Developing Data-Intensive Cloud Applications with Iterative Quality Enhancements

Architecture definition and integration plan - Final version

Deliverable 1.4 Companion

DICE partners

ATC:	Athens Technology Centre
FLEXI:	Flexiant Limited
IEAT:	Institutul E Austria Timisoara
IMP:	Imperial College of Science, Technology & Medicine
NETF:	Netfective Technology SA
PMI:	Politecnico di Milano
PRO:	Prodevelop SL
XLAB:	XLAB razvoj programske opreme in svetovanje d.o.o.
ZAR:	Unversidad De Zaragoza

The DICE project (February 2015-January 2018) has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 644869

Table of contents

List of tables

1. Introduction

This document contains a snapshot of the DICE requirements at the end of Y2. It contains only the technical requirements, i.e., the detailed requirements from WP1-WP5. The contents of this document replace the ones from Requirement Specification M16.

2. Technical requirements

2.1. WP1 Requirements

Table 1: The Stereotyping of UML diagrams with DICE profile Requirement.

ID:	R1.1
Title:	Stereotyping of UML diagrams with DICE profile
Priority of accomplishment:	Must have
Type:	Requirement
Description:	Open-source modelling tool with XMI and UML2.X (2.4 or 2.5) support
Rationale:	Support quality-related decision-making
Supporting material:	N/A
Other comments:	Stereotypes of the DICE profile will be applied in Papyrus UML models

Table 2: The Guides through the DICE methodology Requirement.

ID:	R1.2
Title:	Guides through the DICE methodology
Priority of accomplishment:	Must have
Type:	Requirement
Description:	An action to open an external website with the guide or document of the DICE Methodology
Rationale:	The DICE IDE will guide the developer through the DICE methodology
Supporting material:	N/A
Other comments:	We proposed to use EPF plugins to modelate the methodology. Then you can generate a website with this methodology, and this website could be referenced in the IDE

Table 3: The Quality testing tools IDE integration Requirement.

ID:	R1.6
Title:	Quality testing tools IDE integration
Priority of accomplishment:	Could have

Type:	Requirement
Description:	The IDE COULD provide the means to configure the QTESTING_TOOLS execution
Rationale:	Quality tests may come with parameters such as the number of tests to run or the duration of each tests, which the user should be able to change.
Supporting material:	N/A
Other comments:	Decreased in M24 from Should have to Could have because the functionality expected is likely to be doable through Jenkins and other means

Table 4: The Continuous integration tools IDE integration Requirement.

ID:	R1.7
Title:	Continuous integration tools IDE integration
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The CI_TOOLS MUST be integrated with the IDE.
Rationale:	The continuous integration tools must provide the means to be invoked remotely, with an option of controls and status display built into the IDE.
Supporting material:	N/A
Other comments:	A plugin to connect Eclipse with Jenkins will be provided on the IDE. This plugin allows to execute Continuous Integration (e.g., Jenkins) Tasks from Eclipse. Configuration should be done on Jenkins. This plugin allows to execute them from Eclipse, and see the results from there

Table 5: The Running tests from IDE without committing to VCS Requirement.

ID:	R1.7.1
Title:	Running tests from IDE without committing to VCS
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The CI_TOOLS COULD provide an integration with the IDE that enables deployment and execution of tests on the user's local changes without committing the code into the VCS.

Rationale:	In some cases the DEVELOPER may want to run a test without committing the code into the repository.
Supporting material:	N/A
Other comments:	N/A

Table 6: The Mapping between VCS version and deployment in IDE Requirement.

ID:	R5IDE2
Title:	Mapping between VCS version and deployment in IDE
Priority of accomplishment:	Must hav
Type:	Requirement
Description:	The DEPLOYMENT_TOOLS MUST provide in the IDE a possibility to know, which version of the application was deployed under which deployment (application) ID
Rationale:	The ENHANCEMENT_TOOLS analyse past runtime data given a version of the application in the runtime. The DEPLOYMENT_TOOLS in the IDE MUST provide a history of versions and their associated application IDs used in MONITORING_TOOLS.
Supporting material:	N/A
Other comments:	N/A

Table 7: The IDE support to the use of profile Requirement.

ID:	R2IDE.1
Title:	IDE support to the use of profile
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE IDE MUST support the development of DIA exploiting the DICE profile and following the DICE methodology. This means that it should offer widzards to guide the developer through the steps envisioned in the DICE methodology
Rationale:	An adoption of the DICE profile not supported by a user friendly IDE can be quite cumbersome and limit the benefits of our approach. The more the IDE is user friendly the more the potential of a positive impact of the DICE profile on practitioners increases

Supporting material:	N/A
Other comments:	Related to R1.2

Table 8: The Metric selection Requirement.

ID:	R3IDE.1
Title:	Metric selection
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE IDE MUST allow to select the metric to compute from those defined in the DPIM/DTSM DICE annotated UML model. There are efficiency and reliability related metrics
Rationale:	N/A
Supporting material:	The metrics supported will be all those defined in WP2. Examples of them are Throughput or response time when talking about performance; or MTTF o MTBF, and so on regarding reliability
Other comments:	UI from WP3 DICE tools integrated to DICE IDE

Table 9: The Timeout specification Requirement.

ID:	R3IDE.2
Title:	Timeout specification
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The IDE SHOULD allow the user to set a timeout and a maximum amount of memory (2) to be used when running the SIMULATION_TOOLS and the VERIFICATION_TOOLS. Then, when the timeout expires or when the memory limit is exceeded, the IDE SHOULS notify the user
Rationale:	N/A
Supporting material:	(2) The timeout should be set by the user considering the hardware configuration and the space of the model
Other comments:	UI from WP3 DICE tools integrated to DICE IDE

Table 10: The Usability Requirement.

ID:	R3IDE.3
Title:	Usability
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The TRANSFORMATION_TOOLS and SIMULATION_TOOLS MAY follow some usability, ergonomics or accesibility standard such as ISO/TR 16982:2002, ISO 9241, WAI W3C or similar
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 11: The Loading the annotated UML model Requirement.

ID:	R3IDE.4
Title:	Loading the annotated UML model
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE IDE MUST include plugins to launch the SIMULATION_TOOLS and VERIFICATION_TOOLS for a DICE UML model that is loaded in the IDE
Rationale:	The verification phase is launched from the DICE IDE, it is not meant to be independent, even though it involves launching an external tool (see R3.9.1).
Supporting material:	N/A
Other comments:	IDE will allow to execute external tools providing as a parameter the desired annotated UML model. A Papyrus UML model can be annotated with EAnnotation (from Ecore) in order to extend the Metamodel properties.

Table 12: The Usability of the IDE-VERIFICATION_TOOLS interaction Requirement.

ID:	R3IDE.4.1
Title:	Usability of the IDE-VERIFICATION_TOOLS interaction

Priority of accomplishment:	Should have
Type:	Requirement
Description:	The QA_ENGINEER SHOULD not perceive a difference between the IDE and the VERIFICATION_TOOL; it SHOULD be possible to seamlessly invoke the latter from the former
Rationale:	In a sense the IDE and the VERFICATION_TOOLS reside in a sort of continuum, where the former invokes the latter, but the user should not feel the difference in the environment
Supporting material:	N/A
Other comments:	N/A

Table 13: The Loading of the property to be verified Requirement.

ID:	R3IDE.4.2
Title:	Loading of the property to be verified
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The VERIFICATION_TOOLS MUST be able to handle the verification of the properties to be checked that can be defined through the IDE and the DICE profile
Rationale:	The properties to be checked are defined in the DICE UML model (possibly using templates). The requirement on the VERIFICATION_TOOLS is to be able to handle them.
Supporting material:	N/A
Other comments:	Properties to be verified can be listed in a custom model understandable by the VERIFICATION_TOOLS, where all the properties to be verified can be listed there. Both this model and the UML model will be used as input for the verification tools

Table 14: The Graphical output Requirement.

ID:	R3IDE.5
Title:	Graphical output
Priority of accomplishment:	Should have

Type:	Requirement
Description:	Whenever needed (for better understanding of the response), the IDE SHOULD be able to take the output generated by the VERIFICATION_TOOLS (i.e., execution traces of the modeled system) and represent it graphically, connecting it to the elements of the mod
Rationale:	The output of the VERIFICATION_TOOLS (i.e., traces of the modeled system) should be presented in a user-friendly way to help the user better understand the outcome of the verification task.
Supporting material:	N/A
Other comments:	One way to do that is to create a metamodel that supports to define all the traces and relates them to an element from the UML model. The easiest way is to annotate the Papyrus UML model with EAnnotations (from Ecore) and, programmatically, colorate elements if desired. Also the traces (a string) can be added as annotation and show it within a popup or similar.

Table 15: The Graphical output of erroneous behaviors Requirement.

ID:	R3IDE.5.1
Title:	Graphical output of erroneous behaviors
Priority of accomplishment:	Could have
Type:	Requirement
Description:	In case the outcome of the verification task is "the property does not hold", the VERIFICATION_TOOLS COULD provide, in addition to the raw execution trace of the system that violates the desired property, an indication of where in the trace lies the probl
Rationale:	In case of a property not holding, the VERIFICATION_TOOLS return a trace of the system model that violates the property. Understanding *why* the property is violated (e.g., which part of the trace is the one where the property is violated) is not always an easy task. The output of the VERIFICATION_TOOLS might help in this regard, by highlighting where the problem lies.
Supporting material:	N/A
Other comments:	One way to do that is to create a metamodel that supports to define all the traces and relates them to an element from the UML model. The easiest way is to annotate the Papyrus UML model with EAnnotations (from Ecore) and,

	programmatically, colorate elements if desired. Also the traces (a string) can be added as annotation and show it within a popup or similar.

Table 16: The Loading a DDSM level model Requirement.

ID:	R3IDE.6
Title:	Loading a DDSM level model
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The OPTIMIZATION_TOOLS as part of the IDE MUST provide an interface to load (not design) a DDSM DICE annotated model
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 17: The Output results of simulation in user-friendly format Requirement.

ID:	R3IDE.7
Title:	Output results of simulation in user-friendly format
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The IDE COULD allow the user to see the multiple output results of the SIMULATION_TOOLS in user-friendly format.
Rationale:	When the user has selected many different combinations of scerario properties to simulate, the SIMULATION_TOOLS provide results for each combination. These results should be presented in user-friendly format, to relieve the user task of opening different models to obtain the result of each combination.
Supporting material:	N/A
Other comments:	An option that offers the user to automatically plot results of different combinations in a single graph -without requiring opening each model- will accomplish this requirement

Table 18: The Resource consumption breakdown Requirement.

ID:	R4IDE1
Title:	Resource consumption breakdown
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DEVELOPER MUST be able to see via the ENHANCEMENT_TOOLS the resource consumption breakdown into its atomic components.
Rationale:	Existence of different abstraction levels between design concepts (e.g., abstractions in the DICE profile) and runtime measurements hides the details on what high-level request effectively generated the request for data.
Supporting material:	R4.11
Other comments:	N/A

Table 19: The Bottleneck Identification Requirement.

ID:	R4IDE2
Title:	Bottleneck Identification
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The ENHANCEMENT_TOOLS MUST indicate which classes of requests represent bottlenecks for the application in a given deployment.
Rationale:	N/A
Supporting material:	R4.12
Other comments:	N/A

Table 20: The Model parameter uncertainties Requirement.

ID:	R4IDE3
Title:	Model parameter uncertainties
Priority of accomplishment:	Could have

Type:	Requirement
Description:	The REQ_ENGINEER COULD express uncertainty on some performance/reliability input parameters (e.g., execution times) in the DICE profile by means of a prior distribution or an interval. The ENHANCEMENT_TOOLS COULD take into account these parameters to esti
Rationale:	DoW mentions Bayesian estimation techniques. These techniques can explicitly account for the uncertainty provided by the REQ_ENGINEER.
Supporting material:	R4.20
Other comments:	This requirement may be alternatively stated as part of WP2 or WP3, since it also affects the DICE profile. The requirement would expand the scientific impact of the tool, but if too complex to implement it might be ignored without major consequences.

Table 21: The Model parameter confidence intervals Requirement.

ID:	R4IDE4
Title:	Model parameter confidence intervals
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The ENHANCEMENT_TOOLS COULD return confidence intervals for each inferred parameter of the performance and reliability models.
Rationale:	The WP3 models require to provide a number of parameters, such as CPU speeds. These will be inferred by the ENHANCEMENT_TOOLS of WP4 from the monitoring data. However, the estimation is subject to uncertainties so confidence intervals could be provided to the WP3 tools to quantify such uncertainty. If the CI is too wide, we might issue a warning in SIMULATION_TOOLS that the prediction is not robust.
Supporting material:	R4.21
Other comments:	N/A

Table 22: The Visualization of analysis results Requirement.

ID:	R4IDE5
Title:	Visualization of analysis results

Priority of accomplishment:	Could have
Type:	Requirement
Description:	ENHANCEMENT_TOOLS SHOULD be capable of visualizing analysis results
Rationale:	N/A
Supporting material:	R4.25
Other comments:	One way to do that is to create a metamodel that supports to define all the traces and relates them to an element from the UML model. The easiest way is to annotate the Papyrus UML model with EAnnotations (from Ecore) and, programmatically, colorate elements if desired. Also the traces (an string) can be added as annotation and show it within a popup or similar.

Table 23: The Safety and privacy properties loading Requirement.

ID:	R4IDE6
Title:	Safety and privacy properties loading
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The ANOMALY_TRACE_TOOLS MUST allow the DEVELOPER/ARCHITECT to choose and load the safety and privacy properties from the Model of the application described through the DICE profile
Rationale:	The properties to be analyzed are application-dependent, and they must come from somewhere in the DICE model of the application. The user knows what properties are to be monitored, so he/she should select those that most interest him/her
Supporting material:	R4.28
Other comments:	A wizard where properties to be analyzed can be selected before launching the external tool. So the configuration model and the UML model will be passed as input to these tools

Table 24: The Feedback from safety and privacy properties monitoring to UML models concerning violatedtime bounds Requirement.

