<td>&lt;xs:element name=&quot;OperatingSystem&quot; minOccurs=&quot;0&quot;&gt;</td>
</tr>
<tr>
<td>&lt;xs:annotation&gt;</td>
</tr>
<tr>
<td>&lt;xs:documentation&gt;This element provides information about the required Operating System that is needed by the Component.&lt;/xs:documentation&gt;</td>
</tr>
<tr>
<td>&lt;/xs:annotation&gt;</td>
</tr>
<tr>
<td>&lt;xs:complexType&gt;</td>
</tr>
<tr>
<td>&lt;/xs:sequence&gt;</td>
</tr>
<tr>
<td>&lt;/xs:complexType&gt;</td>
</tr>
<tr>
<td>&lt;/xs:element&gt;</td>
</tr>
<tr>
<td>&lt;xs:element name=&quot;Name&quot; minOccurs=&quot;0&quot;&gt;</td>
</tr>
<tr>
<td>&lt;xs:annotation&gt;</td>
</tr>
<tr>
<td>&lt;xs:documentation&gt;This element represents the descriptive name of the Operating System.&lt;/xs:documentation&gt;</td>
</tr>
<tr>
<td>&lt;/xs:annotation&gt;</td>
</tr>
<tr>
<td>&lt;xs:simpleType base=&quot;xs:string&quot;&gt;</td>
</tr>
<tr>
<td>&lt;xs:restriction value=&quot;Ubuntu14.04LTSamd64&quot;/&gt;</td>
</tr>
<tr>
<td>&lt;xs:enumeration value=&quot;Ubuntu14.04LTSi386&quot;/&gt;</td>
</tr>
<tr>
<td>&lt;/xs:restriction&gt;</td>
</tr>
<tr>
<td>&lt;/xs:simpleType&gt;</td>
</tr>
<tr>
<td>&lt;/xs:element&gt;</td>
</tr>
<tr>
<td>&lt;xs:element name=&quot;ProcessorArchitecture&quot; type=&quot;ProcessorArchitectureType&quot;&gt;</td>
</tr>
<tr>
<td>&lt;xs:annotation&gt;</td>
</tr>
<tr>
<td>&lt;xs:documentation&gt;This element provides information regarding the type of the processor that is required.&lt;/xs:documentation&gt;</td>
</tr>
<tr>
<td>&lt;/xs:annotation&gt;</td>
</tr>
<tr>
<td>&lt;/xs:element&gt;</td>
</tr>
<tr>
<td>&lt;xs:element name=&quot;OSType&quot; minOccurs=&quot;0&quot;&gt;</td>
</tr>
<tr>
<td>&lt;xs:annotation&gt;</td>
</tr>
<tr>
<td>&lt;xs:documentation&gt;This element provides information regarding the type of the Operating System that is required.&lt;/xs:documentation&gt;</td>
</tr>
<tr>
<td>&lt;/xs:annotation&gt;</td>
</tr>
<tr>
<td>&lt;/xs:element&gt;</td>
</tr>
</tbody>
</table>
### D2.2 - Definition of the ARCADIA Context Model

**Element**: ArcadiaComponentModelType/Requirements/ResourceRequirements

<table>
<thead>
<tr>
<th>diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ResourceRequirements</strong></td>
</tr>
<tr>
<td>This element encapsulates the Compute, Memory and Storage Resource requirements that SHALL be met during the Resource allocation process by the ARCADIA Smart Controller.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>minOcc</td>
</tr>
<tr>
<td>maxOcc</td>
</tr>
<tr>
<td>content</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>children</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compute</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>documentation</td>
</tr>
<tr>
<td>This element encapsulates the Compute, Memory and Storage Resource requirements that SHALL be met during the Resource allocation process by the ARCADIA Smart Controller.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>source</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;xs:element name=&quot;ResourceRequirements&quot; minOccurs=&quot;0&quot;&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:annotation&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:documentation&gt;</code>This element encapsulates the Compute, Memory and Storage Resource requirements that SHALL be met during the Resource allocation process by the ARCADIA Smart Controller.&lt;/xs:documentation&gt;`</td>
</tr>
<tr>
<td><code>&lt;xs:annotation&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:complexType&gt;</code></td>
</tr>
</tbody>
</table>

```xml
<xs:complexType>
  <xs:element name="ExecutionEnvironment" type="ExecutionEnvironmentType" minOccurs="0">
    <xs:annotation>
      <xs:documentation>This element provides information about the required execution environment.</xs:documentation>
    </xs:annotation>
  </xs:element>
</xs:complexType>
```
<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>
This element represents the Storage Requirements.

This element represents the minimum size of data storage (in MB) required for the execution (e.g., 1000).

This element represents the maximum size of data storage (in MB) required for the execution (e.g., 4000).

This element represents the type of the storage element that is required.

Valid values for StorageType are:
- SSD
- AmazonS3
- HDD


---

**D2.2 - Definition of the ARCADIA Context Model**

**Diagram: ArcadiaComponentModelType/Requirements/ResourceRequirements/Compute**

- **minCPU Speed**: This element represents the minimum CPU speed in MHz (e.g., 1200).
- **maxCPU Speed**: This element represents the maximum CPU speed in MHz (e.g., 2400).
- **min Cores**: This element represents the minimum number of Cores required for the execution (e.g., 2).
- **max Cores**: This element represents the maximum number of Cores required for the execution (e.g., 10).

**Properties**

- **minOcc**: 0
- **maxOcc**: 1
- **content**: complex

**Children**

- minCPU Speed
- maxCPU Speed
- min Cores
- max Cores

**Annotation**

This element represents the Compute Requirements.

**Source**

```xml
<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="minCPU Speed" 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="maxCPU Speed" 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="min Cores" 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="max Cores" 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>
```
the execution (e.g. 10).</xs:documentation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>

**element**

**ArcadiaComponentModelType/Requirements/ResourceRequirements/Compute/minCPUSpeed**

| diagram | <minCPUSpeed>
| 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 | <maxCPUSpeed>
| 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

<table>
<thead>
<tr>
<th>diagram</th>
<th>This element represents the minimum number of Cores required for the execution (e.g. 2).</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>xs:int</td>
</tr>
<tr>
<td>properties</td>
<td>minOcc 0  maxOcc 1  content simple</td>
</tr>
<tr>
<td>annotation</td>
<td>documentation  This element represents the minimum number of Cores required for the execution (e.g. 2).</td>
</tr>
<tr>
<td>source</td>
<td>&lt;xs:element name=&quot;minCores&quot; type=&quot;xs:int&quot; minOccurs=&quot;0&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td><a href="">xs:annotation</a></td>
</tr>
<tr>
<td></td>
<td><a href="">xs:documentation</a>This element represents the minimum number of Cores required for the execution (e.g. 2).&lt;/xs:documentation&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/xs:annotation&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/xs:element&gt;</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>diagram</th>
<th>This element represents the maximum number of Cores required for the execution (e.g. 10).</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>xs:int</td>
</tr>
<tr>
<td>properties</td>
<td>minOcc 0  maxOcc 1  content simple</td>
</tr>
<tr>
<td>annotation</td>
<td>documentation  This element represents the maximum number of Cores required for the execution (e.g. 10).</td>
</tr>
<tr>
<td>source</td>
<td>&lt;xs:element name=&quot;maxCores&quot; type=&quot;xs:int&quot; minOccurs=&quot;0&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td><a href="">xs:annotation</a></td>
</tr>
<tr>
<td></td>
<td><a href="">xs:documentation</a>This element represents the maximum number of Cores required for the execution (e.g. 10).&lt;/xs:documentation&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/xs:annotation&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/xs:element&gt;</td>
</tr>
</tbody>
</table>
element ArcadiaComponentModelType/Requirements/ResourceRequirements/Memory

properties
minOcc 0
maxOcc 1
content simple

children minRAM maxRAM

annotation
This element represents the Memory Requirements.

source
<x:s:element name="Memory" minOccurs="0">
  <x:s:annotation>
    <x:s:documentation>This element represents the Memory Requirements.</x:s:documentation>
  </x:s:annotation>
  <x:s:complexType>
    <x:s:sequence>
      <x:s:element name="minRAM" type="xs:int" minOccurs="0">
        <x:s:annotation>
          <x:s:documentation>This element represents the minimum size of RAM (in MB) required for the execution (e.g., 512)</x:s:documentation>
        </x:s:annotation>
      </x:s:element>
      <x:s:element name="maxRAM" type="xs:int" minOccurs="0">
        <x:s:annotation>
          <x:s:documentation>This element represents the maximum size of RAM (in MB) required for the execution (e.g., 2048)</x:s:documentation>
        </x:s:annotation>
      </x:s:element>
    </x:s:sequence>
  </x:s:complexType>
</x:s:element>

element ArcadiaComponentModelType/Requirements/ResourceRequirements/Memory/minRAM

diagram

This element represents the minimum size of RAM (in MB) required for the execution (e.g., 512)

properties
minOcc 0
maxOcc 1
content simple

type xs:int
**D2.2 – Definition of the ARCADIA Context Model**

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

- **diagram**

  ![Diagram](image)

- **type** xs:int

- **properties**
  - minOccurs 0
  - maxOccurs 1
  - content simple

- **annotation documentation**
  This element represents the minimum size of RAM (in MB) required for the execution (e.g. 512)

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

- **diagram**

  ![Diagram](image)

- **type** xs:int

- **properties**
  - minOccurs 0
  - maxOccurs 1
  - content simple

- **annotation documentation**
  This element represents the maximum size of RAM (in MB) required for the execution (e.g. 2048)

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

- **diagram**

  ![Diagram](image)

- **type** Storage

- **properties**
  - minOccurs 0
  - maxOccurs 1
  - content complex
This element represents the Storage Requirements.

```xml
<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>
```

This element represents the minimum size of data storage (in MB) required for the execution (e.g. 1000).
element ArcadiaComponentModelType/Requirements/ResourceRequirements/Storage/minStorage
diagram

This element represents the minimum size of data storage (in MB) required for the execution (e.g. 1000).

source 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

This element represents the maximum size of data storage (in MB) required for the execution (e.g. 4000).

source 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

This element represents the type of the storage element that is required.

source 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:element>
This element encapsulates the Hosting requirements that SHALL be met during the placement of the Component by the ARCADIA Smart Controller.

- **OperatingSystem**
  - This element provides information about the required Operating System that is needed by the Component.

- **ExecutionEnvironment**
  - This element provides information about the required execution environment aspect.

