



INDEPENDENT SOFTWARE VERIFICATION & VALIDATION

<http://www.isvv.com>



**VERIFICATION** *Are we building the product right?*

**VALIDATION** *Are we building the right product?*

### Independent Software Verification and Validation (ISVV)

is targeted at safety-critical software systems and aims to increase the quality of software products, thereby reducing risks and costs through the operational life of the software. ISVV provides assurance that software performs to the specified level of confidence and within its designed parameters and defined requirements.

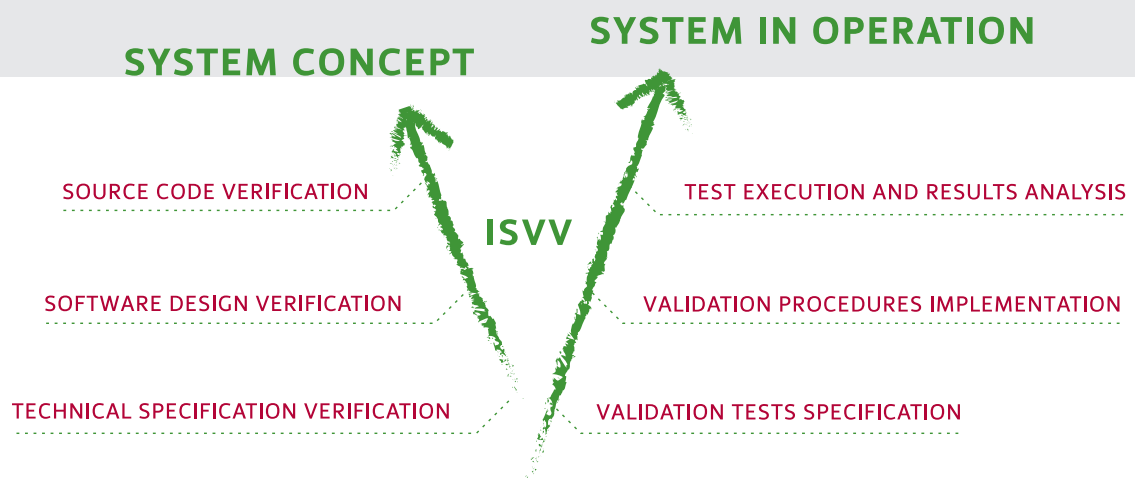
**ISVV** activities are performed by independent engineering teams, not involved in the software development process, to assess the processes and the resulting products. The ISVV team independency is performed at three different levels: financial, management and technical.

**ISVV** goes far beyond “traditional” verification and validation techniques, applied by development teams. While the latter aim to ensure that the software performs well against the nominal requirements, ISVV is focused on non-functional requirements such as robustness and reliability, and on conditions that can lead the software to fail. ISVV results and findings are fed back to the development teams for correction and improvement.

**VERIFICATION** consists of ensuring that the software product is compliant with the requirements, through all phases of the life-cycle. This is accomplished by analysis, inspections, formal evaluations of intermediate and final software items. The software items analysed are selected according to a previously performed criticality analysis, therefore increasing the activity “value for money”.

**VALIDATION** consists of demonstrating that the software accomplishes its intended purpose. This is achieved by testing the product in real or simulated environments. The intent of these validation activities is to assess error handling behaviours, safety and security issues, and areas where a failure might cause undesirable effects.

**THE END RESULTS** of applying ISVV to a system are diverse. While increasing the software dependability, the recommendations provided allow the simplification of software architectures, resulting in increased maintainability and usability. Higher reuse capability is also achieved, which facilitates future developments and provides higher independence from software providers and better knowledge and operation of the system.



*ISVV introduces a small added cost  
and brings a high added value*

“ This research provided us with a new and exciting capability for assuring software in a way previously thought to be impractical ”

Marcus Fisher, Head of Research at the NASA IV&V Facility

Independent Software Verification and Validation suitable for



AEROSPACE



DEFENCE



FINANCE



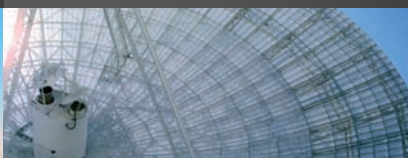
GOVERNMENT



MANUFACTURING



ENERGY



TELECOM

## STANDARDS

- › ECSS-E-40B
- › ECSS-Q-80A
- › RTCA/DO-178B
- › EN-50128
- › ISO15504 (SPICE)
- › S4S
- › DEF-STAN 00-55
- › DEF-STAN 00-56
- › MIL-STD-498
- › IEEE/EIA-12207

## RAMS TECHNIQUES

- › Schedulability Analysis
- › Hardware Software Interaction Analysis
- › Code Inspections
- › Robustness Testing
- › Stress Testing and Fault Injection
- › Software Failure Modes, Effects and Criticality Analysis (SFMECA)
- › Software Fault Tree Analysis (SFTA)
- › Hazard Analysis (HA)

## ABOUT CRITICAL SOFTWARE

Critical Software is an international company that provides solutions, services and technologies for mission and business critical information systems. The company supports customers across several markets including Aerospace, Defence & Homeland Security, Manufacturing, Telecom & Media, Government, Finance & Insurance and Energy & Utilities. Critical Software operates a quality management system certified to CMMI® Level3, ISO 9001:2000 Tick-IT, ISO 15504, NATO AQAP 150, AQAP 2120 and EN9100

<http://www.criticalsoftware.com/>

## ISVV METHODOLOGY

AN ISVV PROGRAMME IS COMPOSED BY FIVE MAIN PHASES THAT CAN EITHER BE EXECUTED SEQUENTIALLY OR AS ISOLATED PROCESSES:

### ISVV PLANNING

- › Planning of ISVV Activities
- › System Criticality Analysis: Identification of Critical Components through a set of RAMS activities (Value for Money)
- › Selection of the appropriate Methods and Tools

### REQUIREMENTS VERIFICATION

- › Traceability between Software and System requirements
- › Verification for: Completeness, Correctness, Testability

### DESIGN VERIFICATION

- › Design adequacy and conformance to Software Requirements and Interfaces
- › Internal and External Consistency
- › Verification of Feasibility and Maintenance

### CODE VERIFICATION

- › Traceability between Design and Code phase
- › Verification for: Completeness, Correctness, Consistency
- › Code Metrics Analysis
- › Coding Standards Compliance Verification

### VALIDATION

- › Identification of unstable components/ functionalities
- › Validation focused on Error-Handling: complementary (not concurrent!) validation regarding the one performed by the Development team (More for the Money, More for the Time)
- › Compliance with Software and System Requirements
- › Black and White Box testing techniques
- › Experience based techniques

ALL FINDINGS GENERATE REVIEW ITEM DISCREPANCIES (RIDS) WITH A COMPLETE PROBLEM DESCRIPTION AND THE CORRESPONDENT RECOMMENDATION