UNISYS
imagine it. done.

white paper

An Overview of SURE

A Software Configuration Management Solution for Developers of Composite Applications for ClearPath Systems

simplicity

Table of Contents

Introduction

What is SURE?

This paper describes the Unisys ClearPath SURE solution for process-centric, workflow-enabled software management and change control. SURE provides life cycle support for software development on Unisys ClearPath Series systems. SURE enables developers to achieve complete development control across multiple operating environments, including MCP, Windows, and Unix/Linux in any combination.

SURE provides a unique solution for the consolidation and use of repeatable processes across all elements of an enterprise to ensure the integrity of the development for "composite" applications. Composite applications involve development artifacts from multiple operating environments and require special discipline to achieve business objectives.

SURE is an excellent tool for software houses, providing support for the development of packages and for the distribution of releases. SURE can also be used by clients who receive packages from software vendors and then apply local changes to customize the release.

The core systems of an enterprise often run on central servers. To enhance and guarantee the quality of this vital functionality, a comprehensive system for software management and change control is essential. Today, many major organizations depend on SURE to maintain the quality and integrity of their mission critical applications.

A common IT workflow

Software development and IT projects require the collaboration and integration of many people across a wide variety of roles in an organization including

- Developers
- Testers
- Team leaders
- Project leaders
- Release coordinators
- Management

Those fulfilling these different roles often employ different development tools and procedures from one another. This management of projects with multiple, independent, and local applications such as spreadsheets, email, databases, tracking systems, and project management systems can be extremely inefficient, if not fundamentally threatening to the success of the operation.

While SURE was created as a source control system, it has evolved into an IT workflow-based system allowing the full integration of desktop tools and IT procedures. A common IT workflow within an organization will dramatically improve both the productivity of the team and the quality of the work product.

Improved quality and productivity

Currently, many organizations using ClearPath do not use a software control management system. Instead, they use their own procedures to archive source modules, to control changes, and to take software into the production environment. If these procedures are designed and implemented well, they may seem satisfactory to an organization. However, such procedures are often rigid and restrictive, providing only limited control and functionality while also becoming quite expensive. With SURE, you may upgrade the software management and change control services provided on any ClearPath server in order to improve the quality of your software.

The SURE solution does not prescribe any development methodology like "waterfall" or Rapid Application Development. Instead, SURE facilitates the automation, control, and management of the development methods and IT procedures used.

For most IT departments, change is equated with risk, and organizations often attempt to maintain the status quo until they are forced to change. There are a number of questions that must be addressed in order to evaluate the current development process and to determine if there is a greater risk inherent in the status quo than there is in changing to a more efficient system. Such questions include

- Can a developer easily (within a minute) compare sources from different stages and identify why (for what task, incident, or requirement) changes were made?
- Can changes be made at any stage of the development without complex procedures and without risking the loss of work?
- Can the build process be reproduced at any stage?
- Can the developer guarantee that an executable corresponds to a specific version of a source and determine the reasons for any changes made?
- Does the current mechanism produce impact analysis, and does it detect impact overlap?

- Can multiple developers work concurrently on the same source without risking the loss of work?
- Is the turnover procedure fully automated and audited, without the need for any paper work?
- Does the current mechanism provide spreadsheets that allow project leaders to verify and discuss the status of a project?
- Does the current mechanism generate the release letter?

Prior to their implementation of SURE, most current SURE customers were using a turnover process based on circulating a sign-off sheet. In general, a paper-based turnover approach takes more than two hours of processing and requires someone to fill and update it, route it to the authorized persons for signatures, and then provide follow-up. With SURE, the turnover process is totally automated and electronically coordinated, thus enabling a much more efficient process and the ability to respond to time-sensitive demands. Additionally, SURE provides a full audit of the process.

Unique to Unisys ClearPath MCP Developers

The MCP operating system provides a variety of functions specific to the architecture of ClearPath Series. Therefore, generic software control management systems can only provide limited support for the MCP platform. SURE is a unique product for Unisys ClearPath Series MCP developers in that it provides functionality tailored and optimized to run on the MCP operating environment even while managing the development processes and artifacts of multiple operating environments.

SURE seamlessly integrates with the MCP security system, file system, disk pack organization, networking facilities, usercode structure, sumlog, and other ClearPath Series specific mechanisms.

Support for distributed environments

Not only does SURE allow control and management of MCP software changes and projects, but it also works in controlling development for Windows and Unix/Linux artifacts. As organizations progress towards implementation of web-based extensions to their core business applications, this capability becomes increasingly vital. And if there is contemplation of a Services Oriented Architecture (SOA) involving the integration of many software components from different technologies, SURE is a required tool.

Integration with IDEs

SURE supports every type of file in Windows or UNIX operating system environments and integrates seamlessly with IDEs like open source Eclipse and Microsoft Visual Studio. This integration allows developers to use the SURE functions directly from an IDE and to check-out and check-in files from menu choices within an IDE without switching between applications. This functionality greatly improves efficiency for developers.

A Visual User Interface

The principal interaction with SURE is through a rich, graphical Explorer-like interface. This natural style of a hierarchically structured interface ensures that users are immediately productive and comfortable. Developers adapt to SURE extremely quickly – usually within a few hours. The SURE explorer is tailored by job role when an end user logs on to SURE. By tailoring the interface to the specifics of a job role, the efficiency of each staff member is assured. The use of dynamically updated colored icons for status information gives immediate context to the state of any task activity and minimizes the need to query continually as work progresses.

Topics addressed in this paper

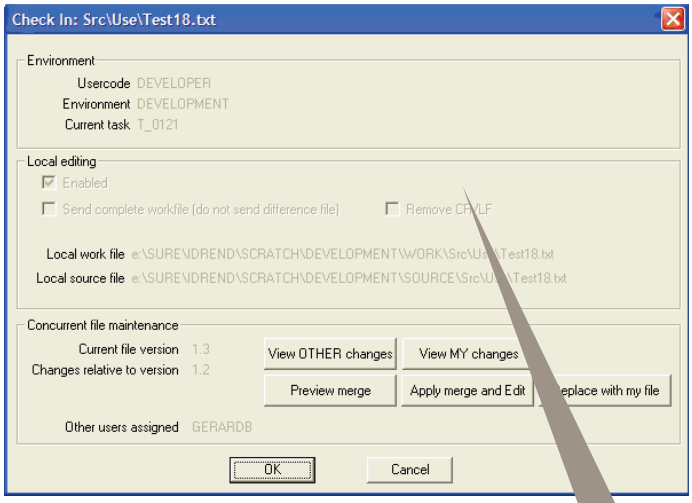
This paper provides a technical overview of the functionality contained in SURE

- A flexible source control system allowing concurrent source maintenance
- Configurable environments (stages like **develop**, **test**, **acceptance**, and **production**) matching requirements in procedures within a company
- An integrated task (incident or requirement) system driving and enforcing the procedures defined within an organization
- A fully automated build process based on changes and procedures, maintaining the deployment of executables in the configured environments
- Modernizing Unisys ClearPath MCP development
- Seamless integration with the current IDEs like Microsoft Visual Studio and Eclipse as well as Unisys tools such as CANDE, Programmer's Workbench, and third-party editors

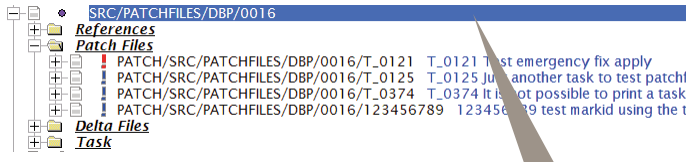
A Flexible Source Control System

Parallel development

SURE contains a flexible source control system that allows for concurrent source maintenance. For PC/Windows/Unix/Linux files, SURE will integrate the “other changes” in the source, helping to avoid errors and unnecessary duplication of effort. For MCP files, SURE supports the PATCH mechanism allowing parallel development and total control of deployment.