**Properties**
- minOccurs: 0
- maxOccurs: 1
- content: complex

**Children**
- OperatingSystem
- ExecutionEnvironment

**Source**
```xml
<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:element>
      <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:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

**XSD**
```xml
<xs:simpleType name="HostingRequirements" base="xs:string">
  <xs:restriction base="xs:string">
    <xs:enumeration value="SSD"/>
    <xs:enumeration value="AmazonS3"/>
    <xs:enumeration value="HDD"/>
  </xs:restriction>
</xs:simpleType>
```
D2.2—Definition of the ARCADIA Context Model

```xml
<xsd:restriction>
  <xsd:simpleType>
    <xsd:element name="ProcessorArchitecture" type="ProcessorArchitectureType">  
      <xsd:annotation>
        <xsd:documentation>This element provides information regarding the type of the processor that is required.</xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="OSType" minOccurs="0">  
      <xsd:annotation>
        <xsd:documentation>This element provides information regarding the type of the Operating System that is required.</xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="ExecutionEnvironment" type="ExecutionEnvironmentType" minOccurs="0">  
      <xsd:annotation>
        <xsd:documentation>This element provides information about the required execution environment aspect.</xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:simpleType>
</xsd:restriction>
```

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

![Diagram of OperatingSystem, ProcessorArchitecture, and OSType elements](image)

**Diagram**

- **OperatingSystem**
  - This element provides information about the required Operating System that's needed by the Component.

- **ProcessorArchitecture**
  - This element provides information regarding the type of the processor that is required.

- **OSType**
  - This element provides information regarding the type of the Operating System that is required.

**Properties**

- minOcc 0
- maxOcc 1
This element provides information about the required Operating System that is needed by the Component.

```xml
<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>
```

This element represents the descriptive name of the Operating System.
D2.2 – Definition of the ARCADIA Context Model

<table>
<thead>
<tr>
<th>properties</th>
<th>minOcc</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>maxOcc</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>content</td>
<td>simple</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>facets</th>
<th>Kind</th>
<th>Value</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>enumeration</td>
<td>Ubuntu14.04LTSamd64</td>
<td></td>
</tr>
<tr>
<td></td>
<td>enumeration</td>
<td>Ubuntu14.04LTSi386</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>annotation</th>
<th>documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This element represents the descriptive name of the Operating System.</td>
</tr>
</tbody>
</table>

| 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**

```
+---------------------------------------------+
| Processor Architecture                     |
+---------------------------------------------+
```

This element provides information regarding the type of the processor that is required.

| type | ProcessorArchitectureType |
|      |                            |

<table>
<thead>
<tr>
<th>properties</th>
<th>content</th>
<th>simple</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>facets</th>
<th>Kind</th>
<th>Value</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>enumeration</td>
<td>amd64</td>
<td></td>
</tr>
<tr>
<td></td>
<td>enumeration</td>
<td>i386</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>annotation</th>
<th>documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This element provides information regarding the type of the processor that is required.</td>
</tr>
</tbody>
</table>

| 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**
  - `OSType` diagram
  - This element provides information regarding the type of the Operating System that is required.

- **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**
  - This element provides information about the required execution environment aspect.

- **Type**
  - `ExecutionEnvironmentType`
This element provides information about the required execution environment aspect.

This element includes information regarding the 'physical' distribution of the executable Component Model instance.

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 redundant since it is already available in a mandatory install hook (see CoreHooks elements).

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 Fossie requirements) and execute the install-hooks upon the Image installation. If a specific Image is defined the ARCADIA Programmable Resource Manager should register and launch this Image prior to the Install Hook.
This element includes information regarding the 'physical' distribution of the executable Component Model instance.

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 redundant since it is already available in a mandatory install hook (see CoreHooks elements).

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.

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 redundant since it is already available in a mandatory install hook (see CoreHooks elements).
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.
### Definition of the ARCADIA Context Model

**element** `ArcadiaComponentModelType/ExposedMicroServices`

<table>
<thead>
<tr>
<th>Diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="ExposedMicroServices Diagram" /></td>
</tr>
</tbody>
</table>

This element describes the set of exposed MicroServices that are the most granular exposable functions of the Component Model instance.

**Properties**
- **content**: `complex`

**Children**
- `ExposedMicroService`

**Annotation**
This element describes the set of exposed MicroServices that are the most granular exposable functions of the Component Model instance.

**Source**

```xml
<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

This element represents the most granular exposable function of a Component Model instance.

diagram

<table>
<thead>
<tr>
<th>type</th>
<th>MicroServiceType</th>
</tr>
</thead>
<tbody>
<tr>
<td>properties</td>
<td></td>
</tr>
<tr>
<td></td>
<td>minOcc</td>
</tr>
<tr>
<td></td>
<td>maxOcc</td>
</tr>
<tr>
<td></td>
<td>content</td>
</tr>
<tr>
<td>annotation</td>
<td>documentation</td>
</tr>
<tr>
<td></td>
<td>This element represents the most granular exposable function of a Component Model instance.</td>
</tr>
</tbody>
</table>
| 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

<table>
<thead>
<tr>
<th>Diagram</th>
<th>This element describes the set of required MicroServices which are the most granular consumable functions required by the Component Model instance.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Properties</td>
<td>minOcc: 0, maxOcc: 1, content: complex</td>
</tr>
<tr>
<td>Children</td>
<td>RequiredMicroService</td>
</tr>
<tr>
<td>Annotation</td>
<td>This element describes the set of required MicroServices which are the most granular consumable functions required by the Component Model instance.</td>
</tr>
<tr>
<td>Source</td>
<td>XML code showing the definition of the RequiredMicroServices element</td>
</tr>
</tbody>
</table>

```xml
<xs:element name="RequiredMicroServices" 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:complexType>
    <xs:sequence>
      <xs:element name="RequiredMicroService" type="MicroServiceType" minOccurs="0" maxOccurs="unbounded">
        <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:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```
This element represents the most granular function that can be consumed by a Component Model instance.

**element** ArcadiaComponentModelType/RequiredMicroServices/RequiredMicroService

- **MicroServiceType**
  - **MicroServiceDescriptor**
    - Describes the most granular function that can be consumed by a Component Model instance.
  - **MonitoringDescriptor**
    - Contains the set of metrics that can be used to perform runtime monitoring activities.
  - **RelationHooks**
    - Represents the hooks associated with the discovery, configuration, and removal of relations.
  - **ActionDescriptor**
    - Describes the set of actions that can be used to perform specific reconfigurations.
  - **LinkMonitoringDescriptor**
    - Represents the measurable metrics that relate to a specific link/relationship.
  - **LinkActionDescriptor**
    - Describes the actions that relate to the QoS of a link/relationship.

**type** MicroServiceType

- **properties**
  - minOccurs: 0
  - maxOccurs: unbounded
  - content: complex

- **children**
  - MicroServiceDescriptor
  - MonitoringDescriptor
  - RelationHooks
  - ActionDescriptor
  - LinkMonitoringDescriptor
  - LinkActionDescriptor

**source**

```xml
<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**

This diagram 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.

**InstallHook**

This element encapsulates the installation actions. Installation runs before any other hook. It should be used to perform one-time setup operations only.

**ConfigChangedHook**

This element encapsulates the actions that have to be executed in order to perform a configuration according to the configuration complex type. 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 executed at least once when the ARCADIA agent is restarted. Finally, these actions CAN be executed after a configuration change.

**StartHook**

This element encapsulates the actions that have to be executed immediately after the first configchanged 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.

**UpgradeHook**

This element encapsulates the actions that have to be executed after any upgrade operation that does not itself interrupt an existing error state.

**StopHook**

This element encapsulates 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.

**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.

<table>
<thead>
<tr>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;xs:element name=&quot;CoreHooks&quot;&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:annotation&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:documentation&gt;</code>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.&lt;/xs:documentation&gt;`</td>
</tr>
<tr>
<td><code>&lt;xs:complexType&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:sequence&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:element name=&quot;InstallHook&quot; type=&quot;HookType&quot;&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:annotation&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:documentation&gt;</code>This element encapsulates the installation actions. Installation runs before any other hook. It should be used to perform one-time setup operations only.&lt;/xs:documentation&gt;`</td>
</tr>
<tr>
<td><code>&lt;xs:element name=&quot;ConfigChangedHook&quot; type=&quot;HookType&quot;&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:annotation&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:documentation&gt;</code>This element encapsulates 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 executed at least once when the ARCADIA agent is restarted. Finally, these actions CAN be executed after a configuration change.&lt;/xs:documentation&gt;`</td>
</tr>
<tr>
<td><code>&lt;xs:element name=&quot;StartHook&quot; type=&quot;HookType&quot;&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:annotation&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:documentation&gt;</code>This element encapsulates 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.&lt;/xs:documentation&gt;`</td>
</tr>
<tr>
<td><code>&lt;xs:element name=&quot;UpgradeHook&quot; type=&quot;HookType&quot;&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:annotation&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:documentation&gt;</code>This element encapsulates the actions that have to be executed after any upgrade operation that does not itself interrupt an existing error state.&lt;/xs:documentation&gt;`</td>
</tr>
<tr>
<td><code>&lt;xs:element name=&quot;StopHook&quot; type=&quot;HookType&quot;&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:annotation&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:documentation&gt;</code>This element encapsulates 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.&lt;/xs:documentation&gt;`</td>
</tr>
</tbody>
</table>
element ArcadiaComponentModelType/CoreHooks/InstallHook

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

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 executed at 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 executed at least once when the ARCADIA agent is restarted. Finally, these actions CAN be executed after a configuration change.</xs:documentation>
  </xs:annotation>
</xs:element>
source

<xs:element name="ConfigChangedHook" type="HookType">
  <xs:annotation>
    <xs:documentation>
      This element encapsulates 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 executed at 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

This element encapsulates 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 encapsulates 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>
This element encapsulates the actions that have to be executed after any upgrade operation that does not itself interrupt an existing error state.

```xml
<xs:element name="UpgradeHook" type="HookType">
  <xs:annotation>
    <xs:documentation>
      This element encapsulates the actions that have to be executed after any upgrade operation that does not itself interrupt an existing error state.<br />
    </xs:documentation>
  </xs:annotation>
</xs:element>
```

This element encapsulates 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.

```xml
<xs:element name="StopHook" type="HookType">
  <xs:annotation>
    <xs:documentation>
      This element encapsulates 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.<br />
    </xs:documentation>
  </xs:annotation>
</xs:element>
```
complexType ArcadiaServiceDeploymentModelType

diagram

ArcadiaServiceDeploymentModelType

ArcadiaServiceGraphModelReference

This element encapsulates the reference graph that is used.

ConfigurationDescriptor

This element encapsulates the information for the selected configuration for each Component that participates in a Service Graph.

ComponentPlacementDescriptor

This element encapsulates the information for the multi-IaaS Placement of each Component that participates in a Service Graph.

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: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:documentation>This element encapsulates the reference graph that is used.</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="ArcadiaServiceGraphModelReference" type="ArcadiaServiceGraphModelType"/>
    <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:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="ComponentPlacementDescriptor" ref="GraphNodeIdentifier"/>
  </xs:sequence>
</xs:complexType>
D2.2 – Definition of the ARCADIA Context Model

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.

