Cradle-7
From concept to creation...

3SL

Cradle-SWE

The **Cradle-SWE** module provides reverse engineering and code generation to maintain the consistency between a detailed system design and its software implementation.

There are many contexts for software engineering, each using its own methods and languages. This module is intended for groups using functional methods to build software in C, Ada® or Pascal.

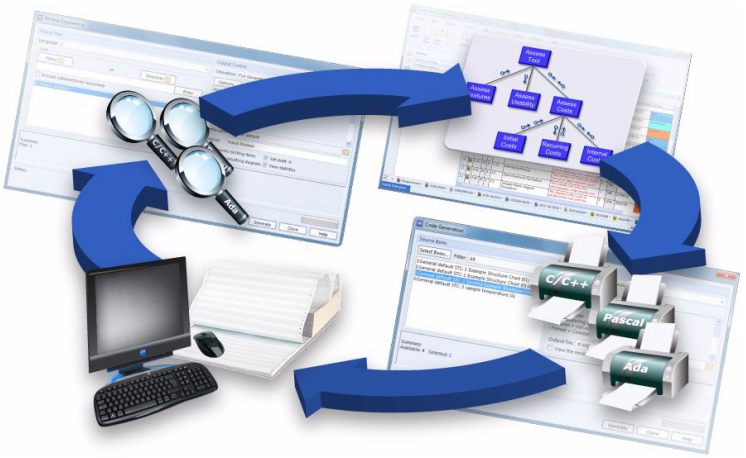
Detailed software designs are represented using Structure Charts (STCs) with 3SL extensions to support hierarchical descriptions of software into systems, programs, subsystems, modules and source files, the representation of functions, and the representation of basic data types.

Software designs are described with diagrams, data definitions and module specifications that hold the pseudo code, descriptions or source code. This software design is cross referenced to architecture, design and analysis models, to the requirements and test cases, and to all other data.

The initial design can be code generated to C, Ada and Pascal type definition header files (built from the model's data definitions), and source files that contain the call hierarchy from the STCs, the call arguments and local variable declarations, and the content of the STCs' module specifications' pseudo code or detailed design material.

Once algorithmic content is added to the generated code, the resulting completed source files can be reverse engineered back into the Cradle database, to update its design diagrams, and both the data definitions and module specifications of these diagrams' symbols.

Reverse engineering merges actual source code into the design definitions and specifications, subdividing each source file into the individual routines and storing their component parts into separate frames in the module specification and data definition items in the database. Every line in each source file is stored in a frame of one of these items in the database.



The Code Generator can be run on the results of reverse engineering to reconstruct the source files, either as they were, or including any changes made in the design model. In this case, the source code is generated using call interfaces built from the (possibly modified) STCs and the routines' bodies are created from either the STC call hierarchy or the source code from the previous reverse engineer operation.

The process can start by reverse engineering legacy source code into an initial design model, recovering designs in situations where only the implementation currently exists.

As reverse engineering loads all source files into the database, the source files could be deleted, and instead configuration managed through the Cradle Configuration Management System as part of the design.

The diagrams, specifications and data definitions can contain any number of attributes, including URLs to reference the source code in a source code control system, such as Git or Subversion.

The format of all generated

SWE
Software Engineering

http://www.threesl.com
salesdetails@threesl.com
support@threesl.com

Structured Software Systems Ltd (3SL)
Suite 2, 22a Duke Street
Barrow-in-Furness
Cumbria LA14 1HH, UK
Tel: +44 (0) 1229 838867
Fax: +44 (0) 1229 870096
Regd: 2153654 VAT: GB 473 2757 28

code can be tailored to match your coding standards. Data definitions can be marked to be standard data types and generated into the source code. Header files can be produced from the composition specifications inside the data definitions to create record and variant structure declarations.

The reverse engineering tool supports any compiler pragmas and conditional compilation directives. It can distinguish application code, application libraries and standard library or operating system / runtime routines, and render the design diagrams accordingly. This uses any combination of regular expressions, and Cradle-supplied or user-defined library routine lists.

Reverse engineering can process one or more source files in one run, creating a hierarchy of design diagrams to represent the code structure beyond individual source files.

Using the reverse engineering tool creates full traceability across the entire system lifecycle, from user needs to system requirements to analysis, architecture and design models, to test procedures, specifications and test cases, to the source code. Cradle's transitive cross reference view facilities allow users to directly see the user requirements and acceptance criteria associated with each source code module, and vice versa.

Feature Summary

Feature	Benefits
Integrated with design model	Provides the link between the design database and the system implementation
Code generation, reconstruction, and reverse engineering tools	Bidirectional exchange of source code with the code development system and synchronization of the detailed design with the software implementation
Customizable code reconstruction	Reorganize source files from original layouts and optionally include design database changes in the generated source code
Absorbs all source code into design database	Cradle database can manage approved software releases in formal baselines, while the code development system is used for interim work
Batch processing	High productivity by code generating or reconstructing any number of design diagrams as one operation, and by reverse engineering entire directory trees as one operation
Cross referenced software implementation	Complete traceability from stakeholder needs to source code, via analysis and/or design models, and full linkage to all test specifications, test results, and all other project data
Support for C, Ada, and Pascal	Applicability to a wide range of new and legacy projects
Customizable code generation	Conform to local coding standards
Choice of reverse engineering operations	Create design diagrams and/or load definitions and/or analyze code characteristics, or all of these
Automatically creates code design diagrams	Recognizes your code, your libraries and third party libraries so design diagrams correctly identify all routines and your application is not obscured by secondary code
Customizable reverse engineering	Optionally suppress system or library routines, control the layout of generated diagrams and options for populating the database with module and data definitions
Supports compiler pragmas and conditional compilation directives	Reverse engineer all code or only the code for a specific combination of compilation options. Uses the same syntax for conditional compilation as the compiler.

Structured Software Systems Ltd (3SL)
Suite 2, 22a Duke Street
Barrow-in-Furness
Cumbria LA14 1HH, UK
Tel: +44 (0) 1229 838867
Fax: +44 (0) 1229 870096
Regd: 2153654 VAT: GB 473 2757 28

RC015/17 Date: June 2019

http://www.threesl.com
salesdetails@threesl.com
support@threesl.com