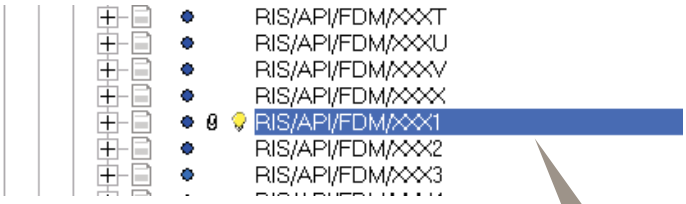
The check-in dialog, allowing concurrent source maintenance and support



File containing patch files, allowing for parallel development and different deployment schedules

Information attached to a source

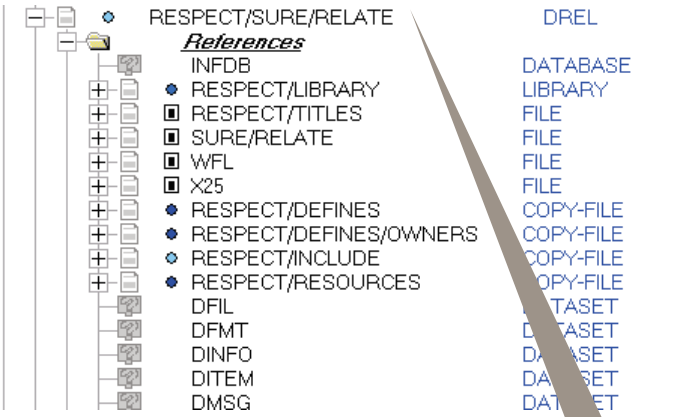
The source control system allows information to be connected to a source. This can vary from notepad-style documentation to .zip files containing complete analysis in Microsoft Word and Microsoft Visio applications.



File containing an attachment and a reminder

Dynamic cross reference

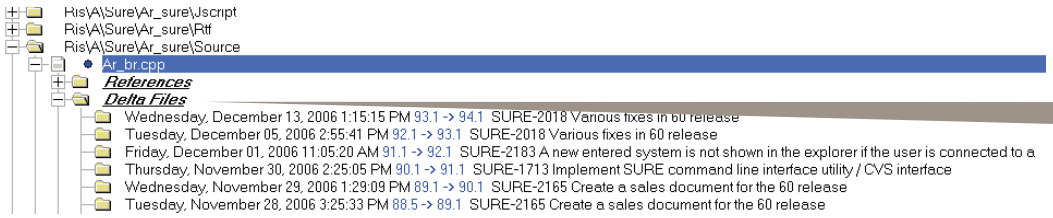
The source control system is the foundation of SURE. It is built on top of a true repository system that keeps track of all source related information and includes a dynamic build cross reference between sources and data structures, stored as references. This information is useful to a database administrator, for example, to issue a compile command for all sources using a specific dataset.



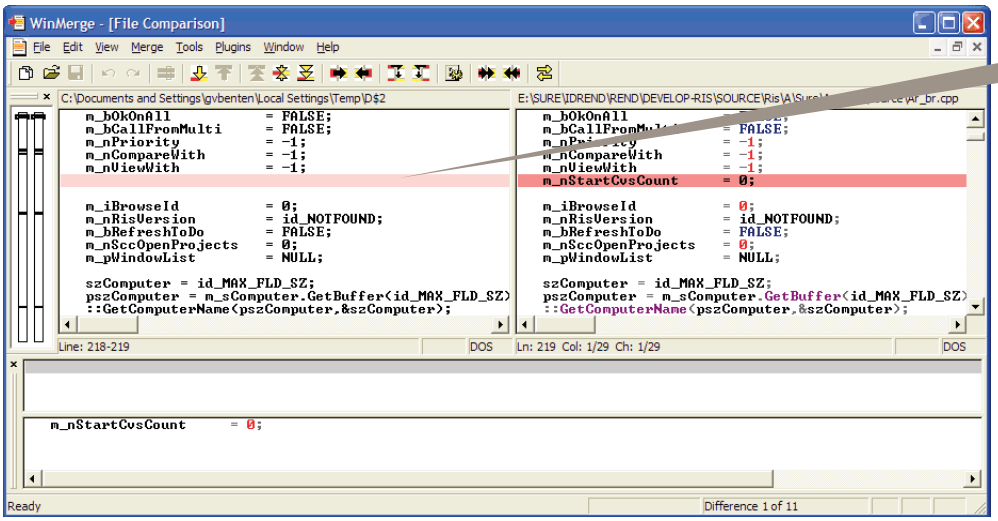
A source file with its references

Delta files and differences

The source control system uses a dynamic history configuration that allows a company to define how long different historical objects such as delta files need to be archived. Delta files allow for graphical comparison of files and concurrent source maintenance. Additionally, SURE has a built-in procedure for managing hot fixes.



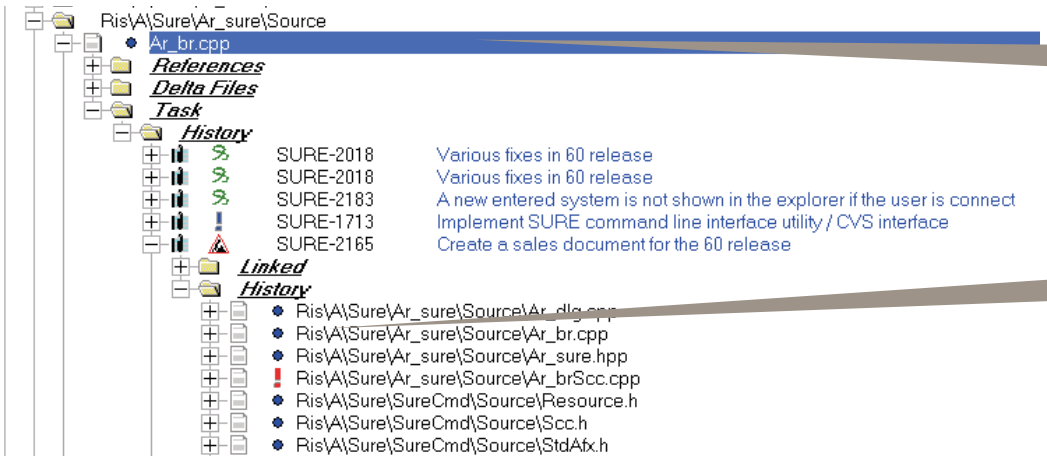
A folder containing delta files which can be used for graphical comparison or rollback purposes



The graphical comparison utility

Integrated task history

A task is a logical unit of work in SURE. The source control system is integrated with the task tracking system, thereby enabling the maintenance of a task history.

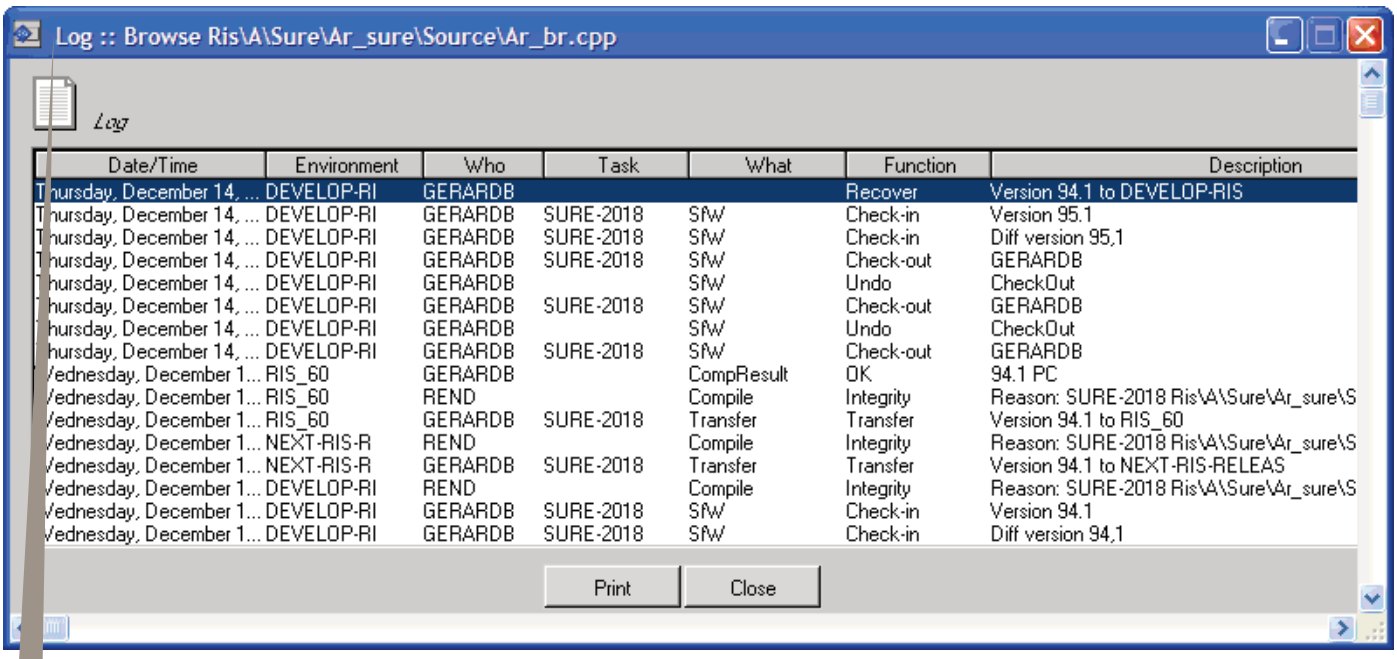


A file contains a set of tasks indicating why the file was changed.

