VELaSSCo

Visual Analysis for Extremely Large-Scale Scientific Computing

D5.1. Evaluation methodology Version #3.0

Deliverable Information

Grant Agreement no	619439
Web Site	http://www.velassco.eu/
Related WP & Task:	WP5 - Usability and Effectiveness Evaluation Task 5.1 - Evaluation methodology
Due date	30/06/2015
Dissemination Level	Public
Nature	Report
Author/s	Ivan Martinez, Miguel Angel Tinte
Contributors	Alavaro Janda, Giuseppe Filippone, Miguel A. Pasenau de Riera, Abel Coll, Javier Mora, Tomas Pariente

Approvals

	Name	Institution	Date	OK
Author	Miguel Angel Tinte, Ivan Martínez	**ATOS**	30/06/2015	
Task Leader	Ivan Martínez	**ATOS**	30/06/2015	
WP Leader	Ivan Martínez	**ATOS**	30/06/2015	
Coordinator	Abel Coll	**CIMNE**	30/06/2015	
Quality Check	Heidi E. I. Dahl	**SINTEF**	29/06/2015	

Table of Contents

1 Introduction

1.1 Purpose of the document

The main purpose of this document is to provide a measuring plan and the associated methodology to perform the evaluation of the VELaSSCo framework.

This document is based on the methodology selected in WP1, described in the deliverable "D1.5 Definition of criteria and methodology for system evaluation" [2] , which defines the approach for the system evaluation and explains the choice of the metrics to interpret evaluation results.

The evaluation methodology selected in that document is the GQM methodology which provides a measurement model on three levels: conceptual (Goals), operational (Questions) and quantitative (Metrics).

Deliverable D1.5 presented a tentative description of the evaluation plan. The main goal of this document is to provide an exhaustive vision of the different evaluation processes as well as completing the definition of measurement levels description. To do so, it will be necessary not only to identify the goals of the evaluation, but also to make explicit both questions and metrics. The definition of metrics is particularly interesting, because it provides the quantitative evaluation which can be classified through different thresholds, and can be an object of comparison among different iterations of the evaluations.

This document presents also the measurement plan and the process of collecting data to perform the actual evaluation, hence covering all aspects of the GQM methodology except the evaluation itself. Therefore, this document meets with the objectives from Task T5.1 of WP5 regarding Evaluation methodology; it provides an overview of system architecture, evaluation methodologies, developing metrics and performance indicators to measure and reviewing similar solutions for DEM[1] and FEM[2] simulations as a baseline for comparison. The evaluation of the framework is covered in other WP5 deliverables (for tasks T5.2, T5.3, T5.4 and T5.5) that applies the measurement plan and analyse the results for each of the evaluation dimensions: Verification of system architecture, verification of algorithms and implementations, effectiveness evaluation of real-time data access/visualization and usability evaluation.

1.2 Structure of the document

The document is structured as follows:

Section1 gives a brief introduction and outlines the major purpose of the document.

[1] https://en.wikipedia.org/wiki/Discrete_element_method

[2] https://en.wikipedia.org/wiki/Finite_element_method

Section 2 recaps on the main aspects of the methodology selected for the evaluation of the VELaSSCo framework and provides an overview of how it will be applied in an iterative fashion.

Section 3 is the core section of the document. It provides a detailed description of how the metrics to evaluate the system are derived according to the methodology. It also provides an overview of the measurement plan to get the values of the metrics and hints how to interpret the results.

Section 4 provides a general overview of practical aspects to prepare the evaluation for a particular iteration. It follows the evaluation methodology defined in section 2 covering aspects such as the definition of the objectives of the iteration, the preparation of the setting (infrastructure, machines), the outline of the tasks given to users, the preparation of the analysis instruments (logs, questionnaires, etc.) and finally the delivery of a clear evaluation plan for the iteration.

Section 5 concludes with consolidated findings and reports on the next steps.

Section 6 contains the references.

Section 7 provides a set of annexes, in particular related to the initial set of questionnaires and material for performing the evaluation.

2 Evaluation Planning

Deliverable D1.5 provided an overview of the methodology selected (GQM) for the evaluation of the VELaSSCo framework. This section recaps on the main aspects of GQM and provides a picture of the main aspects to be taken into account to apply correctly GQM for the evaluation process.

2.1 GQM life-cycle

An overview of the methodology to evaluate the system was described in deliverable D1.5 [2] . This document complements that deliverable by describing in detail the application of the GQM methodology life-cycle to evaluate the VELaSSCo framework. The GQM methodology evaluation process starts by defining the right Goals, Questions and Metrics to assess the quality and effectiveness of the system. While the system evaluation follows an evaluation plan through consecutive phases, GQM follows an iterative approach in which each iteration revisits the GQM metrics and provides feedback to improve the system. Thus it is a spiral software development methodology which provides a sequence of steps through which the methodology will be applied.

The GQM steps are displayed below in Figure 1, which will guide us through the system evaluation process:

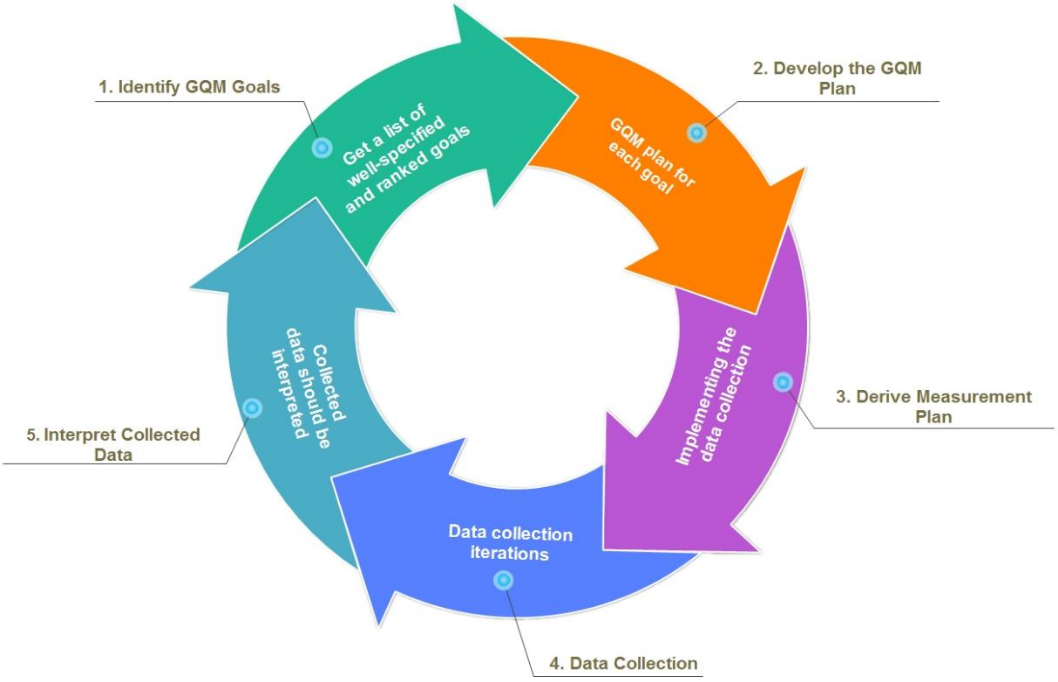

1. Identify GQM Goals
Get a list of well-specified and ranked goals
2. Develop the GQM Plan
GQM plan for each goal
3. Derive Measurement Plan
Implementing the data collection
4. Data Collection
Data collection iterations
5. Interpret Collected Data
Collected data should be interpreted

Figure 1. VeLASCCo GQM standard cycle [2]

Figure 1 shows the steps defined in GQM, which are the following:

1. **Identify GQM Goals**: Develop a set of project business goals and associated measurement goals for productivity and quality.
2. **Develop the GQM Plan**: Generate questions that define those goals as completely as possible in a quantifiable way.
3. **Derive the Measurement Plan**: Specify the measures needed to be collected to answer those questions and track process and product conformance to the goals
4. **Data collection**: Develop mechanisms for data collection in order to cover the different metrics proposed. Collect and validate the data.
5. **Interpret collected data**: Analyse the data to assess conformance to the goals and to make recommendations for future improvements.

2.2 Evaluation dimensions

The Evaluation Plan is based on the GQM Evaluation Framework approach (Figure 2) defined in the deliverable D1.5 including the following sequence of tasks: End-User Functionalities Evaluation, SW Architecture and Deployment Environment Evaluation, Algorithms Evaluation, Navigation and Interaction Evaluation (usability) and Views Evaluation (effectiveness).

