

# A NOVEL APPROACH TO DIAGNOSING PROBLEMS IN SCA WAVEFORMS DURING DEVELOPMENT AND PORTING

Larry Dunst (DataSoft, Scottsdale, AZ, [larry.dunst@datasoft.com](mailto:larry.dunst@datasoft.com))

Shahzad Aslam-Mir, Ph.D. (DataSoft, Scottsdale, AZ, [shahzad.aslam-mir@datasoft.com](mailto:shahzad.aslam-mir@datasoft.com))

Brandon Duthler (DataSoft, Scottsdale, AZ, [brandon.duthler@datasoft.com](mailto:brandon.duthler@datasoft.com))

Eric Miles (DataSoft, Scottsdale, AZ, [eric.miles@datasoft.com](mailto:eric.miles@datasoft.com))

Aaron Goff (DataSoft, Scottsdale, AZ, [aaron.goff@datasoft.com](mailto:aaron.goff@datasoft.com))

Marc Lerch (DataSoft, Scottsdale, AZ, [marc.lerch@datasoft.com](mailto:marc.lerch@datasoft.com))

## ABSTRACT

Much time is spent in diagnosing and isolating waveform resource component problems when they are ported to different platforms. Specifically, integration and testing of temporal correctness and proper transformation of waveform data through components on a target versus what is expected from a simulation model can potentially be troublesome. We present a set of approaches that minimize the time and effort required for porting an SCA or non-SCA waveform to another platform. We also present new unique test tools for SDR development called the Software Defined Radio Tool Suite (STS) that seamlessly provide probes into an SDR and adapter conduits into industry standard tools like MATLAB-Simulink and LabVIEW enabling users to perform diagnostics on a running waveform on real radio targets.

## 1. PROBLEM STATEMENT

Early assessment of requirements prior to design, coupled with a standards based SDR implementation, is an invaluable approach to devising current and future generations of communications technologies. However, present day experiences in assessing embedded, real-time architecture implementation in the areas of porting JTRS assets to new SDRs using SCA with strict compliance to SCA, CORBA and POSIX standards has not yet shown itself to be completely satisfactorily cost effective.

Practitioners in the JTRS SCA communities recognizing this have looked at taking several approaches to try to capture and quantify the extent of the challenges and to try to mitigate the risks involved. As the requirements for compliant SCA standards-based, interoperable, networked JTRS radio-sets increases, the need for short turnaround on waveform ports to various classes and form factors of radios will only become more acute. In addition, the costs associated with the laborious time consuming activities of ensuring interface standards compliance and adequate overall radio system performance in the ported

configurations will increase disproportionately unless new analysis and SCA-SDR targeted diagnostic tool-suites are provided. The cost of ensuring a software system meets the standards and is sufficiently fast and deterministic, from the temporal signal processing perspective, eats into the savings sought by implementing the standard on the device in the first place.

Current generation standards compliance requires many hours of laborious code review and/or lab testing. The many challenges can be (a) Human element – error prone (b) time consuming – leading to slipped schedule and increased cost (c) insufficient skill sets for CORBA, C++, SCA debugging and analysis (d) disproportionate consumption of resources (prototypes, people) thus hindering other development work.

These problems increase cost and slip schedule in every software release. A quasi-automated tool-suite would dramatically reduce the first time compliance test costs of a Core Framework (CF) and waveform (WF) ported solution set and would significantly go towards reducing the test costs for each software release thereafter.

Current generation SCA validation tools such as JTEL's JTRS Test Application (JTAP) and prior Waveform Test Tool (WTT/WFT), though accurately measuring SCA compliance, cannot provide a measure of the degree of complexity, time, and cost to port a compliant WF to another platform. Two independent estimates can be wildly different and albeit inaccurate as to the true nature and complexity of the task.

The JTAP tool used currently to verify and certify SCA compliant implementations of CFs and WFs is a valuable test suite used to gauge degrees of compliance and, more importantly, portability of WFs from one platform to another. JTAP produces useful pass/fail results, but does not provide any guidance or figure of merit that can convey to the porter how difficult it will be to port a WF onto a particular CF and radio set, nor does it measure the extent and detail of additional extensions in the CF and the WFs use of any of those extensions which can cause difficulty in porting and failure of the port certification on another

platform. Successive updates have improved the JTAP tool, but it still falls short of providing measures that can predict ahead of time the difficulty of porting existing waveforms or synthesizing new ones on a platform. Thus, there exists a commercial space for such tools. Our approach does not reinvent the wheel; it reuses and builds upon existing rigorous certification tools and processes that form the backbone of SCA-compliant JTRS solutions such as the JTAP and legacy WTT/WFT tools.

