WP2.3 SRS
GRIDBENCH
(THE CROSSGRID BENCHMARK SUITE)

Document Filename:	**CG-2.3-D2.1-UCY001-1.0-SRS**
Work package:	**Metrics & Benchmarks (WP2.3)**
Partner(s):	**UCY, TUM**
Lead Partner:	**UCY**
Config ID:	**CG-2.3-D2.1-UCY001-1.0**
Document classification:	**PUBLIC**

Abstract: This document describes the basic requirements for the CrossGrid Benchmark Suite, which will be used to investigate experimentally the performance of Grid sites and constellations. The CrossGrid Benchmark Suite will provide a variety of tools, such as *Micro-benchmarks, Micro-kernel benchmarks* and *Application-kernel benchmarks*. These tools can be used to measure the performance of the different elements that make up the Grid. Benchmarking these elements (such as networking and computational power) will provide essential information that is useful for administration of the Grid, real application performance prediction, cost estimation and numerous other uses.

Delivery Slip

	Name	Partner	Date	Signature
From	WP2, task 3	UCY, TUM	30.04.2002	
Verified by				
Approved by				

Document Log

Version	Date	Summary of changes	Author
Draft	30/4/2002	Draft version	M. Dikaiakos G. Tsouloupas
0.9	15/5/2002	Pre-Final	M. Dikaiakos G. Tsouloupas
1.0	1/6/2002	Final	M. Dikaiakos G. Tsouloupas

CONTENTS

1. INTRODUCTION 4

1.1. PURPOSE 4
1.2. SCOPE 4
1.3. DEFINITIONS, ACRONYMS, AND ABBREVIATIONS 5
1.4. REFERENCES 5
1.5. OVERVIEW 6

2. OVERALL DESCRIPTION 7

2.1. PRODUCT PERSPECTIVE 7
2.1.1. System interfaces 7
2.1.2. User interfaces 7
2.1.3. Hardware interfaces 8
2.1.4. Software interfaces 8
2.1.5. Communications interfaces 8
2.1.6. Memory constraints 8
2.1.7. Operations 8
2.1.8. Site adaptation requirements 8
2.2. PRODUCT FUNCTIONS 9
2.2.1. Benchmark Categories 9
2.2.2. Benchmark Targets 10
2.2.3. General use case 10
2.2.4. A Usage Scenario 10
2.3. USER CHARACTERISTICS 11
2.4. CONSTRAINTS 11
2.5. ASSUMPTIONS AND DEPENDENCIES 11
2.5.1. Dependence on other Tasks: 12
2.6. APPORTIONING OF REQUIREMENTS 12

3. SPECIFIC REQUIREMENTS 13

3.1. EXTERNAL INTERFACES 13
3.2. FUNCTIONS 13
3.2.1. Micro-benchmarks 13
3.2.2. Micro-kernels 13
3.2.3. Application kernels 13
3.3. PERFORMANCE REQUIREMENTS 14
3.4. LOGICAL DATABASE REQUIREMENTS 14
3.5. DESIGN CONSTRAINTS 14
3.6. STANDARDS COMPLIANCE 14
3.7. SOFTWARE SYSTEM ATTRIBUTES 14

4. INDEX 15

1. INTRODUCTION

1.1. PURPOSE

This document seeks to provide the Software Requirements Specifications for the CrossGrid benchmark suite, to be developed under task WP2.3 of the CrossGrid project. Software Requirements Specifications will serve as a guideline for future WP2.3 work. Furthermore, this document presents an initial description of the various functionalities and services that WP2.3 software expects to invoke and/or incorporate from other CrossGrid tasks (e.g., WP2.4, WP3).

The intended audience are members of the CrossGrid consortium and the Grid community at large, interested in Grid performance evaluation.

WP2.3 seeks to:

- Propose models and performance metrics for the Grid, to capture and describe concisely:
  - The performance of *Grid constellations,* i.e. of collections of computing resources (clusters, parallel computers) connected through wide-area networks, belonging to a Virtual Organizations and used concurrently for the solution of a particular problem.
  - The performance of Grid applications deployed upon a Grid constellation.
- Propose a suite of *representative* benchmarks (GridBench - the CrossGrid Benchmark Suite) capturing such metrics.
- Implement and run the benchmark suite on the CrossGrid test-bed.
- Provide members of the CrossGrid consortium with access to the benchmark suite and to historical data collected from benchmark invocations, through the CrossGrid portal interface.
- Assess the usefulness of benchmarks in:
  - Identifying factors that affect Grid application performance.
  - Providing a mechanism for an early, low-cost performance evaluation of Grid constellations, and the cost prediction of Grid applications.
- Providing parameter estimates to the analytical performance prediction models, to be developed in CrossGrid.