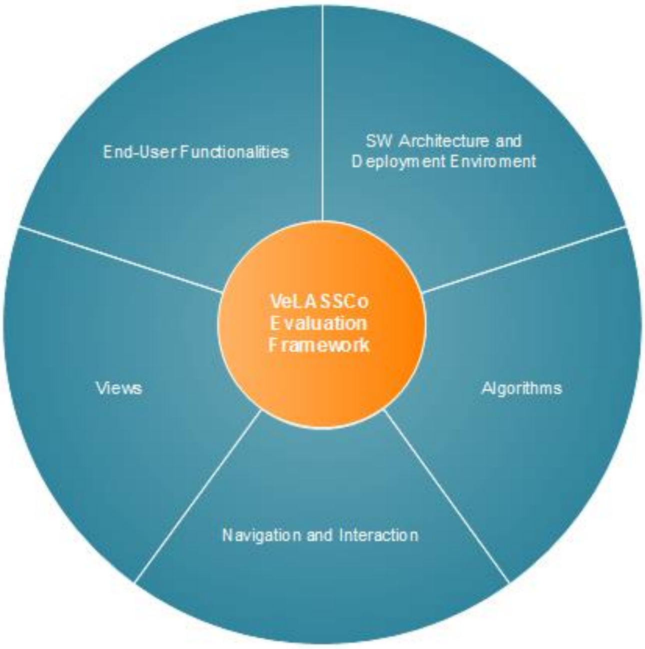
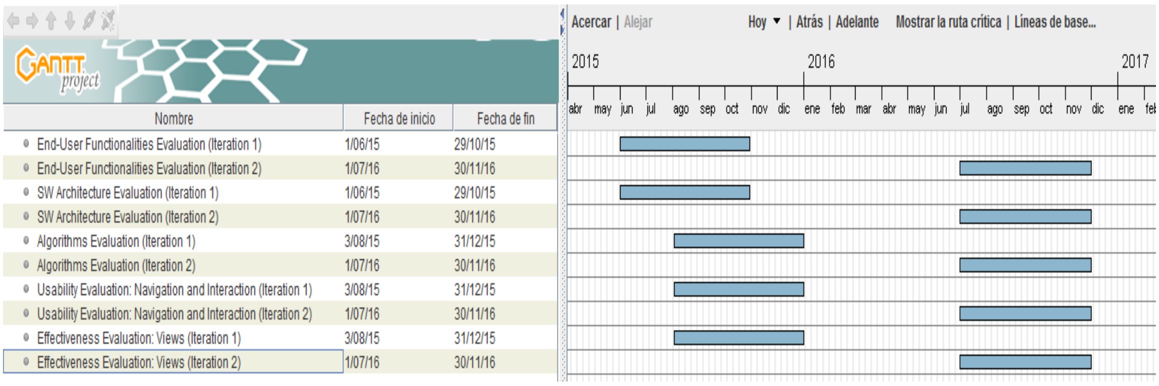

End-User Functionalities
SW Architecture and Deployment Enviroment
VeLASSCo Evaluation Framework
Views
Algorithms
Navigation and Interaction

Figure 2. VELaSSCo Evaluation Framework Dimensions [2]

The scheduling of the Evaluation tasks associated with each one of the five dimensions defined in the Evaluation Framework is presented in Figure 3. The GQM paradigm will be applied to each of the five key dimensions to produce a GQM plan and a Measurement Plan associated for each dimension.



Nombre	Fecha de inicio	Fecha de fin
End-User Functionalities Evaluation (Iteration 1)	1/06/15	29/10/15
End-User Functionalities Evaluation (Iteration 2)	1/07/16	30/11/16
SW Architecture Evaluation (Iteration 1)	1/06/15	29/10/15
SW Architecture Evaluation (Iteration 2)	1/07/16	30/11/16
Algorithms Evaluation (Iteration 1)	3/08/15	31/12/15
Algorithms Evaluation (Iteration 2)	1/07/16	30/11/16
Usability Evaluation: Navigation and Interaction (Iteration 1)	3/08/15	31/12/15
Usability Evaluation: Navigation and Interaction (Iteration 2)	1/07/16	30/11/16
Effectiveness Evaluation: Views (Iteration 1)	3/08/15	31/12/15
Effectiveness Evaluation: Views (Iteration 2)	1/07/16	30/11/16

Figure 3. Evaluation tasks Gantt Diagram

The evaluation of the results will be performed in several iterations. The major iterations will coincide with the release of the major software components (Iteration 1 in M25 and Iteration 2 in M36), but smaller iterations which take place during the whole life-time of the project.

The evaluation will be carried out by professionals from the different domains, as specified in the use case scenarios, both for the usability and for the effectiveness and performance of the proposed solutions.

2.3 Study set-up and methodology

This section attempts to describe important aspects for the configuration of the methodology as: Evaluation iterations, Objectives of the evaluation for each iteration, Evaluation Roles, and Practical methodology for evaluation deciding responsible and participants.

Related to Evaluation iterations, we could say that the evaluation of the results will be performed in several iterations. The major iterations will coincide with the release of the major software components (Iteration 1 in M25 and Iteration 2 in M36), but smaller iterations which take place during the whole life-time of the project. Iteration 1 coincides with the delivery of the first prototype (Early version) of the platform, and Iteration 2 aims the evaluation of the full version (Release version) of the platform.

In both cases the objectives for each one of the iterations is to ensure the usability, effectiveness and performance of the proposed solution. In addition specific objectives associated to the first prototype of the platform consists to enrich the first architecture specification and provide more specific recommendations for the following developments, specifically for large scaling requirements.

In the evaluation process we will find the following roles:

- User testing population: mainly user panel and consortium members.
- Facilitators: in charge of be helping the tester users to perform the tasks defined and taking notes of the users' behaviour.
- Analysts: in charge of analysing the results. ATOS staff and other members of the consortium will take this role.

Finally, the practical aspects of the evaluation process are listed deciding for each one of them who is responsible and participants:

- (Re)Definition of GQM tables: lead by ATOS next to participation of the WP5 tasks leaders.
- Assessment of the VELaSSCo tools available for Iteration 1 evaluation: Decided in Barcelona Consortium meeting 2015 (minutes in https://alfresco.cimne.upc.edu/share/page/site/velassco/document-details?nodeRef=workspace://SpacesStore/0a22cae0-b339-4ce3-9606-277630048a8e)
- Infrastructural needs and tools:
  - Physical infrastructure: CIMNE HPC Cluster with 9 nodes
  - Tools: Visualization Clients (IFX and GiD), VELaSCCo Platform, Big Data Ecosystem (Flume, Hbase, Hive, Hadoop), EDM, Nagios and Nagios Network Analyzer.
  - Questionnaires: Usability and Effectiveness questionnaires are provided.
- Definition of evaluation tasks:
  - The evaluation tasks are based on Telescope Use case (FEM) and Fluidized Bed Use case (DEM):

- *FEM Evaluation tasks (T1…10):*

- T1: Connect to VELaSSCo
- T2: Open a simulation model (*model FEM.M1.)*
- T3: Select coarser mesh.
    - T3.1: *Select coarser mesh for all time steps.*
- T4: Rotate model.
- T5: Select original mesh
- T6: Get the evolution of a result on a node over time.
    - T6.1: *Get the **pressure** value of **node number 5** for **all time steps***
- T7: Visualize a contour fill of a result.
    - *T7.1: Visualize the contour fill of **pressure** in the skin of the volume mesh in **time step 8***
- T8: Do a cut in the volume mesh.
    - *T8.1: Do and visualize a cut in the volume mesh, parallel to AA direction, and passing through (x, y, z) coordinates.*
- T9: Visualize a result onto the cut plane
    - T9.1: Visualize the velocity vectors onto the cut plane in time step 7.
- T10: Logout.

- *DEM Evaluation tasks (T1…11):*

- T1: Connect to VELaSSCo
- T2: Open a simulation model (*model DEM.M1.)*
- T3: Visualize a contour fill of a particle result.
    - *T3.1: Visualize the **velocity-Y** in the skin of the particles for **time step 2939000***
- T4: Rotate model.
- T5: Get the evolution of a result on a particle over time.
    - *T5.1Get the velocity y-component value for:*
        - *Analysis = **DEM***
        - *Coordinates = **Particles***
        - *Time-steps: **ALL***
        - *Result = **Velocity-Y***
        - *Node number **2724***
- T6: Visualize p2p contacts.
    - *T6.1: Visualize the **p2p contacts** mesh for **time step 2939000***

- T7: Visualize a contour fill of a p2p result.
    - *T7.1: Visualize the **Force-Y** in the skin of the p2p contacts for **time step 2939000***
- T8: Compute d2c of the model
    - *T8.1: Compute discrete to continuum for:*
        - *Static mesh = d2c_1*
        - *D2C analysis name = d2C_FB2*
        - *Time-step options = ALL*
        - *Coarse-graining method = Gaussian*
        - *Coarse-graining options:*
        - *Width = 0.003*
        - *Cut-off factor = 3*
        - *Process contacts = True*
        - *Do temporal averaging = True*
        - *Temporal averaging options = ALL*
- T9: Do a cut plane in the d2c mesh.
    - *T9.1: Do and visualize a cut in the d2c mesh, parallel to Y direction, and passing through (0, 0, 0) coordinates.*
- T10: Visualize a result of d2c onto the cut plane
    - T10.1: Visualize the **Velocity-Y** onto the cut plane for computed d2c in time step ***0***
- T11: Logout.

- Mapping the tasks to specific measurements:
    - Quantitative: we have prepared the evaluation framework to record the measurements adding logs and traces associated to each one of the SW component of VELaSCCo Architecture. Besides this we will use Nagios and Nagios Network Analyser as a monitoring global tool in the VELaSCCo Platform which let us get most of quantitative metrics defined in the GQM metric tables associated to each one of the evaluation dimensions.
    - Qualitative:
        - We will use a Usability and Effectiveness evaluation for the evaluation of the first prototype.
        - Mapping specific metrics to questions in questionnaires
        - Prepare and instruct facilitators to help users and record impressions, doubts, comments, suggestions, etc.
- Perform the evaluation:
    - Record the measurements and data from questionnaires
    - Analyse the measurements and map them to metrics
    - Use facilitators feedback for qualitative evaluation
    - Conclusions and feedback of the quantitative and qualitative evaluation

3 GQM Cycle application to framework

The cornerstone of the methodology is the correct application of the GQM to derive the right metrics to assess the goodness of the system. This section provides a detailed explanation of the application of GQM to derive the metrics needed to evaluate the VELaSSCo framework in each of the evaluation dimensions (End-User Functionalities Evaluation, SW Architecture and Deployment Environment Evaluation, Algorithms Evaluation, Usability and Effectiveness) listed in section 2.2. The full GQM cycle for the each of the evaluations dimension is shown in Figure 4:

