

# Surveyor

from Lexient Corp.

## Technical Overview

### What is Surveyor?

Surveyor is a code analyzer for software development teams that reduces the overall costs associated with application extension, migration, and integration activities. It does this by streamlining the development cycle, automating the code analysis process, and unifying development teams.

Teams enjoy these benefits by using Surveyor to:

- Examine and interact with code elements' structure, interactions, and relationships
- Visually navigate code elements just as one can browse web pages
- Review code metrics and other analytical information
- Automatically document code

### How does Surveyor work?

Surveyor from Lexient captures, analyzes, and builds an enterprise-wide library of meta-data for a development project. Once built, team members can use Surveyor's intuitive interface to examine, traverse, and manipulate source code in order to help them work together to accomplish design, analysis, and coding tasks.

First, Surveyor captures, or parses, source code just as a compiler does. In fact, the parsing method used is designed to closely emulate the compiler or interpreter used by a particular development organization. For C/C++, over 25 different compiler/target (as well as generic) configurations are supported. Like a compiler, Surveyor's parser conducts a full semantic parse of 100% of the source files, analyzing all symbols present in the code.

Extensive work has been done to ensure the quality of Surveyor's parsing engine. Currently, Surveyor supports the following languages:

- C (Non-ANSI, ANSI, K&R C) (with or without inline assembly)
- C++ (with or without inline assembly)
- Java (1.0 or 2.0)
- COBOL (ANSI '74, ANSI '85, IBM OS/VS COBOL, Micro Focus COBOL, and others)
- Tcl/Tk (versions < 8.1)

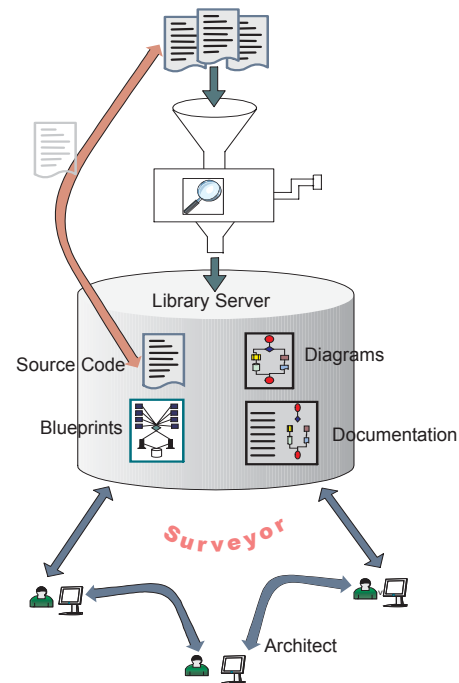
After capture, a thorough analysis is conducted on the resulting 'marked up' source code. This analysis builds a set of meta-data on the source, containing metric, structure, interaction, and relationship information for all of the analyzed symbols in the development project.

The goal of the analysis process is to enable users to:

- Clearly and easily understand complex existing source-code
- Determine the intent of the application's originators
- Receive clear impact analysis information before, during and after code changes
- Provide overview metrics and documentation to automate code reviews and inspections

Once code is fully read in and analyzed, an entity-relationship-meta-database is created. This XML compliant database contains information on all of the symbols in the project, ranging from primitive data types, such as `int`, all the way up to files and directories.

Changes to the source code can be inserted into the database in a variety of ways, depending upon an organization's



Surveyor enables teammates to communicate about diagnostics and other analyses visible to all authorized constituents

needs. Changes to source code at the local and enterprise level can also be incorporated, in order maintain an appropriate level of synchronicity between the source code in the project, and the Surveyor Library.

Once the Surveyor Library is built and on-line, information about the source code in a project is available through a rich and intuitive interface. The interface is designed to accommodate the variety of different ways in which developers are accustomed to interacting with source code within their own particular tools. The display can be fully customized to suit a particular perspective and mode that developers are most comfortable using.

Source code elements of interest within the library can be selected and viewed in a number of different ways:

- Structural, interactions, relationships
- Flow charts of functions, procedures, & programs
- Metric information
- Source code text (via integration with popular editors)

In order to organize and better manage the natural complexity present in source code, each of the symbols present in the library (such as files, classes, functions, and variables), may be interacted with in many ways, for example:

- Grouped in any way
- Filtered through structured queries, configurations, or completely customized
- Bookmarked

The depth and breadth of the ways in which users can visually interact with their source code elements provides developers with a new level of understanding of an existing body of source code. The richness of the interaction in turn facilitates comprehension and speeds development activities.