1.2. SCOPE

The product of this work will be a set of metrics definitions and a set of benchmarks to produce these metrics. Metrics will seek to capture the performance characteristics of micro-benchmarks, micro-kernels, and application benchmarks, in three different domains of the Grid: (i) At the site-level, that is a set of computing resources connected via a local network, under a common administrative domain (e.g. a cluster); (ii) At the Grid-constellation level, which involves multiple “sites” connected within a Virtual Organization and used for the solution of a common problem (e.g. a large FFT computation, a flood simulation, etc.), and (iii) at the Grid middleware level, i.e. services developed by CrossGrid, DataGrid or Globus.

A suite of benchmarking software, GridBench, will be developed to produce estimates of these metrics for each of the three domains.

1.3. DEFINITIONS, ACRONYMS, AND ABBREVIATIONS

CE	Compute Element
APART	Automatic Performance Analysis: Real Tools
API	Application Program Interface
CG	CrossGrid
CrossGrid	The EU CrossGrid Project IST-2001-32243
DataGrid	The EU DataGrid Project IST-2000-25182
G_PM	Grid-enabled Performance Measurement tool
Grid Constellation	A collection of "sites" (see "*Site*"), connected within a Virtual Organization and used for the solution of a common problem
GridBench	The CrossGrid benchmark suite
GridFTP	A Grid-enabled version of FTP (Part of Globus)
GRIS	Grid Resource Information Service
HLAC	High-level analysis component
HPC	High Performance Computing
HTC	High Throughput Computing
MPI	Message Passing Interface
MPICH-G2	A free, Globus-enabled MPI implementation
NAS	NASA Advanced Supercomputing
PVFS	Parallel Virtual File System
SE	Storage Element
Site	A set of Computing Elements, Storage Elements or other resources in a single geographical location.
SPLASH	Stanford Parallel Applications for Shared Memory
VO	Virtual Organization

1.4. REFERENCES

M. Dikaiakos, G. Samaras, "Performance Evaluation of Mobile Agents: Issues and Approaches." In *Performance Engineering. State of the Art and Current Trends,* Dumke, Rautenstrauch, Schmietendorf and Scholz (eds). Lecture Notes of Computer Science series, State of the art survey, vol. 2047, pages 148-166, Springer, May 2001.

I. Foster and C. Kesselman, editors, *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, 1998.

M. Dikaiakos, A. Rogers and K. Steiglitz, "Performance Modeling through Functional Algorithm Simulation," in *Advanced Computer System Design*, Chapter 3, G. Zobrist, K. Bagchi and K. Trivedi (editors), pages 43-62, Gordon & Breach Science Publishers, 1998, ISBN 90-5699-634-7.

Bailey, D. H., E. Barszcz, J. Barton, D. Browning, R. Carter, L. Dagum, R. Fatoohi, S. Fineberg, P. Frederickson, T. Lasinski, R. Schreiber, H. Simon, V. Venkatakrishnan, and S. Weeratunga, "The

NAS Parallel Benchmarks," Technical Report RNR-94-007, NASA Ames Research Center, (March 1994).

Steven Woo, Moriyoshi Ohara, Evan Torrie, Jaswinder Pal Singh and Anoop Gupta. “The SPLASH2 Programs: Characterization and Methodological Considerations.” In *Proceedings of the Twenty-first International Symposium on Computer Architecture*, June 1995.

M. Dikaiakos, "Analyzing the Workload of Scientific Visualization Tools: A Preliminary Study with TIPSY." In *Proceedings of the 1997 International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems - MASCOTS '97*, IEEE-Computer Society.

1.5. OVERVIEW

This task will propose a set of performance metrics to describe the performance capacity of Grid constellations and Grid application performance, and it will develop and implement benchmarks that are representative of Grid workloads.