## 2. PROPOSED SOLUTION

The purpose of this paper is to show that it is possible to synthesize the processes to accurately estimate software complexity, porting metrics, and provide detailed real-time online standards compliant SCA-SDR diagnostic and event data to reduce overall SCA-SDR system integration costs to produce a networked JTRS – an SCA SDR based GIG.

Our analytics are applied to and build upon the workings and the results of existing industry standard tools. First we present the DataSoft Waveform Analysis Tool (DSWFAT) that measures a waveforms complexity and porting risk. Second is the DataSoft Software Probe (DSSP) technology that provides real-time debugging and incremental test, integration, and porting capabilities. Both tools are shown at a high level in Figure 1.

### 2.1 DataSoft Waveform Analysis Tool (DSWFAT)

We researched creating an initial pool of measured metrics from the source code using existing tools upon which we then operated. We tried a number of approaches for quantifying complexity metrics in addition to traditional approaches such as McCabe [1]. We devised a hybrid approach that works well. In this approach we used the work of the team at NEDT&E, who have compiled the waveform portability guidelines document [2]. In this manner we created a highly SCA and WF focused analysis tool that measures both the complexity of the waveform elements as well as their portability.

We took an approach owing to the complexity introduced by CORBA and SCA APIs in C++ that have added complexities of memory management and rules that must be followed for object instantiation. In addition, the use of custom transports with different semantics under CORBA ORBs in SCA radios gave rise to the concern that perhaps we needed to modify and adapt the McCabe model for this added complexity.

We simulated a number of approaches for quantifying software metrics. Principally, we analyzed the techniques of McCabe, Halsted, and others. We also looked at approaches to collect these metrics in an automated fashion from source code. Using industry tools we were able to get first order estimates of several standard software metrics which we then use to estimate complexity and porting risk. In order to do this latter part we augmented the industry tools with our own scripts to collate further data and prepare data for our portability risk calculations.

We employed parsers that used simple grammars for C/C++ and we measured complexity using higher level constructs. We then augmented this with the rules we had codified into a set of scripts and we constructed a data matrix of measures that we have garnered from the waveform source code.

We then invoked the DataSoft pre and post processing scripts to collect various complexity metrics and compared the data with the module on which it is reported for the purposes of a sanity check. As an example, recognizing large amounts of fan-in in a waveform's assembly controller is a simple illustration of such a check. Therefore our approach mines the source for metrics and our scripts further refine them. Based upon this approach we were able to derive a set of rules from which we based further analysis. We thus formed a set of waveform portability rules that we could then utilize to form a report on WF portability and complexity.

We then classify and record non-SCA tokens, lexemes and aspects for vendor specifics, ORB, ORB vendor specifics, OS, OS vendor specifics, and POSIX specific tokens and lexemes. We then used the expression aspects or clusters to formulate a database of expressions. Next we applied software metric estimation techniques that are well documented in literature [3] [4] and produced figures of merit from them for each cluster or aspect. We then experimented with ways to combine these measures to produce simpler single figures of merit that reflect WF porting risk, effort estimates, and complexity of porting a WF to another platform. These figures were combined together in different manners in our experiments and reviewed.

Figure 2 illustrates our unified high level approach to WF analysis for SCA compliance as well as portability. Note however here that we also combined the results of JTAP testing to provide metrics on a more complete picture.

