# K2 blackpearl Performance Testing

## GUIDANCE FOR PERFORMANCE TESTING OF THE K2 PLATFORM

March 27, 2009

In order to evaluate the performance of a K2 environment, guidance in this document can be used for customers to set up their own performance testing. Actual K2 performance can vary greatly across environments and scenarios due to variable hardware and network configurations, as well as the types of business solutions developed and technologies with which K2 blackpearl is integrated.

*SourceCode Confidential*

## INTRODUCTION

K2 blackpearl provides the platform for delivering process-driven applications that improve business efficiency through the use of visual design tools and scalable server components.  As an enterprise-class platform, K2 blackpearl caters to numerous installation configurations, allowing for great flexibility in hardware and software configurations that will best meet the needs of the customer.  The performance testing guidelines in this document can be used to determine the optimal hardware/software combination for a K2 blackpearl installation in a customer environment.

## ABOUT K2 BLACKPEARL

With K2 blackpearl, developers and business users can use information that already exists and processes that have already been built to create flexible, scalable applications that span employees, departments, organizations and line-of-business systems. These process-driven applications can be set up to automate and manage business processes or pull together business processes, people, services, information and systems into a single application that helps drive business.

K2 is built on .NET and leverages existing investments in Microsoft software infrastructure — Microsoft Office SharePoint Server, InfoPath, Visio, Visual Studio, Office, Exchange, BizTalk, Office Communications Server and others.

## AUDIENCE

This paper is intended for system administrators and operations personnel responsible for the installation and performance of the K2 blackpearl platform at their company. The tests and guidance provided in this paper are meant to help customers estimate performance in their own environments. The performance testing methodologies, as explained, should be used to configure tests in your environment to provide baselines that can be used to monitor overall health of the K2 blackpearl platform.

## K2 PERFORMANCE TESTING METHODOLOGIES

K2 employs a carefully designed performance-testing methodology to ensure testing is performed accurately and adheres to business scenarios encountered by customers. Performance testing has two aspects – load testing and business-scenario testing. Load testing ensures that the system is designed to handle a maximum and realistic load of transactions and provide a baseline for scenario testing. Business-scenario testing ensures that the system can meet the customer’s business needs.

During load testing, the components of the application are tested to ensure they can handle the data efficiently. Components such as workflow, SmartObjects and reports are the primary components tested. Performance testing of these components sets a benchmark of performance as a foundation for the business scenario testing. Functional testing is completed prior to performance testing. Testing is also completed to ensure that the logical flow of the user/business transaction works from end to end.

The emphasis of business scenario performance testing is on the overall scenario. The test scenarios therefore are developed around a full end-to-end business process. Factors such as user think times and wait times are factored into the test scripts where response times are measured in relation to user experience, rather than the performance of the components at maximum load.

## TESTING TOOLS USED

The primary testing tool to use is Visual Studio Team System 2008, with a test controller and test load agents for generating the load. Typically a load of 1,000 virtual users per load agent can be simulated on the Office Standard desktop computer available with the Visual Studio Test Load Agent; however, executing 1,000 virtual users on a single load agent could saturate the load agent in other ways (e.g., the network card) and cause testing to fail. For a load of 1,000 virtual users, two load agents running 500 users per load agent should be used, thereby minimizing load-agent limitations.

To facilitate automated testing, the Visual Studio Team System 2008 Test Edition is used. An additional ASP.NET application can be created that adds functionality to the automated test tool for performing K2 blackpearl-specific actions using the K2 API, such as:

- Start a Process Instance
- Retrieve a User's Worklist
- Action a Worklist Item

To gather metrics and test results additional tools including SQL Server Performance Dashboard, SQL Server Profiler, and Perfmon.exe can be used. For more information on the K2 specific actions and the K2 API, please refer to the *K2 Developer Reference*.

## LOAD TESTING OVERVIEW

Load testing establishes the baseline performance of the software. The load test results are analyzed and reviewed within the guidelines of the test goals and objectives. Generally, testing starts with smaller, more realistic load scenarios and then increases the load systematically. This approach helps to pinpoint problems that are encountered as load is increased and helps identify opportunities for streamlining processes.