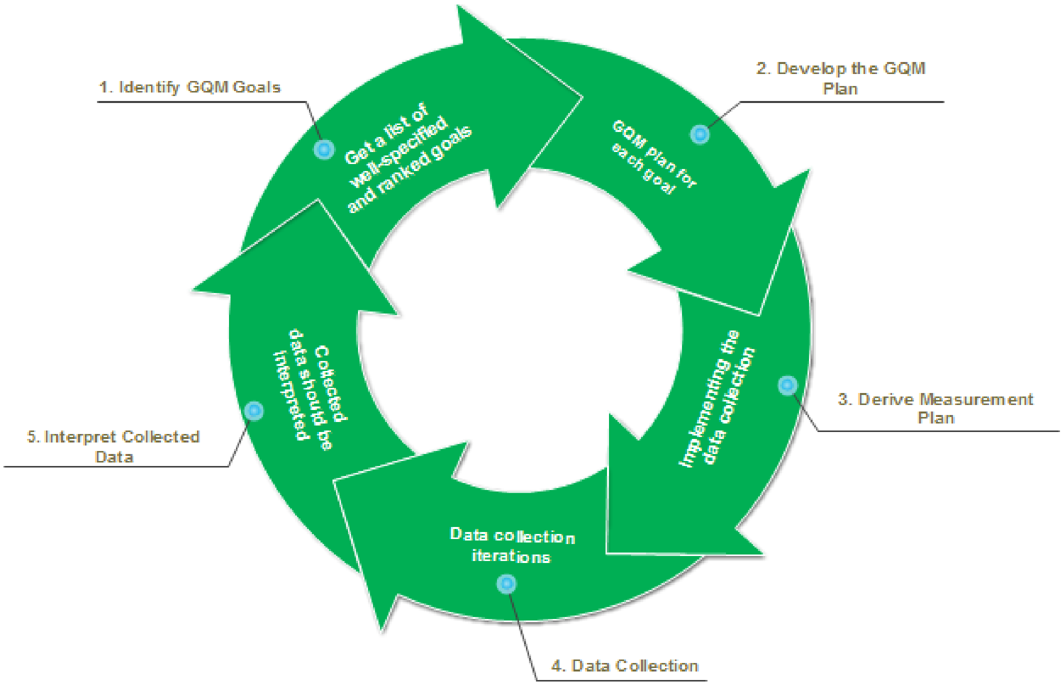

1. Identify GQM Goals
Get a list of well-specified and ranked goals
2. Develop the GQM Plan
GQM plan for each goal
3. Derive Measurement Plan
Implementing the data collection
4. Data Collection
Data collection iterations
5. Interpret Collected Data
Collected data should be interpreted

Figure 4. GQM Full cycle application over each of the evaluation dimensions

Therefore, this section focuses on the application of the GQM methodology for each of the dimensions of the system as follows:

1. Provide a table with the **definition of the business goals** to assess the quality and productivity of the system.
2. List a **set of questions** that define those goals as completely as possible in a quantifiable way.
3. Create a list of **measurable metrics** associated to each of those questions.
4. This section provides also **instructions on how to get the measurements** (tools and methods) to assign values to the metrics for the different dimensions of the evaluation, covering aspects such as **data collection and interpretation**.

The actual evaluation of the VELaSSCo framework will be performed in several iterations according to the evaluation plan and will be covered in future WP5 deliverables.

It is worth noticing that due to the availability of elements and the expected evolution of the VELaSSCo framework, not all the metrics make sense in each of the iterations. The evaluation should take into account several aspects such as dimensioning of the platform (going from the first standalone deployment to a final scalable and distributed system), complexity, amount of data, availability of components, etc. GQM iterative steps will be described specifically for each of the proposed dimensions. The content of these tables, especially the metrics that can be evaluated, will be continuously updated in future evaluations as the system evolves.

3.1 End-User Functionalities Dimension

The End-User Functionalities dimension refers to business actions offered by VELaSSCo system from the end user perspective. This section aims to describe how the full GQM cycle can be assessed and reported for this dimension. To do so, the first step is checking and redefining the Goals, Questions and Metrics defined in deliverable [2] . Once the refinement process is finished, a GQM table is presented along with data collection tasks achieved and how this collected data can be interpreted.

This dimension contains some of the VQueries components which have just started to be developed. The Goal/Question/Metric description below takes into account these first results in order to optimize the methodology implementation for future evaluations.

Also, this implementation has been based on the use of the Acuario cluster provided by CIMNE[3]

3.1.1 Identification of GQM Goals

In order to identify the Goals for End-User dimension the approach followed has been matching main Use Cases functionalities for DEM and FEM with Goals to be identified. Once do that, the goals obtained in step 1 are listed in Table 1:

Goal	Description	WP linked to
G.EU#1	Inject Data into VELaSCCo Platform coming from DEM and FEM simulation files.	WP2
G.EU#2	Connect to VELaSSCo Platform	WP2, WP3
G.EU#3	Open a simulation Model	WP2, WP3
G.EU#4	Select Coarser Mesh for all time steps	WP2, WP3
G.EU#5	Rotate Model	WP2, WP3

[3] https://hpc.cimne.upc.edu/portfolio-view/acuario-cluster/

G.EU#6	Select Original Mesh	WP2, WP3
G.EU#7	Get the evolution of a result on a node over time.	WP2, WP3, WP4
G.EU#8	Visualize a Contour Fill of a result	WP2, WP3, WP4
G.EU#9	Do a cut in the Volume Mesh	WP2, WP3, WP4
G.EU#10	Visualize a result onto a cut plane	WP2, WP3, WP4
G.EU#11	Disconnect the VELaSSCo Platform	WP2, WP3, WP4

Table 1. GQM Goals for End-User Functionalities

3.1.2 Development of GQM Plan

The Model (set of questions) obtained in step 2 has been described again after aligning them with new goals identified above. Questions identified so far are listed in Table 2:

Question	Description	Associated Goal
Q.EU#1	What is the size of the data files?	G.EU#1
Q.EU#2	How many particles are involved in the simulation?	G.EU#1
Q.EU#3	How many time steps per simulation?	G.EU#1
Q.EU#4	How many results/variable can be handled?	G.EU#1
Q.EU#5	How can data file size be optimized?	G.EU#1
Q.EU#6	Should be the user previously registered on the platform?	G.EU#2
Q.EU#7	How we know that the user has successfully authenticated to the platform?	G.EU#2
Q.EU#8	How long does the opening model take?	G.EU#3
Q.EU#9	How the simplified model is calculated?	G.EU#4

Q.EU#10	Where the simplified model is stored?	G.EU#4
Q.EU#11	What should be an acceptable rotation time for the model?	G.EU#5
Q.EU#12	Is the rotation process dependant of the visualization client (GiD, IFX)?	G.EU#5
Q.EU#13	How long does getting model with draw data take?	G.EU#6
Q.EU#14	How long does getting the result on a node over time taking into account one node all the time steps?	G.EU#7
Q.EU#15	How long does getting the contour fill for a concrete result?	G.EU#8
Q.EU#16	How long does getting a cut in a volume mesh?	G.EU#9
Q.EU#17	How long does getting a cut in a volume mesh with results?	G.EU#10
Q.EU#18	How we know that the user has successfully logout to the platform?	G.EU#11

Table 2. GQM questions for End-User Functionalities Dimension.

3.1.3 Measurement Plan

The metrics obtained in step 3 are finally calculated again in order to satisfy new goals and questions redefined, in order to provide an empirical assessment report of Use Cases evaluation. They are listed in Table 3:

Metrics	Description	Associated Question	Value
M.EU#1	Simulation File Size	Q.EU#1, Q.EU#5	up to 100MB
M.EU#2	Number of particles	Q.EU#2,	up to 12000

		Q.EU#5	
M.EU#3	Number of computational time steps	Q.EU#3, Q.EU#5	up to 10000
M.EU#4	Number of results at particle level	Q.EU#4, Q.EU#5	Min. of 3 variables (Mass, Volume, Velocity)
M.EU#5	User Credentials	Q.EU#6	User, Password and Group
M.EU#6	Security Token	Q.EU#7	32 alphanumeric characters
M.EU#7	Time of opening model query execution	Q.EU#8	Ms => **VQ002 + VQ010 + VQ012**
M.EU#8	Time of getting simplified mesh query execution	Q.EU#9, Q.EU#10	Ms => **VQ217**
M.EU#9	GiD Model Rotation Velocity	Q.EU#11, Q.EU#12	Degree / Sec.
M.EU#10	IFX Model Rotation Velocity	Q.EU#11, Q.EU#12	Degree / Sec.
M.EU#11	Time of getting original mesh query execution	Q.EU#13	Ms => **VQ114**
M.EU#12	Time of getting result on a node over time	Q.EU#14	Ms => **VQ100**
M.EU#13	Time of getting the contour fill for a concrete result	Q.EU#15	Ms => **VQ214 + VQ100**
M.EU#14	Time of getting a cut in a volume mesh	Q.EU#16	Ms => **VQ215**
M.EU#15	Time of getting a cut in a volume mesh with results	Q.EU#17	Ms => **VQ216**
M.EU#16	User session logout trace	Q.EU#18	User:Session:Action:Boolean

Table 3. GQM Metrics for End-User Functionalities.

3.1.4 Data Collection

Data collection for End-User functionalities dimension has been focused on reviewing all the metrics defined in the previous chapter and calculate them if possible. The aim is not to offer an exhaustive performance analysis, but to offer a first approach to how this dimension can be assessed in future iterations of the evaluation.

Data collection process will be widely reported in each specific dimension evaluation report.

3.1.5 Interpret Collected Data

Data collected in previous table will be interpreted on the specific evaluation report, which will check viability of functionalities required as well as the deployment on HPC cluster. Despite not all the metrics will be calculated, first evaluation will allow us to check first prototype performance. Following iterations will allow increment complexity of infrastructure and extending it to the rest of the cluster nodes in order to obtain more metrics results.