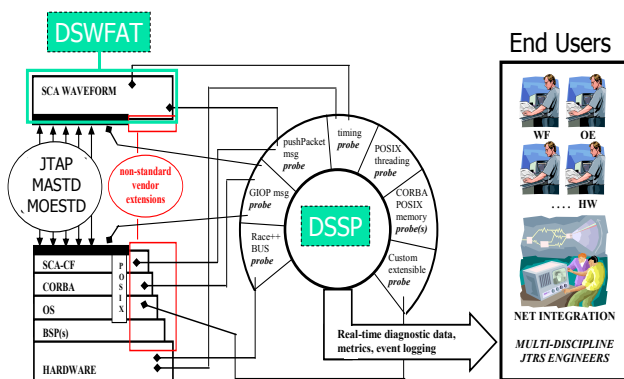


Figure 1: The STS Methodology

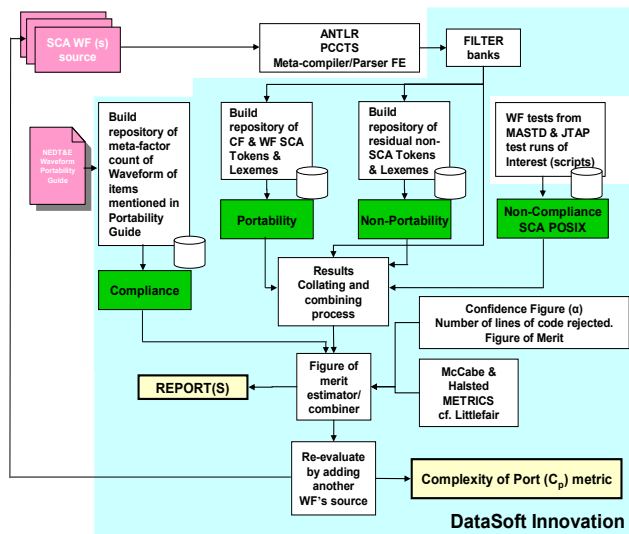


Figure 2: Unified Approach to WF Porting Profiler

Figure 3 illustrates a portion of our database of metrics and considerations that are reviewed when parsing and analyzing waveform source code. We show here an illustration of a test run over FM3TR and the reported figures that show the measured metrics.

Also in the report are the various details of links to the results of parsing the source for object oriented, structural and procedural metrics. We then combined these metrics with a weighting scheme that uses a Knapsack algorithm like approach to linearly combine these figures of merit to produce single numbers that represent porting effort

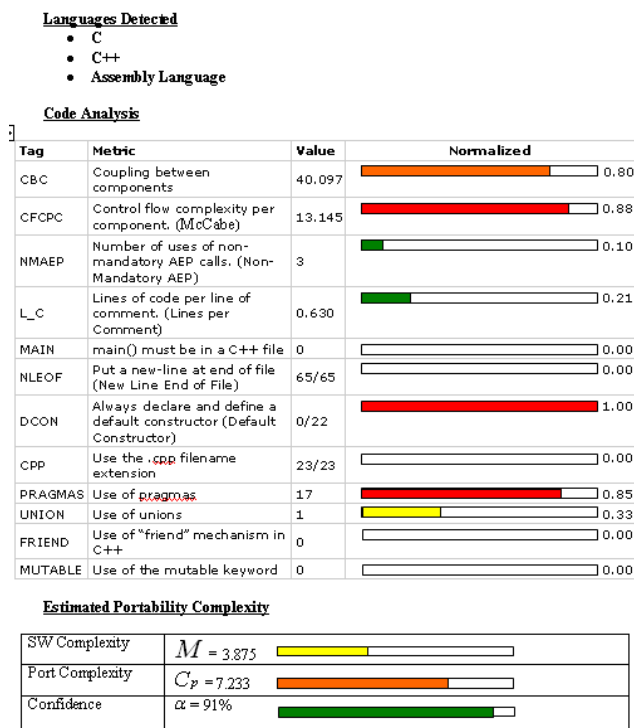


Figure 3: Example DSWFAT Level 1 Summary Report

estimates. But before we do that we reframed this data based on the findings of the NEDT&E document.

DSWFAT reports on non-compliance with respect to SCA where we show areas of waveform source code that are using non mandatory SCA AEP calls that reduce portability of the waveform owing to the fact that those APIs may not be available on the next platform. We provide a tool to do interface-API matching between *from* and *to* platforms to which the waveform will be ported later in the study.

Additionally, we worked with Zeligsoft to integrate our WF Analysis engine technology into their Zeligsoft CX tool (a.k.a. Spectra CX) to provide a waveform porter with an easy to use IDE to manage all aspects of porting a waveform. Once integrated with Zeligsoft CX, our tool first provides the ability to import the XML from an existing SCA compliant WF and generate a model of the application. Using the source code analysis techniques described above, our tool can generate and display WF portability metrics from the application level down to the component level. This provides a unified development environment that simplifies the analysis and porting process.

## 2.2 DataSoft Software Probe (DSSP)

We adopted a two-pronged approach for providing real-time diagnostic and logging information during the waveform porting and development process:

- A **probe** that can be inserted into SCA waveforms and core frameworks. This can be done *without having to build in any source code via a compile cycle*. The probe is self-attaching and/or a listener for WF SCA traffic.
- An **adapter** that is connectable to the probe(s) that enables a porting engineer to *visualize* the waveform component's behavior in one of several *industry standard tools* of the engineer's choosing.