Such benchmarks can be used to estimate the values of performance metrics for different Grid constellations, to identify important factors that affect end-to-end application performance, and to provide application developers with the initial estimates of expected application performance and cost-performance ratios.

GridBench will be deployed and validated in the CrossGrid Testbed.

2. OVERALL DESCRIPTION

2.1. PRODUCT PERSPECTIVE

Figure 1 gives an overview of task 2.3 interfaces. The benchmarks will be invoked by the portal and the result conveyed back to the portal. In addition to tools developed in Task 2.3, some of the tools from performance evaluation will be used to gather performance data. The metrics will also be published either through GRIS or by a GridBench Service.

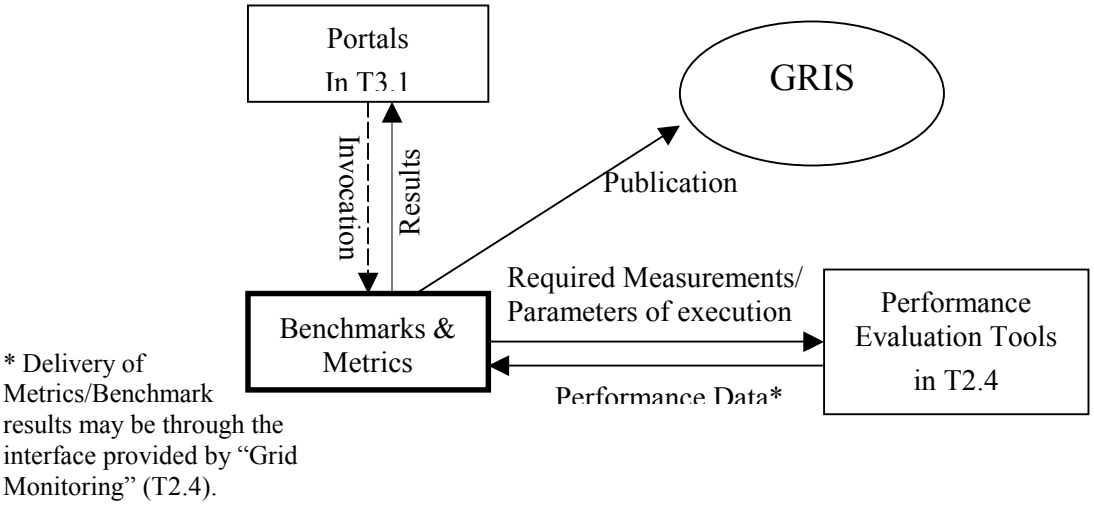

Figure 1:Diagramatic view of general functionality and interaction with other components

2.1.1. System interfaces

WP 2.4: Task 2.4 plans to develop the tools for collecting and delivering performance metrics of WP1 applications. In a similar manner, benchmarking will use WP2.4 software to report and collect performance properties of benchmarks invoked and executed in the CrossGrid test-bed. In particular, we will use the HLAC (High-Level Analysis Component) part of T2.4 and its notion of *probes* to extract performance data during benchmarking.

The performance properties derived from benchmarking can be used by WP2.4 for specifying the values of parameters employed in the performance prediction model.

WP3: Benchmarks will be invoked via the portal created by Task 3.1.

The tools of WP3.3 will monitor runtime behaviour of benchmarks. Furthermore, selected subsets of benchmark data will be archived for further analysis, through the tools provided by WP3.3. Alternative options for publishing benchmark data will be considered, such as the use of facilities supplied by Globus (through GIS).

2.1.2. User interfaces

Access to the benchmarks will be provided through the Web-based User Interface developed by Task 3.1 for CrossGrid applications.

A user of GridBench will be able to define the parameters specifying a particular benchmark invocation through the User Interface: for instance, problem size, input and output files, the configuration of the Grid constellation to be used, etc.

2.1.3. Hardware interfaces

N/A

2.1.4. Software interfaces

Middleware (Globus,DataGrid,MPICH-G2, etc.)

GridBench seeks to provide a set of well-understood codes with high-performance, high-throughput requirements, which are representative of applications deployed upon Grid constellations. To this end, most of the CG benchmarks will follow the application programming model adopted by CrossGrid:

- Coding in C/C++;
- Employment of MPICH-G2 for parallelization and inter-process communication;
- Invocation of Grid services through the APIs of CrossGrid middleware, which will also include Globus and DataGrid middleware services.