In the specific dimension evaluation reports, the focus will be on improving performance, ensuring scalability by using all the available cluster nodes. Moreover, the use of Radar Charts[4] will allow us compare results from the main tests scenarios after the collection of metrics results.

3.2 SW Architecture and Deployment Environment Dimension

The End Software Architecture and Deployment dimension considers the definition, implementation and deployment in the production environment of the SW pieces integrated by means of an integration framework. This section aims to describe how the full GQM cycle can be assessed and reported for this dimension. To do so, the first step is checking and redefining the Goals, Questions and Metrics defined in deliverable [2] . Once the refinement process is finished, a GQM table is presented along with data collection tasks achieved and how this collected data can be interpreted. The GQM process for architecture dimension is based on ISO 9126-1[5].

3.2.1 Identification of GQM Goals

The approach adopted to identify the goals is using the characteristics of the ISO 9126 quality model. GQM Goals for this dimension were identified and defined in [2] and are listed in Table 4:

Goal	Description	WP linked to
G.AR#1	VELaSCCo Platform Reliable	WP2, WP3, WP4
G.AR#2	VELaSCCo Platform Efficient	WP2, WP3, WP4
G.AR#3	VELaSCCo Platform Maintainable	WP2, WP3, WP4
G.AR#4	VELaSCCo Platform Portable	WP2, WP3, WP4
G.AR#5	VELaSCCo Platform Usable	**WP2, WP3, WP4 (defined as Navigation and Interaction Dimension)**

[4] https://developers.google.com/chart/image/docs/gallery/radar_charts

[5] http://www.iso.org/iso/catalogue_detail.htm?csnumber=22749

G.AR#6	VELaSCCo Platform Functional	**WP2, WP3, WP4 (defined as End User Functionality Dimension)**

Table 4. GQM Goals for SW Architecture

3.2.2 Development of GQM Plan

Besides Goals identification, the GQM Cycle iteration described in this document also aims to define Questions and Metrics.

Question	Description	Associated Goal
Q.AR#1	How we can assess that the VELaSSCO Platform has sufficiently maturity degree for the first version or prototype?	G.EU#1
Q.AR#2	How we can assess that the VELaSSCO Platform has the capability to maintain a desired performance level in case of operational failures?	G.EU#1
Q.AR#3	How we can assess that the VELaSSCO Platform has the capability to re-establish an adequate level of performance and the ability to recover the data directly affected in case of a failure?	G.EU#1
Q.AR#4	How the VELaSSCO Platform will respond (timing-wise) during operation?	G.EU#2
Q.AR#5	How many resources VELaSSCO Platform consumes during testing or operation?	G.EU#2
Q.AR#6	How we can to diagnose deficiencies and causes of failure into VELaSSCO Platform?	G.EU#3
Q.AR#7	How changeable is the VELaSSCO Platform when trying to implement a specified modification?	G.EU#3
Q.AR#8	How stable is the VELaSSCO Platform after modification?	G.EU#3
Q.AR#9	How testable is the VELaSSCO Platform?	G.EU#3
Q.AR#10	How adaptable is the VELaSSCO Platform to different environments?	G.EU#4
Q.AR#11	What is the coexistence of the VELaSSCO Platform?	G.EU#4

Q.AR#12	How replaceable is the VELaSSCO Platform?	G.EU#4

Table 5. GQM Questions for SW Architecture.

3.2.3 Measurement Plan

Besides Goals and Questions, one of the most relevant issues is defining some useful metrics which can provide an overview of Software architecture and deployment performance. The standard ISO-9126 adopted to measure Architecture dimension provides by default a wide set of metrics in order to assess generic software architecture. These lists of metrics cover different and critical aspects of architecture such as fault tolerance, availability, performance, etc.

Table 6 displays the list of metrics identified to assess the software architecture, as well as the goals related to each one of the metric:

Metrics	Description	Type	Associated Question	Value
M.AR#1	Fault detection	Internal	Q.AR#1,	**X = A / B**, where A is the absolute number of bugs detected (from the review report) and B is the estimated number expected.
M.AR#2	Fault removal	Internal	Q.AR#1	**X = A / B,** where A is the number of bugs fixed during design and coding and B is the number that were found during review.
M.AR#3	Estimated latent fault density	External	Q.AR#1	**X = {abs(A1 - A2)} / B,** where A1 is the total number of predicted latent defects in the system, A2 is the total number of actually occurring failures, and B is the product size.
M.AR#4	Failure density against test cases	External	Q.AR#1	**X = A1 / A2,** where A1 is the number of detected failures during the period and A2 is thenumber of executed test cases.
M.AR#5	Failure resolution	External	Q.AR#1	**X = A1 / A2,** where A1 is the total number of failures

				that are resolved and never reoccur during the trial period and A2 is the total number of failures that were detected.
M.AR#6	Fault density	External	Q.AR#1	**X = A / B**, where A is the number of detected failures and B is the system size (ISO 9126-2 does not define how size is measured).
M.AR#7	Fault removal	External	Q.AR#1	**X = A1 / A2, Y = A1 / A3,** where A1 is the number of corrected defects, A2 is the total number of actually detected defects, and A3 is the total number of estimated latent defects in the system. In reality, the first formula is measuring how many found defects are not being removed.
M.AR#8	Mean time between failures	External	Q.AR#1	**X = T1 / A, Y = T2 / A,** where T1 is the total operation time, and T2 is the sum of all the time ntervals when the system was running. The second formula can be used when there were time during the interval when the system was not running. Whichever formula is used, A is the total number of failures that were observed during the time the system was actually operating.
M.AR#9	Failure avoidance	Internal	Q.AR#2	**X = A / B**, where A is the number of fault patterns that were explicitly avoided in the design and code and and B is the number to be considered as defined by the requirements specification document.

M.AR#10	Incorrect operation avoidance	Internal	Q.AR#2	**X = A / B**, where A is the number of incorrect operations that are explicitly designed to be prevented and B is the number to be considered as given in the requirements.
M.AR#11	Breakdown avoidance	External	Q.AR#2	**X = 1 - A / B**, where A is the number of total breakdowns and B is the number of failures.
M.AR#12	Failure avoidance	External	Q.AR#2	**X = A / B**, where A is the number of avoided critical and serious failure occurrences against test cases for a given fault pattern and B is the number of executed test cases for the fault pattern.
M.AR#13	Incorrect operation avoidance	External	Q.AR#2	**X = A / B**, where A is the number of test cases that pass (i.e., no critical or serious failures occur) and B is the total number run.
M.AR#14	Restorability	Internal	Q.AR#3	**X = A / B**, where A is the number of restoration requirements that are found in the review documents and B is the number called for in the requirements or design documents
M.AR#15	Restoration effectiveness	Internal	Q.AR#3	**X = A / B**, where A is the number of implemented restoration requirements meeting the target restore time and B is the total number of requirements that have a specified target time. For example, suppose that the requirements specification not only

				requires the ability to roll back a transaction, but it also defines that it must do so within *N* milliseconds once it is triggered. If this capability is actually implemented, and we calculate/simulate that it would actually work correctly, the metric would equal 1/1. If not implemented, the metric would be 0/1.
M.AR#16	Availability	External	Q.AR#3	**X = { To / (To + Tr)}, Y = A1 / A2,** where To is the total operation time and Tr is the time the system takes to repair itself (such that the system is not available for use); A1 is the total number of test cases that were successful and A2 is the number of total test cases run. X is the total time available (the closer to 1, the more available the system was) while Y is a measure of the number of test cases that showed successful availability of the system (the closer to 1, the better).
M.AR#17	Mean down time	External	Q.AR#3	**X = T / N,** where T is the total amount of time the system is not available and N is the number of observed times the system goes down.
M.AR#18	Mean recovery time	External	Q.AR#3	**X = Sum(T) / N**, where T is the time to recover for each failure and N is the number of test cases that triggered a failing condition for which recovery occurred.

M.AR#19	Restartability	External	Q.AR#3	**X = A / B**, where A is the number of restarts that were performed within the target time and B is the total number of restarts that occurred.
M.AR#20	Restorability	External	Q.AR#3	**X = A / B,** where A is the number of restorations successfully made and B is the number of test cases run based on the requirements.
M.AR#21	Restore effectiveness	External	Q.AR#3	**X = A / B**, where A is the number of test cases where restoration was successfully completed within the target time and B is the number of test cases performed.
M.AR#22	Response Time	External	Q.AR#4	**VQueries response time** defined in EUF Dimension (see metrics table)
M.AR#23	Mean Time to response	External	Q.AR#4	**X = Tmean / TXmean**, where Tmean is the average time to complete the task (for N runs) and TXmean is the required mean time to response. In our case **T=VQuery**.
M.AR#24	Worst case response time	External	Q.AR#4	**X = Tmax / Rmax,** where Tmax is the maximum time any one iteration of the task took and Rmax is the maximum required response time. In our case **T=VQuery**.
M.AR#25	Throughput	External	Q.AR#4	**X = A / T**, where A is the number of completed tasks and T is the observational time period.
M.AR#26	Mean amount of	External	Q.AR#4	**X = Xmean / Rmean**, where