Load testing incorporates a variety of test types that are performed by a specified number of virtual users. Typical types of load tests include:

- **Unit tests** – A unit test can be generated from any existing code method of any class.
- **Web tests** – A Web test records specific test actions when a URL to an existing site is provided. Web tests can also be parameterized and correlated to provide stable tests that can execute with different test data per iteration.
- **Generic tests** – Generic tests execute executable files.
- **Manual tests** – Manual tests are used when human interaction is the only possible way to execute the test or part of the test.
- **Ordered tests** – An ordered test acts as a container for the above types of tests and provides ordering functionality, or precedence, to manage the timing and sequence of the different tests in the overall testing process.

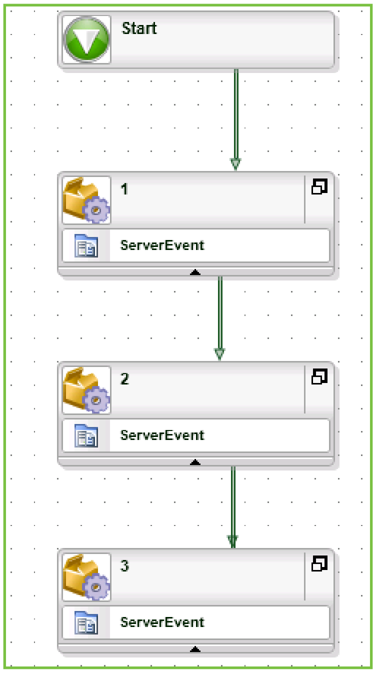
In load testing, various configuration options are available to create a scenario suited to the test needs. For each virtual user specified in a test, a thread is spawned on which the test is executed. Think times allow a break between test iterations, for each thread independently. Load patterns can be specified as constant load or step load. Constant loads start all threads (virtual users) at the same time. Step loads increase the thread count (or number of virtual users) by a specific number at a given time interval. During the test, the performance counters of all machines involved in the test are monitored through Windows Performance Monitor (Perfmon.exe).

### SCENARIO TESTING OVERVIEW

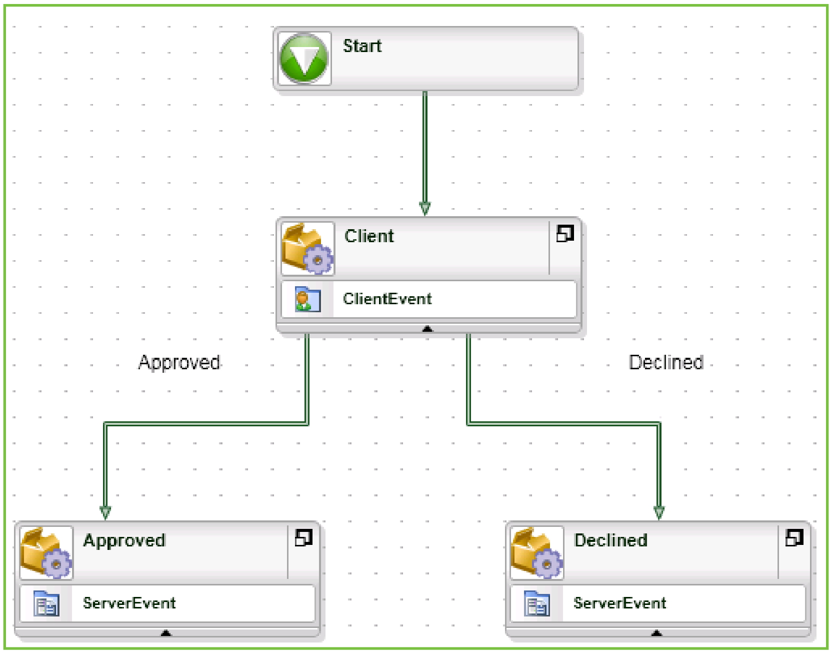
Scenario testing follows the load testing. Scenarios are created based on logical and/or user transaction flows. The scenarios are reviewed for accuracy and signed-off. Test scripts and data are then created and tested to provide a solid foundation for the performance testing. Then the scenario tests are loaded and executed. The scenarios are executed both with and without client events. Introducing a client event to the scenario simulates user interaction with the application. A ramping up (or step load) style is used to simulate real-world-user logon rates and helps to pinpoint any performance issues. In a ramping up approach, a defined number of users are added to the process at specific time intervals (e.g., five users are added every 10 seconds).