ID:	R4IDE7

Title:	Feedback from safety and privacy properties monitoring to UML models concerning violated time bounds
Priority of accomplishment:	Could have
Type:	Requirement
Description:	In the feedback provided by the ANOMALY_TRACE_TOOLS to the DEVELOPER/ARCHITECT, the tools COULD highlight when a timing requirement is violated, and what is the value of the violation
Rationale:	The specific feedback about timing violations might help the DEVELOPER/ARCHITECT adjust the parameters of the models/properties
Supporting material:	R4.31.1
Other comments:	N/A

Table 25: The Relation between ANOMALY_TRACE_TOOLS and IDE Requirement.

ID:	R4IDE8
Title:	Relation between ANOMALY_TRACE_TOOLS and IDE
Priority of accomplishment:	Should have
Type:	Requirement
Description:	It SHOULD be possible to launch the ANOMALY_TRACE_TOOLS from the IDE
Rationale:	The idea is that the trace checking is performed starting from the elements that are described in the DICE UML model (see requirement R4.32). Hence, it makes sense that the tool is invoked from the UML IDE. The idea could be that the IDE has a link to the DW, and when the user asks for performing trace checking, the IDE queries the DW, retrieves the information for the trace checking, then feeds the ANOMALY_TRACE_TOOLS with the traces to be checked.
Supporting material:	R4.33
Other comments:	N/A

2.2. WP2 Requirements

Table 26: The Profile Structure Requirement.

ID:	PR2.0
Title:	Profile Structure
Priority of accomplishment:	Must have
Type:	Requirement
Description:	Following the basic approaches to formal languages design, the DICE profile will necessarily require a meta-modelling notation to cover for the basic structure and semantics of the language intended behind the DICE profile. Also, the DICE profile will need the implementation of said basic structure and semantics following a commonly usable format as best fit with respect to DICE goals and tenets.
Rationale:	formal lanugages specification requires both abstract and concrete syntax for a language to be well-formed.
Supporting material:	http://www.igi-global.com/chapter/design-of-formal-languages-and-interfaces/87050
Other comments:	in the scope of this document, Requirements ID follow a naming pattern that reflects the nature behind said requirements. More in particular: (a) IDs starting with PR.xx indicate Profile Requirements; (b) IDs starting with MR.xx indicate Methodology Requirements; (c) IDs starting with PRD.xx indicate Profile Requirements specific for Deployment modelling

Table 27: The Profile Basis Requirement.

ID:	PR2.1
Title:	Profile Basis
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE profile MUST follow the default abstraction layers known and supported in Model-Driven Engineering, namely, Platform-Independent Model, Platform-Specific Model and add an additional layer specific to supporting the modelling of Deployment-ready implementations, i.e., a Deployment-Specific Model.
Rationale:	UML is the de-facto standard for industrial-strength modelling and the basis behind Model-Driven Engineering. It is therefore natural that DICE shall inherit abstraction layers

	from MDE as much as it shall inherit conceptual foundations from UML (e.g., classes, associations, their relation, their configuration, etc.). In addition however, the DICE profile's novelty lies mainly in its unique support to development of deployment-ready applications. Hence, a new abstraction layer shall be explicitly supported with specific models addressing it.
Supporting material:	http://www.omg.org/mda/specs.htm
Other comments:	N/A

Table 28: The Abstraction Layer Origin Requirement.

ID:	PR2.2
Title:	Abstraction Layer Origin
Priority of accomplishment:	Must have
Type:	Requirement
Description:	Every abstraction layer (namely, DPIM, DTSM and DDSM) of the DICE profile MUST stem from UML.
Rationale:	The DICE profile shall mimic the standard assumptions behind Model-Driven Engineering, including the separation of concerns across three disjoint but related layers (Platform-Independent, Platform-Specific and Deployment-Specific).
Supporting material:	http://www.omg.org/spec/UML/
Other comments:	

Table 29: The Relation with MARTE UML Profile Requirement.

ID:	PR2.3
Title:	Relation with MARTE UML Profile
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE Profile MUST define required and provided properties of a DIA as well as metrics (estimated, measured, calculated and requirements) to monitor them. Said metrics will be specifed following the MARTE NFP framework.
Rationale:	MARTE provides valuable foundations for specifying non-functional properties and shall be considered for extension

Supporting material:	http://www.omgmarte.org/
Other comments:	N/A

Table 30: The Constraints Definition Requirement.

ID:	PR2.4
Title:	Constraints Definition
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE Profile shall allow definition of values of constraints (e.g., maximum cost for the DIA), properties (e.g., outgoing flow from a Storage Node) and stereotype attributes (batch and speed DIA elements) using the MARTE VSL standard.
Rationale:	VSL is a part of the MARTE standard dedicated specifically to the (semi-)formal specification of quality attribute values across profiles for qauality properties definition and their analysis. DICE shall make use of these modelling facilities inherited form MARTE
Supporting material:	http://www.omg.org/omgmarte/Documents/tutorial/part2.pdf
Other comments:	N/A

Table 31: The DICE Profile Performance Annotations Requirement.

ID:	PR2.5
Title:	DICE Profile Performance Annotations
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE Profile shall define annotations for performance based on the MARTE::GQAM framework.
Rationale:	Relevant part inherited from MARTE for the specifcations of performance values.
Supporting material:	N/A
Other comments:	N/A

Table 32: The DICE Profile Reliability Annotations Requirement.

ID:	PR2.6
Title:	DICE Profile Reliability Annotations
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE Profile shall define annotations for reliability based on the DAM profile.
Rationale:	DAM is a profile designed to extend MARTE in support of reliability, and therefore shall be considered within DICE and the profile specification.
Supporting material:	N/A
Other comments:	N/A

Table 33: The DICE Profile Main DIA Concerns - Structure and Topology Requirement.

ID:	PR2.7
Title:	DICE Profile Main DIA Concerns - Structure and Topology
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE Profile shall define annotations that address structural and topological concerns behind DIAs. Also, the DICE Profile shall separately define storage and computation elements to allow for fine-grained specification.
Rationale:	Data-Intensive Application (DIA) elements are often designed and thought out as a topology of constructs operating under a prescribed behavior.
Supporting material:	N/A
Other comments:	N/A

Table 34: The DICE Profile Main DIA Concerns - Flow and Behavior Requirement.

ID:	PR2.8
Title:	DICE Profile Main DIA Concerns - Flow and Behavior
Priority of accomplishment:	Must have
Type:	Requirement

Description:	The DICE Profile shall define annotations that address behavioral and flow concerns behind DIAs. Also, the DICE Profile shall define annotations for flow-control across DIAs.
Rationale:	Many of the characteristics behind DIAs are sensibly influenced by the flow of information, its management and the application's behavior in managing and handling data. These aspects shall be made explicit for DICE-supported analysis.
Supporting material:	N/A
Other comments:	N/A

Table 35: The DICE Profile Pre- and Post-Processing Requirement.

ID:	PR2.9
Title:	DICE Profile Pre- and Post-Processing
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE Profile shall define constructs for pre- and post-processing of Big Data (e.g., for filtering input data or visualising data).
Rationale:	Many DIAs are structured using filters that, e.g., aggregate and decompose data before processing or post-process data for the purpose of visualization. Said components are themselves Data-intensive and shall be explicitly supported in the DICE profile.
Supporting material:	N/A
Other comments:	N/A

Table 36: The DICE Profile Tech-Specific Constraints Requirement.

ID:	PR2.10
Title:	DICE Profile Tech-Specific Constraints
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE Profile shall define structural and behavioral constraints typical in targeted technologies (e.g., Hadoop, Storm, Spark, etc.).

Rationale:	many technologies have different possible structural or behavioral concerns and consequent constraints. These must be explicitly supported across the DICE profile.
Supporting material:	N/A
Other comments:	N/A

Table 37: The DICE Profile Separation-of-Concerns Requirement.

ID:	PR2.11
Title:	DICE Profile Separation-of-Concerns
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE Profile shall use packages to separately tackle the description of targeted technologies in the respective profile abstraction layers (e.g., DTSM and DDSM). Said packages shall be maintained consistent.
Rationale:	Separation of concerns is one of the basic principles behind model-driven engineering and related technologies.
Supporting material:	N/A
Other comments:	N/A

Table 38: The DICE Profile Supervision and Control Requirement.

ID:	PR2.12a
Title:	DICE Profile Supervision and Control
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE Profile shall define constructs and annotations for DIA supervision and process control.
Rationale:	a big part of the needs behind DIAs is reflected in how resources are managed, supervised and allocated. DICE addresses not only the monitoring concerns behind said statement but also it shall offer constructs that allow planning and analysis of supervision and control mechanisms at design time.
Supporting material:	N/A

Other comments:	N/A

Table 39: The DICE Privacy & Security Aspects Requirement.

ID:	PR2.12b
Title:	DICE Privacy & Security Aspects
Priority of accomplishment:	Must have
Type:	Domain Assumption
Description:	The DICE Profile shall focus on DIA-specific privacy and/or security restrictions.
Rationale:	We restrict the privacy and security policies to be concerned explicitly about the DIA itself rather than the circumstantial technology with which the DIA is developed, operated and evolved. For example, restricting the behaviour of the monitoring platform on top of the privacy-sensitive DIA or reducing monitoring operations in any way due to privacy concerns is out of the scope of the support intended in DICE.
Supporting material:	Delivery of D2.4
Other comments:	N/A

Table 40: The DICE Profile Data Structure Requirement.

ID:	PR2.13
Title:	DICE Profile Data Structure
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE Profile shall define QoS annotations for data structure and its specification.
Rationale:	Data-Structure is a big concern in Data-Intensive Applications. Also, said concern must be explicitly supported with ad-hoc constructs such that its relations with DIAs is properly analysed and supported at Design time.
Supporting material:	N/A
Other comments:	N/A

Table 41: The DICE Profile Data Communication Requirement.

ID:	PR2.14
Title:	DICE Profile Data Communication
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE Profile shall define annotations to elaborate on structural and behavioral details concerning the channeling and marshalling of information across specified DIAs.
Rationale:	the flow of information across a DIA, e.g., for further processing or visualization shall be supported at both structural (i.e., nodes involved) and behavioral (i.e., behavior of said nodes) level. Thsi is because data flow and manipulation of data can vary sensibly depending on the kind of DIA being designed (e.g., for the purpose of analysing streaming data).
Supporting material:	N/A
Other comments:	N/A

Table 42: The DICE Profile Sub-Structures Requirement.

ID:	PR2.15
Title:	DICE Profile Sub-Structures
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE Profile shall provide annotations for specifying node nesting and replication across the structure of DIAs.
Rationale:	DIAs often are requried to be designed as nested applications. For example, compute nodes may hide internal logic from multiple possible technological specification within them. Therefore, the ability to support nesting and sub-structure across DIAs shall be supported.
Supporting material:	N/A
Other comments:	N/A

Table 43: The DICE Analysis Focus Requirement.

ID:	PR2.16
Title:	DICE Analysis Focus

Priority of accomplishment:	Must have
Type:	Domain Assumption
Description:	The DICE profile and its design shall work under the assumption that their focus of application is limited to providing facilities and methdological approaches to support those properties that are relevant to perform analysis (e.g., for fine-tuning, load-estimation, etc.), testing (e.g., for run-time verification and adaptation towards continuous integration), monitoring (e.g., for flexible continuous improvement, etc.).
Rationale:	being an emerging field, DIAs design and analysis may entail a great variety of possible analyses and venues for research and development. Our assumption however, is that DIAs are either modelled to analyse and estimate their properties, test these estimations in practice or monitor their actioned behavior for continuous improvement. Other endeavours, however connected to DIAs, are out of the scope of DICE.
Supporting material:	https://docs.google.com/presentation/d/1aAeoGJox42pHBpmLCDDhwGtmb-J7RmzFobqm-QB7tV8/edit#slide=id.gb6c695009_2_115
Other comments:	N/A

Table 44: The DICE Transformations Focus Requirement.

ID:	PR2.17
Title:	DICE Transformations Focus
Priority of accomplishment:	Must have
Type:	Domain Assumption
Description:	There are many possible transformations that can be covered by the DICE profile in terms of constructs that support said transformations. However, we assume that many such transformations are blatant methodological issue to be reflected in how models are constructed, evolved after analysis or refined in place. DICE methodological abstractions and procedures will cover whatever in-place refinement is required at every abstraction level whereas technlogy-supported transformations can focus on reducing the abstraction by means of automation. For example, the seamless application of refinements to the same DTSM model is a methodological issue while the creation of a TOSCA blueprint from a DDSM model is not. DICE assumes that the latter shall be supported by ad-hoc M2M and M2T transformations while the former can be specified as part of a methodological approach part of DICE.

Rationale:	This assumption covers the differentiation between what shall be considered manual transformation and what automation DICE can offer to designers in their DIA Architecting endeavours. The assumpion is justified by the fact that we need to distinguish between methodological approaches part of DICE and actual technologies which support concrete transformations. Following this assumption, a series of transformation requirements are stated stemming from online tutorials into big data applications design and analysis.
Supporting material:	http://www2.informatik.hu-berlin.de/~scheidge/downloads/MBD06ScheidgenModelPattern.pdf
Other comments:	N/A

Table 45: The DICE Deployment Transformation Requirement.

ID:	PR2.18
Title:	DICE Deployment Transformation
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE IDE needs to be provided with a fully automated transformation that is capable of constructing an ad-hoc TOSCA blueprint stemming from the deployment information that can be made available in a DTSM and DDSM model. The usage of deployment knowledge for each technology in the DTSM shall be used by such transformation as a means to determine the deployment structure. Subsequently, a DDSM model proposal shall be built from this automated understanding. Finally, a TOSCA blueprint shall be constructed from such DDSM model using an appropriate mirroring between the DDSM model instance and the TOSCA notation.
Rationale:	this requirement covers the specification of appropriate deployment transformations that are required to generate TOSCA-ready blueprints out of DICE specifications.
Supporting material:	N/A
Other comments:	N/A

Table 46: The DICE Architecture Trade-Off Requirement.