	Throughput			Xmean is the average throughput and Rmean is the required mean throughput
M.AR#27	Worst case throughput ratio	External	Q.AR#4	**X = Tmax / Rmax**, where Tmax is the worst-case time of a single task and Rmax is the required throughput. In our case **T=VQuery**.
M.AR#28	I/O devices utilization	External	Q.AR#5	**X = A / B**, where A is the amount of time the devices are occupied and B is the specified time the system was expected to use them.
M.AR#29	I/O loading limits	External	Q.AR#5	**X = Amax / Rmax**, where Amax is the maximum number of I/O messages from a given number of runs and Rmax is the required maximum number of messages the system was designed to use.
M.AR#30	I/O related errors	External	Q.AR#5	**X = A / T**, where A is the number of warning messages or errors encountered and T is the user operating time.
M.AR#31	Mean I/O fulfilment ratio	External	Q.AR#5	**X = Amean / Rmean**, where Amean is the average number of I/O error messages and failures over a number of runs and Rmean is the required average number of I/O-related error messages.
M.AR#32	Maximum memory utilization	External	Q.AR#5	**X = Amax / Rmax**, where Amax is maximum number of memory-related error messages (taken from one run of many) and Rmax is the maximum (allowed) number of memory-related

				error messages.
M.AR#33	Mean occurrence of memory error	External	Q.AR#5	**X = Amean / Rmean**, where Amean is the average number of memory error messages over a number of runs and Rmean is the maximum allowed mean number of memory-related error messages.
M.AR#34	Ratio of memory error/time	External	Q.AR#5	**X = A / T**, where A is the number of memory-related warning messages and system errors that occurred and T is the amount of time.
M.AR#35	Maximum transmission utilization	External	Q.AR#5	**X = Amax / Rmax**, where Amax is the maximum number of transmission-related error messages (taken from one run of many) and Rmax is the maximum (allowed) number of transmission-related error messages.
M.AR#36	Mean occurrence of transmission error	External	Q.AR#5	**X = Amean / Rmean**, where Amean is the average number of transmission-related error messages and failures over multiple runs and Rmean is the maximum allowed number as defined earlier.
M.AR#37	Mean of transmission error per time	External	Q.AR#5	**X = A / T**, where A is the number of warning messages or system failures and T is the operating time being measured.
M.AR#38	Transmission capacity utilization	External	Q.AR#5	**X = A / B**, where A is the transmission capacity and B is the specified transmission capacity

				designed for the software to use.
M.AR#39	Audit trail capability	External	Q.AR#6	**X = A / B**, where A is the number of data items that are actually logged during the operation and B is the number of data items that should be recorded to sufficiently monitor status of the software during operation.
M.AR#40	Diagnostic function support	External	Q.AR#6	**X = A / B**, where A is the number of failures that can be successfully analyzed using the diagnostic function and B is the total number of registered failures.
M.AR#41	Failure analysis capability	External	Q.AR#6	**X = 1 - A / B**, where A is the number of failures that are still not found and B is the total number of registered failures.
M.AR#42	Failure analysis efficiency	External	Q.AR#6	**X = Sum(T) / N**, where T is the amount of time for each failure resolution and N is the number of problems resolved.
M.AR#43	Status monitoring capability	External	Q.AR#6	**X = 1 - A / B**, where A is the number of cases for which the user or maintainer failed to get monitor data and B is the number of cases for which they attempted to get monitored data during operation.
M.AR#44	Change cycle efficiency	External	Q.AR#7	**Tav = Sum(Tu) / N**, where Tav is the average amount of time, Tu is the elapsed time for the user between sending the problem report and receiving a revised

				version, and N is the number of revised versions sent.
M.AR#45	Change implementation elapse time	External	Q.AR#7	**Tav = Sum(Tm) / N**, where Tav is the average time, Tm is the elapsed time between when a failure is detected and when the failure cause is found, and N is the number of registered and removed failures.
M.AR#46	Modification complexity	External	Q.AR#7	**T = Sum(A / B) / N**, where T is the average time to fix a failure, A is work time spent to change a specific failure, B is the size of the change, and N is the number of changes.
M.AR#47	Parameterized modifiability	External	Q.AR#7	**X = 1 - A / B**, where A is the number of cases for which the maintainer fails to resolve the failure and B is the number of cases for which the maintainer tried to resolve by changing the parameter.
M.AR#48	Software change control capability	External	Q.AR#7	**X = A / B**, where A is the number of items actually written to the change log and B is the number of change log items planned such that we can trace the software changes.
M.AR#49	Change success ratio	External	Q.AR#8	**X = Na / Ta, Y = {(Na / Ta) / (Nb / Tb)}**, Na is number of cases in which the user encounters failures after the software is changed, Nb is the number of times the user encounters failures before the software is changed, Ta is the operation time (a

				specified observation time) after the software is changed, and Tb is the time (a specified observation time) before the software is changed.
M.AR#50	Modification impact localization	External	Q.AR#8	**X = A / N**, where A is the number of failures emerging after modification of the system (during a specified period) and N is the number of resolved failures.
M.AR#51	Availability of built-in test function	External	Q.AR#9	**X = A / N**, where A is the number of cases in which the maintainer can use built-in test functionality and B is the number of test opportunities
M.AR#52	Re-test efficiency	External	Q.AR#9	**X = Sum(T) / N**, where T is the time spent to make sure the system is ready for release after a failure is resolved and N is the number of resolved failures.
M.AR#53	Adaptability of data structures	External	Q.AR#10	**X = A / B**, where A is the number of data that are not usable in the new environment because of adaptation limitations and B is the number of data that were expected to be operable in the new environment.
M.AR#54	Hardware environmental adaptability	External	Q.AR#10	**X = 1 - A / B**, where A is the number of tasks that were not completed or did not work to adequate levels during operational testing with the new environment hardware and B is the total number of functions that were tested.

M.AR#55	System software environmental adaptability	External	Q.AR#10	**X = 1 - A / B**, where A is the number of tasks that were not completed or did not work to adequate levels during operational testing with operating system software or concurrently running application software and B is the total number of functions that were tested.
M.AR#56	Ease of installation	External	Q.AR#10	**X = A / B**, where A is the number of cases in which a user succeeded in changing the install operation for their own convenience and B is the total number of cases in which a user tried to change the install procedure.
M.AR#57	Ease of setup retry	External	Q.AR#10	**X = 1 - A / B**, where A is the number of cases where the user fails in retrying the setup and B is the total number of times attempted.
M.AR#58	Available coexistence	External	Q.AR#11	**X = A / T**, where A is the number of constraints or failures that occur when operating concurrently with other software and T is the time duration of operation.
M.AR#59	Continued use of data	External	Q.AR#12	**X = A / B**, is used, where A is the number of data items that are able to be used continually after software replacement and B is the number of data items that were expected to be used continuously.

M.AR#60	Function inclusiveness	External	Q.AR#12	**X = A / B**, where A is the number of functions that produce similar results in the new software where changes have not been required and B is the number of similar functions provided in the new software as compared to the old.
M.AR#61	User support functional consistency	External	Q.AR#12	**X = 1 - A / B**, where A is the number of functions found by the user to be unacceptably inconsistent to that user's expectation and B is the number of new functions.

Table 6. GQM Metrics for SW Architecture.

These metrics focus on general processes and architectural aspects rather than evaluating specific components of the architecture implementation. Following the approach described at the beginning of the chapter, our main objective is first to check the main functionalities requirements and then evaluating performance in future GQM cycle iterations.

3.2.4 Data Collection

The data collection step focuses on retrieving metric results after some tests. Similarly to End-User dimension, a wide set of tools can be employed to extract these empirical values. Regarding Software architecture dimension, some monitoring tools like Nagios, Nagios Network Analyser (focused on network traffic exchange), SOASTA Cloud Lite, Ganglia, testing with JUnit tool or even simple sequence diagram could be particularly useful. Future GQM evaluations will be reported in the specific architecture dimension evaluation deliverables, where Data Collection chapter will present empirical values for metrics displayed above.

3.2.5 Interpret Collected Data

Software architecture and deployment evaluation depends on metrics results collected in previous step. These results display how components developed have been deployed into cluster, firstly into standalone approach. Also, results can show good component integration and platform design but also can mark weak points or lack of integration in some cases. The main goal of this dimension is a fully integration among components and successful infrastructure use.

During following GQM iterations, first iteration prototype will be extended into a distributed one, using several nodes optimally, so new data collection and evaluation will be necessary to check new results. Future GQM iterations will contain more detailed metric results which can be displayed in Radar Charts in order to provide extra information about the data collected.

The SW Architecture and deployment evaluation report will present full GQM cycle evaluation, interpreting data collected for different architecture and deployments implemented during the project.

3.3 Algorithms Dimension

The Algorithms dimension represents the processes of multi-resolution, coarsening, coordinates, connectivity and results compaction. This section aims to describe how the full GQM cycle can be assessed and reported for this dimension. To do so, the first step is checking and redefining the Goals, Questions and Metrics defined in deliverable [2] . Once the refinement process is finished, a GQM table is presented along with data collection tasks achieved and how this collected data can be interpreted.

3.3.1 Identification of GQM Goals

An initial set of GQM Goals were defined for this dimension in deliverable [2] For this deliverable, the goals have been reviewed and modified in order to capture in the key goals in the algorithms dimension of the VELaSSCo platform. These goals are listed in Table 7:

Goal	Description	WP linked to
G.AL#1	Definition of algorithms to generate multi-resolution models and compressed geometry and results information	WP3
G.AL#2	Definition of common and specific algorithms to extract the desired results for DEM/FEM simulations	WP3, WP2
G.AL#3	Definition of algorithms to format the generated results for the visualization platform	WP3, WP4
G.AL#4	Definition of common and specific analytics algorithms to compute new results from DEM/FEM simulations	WP3
G.AL#5	Definition of algorithms to generate multi-resolution models and compressed geometry and results information	WP3

Table 7. GQM Goals for Algorithms.