This element represents the actual configuration value that is enforced by the ARCADIA Smart Controller.

This element encapsulates the information for the multi-IaaS Placement of each Component that participates in a Service Graph.

This element encapsulates all deployable actions that have to be performed by the ARCADIA Smart Controller through the ARCADIA Agents.

This element represents one Graph Node Component of the DG that represents a Service Graph.

This element encapsulates information regarding the association of one Component with the selected IaaS resources.

This element provides information about a specific IaaS.
<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 base="xs:string">pattern="OpenStack"/></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:element name="DeploymentConstraints">
    <xs:annotation>
      <xs:documentation>This element encapsulates 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:element>
  <xs:element name="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></xs:documentation>
    </xs:annotation>
  </xs:element>
</xs:sequence>
<xs:documentation>This element encapsulates information related to actual execution environment that SHALL be instantiated in an IaaS.</xs:documentation>

<xs:complexType>
  <xs:sequence>
    <xs:element name="InstanceName" type="xs:string"/>
    <xs:getDocumentation>InstanceName will be autogenerated based on the ComponentIdentifier</xs:getDocumentation>
    <xs:element name="ImageIdentifier" type="xs:string"/>
    <xs:getDocumentation>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:getDocumentation>
    <xs:element name="Flavor">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="FlavorIdentifier"/>
          <xs:element name="FlavorName" type="xs:string"/>
          <xs:element name="vCPUs" type="xs:int"/>
          <xs:getDocumentation>Number of CPUs</xs:getDocumentation>
          <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:element name="SecurityRequirements">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="SecurityGroupId"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="NetworkingRequirements">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="SecurityGroupId"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>

This element provides information related to the desired flavor that should be used in the chosen IaaS. It will be elaborated in version 2 of the model.

This element provides security constraints that have to be met upon deployment. It will be elaborated in version 2 of the model.

This element provides network constraints that have to be met upon deployment. It will be refined in version 2 of the model.
element ArcadiaServiceDeploymentModelType/ArcadiaServiceGraphModelReference

diagram

This element encapsulates the reference graph that is used.

ArcadiaServiceGraphModelReference

This element encapsulates information regarding the graph nodes of the Directed Acyclic Graph (DAG).

VirtualLinkDescriptor

This element provides information related to the links of the DAG.

GraphMonitoringDescriptor

This element encapsulates information regarding monitoring metrics that relate to the entire graph. These metrics are measured by monitoring probes which are under the supervision of the Arcadia Smart Controller.

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>
This element encapsulates the information for the selected configuration for each Component that participates in a Service Graph.

This element encapsulates all the configuration actions that can be undertaken by the ARCADIA Smart Controller for configuring a specific component.

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.

This element represents the actual configuration value that is enforced by the ARCADIA Smart Controller.
element ArcadiaServiceDeploymentModelType/ConfigurationDescriptor/ConfigurationAction

This element encapsulates all the configuration actions that can be undertaken by the Smart Controller for configuring a specific component.

children
- GraphNodeIdentifier
- ConfigurationElementIdentifier
- ConfigurationValue

annotation
This element encapsulates all the configuration actions that can be undertaken by the Smart Controller for configuring a specific component.

source

\[
<\text{xs:element} name="ConfigurationAction" maxOccurs="unbounded">
  <\text{xs:annotation}>
    <\text{xs:documentation}>This element encapsulates all the configuration actions that can be undertaken by the Smart Controller for configuring a specific component.\text{}</xs:documentation>
  </xs:annotation>
  <\text{xs:complexType}>
    <\text{xs:sequence}>
      <\text{xs:element ref="GraphNodeIdentifier"}>
        <\text{xs:annotation}>
          <\text{xs:documentation}>This element represents an identifier that differentiates Graph Nodes in the frame of one Graph. Please bear in mind that one Component MAY participate as multiple Graph Nodes in the same graph.\text{}</xs:documentation>
        </xs:annotation>
      </xs:element>
      <\text{xs:element ref="ConfigurationElementIdentifier"}>
        <\text{xs:annotation}>
          <\text{xs:documentation}>This element represents an identifier that differentiates Configuration Element instances. This identifier is used in the frame of an ARCADIA Service DeploymentModel instance.\text{}</xs:documentation>
        </xs:annotation>
      </xs:element>
      <\text{xs:element name="ConfigurationValue" type="xs:string"}>
        <\text{xs:annotation}>
          <\text{xs:documentation}>This element represents the actual configuration value that is enforced by the ARCADIA Smart Controller.\text{}</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
\]
element  
ArcadiaServiceDeploymentModelType/ConfigurationDescriptor/ConfigurationAction/ConfigurationValue

- Diagram:
  - This element represents the actual configuration value that is enforced by the ARCADIA Smart Controller.

- Properties:
  - content: simple

- Annotation:
  - 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:annotation>`
- `<xs:element>`

---

element  
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor

- Diagram:
  - This element encapsulates the information for the multi-IaaS Placement of each Component that participates in a Service Graph.

- Properties:
  - content: complex

- Children:
  - ComponentPlacementAction

- Annotation:
  - 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:annotation>`
- `<xs:complexType>`
- `<xs:sequence>`
  - `<xs:element name="ComponentPlacementAction" maxOccurs="unbounded" type="ComponentPlacementAction"/>`
  - `<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:annotation>`
  - `<xs:element>`
<x:s:complexType>
  <x:s:sequence>
    <x:s:element name="DeployableComponent">
      <x:s:annotation>
        <x:s:documentation>This element represents one Graph Node Component of the DG that represents a Service Graph.</x:s:documentation>
      </x:s:annotation>
    </x:s:element>
    <x:s:element ref="GraphNodeIdentifier"/>
    <x:s:complexType>
      <x:s:sequence>
        <x:s:element name="IaaSConnectivity">
          <x:s:annotation>
            <x:s:documentation>This element provides information about a specific IaaS that is selected in the frame of one placement.</x:s:documentation>
          </x:s:annotation>
        </x:s:element>
        <x:s:element name="IaaSType">
          <x:s:annotation>
            <x:s:documentation>This element provides information about the type of the IaaS that is selected in the frame of one placement.</x:s:documentation>
          </x:s:annotation>
        </x:s:element>
        <x:s:element name="APIEndpoint" type="xs:string">
          <x:s:annotation>
            <x:s:documentation>This element provides the public accessible API of the IaaS provider.</x:s:documentation>
          </x:s:annotation>
        </x:s:element>
        <x:s:element name="APIVersion">
          <x:s:annotation>
            <x:s:documentation>This element provides information regarding the version of the selected API in order to assure compatibility.</x:s:documentation>
          </x:s:annotation>
        </x:s:element>
        <x:s:element name="Credentials" type="xs:string">
          <x:s:annotation>
            <x:s:documentation>This elements encapsulates security information required in order to perform authentication and authorization to the IaaS.</x:s:documentation>
          </x:s:annotation>
        </x:s:element>
      </x:s:sequence>
      <x:s:complexType>
        <x:s:sequence>
          <x:s:element name="ServiceProviderDescriptor">
            <x:s:annotation>
              <x:s:documentation>This element encapsulates information regarding the association of one Component with the selected IaaS resources.</x:s:documentation>
            </x:s:annotation>
          </x:s:element>
        </x:s:sequence>
      </x:s:complexType>
    </x:s:element>
  </x:s:sequence>
</x:s:complexType>
**Definition of the ARCADIA Context Model**

- **TenantIdentifier**: This element provides information regarding the specific tenant that is associated with the deployment.

- **DeploymentConstraints**: This element encapsulates information related to the technical constraints that are imposed by the ARCADIA Smart Controller in order to perform proper instantiation of the Execution Environment.

- **ExecutionEnvironment**: This element provides information related to the type of the execution environment that is required.

- **InstanceName**: InstanceName will be autogenerated based on the ComponentIdentifier.

- **ImageIdentifier**: The Image that will be used will either be defined by the component or selected based on the available registered images in the IaaS.

- **Flavor**: This element provides information related to the desired flavor that should be used in the chosen IaaS.

### Complex Types

#### ExecutionEnvironmentType

- **InstanceName**
  - Type: `xs:string`

- **ImageIdentifier**
  - Type: `xs:string`

- **Flavor**
  - **FlavorIdentifier**
    - Type: `xs:string`
  - **FlavorName**
    - Type: `xs:string`
  - **vCPUs**
    - Type: `xs:int`
<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: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:complexType>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
**element**

ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction

<table>
<thead>
<tr>
<th>Diagram</th>
<th><strong>ComponentPlacementAction</strong> 1..∞</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>DeployableComponent</strong></td>
</tr>
<tr>
<td></td>
<td><strong>ServiceProviderDescriptor</strong></td>
</tr>
</tbody>
</table>

This element represents one Graph Node Component of the DAG that represents a Service Graph.

This element encapsulates information regarding the association of one Component with the selected IaaS resources.

---

**properties**

- minOcc: 1
- maxOcc: unbounded
- content: complex

**children**

- DeployableComponent
- ServiceProviderDescriptor

**annotation**

This element encapsulates all deployable actions that have to be performed by the ARCADIA Smart Controller through the ARCADIA Agents.

**source**

```xml
<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 ref="GraphNodeIdentifier"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

This element represents one Graph Node Component of the DG that represents a Service Graph.

```xml
<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>
```

This element encapsulates information regarding the association of one Component with the selected IaaS resources.

```xml
<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</xs:documentation>
              </xs:annotation>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```
that is selected in the frame of one placement.

```xml
<xs:complexType name="APIEndpoint" type="xs:string">
</xs:complexType>
```

This element provides the public accessible API of the IaaS provider.

```xml
<xs:complexType name="APIVersion">
</xs:complexType>
```

This element provides information regarding the version of the selected API in order to assure compatibility.

```xml
<xs:complexType name="Credentials" type="xs:string">
</xs:complexType>
```

This elements encapsulates security information required in order to perform authentication and authorization to the IaaS.

```xml
<xs:complexType name="TenantIdentifier" type="xs:string">
</xs:complexType>
```

This element provides information regarding the specific tenant that is associated with the deployment.

```xml
<xs:complexType name="DeploymentConstraints">
</xs:complexType>
```

This element encapsulates information related to the technical constraints that are imposed by the ARCADIA Smart Controller in order to perform proper instantiation of the Execution Environment.

```xml
<xs:complexType name="ExecutionEnvironment" type="ExecutionEnvironmentType">
</xs:complexType>
```

This element provides information related to the type of the execution environment that is required.

```xml
<xs:complexType name="CoreRequirements">
</xs:complexType>
```

This element encapsulates information related to actual execution environment that SHALL be instantiated in an IaaS.
<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: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="SecurityGroupId"/>
    </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>
This element represents one Graph Node Component of the DG that represents a Service Graph.
This element encapsulates information regarding the association of one Component with the selected IaaS resources.

This element provides information about a specific IaaS.