ID:	PR2.19
Title:	DICE Architecture Trade-Off

Priority of accomplishment:	Must have
Type:	Domain Assumption
Description:	We assume tht a DIA architect is compelled to evaluate several equally valuable alternatives for technological composition of its own DIA solution. In so doing, said architect will evaluate the possible combinations of all technologies in a technological library (e.g., such as the one provided by DICE). From this library the architect will need to instantiate the possible compatible compositions of technologies that match its higher-order architectural specification (i.e., his DPIM model).
Rationale:	this assumption is reasonable since architects are often required to run trade-off or trade-space analysis techniques to brainstorm and reason on their own DIA design. This is true for any scenario in which several possible opions are available and rationale needs to be produced for every option to allow for comparative analysis.
Supporting material:	http://www.seaclouds-project.eu/content/continuous-architecting-stream-based-systems
Other comments:	N/A

Table 47: The DICE Architecture Trade-Off Transformation Requirement.

ID:	PR2.20
Title:	DICE Architecture Trade-Off Transformation
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The DICE IDE needs to be rigged with a M2M transformation that provides coherent and comparable aggregates of the elements in the DICE technological library such as to allow for architecture trade-off analysis specified in PR2.19.
Rationale:	this requirement is linked to the requirement of reducing the abstraction layer between the DPIM and DTSM by means of architecture trade-off analysis.
Supporting material:	N/A
Other comments:	N/A

Table 48: The DICE Architecture Transformation Focus Requirement.

ID:	PR2.21
Title:	DICE Architecture Transformation Focus
Priority of accomplishment:	Must have
Type:	Domain Assumption
Description:	The DICE transformation set is intended to be the entire set of transformations that lower or increse the level of abstraction with the purpose of allowing more detailed or general modelling for DIA solutions. Whatever transformation is not concentrated on producing modelling notations which are more abstract or more concrete than the ones in input (e.g., transformations that modify an in-place model for the purpose of analysis) is intended to be out of scope for the DICE Profile, DICE methodology and the underlying processes and meta-models.
Rationale:	The rationale for this assumption is that every analysis format will require its own in-place transformation which depends solely on the information to be produced for that tool and according to that tool's input format. Therefore, said transformation abstracts from the modelling notations, their meta-model or how they are produced and maintained. Rather, said transformations are ad-hoc in-place abstractions of any DICE modelling layer (DPIM to DTSM to DDSM) and threfore out of the scope intended in the DICE modelling IDE.
Supporting material:	N/N
Other comments:	N/A

Table 49: The DICE Agnostic Data Specification Requirement.

ID:	PR2.22
Title:	DICE Agnostic Data Specification
Priority of accomplishment:	Must have
Type:	Domain Assumption
Description:	
Rationale:	
Supporting material:	N/N
Other comments:	N/A

Table 50: The DICE Agnostic Data Integration Requirement.

ID:	PR2.23
Title:	DICE Agnostic Data Integration
Priority of accomplishment:	Should have
Type:	Requirement
Description:	
Rationale:	
Supporting material:	N/N
Other comments:	N/A

Table 51: The DICE Data Technology Diversity Requirement.

ID:	PR2.24
Title:	DICE Data Technology Diversity
Priority of accomplishment:	Should have
Type:	Requirement
Description:	
Rationale:	
Supporting material:	N/N
Other comments:	N/A

Table 52: The DICE Actionable Architecture Paradigm Requirement.

ID:	MR2.0
Title:	DICE Actionable Architecture Paradigm
Priority of accomplishment:	Must have
Type:	Domain Assumption
Description:	The DICE methodology shall focus on producing and supporting at least two views for a DIA architecture. First, a structural modelling view and Second, a behavioral modelling one. While for the first view a series of component, class, object and deployment structure diagrams are sufficient, for the second view, the DICE methodology shall strive to cover any behavioral notation which is functional to conducting QoS and QoD analyses intended within the DICE project. As a consequence, the methodological specification

	shall initially concentrate on agree where and how does the first structural view need to be aggregated and then the specification shall focus on eliciting which behavioral specification notation needs to be supported at methodological level.
Rationale:	this assumption covers what we learned from the basis digrammatic requirements emerged as part of the elaboration of the DICE profile in action. We observed that a number of diagrams are clearly to be used for structural representation purposes. Conversely, we also learned that a series of behavioral specifications are dependent on the means by which certain QoS and QoD properties will be specified (e.g., privacy) and supported by DICE. When these diagram requirements will become clear, then the methodological approach can cover for them as well.
Supporting material:	N/A
Other comments:	N/A

Table 53: The DICE Methodological Paradigm Requirement.

ID:	MR2.1
Title:	DICE Methodological Paradigm
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE profile and methodology shall support the incremental specification of Data-Intensive Applications (DIAs) following a Model-Driven Engineering approach, as defined in standard OMG guidelines.
Rationale:	The DICE profile and Methodology both follow the MDE paradigm and the models envisioned thereto.
Supporting material:	N/A
Other comments:	N/A

Table 54: The DICE Methodology support Diagrams Requirement.

ID:	MR2.2
Title:	DICE Methodology support Diagrams
Priority of accomplishment:	Should have

Type:	Domain Assumption
Description:	Every abstraction layer (namely, DPIM, DTSM and DDSM) of the DICE profile shall stem from UML, wherever possible.
Rationale:	several notations are being considered in the scope of DICE (e.g., MDA, MDE, MARTE, SecureML) - these notations already provide diagramming facilities that may be assumed as directly related to the needs and requirements of the DICE profile.
Supporting material:	N/A
Other comments:	N/A

Table 55: The DICE Design Process Requirement.

ID:	PR2.16
Title:	DICE Design Process
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE profile and methodology shall support the design of DIAs across three layers of abstractions: The DPIM, the DTSM and the DDSM, addressing platform-independent, technology-specific and deployment-specific details respectively.
Rationale:	Designing DIAs via the DICE profile shall also follow the MDE paradigm.
Supporting material:	N/A
Other comments:	N/A

Table 56: The DICE Profile Views Requirement.

ID:	MR2.3
Title:	DICE Profile Views
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE profile framework MUST envision that the designer obtains views using the DICE profile and following the methodology. Said views shall isolate separately all and only elements necessary to perform DICE quality

	evaluations. To this purpose, the DPIM shall elaborate on five (5) views with cross-cutting design concerns: (1) A Component View; (2) A State-Behavioral View; (3) A Sequence-Behavioral View; (4) A QoS Cross-Cutting View; (5) A Usage Cross-Cutting View;
Rationale:	the views in the requirement emerged from a preliminary analysis of concerns to be addressed at design time for DIAs.
Supporting material:	N/A
Other comments:	N/A

Table 57: The DICE Component View: this view allows designers to elaborate on the organizational structure ofthe components and possibly the responsible entities involved in the DIAinteractions for the purpose of realising the DIA's intended use; (4) A QoS Cross-Cutti Requirement.

ID:	MR2.3a
Title:	DICE Component View: this view allows designers to elaborate on the organizational structure of the components and possibly the responsible entities involved in the DIAinteractions for the purpose of realising the DIA's intended use; (4) A QoS Cross-Cutti
Priority of accomplishment:	Must have
Type:	Requirement
Description:	this view allows designers to elaborate on the organizational structure of the components and possibly the responsible entities involved in the DIA
Rationale:	the views in the requirement emerged from a preliminary analysis of concerns to be addressed at design time for DIAs.
Supporting material:	N/A
Other comments:	N/A

Table 58: The DICE State-Behavioral View Requirement.

ID:	MR2.3b
Title:	DICE State-Behavioral View
Priority of accomplishment:	Must have
Type:	Requirement
Description:	this view allows designers to elaborate on the internal components

	behavior rather than high-level components interactions across the DIA
Rationale:	the views in the requirement emerged from a preliminary analysis of concerns to be addressed at design time for DIAs.
Supporting material:	N/A
Other comments:	N/A

Table 59: The DICE Sequence-Behavioral View Requirement.

ID:	MR2.3c
Title:	DICE Sequence-Behavioral View
Priority of accomplishment:	Must have
Type:	Requirement
Description:	this view allows designers to elaborate on components interactions for the purpose of realising the DIA's intended use
Rationale:	the views in the requirement emerged from a preliminary analysis of concerns to be addressed at design time for DIAs.
Supporting material:	N/A
Other comments:	N/A

Table 60: The DICE QoS Cross-Cutting View Requirement.

ID:	MR2.3d
Title:	DICE QoS Cross-Cutting View
Priority of accomplishment:	Must have
Type:	Requirement
Description:	this view shall consist of cross-cutting annotations to elements in views "a", "b" and "c". The purpose of this view is to elaborate on the QoS constraints, limitations or requirements specified for annotated elements. The DICE profile shall focus on QoS alone. Therefore, elements not requiring any annotation shall not go in the DICE profile unless their presence determines a need for further analysis in the subsequent layers
Rationale:	the views in the requirement emerged from a preliminary analysis of concerns to be addressed at design time for DIAs.

Supporting material:	N/A
Other comments:	N/A

Table 61: The A Usage Cross-Cutting View; Requirement.

ID:	MR2.3e
Title:	A Usage Cross-Cutting View;
Priority of accomplishment:	Must have
Type:	Requirement
Description:	this view shall consist of cross-cutting annotations or graphical notations containing information related to the expected entrance load for the DIA and its composing elements.
Rationale:	the views in the requirement emerged from a preliminary analysis of concerns to be addressed at design time for DIAs.
Supporting material:	N/A
Other comments:	N/A

Table 62: The Data-Intensive QoS Requirement.

ID:	MR2.4
Title:	Data-Intensive QoS
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DPIM shall be generic enough so as not to require any specialization, e.g., for domain-specific DIAs. Conversely, the DPIM layer shall contain generic constructs with which to instantiate all possible DIAs together with all relevant QoS and Data-intensive analyses.
Rationale:	the first layer of abstraction of the DICE profile shall at least address the quality annotations as well as the safety & privacy characteristics (cfr. WP3) needed to further the design of a DIA in a QoS-Aware way.
Supporting material:	N/A
Other comments:	N/A

Table 63: The DICE DPIM Relations Requirement.

ID:	MR2.5
Title:	DICE DPIM Relations
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DPIM shall inherit notations and concepts from conceptual notations intended for similar purposes. For example, ModaCloudML offers modeling facilities to reason on cloud-based applications from multiple, functionally-complete perspectives (e.g., data, resources, etc.). Similarly, the UML-NIEM profile defines facilities to reason on information interchange at multiple layers (organizational, social, societal, etc.).
Rationale:	there exist a number of profiles that alaready (partially) cover the needs behind the DICE profile. Rather than reinventing new concepts, DICE may well inherit from said notations reusing where possible.
Supporting material:	N/A
Other comments:	N/A

Table 64: The DICE DPIM Concern - Data and I/O Logic Requirement.

ID:	MR2.6
Title:	DICE DPIM Concern - Data and I/O Logic
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The DPIM shall provide annotations to specify data-retrieval (i.e., where does the data come from and how is it transferred to its destination). Hence, I/O logic shall also be specified at the DPIM layer. Therefore, the DICE profile has to provide annotations for application requirements and topological specification starting from the very first level of specification.
Rationale:	the DPIM layer shall be conceived for requirements engineering of DIAs. In so doing, data and I/O shall be equally covered in the first layer of DIA abstraction.
Supporting material:	N/A
Other comments:	N/A

Table 65: The DICE Extension-Points Requirement.

ID:	MR2.7
Title:	DICE Extension-Points
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DTSM shall include extension facilities. These facilities shall be used to "augment" the DICE profile with technologies beyond the DICE project assumptions (e.g., Storm, Spark, Hadoop/MR, etc.). Similarly, every technological space embedded within the DICE profile shall exist in the form of such extensions, e.g., as conceptual packages (at the DTSM layer) and refined implementation-specific packages (at the DDSM layer).
Rationale:	because Big-Data Applications and their domain are extremely rich with technology and very highly evolving, the DICE profile shall define extension points where possible, i.e., points where further technologies may be specified and "plugged-in" within the profile itself.
Supporting material:	N/A
Other comments:	N/A

Table 66: The DICE Splits Requirement.

ID:	MR2.8
Title:	DICE Splits
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The DTSM layer shall support the definition and reasoning of "Splits", i.e., computable portions of data for the DIA at hand.
Rationale:	The DICE profile shall support the design of logically processable portions of information, i.e., "splits". This construct is technology-specific and is therefore needed starting from the DTSM layer. For example, if the designer is interested in knowing or manipulating/configuring the data processing policy he may want to vary the size, shape and processing for splits in his ad-hoc DIA.
Supporting material:	N/A
Other comments:	N/A

Table 67: The DICE Topologies Requirement.

ID:	MR2.9
Title:	DICE Topologies
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DTSM layer shall support the definition of Technology-specific DIA Topologies (e.g., Namenode-Datanode-SecondaryNamenode vs. Master-Region-Zookeeper, etc.).
Rationale:	similarly to other modelling technologies (e.g., TOSCA) DICE shall support the definition and design of DIA as topologies of connected services/components/nodes. Given that different technologies require different topologies, this concern is especially relevant at the DTSM layer and shall be supported as such.
Supporting material:	N/A
Other comments:	N/A

Table 68: The DICE Access Policies Requirement.

ID:	MR2.10
Title:	DICE Access Policies
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The DTSM layer shall support the definition of Access Policies, e.g., to data or to DIA frameworks.
Rationale:	normally a designer is also required to specify which access policies will be used across the DIAs. Given that different tchnologies require different access policies and related mechanisms, reasoning on Access policies shall take place initailly at the DTSM layer.
Supporting material:	N/A
Other comments:	N/A

Table 69: The DICE Functional Definition Requirement.