Additionally, a subset of benchmarks will seek to isolate performance properties of the CrossGrid middleware, hence interfacing directly with Grid services of WP3.

2.1.5. Communications interfaces

Benchmarks will utilize MPICH-G2 for communication.

2.1.6. Memory constraints

Memory constraints of benchmarks will be configurable by benchmark parameters determining the problem size of a particular benchmark invocation. The goal is to come up with benchmark memory requirements representative of typical WP1-application runs.

2.1.7. Operations

CrossGrid administrators and users will be able to launch a benchmark on a target configuration, a Grid constellation or a particular Site, through the Portal developed in Task 3.1.

During the project, we will investigate the development of a *Grid Benchmarking Service*, so that another service (such as scheduler or the Resource Manager) may find it useful to run the benchmarks on demand or at regular intervals.

2.1.8. Site adaptation requirements

N/A

2.2. PRODUCT FUNCTIONS

GridBench seeks to investigate performance properties of three "layers" of the Grid architecture: (i) the *Site*, which is a collection of resources interconnected through a local- or system-area network, and belonging to one administrative domain (e.g. a cluster of PCs, a symmetric multiprocessor system); (ii) the *Grid Constellation*, which includes multiple sites constituting the computing platform of a Virtual Organization, and used for the solution of a problem; (iii) the *Middleware*, that is the software layer providing access to shared resources of a Grid constellation, through an API available at the application programming level.

GridBench will be composed of three families of benchmarks: *Micro-benchmarks, Micro-kernels* and *application-kernels*.

		Benchmark Targets		
		Site*	Grid Constellation	Middleware
Benchmark Categories	Micro-Benchmarks	(1)		(2)
	Micro-Kernels	(3)	(4)	
	Application Kernels	(5)	(6)	

Table 1: the GridBench matrix

* Benchmarks that target specific Sites will also indirectly measure performance of the Computing and Storage Elements at those sites.

Micro-benchmarks will target Sites (1) and middleware (2) (e.g., GridFTP transfers). Micro-kernels will target Sites (3) and Grid constellations (4). Application kernels will target Sites (5) and Grid constellations (6). Even though (4) and (6) do not target the middleware explicitly, the middleware will affect their performance.

2.2.1. Benchmark Categories

Micro-benchmarks

- For identifying basic performance properties of Grid services, sites, and constellations.
- Each micro-benchmark tests a single performance aspect, through "stress testing" of a simple operation invoked in isolation.
- Metrics captured represent computing power (flops), memory capacity and throughput, I/O performance, network, etc.

Micro-kernels

- "Stress-test" several performance aspects of a system at once.
- Generic HPC/HTC kernels, including general and often-used kernels in Grid/Cluster environments.
- Fit for cost estimation.

Application kernels

- Characteristic of representative CG applications (extracted from CrossGrid applications).
- Capturing higher-level metrics, e.g. completion time, throughput, speedup.

2.2.2. Benchmark Targets

Each benchmark category targets a particular "layer" of a Grid architecture. In particular, we anticipate benchmarks that:

- Isolate performance properties of Sites participating in a Grid constellation.
- Capture the performance behaviour of a particular Grid constellation at a particular point in time.
- Isolate the performance capacity of Grid middleware services, which are available at the application-programming level.

Benchmarks that target a Site will report on metrics that are applicable to sites (clusters) such as flops, I/O, and network performance. Benchmarks targeting a Grid constellation will report on similar metrics. Benchmarking the middleware aims at (a) investigating the effects of middleware on end-to-end application performance, and (b) characterizing middleware performance.

2.2.3. General use case

GridBench is intended for use by a wide variety of users, ranging from application programmers to site administrators. Before invoking a particular benchmark, a user will have to make the necessary decisions based on what attributes he wishes to measure. In particular, the user will specify:

1. Which benchmark to run.
2. Benchmark-specific parameters.
3. Location (where to run it).

Parameter specification and benchmark invocation will be performed through the User Interface of the CrossGrid Portal (Task 3.1), which will employ the job submission mechanisms provided by the CrossGrid platform.

2.2.4. A Usage Scenario

The following scenario of use, is provided to describe a typical use-case of the GridBench:

George is a CrossGrid user who wants to run n-body simulations.