For the task SURE-2165, all of these files were changed.

Audit log

The source control system uses extensive logging. This logging answers questions about **who** changed **what**, **why**, and **when** and satisfies the requirements of auditors.

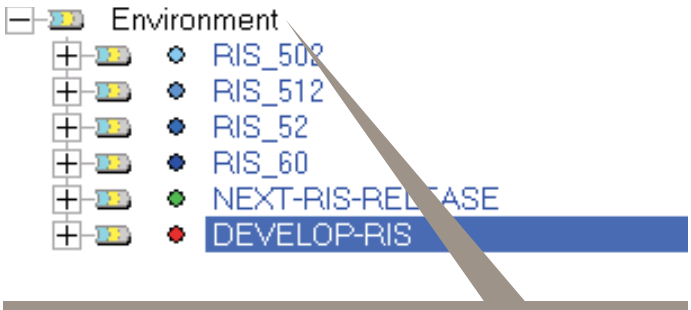


Date/Time	Environment	Who	Task	What	Function	Description
Thursday, December 14, ...	DEVELOP-RI	GERARDB			Recover	Version 94.1 to DEVELOP-RIS
Thursday, December 14, ...	DEVELOP-RI	GERARDB	SURE-2018	SfW	Check-in	Version 95.1
Thursday, December 14, ...	DEVELOP-RI	GERARDB	SURE-2018	SfW	Check-in	Diff version 95,1
Thursday, December 14, ...	DEVELOP-RI	GERARDB	SURE-2018	SfW	Check-out	GERARDB
Thursday, December 14, ...	DEVELOP-RI	GERARDB		SfW	Undo	CheckOut
Thursday, December 14, ...	DEVELOP-RI	GERARDB	SURE-2018	SfW	Check-out	GERARDB
Thursday, December 14, ...	DEVELOP-RI	GERARDB		SfW	Undo	CheckOut
Thursday, December 14, ...	DEVELOP-RI	GERARDB	SURE-2018	SfW	Check-out	GERARDB
Wednesday, December 1...	RIS_60	GERARDB		CompResult	OK	94.1 PC
Wednesday, December 1...	RIS_60	REND		Compile	Integrity	Reason: SURE-2018 Ris\A\Sure\Ar_sure\S
Wednesday, December 1...	RIS_60	GERARDB	SURE-2018	Transfer	Transfer	Version 94.1 to RIS_60
Wednesday, December 1...	NEXT-RIS-R	REND		Compile	Integrity	Reason: SURE-2018 Ris\A\Sure\Ar_sure\S
Wednesday, December 1...	NEXT-RIS-R	GERARDB	SURE-2018	Transfer	Transfer	Version 94.1 to NEXT-RIS-RELEAS
Wednesday, December 1...	DEVELOP-RI	REND		Compile	Integrity	Reason: SURE-2018 Ris\A\Sure\Ar_sure\S
Wednesday, December 1...	DEVELOP-RI	GERARDB	SURE-2018	SfW	Check-in	Version 94.1
Wednesday, December 1...	DEVELOP-RI	GERARDB	SURE-2018	SfW	Check-in	Diff version 94,1

The command log of a file, used for auditing purposes

Configurable Environments

SURE allows for configurable environments (stages like develop, test, acceptance, and production) matching the requirements described in the procedures within an organization.
An environment contains all software components for the organization. So if an organization employs different application systems, each with different versions, then an environment contains the specific versions of the application systems in that stage. An environment can also be considered a predefined baseline for an organization when using other source control systems terminology.

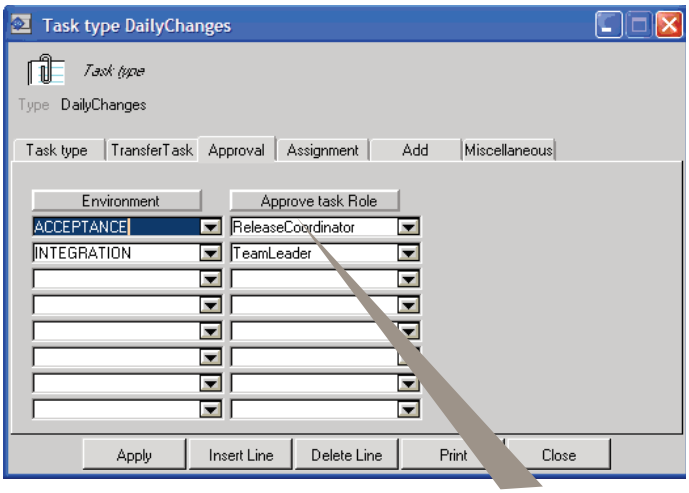


Environments for a software supplier supporting historical releases

Environments for an organization supporting on-going software development

Configuring workflow

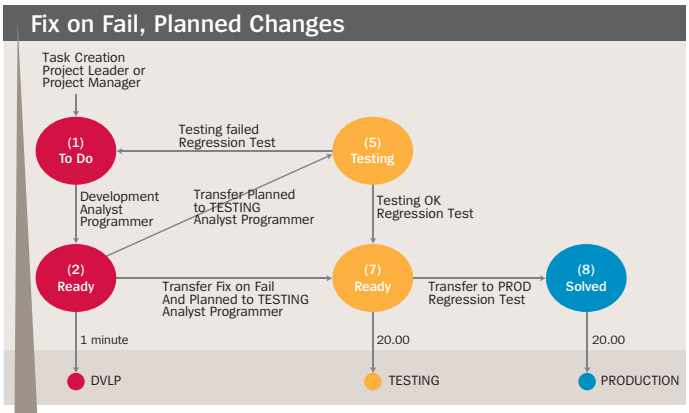
What is the difference between baselines created after completion of software and predefined baselines? Using predefined baselines allows a defining of the workflow, the authorization, and the approval process between the different stages. For example, it can be configured so that an employee role named “release manager” must approve a task before it is deployed to the environment named “production.” Therefore, by defining the workflow, authorization, and approval, the organization’s procedures are automated, audited, and enforced by SURE.



The task type allows for specific workflow procedures. Here, roles are assigned for approval in each environment.

Workflow diagram

Using this configurable model allows creation of a workflow per task type that defines precisely how the procedure is implemented within an organization. These workflow diagrams are then used as information to help SURE users.



Example simple workflow diagram showing the activities and roles performing the different functions

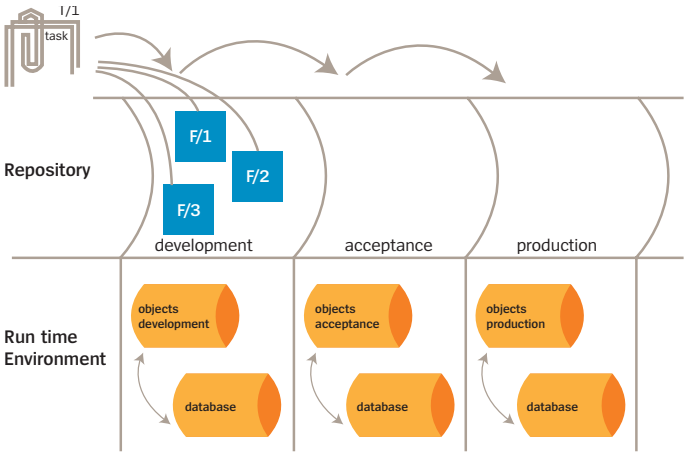
An Integrated Task System

SURE contains an integrated task (incident or requirement) system that drives and enforces the procedures established within an enterprise. Many organizations use an incident management system and a requirement registration or planning system. The SURE task system contains the basic functionality present in the systems previously discussed. If these systems are not already in place, the SURE task functionality can be utilized. If these systems are already in place and functioning satisfactorily, then the SURE task system can be linked to them.

The advantage of integration is that SURE connects all of the changed sources and software components to a task that identifies an incident or a requirement. This task can then be promoted to another stage in development, which implicitly promotes the changed sources or software components, starts the build process, and deploys the new software in the new stage. For this reason you can progress the release of incidents or requirements to final production.

This integration will automatically update the status of the tasks, eliminating the need for project leaders to manually update local spreadsheets.