This element encapsulates information related to the technical constraints that are imposed by the ARCADIA Smart Controller in order to perform proper instantiation of the Execution Environment.

This element encapsulates information regarding the association of one Component with the selected IaaS resources.

This element provides information about the type of the IaaS that is selected in the frame of one placement.

This element provides the public accessible API of the IaaS provider.

This element provides information regarding the version of the selected API in order to assure compatibility.
<xs:element name="Credentials" type="xs:string">
  <xs:annotation>
    <xs:documentation>This element 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:complexType>
  <xs:element name="DeploymentConstraints" type="xs:string">
    <xs:annotation>
      <xs:documentation>This element encapsulates 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:element>
</xs:complexType>

<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" type="xs:string">
  <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:element>

<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 registered 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:element>
<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 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>
This element provides information about a specific IaaS.:

- **IaaSType**: This element provides information about the type of the IaaS that is selected in the frame of one placement.

- **APIEndpoint**: This element provides the public accessible API of the IaaS provider.

- **APIVersion**: This element provides information regarding the version of the selected API in order to assure compatibility.

- **Credentials**: This element encapsulates security information required in order to perform authentication and authorization to the IaaS.

- **TenantIdentifier**: This element provides information regarding the specific tenant that is associated with the deployment.

**properties**

- content: complex

**children**

- IaaSType
- APIEndpoint
- APIVersion
- Credentials
- TenantIdentifier

**annotation**

This element provides information about a specific IaaS.

**source**

```xml
<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 base="xs:string">
          <xs:pattern value="OpenStack"/>
        </xs:simpleType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```
D2.2 – Definition of the ARCADIA Context Model

```
<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 element 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>
```

---

**IaaSType**

- **type**: restriction of `xs:string`
- **properties**: content simple
- **facets**:
  - Kind: pattern
  - Value: OpenStack
- **annotation documentation**: This element provides information about the type of the IaaS that is selected in the frame of one placement.

```
<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 base="xs:string">
    <xs:pattern value="OpenStack"/>
  </xs:simpleType>
</xs:element>
```
element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction/ServiceProviderDescriptor/IaaSConnectivity/APIEndpoint
diagram

This element provides the public accessible API of the IaaS provider.

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

This element provides information regarding the version of the selected API in order to assure compatibility.

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/laaSConnectivity/Credentials`

<table>
<thead>
<tr>
<th>Diagram</th>
<th>This element encapsulates security information required in order to perform authentication and authorization to the IaaS.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>type</strong></td>
<td><code>xs:string</code></td>
</tr>
<tr>
<td><strong>properties</strong></td>
<td><code>content simple</code></td>
</tr>
<tr>
<td><strong>annotation</strong></td>
<td>This element encapsulates security information required in order to perform authentication and authorization to the IaaS.</td>
</tr>
<tr>
<td><strong>source</strong></td>
<td><code>&lt;xs:element name=&quot;Credentials&quot; type=&quot;xs:string&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;xs:annotation&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;xs:documentation&gt;This elements encapsulates security information required in order to perform authentication and authorization to the IaaS.&lt;/xs:documentation&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;xs:annotation&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;xs:element&gt;</code></td>
</tr>
</tbody>
</table>

**element**

`ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction/ServiceProviderDescriptor/laaSConnectivity/TenantIdentifier`

<table>
<thead>
<tr>
<th>Diagram</th>
<th>This element provides information regarding the specific tenant that is associated with the deployment.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>type</strong></td>
<td><code>xs:string</code></td>
</tr>
<tr>
<td><strong>properties</strong></td>
<td><code>content simple</code></td>
</tr>
<tr>
<td><strong>annotation</strong></td>
<td>This element provides information regarding the specific tenant that is associated with the deployment.</td>
</tr>
<tr>
<td><strong>source</strong></td>
<td><code>&lt;xs:element name=&quot;TenantIdentifier&quot; type=&quot;xs:string&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;xs:annotation&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;xs:documentation&gt;This element provides information regarding the specific tenant that is associated with the deployment.&lt;/xs:documentation&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;xs:annotation&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;xs:element&gt;</code></td>
</tr>
</tbody>
</table>
This element encapsulates information related to the technical constraints that are imposed by the ARCADIA Smart Controller in order to perform proper instantiation of the Execution Environment.

**DeploymentConstraints**

This element encapsulates information related to the technical constraints that are imposed by the ARCADIA Smart Controller in order to perform proper instantiation of the Execution Environment.

**CoreRequirements**

This element encapsulates information related to actual execution environment that **SHALL** be instantiated in an IaaS.

**SecurityRequirements**

This element provides security constraints that have to be met upon deployment. It will be elaborated in version 2 of the model.

**NetworkingRequirements**

This element provides network constraints that have to be met upon deployment. It will be refined in version 2 of the model.

**ExecutionEnvironment**

This element provides information related to the type of the execution environment that is required.

**InstanceName**

InstanceName will be autogenerated based on the
ComponentIdentifier</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="ImageIdentifier" type="xs:string">The Image that will be used will either be defined by the component or selected based on the available registered images in the IaaS.</xs:documentation>
</xs:annotation>
</xs:element>
<xs:element name="Flavor">This element provides information related to the desired flavor that should be used in the chosen IaaS.</xs:documentation>
</xs:annotation>
</xs:element>
<xs:complexType>
<xs:sequence>
<xs:element name="FlavorIdentifier" type="xs:string"/>
<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:element name="SecurityRequirements">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:element>
<xs:element name="SecurityGroupIdentifier"/>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="NetworkingRequirements">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:element>
<xs:element name="SubNetIdentifier"/>
</xs:sequence>
</xs:complexType>
element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction/ServiceProviderDescriptor/Diagram

This element provides information related to the type of the execution environment that is required.

<table>
<thead>
<tr>
<th>type</th>
<th>ExecutionEnvironmentType</th>
</tr>
</thead>
<tbody>
<tr>
<td>properties</td>
<td>content complex</td>
</tr>
<tr>
<td>VirtualMachine BareMetal Container</td>
<td></td>
</tr>
<tr>
<td>annotation</td>
<td>documentation</td>
</tr>
</tbody>
</table>

This element represents the Virtual Machine characteristics that are required.

This element represents the BareMetal characteristics that are required.

This element represents the Container characteristics that are 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>
This element encapsulates information related to actual execution environment that SHALL be instantiated in an IaaS.

- **InstanceName**
  - InstanceName will be autogenerated based on the ComponentIdentifier.

- **ImageIdentifier**
  - The Image that will be used will either be defined by the component or selected based on the available registered images in the IaaS.

- **Flavor**
  - This element provides information related to the desired flavor that should be used in the chosen IaaS.

XML content:

```xml
<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:element name="ImageIdentifier" type="xs:string"/>
      <xs:element name="Flavor" type="xs:string"/>
      <xs:element name="FlavorName" type="xs:string"/>
      <xs:element name="vCPUs" type="xs:int"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```
### Definition of the ARCADIA Context Model

#### XML Schema Definition

```xml
<xs:complexType name="RootDisk" type="xs:int"/>
<xs:complexType name="EphemeralDisk" type="xs:int"/>
<xs:complexType name="RAM" type="xs:int"/>
```

#### InstanceName Element

InstanceName will be autogenerated based on the ComponentIdentifier.

<table>
<thead>
<tr>
<th>Source</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;xs:element name=&quot;InstanceName&quot; type=&quot;xs:string&quot;&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;xs:annotation&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;xs:documentation&gt;</code></td>
<td></td>
</tr>
<tr>
<td>InstanceName will be autogenerated based on the ComponentIdentifier</td>
<td></td>
</tr>
<tr>
<td><code>&lt;/xs:documentation&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;/xs:annotation&gt;</code></td>
<td></td>
</tr>
</tbody>
</table>

#### ImageIdentifier Element

The Image that will be used will either be defined by the component or selected based on the available registered images in the IaaS.

<table>
<thead>
<tr>
<th>Source</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;xs:element name=&quot;ImageIdentifier&quot; type=&quot;xs:string&quot;&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;xs:annotation&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;xs:documentation&gt;</code></td>
<td></td>
</tr>
<tr>
<td>The Image that will be used will either be defined by the component or selected based on the available registered images in the IaaS.</td>
<td></td>
</tr>
<tr>
<td><code>&lt;/xs:documentation&gt;</code></td>
<td></td>
</tr>
<tr>
<td><code>&lt;/xs:annotation&gt;</code></td>
<td></td>
</tr>
</tbody>
</table>
The Image that will be used will either be defined by the component or selected based on the available registered images in the IaaS.

This element provides information related to the desired flavor that should be used in the chosen IaaS.

This element provides information related to the desired flavor that should be used in the chosen IaaS.

This element provides information related to the desired flavor that should be used in the chosen IaaS.
element 
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction 
/ServiceProviderDescriptor/DeploymentConstraints/CoreRequirements/Flavor/vCPUs

diagram 

<table>
<thead>
<tr>
<th>diagram</th>
<th>vCPUs</th>
</tr>
</thead>
<tbody>
<tr>
<td>content</td>
<td>Number of CPUs</td>
</tr>
<tr>
<td>source</td>
<td>&lt;xs:element name=&quot;vCPUs&quot; type=&quot;xs:int&quot;&gt;</td>
</tr>
<tr>
<td></td>
<td><a href="">xs:annotation</a></td>
</tr>
<tr>
<td></td>
<td><a href="">xs:documentation</a>Number of CPUs&lt;/xs:documentation&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/xs:annotation&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;/xs:element&gt;</td>
</tr>
</tbody>
</table>

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

diagram 

<table>
<thead>
<tr>
<th>diagram</th>
<th>RootDisk</th>
</tr>
</thead>
<tbody>
<tr>
<td>content</td>
<td>simple</td>
</tr>
<tr>
<td>source</td>
<td>&lt;xs:element name=&quot;RootDisk&quot; type=&quot;xs:int&quot;/&gt;</td>
</tr>
</tbody>
</table>
element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction/ServiceProviderDescriptor/DelayConstraints/CoreRequirements/Flavor/EphemeralDisk

diagram

| EphemeralDisk |

type  
xs:int

properties  
content  simple

source  
<x:s:element name="EphemeralDisk" type="xs:int"/>

---

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

diagram

| RAM |

type  
xs:int

properties  
content  simple

source  
<x:s:element name="RAM" type="xs:int"/>

---

element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction/ServiceProviderDescriptor/DelayConstraints/SecurityRequirements

diagram

| SecurityRequirements | SecurityGroupIdentifier |

This element provides security constraints that have to be met upon deployment. It will be elaborated in version 2 of the model.