3.3.2 Development of GQM Plan

From the definition of the Goals identified in the previous subsection, first GQM cycle iteration has been conducted in order to define the necessary questions and metrics. The list of identified questions is shown in Table 8:

Question	Description	Associated Goal
Q.AL#1	What is the robustness of the algorithm?	G.AL#1, G.AL#2, G.AL#3, G.AL#4, G.AL#5
Q.AL#2	What is the effectiveness of the algorithm?	G.AL#1, G.AL#2, G.AL#3, G.AL#4, G.AL#5
Q.AL#3	What is the efficiency of the algorithm?	G.AL#1, G.AL#2, G.AL#3, G.AL#4, G.AL#5
Q.AL#4	What is the scalability of the algorithm?	G.AL#1, G.AL#2, G.AL#3, G.AL#4, G.AL#5

Table 8. GQM Questions for Algorithms.

3.3.3 Measurement Plan

In order to evaluate the algorithms implemented in the VELaSSCo platform we consider four essential characteristics:

- Correctness
- Robustness
- Efficiency
- Scalability

The first property (**Correctness**) deals with the accuracy of the obtained results. In particular, in this phase we verify the correctness of an algorithm, i.e., whether it accomplishes its purpose. In this context methods of validity for algorithms will be used as a proof of expected output. The algorithm is correct if for any valid input it produces the result required by the algorithm's specification.

The second important characteristic that algorithms should have is **robustness**. It can be considered as the ability of a system to cope with errors during the execution. A key aspect of this evaluation will be the ability, for a given algorithm, to continue running despite abnormalities in input (invalid or unexpected input parameters or simulations data values) or calculations.

The **efficiency** [11] aspect will consider both the computational time and the amount of memory required for obtaining a result in the execution of an algorithm. Efficiency can be considered as the measure of the processing power that is being used and it can be calculated based on the speedup per single processor.

The speedup S_n is a metric to measure performance when executing a task. It is defined as the execution time T_1 of a sequential algorithm divided by the execution time T_n of the parallel version with the adoption of n processors, i.e., $S_n = \frac{T_1}{T_n}$. The ideal result, in this context, is the linear speedup that occurs when $S_n = n$.

Finally, the efficiency E_n of the algorithm defined as $E_n = \frac{S_n}{n}$.

The last characteristic taken into account to evaluate the VELaSSCo algorithms is the **scalability**.

It can be included in the efficiency dimension and indicates how efficient the algorithm is when the numbers of nodes (or processing elements) increases [12] . Specifically, the strong scaling efficiency and weak scaling efficiency are two descriptors that can be used to quantify the efficiency of the algorithm in terms of percentage.

In the case of the strong scaling efficiency, the efficiency of the algorithm is evaluated as a function of the number of processing units. It can be computed as $t_1/(t_N*N)*100$ where t_1 and t_N are the computing times by using one and N processors respectively.

The weak scaling efficiency is defined as $(t_1/t_N)*100$ where t_1 is the time required to complete a work unit with one processing element and t_N is the amount of time to complete N of the same work units with N processing elements.

The concrete metrics for GQM are reported in Table 9:

Metrics	Description	Associated Question	Value
M.AL#1	Percentage of success of the algorithm to continue operating	Q.AL#1	The values have to be in the range [0% - 100%]
M.AL#2	Percentage of effectiveness of the algorithm	Q.AL#2	The values have to be in the range [0% - 100%]
M.AL#3	Speed-up defined as the relative performance improvement when executing a task.	Q.AL#3	From sub-linear up to linear so the efficiency value should be contained in the range [0.1, 1]
M.AL#4	Amount of memory required due to need to replicate data	Q.AL#3	The ratios have to be in the range [1/n, n] where n is the number of processors.

M.AL#5	Strong Scaling Efficiency (fixed problem size): amount of time a work unit with 1 processing, and the amount of time to complete the same unit of work with N processing elements.	Q.AL#4	The values have to be in the range [10% - 100%].
M.AL#6	Weak Scaling Efficiency (problem size grows with additional resources): amount of time a work unit with 1 processing, and the amount of time to complete the same unit of work with N processing elements.	Q.AL#4	The values have to be in the range [10% - 100%].

Table 9. GQM Metrics for Algorithms.

3.3.4 Data Collection

In this section, we present the guidelines for conducting the evaluation of the algorithms.

The **correctness** of the algorithm will be evaluated by adopting the unit testing method [13] . One or more functions (sub-routines), belonging to the same algorithm, are identified by the algorithm developers and will be tested to determine whether they produce the expected result. The basis for this component testing is algorithms created by programmers during and after the development process to ensure that a specific part of code behaves as intended. More specifically, the Apache MRUnit tool[6] can be used to conduct this test. Apache MRUnit is a Java library that helps developers to unit test Apache Hadoop MapReduce jobs. MRUnit lets developer define key-value pairs to be given to map and reduce functions, and it tests that the correct key-value pairs are emitted from each of these functions. MRUnit tests are similar to traditional unit tests in that they are simple, isolated, and don’t require Hadoop daemons to be running.

For each algorithm to be tested, a set of inputs and expected outputs has to be provided. The final result emitted by the Job program is compared with the expected result and when they match the single test is considered as “passed”. On the other hand, if the expected and emitted results are different the test is marked as “failed”. For each algorithm, a success rate of 100% has to be achieved.

[6] https://mrunit.apache.org/

In order to evaluate the **robustness** of the algorithms, a similar approach to the one proposed for evaluating the effectiveness can be followed. A fault injection strategy can be applied to test the robustness, comparing the result of a set of invalid/unexpected input parameters or simulation data values with the related expected behaviour of the algorithm. For this test, a success rate 100% of has to be achieved.

Although these types of faults can be injected by hand or by implementing short fragments of code, the possibility of introducing an unintended fault is very high. For this reason, we plan to use a fault injection tools like the Hadoop Fault Injection Framework[7] to parse a program automatically and insert faults.

For evaluating the **efficiency** and the **scalability** of the algorithms, a set of benchmark tests have to be provided in order to assess the performance by running a number of different trials.

The parameters taken into account for reproducing different benchmark test sets are the **number of processors** used for the computation (from 1 up to the number provided by the platform) and the input (simulation data) by considering different **dimensions** (number of time steps and size of meshes).

In the first phase, as a simple monitoring tool for the jobs execution, the Hadoop JobTracker web interface visualization tool will be adopted. The JobTracker web interface provides a wealth of information on jobs and tasks that are running on the cluster as well as historical information on completed jobs (including ones that failed). The *Analyse job history* link on the *Job details history* page displays a summary of task performance statistics and details of individual task attempts can be extracted.

For more meaningfully statistical result, it is possible to explore different solutions like the Hadoop Vaidya tool[8]. Hadoop Vaidya is a rule based performance diagnostic tool for MapReduce jobs that performs a post execution analysis of a map/reduce job by parsing and collecting execution statistics through the job history and job configuration files. It runs a set of predefined tests/rules against job execution statistics to diagnose various performance problems. Each test rule detects a specific performance problem with the MapReduce job and provides a targeted advice to the user. Hadoop Vaidya generates an XML report based on the evaluation results of individual test rules.

For more advanced reporting, in future, we can also explore different solutions to measure the algorithm performance. In particular, Hadoop includes built-in connections to Ganglia cluster monitoring, which is a tool to measure hardware statistics and the Hadoop job history log for analysing job performance.

3.3.5 Interpret Collected Data

Depending on the specific test, several graphs can be used to better understand the results and to provide extra information about data collected.

[7] https://hadoop.apache.org/docs/r2.4.1/hadoop-project-dist/hadoop-hdfs/FaultInjectFramework.html
[8] http://hadoop.apache.org/docs/r1.2.1/vaidya.html

In case of **robustness test**, it is possible to categorize results depending on the test case sets adopted, and produce chart graphs to emphasize where the algorithm copes well and where it is sensitive to problems. For example, abnormalities in the input can be further divided into different categories like: missing values, wrong values, wrong types (character instead of integer).

Regarding the **correctness experiments**, comparing expected and obtained results, a series of different graphs (pie chart or bar chart) can be produced to evaluate the relative error for each single expected local result, such as result expected for a given static mesh node.

Results provided by the **efficiency and scalability** tests can be represented by plotting a function where the x axis shows the number of processors used to run the algorithm and the y axis shows the achieved speedup. From the results obtained by adopting the monitoring tools, we will be also able to plot more functions such as job phases and job histograms. The job phase plots can give us important information about the duration of each map, reduce, shuffle, and sort phase. The job histograms can uncover load-balancing issues such as the well-known “hot-spotting” problem where the workload for one node is more than others, i.e., when HBase rows are not well distributed over the cluster.

3.4 Navigation and Interaction Dimension

The End-User Functionalities dimension characterizes the ease of use of VELaSSCo SW System. This section aims to describe how the full GQM cycle can be assessed and reported for this dimension. To do so, the first step is checking and redefining the Goals, Questions and Metrics defined in deliverable [2] . Once the refinement process is finished, a GQM table is presented along with data collection tasks achieved and how this collected data can be interpreted.

Currently this dimension aims to define the evaluation plan in terms of validation with real users (the VELaSSCo User Panel) using a more qualitative approach than the rest of the previous dimensions by the means of a specific questionnaire. Therefore, this deliverable will focus Navigation and Interaction dimension in terms of defining a proper list of Goals, Questions and Metrics identified at this stage of the project.