### Capturing Original Intent

Surveyor puts the original author's knowledge in the hands of an entire development organization quickly and easily. The original intent of what the author created is embodied in the source code that they wrote. This intent is captured through the meta-data creation process, and is communicated back to the user in a variety of ways, including the following:

- Flow-charting of functions, procedures & programs shows the logic paths that the original author took in the code
- Containment of member functions and other elements within classes, files and machines
- Call-diagramming shows the critical interactions between various source code elements
- Viewing the author's comments as a whole, or integrated within the code and the flow-chart views provides a level of hyper-understanding that was previously unavailable

Surveyor's functionality highlights the original intent of the author in a much faster and more accessible manner than was previously available to development teams. Making this information available for any project 'on demand' after Surveyor's automated analysis process has been shown to lead directly to reducing overall development expenses.

### Unifying the Development Team

Surveyor provides a development team with a common rich interface to their code resources. The interface has been constructed in such a way that both designers and coders, senior and junior members of a team can interact with code resources in a way which facilitates collaboration between team members.

Marathon 'white-boarding' sessions to achieve coordination between stake-holders can be eliminated with Surveyor. No matter how complicated the spaghetti code, Surveyor's capture and analysis facility automatically creates whiteboard diagrams for teams in an interactive, realtime environment, right on everyone's desktops.

Coordinating activities between team members is eased since all team members are able to seamlessly share code visualizations and organizations. Partitioning, containment, review, and assignment activities are simplified due to the common visualization between various constituents.

### Streamlining the Development Process

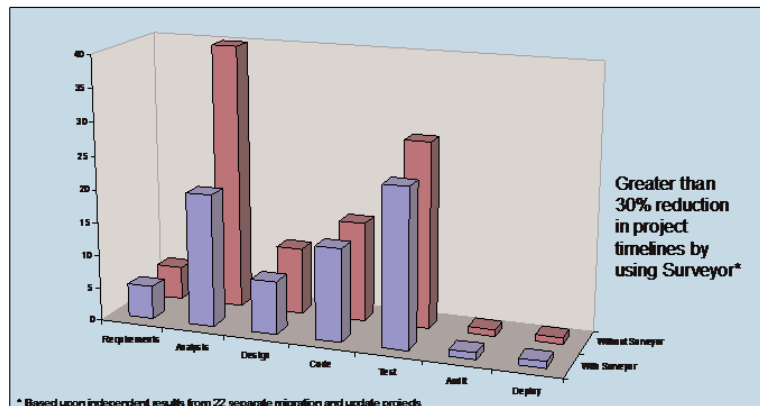
Most development organizations respond to changing business requirements by initiating an iterative cycle of analysis, design, coding and testing. Surveyor enables several of these steps to be conducted at the same time, and by the same personnel, which speeds both the quality of and the rate at which this code is able to be produced.

Thus, designers and coders can work more closely together, thereby preventing miscommunications, improper implementations, and inefficient executions of designs. By providing a forum which enables the analysis, design, and coding process to be streamlined, team members are able to work more closely together, speeding accomplishment of project milestones.

### Integration with Current Tools

Surveyor was designed with an open structure to allow for integration with other team-critical development tools. Not only can developers use the IDE or editor of their choice, which is seamlessly synchronized with Surveyor's various presentations, but Surveyor also provides a punch-out of information to major Source Code Management tools.

This ensures that teams can get the benefits of Surveyor while easily giving management the progress information they need within the prevailing processes and tools. Lexient is committed to constantly integrating additional tools across the entire developer toolset currently available in the marketplace.



*Surveyor has been shown to provide development teams with an average of greater than a 30% reduction in project timelines*

### Perspective and Focus

One of the greatest challenges facing the typical developer is to maintain a sense of perspective. They must be able to see the big picture, while paying due attention to little details. In a single interface, Surveyor provides this needed sense of perspective.