- Integrated task administration in order to group changes together, implicitly allowing “non-technical personnel” to release tasks into production – in SURE, a task describes a logical unit of work
- A provision for multiple distinct environments, each containing a complete software system that matches the status of that environment, allowing quality assurance of production software to be made independent of development – an environment represents a “slot” or stage of the application development cycle (development, test, acceptance, production, etc.), and SURE has a provision for up to 8 separate environments (most clients require far fewer)



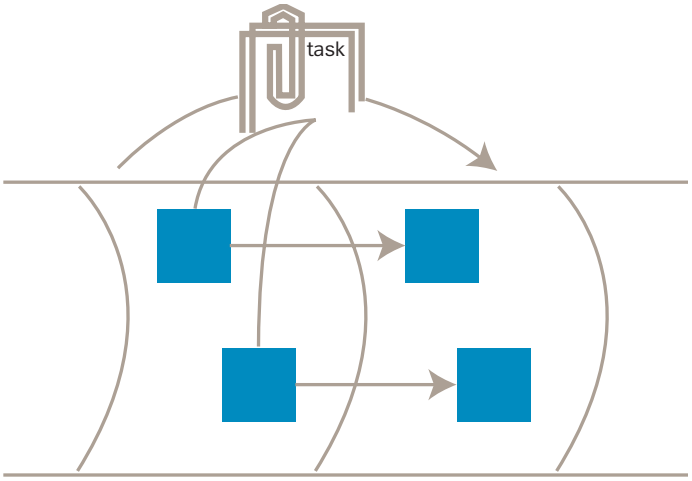
This picture shows SURE with three configured environments – development, acceptance, and production. Each of these environments contains its source modules in a “repository slot,” with the resulting objects and matching database(s) in the corresponding run-time environment.

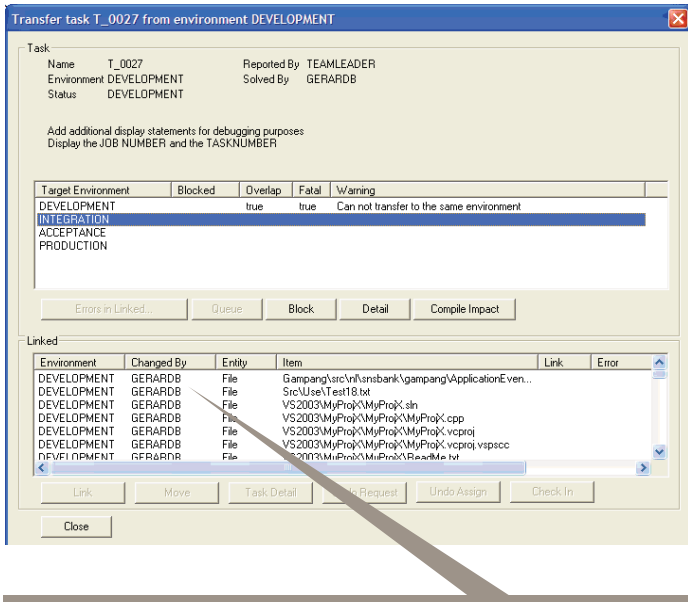
A task identifies not only a project, but also the associated developers. Thus, defining a task will implicitly define the workload for the various developers working on that project. A task can represent the work needed for any project or incident report that needs to be resolved. Tasks in SURE are hierarchical so they can be linked to represent any degree of project scope.

Workflow between the stages of the development process occurs when tasks are moved to another slot. SURE utilizes built-in queues and other mechanisms that facilitate workflow automation.

Task promotion

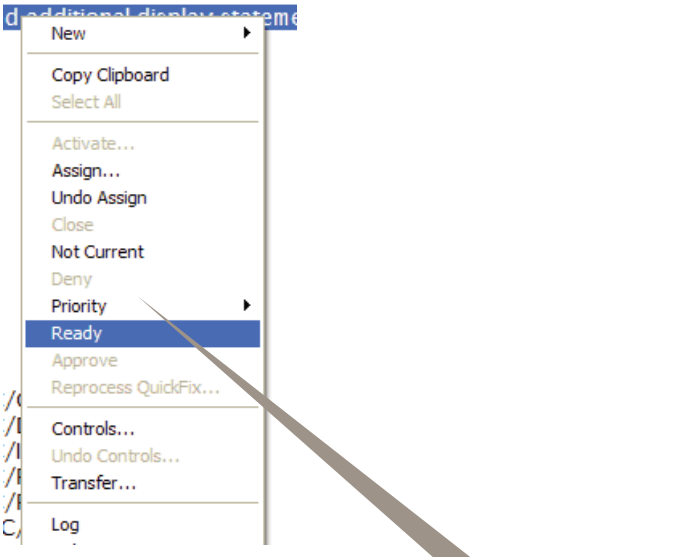
If a developer checks-out a source file, SURE automatically creates a link between the task and the source file, enabling maintenance of a complete history of multiple developers checking-out and working on multiple source modules. Transferring the changes to another environment (e.g., acceptance) is performed by transferring the task to the new environment. Therefore, the task is the actual unit of work and the sole mechanism to control changes.
In SURE, tasks are the managed unit. Developers and administrators do not manage files. It is the responsibility of SURE to record changes in files in the task so it can then maintain a complete picture of what needs to be transferred to the next slot, relieving individuals of this responsibility. This results in much higher productivity as developers can concentrate on creating new business functionality and leave the administration details to SURE.





Task promotion dialog containing all the linked changes

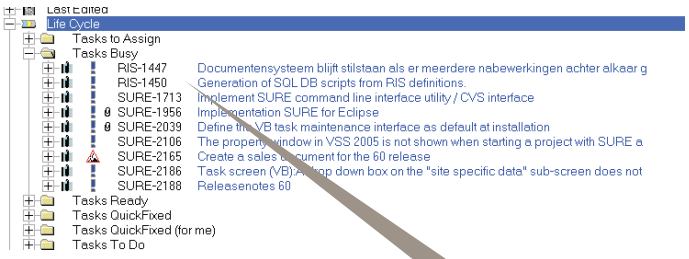
As with most IT turnover procedures, the task is the driving force in SURE. But with SURE, no longer must IT turnover procedures rely on manual input or paper. SURE employs a fully automated process with mechanisms for approval by different employee roles and maintenance of an electronic journal for auditing purposes.



A part of the task drop-down menu that allows different functions, like **Assign**, **Close**, **Ready**, **Priority**, and **Approve** according to the status in the workflow and the role of the employee

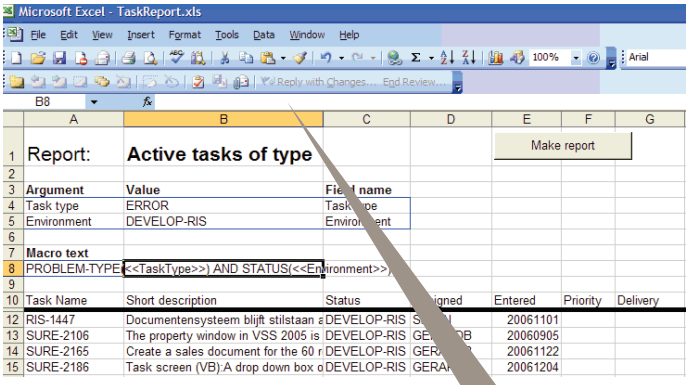
Workflow folders

SURE groups these tasks in different workflow folders, allowing those in different employee roles to view a list of tasks relevant to their respective roles. These lists can be imported in any Microsoft tool like Excel, saving a significant amount of time for project or team leaders by eliminating the need to maintain separate spreadsheets.



SURE maintains different workflow folders tailored for every employee.