3.4.1 Identification of GQM Goals

The list of goals obtained in step 1 has been refined compared to [2] and are listed in Table 10:

Goal	Description	WP linked to
G.NI#1	VELaSCCo Platform Usable	WP2, WP3, WP4

Table 10. GQM Goals for Navigation and Interaction dimension

3.4.2 Development of GQM Plan

As in deliverable [2] , the set of 23 questions about the model associated to the two high level goals defined in step 1 is represented by the “VELaSSCo Usability

Questionnaire" attached in Section 7.2. In addition to the questionnaire, Table 11 shows some specific questions, in order to adapt standard ISO-9126 to Navigation and Interaction dimension:

Question	Description	Associated Goal
Q.NI#1	Has the VELaSCCo Platform (GUI, API …) any kind of end user documentation, manuals or support info associated?	G.EU#1
Q.NI#2	What is the quality of the manuals?	G.EU#1
Q.NI#3	What is the quality of the demo/prototype?	G.EU#1
Q.NI#4	What is the quality of the help system?	G.EU#1
Q.NI#5	What is the quality of the complexity of GUI and API design?	G.EU#1

Table 11. GQM Questions for Navigation and Interaction dimension

3.4.3 Measurement Plan

As in deliverable [2] the list of quantitative metrics for each of the questions included in the Questionnaire is represented by the scale defined in "VELaSSCo Usability Questionnaire" attached in Section 7.2 and based on [10] Values range goes from "Strongly Agree" (1) to "Strongly Disagree" (7).

The values taken from the questionnaire will be complemented with other quality perceptions gathered by the interaction between the facilitators of the tests done with the User Panel.

Similarly to previous chapter, Measurement plan includes a new table of metrics not presented in deliverable [2] , which could complement information collected with Questionnaires attached in Annex 0.

Table 12 display the metrics defined for Navigation and Interaction dimension:

Metrics	Description	Associated Question	Value
M.NI#1	Manuals Coverage	Q.NI#1, Q.NI#2	MCov= %FC, where FC = Proportion of functional elements described in Manuals.
M.NI#2	Manuals Consistency	Q.NI#1, Q.NI#2	MCon = (%FEI + %DCVMV)/2, where FEI= Proportion of

			functional elements incorrectly described in Manuals and DCVMV = Difference between component version and the manual version.
M.NI#3	Manuals Legibility	Q.NI#1, Q.NI#2	Ratio of Figures per manual pages, Ratio of Tables per manual pages and Ratio of Diagrams per manual pages.
M.NI#4	Manuals Suitability	Q.NI#1, Q.NI#2	Average pages per functional elements.
M.NI#5	Effectiveness Ratio	Q.NI#1, Q.NI#2	Proportion of functional elements correctly used after reading the manual.
M.NI#6	Understandability Ratio	Q.NI#1, Q.NI#2	Proportion of functional elements correctly understood after reading the manual.
M.NI#7	Prototype/Demonstration Coverage	Q.NI#3	Proportion of functional elements showed in demos/prototype.
M.NI#8	Prototype/Demonstration Consistency	Q.NI#3	Difference between demo/prototype version and component version.
M.NI#9	Help System Coverage	Q.NI#4	Proportion of functional elements showed in help system.
M.NI#10	Help System Consistency	Q.NI#4	Proportion of functional elements incorrectly described in help system.
M.NI#11	Help System Suitability	Q.NI#4	Help system word Ratio.
M.NI#12	Help System Effectiveness Ratio	Q.NI#4	Proportion of functional elements correctly used after using the help system.
M.NI#13	Help System Understandability Ratio	Q.NI#4	Proportion of functional elements correctly understood after using the help system.
M.NI#14	Readability	Q.NI#5	Proportion of functional elements with meaningful names.

M.NI#15	User Interface Understandability	Q.NI#5	Proportion of functional elements used without errors.
M.NI#16	I/O Understandability	Q.NI#5	MIOUnd = (%ECU + %RVC + %ACU) /3, where ECU =Proportion of exceptions correctly understood, RVC= Proportion of return values correctly understood, and ACU= Proportion of arguments correctly understood.
M.NI#17	Ease of Component Learning	Q.NI#5	Time to use = average time to use correctly the component, Time to expertise = average time to master the component functionality
M.NI#18	Customisability	Q.NI#5	Configurable parameters per interface Ratio and Configurable parameters per operations Ratio.
M.NI#19	Error messages Suitability	Q.NI#5	Error message per functional element density.
M.NI#20	Error messages Clearness	Q.NI#5	Proportion of error messages correctly understood.
M.NI#21	User Interfaces complexity	Q.NI#5	Operations per interface density, and Events per interface density.

Table 12. GQM Metrics for Navigation and Interaction dimension

3.4.4 Data Collection

The data collection will be done by means of the "VELaSSCo Usability Questionnaire" attached in Section 7.2 to be provided to specific testers (members of the VELaSSCo User panel and others). Also, results for metrics defined in previous chapter will be collected by means of different monitor tools, user interface evaluation software, etc. The main goal is having a complete overview of Navigation evaluation, including both aspects: user feedback and metrics results.

3.4.5 Interpret Collected Data

This step is directly dependent on step 4. After completion of the Usability questionnaire by User panel it will be possible to interpret the collected data.

Future Navigation and Interaction evaluation report will include a complete GQM iteration process, providing a full assessment of the dimension, including Questionnaires responses and metrics results.

3.5 Views Dimension

The Views dimension characterizes the perspective of the VELaSSCo observers focusing on effectiveness aspects that can help them in the decision-making process. This section aims to describe how the full GQM cycle can be assessed and reported for this dimension. To do so, the first step is checking and redefining the Goals, Questions and Metrics defined in deliverable [2] . Once the refinement process is finished, a GQM table is presented along with data collection tasks achieved and how this collected data can be interpreted.

3.5.1 Identification of GQM Goals

Similarly to previous dimensions, in order to align GQM process with main Use Cases, goals have been completely redefined again. The goals obtained as result of covering step 1 are listed in Table 13:

Goal	Description	WP linked to
G.VI#1	VELaSCCo Platform real-time data access Effective	WP2, WP3, WP4
G.VI#2	VELaSCCo Platform Visualization clients Effectives	WP2, WP3, WP4

Table 13.GQM Goals for Views.

3.5.2 Development of GQM Plan

From the definition of the Goals identified in the previous subsection, first GQM cycle iteration has been conducted in order to define the necessary questions and metrics. The list of identified questions is shown in Table 14:

Question	Description	Associated Goal
Q.VI#1	How long does the data loading take during offline injection?	G.VI#1
Q.VI#2	How long does the data writing take during offline injection?	G.EU#1
Q.VI#3	What is the response time for interactive operations (zoom, rotate …)?	G.EU#2
Q.VI#4	What is the response time between the user´s request and the visualization of results?	G.EU#2
Q.VI#5	Are the interactive operations dependants partially of the visualization client?	G.EU#2
Q.VI#6	What are the most convenient time steps for interactive visualization?	G.EU#2

Q.VI#7	What is the maximum size of visualization?	G.EU#2
Q.VI#8	What is the reasonable resolution for visualization of many results?	G.EU#2

Table 14. GQM Questions for Views.

3.5.3 Measurement Plan

These metrics complements the previous ones and focuses on evaluating the visualization components of the architecture deployment and perception by the end-users. Following the GQM methodology and aligned with main Use Cases, the metrics obtained as result of covering step 3 are listed in Table 15:

Metrics	Description	Associated Question	Value
M.VI#1	Data loading time per time step	Q.VI#1	Sec.
M.VI#2	Data writing time per time step	Q.VI#2	Sec.
M.NI#3	GiD handling time	Q.VI#3, Q.VI#4, Q.VI#5	GiD handling time + VQuery_X + … +Vquery_Y
M.NI#4	IFX handling time	Q.VI#3, Q.VI#4, Q.VI#5	IFX handling time + VQuery_X + … +Vquery_Y
M.NI#5	Average time steps	Q.VI#6	Avg. time steps (depending on simulation scale)
M.NI#6	Results Maximun Size	Q.VI#7	GB
M.NI#7	Results Reduction Ratio	Q.VI#8	$1/10^n$

Table 15. GQM Metrics for Views.

3.5.4 Data Collection.

Data collection for the View dimension includes much of the evaluation methodology described in the previous sections, as this dimension shares many of the goals, questions and metrics with the other project dimensions, taking into account the well-defined four essential vectors of our interest: effectiveness, robustness, efficiency and scalability.

During Views dimension specific evaluation, a deliverable report related will be created where metrics results will be presented and interpreted.

3.5.5 Interpret Collected Data

As it has been described, in order to interpret the collected data in the precedent step 4, we will present complete GQM iteration in next evaluation reports that allow us to compare results from main test scenarios defined in Section 4. In the case of this view dimension, the data coming from the other vectors are essential for a better understanding of the results, for those questions and metrics involving architectural aspects (SW Architecture and Deployment Environment Dimension), algorithmic performance (Algorithms Dimension) and user-related issues (End-User Functionalities Dimension, Navigation and Interaction Dimension), all of them also included in this dimension.

4 Evaluation scenarios and tasks of the first iteration

This section describes the setting of the first iteration of the evaluation scenarios following the methodological approach explained in section 2.3.

4.1 Objectives and setting

The objective of the first iterations of the evaluation of the VELaSSCo framework is to ensure the usability, effectiveness and performance of the proposed solution. This evaluation should be understood as an opportunity to get important feedback in order to develop a more robust solution by the end of the next iteration. This feedback will serve to enrich the VELaSSCo architecture and its main software elements, especially for the large-scale trials expected to come by the end of the second iteration.

In order to check the feasibility of using GQM methodology for the evaluation of the VELaSSCo Framework, we have carried out minor evaluation iteration with simple scenarios. This document reports on this preliminary iteration cycle, where the methodology has been tested to assess its viability to evaluate the functionality of the system. To do so, DEM scenarios have been redefined with test cases reported above in chapter 3.

The evaluation of the first iteration will be done in the following setting:

- **Infrastructure**: CIMNE HPC Cluster with 9 nodes
- **Tools**:
  - Visualization Clients (IFX and GiD),
  - VELaSCCo Platform, Big Data Ecosystem (Flume, Hbase, Hive, Hadoop)
  - EDM,
  - Nagios and Nagios Network Analyzer.
- **Questionnaires**: Usability and Effectiveness questionnaires are provided.
- **GQM** tables as explained in section 3.
- **Requirements for the user testers**: The users will make use of their own/other GPUs with these minimal characteristics to run the client components listed above.