ID:	MR2.11
Title:	DICE Functional Definition
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DTSM layer shall support Technology-specific functions definition (Map-Reduce-Combine vs. Transformation-Action-Filter etc.).
Rationale:	The technological compound within DIAs consists of functional definitions which are specific for certain technologies. This means that functional specification for said technologies shall take place initially at the DTSM layer.
Supporting material:	N/A
Other comments:	N/A

Table 70: The DICE Deployment Specific Views Requirement.

ID:	MR2.12
Title:	DICE Deployment Specific Views
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DDSM layer shall support the definition of an Actionable deployment view (TOSCA-ready): this view offers conceptual mappings between the technological layer defined in the DTSM and concepts in the TOSCA metamodeling infrastructure such that one-way transformation between the technological layer and the actionable deployment view is possible.
Rationale:	because the instantiation for execution of different technologies may be optional and supported via TOSCA, the DDSM layer shall allow designers to use or not use the TOSCA-based deployment model for execution. This requirement assumes that further standards may be presented beyond TOSCA in the future.
Supporting material:	N/A
Other comments:	N/A

Table 71: The DICE Framework Overrides Requirement.

ID:	MR2.13
Title:	DICE Framework Overrides
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The DDSM layer shall support the definition of framework overrides. This allows designers to provide ad-hoc tweaks to framework settings based on specific constraints or design concerns.
Rationale:	many applications require ad-hoc configuration of the frameworks on which they are based. These tweaks are, by design, only allowed to change execution and deployment dynamics. Therefore, this ability shall be given to designers at the DDSM layer.
Supporting material:	N/A
Other comments:	N/A

Table 72: The DICE Resource Control Requirement.

ID:	MR2.14
Title:	DICE Resource Control
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DDSM layer shall support the management of VMs and similar resources as well as the necessary environmental setup connected to the application of specific frameworks (e.g., Hadoop/MapReduce).
Rationale:	many DIAs require fine-grained handling and management of resources beyond transparent resource-provisioning. Designers shall be given the ability to govern said aspects of deployment at the DDSM layer.
Supporting material:	N/A
Other comments:	N/A

Table 73: The DICE Scripting Support Requirement.

ID:	MR2.15
Title:	DICE Scripting Support

Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DDSM layer shall allow the support for linking ad-hoc config. scripts or default config. scripts within the DIA.
Rationale:	a big part in specifying and deploying/running DIAs consists in the definition/reuse of configuration scripts. The DICE profile shall allow designers to link scripts to modelling elements specific to their designed DIA.
Supporting material:	N/A
Other comments:	N/A

Table 74: The DIA Application Bundling Requirement.

ID:	MR2.16
Title:	DIA Application Bundling
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The Actionable Deployment View within the DDSM layer shall support DIA application bundling, e.g., using the CSAR formalism adopted by the TOSCA notation.
Rationale:	Container technologies are the de-facto standard for deploying DIAs. The TOSCA reference format for DICE deployment models already pre-defines a deployment bundle possibly for reuse within the DICE profile itself.
Supporting material:	N/A
Other comments:	N/A

Table 75: The IDE support to the use of profile Requirement.

ID:	MR2.17
Title:	IDE support to the use of profile
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE IDE MUST support the development of DIA exploiting the DICE profile and following the DICE methodology. This means that it should offer widzards to

	guide the developer through the steps envisioned in the DICE methodology
Rationale:	An adoption of the DICE profile not supported by a user friendly IDE can be quite cumbersome and limit the benefits of our approach. The more the IDE is user friendly the more the potential of a positive impact of the DICE profile on practitioners increases
Supporting material:	N/A
Other comments:	N/A

Table 76: The DICE Deployment Constructs Origin Requirement.

ID:	PRD2.1
Title:	DICE Deployment Constructs Origin
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The DICE Profile shall define deployment-specific construct contiguously to TOSCA-specific constructs and their relations.
Rationale:	TOSCA is the key reference format to be supported for deployment-ready DIAs - reference to its constructs shall be constant in the definition of the DICE profile.
Supporting material:	N/A
Other comments:	N/A

Table 77: The DICE Deployment Required and Provided Properties Requirement.

ID:	PRD2.2
Title:	DICE Deployment Required and Provided Properties
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE Profile shall define technology-specific properties in terms of required- and provided-properties.
Rationale:	Provided- and required-properties are an essential concept behind TOSCA-ready cloud applications. TOSCA-ready orchestrators use said constructs as requirements to drive the deployment process of parsed specifications. As a

	consequence, said constructs shall be used massively across the definition of DICE profile and its modeling elements.
Supporting material:	N/A
Other comments:	N/A

Table 78: The DICE Deployment Required and Provided Execution Platforms Requirement.

ID:	PRD2.3
Title:	DICE Deployment Required and Provided Execution Platforms
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DICE Profile shall define annotations support the specification of required- and provided-execution platforms for the deployment of DIAs.
Rationale:	execution platforms are coherent specifications that describe the environment atop which the DIA needs to be processed. DIAs specified within DICE shall include said specifications since they are required to map DICE-specified DIAs into TOSCA-ready executable CSAR bundles.
Supporting material:	N/A
Other comments:	N/A

Table 79: The DICE Deployment - NFV Requirement.

ID:	PRD2.4
Title:	DICE Deployment - NFV
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The DICE Profile shall provide facilities to model virtualized network-functions and their respective relations in an NFV topology.
Rationale:	Network-Function Virtualization shall be an integral part to DICE profile definition. Also, in defining TOSCA-compliant specifications, DIAs specified within DICE shall need to elaborate on NFV constructs to be possibly expressed using TOSCA-YAML syntax.

Supporting material:	N/A
Other comments:	N/A

2.3. WP3 Requirements

Table 80: The M2M Transformation Requirement.

ID:	R3.1
Title:	M2M Transformation
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The TRANSFORMATION_TOOLS MUST perform a model-to-model transformation taking the input from a DPIM or DTSM DICE annotated UML model and returning a formal model (e.g. Petri net model or a temporal logic model).
Rationale:	This is the main functionality needed to perform simulations and verification activities
Supporting material:	N/A
Other comments:	N/A

Table 81: The Taking into account relevant annotations Requirement.

ID:	R3.2
Title:	Taking into account relevant annotations
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The TRANSFORMATION_TOOLS MUST take into account the relevant annotations in the DICE profile (properties, constraints and metrics) whether related to performance, reliability, safety, privacy, and transform them into the corresponding artifact in the formal model
Rationale:	N/A
Supporting material:	A property is a characteristic of a system's element (e.g. tranfer rate of a disk)

Other comments:	N/A

Table 82: The Transformation rules Requirement.

ID:	R3.3
Title:	Transformation rules
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The TRANSFORMATION_TOOLS MAY be able to extract, interpret and apply the transformation rules from an external source(1).
Rationale:	An external source joined to a declarative style make it possible to extend the behavior of the system without having to modify source code. In the last term, these two requirements, will permit to provide an extension mechanism to the DICE profile (e.g. to support the impact of new parameters coming from new technologies or algorithms).
Supporting material:	1) External source: Probably a repository with the transformation rules in declarative format to be processed by QVT (Query/View/Transformation) or a similar tool
Other comments:	N/A

Table 83: The Simulation solvers Requirement.

ID:	R3.4
Title:	Simulation solvers
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The SIMULATION_TOOLS will select automatically and acording to the metric selected, the right SOLVER whether simulation or analytical solvers (e.g. Markov sollution)
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 84: The Transparency of underlying tools Requirement.

ID:	R3.6
Title:	Transparency of underlying tools
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The TRANSFORMATION_TOOLS and SIMULATION_TOOLS MUST be transparent to users. From their point of view the user is analyzing metrics from and making simulations over an enriched UML Model.
Rationale:	N/A
Supporting material:	The whole process must be atomic to the user. s/he just need to know that is simulating the behaviour of an UML model. Any tranformation or analysis we are doing to compute the metrics doesn't need to be explicited to the user (or even better expressed, there is no a first transformation phase where we show a petri net to the user. Instead, from user perspective,we compute the metric in one step). That's what we mean by "transparent"
Other comments:	N/A

Table 85: The Generation of traces from the system model Requirement.

ID:	R3.7
Title:	Generation of traces from the system model
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The VERIFICATION_TOOLS MUST be able, from the UML DICE model a system, to show possible execution traces of the system, with its corresponding time stamps. This sequence SHOULD be used by the QA_ENGINEER to determine whether the system model captures the behavior of the application or not, for model validation purposes.
Rationale:	One way to validate whether the actual system has been sufficiently captured by the model is to produce traces of the model, and see whether they are consistent with the expected behavior of the system.
Supporting material:	N/A
Other comments:	The checking of whether the trace is "reasonable" or not can only be done by the user, it cannot be done automatically by the tool. In fact, the tool will always produce traces that are

	compatible with the system model; the question is whether the system model is reasonable or not.

Table 86: The Cost/quality balance Requirement.

ID:	R3.8
Title:	Cost/quality balance
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The OPTIMIZATION_TOOLS will minimize deployment costs trying to fulfill reliability and performance metrics (e.g., map reduce jobs execution deadlines)
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 87: The Relaxing constraints Requirement.

ID:	R3.9
Title:	Relaxing constraints
Priority of accomplishment:	Could have
Type:	Requirement
Description:	Being not possible to fulfill all requirements (SLA vs cost), the OPTIMIZATION_TOOLS COULD suggest what constraints should be relaxed (whether cost related or SLA related) to obtain a compliant model
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 88: The SLA specification and compliance Requirement.

ID:	R3.10
Title:	SLA specification and compliance

Priority of accomplishment:	Must have
Type:	Requirement
Description:	All three tool types, SIMULATION_TOOLS, VERIFICATION_TOOLS and OPTIMIZATION_TOOLS MUST permit users to check their outputs against SLA's included in UML model annotations. If an SLA is violated the tools will inform the user
Rationale:	The DICE Profile inherits from MARTE how to specify non-functional properties, i.e., how to specify SLA's as requirements. Then, the WP3 TOOLS must read these SLA's and compute in the formal model results that help to verify them. For example, the UML model could specify a performance requirement of 1 sec. as the response time of a given service. Then, the SIMULATION_TOOLS must analyze the Petri net performance model to tell the response time of such service, according to the current model input parameters. The tool could highlight those SLA's that are not fulfilled.
Supporting material:	N/A
Other comments:	N/A

Table 89: The Optimization timeout Requirement.

ID:	R3.11
Title:	Optimization timeout
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The OPTIMIZATION_TOOLS MUST explore the design space and should accept the specification of a timeout and return results gracefully when this timeout is expired
Rationale:	The user should not be waiting for a response indefinitely
Supporting material:	N/A
Other comments:	N/A

Table 90: The White/black box transparency Requirement.

ID:	R3.13
Title:	White/black box transparency

Priority of accomplishment:	Must have
Type:	Requirement
Description:	For the TRANSFORMATION_TOOLS and the SIMULATION_TOOLS there will be no difference between white box and black box model elements.
Rationale:	In both cases, black or white model elements, the processes remain the same. First, annotations will come from well-known sources for some components while others will be guessed by the ARCHITECT. Later, the reasoning about the system through the formal model will lead to improvements of some attributes, parameters or constraints. Finally, the analysis of the logs coming from WP4 will provide information from real application execution. It doesn't matter whether the improved parameter refers to a black box model element (e.g., MP job or any other Hadoop framework executed in the cloud) or an ad hoc well known algorithm modeled as a white-box component.
Supporting material:	N/A
Other comments:	N/A

Table 91: The Ranged or extended what if analysis Requirement.

ID:	R3.14
Title:	Ranged or extended what if analysis
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The SIMULATION_TOOLS will be able to cover a range of possible values for a parameter and run a simulation for every different scenario (according to a gap parameter that splits the range to cover in a list of discrete values to evaluate)
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 92: The Verification of temporal safety/privacy properties Requirement.

ID:	R3.15
Title:	Verification of temporal safety/privacy properties

Priority of accomplishment:	Must have
Type:	Requirement
Description:	Taking the DICE annotated UML model (which includes the class of the property to be verified) as an input, the VERIFICATION_TOOLS MUST be able to answer questions related to whether a specific instance of the class in the UML diagram holds for the modeled system or not.
Rationale:	This is the main role of the VERIFICATION_TOOL: to be able to verify the properties defined in the DICE UML model
Supporting material:	N/A
Other comments:	N/A

2.4. WP4 Requirements

Table 93: The Monitoring data warehousing Requirement.

ID:	R4.1
Title:	Monitoring data warehousing
Priority of accomplishment:	Must have
Type:	Requirement
Description:	There will be multiple 'monitoring data collector' tools that will retrieve monitoring data from different platforms and store it under the monitoring data warehouse. The data warehouse will support different data types, providing near real-time access.
Rationale:	We expect that the monitoring agents will produce a high number of monitoring data. This data needs to be stored in the application's test and runtime environment, capable of handling the bulk of data.
Supporting material:	In the early stage, the monitoring data refers to logs produced by the Big Data applications (Hadoop, NOSQL).
Other comments:	In the early stage, the monitoring data refers to logs produced by the Big Data applications (Hadoop, NOSQL) Technologies supported: Apache Hadoop (yarn, hdfs), Apache Spark, Apache Storm, Apache Cassandra, MongoDB

	Also, historical data is collected for Spark and YARN.

Table 94: The Monitoring data warehouse schema Requirement.

ID:	R4.2
Title:	Monitoring data warehouse schema
Priority of accomplishment:	Must have
Type:	Requirement
Description:	MONITORING_TOOLS storing the monitoring data MUST use a schema that lets identify the sources of the monitoring data, but is general enough to permit adding new sources.
Rationale:	The monitoring data warehousing needs to accommodate for any monitoring data input format and content without losing any relevant data. The monitoring entries need to be equipped with metadata, but the contents need to stay intact.
Supporting material:	N/A
Other comments:	N/A

Table 95: The Monitoring data versioning Requirement.