1. George is developing an n-body code for a molecular dynamics application. He wants to have an estimate on where he can conduct his simulations. To this end, he selects a benchmark with an N-body micro-kernel and spawns it on his local Grid site. He runs a few simulations with different input parameters, studying the scalability of measured performance, for increasing problem sizes (up to 500,000 particles and 50 time-steps) and processor numbers. After a couple of hours of benchmarking, performance monitoring, and metrics' study, George has shaped an early idea of the rough performance he can receive locally.
2. George deploys his code on the local site and runs a first simulation on a realistic problem size (20 million particles, 1000 time-steps) expecting to receive results after 24 hours.

3. While waiting for his first simulation to finish, George decides to run a larger simulation that could benefit from the aggregated computing power of the CrossGrid Virtual Organization, deploying his computation in four remote sites: two sites in Spain, one in Greece, and one in Poland.
4. To assess the potential performance improvement and the incurring cost, George configures the N-body benchmark to run on 5,000,000 particles for 20 time-steps. He submits this benchmark through the CrossGrid Portal to the four sites. The Portal reports back to him that the benchmarking cycles of the first Spanish site are reserved for the next week by someone else, and provides him with a pre-reservation for the other three sites.
5. George dispatches the benchmark to the three sites, with more modest parameters (2,000,000 particles for 10 time-steps). While the benchmark runs, he monitors its performance using CrossGrid tools and observers a much higher than expected communication to computation ratio, which leads to a very slow overall execution.
6. George aborts the benchmark and invokes a few micro-benchmarks to conclude that the network link connecting the Polish to the Spanish site is congested.
7. George re-spawns the N-body benchmark, excluding the Polish site. Following four hours of execution, the CrossGrid Portal provides George with the outcome of his benchmark: statistics about computing time, memory consumption, network bandwidth used, and an estimate of the cost to be charged for a simulation of similar size on the same nodes. George compares these results with historical data from prior N-body benchmark executions on the same and other nodes of the CrossGrid Virtual Organization. The overall conclusions are positive.
8. In the mean time, the simulation running on the local site has finished in 20 hours. Now, George has a better understanding of what he should expect from his own code and from the resources available through CrossGrid, in terms of performance and completion time. Therefore, he goes on to reserve the Spanish, Greek and Polish sites for a number of large runs of his code.

2.3. USER CHARACTERISTICS

GridBench will be geared towards the following main groups of users:

1. **End Users:** This is the group of users specified by all the applications of WP1. These users will run mostly micro-kernels and application-kernel benchmarks.
2. **Administrators:** Site administrators or VO integrators (i.e. people who put together Virtual Organizations) that need to evaluate periodically the performance characteristics of their Grid configurations and publicize the performance capacity of their nodes; they may need to run any type of benchmark.

2.4. CONSTRAINTS

Application-kernel benchmarks will have constraints similar to those of applications in WP1.

2.5. ASSUMPTIONS AND DEPENDENCIES

Application-kernel benchmarks are inherently dependent on the choices of middleware and platform by the WP1 applications. Therefore, many of the assumptions and dependencies of the WP1 applications are “inherited” by the respective benchmarks. As these are still in limbo for some of the WP1 applications, assumptions and dependencies of the application-kernel benchmarks remain undefined.

Micro-kernel benchmarks will assume a working installation of Globus and MPICH-G2 at each test-bed site (as dictated by WP4). Platform uniformity, to the extent dictated by WP4, is another important dependency.

2.5.1. Dependence on other Tasks:

- GridBench is also dependent on Task 2.4 for the development of the Performance Measurement mechanisms. Mechanisms such as G-PM and HLAC, the notion of "probes" and the use of the APART Specification Language will form some of the building blocks of GridBench.
- GridBench is dependent on Task 3.1 for the provision of the final User Interface.
- There is also dependence on Task 3.3 for archival of benchmark data (in the form of monitoring) if it is decided that archival is necessary.

2.6. APPORTIONING OF REQUIREMENTS

By the end of Month 6:

- Propose a set of performance metrics to describe the performance capacity of CrossGrid configurations and application performance.
- Design benchmarks representative for the applications of WP1 and their implementation into the programming environment architecture.

By the end of Month 12:

- Design test scenarios

By the end of Month 18:

- Use the feedback from testbeds to refine or redesign the benchmark code and architecture.

By the end of Month 24:

- Design Grid-enabled GridBench architecture
- Design test scenarios in a Grid environment.