Four basic scenarios can be used to performance test the K2 blackpearl application:

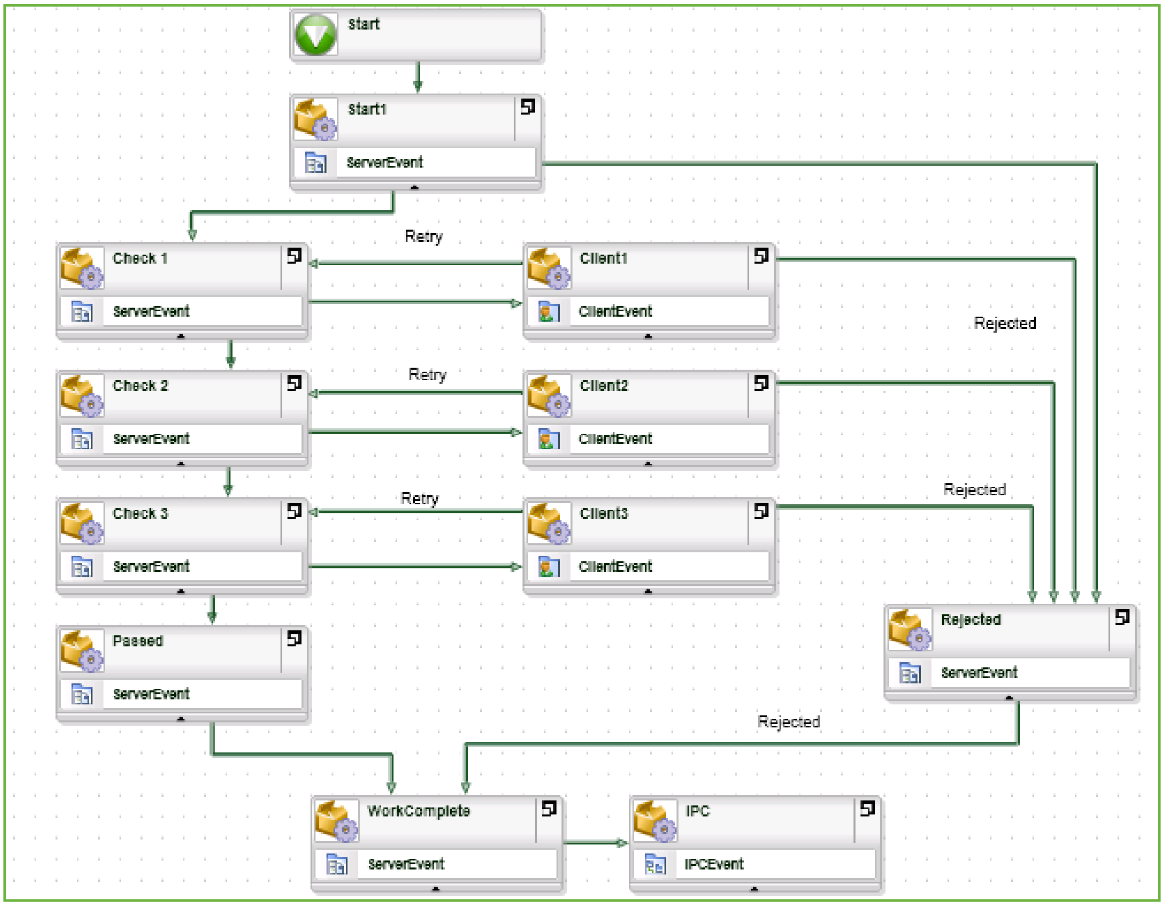
1. **Simple process with no client event** – Since no client events are included within this process, when an instance starts it will progress through the three activities without pausing for human interaction. This process instance should complete shortly after it starts.