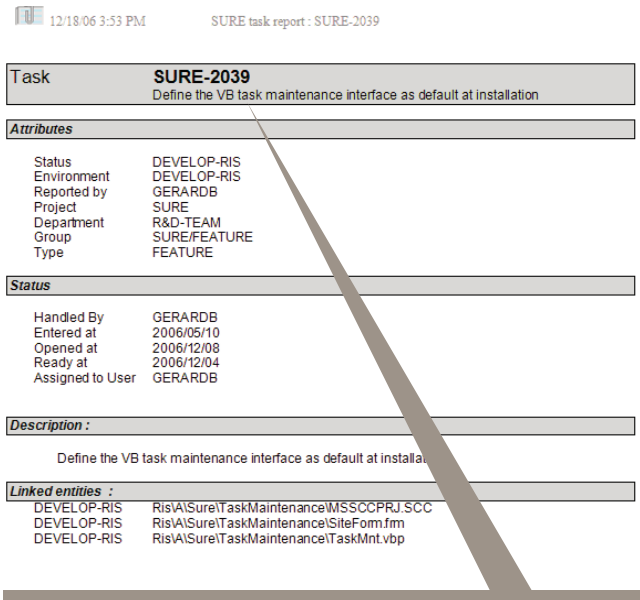
Extract task information in Excel



The SURE task list that is extracted to Microsoft Excel

Customized reports

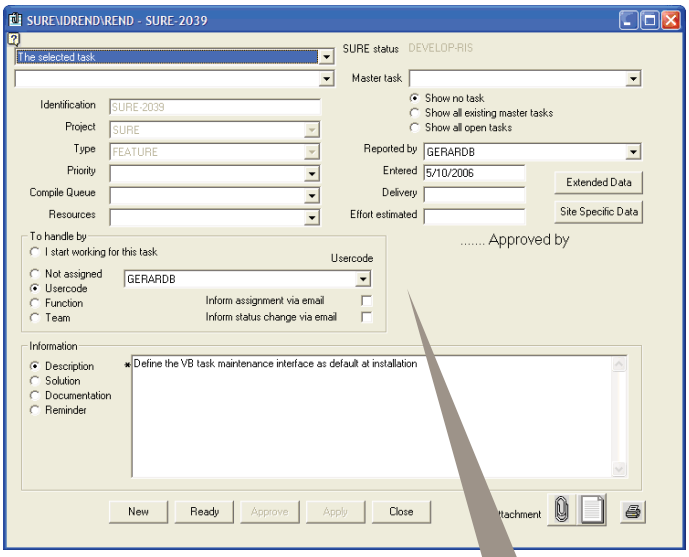
SURE allows creation of customized reports, whereby a paper version of the turnover form can easily be generated and used in the transition phase to SURE.



Example generic task report, easily customizable for specific installations

Customizable task form

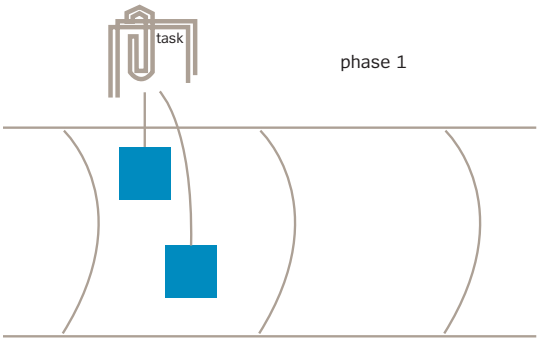
SURE contains a customizable Visual Basic program for task registration and maintenance. This, in combination with the custom definable task properties, allows customization of the task maintenance according to the enterprise's requirements.

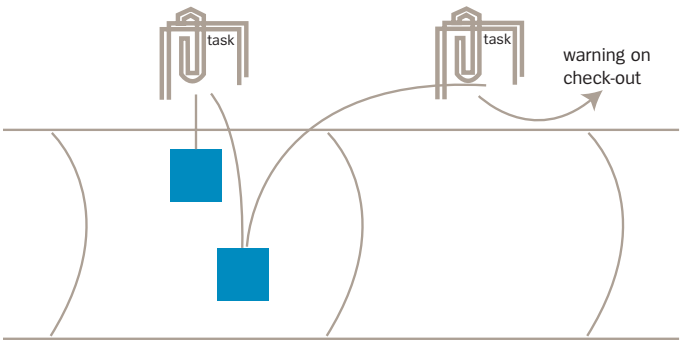


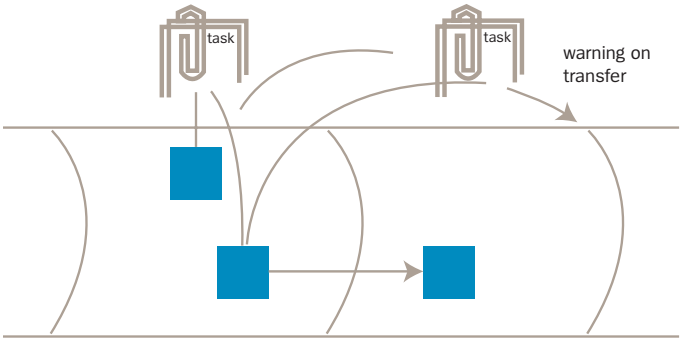
Customizable Visual Basic form

What is an overlapping task?

The relationship between task developers and task source files is a "many-to-many" relationship. For this reason, it is possible to modify a source file for multiple different tasks. In other words, such a source file contains changes for multiple different tasks. If one of these tasks is transferred to another environment, any dependent source file (which may also contain changes for other tasks) is also transferred to that environment. The SURE system allows transfer of such "overlapping tasks" but warns a programmer when he checks-out a dependent source file. It also warns any employee transferring the task.





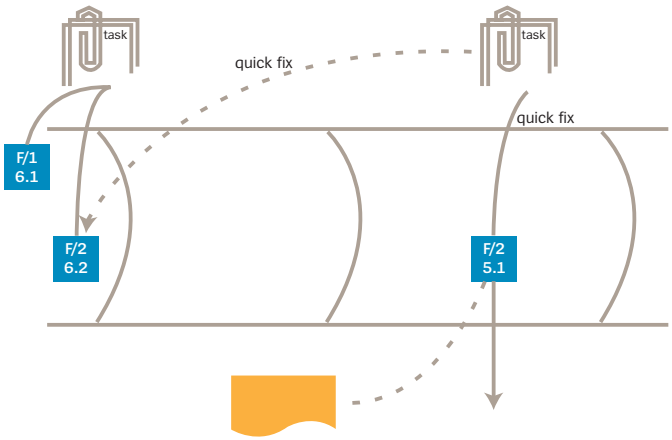


What is a quick fix?

Overlapping tasks result in a program that contains changes for different tasks. However, there may be situations in which this overlap is unacceptable. For example, in the case of solving fatal production errors, it must be possible to get rid of the fatal error without taking any unwanted new features into production. Hence, SURE notifies the developer when such changes have occurred.

By default, developers make software changes in the lowest hierarchical environment and then transfer these changes to higher environments (e.g., from development to acceptance to production). In the case of a quick fix, a different approach is used wherein the developer makes the changes in a higher environment. SURE then reports to the lower environments that these changes have been made. By insuring that changes occurring in higher level slots are reported at lower levels, SURE avoids a common problem of having the same error recur because an emergency fix in production was not applied to new versions in development.

Therefore, a quick fix is the ability to make changes to source modules in a higher environment without disturbing the development process for production environments or releases. The SURE system provides the appropriate implementation and feedback mechanisms to support quick fixes. SURE also provides a mechanism to merge patches in a lower environment.



The file F/2 was modified in development. A production error required modification of the file F/2. However, during the check-out procedure, the system indicated overlapping tasks and offered a quick fix option. This quick fix allowed the file F/2 to be changed in production so that the actual production version was fixed without the new functionality. Thereafter, the file F/2 shows the presence of a quick fix, and it is also reported in a batch listing. It is likely that the changes to F/2 made in production also need to be incorporated into F/2 in development. After this incorporation, the quick fix notification disappears.

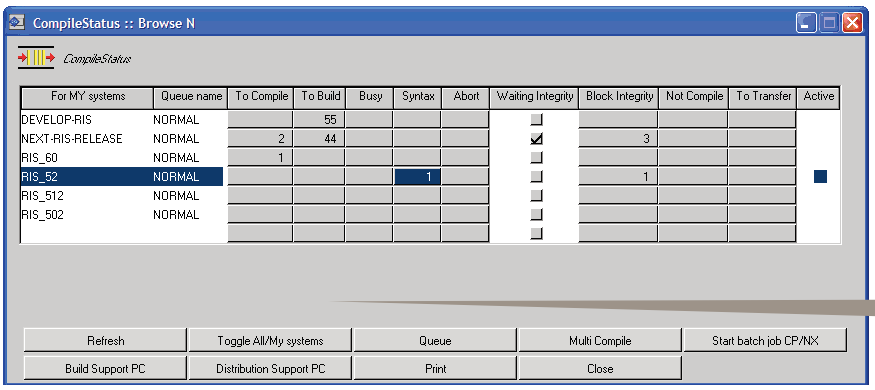
A Fully Automated Build Process

Software development projects often depend on the skills and creativity of individuals. However, this creativity should be concentrated on the development of the software, not on the control of it. SURE provides control measures to protect any project from dependency on individuals for this function. One such measure is the unique registration in the repository of file locations, security aspects, non-standard object names, or object characteristics (e.g., privileged programs). The same applies to other aspects such as program binding or any special action upon which the SURE compile software acts afterwards.