ID:	R4.2.1
Title:	Monitoring data versioning
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The metrics records MUST include the information on the version of the APPLICATION's build.
Rationale:	Association between the monitored application's version and the monitoring data is crucial for quality enhancement and configuration recommendation engine.
Supporting material:	N/A
Other comments:	N/A

Table 96: The Supplying the version number Requirement.

ID:	R4.2.2
Title:	Supplying the version number

Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DEPLOYMENT_TOOLS MUST supply the APPLICATION's current version number when starting the MONITORING_TOOLS
Rationale:	The version number has to arrive from tools external to monitoring tools.
Supporting material:	N/A
Other comments:	N/A

Table 97: The Monitoring data extractions Requirement.

ID:	R4.3
Title:	Monitoring data extractions
Priority of accomplishment:	Must have
Type:	Requirement
Description:	MONITORING_TOOLS MUST perform monitoring data pre-processing (extraction) before storing the data in the data warehouse in order to facilitate usage by other tasks.
Rationale:	Different actors have different /expectations from the monitoring data stored in DW, such that aggregations over time periods, different granularities etc.
Supporting material:	N/A
Other comments:	Pre-processing refers to extraction and validation operations in order to extract (parse) log files and validate the obtained data (e.g. valid email address, valid IP address etc.).

Table 98: The Monitoring data format transformations Requirement.

ID:	R4.4
Title:	Monitoring data format transformations
Priority of accomplishment:	Must have
Type:	Requirement

Description:	MONITORING_TOOLS MUST perform data transformation when the data is retrieved from the data warehouse.
Rationale:	Tools may require data in different formats in order to function. This transformation from the DW internal format to the required format is done at data retrieval.
Supporting material:	N/A
Other comments:	cleaning, normalization, projection, windowing in time series,

Table 99: The Monitoring data access restrictions Requirement.

ID:	R4.6
Title:	Monitoring data access restrictions
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The data warehouse MUST provide the ability to prevent unauthorised access to the monitoring data.
Rationale:	The monitored data may contain sensitive and private data.
Supporting material:	N/A
Other comments:	N/A

Table 100: The Monitoring tools REST API Requirement.

ID:	R4.7
Title:	Monitoring tools REST API
Priority of accomplishment:	Must have
Type:	Requirement
Description:	MONITORING_TOOLS MUST expose their functionality using simple REST API.
Rationale:	This interface will facilitate querying, data transformation and extraction tasks.
Supporting material:	N/A
Other comments:	The REST interface will support monitoring data storage, retrieval, transformation, versioning etc.

Table 101: The Monitoring Visualization Requirement.

ID:	R4.8
Title:	Monitoring Visualization
Priority of accomplishment:	Should have
Type:	Requirement
Description:	MONITORING_TOOLS SHOULD support interactive visualization of monitoring data
Rationale:	Visualization will give human actors an initial overview over the monitoring data available for their APPLICATION.
Supporting material:	N/A
Other comments:	This will reuse an existing Web-based visualization tool available for the data warehouse platform (e.g. Kibana Web tool for Elastic platform)

Table 102: The Data Warehouse replication Requirement.

ID:	R4.9
Title:	Data Warehouse replication
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The data warehouse COULD have replication capabilities.
Rationale:	Replication will offer increased availability and storage size in case monitoring data collected will be very large. Initially, we will adopt a centralized deployment.
Supporting material:	N/A
Other comments:	N/A

Table 103: The Resource consumption breakdown Requirement.

ID:	R4.11
Title:	Resource consumption breakdown
Priority of accomplishment:	Must have
Type:	Requirement

Description:	The DEVELOPER MUST be able to see via the ENHANCEMENT_TOOLS the resource consumption breakdown into its atomic components.
Rationale:	Existence of different abstraction levels between design concepts (e.g., abstractions in the DICE profile) and runtime measurements hides the details on what high-level request effectively generated the request for data.
Supporting material:	R4IDE1
Other comments:	N/A

Table 104: The Bottleneck Identification Requirement.

ID:	R4.12
Title:	Bottleneck Identification
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The ENHANCEMENT_TOOLS MUST indicate which classes of requests represent bottlenecks for the application in a given deployment.
Rationale:	N/A
Supporting material:	R4IDE2
Other comments:	N/A

Table 105: The Semi-automated anti-pattern detection Requirement.

ID:	R4.13
Title:	Semi-automated anti-pattern detection
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The ENHANCEMENT_TOOLS MUST feature a semi-automated analysis to detect and notify the presence of anti-patterns in the application design.
Rationale:	N/A
Supporting material:	N/A

Other comments:	Anti-patterns will most probably use both UML information combined with monitoring data.

Table 106: The Refactoring methods Requirement.

ID:	R4.14
Title:	Refactoring methods
Priority of accomplishment:	Should have
Type:	Requirement
Description:	Once correlation between anomalies in runtime and anti-patterns has been detected, the ENHANCEMENT_TOOLS SHOULD propose methods for refactoring the design leveraging parameters extracted from the traces.
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 107: The Enhancement tools version difference Requirement.

ID:	R4.16
Title:	Enhancement tools version difference
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The ENHANCEMENT_TOOLS COULD compare two versions of the application to identify relevant changes.
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 108: The Enhancement tools data acquisition Requirement.

ID:	R4.17
Title:	Enhancement tools data acquisition
Priority of accomplishment:	Must have

Type:	Requirement
Description:	The ENHANCEMENT_TOOLS MUST perform its operations by retrieving the relevant monitoring data from the MONITORING_TOOLS.
Rationale:	Local data processing appears more flexible than processing directly inside the data warehouse.
Supporting material:	N/A
Other comments:	N/A

Table 109: The Enhancement tools model access Requirement.

ID:	R4.18
Title:	Enhancement tools model access
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The ENHANCEMENT_TOOLS MUST be able to access the DICE profile model associated to the considered version of the APPLICATION.
Rationale:	Parameter inference and anti-pattern detection need UML model.
Supporting material:	N/A
Other comments:	N/A

Table 110: The Parameterization of simulation and optimization models. Requirement.

ID:	R4.19
Title:	Parameterization of simulation and optimization models.
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The ENHANCEMENT_TOOLS MUST extract or infer the input parameters needed by the SIMULATION_TOOLS and OPTIMIZATION_TOOLS to perform the quality analyses.
Rationale:	N/A
Supporting material:	N/A

Other comments:	Input parameters inferred as a result of this requirement may be completed by additional parameters provided by end-user or other tools (e.g. configuration recommender).

Table 111: The Model parameter uncertainties Requirement.

ID:	R4.20
Title:	Model parameter uncertainties
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The REQ_ENGINEER COULD express uncertainty on some performance/reliability input parameters (e.g., execution times) in the DICE profile by means of a prior distribution or an interval. The ENHANCEMENT_TOOLS COULD take into account these parameters to esti
Rationale:	DoW mentions Bayesian estimation techniques. These techniques can explicitly account for the uncertainty provided by the REQ_ENGINEER.
Supporting material:	R4IDE3
Other comments:	This requirement may be alternatively stated as part of WP2 or WP3, since it also affects the DICE profile. The requirement would expand the scientific impact of the tool, but if too complex to implement it might be ignored without major consequences.

Table 112: The Model parameter confidence intervals Requirement.

ID:	R4.21
Title:	Model parameter confidence intervals
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The ENHANCEMENT_TOOLS COULD return confidence intervals for each inferred parameter of the performance and reliability models.
Rationale:	The WP3 models require to provide a number of parameters, such as CPU speeds. These will be inferred by the ENHANCEMENT_TOOLS of WP4 from the monitoring data. However, the estimation is subject to uncertainties so confidence intervals could be provided to

Supporting material:	R4IDE4
Other comments:	N/A

Table 113: The Time-based ordering of monitoring data entries Requirement.

ID:	R4.22
Title:	Time-based ordering of monitoring data entries
Priority of accomplishment:	Must have
Type:	Domain Assumption
Description:	Monitoring data MUST support the reconstruction of a sequence of events and the identification of the time when things occurred (for example a consistent timestamp in a distributed system)
Rationale:	While in general data is application-dependent, for running trace checking it is important that data is time-based ordered.
Supporting material:	N/A
Other comments:	In case of data collected from multiple nodes of a distributed system, MONITORING_TOOLS must ensure data is consistently ordered when providing answer to actors' queries.

Table 114: The Anomaly detection in APPLICATION quality Requirement.

ID:	R4.24
Title:	Anomaly detection in APPLICATION quality
Priority of accomplishment:	Must have
Type:	Requirement
Description:	ANOMALY_DETECTION_TOOL MUST provide means to detect anomalies in APPLICATION's quality after deployment
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 115: The Unsupervised Anomaly Detection Requirement.

ID:	R4.24.1
Title:	Unsupervised Anomaly Detection
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The ANOMALY_DETECTION_TOOL must be able to detect anomalies from the APPLICATION using unsupervised methods. It is assumed that normal data instances lie closer to their closest centrid while anomalies are far away.
Rationale:	Monitored data may come in unlabeled (training dataset hard to create) form thus it is important to detect anomalies based on unsupervised methodology. It is assumed that normal data instanes are more frequent than anomalies.
Supporting material:	N/A
Other comments:	N/A

Table 116: The Supervised Anomaly Detection Requirement.

ID:	R4.24.2
Title:	Supervised Anomaly Detection
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The ANOMALY_DETECTION_TOOL must be able to detect anomalies from the APPLICATION using supervised methods.
Rationale:	Creation of training dataset can be created thus it is posible to train predictive models based in supervised methodology.
Supporting material:	N/A
Other comments:	N/A

Table 117: The Contextual Anomalies Requirement.

ID:	R4.24.3
Title:	Contextual Anomalies
Priority of accomplishment:	Should have

Type:	Domain Assumption
Description:	The ANOMALY_DETECTION_TOOL should be able to detect that data instances of a given APPLICATION are anomalouse in a specific instance but not otherwise.
Rationale:	This is induced by the structure of the dataset and has to be specified as part of the problem formulation using the MONITORING_TOOLS. Data instances must be defined using: contextual attributes and behavioural attributes. Time-series data.
Supporting material:	N/A
Other comments:	N/A

Table 118: The Collective anomalies Requirement.

ID:	R4.24.4
Title:	Collective anomalies
Priority of accomplishment:	Should have
Type:	Domain Assumption
Description:	The ANOMALY_DETECTION_TOOL must be able to detect that a collection of related data instances of a given APPLICATION can be anomalouse with respect to the entire colleted dataset.
Rationale:	Data instances might not be anomalouse by themselves.This type of anomalies occur when the data instances are related. Sequence data.
Supporting material:	N/A
Other comments:	N/A

Table 119: The Predictive Model saving for Anomaly Detection Requirement.

ID:	R4.24.5
Title:	Predictive Model saving for Anomaly Detection
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The ANOMALY_DETECTION_TOOL must be able to save the predictive model trained using monitored

	APPLICATION data. These models can be reused and serve as a bootstrap for future predictive models.
Rationale:	Two APPLICATIONS can be similar or a single APPLICATION can have many versions thus a trained predictive model can be reused or can serve as a starting point. Can use PMML format.
Supporting material:	N/A
Other comments:	N/A

Table 120: The Semi-automated data labelling Requirement.

ID:	R4.24.6
Title:	Semi-automated data labelling
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The ANOMALY_DETECTION_TOOL COULD have the capability to insert labeled anomalous data instances in order to create training datasets for supervised training for Anomaly detection.
Rationale:	As anomalouse instances are far fewer than normal data instances (unbalanced class distribution) the insertion of labeled anomalies can help create a more viable predictive model. Optaining fully labeled data is most often unfeasible.
Supporting material:	N/A
Other comments:	N/A

Table 121: The Adaptation of thresholding Requirement.

ID:	R4.24.7
Title:	Adaptation of thresholding
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The ANOMALY_DETECTION_TOOL COULD ask feedback to the user about the predefined threshold used to detect an outlier and adjust based on the feedback received.
Rationale:	A given anomaly detection result could be scored by the user. A simple algorithm could interpret this to refine the threshold.

Supporting material:	N/A
Other comments:	N/A

Table 122: The Visualization of analysis results Requirement.

ID:	R4.25
Title:	Visualization of analysis results
Priority of accomplishment:	Should have
Type:	Requirement
Description:	ENHANCEMENT_TOOLS SHOULD be capable of visualizing analysis results
Rationale:	N/A
Supporting material:	R4IDE5
Other comments:	N/A

Table 123: The Report generation of analysis results Requirement.

ID:	R4.26.1
Title:	Report generation of analysis results
Priority of accomplishment:	Should have
Type:	Requirement
Description:	ENHANCEMENT_TOOLS SHOULD be able to generate reports with analysis results
Rationale:	This feature is needed: a) for when DEVELOPER/ARCHITECT needs to make a decision and make changes manually, b) to create history of changes (may be useful)
Supporting material:	N/A
Other comments:	N/A

Table 124: The Report generation of analysis results Requirement.

ID:	R4.26.2
Title:	Report generation of analysis results

Priority of accomplishment:	Should have
Type:	Requirement
Description:	ANOMALY_DETECTION_TOOL SHOULD be able to generate reports with analysis results
Rationale:	This feature is needed: a) for when DEVELOPER/ARCHITECT needs to make a decision and make changes manually, b) to create history of changes (may be useful)
Supporting material:	N/A
Other comments:	N/A

Table 125: The Propagation of changes/automatic annotation of UML models Requirement.

ID:	R4.27
Title:	Propagation of changes/automatic annotation of UML models
Priority of accomplishment:	Must have
Type:	Requirement
Description:	ENHANCEMENT_TOOLS MUST be capable of automatically updating UML models with analysis results (new values)
Rationale:	Increase efficiency of iterative enhancement process
Supporting material:	N/A
Other comments:	N/A

Table 126: The Safety and privacy properties loading Requirement.