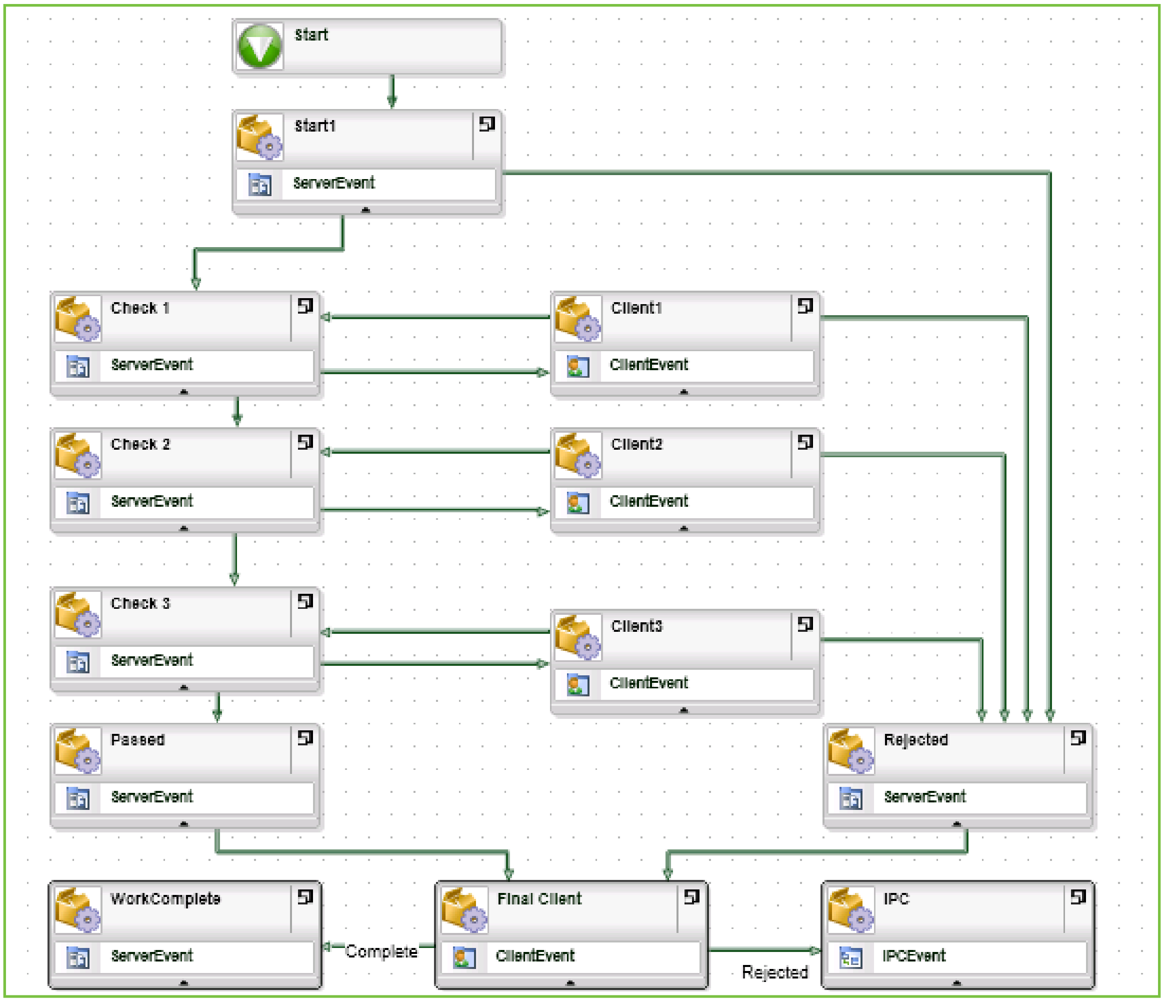
2. **Simple process with client event** – In this process definition, a client event is created in the first activity after the start of the process. The process instance starts and then pauses processing after it creates the worklist item(s) for this client event. The test is designed to simulate the wait time for a person to take action on the worklist item.