SURE allows for a complete automated build process based on the information in the repository. This build process is initiated with a single mouse click and can produce MCP object files, PC executables, and UNIX executables, all controlled from the central SURE repository. All of the information about these builds such as timestamps and error reports are stored in the SURE repository and accessible from the SURE explorer.

Daily build process

In papers about the software development process, Microsoft will often refer to a “daily build” as well as the follow-up to this “daily build” which identifies the sources that fail. SURE supports a daily build process for every defined environment based on the changed files or other software components. This means that SURE controls, starts, and logs the build process so that this information is always available and accessible from every workstation.



The compile console of SURE, holding all files changed in the different environments and files that trigger compilation

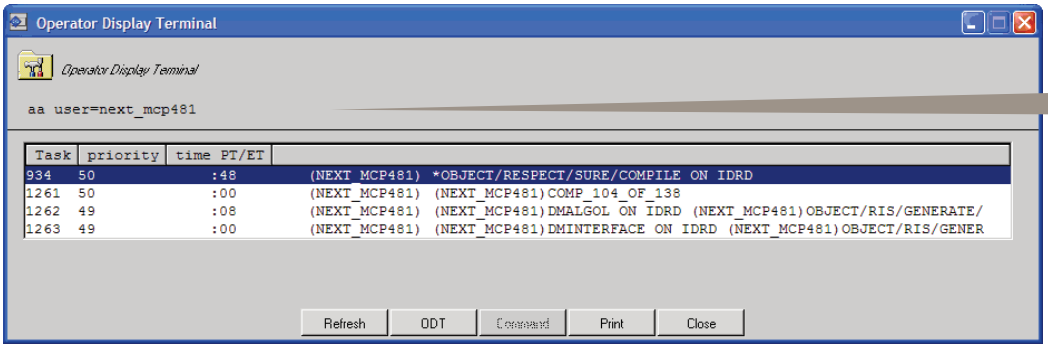
The software systems of today have a modular character, implicitly making many software modules dependent on one another. For example, checking-in a copy file requires recompilation of all modules using that copy file. Only through this process can the integrity of the application system be maintained.

The repository used by SURE has a “self-learning” mechanism with regard to dependencies between source modules. Each time a file is checked-in to the SURE environment, parsing by an internal syntax scanner takes place (based on the file kind), and SURE automatically stores all relevant dependencies in its repository. SURE also supports a manual dependency declaration which is often used for multiple platform development with common interfaces.

The information extracted from source modules is used to maintain the integrity of the application system. Therefore, checking-in a copy file will result in the compilation of all modules using that copy file. Of course, the status in one environment (e.g., development) may be different from that in another (e.g., acceptance). Unlike manual processes, SURE maintains these environments electronically, assuring visibility of all stages in the development process.

Build MCP

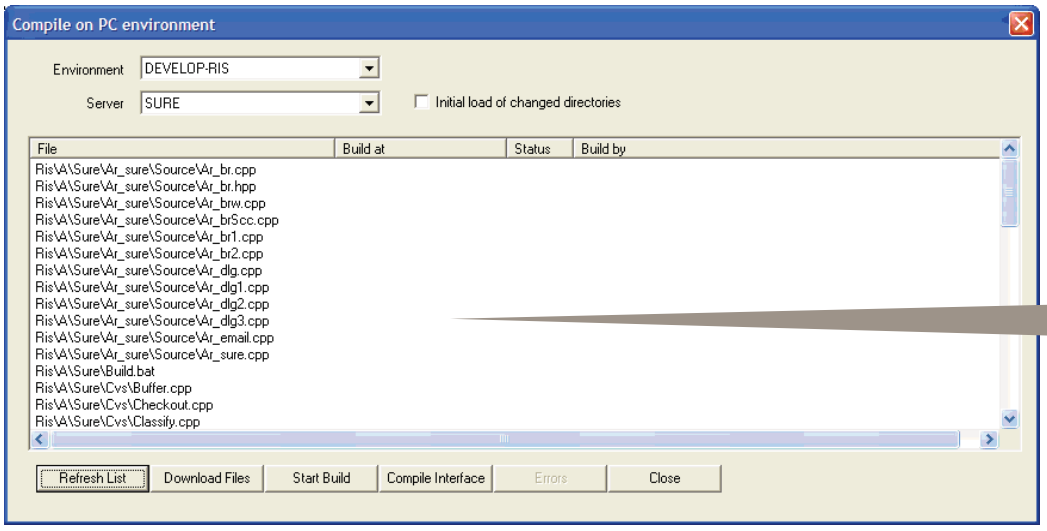
With SURE, the build for Unisys MCP is fully automated and requires minimal effort to configure.



For the MCP, a program compiles all changes files, and if a copy/include file is changed, all related files are also compiled.

Build Windows/UNIX

For Windows and UNIX, SURE can employ the current build procedures using MAKE, NMAKE, Ant, NAnt, or any other script in place. SURE can easily handle a project-based setup and allows for generic script files.



For Windows/UNIX files, SURE runs scripts connected to logical file types or build .bat files. These scripts can access any MAKE, NMAKE, Ant, NAnt, BAT, or SHELL script. The errors or results are reported back in the repository.

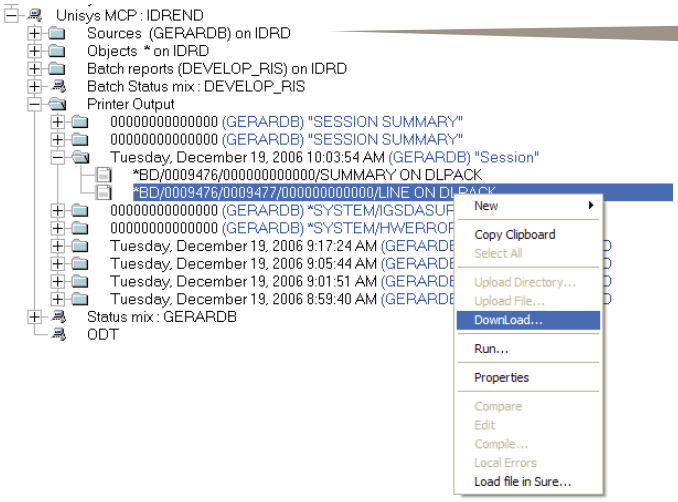
The end result of the build and deployment configuration is that everyone (authorized) can start the build process with a single click (or time-scheduled), and all results and errors are visible.

Modernizing Unisys ClearPath MCP Development

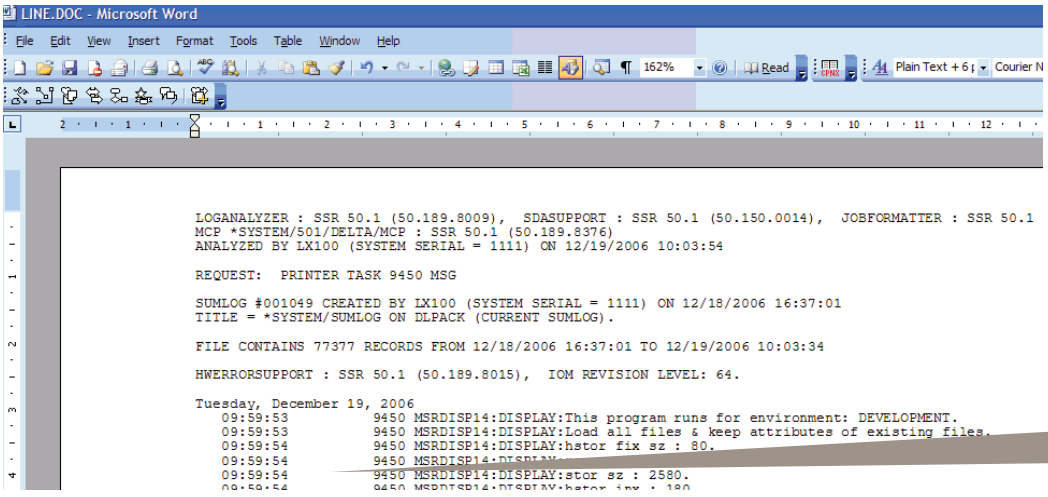
Many organizations run their core systems on MCP servers, and modernizing this development environment increases the lifetime of these systems, potentially enabling significant cost-savings.

Workbench

SURE contains a workbench showing the status on the MCP system, the downloading and viewing of print files, and PC-based editing of source files.