ID:	R4.28
Title:	Safety and privacy properties loading
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The TRACE_CHECKING_TOOL MUST allow the DEVELOPER/ARCHITECT to specify the class of properties (either safety or privacy) in the UML model of the application described through the DICE profile

Rationale:	The properties to be analyzed are application-dependent. They are defined in the DICE model of the application as a class of properties. The user knows what properties are to be monitored, so he/she should select those that most interest him/her in the ID
Supporting material:	R4IDE6
Other comments:	N/A

Table 127: The Definition of time window of interest for safety/privacy properties Requirement.

ID:	R4.28.1
Title:	Definition of time window of interest for safety/privacy properties
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The TRACE_CHECKING_TOOL MUST allow the DEVELOPER/ARCHITECT to choose the time window of interest, which must be considered when choosing the traces to be analyzed.
Rationale:	The user selects only the relevant part of the application history to analyze
Supporting material:	N/A
Other comments:	Trace checking is not a real-time analysis of a stream of events; it is done in batch mode (see also R4.30), so the user should select the window of interest

Table 128: The Mechanisms for the definition of the time window of interest for safety/privacy properties Requirement.

ID:	R4.28.1.1
Title:	Mechanisms for the definition of the time window of interest for safety/privacy properties
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The TRACE_CHECKING_TOOL COULD offer the DEVELOPER/ARCHITECT different ways to choose the time window of interest; the time window could be indicated

	though a size (to computed in the past from the current instant), or using a starting and ending event.
Rationale:	The user chooses the best way to specify the slice of the runtime history of the application to be analyzed.
Supporting material:	N/A
Other comments:	N/A

Table 129: The Event occurrences detection for safety and privacy properties monitoring Requirement.

ID:	R4.29
Title:	Event occurrences detection for safety and privacy properties monitoring
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The TRACE_CHECKING_TOOL MUST be able to retrieve, depending on the properties to be checked, the relevant data stored in the DW, and translate them into traces of relevant events for the trace checking
Rationale:	The ANOMALY_DETECTION_TOOL, and the trace checking tool in particular, requires as input traces of events of interest, which must be identified before they are fed to the tool. There is probably a translation to be performed from what is stored in the DW
Supporting material:	N/A
Other comments:	This is similar/related to R4.4, but it is probably worth it to highlight this issue. It is also linked to R4.32

Table 130: The Safety and privacy properties monitoring Requirement.

ID:	R4.30
Title:	Safety and privacy properties monitoring
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The TRACE_CHECKING_TOOL MUST be able to check, given a trace of the events of interest of the application, whether that trace is compatible with the desired safety and privacy properties

Rationale:	This is the main functionality of the trace cheking tool
Supporting material:	N/A
Other comments:	The check is performed off-line, i.e., in batch mode (a trace is retrieved from the DW, then analysed by the trace checking tool)

Table 131: The Safety and privacy properties result reporting Requirement.

ID:	R4.30.1
Title:	Safety and privacy properties result reporting
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The TRACE_CHECKING_TOOL MUST be able to notify the DEVELOPER/ARCHITECT when a safety/privacy property is/might be violated by the application.
Rationale:	The trace checking tool must be able to give feedback to the developers
Supporting material:	N/A
Other comments:	This requirement is linked to R4.26, maybe it is a sub-requirement

Table 132: The Feedback from safety and privacy properties monitoring to UML models Requirement.

ID:	R4.31
Title:	Feedback from safety and privacy properties monitoring to UML models
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The TRACE_CHECKING_TOOL COULD provide feedback about safety/privacy properties violated at runtime in the UML DICE models
Rationale:	Providing feedback in the UML DICE models might help the DEVELOPER/ARCHITECT get a picture of where the problems are in the application
Supporting material:	N/A
Other comments:	N/A

Table 133: The Feedback from safety and privacy properties monitoring to UML models concerning violatedtime bounds Requirement.

ID:	R4.31.1
Title:	Feedback from safety and privacy properties monitoring to UML models concerning violated time bounds
Priority of accomplishment:	Could have
Type:	Requirement
Description:	In the feedback provided by the TRACE_CHECKING_TOOL to the DEVELOPER/ARCHITECT, the tools COULD highlight when a timing requirement is violated, and what is the value of the violation
Rationale:	The specific feedback about timing violations might help the DEVELOPER/ARCHITECT adjust the parameters of the models/properties
Supporting material:	R4IDE7?
Other comments:	N/A

Table 134: The Correlation between data stored in the DW and DICE UML models Requirement.

ID:	R4.32
Title:	Correlation between data stored in the DW and DICE UML models
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The TRACE_CHECKING_TOOL is able to link the information that is stored in the data warehouse with the features and concepts of the DICE UML models (operations, attributes, objects, etc.)
Rationale:	The properties analyzed by the TRACE_CHECKING_TOOL through trace checking are expressed in terms of the elements of the DICE UML model. Hence, to run the trace checking the events stored in the data warehouse must be correlated with what is described by t
Supporting material:	N/A

Other comments:	It is unclear which component should bear responsibility of this fundamental part. Would it be ENHANCEMENT_TOOLS or a dedicated component?

Table 135: The Relation between TRACE_CHECKING_TOOL and IDE Requirement.

ID:	R4.33
Title:	Relation between TRACE_CHECKING_TOOL and IDE
Priority of accomplishment:	Should have
Type:	Requirement
Description:	It SHOULD be possible to launch the TRACE_CHECKING_TOOL from the IDE
Rationale:	The idea is that the trace checking is performed starting from the elements that are described in the DICE UML model (see requirement R4.32). Hence, trace checking is invoked from the UML IDE. The IDE has a link to the DW, and when the user asks for perfo
Supporting material:	R4IDE8
Other comments:	N/A

Table 136: The Monitoring for quality tests Requirement.

ID:	R4.34
Title:	Monitoring for quality tests
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The MONITORING_TOOLS MUST support and collect all the metrics relevant for the QTESTING_TOOLS
Rationale:	The quality testing tools rely on the data obtained by monitoring the runtime of the application during the test runs.
Supporting material:	N/A
Other comments:	N/A

Table 137: The Tag monitoring data with OSLC tags Requirement.

ID:	R4.35

Title:	Tag monitoring data with OSLC tags
Priority of accomplishment:	Must have
Type:	Requirement
Description:	MONITORING_TOOLS MUST exports monitoring data in OSLC-compliant format
Rationale:	DICE tools need to show compliance with OSLC standard
Supporting material:	N/A
Other comments:	N/A

Table 138: The Detect anomalies between two versions of DIA Requirement.

ID:	R4.36
Title:	Detect anomalies between two versions of DIA
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The ANOMALY_DETECTION_TOOL MUST compare two versions of the application to identify the presence/absence of anomaly(-ies).
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 139: The ANOMALY_DETECTION_TOOL should get input parameters from IDE Requirement.

ID:	R4.37
Title:	ANOMALY_DETECTION_TOOL should get input parameters from IDE
Priority of accomplishment:	Must have
Type:	Requirement
Description:	Model training block of ANOMALY_DETECTION_TOOL MUST accept the following information from the IDE: quality/performance metric to investigate for the presence of anomaly, list of input parameters and their levels (high/low/other)

Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 140: The MONITORING_TOOL integration in DICE IDE Requirement.

ID:	R4.38
Title:	MONITORING_TOOL integration in DICE IDE
Priority of accomplishment:	Must have
Type:	Requirement
Description:	It must be possible to connect to monitoring service (tool) from DICE IDE
Rationale:	DEVELOPER/ARCHITECT has a unique point of access to all DICE tools/services
Supporting material:	R4IDE9
Other comments:	N/A

Table 141: The Discover of Storm topologies Requirement.

ID:	R4.39
Title:	Discover of Storm topologies
Priority of accomplishment:	Should have
Type:	Requirement
Description:	MONITORING_TOOLS SHOULD discover existing Storm topologies in monitored cluster and configure those nodes appropriatelly
Rationale:	Streamlines the usage of the tool
Supporting material:	N/A
Other comments:	N/A

Table 142: The Collect and index raw data from Storm worker nodes log files Requirement.

ID:	R4.40
Title:	Collect and index raw data from Storm worker nodes log files

Priority of accomplishment:	Must have
Type:	Requirement
Description:	MONITORING_TOOLS MUST collect and index raw data from Storm worker nodes log files
Rationale:	Trace checking (TC) tool needs detailed log data, which is not provided through Storm Metrics API
Supporting material:	N/A
Other comments:	N/A

Table 143: The Collect and index application-specific data coming from Posidonia Operations applicationsRequirement.

ID:	R4.41
Title:	Collect and index application-specific data coming from Posidonia Operations applications
Priority of accomplishment:	Must have
Type:	Requirement
Description:	MONITORING_TOOLS MUST collect and index raw data from the log files producd by Complex Event Processing (CEP) module of PRO's Posidonia Operations
Rationale:	PRO want to use DMon platform to collect, query and visualize Posidonia cluster
Supporting material:	N/A
Other comments:	N/A

2.5. WP5 Requirements

Table 144: The Versioning Requirement.

ID:	R5.1
Title:	Versioning
Priority of accomplishment:	Must have
Type:	Domain Assumption

Description:	Everything in the user's project MUST be treated as code. All code MUST be versioned and the DEPLOYMENT_TOOLS and CI_TOOLS tools MUST involve the version information in their process.
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 145: The Testing project Requirement.

ID:	R5.2
Title:	Testing project
Priority of accomplishment:	Must have
Type:	Domain Assumption
Description:	An ADMINISTRATOR MUST configure a project or an account in the fault injection environment with resource quotas set to accommodate application tests.
Rationale:	The DICE tools will deploy and test the application in the fault injection environment running either in the private or the public cloud. As a pre-requiste of the tests, the fault injection environment needs to be pre-configured to allow provisionning of resources without going over the set quotas.
Supporting material:	resources: CPU, RAM, hard drive space, network connectivity project or account: an environment in the cloud permitting provisioning of a limited or an unlimited set of virtual machines
Other comments:	In the context of DICE development, we assume this will be in a testbed. Otherwise the development team has a private data centre or a community cloud computing account to be used.

Table 146: The Continuous integration tools deployment Requirement.

ID:	R5.3
Title:	Continuous integration tools deployment
Priority of accomplishment:	Should have
Type:	Requirement

Description:	The ADMINISTRATOR MUST manually install and configure CI_TOOLS MUST upon installation of the CI_TOOLS and can be updated later on. The configuration MUST enable CI_TOOLS to access the TESTBED.
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 147: The TOSCA format for blueprints Requirement.

ID:	R5.4
Title:	TOSCA format for blueprints
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DEPLOYMENT_TOOLS MUST be able to support TOSCA blueprints as the target cloud orchestrator's DSL
Rationale:	The specialised tools for configuring the environment and orchestrating applications (e.g., Chef) use their own DSL other than TOSCA.
Supporting material:	DSL: domain-specific language
Other comments:	Changed the name and updated the text

Table 148: The Big Data technology support Requirement.

ID:	R5.4.1
Title:	Big Data technology support
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DEPLOYMENT_TOOLS MUST be able to deploy all the DICE supported core building blocks.
Rationale:	DICE will provide support for the initial set of services that support use cases and basic needs.
Supporting material:	N/A
Other comments:	Changed the description to include the notion of the DICE technology library. Also changed the title.

Table 149: The Translation tools autonomy Requirement.

ID:	R5.4.2
Title:	Translation tools autonomy
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DEPLOYMENT_TOOLS MUST take all of its DIA-related input from the DDSM, which directly translate into the TOSCA model, or from the ADMINISTRATOR set values. Therefore it MUST NOT require any additional user's input in an interactive way.
Rationale:	The DEPLOYMENT_TOOLS have to operate transparently for the users.
Supporting material:	N/A
Other comments:	Changed the description slightly to account for the DICER tool translation from DDSM. Updated to account for platform-related inputs in the DICE deployment service.

Table 150: The Deployment blueprint contents Requirement.

ID:	R5.4.3
Title:	Deployment blueprint contents
Priority of accomplishment:	Must have
Type:	Domain Assumption
Description:	The contents of the deployment plan (i.e., the blueprint) must describe the application to be deployed. The DEPLOYMENT_TOOLS MUST interpret the supported blueprint by employing the DICE technology library to take the installation and configuration steps necessary to deploy the application in the fault injection environment as per blueprint.
Rationale:	N/A
Supporting material:	N/A
Other comments:	Changed the title and description to follow the Y1 terminology better.

Table 151: The Deployment plans execution tools Requirement.

ID:	R5.4.4
Title:	Deployment plans execution tools
Priority of accomplishment:	Should have
Type:	Domain Assumption
Description:	The DEPLOYMENT_TOOLS SHOULD rely on third-party runtime configuration and deployment tools.
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 152: The Deployment tools transparency Requirement.

ID:	R5.4.5
Title:	Deployment tools transparency
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The DEPLOYMENT_TOOLS SHOULD NOT require from ADMINISTRATOR to take part in any individual deployment.
Rationale:	For ease of use and extensibility, the DEPLOYMENT_TOOLS should hide their inner details to the external world
Supporting material:	N/A
Other comments:	Changed the description for a better clarification

Table 153: The Deployment plans extendability Requirement.

ID:	R5.4.6
Title:	Deployment plans extendability
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The DEPLOYMENT_TOOLS MAY be extended by the ADMINISTRATOR with other building blocks not in the core set.

Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 154: The Deployment of the application in a test environment Requirement.

ID:	R5.4.7
Title:	Deployment of the application in a test environment
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DEPLOYMENT_TOOLS MUST provision the resources required by the application
Rationale:	Assuming that there is an application, its model and a set of quality test, a dedicated set of resources need to exist and be assigned to the tests.
Supporting material:	resources: CPU, RAM, hard drive space, network connectivity
Other comments:	N/A

Table 155: The Starting the monitoring tools Requirement.