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  
<x:s:element name="SecurityRequirements">
  <x:s:annotation>
    <x:s:documentation>This element provides security constraints that have to be met upon deployment. It will be elaborated in version 2 of the model.</x:s:documentation>
  </x:s:annotation>
  <x:s:complexType>
    <x:s:sequence>
      <x:s:element name="SecurityGroupIdentifier"/>
    </x:s:sequence>
  </x:s:complexType>
</x:s:element>
element
ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction
/ServiceProviderDescriptor/DeliveryConstraints/DeploymentConstraints/SecurityRequirements/SecurityGroupIdentifier

diagram

| SecurityGroupIdentifier |

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

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

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/DeliveryConstraints/DeploymentConstraints/NetworkingRequirements/SubNetIdentifier

source
<xs:element name="SubNetIdentifier"/>
complexType ArcadiaServiceGraphModelType

diagram

ArcadiaServiceGraphModelType

GraphNodeDescriptor

VirtualLinkDescriptor

GraphMonitoringDescriptor

children

GraphNodeDescriptor VirtualLinkDescriptor GraphMonitoringDescriptor

used by

elements
ArcadiaServiceGraphModel
ArcadiaServiceDeploymentModelType/ArcadiaServiceGraphModelReference

annotation
documentation

The specific element represents a Service Graph. A Service Graph instance is practically a Directed Acyclic Graph (DAG).

source

<x:simpleType name="ArcadiaServiceGraphModelType">
    <xs:annotation>
        <xs:documentation>The specific element represents a Service Graph. A Service Graph instance is practically a Directed Acyclic Graph (DAG).</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 Acyclic Graph (DAG).</xs:documentation>
            </xs:annotation>
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="GraphNodeComponentDescriptor" maxOccurs="unbounded">This element encapsulates the entire model that represents one component. This representation is necessary in order to have self-sustained schemas.</xs:element>
                    <xs:element name="GraphNodeIdentifierDescriptor" type="GraphNodeIdentifierType">This element provides the definition of the GraphNodeIdentifier along with its dependent identifiers i.e. the Component Identifier and the MicroService Identifier.</xs:element>
                    <xs:element name="ComponentDescriptor" type="ArcadiaComponentModelType">This element represents one Component Model instance that is
used one or more times in the frame of a graph.<xs:documentation>
</xs:documentation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:documentation>
</xs:element>
<xs:element name="VirtualLinkDescriptor">
<xs:annotation>
<xs:documentation>This element provides information related to the links of the DG.<xs:documentation>
</xs:documentation>
</xs:element>
</xs:complexType>
</xs:element>
</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:documentation>
</xs:element>
</xs:complexType>
</xs:element>
<xs:element name="SourceComponent">
<xs:annotation>
<xs:documentation>The element encapsulates information about the Source Node of the Virtual Link.<xs:documentation>
</xs:documentation>
</xs:element>
</xs:complexType>
</xs:element>
<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:documentation>
</xs:element>
</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:documentation>
</xs:element>
</xs:complexType>
</xs:element>
<xs:element ref="GraphNodeIdentifier"/>
<xs:element ref="MicroServiceIdentifier"/>
<xs:element name="GraphMonitoringDescriptor">
<xs:annotation>
<xs:documentation>The element encapsulates information about the Source Node of the Virtual Link.<xs:documentation>
</xs:documentation>
</xs:element>
</xs:complexType>
</xs:element>
<xs:element ref="GraphNodeIdentifier"/>
<xs:element ref="MicroServiceIdentifier"/>
<xs:element name="GraphMonitoringDescriptor" minOccurs="0">
<xs:annotation>
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.

This element represents one metric that refers to the graph.

This element encapsulates the entire model that represents one component. This representation is necessary in order to have self-sustained schemas.

This element encapsulates information regarding the graph nodes of the Directed Acyclic Graph (DAG).

This element provides the definition of the GraphNodeIdentifier along with its dependent identifiers i.e. the Component Identifier and the MicroService Identifier.
element `ArcadiaServiceGraphModelType/GraphNodeDescriptor/GraphNodeComponentDescriptor`

- **Diagram:**
  - `GraphNodeComponentDescriptor` (1,∞) - This element encapsulates the entire model that represents one component. This representation is necessary in order to have self-sustained schemas.

- **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 Code:**
```
<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:element>
```

- **Example:**
```
<xs:element name="GraphNodeComponentDescriptor" 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>
```
This element provides the definition of the `GraphNodeIdentifier` along with its dependent identifiers i.e. the Component Identifier and the MicroService Identifier.

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 Instances in the same graph.

This element is used in the frame of an ARCADIA Service Graph Model instance.
element
ArcadiaServiceGraphModelType/GraphNodeDescriptor/GraphNodeComponentDescriptor/ComponentDescriptor
diagram
ArcadiaComponentModelType

- **ComponentMetadata**
  The Component Metadata element encapsulates descriptive information of the Component Model instance.

- **ComponentConfiguration**
  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.

- **Requirements**
  This element encapsulates the set of requirements that have to be met for the smooth execution of the Component Model instance.

- **Distribution**
  This element includes information regarding the 'physical' distribution of the executable Component Model instance.

- **ExposedMicroServices**
  This element describes the set of exposed MicroServices that are the most granular exposed functions of the Component Model instance.

- **RequiredMicroServices**
  This element describes the set of required MicroServices which are the most granular consumable functions required by the Component Model Instance.

- **CoreHooks**
  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.

<table>
<thead>
<tr>
<th>type</th>
<th>ArcadiaComponentModelType</th>
</tr>
</thead>
<tbody>
<tr>
<td>properties</td>
<td>content complex</td>
</tr>
</tbody>
</table>
## D2.2 – Definition of the ARCADIA Context Model

<table>
<thead>
<tr>
<th>children</th>
<th>ComponentMetadata</th>
<th>ComponentConfiguration</th>
<th>Requirements</th>
<th>Distribution</th>
<th>ExposedMicroServices</th>
</tr>
</thead>
</table>

### Annotation

- **ComponentDescriptor**

  This element represents one Component Model instance that is used one or more times in the frame of a graph.

### Source

```xml
<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

![Diagram](image)

- **VirtualLinkDescriptor**: This element provides information related to the links of the DG.
- **VirtualLink**: This element represents one Virtual link of the Service Graph.

#### Properties

- **content**: complex

#### Children

- **VirtualLink**

#### Annotation

- **documentation**: This element provides information related to the links of the DG.

### Source

```xml
<xs:element name="VirtualLinkDescriptor">
  <xs:annotation>
    <xs:documentation>This element provides information related to the links of the DG.</xs:documentation>
  </xs:annotation>
</xs:element>
```

```xml
<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:element>
```

```xml
<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:element>
    <xs:element ref="GraphNodeIdentifier"/>
    <xs:element ref="MicroServiceIdentifier"/>
  </xs:sequence>
</xs:complexType>
```

```xml
<xs:element name="DestinationComponent">
  <xs:annotation>
    <xs:documentation></xs:documentation>
  </xs:annotation>
</xs:element>
```
The element encapsulates information about the Destination Node of the Virtual Link. This element represents one Virtual Link of the Service Graph. The element encapsulates information about the Source Node of the Virtual Link. The element encapsulates information about the Destination Node of the Virtual Link.
D2.2 – Definition of the ARCADIA Context Model

<xs:complexType>
  <xs:sequence>
    <xs:element ref="GraphNodeIdentifier"/>
    <xs:element ref="MicroServiceIdentifier"/>
  </xs:sequence>
</xs:complexType>

<xs:element name="DestinationComponent">
  The element encapsulates information about the Destination Node of the Virtual Link.
</xs:element>

<xs:complexType>
  <xs:sequence>
    <xs:element ref="GraphNodeIdentifier"/>
    <xs:element ref="MicroServiceIdentifier"/>
  </xs:sequence>
</xs:complexType>

<xs:element name="SourceComponent">
  The element encapsulates information about the Source Node of the Virtual Link.
</xs:element>

<xs:complexType>
  <xs:sequence>
    <xs:element ref="GraphNodeIdentifier"/>
    <xs:element ref="MicroServiceIdentifier"/>
  </xs:sequence>
</xs:complexType>

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

This element represents an identifier that differentiates Graph Nodes in the same graph.

This element represents an identifier that differentiates Micro Service instances.

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

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

This element represents an identifier that differentiates Graph Nodes in the frame of one Graph. Please be aware that one Component MAY participate at multiple Graph Nodes in the same graph.

This element represents an identifier that differentiates Micro Service instances.

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

This element represents one metric that refers to the graph.

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

### complexType ArcadiaServiceRuntimeModelType

<table>
<thead>
<tr>
<th>Diagram</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Diagram" /></td>
<td>This element represents one deployed ARCADIA ServiceGraph.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Children</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeploymentModel</td>
<td>This element encapsulates an ARCADIA Deployment model instance.</td>
</tr>
<tr>
<td>RuntimeBindings</td>
<td>This element encapsulates runtime information of all running Component instances that have been placed by the ARCADIA Smart Controller in the available IaaS resources.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;xs:complexType name=&quot;ArcadiaServiceRuntimeModelType&quot;&gt;</td>
<td>This element represents one deployed ARCADIA ServiceGraph.</td>
</tr>
<tr>
<td><a href="">xs:documentation</a>This element represents one deployed ARCADIA ServiceGraph.&lt;/xs:documentation&gt;</td>
<td></td>
</tr>
<tr>
<td><a href="">xs:sequence</a></td>
<td></td>
</tr>
<tr>
<td>&lt;xs:element name=&quot;DeploymentModel&quot; type=&quot;ArcadiaServiceDeploymentModelType&quot;&gt;</td>
<td>This element encapsulates an ARCADIA Deployment model instance.</td>
</tr>
<tr>
<td><a href="">xs:annotation</a></td>
<td></td>
</tr>
<tr>
<td><a href="">xs:documentation</a>This element represents one deployed ARCADIA ServiceGraph.&lt;/xs:documentation&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;xs:element name=&quot;RuntimeBindings&quot;&gt;</td>
<td></td>
</tr>
<tr>
<td><a href="">xs:annotation</a></td>
<td></td>
</tr>
<tr>
<td><a href="">xs:documentation</a>This element encapsulates runtime information of all running Component instances that have been placed by the ARCADIA Smart Controller in the available IaaS resources.&lt;/xs:documentation&gt;</td>
<td></td>
</tr>
<tr>
<td><a href="">xs:sequence</a></td>
<td></td>
</tr>
<tr>
<td>&lt;xs:element name=&quot;RuntimeBinding&quot; maxOccurs=&quot;unbounded&quot;&gt;</td>
<td>This element represents all binding information of one running Component instance.</td>
</tr>
<tr>
<td><a href="">xs:annotation</a></td>
<td></td>
</tr>
<tr>
<td><a href="">xs:documentation</a>This element represents one metric that refers to the graph.&lt;/xs:documentation&gt;</td>
<td></td>
</tr>
<tr>
<td><a href="">xs:sequence</a></td>
<td></td>
</tr>
<tr>
<td>&lt;xs:element ref=&quot;GraphNodeIdentifier&quot;/&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;xs:element name=&quot;IaaSPlacementDescriptor&quot;/&gt;</td>
<td></td>
</tr>
</tbody>
</table>
This element provides IaaS runtime details. It will be refined in version 2 of the model.

This element provides Configuration runtime details. It will be refined in version 2 of the model.

This element provides Endpoint runtime details. It will be refined in version 2 of the model.

This element provides runtime status details. It will be refined in version 2 of the model.

This element provides runtime monitoring details. It will be refined in version 2 of the model.

This element provides Link Monitoring runtime details. It will be refined in version 2 of the model.
**element ArcadiaServiceRuntimeModelType/DeploymentModel**

This element encapsulates an ARCADIA Deployment model instance.

*type* ArcadiaServiceDeploymentModelType

*properties* content complex

*children* ArcadiaServiceGraphModelReference ConfigurationDescriptor ComponentPlacementDescriptor

*annotation* documentation This element encapsulates an ARCADIA Deployment model instance.

*source*