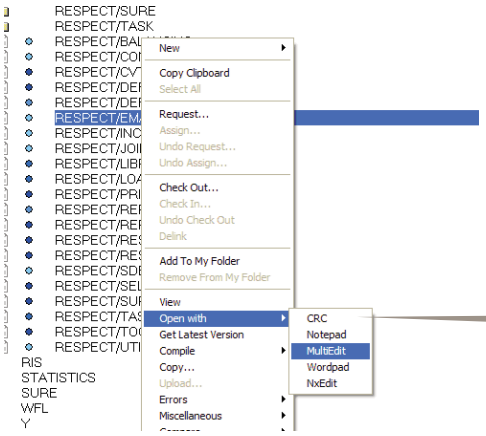
The SURE MCP explorer provides easily accessible functions for a developer. This example shows the listing of a printer backup file in Word.



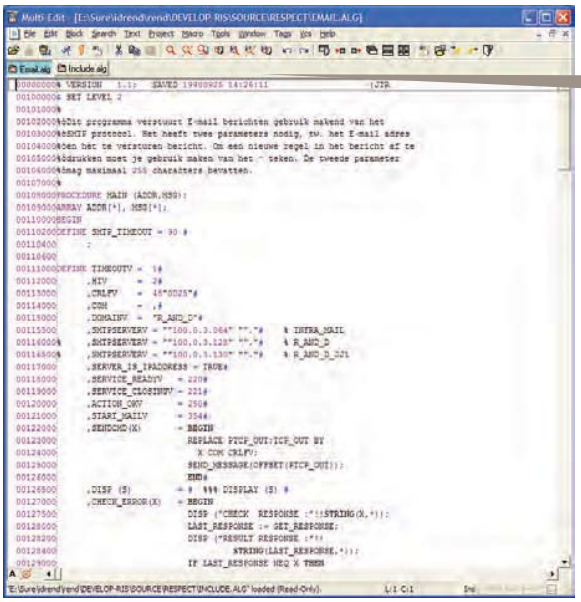
Showing a printer backup file in Word can be performed in seconds, greatly simplifying the work of developers.

Windows-based editing

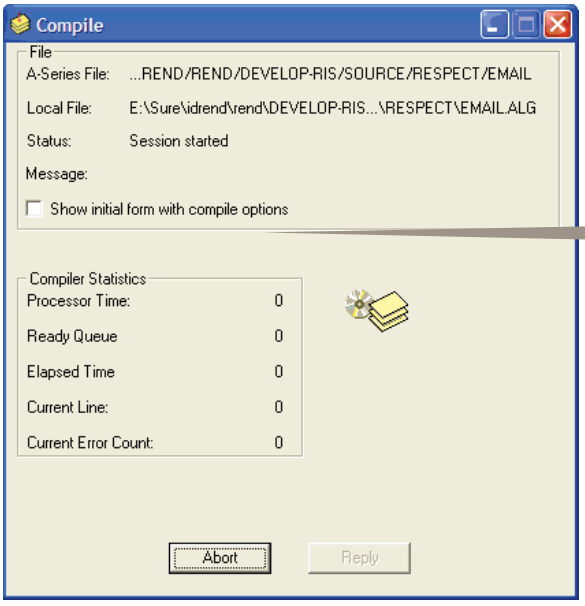
SURE allows the coding of source software using industry standard editors. The programmer is free to use Unisys Programmer's Workbench or another editor such as MultiEdit by American Cybernetics. When using a PC-based editor, SURE takes care of the required file transfers and synchronization with the MCP file system.



SURE allows the opening of MCP files using any configured PC-based editor, downloading and converting these files so that they are easily maintained.



Editing an MCP file in a commercial editor allows the opening of multiple files and makes use of the syntax coloring facilities.



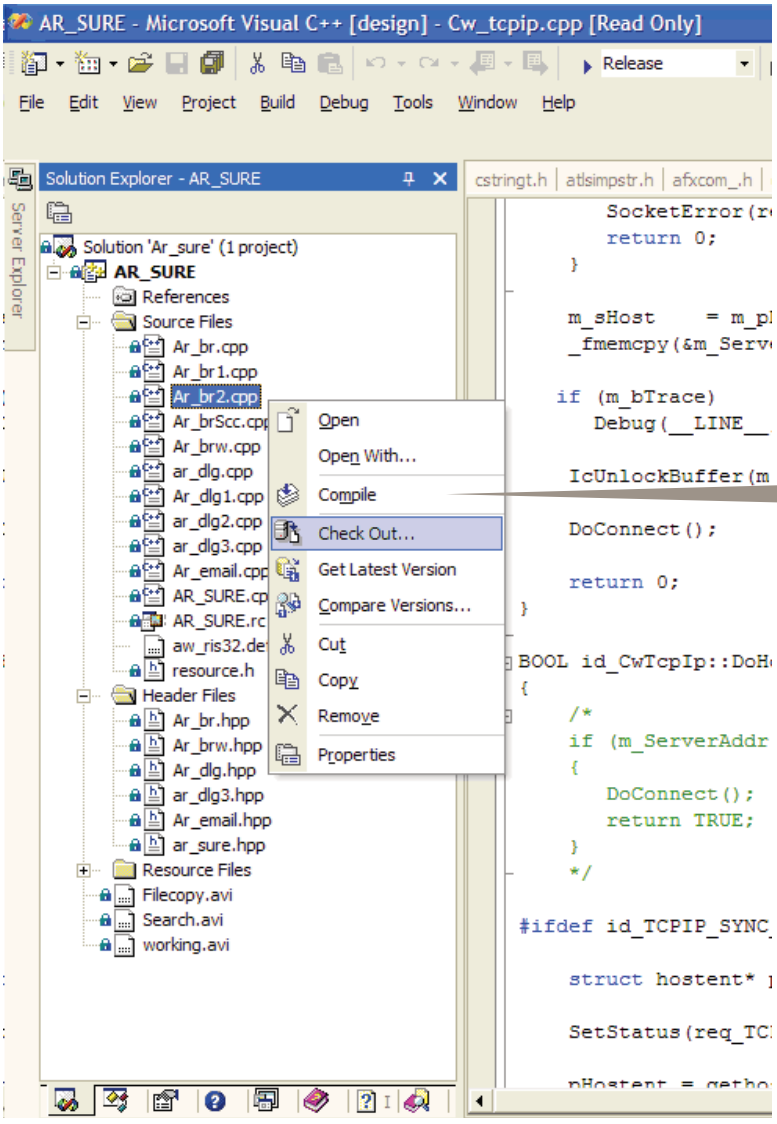
SURE includes some utilities allowing initiation of MCP compilation and error feedback into the editor, thus reducing work for developers.

A Seamless Integration with IDEs

SURE supports a seamless integration with prominent IDEs like Microsoft Visual Studio and open source Eclipse. Whether developers use Microsoft Visual Studio or Eclipse and regardless of the editor used (MultiEdit, Programmer's Workbench, Notepad, or other Unisys MCP editors), SURE offers a seamless integration of these development tools. Developers can check-out and check-in files from their IDE without switching between applications.