ID:	R5.4.8
Title:	Starting the monitoring tools
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DEPLOYMENT_TOOLS MUST start the MONITORING_TOOLS agents on the deployed nodes for the application.
Rationale:	Monitoring tools are an essential part of the DICE quality testing tools.
Supporting material:	N/A
Other comments:	Changed description for a better clarification

Table 156: The Deployment plans portability Requirement.

ID:	R5.4.9
Title:	Deployment plans portability
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The DEPLOYMENT_TOOLS SHOULD be able to support more than one vendor's IaaS.
Rationale:	N/A
Supporting material:	N/A
Other comments:	

Table 157: The Translation of DDSM Requirement.

ID:	R5.4.10
Title:	Translation of DDSM
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DEPLOYMENT_TOOLS MUST provide actionable translation from the DDSM to TOSCA blueprints.
Rationale:	DICE methodology must enable automated workflow for the steps where additional user input is not required.
Supporting material:	N/A
Other comments:	New requirement

Table 158: The Use of TOSCA standard Requirement.

ID:	R5.4.11
Title:	Use of TOSCA standard
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The DEPLOYMENT_TOOLS SHOULD accept blueprints that are OASIS TOSCA compliant
Rationale:	N/A
Supporting material:	N/A

Other comments:	New requirement

Table 159: The User-provided initial data retrieval Requirement.

ID:	R5.5
Title:	User-provided initial data retrieval
Priority of accomplishment:	Must have
Type:	Requirement
Description:	CI_TOOLS MUST retrieve from the artifact repository or use input from the code versioning system any user-provided initial data
Rationale:	Applications may require initial data prepared by the DEVELOPER to be loaded in the databases. If the DEVELOPER prepares them in a dedicated place, the CI_TOOLS are responsible to retrieve them and have them loaded in the databases.
Supporting material:	artifact repository: a dedicated repository for built application programs and libraries and any additional data such as bulk data
Other comments:	N/A

Table 160: The Test workload generation Requirement.

ID:	R5.6
Title:	Test workload generation
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The QTESTING_TOOLS MUST be able to generate the workload with pre-specified characteristics for the APPLICATION
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 161: The Data loading support Requirement.

ID:	R5.7

Title:	Data loading support
Priority of accomplishment:	Should have
Type:	Requirement
Description:	DEPLOYMENT_TOOLS and QTESTING_TOOLS SHOULD support bulk loading and bulk unloading of the data for the core building blocks.
Rationale:	DICE should support the core building blocks (e.g., technologies such as CEPH/HDFS, SQL, NoSQL) with the ability to load the inital data in a standard and documented form (eg SQL scripts, files, etc). DICE should also allow to unload that data (delete files, drop table, etc).
Supporting material:	N/A
Other comments:	N/A

Table 162: The Data loading hook Requirement.

ID:	R5.7.1
Title:	Data loading hook
Priority of accomplishment:	Should have
Type:	Requirement
Description:	DEPLOYMENT_TOOLS SHOULD provide a well-defined way to accept the initial bulk data that they can load.
Rationale:	This requirement provides to the DEVELOPER a way to prepare the initial data, which DEPLOYMENT_TOOLS load into the databases.
Supporting material:	N/A
Other comments:	Changed the description and the Tool associated, because the initial data only applies to the deployment tools, while the quality testing tools provide data fed during runtime

Table 163: The Data feed actuator Requirement.

ID:	R5.7.2
Title:	Data feed actuator
Priority of accomplishment:	Should have

Type:	Requirement
Description:	QTESTING_TOOLS SHOULD provide an actuator for sending generated or user-provided data to the application under test.
Rationale:	This requirement provides to the DEVELOPER a way to prepare the initial data, which QTESTING_TOOLS feed to the application during testing.
Supporting material:	N/A
Other comments:	New requirement after splitting R5.7.1

Table 164: The Definition of quality test Requirement.

ID:	R5.8
Title:	Definition of quality test
Priority of accomplishment:	Must have
Type:	Domain Assumption
Description:	A quality test of the QTESTING_TOOLS MUST include at least executable code to generate the workload for the application, a timeout and a set of target monitoring metrics to be collected by the MONITORING_TOOLS.
Rationale:	N/A
Supporting material:	Workload may be artificial or from real-traces collected by the MONITORING_TOOLS.
Other comments:	N/A

Table 165: The Starting the quality testing Requirement.

ID:	R5.8.2
Title:	Starting the quality testing
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The QTESTING_TOOLS MAY be invoked by the CI_TOOLS or by the QA_TESTER
Rationale:	Addresses the responsibility of executing the programs or scripts, which implement the quality assurance runs.
Supporting material:	N/A

Other comments:	N/A

Table 166: The Test run independence Requirement.

ID:	R5.8.3
Title:	Test run independence
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The QTESTING_TOOLS MUST ensure that no side effects from past or ongoing tests leak into the runtime of any other test.
Rationale:	Each test needs to be run independently from the other test runs. The test results should be as repeatable as possible.
Supporting material:	N/A
Other comments:	N/A

Table 167: The Test outcome Requirement.

ID:	R5.8.5
Title:	Test outcome
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The QTESTING_TOOLS MUST provide the test outcome to CI_TOOLS: success or failure
Rationale:	The outcome of each test must be a clear "success" of "failure". The tests with clear criteria of success or failure must provide the decision. The tests, which run a survey, benchmark or stress-test always succeed unless there is an error in the runtime.
Supporting material:	N/A
Other comments:	Relates to R5.16

Table 168: The User's unit and regression tests code execution inclusion Requirement.

ID:	R5.9
Title:	User's unit and regression tests code execution inclusion

Priority of accomplishment:	Must have
Type:	Requirement
Description:	The CI_TOOLS MUST offer the ability to run unit tests and regression tests. The unit tests and regression tests SHOULD be written by the DEVELOPER, who SHOULD have the ability of choosing which ones to run.
Rationale:	Addresses the responsibility of executing the programs or scripts, which implement the quality assurance runs.
Supporting material:	N/A
Other comments:	N/A

Table 169: The Continuous integration tools dashboard Requirement.

ID:	R5.10
Title:	Continuous integration tools dashboard
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The CI_TOOLS SHOULD offer a dashboard that consolidates the view on the state of the application and the deployed components.
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 170: The Quality testing tools IDE integration Requirement.

ID:	R5.11
Title:	Quality testing tools IDE integration
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The IDE SHOULD provide the means to configure the QTESTING_TOOLS execution
Rationale:	Quality tests may come with parameters such as the number of tests to run or the duration of each tests, which the user should be able to change.

Supporting material:	N/A
Other comments:	

Table 171: The Testing results feedback Requirement.

ID:	R5.12
Title:	Testing results feedback
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The CI_TOOLS MUST provide feedback to the DEVELOPER on the results of the unit tests.
Rationale:	The CI_TOOLS invoke the testing on the user's behalf. Therefore they must indicate what the QTESTING_TOOLS returned as their outcome.
Supporting material:	N/A
Other comments:	N/A

Table 172: The Test the application for efficiency Requirement.

ID:	R5.13
Title:	Test the application for efficiency
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The QTESTING_TOOLS MUST be capable of running tests with any configuration provided to it.
Rationale:	N/A
Supporting material:	Reference metrics for performance and costs should be defined project-wise.
Other comments:	N/A

Table 173: The Test the application for reliability Requirement.

ID:	R5.14
Title:	Test the application for reliability

Priority of accomplishment:	Must have
Type:	Requirement
Description:	The QTESTING_TOOLS MUST be tested for the application's ability to maintain the functionality and data integrity even when there are outages and faults in the supporting system.
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 174: The Test the behaviour when resources become exhausted Requirement.

ID:	R5.14.1
Title:	Test the behaviour when resources become exhausted
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The QTESTING_TOOLS MUST provide the ability to saturate and exhaust resources used by the application.
Rationale:	DICE tools must enable getting a feedback on what happens when a resource is exhausted. The application may crash, corrupt data, request scale-up of infrastructure or stop gracefully.
Supporting material:	Source literature: The Pragmatic Programmer
Other comments:	N/A

Table 175: The Trigger deliberate outages and problems to assess the application's behaviour under faultsRequirement.

ID:	R5.14.2
Title:	Trigger deliberate outages and problems to assess the application's behaviour under faults
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The QTESTING_TOOLS SHOULD use the fault injection environments functionality to test the application's resilience.

Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 176: The Test the application for safety Requirement.

ID:	R5.15
Title:	Test the application for safety
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The QTESTING_TOOLS COULD test the application for safety properties.
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 177: The Test the application for data protection Requirement.

ID:	R5.15.1
Title:	Test the application for data protection
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The QTESTING_TOOLS COULD test the application for its ability to protect the data from unauthorized access.
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 178: The Provide monitoring of the quality aspect of the development evolution (quality regression) Requirement.

ID:	R5.16
Title:	Provide monitoring of the quality aspect of the development evolution (quality regression)

Priority of accomplishment:	Must have
Type:	Requirement
Description:	The CI_TOOLS MUST record the results of each test and map them to the momentary project's (model, code etc.) version.
Rationale:	While the QTESTING_TOOLS produce the direct results of success or failure, it must be CI_TOOLS that ensure these results are stored and available for inspection of history.
Supporting material:	results: success/failure, quality indicators
Other comments:	See also R5.1 and R5.8.4

Table 179: The Quick testing vs comprehensive testing Requirement.

ID:	R5.17
Title:	Quick testing vs comprehensive testing
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The QTESTING_TOOLS MUST receive as input parameter the scope of the tests to be run.
Rationale:	Speed is important when designing and developing code. DICE should provide two (or more) profiles for testing: a quick one running only the representative tests, and a long one (for "overnight" tests) giving a more comprehensive assessment.
Supporting material:	N/A
Other comments:	N/A

Table 180: The Deployment configuration review Requirement.

ID:	R5.19
Title:	Deployment configuration review
Priority of accomplishment:	Must have
Type:	Requirement

Description:	The CI_TOOLS MUST enable that ADMINISTRATOR assigns one or more users (including self) for reviewing the deployment configuration
Rationale:	Automated quality tests have to be complemented with the input from humans, who must be able to review the model, the parameters affecting the deployment, and also possibly the results of the quality tests.
Supporting material:	N/A
Other comments:	N/A

Table 181: The Build acceptance Requirement.

ID:	R5.20
Title:	Build acceptance
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The CI_TOOLS MUST NOT run the deployment of the application to pre-production if the quality test fail or the reviewers have not provided a positive score.
Rationale:	No build should be promoted to pre-production accidentally. ADMINISTRATOR or other actor has to have the means to block harmful updates.
Supporting material:	N/A
Other comments:	N/A

Table 182: The Continuous integration tools access control Requirement.

ID:	R5.22
Title:	Continuous integration tools access control
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The access to CI_TOOLS SHOULD be protectable with good credentials (e.g., username and password or a single sign-on token)
Rationale:	In the environments where the access to code and the builds need to be restricted to only the authorised staff, the

	CI_TOOLS should enable setting up of accounts, roles of accounts, and prevent access to unauthorised users.
Supporting material:	N/A
Other comments:	N/A

Table 183: The Continuous integration tools IDE integration Requirement.

ID:	R5.23
Title:	Continuous integration tools IDE integration
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The CI_TOOLS MUST be integrated with the IDE.
Rationale:	The continuous integration tools must provide the means to be invoked remotely, with an option of controls and status display built into the IDE.
Supporting material:	N/A
Other comments:	N/A

Table 184: The Running tests from IDE without committing to VCS Requirement.

ID:	R5.23.1
Title:	Running tests from IDE without committing to VCS
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The CI_TOOLS COULD provide an integration with the IDE that enables deployment and execution of tests on the user's local changes without committing the code into the VCS.
Rationale:	In some cases the DEVELOPER may want to run a test without committing the code into the repository.
Supporting material:	N/A
Other comments:	N/A

Table 185: The Flexiant platform simulated or induced faults Requirement.

ID:	R5.24

Title:	Flexiant platform simulated or induced faults
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The fault injection environment MUST enable simulating or inducing at least the following platform faults: High CPU usage, High Memory usage, Node Power outage, Network outage/ fault, Lack of resources
Rationale:	One set of problems an application may encounter is that a part of the host's resources are exhausted. The fault injection environment in DICE will provide a controled and reliable way of inducing resource ourages.
Supporting material:	N/A
Other comments:	N/A

Table 186: The Configuration Optimization Requirement.

ID:	R5.27
Title:	Configuration Optimization
Priority of accomplishment:	Must have
Type:	Requirement
Description:	CONFIGURATION_OPTIMIZATION MUST use the initial configuration parameters provided upstream by the QA_TESTER and find optimal values of the parameters that have been selected by the QA_TESTER.
Rationale:	Data intensive systems comprise of several frameworks such as Hadoop, Storm, Spark, each of which have many different configuration parameters. However, the default parameters are typically used which are suboptimal comparing with the optimum ones.
Supporting material:	N/A
Other comments:	A requirement for CONFIGURATION_OPTIMIZATION tool. Changed to highlight that CO is used by other tools, so it's their responsibility to supply the initial data

Table 187: The Brute-force approach for CONFIGURATION_OPTIMIZATION deployment Requirement.

ID:	R5.27.1

Title:	Brute-force approach for CONFIGURATION_OPTIMIZATION deployment
Priority of accomplishment:	Should have
Type:	Requirement
Description:	CONFIGURATION_OPTIMIZATION SHOULD apply intelligent ML methods in order to enable a sequential decision making approach that selects a promising configuration setting at each iteration. CONFIGURATION_OPTIMIZATION should find the best possible configuration at the end within the experimental budget specified by the QA_TESTER.
Rationale:	Alternative to ML approach
Supporting material:	N/A
Other comments:	N/A

Table 188: The CONFIGURATION_OPTIMIZATION API Requirement.