```xml
<x:s:element name="DeploymentModel" type="ArcadiaServiceDeploymentModelType">
  <x:s:annotation>
    <x:s:documentation>
      This element encapsulates an ARCADIA Deployment model instance.
    </x:s:documentation>
  </x:s:annotation>
</x:s:element>
```

**element ArcadiaServiceRuntimeModelType/RuntimeBindings**

This element encapsulates runtime information of all running Component instances that have been placed by the ARCADIA Smart Controller in the available IaaS resources.

*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*

```xml
<x:s:element name="RuntimeBindings">
  <x:s:annotation>
    <x:s: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.
    </x:s:documentation>
  </x:s:annotation>
</x:s:element>
```
<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:element>
    <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>
element ArcadiaServiceRuntimeModelType/RuntimeBindings/RuntimeBinding

diagram

- GraphNodeIdentifier
  - 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.

- IaaSPlacementDescriptor
  - This element provides IaaS runtime details. It will be refined in version 2 of the model.

- MicroServiceRuntimeConfiguration
  - This element provides Configuration runtime details. It will be refined in version 2 of the model.

- MicroServiceStatusDescriptor
  - This element provides runtime status details. It will be refined in version 2 of the model.

- MicroServiceMonitoringDescriptor
  - This element provides runtime monitoring details. It will be refined in version 2 of the model.

- LinkMonitoringDescriptor
  - This element provides Link Monitoring runtime details. It will be refined in version 2 of the model.

properties
- minOcc: 1
- maxOcc: unbounded
- content: complex

children
- GraphNodeIdentifier
- IaaSPlacementDescriptor
- LinkMonitoringDescriptor
- MicroServiceRuntimeConfiguration
- MicroServiceStatusDescriptor
- MicroServiceMonitoringDescriptor

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>
          <xs:documentation></xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="LinkMonitoringDescriptor">
        <xs:annotation>
          <xs:documentation></xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="MicroServiceRuntimeConfiguration">
        <xs:annotation>
          <xs:documentation></xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="MicroServiceStatusDescriptor">
        <xs:annotation>
          <xs:documentation></xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="MicroServiceMonitoringDescriptor">
        <xs:annotation>
          <xs:documentation></xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
This element provides IaaS runtime details. It will be refined in version 2 of the model.
**Definition of the ARCADIA Context Model**

### MicroServiceRuntimeConfiguration

- **Diagram:** [Diagram of MicroServiceRuntimeConfiguration]
  - This element provides Configuration runtime details. It will be refined in version 2 of the model.

- **Source:**
  ```xml
  <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>
  ```

### MicroServiceEndpointDescriptor

- **Diagram:** [Diagram of MicroServiceEndpointDescriptor]
  - This element provides Endpoint runtime details. It will be refined in version 2 of the model.

- **Source:**
  ```xml
  <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>
  ```

### MicroServiceStatusDescriptor

- **Diagram:** [Diagram of MicroServiceStatusDescriptor]
  - This element provides runtime status details. It will be refined in version 2 of the model.

- **Source:**
  ```xml
  <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

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

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

documentation

This element encapsulates the entire configuration profile of one component.

Each ConfigurationElement represents a configuration variable of the Component.

children ConfigurationElement

used by element ArcadiaComponentModelType/ComponentConfiguration
### Definition of the ARCADIA Context Model

This element encapsulates the entire configuration profile of one component.

```xml
<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>
```
Each ConfigurationElement represents one configuration variable of the Component.

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
```xml
<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>

**element ComponentConfigurationType/ConfigurationElement/DescriptiveLabel**

<table>
<thead>
<tr>
<th>diagram</th>
<th>The element represents the descriptive label of a configuration variable (e.g. port).</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>xs:string</td>
</tr>
<tr>
<td>properties</td>
<td>content simple</td>
</tr>
<tr>
<td>annotation documentation</td>
<td>The element represents the descriptive label of a configuration variable (e.g. port).</td>
</tr>
<tr>
<td>source</td>
<td>&lt;xs:element name=&quot;DescriptiveLabel&quot; type=&quot;xs:string&quot;&gt;</td>
</tr>
</tbody>
</table>
The element represents the descriptive label of a configuration variable (e.g. port).

```
<xs:annotation>
  <xs:documentation>
  This element represents the descriptive label of a configuration variable. This element SHALL be validated based on the Value Type provided below.
  </xs:documentation>
</xs:annotation>
```

**Element** ComponentConfigurationType/ConfigurationElement/DefaultValue

```
<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 Value Type provided below.
    </xs:documentation>
  </xs:annotation>
</xs:element>
```

**Element** ComponentConfigurationType/ConfigurationElement/Description

```
<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**

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**

```xml
<xsd:element name="ValueType" type="ConfigurationValueType">
  <xsd:annotation>
    <xsd: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.</xsd:documentation>
  </xsd:annotation>
</xsd:element>
```

**element ComponentConfigurationType/ConfigurationElement/ProposedEnumeration**

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**

```xml
<xsd:element name="ProposedEnumeration" type="xs:string" minOccurs="0">
  <xsd:annotation>
    <xsd: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.</xsd:documentation>
  </xsd:annotation>
</xsd:element>
```
regarding possible values (e.g. 8080,9090 for a port variable) of a configuration variable in case the ValueType is denoted as ENUMERATION.
This element encapsulates the descriptive information of one Component Model instance.

```xml
<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" type="xs:string" />
    <xs:element name="ComponentName" type="xs:string" />
    <xs:element name="Version" type="xs:string" />
    <xs:element name="Description" type="xs:string" />
    <xs:element name="Icon" type="xs:base64Binary" minOccurs="0" />
    <xs:element name="Maintainer" type="MaintainerType" />
    <xs:element name="Tags" type="xs:string" />
  </xs:sequence>
</xs:complexType>
```
element ComponentMetadataType/ComponentName

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

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

<table>
<thead>
<tr>
<th>diagram</th>
<th>A detailed description of the Component Model instance. Description provides information regarding the functionality of the Component Model instance.</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>xs:string</td>
</tr>
<tr>
<td>properties</td>
<td>content simple</td>
</tr>
<tr>
<td>annotation</td>
<td>A detailed description of the Component Model instance. Description provides information regarding the functionality of the Component Model instance.</td>
</tr>
</tbody>
</table>

**Source**

```xml
<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

<table>
<thead>
<tr>
<th>diagram</th>
<th>Vector image that symbolizes the Component Model instance.</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>xs:base64Binary</td>
</tr>
<tr>
<td>properties</td>
<td>minOcc 0  maxOcc 1  content simple</td>
</tr>
<tr>
<td>annotation</td>
<td>Vector image that symbolizes the Component Model instance.</td>
</tr>
</tbody>
</table>

**Source**

```xml
<xs:element name="Icon" type="xs:base64Binary" minOccs="0">
  <xs:annotation>
    <xs:documentation>Vector image that symbolizes the Component Model instance.</xs:documentation>
  </xs:annotation>
</xs:element>
```
element ComponentMetadataType/Maintainer

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

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`

This element provides information regarding the execution environment of one component.

Diagram:

- `VirtualMachine`:
  - This element represents the Virtual Machine characteristics that are required.
- `BareMetal`:
  - This element represents the bare-metal characteristics that are required.
- `Container`:
  - This element represents the Container characteristics that are required.

**children**: VirtualMachine BareMetal Container

**used by**:
- ArcadiaComponentModelType/Requirements/HostingRequirements/ExecutionEnvironment
- ArcadiaServiceDeploymentModelType/ComponentPlacementDescriptor/ComponentPlacementAction/ServiceProviderDescriptor/DeploymentConstraints/ExecutionEnvironment

**documentation**:
This element provides information regarding the execution environment of one component.

**source**:
```xml
<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`

<table>
<thead>
<tr>
<th>Diagram</th>
<th>Properties</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Diagram" /></td>
<td>minOcc: 0, maxOcc: 1</td>
<td>This element represents the Virtual Machine characteristics that are required.</td>
</tr>
</tbody>
</table>

```xml
<xs:element name="VirtualMachine" minOccurs="0">
    <xs:documentation>This element represents the Virtual Machine characteristics that are required.</xs:documentation>
</xs:element>
```

### element `ExecutionEnvironmentType/BareMetal`

<table>
<thead>
<tr>
<th>Diagram</th>
<th>Properties</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image2.png" alt="Diagram" /></td>
<td>minOcc: 0, maxOcc: 1</td>
<td>This element represents the bare-metal characteristics that are required.</td>
</tr>
</tbody>
</table>

```xml
<xs:element name="BareMetal" minOccurs="0">
    <xs:documentation>This element represents the bare-metal characteristics that are required.</xs:documentation>
</xs:element>
```

### element `ExecutionEnvironmentType/Container`

<table>
<thead>
<tr>
<th>Diagram</th>
<th>Properties</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image3.png" alt="Diagram" /></td>
<td>minOcc: 0, maxOcc: 1</td>
<td>This element represents the Container characteristics that are required.</td>
</tr>
</tbody>
</table>

```xml
<xs:element name="Container" minOccurs="0">
    <xs:documentation>This element represents the Container characteristics that are required.</xs:documentation>
</xs:element>
```
<xs:documentation>This element represents the Container characteristics that are required.</xs:documentation>
</xs:annotation>
</xs:element>

### complexType `GraphNodeIdentifierType`

<table>
<thead>
<tr>
<th>diagram</th>
<th><code>GraphNodeIdentifierType</code></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This element encapsulates the basic information that is required in order to represent one Microservice as part of a Service Graph.</td>
</tr>
</tbody>
</table>

#### children

- `GraphNodeIdentifier ComponentIdentifier`

#### used by

- `ArcadiaServiceGraphModelType/GraphNodeDescriptor/GraphNodeComponentDescriptor/GraphNodeIdentifierDescriptor`

#### annotation

This element encapsulates the basic information that is required in order to represent one Microservice as part of a Service Graph.