Microsoft Visual Studio

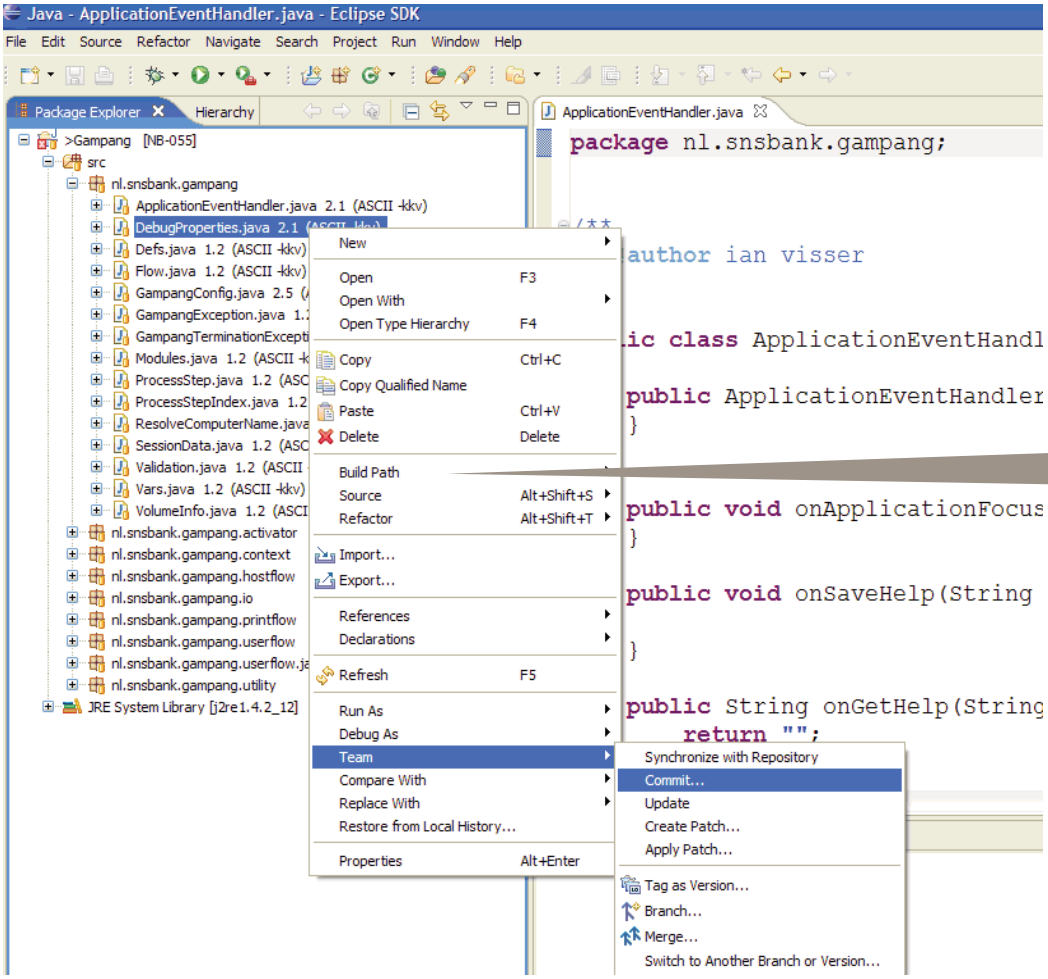
For supporting Microsoft .NET tools, SURE provides the Microsoft SCC interface for communication, allowing for the use of tools like Microsoft Visual Studio 2005, Microfocus COBOL, Powerbuilder, Visual Basic, etc. Also, Unisys Agile Business Suite uses Microsoft Visual Studio 2005 as the foundation for the solution, and therefore, AB Suite can store artifacts in the SURE repository and participate in the managed environment along with other components in composite application development.



Microsoft Visual Studio links up to the SURE software using the SCC interface. This allows the developer access to the SURE functions from within Microsoft Visual Studio, thereby easing developer workload.

Eclipse

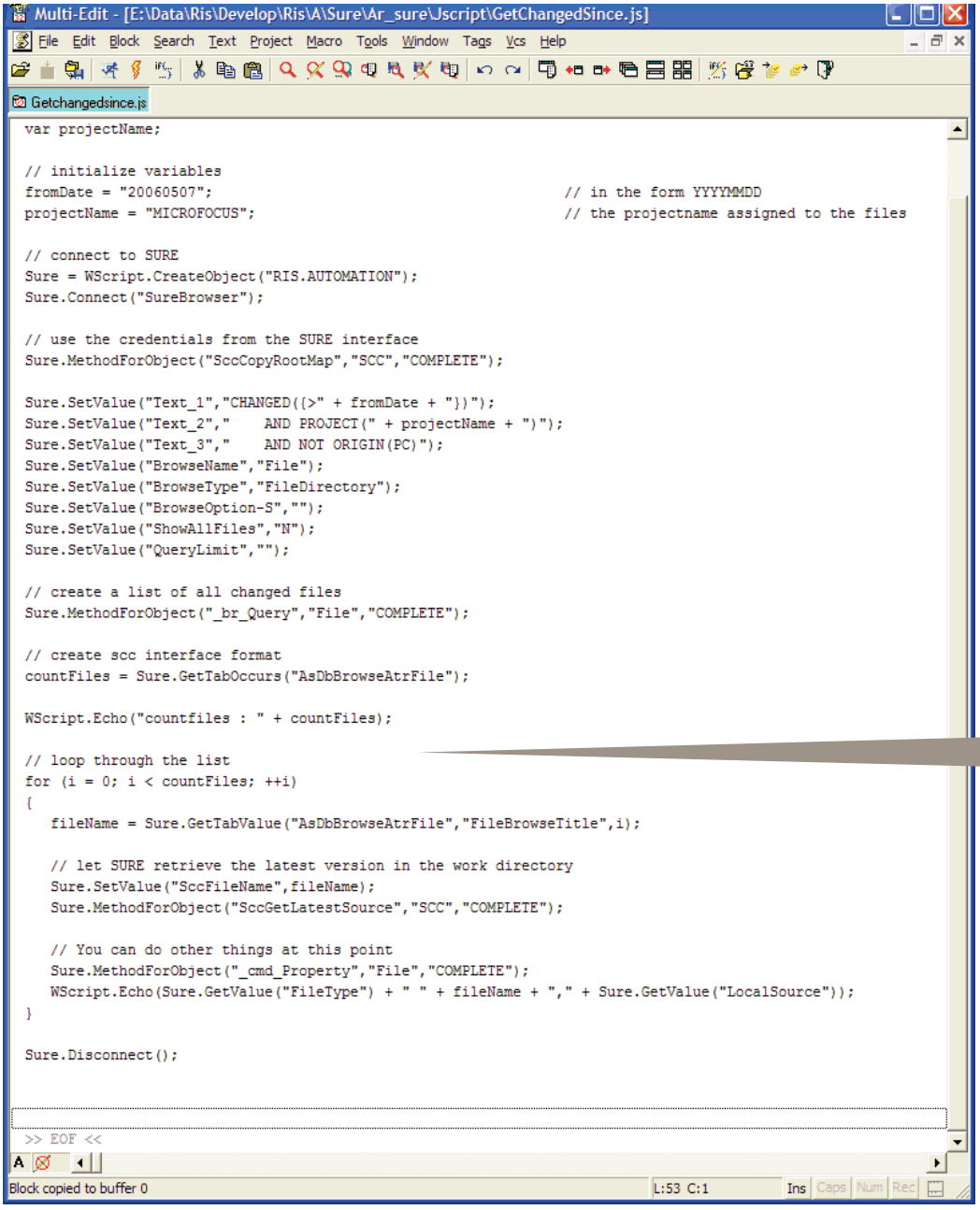
For Eclipse, SURE uses the CVS pServer protocol. Currently this interface is qualified for Eclipse version 3.1.2 and 3.2. Eclipse is the preferred IDE for many Java developers.



Eclipse links up to SURE using the CVS pServer protocol. This allows the developer access to the SURE functions from within the Eclipse software, thereby easing developer workload.

Relativity

SURE allows integration with Relativity's Modernization Workbench by using a VB script or JavaScript to export an object model. By invoking this script, SURE can publish changes that have occurred during development to the Relativity Workbench. The Relativity Modernization Workbench has sophisticated capabilities for discovery and transformation of applications where knowledge of the business rules and documentation of the application systems need to be updated.



SURE supports an OLE interface that allows the calling of all functions within the SURE interface. This small JavaScript retrieves all files that changed since the last run.

Transitioning to SURE

Most organizations have developed custom processes for the management of their development projects. Successful projects are defined within an environment that has evolved over many years, and there is often reluctance to disturb what has been working for so long. Any new framework represents business risks that must be assessed before undertaking a transformation. With SURE, the approach to this transformation mitigates most of the risks.

SURE is a flexible architecture for application development. SURE adapts the client's existing processes and provides complete automation for any manual procedures. Every SURE installation is completely customized to the unique requirements of each organization. For all but the very largest development organizations, the transition to SURE is measured in days and weeks, not months.

Conclusion

SURE provides a comprehensive approach to managing the modern software development process that can enable your organization to realize substantial gains in productivity and flexibility. Software assets are among a company's most valuable and should be aggressively managed to achieve optimum business results.

Most businesses have a substantial dependency on applications to support their business processes. IT budgets must deliver more value to the business, and the adoption of the SURE software lifecycle management solution will ensure that a competitive advantage is maintained as it reduces both cost and risk for businesses.

Putting these application assets under a single, comprehensive, enterprise-wide management control system will have an extremely positive impact on your enterprise. And for organizations with public exposure and auditing mandates, SURE provides the ability to meet and exceed compliance requirements. Those enterprises that have made this decision are improving their competitive postures already. Act now and start enjoying the benefits of SURE.

For more information, contact your Unisys representative.

In a hurry to learn more?
Visit: http://unisys.com/cp

For even more detail,
visit: http://unisys.com/cp/ecommunity

4/07

BL100031-100