ID:	R5.27.2
Title:	CONFIGURATION_OPTIMIZATION API
Priority of accomplishment:	Could have
Type:	Requirement
Description:	CONFIGURATION_OPTIMIZATION COULD provide APIs to access CO functionalities (run, push data, get optimum configuration, etc.)
Rationale:	N/A
Supporting material:	N/A
Other comments:	A command-line interface can will work in the integration.

Table 189: The Starting the CONFIGURATION_OPTIMIZATION Requirement.

ID:	R5.27.3
Title:	Starting the CONFIGURATION_OPTIMIZATION
Priority of accomplishment:	Must have
Type:	Domain Assumption

Description:	The CONFIGURATION_OPTIMIZATION tool is invoked by the CI_TOOLS or by the QA_TESTER
Rationale:	Addresses the responsibility of executing the programs or scripts, which implement the CO runs.
Supporting material:	N/A
Other comments:	N/A

Table 190: The Optimization run independence Requirement.

ID:	R5.27.4
Title:	Optimization run independence
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The CONFIGURATION_OPTIMIZATION MUST ensure that no side effects from past or ongoing optimizations leak into the runtime of any other tests.
Rationale:	Each experiment needs to be run independently from the others. The experimental results should be as repeatable as possible.
Supporting material:	N/A
Other comments:	N/A

Table 191: The CONFIGURATION_OPTIMIZATION Outcome Requirement.

ID:	R5.27.5
Title:	CONFIGURATION_OPTIMIZATION Outcome
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The CONFIGURATION_OPTIMIZATION MUST provide the most optimal configuration outcome given the search budget.
Rationale:	The outcome of each CO run lead to a optimum options for several configuration parameters.
Supporting material:	N/A
Other comments:	N/A

Table 192: The CONFIGURATION_OPTIMIZATION experiment runs Requirement.

ID:	R5.27.6
Title:	CONFIGURATION_OPTIMIZATION experiment runs
Priority of accomplishment:	Must have
Type:	Requirement
Description:	CONFIGURATION_OPTIMIZATION MUST be able to derive the experiment by running the application under test with specific configuration setting by contacting DEPLOYMENT_TOOL. CONFIGURATION_OPTIMIZATION MUST be able to retrieve the monitoring data regarding the experiments by contacting MONITORING_PLATFORM.
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 193: The Configuration optimization of the system under test over different versions Requirement.

ID:	R5.27.7
Title:	Configuration optimization of the system under test over different versions
Priority of accomplishment:	Should have
Type:	Requirement
Description:	CONFIGURATION_OPTIMIZATION SHOULD be able to utilize the performance data that have been collected regarding previous versions of the system under test in the delivery pipeline.
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 194: The Configuration Optimization's input and output Requirement.

ID:	R5.27.8
Title:	Configuration Optimization's input and output

Priority of accomplishment:	Must have
Type:	Requirement
Description:	CONFIGURATION_OPTIMIZATION MUST be able to receive a TOSCA blueprint, which describes the application under test including any initial configuration. It MUST return a TOSCA blueprint updated with optimal parameters, or a stand-alone configuration file.
Rationale:	N/A
Supporting material:	N/A
Other comments:	N/A

Table 195: The Induced faults in the guest environment Requirement.

ID:	R5.30
Title:	Induced faults in the guest environment
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The TESTBED COULD enable simulating or inducing at least the following VM Level faults: High CPU usage, High Memory usage, Network fault
Rationale:	N/A
Supporting material:	N/A
Other comments:	

Table 196: The Reactions to problems in the runtime Requirement.

ID:	R5.31
Title:	Reactions to problems in the runtime
Priority of accomplishment:	Could have
Type:	Requirement
Description:	The DEPLOYMENT_TOOLS COULD provide the means to trigger special actions such as reconfiguration or problem notifications when problems are detected
Rationale:	N/A

Supporting material:	N/A
Other comments:	

Table 197: The Testbed problem notifications Requirement.

ID:	R5.32
Title:	Testbed problem notifications
Priority of accomplishment:	Should have
Type:	Requirement
Description:	The TESTBED SHOULD output notifications of faults to at least one of the regular channels (RESTful URL subscription, e-mail, queue...)
Rationale:	The TESTBED needs to provide the means for sending notifications when it detects faults regardless of whether they occur deliberately or accidentally.
Supporting material:	N/A
Other comments:	

Table 198: The Practices and patterns for security and privacy Requirement.

ID:	R5.43
Title:	Practices and patterns for security and privacy
Priority of accomplishment:	Must have
Type:	Requirement
Description:	The DEPLOYMENT_TOOLS MUST enable applying practices and patterns to ensure that the deployed application is reasonably secure and protecting privacy.
Rationale:	Protecting privacy and security in Big Data applications is vital in production, thus measures to uphold them it need to be introduced during the development already
Supporting material:	N/A
Other comments:	New requirement

3. Change history

In the previous chapter we provided the latest snapshot of the requirements at the time of publishing this document. In Table 199 the we provide the history of changes for the requirements.

Table 199: History of requirements changes

Workpackage	Change description	Date of change
WP2-PMI-Rq	PMI added Requirement/Domain-Assumption PR2.16	05/18/2016
WP2-PMI-Rq	PMI added Requirement/Domain-Assumption PR2.17	05/18/2016
WP2-PMI-Rq	PMI added Requirement/Domain-Assumption PR2.18	05/18/2016
WP2-PMI-Rq	PMI added Requirement/Domain-Assumption PR2.19	05/18/2016
WP2-PMI-Rq	PMI added Requirement/Domain-Assumption PR2.20	05/18/2016
WP2-PMI-Rq	PMI added Requirement/Domain-Assumption PR2.21	05/18/2016
WP2-PMI-Rq	PMI added Requirement/Domain-Assumption MR2.0	05/18/2016
WP6-NETF-Rq	NETF updated Requirements NETF.1	05/18/2016
WP6-NETF-Rq	NETF updated Requirements NETF.2	05/18/2016
WP6-NETF-Rq	NETF updated Requirements NETF.3	05/18/2016
WP6-NETF-Rq	NETF updated Requirements NETF.4	05/18/2016
WP6-NETF-Rq	NETF updated Requirements NETF.5	05/18/2016
WP6-NETF-Rq	NETF updated Requirements NETF.6	05/18/2016
WP6-NETF-Rq	NETF updated Requirements NETF.7	05/18/2016
WP6-NETF-Rq	NETF added Requirements NETF.8	05/18/2016
WP6-NETF-Rq	NETF added Requirements NETF.9	05/18/2016
WP6-NETF-Rq	NETF added Requirements NETF.10	05/18/2016
WP6-NETF-Rq	NETF added Requirements NETF.11	05/18/2016
WP6-NETF-Rq	NETF added Requirements NETF.12	05/18/2016
WP6-ATC-Rq	ATC added Requirements ATC.13	05/18/2016
WP6-ATC-Rq	ATC updated Requirement ATC.8	05/18/2016
WP5-Rq	IMP updated Requirement R5.27	05/24/2016
WP5-Rq	IMP updated Requirement R5.27.1	05/24/2016
WP5-Rq	IMP updated Requirement R5.27.2	05/24/2016
WP5-Rq	IMP added Requirement R5.27.3	05/24/2016
WP5-Rq	IMP added Requirement R5.27.4	05/24/2016
WP5-Rq	IMP added Requirement R5.27.5	05/24/2016

WP5-Rq	IMP added Requirement R5.27.6	05/24/2016
WP5-Rq	IMP added Requirement R5.27.7	05/24/2016
WP5-Rq	XLAB updated Requirement R5.4	05/25/2016
WP5-Rq	XLAB updated Requirement R5.4.1	05/25/2016
WP5-Rq	XLAB updated Requirement R5.4.2	05/25/2016
WP5-Rq	XLAB updated Requirement R5.4.3	05/25/2016
WP5-Rq	XLAB updated Requirement R5.4.5	05/25/2016
WP5-Rq	XLAB updated Requirement R5.4.8	05/25/2016
WP5-Rq	XLAB added Requirement R5.4.10	05/25/2016
WP5-Rq	XLAB added Requirement R5.4.11	05/25/2016
WP5-Rq	XLAB updated Requirement R5.7.1	05/25/2016
WP5-Rq	XLAB added Requirement R5.7.2	05/25/2016
WP5-Rq	XLAB removed Requirement R5.21	05/25/2016
WP5-Rq	XLAB added Requirement R5.43	05/25/2016
Actors	IMP added CONFIGURATION_OPTIMIZATION	05/27/2016
Actors	FLEXI changed TESTBED to FAULT_INJECTION_TOOL	05/27/2016
WP6-PRO-Rq	PRO updated Requirement PO.5	05/27/2016
WP6-PRO-Rq	PRO updated Requirement PO.7	05/27/2016
WP6-PRO-Rq	PRO updated Requirement PO.8	05/27/2016
WP6-PRO-Rq	PRO updated Requirement PO.9	05/27/2016
WP6-PRO-Rq	PRO updated Requirement PO.11	05/27/2016
WP6-PRO-Rq	PRO updated Requirement PO.13	05/27/2016
WP6-PRO-Rq	PRO deleted Requirement PO.6	05/27/2016
WP6-PRO-Rq	PRO deleted Requirement PO.10	05/27/2016
WP6-PRO-Rq	PRO deleted Requirement PO.14	05/27/2016
WP6-PRO-Rq	PRO deleted Requirement PO.15	05/27/2016
WP4-Rq	IEAT filled-in Tools column	05/27/2016
WP4-Rq	IEAT splitted ANOMALY_TRACE_TOOLS into ANOMALY_DETECTION_TOOL and TRACE_CHECKING_TOOL	05/27/2016
WP4-Rq	IEAT revised the priority for R4.6	05/27/2016

WP1-Rq	PRO updated Requirement R1.2	05/27/2016
WP1-Rq	PRO updated Requirement R1.7.1	05/27/2016
WP4-Rq	IMP added Requirement 4.37	05/29/2016
WP4-Rq	IMP added Requirement 4.36	05/29/2016
WP2-PMI-Rq	PMI added TOOLS column for all requirements	05/30/2016
WP2-PMI-Rq	PMI added Requirement/Domain-Assumption PR2.12b	05/30/2016
WP6-PRO-Rq	PRO updated Requirement PO.16	05/30/2016
WP5-Rq	XLAB updated Requirement R5.27	05/30/2016
WP5-Rq	XLAB added Requirement R5.27.8	05/30/2016
WP5-Rq	XLAB updated Requirement R5.7	05/30/2016
WP5-Rq	XLAB updated Requirement R5.7.1	05/30/2016
WP5-Rq	XLAB updated Requirement R5.7.2	05/30/2016
WP5-Sc	FLEX updated U5.3	05/30/2016
WP5-Sc	FLEX updated U5.4	05/30/2016
WP5-Sc	FLEX updated U5.9	05/30/2016
WP5-Sc	FLEX updated U5.10	05/30/2016
WP5-Sc	FLEX updated U5.11	05/30/2016
WP5-Rq	FLEX updated Requirement R5.2	05/30/2016
WP5-Rq	FLEX updated Requirement R5.4.3	05/30/2016
WP5-Rq	FLEX updated Requirement R5.14.2	05/30/2016
WP5-Rq	FLEX updated Requirement R.5.24	05/30/2016
WP4-Rq	PMI revised and updated from R4.28 to R4.33	05/30/2016
WP4-Rq	IEAT removed R4.5 "Data retention policy"	05/30/2016
WP1-Rq	PRO updated Requirement R3IDE.1	05/30/2016
WP1-Rq	PRO updated Requirement R3IDE.2	05/30/2016
WP1-Rq	PRO updated Requirement R3IDE.4	05/30/2016
WP3-Rq	ZAR marked Requirement R3.5 as DEPRECATED	05/30/2016
WP3-Rq	ZAR marked Requirement R3.12 as DEPRECATED	05/30/2016
WP2-PMI-Rq	PMI added Requirement/Domain-Assumption PR 2.22	05/31/2016
WP4-Rq	IMP deleted requirement R4.23	05/31/2016
WP2-PMI-Rq	PMI added Requirement/Domain-Assumption PR 2.23	06/01/2016

WP2-PMI-Rq	PMI added Requirement/Domain-Assumption PR 2.24	06/02/2016
WP4-Rq	IMP changed requirement R4.17 to Must Have	07/17/2016
WP1-Rq	ZAR updated Requirement R3IDE.7	09/12/2016
WP4-Rq	IEAT added R4.38	12/12/2016
WP4-Rq	IEAT updated R4.1	12/12/2016
WP4-Rq	IMP deprecated R4.26.1	12/12/2016
WP5-Rq	IMP updated Requirement R5.6	12/14/2016
WP5-Rq	IMP updated Requirement R5.8	12/14/2016
WP5-Rq	IMP deprecated Requirement R5.8.1	12/14/2016
WP5-Rq	IMP updated Requirement R5.13	12/14/2016
WP3-Rq	PMI updated Requirement R3.15	12/15/2016
WP3-Rq	PMI updated Requirement R3IDE.4.2	12/15/2016
WP4-Rq	PMI updated Requirement R4.28	12/15/2016
WP4-Rq	IEAT added R4.39	12/18/2016
WP4-Rq	IEAT added R4.40	12/18/2016
WP4-Rq	IEAT added R4.41	12/18/2016
WP5-Rq	IMP revised the title of Requirement R5.6	01/22/2017
WP5-Rq	IMP updated Requirement R5.14.2	01/22/2017
WP5-Rq	XLAB updated description of R5.4.2	01/22/2017
WP1-Rq	XLAB added R5IDE2	01/30/2017
WP1-Rq	IMP decreased priority of R1.6 from Should have to Could have	01/30/2017
WP5-Rq	IMP decreased priority of R5.27.2 from Must have to Could have	01/30/2017