#### source

```xml
<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:element ref="ComponentIdentifier"/>
   </xs:sequence>
</xs:complexType>
```

- `GraphNodeIdentifier`: 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 Instances in the same graph.
- `ComponentIdentifier`: 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.
complexType HookType

diagram

This element encapsulates the proper actions that have
to be performed in the frame
of one hook.

HookAction

1..\infty

This element represents a
specific hook that is executed
by an ARCADIA Agent.

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

<x:s:complexType
    name="HookType">
  <x:s:annotation>
    <x:s:documentation>This element encapsulates the proper actions that have to be performed in the frame of one hook.</x:s:documentation>
  </x:s:annotation>
  <x:s:sequence>
    <x:s:element name="HookAction" maxOccurs="unbounded">
      <x:s:annotation>
        <x:s:documentation>This element represents a specific hook that is executed by an ARCADIA Agent.</x:s:documentation>
      </x:s:annotation>
      <x:s:complexType>
        <x:s:sequence>
          <x:s:element name="HookActionIdentifier">
            <x:s:annotation>
              <x:s:documentation>This element represents an identifier that differentiates the Hook Actions.</x:s:documentation>
            </x:s:annotation>
          </x:s:element>
          <x:s:element name="HookActionLabel">
            <x:s:annotation>
              <x:s:documentation>This element provides a descriptive name for a Hook Action.</x:s:documentation>
            </x:s:annotation>
          </x:s:element>
          <x:s:element name="ExecutionLanguage" type="ExecutionLanguageType">
            <x:s:annotation>
              <x:s:documentation>This element defines the execution language that is used in order to execute the ExecutablePayload.</x:s:documentation>
            </x:s:annotation>
          </x:s:element>
        </x:s:sequence>
      </x:s:complexType>
    </x:s:element>
  </x:s:sequence>
</x:s:complexType>
This element represents a specific hook that is executed by an ARCADIA Agent.

```xml
<xs:complexType name="HookAction">
  <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">
      <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 encapsulates the executable payload.</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

Diagram:

- `HookActionIdentifier`: This element represents an identifier that differentiates the Hook Actions.
- `HookActionLabel`: This element provides a descriptive name for a Hook Action.
- `ExecutionLanguage`: This element defines the execution language that is used in order to execute the ExecutablePayload.
- `ExecutablePayload`: This element encapsulates the executable payload.

Properties:
- minOccurs: 1
- maxOccurs: unbounded
- content: complex

Children:
- `HookActionIdentifier`
- `HookActionLabel`
- `ExecutionLanguage`
- `ExecutablePayload`

Documentation:
- This element represents a specific hook that is executed by an ARCADIA Agent.
 element **HookType/HookAction/HookActionIdentifier**

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

**Diagram**

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**

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

**Diagram**

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

```
<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>
```

## ExecutionLanguageType

- **content**: simple

### Facets

<table>
<thead>
<tr>
<th>Kind</th>
<th>Value</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>enumeration</td>
<td>BASH</td>
<td></td>
</tr>
<tr>
<td>enumeration</td>
<td>JAVA</td>
<td></td>
</tr>
<tr>
<td>enumeration</td>
<td>PYTHON</td>
<td></td>
</tr>
<tr>
<td>enumeration</td>
<td>RUBY</td>
<td></td>
</tr>
</tbody>
</table>

### 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

```
<xs:element name="ExecutablePayload">
<xs:annotation>
<xs:documentation>
This element encapsulates the executable payload.
</xs:documentation>
</xs:annotation>
</xs:element>
```

### Annotation

- **documentation**: This element encapsulates the executable payload.

### Source

```
<xs:element name="ExecutablePayload">
<xs:annotation>
<xs:documentation>
This element encapsulates the executable payload.
</xs:documentation>
</xs:annotation>
</xs:element>
```
complexType MaintainerType

This element represents one natural person that is responsible for maintaining an ARCADIA Component Model instance description.

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:documentation>This element represents one natural person that is responsible for maintaining an ARCADIA Component Model instance description.</xs:documentation>
  <xs:documentation>
    The firstname of the maintainer.
  </xs:documentation>
  <xs:documentation>
    The lastname of the maintainer.
  </xs:documentation>
  <xs:documentation>
    The email address of the maintainer.
  </xs:documentation>
  <xs:documentation>
    The Public Key which CAN be used in order to verify the signature of a Component Model instance.
  </xs:documentation>

  <xs:sequence>
    <xs:element name="Firstname" type="xs:string" minOccurs="0">
      <xs:documentation>The firstname of the maintainer.</xs:documentation>
    </xs:element>
    <xs:element name="LastName" type="xs:string" minOccurs="0">
      <xs:documentation>The lastname of the maintainer.</xs:documentation>
    </xs:element>
    <xs:element name="Email" type="xs:string">
      <xs:documentation>The email address of the maintainer.</xs:documentation>
    </xs:element>
    <xs:element name="PubKey" type="xs:string" minOccurs="0">
      <xs:documentation>The Public Key which CAN be used in order to verify the signature of a Component Model instance.</xs:documentation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
### element MaintainerType/Firstname

**Diagram:**

```
Firstname
```

The firstname of the maintainer.

**Type:** xs:string

**Properties:**
- minOccurs: 0
- maxOccurs: 1
- content: simple

**Annotation:**

The firstname of the maintainer.

**Source:**

```xml
<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 MmaintainerType/LastName

**Diagram:**

```
LastName
```

The lastname of the maintainer.

**Type:** xs:string

**Properties:**
- minOccurs: 0
- maxOccurs: 1
- content: simple

**Annotation:**

The lastname of the maintainer.

**Source:**

```xml
<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 MmaintainerType/Email

**Diagram:**

```
Email
```

The email address of the maintainer.

**Type:** xs:string

**Properties:**
- content: simple

**Annotation:**

The email address of the maintainer.

**Source:**

```xml
<xs:element name="Email" type="xs:string">
</xs:element>
```
D2.2 – Definition of the ARCADIA Context Model

---

**element** `MaintainerType/PubKey`

<table>
<thead>
<tr>
<th>Diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Public Key which CAN be used in order to verify the signature of a Component Model instance.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th><code>xs:string</code></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>minOcc: 0</td>
</tr>
<tr>
<td>maxOcc: 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Public Key which CAN be used in order to verify the signature of a Component Model instance.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;xs:element name=&quot;PubKey&quot; type=&quot;xs:string&quot; minOccurs=&quot;0&quot;&gt;</code></td>
</tr>
</tbody>
</table>

---

**complexType** `MeasureableMetricType`

<table>
<thead>
<tr>
<th>Diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>This element represents one quantitative measurable metric.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Children</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>MetricIdentifier</code>, <code>MetricName</code>, <code>MeasurementUnit</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Used by Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>MicroServiceType/ActionDescriptor/QoSActions/QoSAction/AffectedByMetric</code></td>
</tr>
<tr>
<td><code>MicroServiceType/LinkActionDescriptor/QoSActions/QoSAction/AffectedByMetric</code></td>
</tr>
<tr>
<td><code>ArcadiaServiceGraphModelType/GraphMonitoringDescriptor/GraphMeasurableMetric</code></td>
</tr>
<tr>
<td><code>MicroServiceType/LinkMonitoringDescriptor/MeasurableMetric</code></td>
</tr>
<tr>
<td><code>MicroServiceType/MonitoringDescriptor/MeasureableMetric</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>documentation</code></td>
</tr>
</tbody>
</table>

---
### Element Definition

**MeasureableMetricType**

This element represents one quantitative measurable metric.

```xml
<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**
  - **MetricName**
    - The element represents the descriptive name of the metric (e.g. HTTPResponseTime).

- **Source**
  ```xml
  <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**

<table>
<thead>
<tr>
<th>diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Diagram" /></td>
</tr>
</tbody>
</table>

- **This element represents the unit of measurement that is used for a specific MeasurableMetric (e.g., seconds).**

<table>
<thead>
<tr>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td>MeasurementUnit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>content</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>facets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kind</td>
</tr>
<tr>
<td>enumeration</td>
</tr>
<tr>
<td>enumeration</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>This element represents the unit of measurement that is used for a specific MeasurableMetric (e.g., seconds).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>source</th>
</tr>
</thead>
</table>
| <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

This element represents the most granular exposable function of a Component Model instance.

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
```xml
<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 information regarding one exposed microservice.</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="MonitoringDescriptor">
        <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:element>
      <xs:element name="RelationHooks">
        <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 Agents which is coordinated by the ARCADIA Smart Controller.</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="ActionDescriptor">
        <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:element>
      <xs:element name="LinkMonitoringDescriptor">
        <xs:annotation>
          <xs:documentation>This element encapsulates the measurable invariants that relate to a specific link/relationship.</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="LinkActionDescriptor">
        <xs:annotation>
          <xs:documentation>This element encapsulates the actions that relate to the QoS of a link/relationship.</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
</xs:complexType>
```
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.

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.

This element encapsulates the set of Metrics that CAN be used in order to perform runtime monitoring activities.

This element refers to a specific quantitative monitoring metric which is a performance indicator for the specific MicroService.

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.

This element encapsulates the set of actions that has to be performed when a new relationship is discovered and established.

This element encapsulates the set of actions that has to be performed when an established relationship is affected by a change.

This element encapsulates the set of actions that has to be
performed when an established relationship is going to be gracefully terminated. This element encapsulates the set of actions that has to be performed when an established relationship is going to be abnormally terminated. This element encapsulates the set of Actions that CAN be used in order to perform specific reconfigurations. This element represents the set of actions that are binded to specific Measurable Metrics. This element represents a specific QoS action. 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.
This element encapsulates a set of custom actions that can be invoked by the ARCADIA Smart Controller. This element represents one atomic action that can be invoked.

This element encapsulates the measurable metrics that relate to a specific link/relationship. This element refers to a specific quantitative monitoring metric which is a performance indicator for a specific relationship/link.

This element encapsulates the actions that relate to the QoS of a link/relationship. This element represents the set of actions that are binded to specific Measurable Metrics of the link/relationship. This element represents one atomic action that can be invoked.
element MicroServiceType/MicroServiceDescriptor

diagram

This element encapsulates descriptive information regarding one exposed microservice.

MicroServiceIdentifier
This element represents an identifier that differentiates Micro Service instances.

MicroServiceName
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.

ServiceCategory
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.

properties
content complex

children MicroServiceIdentifier MicroServiceName ServiceCategory
### MicroServiceType/MicroServiceDescriptor/MicroServiceName

#### Source

```xml
<xs:element name="MicroServiceDescriptor">
  <xs:annotation>
    <xs:documentation>This element encapsulates descriptive information 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>
```

#### Annotation

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.
element `MicroServiceType/MicroServiceDescriptor/ServiceCategory`

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

```xml
<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`

This element encapsulates the set of Metrics that CAN be used in order to perform runtime monitoring activities.

properties

- minOccurs: 0
- maxOccurs: 1
- content: complex

children

- `MeasureableMetric`

source

```xml
<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>
```
### D2.2 – Definition of the ARCADIA Context Model

**element** `MicroServiceType/MonitoringDescriptor/MeasureableMetric`

<table>
<thead>
<tr>
<th>diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Diagram" /></td>
</tr>
</tbody>
</table>

- **MeasureableMetricType**
  - **MetricIdentifier**
    - This element represents an identifier that differentiates Metric instances.
  - **MetricName**
    - This element represents the descriptive name of the metric (e.g., `HTTPResponseTime`).
  - **MeasurementUnit**
    - This element represents the unit of measurement that is used for a specific `MeasureableMetric` (e.g., seconds).