3. **Large process with no client event** – The large process is used to show impacts on the application performance due to a larger, more complex process definition. For each test run, a static route is followed through the complex process and not all events are used. In this test all client events are bypassed, so when an instance starts it will process through all steps without pausing for human interaction.


Start
Start1
ServerEvent
Check 1
ServerEvent
Retry
Client1
ClientEvent
Rejected
Check 2
ServerEvent
Retry
Client2
ClientEvent
Check 3
ServerEvent
Retry
Client3
ClientEvent
Rejected
Passed
ServerEvent
Rejected
ServerEvent
Rejected
WorkComplete
ServerEvent
IPC
IPCEvent

4. **Large process with client event** – The large process is executed again with client interactions included in the test script. Again, a single static path is used for each test run.



Depending on the processes you intend to use in your environment, these can be customized or actual business processes can be used to test the performance of your environment.

## SUMMARY

K2 blackpearl provides a highly-scalable platform for building and executing process-driven applications. Performance characteristics are based on many factors including hardware configurations, software configurations, network load, user client hardware, user usage profiles (process originator vs. process participant, SmartObject single item vs. multiple item listings, single report vs. all reports, etc.). Mapping the data presented in this document as it pertains to the hardware and software configurations tested is key to applying the results to your environment.

In summary, with K2 deployed in a large, distributed environment (two K2 Server nodes, two K2 Workspace nodes, two K2 for SharePoint nodes, and one SQL Server with K2 for Reporting Services), customers can expect to service enterprise volumes of new process instances, SmartObject queries and report renderings.

## BEST PRACTICE RECOMMENDATIONS

The following minimum hardware configurations are recommended. Optimal configurations for the K2 server and the SQL server would have more powerful processors and more memory. For the SQL server hard disks, RAID level 1 or 10 is recommended.

| Machine role | K2 component | CPU | Memory | Hard disk drive |
|---|---|---|---|---|
| **K2 Server node*** | K2 Server | Intel Core 2 Duo 2.8 GHz | 4 GB DDR2 | 200 MB** |
| **Web Server node*** | K2 Workspace | Intel Pentium D 3.4 GHz | 2 GB DDR2 | 200 MB** |
| **SharePoint Web front end*** | K2 for SharePoint | Intel Core 2 Duo 3.4 GHz | 4 GB DDR2 | 200 MB** |
| **SQL Server** | K2 Databases<br>K2 for Reporting Services | Intel Core 2 Duo 3.4 GHz | 4 GB DDR2 | 200 MB** |

*Two Servers load balanced, each with the recommended hardware configuration

**Required disk space for just K2 blackpearl, additional disk space may be required for prerequisites

## FOR MORE INFORMATION

Please refer to the *K2 blackpearl Getting Started Guide* for additional information regarding prerequisites and the various K2 components.

**CORPORATE HEADQUARTERS**
4042 148th Avenue NE
Redmond, WA 98052
USA
**PH** +1 (425) 883 4200
**FAX** +1 (425) 671 0411

**EMEA HEADQUARTERS**
26 Worple Road
Wimbledon
London
UK
**PH** +44 (0) 845 612 0912
**FAX** +44 (0) 845 612 0911

**APAC HEADQUARTERS**
9 Shenton Way #06-02
Singapore
068813
**PH** +65 6327 4110
**FAX** +65 6327 4120

[ WWW.K2.COM ]

The information provided relates to pre-release software products, may include features only available after installation of additional add-ins, and may be substantially modified before the commercial release. This information is provided for informational purposes only, and SourceCode Technology Holdings, Inc. makes no warranties, expressed or implied, with respect to this document or the information contained within it.