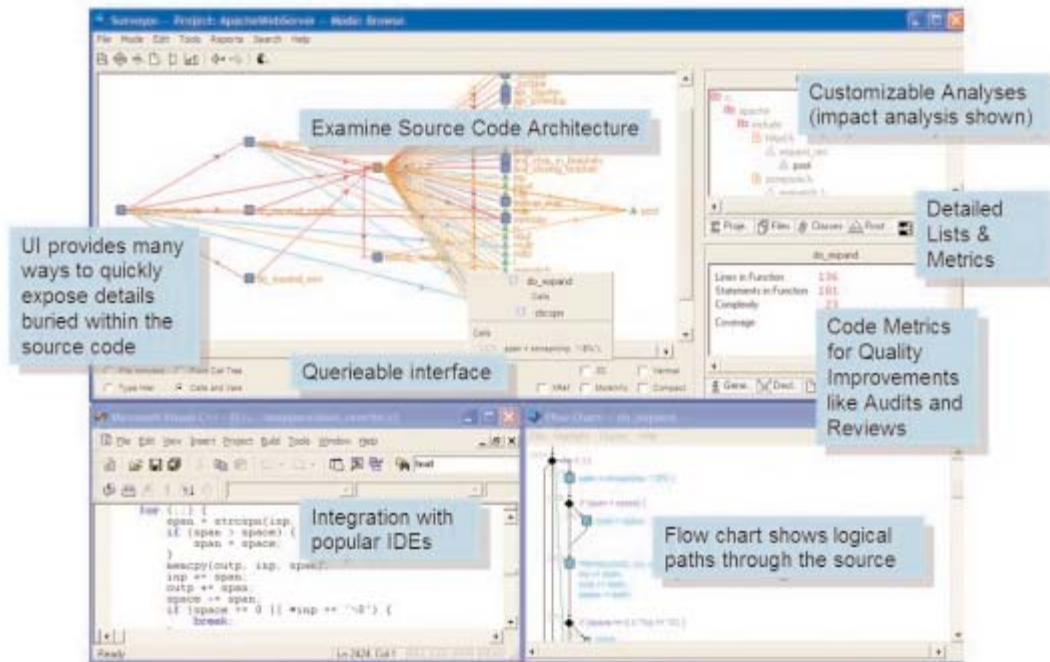
While maintaining a sense of the relationship to the whole structure of an application, developers can use Surveyor to narrow their focus down to examine the text or logic paths within a particular function, or broaden their focus to examine global interactions and relationships. This facilitates quicker understanding of complex code, and eases design and coding tasks.

## Application Extensions Done Better

Surveyor adds a powerful tool to an organization's development team by enhancing communication between team members, providing an automated analysis process, and streamlining the various iterative steps in the development

cycle. By becoming a part of an organization's current development process, Surveyor delivers on the goal of increasing the speed and quality of completed software, and decreases overall software development costs.

## Enhanced Source Code Understanding with Surveyor



## Other Features

Surveyor includes the following additional features:

- **Automatic Documentation** – call-tree, source-flow, resource usage, flow-charts, metrics & project catalogues, presented in formats including HTML, RTF, ASCII, and Visio.
- **Code Reviews** - lines of code, comment ratios, counts of statements, variables, declarations, all are presented within an enforceable standards scheme
- **Reports** – various reports highlighting quality metrics
- **Custom Query Facility** – query the meta-database of captured information about your source code
- **Automatic Impact Analysis** – algorithms perform an analysis showing the impact of proposed changes to functions, classes and variables.
- **UML Diagrams** – generated in an interactive form presented in the interface or created as a part of auto documentation.
- **Custom Views** – create custom views showing just the right amount of information in visual presentations.
- **Grouping** – any source code element can be grouped and encapsulated in one representation to help manage the complexity of your environment.
- **Bookmarks** – bookmark any source code element for fast reference
- **Filtering** – Surveyor contains robust filtering in many different ways and in various places in the interface
- **3D or 2D** – developers can navigate their source code with either three-dimensional or two-dimensional representations
- **Summary Reporting** – class and file summaries, function catalogues, and variable usage catalogues available in dynamic form in the interface or in static form through the automated documentation facility.

## Hardware Requirements

The following hardware and software resources are required to run Surveyor on Sun, Hewlett-Packard and SGI workstations, or on a PC running Windows or Linux. Contact sales support about system requirements on other platforms.

## Client

- color monitor
- disk space - 10 MBytes (installation)
- swap space - 50 Mbytes
- for Windows Systems, Pentium, 500 MHz, 256 MB RAM

## Server

### Windows Systems:

- Dedicated 1 processor Pentium 3, 1.3 GHz, 1 GB RAM, (Win2K or WinXP)
- Concurrently in-line 2 processor Pentium 4, 1.8 GHz 1 GB RAM, (Win2K or WinXP)

### Other Systems:

Concurrently on any existing systems using OS below

### Operating System:

- for HP-UX - 9.X or 10.X
- for IRIX - 5.3 or later, or 6.2 or later
- for Linux - 2.X
- for Solaris - 2.3 or later (5.2 or later)
- for SunOS - 4.1.3 or later
- for Windows - Win95, Win98, WinNT 4.0, Win2K, WinXP
- for other vendors - contact sales support