<table>
<thead>
<tr>
<th>type</th>
<th><code>MeasureableMetricType</code></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>minOcc</td>
</tr>
<tr>
<td>maxOcc</td>
</tr>
<tr>
<td>content</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>children</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>MetricIdentifier</code> <code>MetricName</code> <code>MeasurementUnit</code></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>documentation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>source</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;xs:element name=&quot;MeasureableMetric&quot; type=&quot;MeasureableMetricType&quot; maxOccurs=&quot;unbounded&quot;&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:annotation&gt;</code></td>
</tr>
<tr>
<td><code>&lt;xs:documentation&gt;</code> This element refers to a specific quantitative monitoring metric which is a performance indicator for the specific MicroService.</td>
</tr>
<tr>
<td><code>&lt;/xs:annotation&gt;</code></td>
</tr>
<tr>
<td><code>&lt;/xs:element&gt;</code></td>
</tr>
</tbody>
</table>
**element MicroServiceType/RelationHooks**

<table>
<thead>
<tr>
<th>properties</th>
<th>minOcc</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>maxOcc</td>
<td>1</td>
</tr>
<tr>
<td>content</td>
<td>complex</td>
<td></td>
</tr>
</tbody>
</table>

**children**
- RelationJoinedHook
- RelationChangedHook
- RelationDepartedHook
- RelationBrokenHook

**annotation**

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/RelationBrokenHook

diagram

RelationBrokenHook

HookType

This element encapsulates the set of actions that has to be performed when an established relationship is going to be gracefully terminated.

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

- name="RelationBrokenHook" type="HookType">

element MicroServiceType/RelationHooks/RelationJoinedHook

diagram

RelationJoinedHook

HookType

This element encapsulates the set of actions that has to be performed when a new relationship is discovered and established.

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

- name="RelationJoinedHook" type="HookType">

element MicroServiceType/RelationHooks/RelationChangedHook

diagram

RelationChangedHook

HookType

This element encapsulates the set of actions that has to be performed when an established relationship is affected by a change.

properties

- content: complex

children

- HookAction

annotation
documentation

This element represents a specific hook that is executed by an ARCADIA Agent.
This element encapsulates the set of actions that has to be performed when an established relationship is affected by a change.

```xml
<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>
```

This element encapsulates the set of actions that has to be performed when an established relationship is going to be gracefully terminated.

```xml
<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>
```

This element encapsulates the set of actions that has to be performed when an established relationship is going to be abnormally terminated.

```xml
<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>
```
**D2.2 - Definition of the ARCADIA Context Model**

### 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
```xml
<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**

- **QoSActions**
  This element represents the set of actions that are binded to specific Measurable Metrics.

- **CustomActions**
  This element encapsulates a set of custom actions that can be invoked by the ARCADIA Smart Controller.

### properties
- minOccurs 0
- maxOccurs 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
```xml
<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:complexContent>
            <xs:extension base="ActionType">  
              ...
            </xs:extension>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
      <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:extension>
          </xs:complexContent>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```
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.

This element encapsulates a set of custom actions that can be invoked by the ARCADIA Smart Controller.

This element represents one atomic action that can be invoked.

element MicroServiceType/ActionDescriptor/QoSActions

diagram

| properties | minOcc | 0 |
|           | maxOcc | 1 |
| content   | complex |

children QoSAction
<table>
<thead>
<tr>
<th>annotation</th>
<th>documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>This element represents the set of actions that are bound to specific Measurable Metrics.</td>
<td></td>
</tr>
</tbody>
</table>

```xml
<xs:element name="QoSActions" minOccurs="0">
  <xs:annotation>
    <xs:documentation>This element represents the set of actions that are bound 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:element>
              </xs:complexContent>
            </xs:extension>
          </xs:complexType>
        </xs:sequence>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```
This element represents a specific QoS action.

```
<xs:complexType name="QoSAction">
  <xs:annotation>
    <xs:documentation>This element represents a specific QoS action.</xs:documentation>
  </xs:annotation>
  <xs:complexContent base="ActionType">
    <xs:sequence>
      <xs:element name="AffectedByMetric" base="MeasureableMetricType"/>
    </xs:sequence>
  </xs:complexContent>
</xs:complexType>
```

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.
element MicroServiceType/ActionDescriptor/QoSActions/QoSAction/AffectedByMetric

Diagram:

- **AffectedByMetric**
  - **MetricIdentifier**
    - This element represents an identifier that differentiates Metric instances.
  - **MetricName**
    - This element represents the descriptive name of the metric (e.g., HTTPResponseTime).
  - **MeasurementUnit**
    - This element represents the unit of measurement that is used for a specific MeasurableMetric (e.g., seconds).

Properties:
- 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:
```xml
<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:

- **CustomActions**
  - **Action**
    - 1..* This element represents one atomic action that can be invoked.

This element encapsulates a set of custom actions that can be invoked by the ARCADIA Smart Controller.
This element encapsulates a set of custom actions that can be invoked by the ARCADIA Smart Controller.

```xml
<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**

- **ActionType**
  - **ActionName**
  - **ActionDescription**

- **Type** Action
- **Properties**
  - minOcc 1
  - maxOcc unbounded
  - content complex
- **Children** ActionName ActionDescription
- **Annotation**
  - This element represents one atomic action that can be invoked.

```xml
<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>
```
This element encapsulates the measurable metrics that relate to a specific link/relationship.

This element refers to a specific quantitative monitoring metric which is a performance indicator for a specific relationship/link.

| properties | minOcc | 0 |
|           | maxOcc | 1 |
| content   | complex |

This element encapsulates the measurable metrics that relate to a specific link/relationship.

This element refers to a specific quantitative monitoring metric which is a performance indicator for a specific relationship/link.
element MicroServiceType/LinkMonitoringDescriptor/MeasurableMetric

- **Diagram**
  - **MeasureableMetricType**
    - **MetricIdentifier**
    - **MetricName**
    - **MeasurementUnit**
  - This element represents a specific, quantitative monitoring metric which is a performance indicator for a specific relationship/link.

- **Type**
  - **MeasureableMetricType**

- **Properties**
  - minOcc: 1
  - maxOcc: unbounded
  - content: complex

- **Children**
  - MetricIdentifier, MetricName, MeasurementUnit

- **Annotation**
  - This element refers to a specific quantitative monitoring metric which is a performance indicator for a specific relationship/link.

- **Source**
  ```xml
  <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**
  - **LinkActionDescriptor**
  - **QoSActions**
  - This element encapsulates the set of actions that relate to the QoS of a link/relationship.

- **Properties**
  - minOcc: 0
  - maxOcc: 1
  - content: complex

- **Children**
  - QoSActions

- **Annotation**
  - This element encapsulates the actions that relate to the QoS of a link/relationship.

- **Source**
  ```xml
  <xs:element name="LinkActionDescriptor" minOccurs="0"/>
  ```
This element encapsulates the actions that relate to the QoS of a link/relationship.

This element represents the set of actions that are binded to specific Measurable Metrics of the link/relationship.

This element represents one atomic action that can be invoked.

Will define the Triggering Event when implemented.

This element represents one atomic action that can be invoked.

minOcc = 0
maxOcc: 1  
  content: complex

<table>
<thead>
<tr>
<th>children</th>
<th>QoSAction</th>
</tr>
</thead>
<tbody>
<tr>
<td>annotation</td>
<td>documentation</td>
</tr>
</tbody>
</table>

This element represents the set of actions that are binded to specific Measurable Metrics of the link/relationship.

```xml
<xs:element name="QoSActions" minOccurs="0">
  <xs:documentation>
    This element represents the set of actions that are binded to specific Measurable Metrics of the link/relationship.
  </xs:documentation>
  <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 base="MeasureableMetricType"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```
element MicroServiceType/LinkActionDescriptor/QoSActions/QoSAction

This element represents one atomic action that can be invoked.

properties
- minOcc: 1
- maxOcc: unbounded
- content: complex

children: ActionName, ActionDescription, AffectedByMetric

annotation documentation
This element represents one atomic action that can be invoked.

source
```xml
<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:element>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:element>
```
element MicroServiceType/LinkActionDescriptor/QoSActions/QoSAction/AffectedByMetric
diagram

MeasureableMetricType (extension)

 affected Metric

MetricIdentifier

This element represents an identifier that differentiates Metric instances.

MetricName

This element represents the descriptive name of the metric (e.g., I-TTPResponseTime).

MeasurementUnit

This element represents the unit of measurement that is used for a specific MeasureableMetric (e.g., second).

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.

```xml
<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>
</source>
```

**simpleType ExecutionLanguageType**

- **type**: restriction of `xs:string`
- **properties**: base `xs:string`
- **used by**: element `HookType/HookAction/ExecutionLanguage`

<table>
<thead>
<tr>
<th>Kind</th>
<th>Value</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>enumeration</td>
<td>BASH</td>
<td></td>
</tr>
<tr>
<td>enumeration</td>
<td>JAVA</td>
<td></td>
</tr>
<tr>
<td>enumeration</td>
<td>PYTHON</td>
<td></td>
</tr>
<tr>
<td>enumeration</td>
<td>RUBY</td>
<td></td>
</tr>
</tbody>
</table>

**annotation documentation**

This element represents the enumeration of the supported execution languages.

```xml
<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>
</source>
```

**simpleType MeasurementUnit**

- **type**: restriction of `xs:string`
- **properties**: base `xs:string`
- **used by**: element `MeasureableMetricType/MeasurementUnit`

<table>
<thead>
<tr>
<th>Kind</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>enumeration</td>
<td>HTTP Requests / second</td>
</tr>
</tbody>
</table>
This element represents the measurable unit of one quantitative metric.

```
<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>
```

This element represents the enumeration of the supported processor architectures.

```
<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>
```

This element represents the enumeration of the predefined MicroService categories.

```
<xs:simpleType name="ServiceCategoryType">
  <xs:annotation>
    <xs:documentation>
      This element represents the enumeration of the predefined MicroService categories.
    </xs:documentation>
  </xs:annotation>
</xs:simpleType>
```
categories.</xs:documentation>
</xs:annotation>
<xs:restriction base="xs:string">
  <xs:pattern value="Database"/>
</xs:restriction>
</xs:simpleType>