3. SPECIFIC REQUIREMENTS

N/A

3.1. EXTERNAL INTERFACES

N/A

3.2. FUNCTIONS

As stated earlier, GridBench will provide three families of benchmarks:

3.2.1. Micro-benchmarks

Micro-benchmarks measure performance properties of programming primitives used at the application level for inter-process communication, I/O, invocation of basic Grid services, etc. These are applicable to API's that are provided by the middleware and MPICH-G2, and that have a strong and measurable impact on the overall Grid or Site performance. Middleware micro-benchmarks will seek to isolate the performance properties and extract the performance capacity of middleware services that are invoked by typical CrossGrid applications through the CrossGrid API's.

Some of these benchmarks will address the performance of the CE's (Compute Elements) and SE's (Storage Elements) directly. The properties of CE's and SE's will be measured either individually or collectively at a site.

Especially in the case of SE's, the performance each *autonomous* SE (i.e. a machine or a set of machines that is independent of other SE's) will be measured. "Autonomous" is stressed because it is not our purpose to study performance at the single machine level. An autonomous SE could be a cluster of machines running (for example) PVFS, in which case the performance of the whole set of machines will be measured collectively, not each node of the specific PVFS system.

3.2.2. Micro-kernels

Micro-kernels will be based on existing kernels (i.e. pieces of code that form the core of a computation) with well-known codes and well-known behaviour that appear often in parallel processing. (Such as NAS, SPLASH, etc.)

- For "stress testing" and identification of basic performance of sites and Grid constellations. They characterize a set of grid nodes constituting a Grid constellation.
- Codes with HPC and/or HTC requirements.
- Measure Computing power (flops) at the Grid level, Storage/Network capacity, cost estimate, scalability, etc.
- To investigate the effects of middleware on end-to-end application performance.

The user will be able to choose to run all or a subset of the developed benchmarks at the Grid constellation or Site level.

3.2.3. Application kernels

- Characteristic of representative CG applications.
- Higher-level metrics (completion time, throughput, speedup…).

The applications from WP1 will be analysed and, in cooperation with WP1, the critical kernels will be identified. These kernels will then be utilized in benchmark development.

Application kernel benchmarks at the **Grid level** will be a measure of performance of a Grid constellation. A *Grid constellation* is the set of Grid nodes belonging to the same Virtual Organization and used collectively in the solution of a particular problem.

Benchmarking at the Grid constellation level is necessary, as the performance of a Grid constellation cannot be derived directly from the aggregate performance of participating Sites: when using a Grid constellation to run a particular application, different components of the application may run at separate sites, and communicate or transfer files over Internet. Therefore, the benchmark should mimic the collective use of the constellation's resources in the execution of that particular application.

3.3. PERFORMANCE REQUIREMENTS

There will be upper bounds on completion time for benchmarks, which are to be determined and imposed.

3.4. LOGICAL DATABASE REQUIREMENTS

GridBench will use different database engines only indirectly through other components (Such as Task 2.4 components or the Globus GRIS)

3.5. DESIGN CONSTRAINTS

Benchmarks must be compatible with the basic building blocks of CrossGrid. Therefore benchmarks are designed based on Globus, DataGrid software and other middleware as dictated by WP1 and WP4.

3.6. STANDARDS COMPLIANCE

Benchmarking code will comply with MPICH-G2 where applicable.

3.7. SOFTWARE SYSTEM ATTRIBUTES

N/A

4. INDEX

Application kernel, 9, 10, 14

Benchmark invocation, 7, 8, 10, 13

Constellations. *See* Grid Constellations

Constraints, 8, 11, 14

DataGrid 4, 5, 8, 14

Dependencies, 3, 12

Flops, 9, 10, 13

Globus 4, 5, 7, 8, 11, 14

Grid constellations, 4, 5, 6, 8, 9, 13

Interfaces, 7, 8, 13

Metrics, 4, 5, 6, 7, 10, 14

Micro-benchmarks, 9, 13

Micro-kernel, 12

Middleware, 4, 8, 9, 10, 11, 13, 14

MPICH-G2, 6, 8, 12, 13, 14

Usage Scenario, 10

Use case, 10

Users, 11

Virtual Organization, 4, 6, 9, 11, 14

VO. *See* Virtual Organization