4.2 Tasks given to users

Not all the evaluation is based on the behavior of a human user with the system, but this section deals with precisely these aspects that have to do with the user interaction with the system, their perception and the performance of the underlying framework. In the evaluation process we will find the following roles:

- **User testing population**: mainly user panel and consortium members, but also monitoring tools that could impersonate user roles or give monitoring results, such as Nagios.

- **Facilitators**: in charge of be helping the tester users to perform the tasks defined and taking notes of the users' behaviour. Several members of the consortium will help on this.
- **Analysts**: in charge of analysing the results. ATOS staff and other members of the consortium will take this role.

For the actual evaluation, the users will be given a set of tasks based on the GQM tables defined in section 4 for each of the dimensions. It is worth noticing that some particular questions cannot be assessed in the first iteration due to the maturity of the solution, so they will be evaluated in the final iteration.

The list of task is provided in section 2.3 and consists of a series of 10 tasks for the Telescope Use case (FEM) and 11 tasks for the Fluidized Bed Use case (DEM). These tasks have been specifically designed to assess the majority of the metrics proposed in the GQM tables. Specific traces and logs have been placed to monitor the system and measure the metrics.

The Facilitators will take care of easing the tasks given to the users. In particular, facilitators have been instructed to help users and record impressions, doubts, comments, suggestions, etc. These observations will help understanding how users react to the given tasks and are an invaluable element for improving the system.

4.3 Analysis instruments

There will be several analysis instruments, namely the following:

- Quantitative instruments:
  - Log preparation: As it was already explained, logs and traces associated to SW component of the VELaSCCo platform has been placed to track the tasks and measure specific metrics.
  - Nagios and Nagios Network Analyser will be used to monitor the VELaSCCo Platform, which let us get most of quantitative metrics defined in the GQM metric tables associated to each one of the evaluation dimensions.
- Qualitative instruments
  - We will use a Usability and Effectiveness evaluation for the evaluation of the first prototype.
  - Several questionnaires have been prepared for the user testers to give their feedback about the system.

The set of observations of the facilitators and the results of the questionnaires will be analyzed using specially designed Excel tools. In particular, the analysis task will:

- Analyze the measurements and map them to metrics
- Use facilitators feedback for qualitative evaluation

- Provide conclusions and feedback of the quantitative and qualitative evaluation to improve the VELaSSCo Framework.

4.4 Evaluation plan of the initial iteration

At the time of writing this document, the consortium is preparing an evaluation day in Edinburgh for December 2015. Invitations have been sent to member of the User Panel and interested parties to be part of this evaluation event.

The results of the evaluation will be reported in subsequent deliverables of WP5 for each of the dimensions listed in section 3.

5 Conclusions and future work

5.1 Future work

This document focuses on a minor evaluation iteration which aims to cover the full GQM life-cycle in order to check its suitability over the architecture dimensions identified. The next evaluation iterations will be reported more widely, presenting each dimension in a separate evaluation report, where specific GQM plan should be achieved. Concretely, the list of evaluation reports to be delivered is:

- *D5.2, D5.3 Architecture evaluation*: This deliverable led by ATOS will be based on GQM plan described in the present document. Following this plan an evaluation of first prototype and final prototype will be achieved.
- *D5.4, D5.5 Algorithm evaluation*: This deliverable led by UEDIN will be based on GQM plan described in the present document. Following this plan an evaluation of first prototype and final prototype will be achieved.
- *D5.6, D5.7 Effectiveness evaluation of real-time data access and visualization*: This deliverable led by CIMNE will be based on GQM plan described in the present document. Following this plan an evaluation of first prototype and final prototype will be achieved.
- *D5.8, D5.9 Usability evaluation*: This deliverable led by ATOS will be based on GQM plan described in the present document. Following this plan an evaluation of first prototype and final prototype will be achieved. Both deliverables will merge two dimensions into the same document report: End-Users functionalities and Navigation and Interaction dimensions.

All deliverables mentioned above will apply the GQM plan according to the evaluation scenarios described and refined in Chapter 0. Iterative evaluation will be used to improve the deployment of the platform. Ideally, the GQM Metrics results for the early prototype will serve as a basis of comparison for final prototype metrics results, in order to show improvement in the performance aspects which first evaluation could show as insufficient.

5.2 Conclusions

In this deliverable, we described a first attempt to apply the GQM methodology evaluation to the VELaSSCo dimensions framework. Rather than carrying out an intensive system evaluation, this report seeks to verify the viability of the chosen methodology.
This has been based on the work described in deliverable [3] , taking the GQM plan for each dimension and refining them if possible, updating goals and questions or adding new metrics.

Firstly, the GQM plan, covering the full cycle, has been applied in each dimension in chapters 3.1 and 3.2. The full cycle includes data collection and interpretation in Steps

4 and 5. This metrics report aims to show how the system can be improved in future iterations, and should be taken into account for future evaluations. The infrastructure on the CIMNE Cluster has been tested, in order to evaluate how well it fits with the current stage of the development.

Secondly, we explained how the identified scenarios in chapter 4 have been used for current evaluation and updated for future iterations. In our case, a small use case has been chosen, as performance is not yet a goal of this evaluation.

Finally, an evaluation of future work is described, where the main goal is delivering an individual report for each framework dimension. These reports will be released in two versions, the first one for early prototype and second one for final prototype.

6 References

[1] VELaSSCo D1.1. End-users requirements and Users panel.

[2] VELaSSCo D1.5. Definition of criteria and methodology for system evaluation

[3] VELaSSCo D2.2. Specification of Big Data architecture

[4] VELaSSCo D2.4. Design a petabyte sized engineering data solution

[5] D. Rud, A. Schmietendorf, R. Dumke, "Product metrics for service-oriented infrastructures," in Proceedings of "16th International Workshop on Software Measurement/DASMA Metrik Kongress 2006" (IWSM/MetriKon 2006), pp. 161-174, November 2-3, 2006, Potsdam, Germany

[6] V. Basili, G. Caldiera, and H.D. Rombach, "The Goal Question Metric Paradigm," Encyclopedia of Software Eng., vol. 2, pp. 528-532, John Wiley & Sons, 1994.

[7] S. Bassil and R. Keller, "A Qualitative and Quantitative Evaluation of Software Visualization Tools," Proc. 23rd IEEE Int'l Conf. Software Eng. Workshop Software Visualization, pp. 33-37, 2001.

[8] K. Gallagher, A. Hatch, M. Munro, "Software Architecture Visualization: An Evaluation Framework and Its Application". IEEE Transactions on Software Engineering, vol. 34, no. 2, March/Aapril 2008.

[9] Markus Nick, Klaus-Dieter Althoff, Carsten Tautz, "Facilitating the Practical Evaluation of Organizational Memories Using the Goal-Question-Metric Technique". KAW'99 – Twelfth Workshop on Knowledge Acquisition, Modeling and Management Track "Evaluation of KE Techniques"

[10] Lewis, J. R. (1995) IBM Computer Usability Satisfaction Questionnaires: Psychometric Evaluation and Instructions for Use. International Journal of Human-Computer Interaction, 7:1, 57-7.

[11] Vipin Kumar (2002). Introduction to Parallel Computing, Addison-Wesley Longman Publishing Co.

[12] Bondi, André B. (2000). Characteristics of scalability and their impact on performance. Proceedings of the second international workshop on Software and performance, WOSP '00, 195-203

[13] Beck, K. Test-Driven Development by Example, Addison Wesley - Vaseem, 2003

7 Annex

7.1 VELaSSCo System Usability Questionnaire

Participant: ________________________

System: ____________________________

This questionnaire gives you an opportunity to tell us your reactions to the system you used. Your responses will help us understand what aspects of the system you are particularly concerned about and the aspects that satisfy you.

To as great a degree as possible, think about all the tasks that you have done with the system while you answer these questions.

Please read each statement and indicate how strongly you agree or disagree with the statement by circling a number on the scale. If a statement does not apply to you, circle N/A.

Please write comments to elaborate on your answers.

As you complete the questionnaire, please do not hesitate to ask any questions.

Thank you!

7.2 Usability Questionnaire Questions

1. Overall, I am satisfied with how easy it is to use this system.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

2. It was simple to use this system.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

3. I could effectively complete the tasks and scenarios using this system.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

4. I was able to complete the tasks and scenarios quickly using this system.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

5. I was able to efficiently complete the tasks and scenarios using this system.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

6. I felt comfortable using this system.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

7. It was easy to learn to use this system.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

8. I believe I could become productive quickly using this system.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

9. The system gave error messages that clearly told me how to fix problems.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

10. Whenever I made a mistake using the system, I could recover easily and quickly.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

11. The information (such as on-line help, on-screen messages and other documentation) provided with this system was clear.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

12. It was easy to find the information I needed.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

13. The information provided for the system was easy to understand.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

14. The information was effective in helping me complete the tasks and scenarios.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

15. The organization of information on the system screens was clear.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

Note: *The "interface" includes those items that you use to interact with the system. For example, some components of the interface are the keyboard, the mouse, the microphone, and the screens (including their use of graphics and language).*

16. The interface of this system was pleasant.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

17. I liked using the interface of this system.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

18. This system has all the functions and capabilities I expect it to have.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

19. Overall, I am satisfied with this system.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

20. I would buy and use this system software.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

21. I would recommend this system software to others.

STRONGLY AGREE 1 2 3 4 5 6 7 STRONGLY DISAGREE

COMMENTS:

22. Please list the three things you liked most about this system software.

23. Please list the three things you liked least about this system