We believe that this unburdens the porting engineer from having to learn a new tool. The tool allows them to choose the probe and connect it to the right adapter and immediately be able to study the real-time data flows in any connected waveform with complete ease. Our adapter technology allows visualization in MATLAB-Simulink and LabVIEW to name but a few of the industry tools. In addition, when data is of a particularly low latency where data is taken from FPGA interfaces, we plan to provide a signal conditioning board that can be inserted upstream of the visualization tools to provide any decimation or other transformation before bringing it into signal processing environments. In our approach, which is illustrated in Figure 4, any tool adapter can be hooked up to any probe on any waveform.

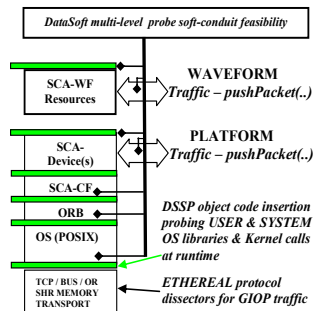


Figure 4: DataSoft multi-level probe

We then researched novel techniques to augment our waveform traffic snooping mechanisms that in the first instance uses Wireshark and some new non-IP techniques. We worked with several industry partners to provide a real-time report of simulated traffic among the waveform blocks for test messages that the user can design. In addition, our approach supports importing models that can be simulated ahead of any porting exercise. This way a hardware-in-the-loop simulation of parts of the waveform can be performed giving the porter vital knowledge of the waveform components and their traffic before it's ever implemented on the real target.

In addition we created a new approach to instrumenting an SCA waveform's components in a minimally intrusive manner by not needing to build in anything into the build waveform components. This gives the designer knowledge of how the waveform works before a port is commenced and even during part of the porting cycle. This approach is illustrated in Figure 5.

We do this by providing a simple daemon that can be launched on the radio of choice which then sniffs SCA traffic across all interfaces and reports it to our Beagle Board. In addition the tool allows for snooping MHAL messages between the GPP, DSP and FPGA using this technique. Our approach was further refined to trace the number of threads, memory and the execution of some methods that are user identified as data of interest. We also chose to do this work using a signal conditioning board that we provide as a workhorse to do MHAL signal filtering and processing before reporting it to the waveform user in any tool. For the signal conditioning board we chose the Beagle Board due to its inherent low cost and sophisticated GPP-DSP combination.

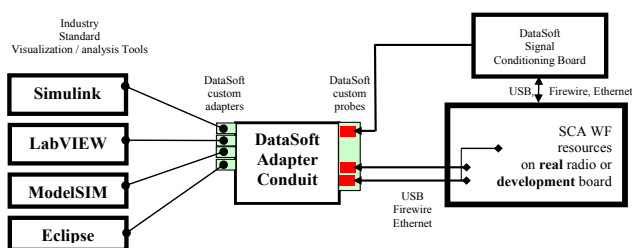


Figure 5: Real-time minimally intrusive probe approach

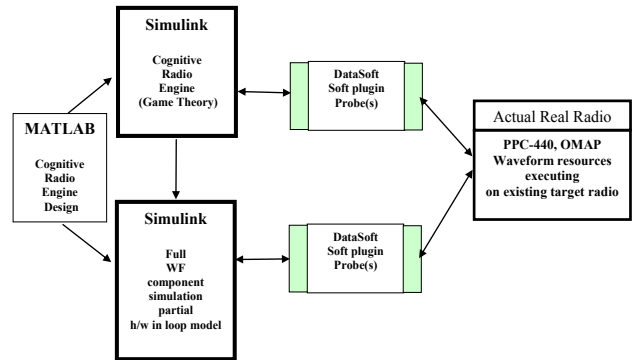


Figure 6: DSSP technology for Cognitive Radio Synthesis

DSSP can also be used for research in the cognitive radio arena; this shows great promise for soft-radio devices of the future. With this in mind, and given that there are several schools of thought currently on how to realize or synthesize a cognitive soft-radio, we chose to select a few choice areas where we believe DSSP can be used. DSSP helps to probe existing SCA and non SCA waveforms by providing data, control and timing information that can be used to make cognition decisions and alter the radio's behaviors. With this in mind we experimented with integrating with the MATLAB-Simulink simulation engine with a view to integrating partially ported SCA waveform components with the remainder of the waveform in Simulink. This integration then allows the addition of cognition engines to the waveform in MATLAB-Simulink and directly impact waveform components. To look at how this would work in more detail we used [5] [6]. In this case we model elements of a real waveform in Simulink and add to it a cognitive element in the guise of a Game Theory predictive model for protocol policy adaptation. We then run a target radio in which there are elements of a real waveform coupled with simulated cognitive waveform elements run out of Simulink, governed by the Game Theory model. We use the DataSoft probes to glue together partial implementations of a real waveform governed and controlled by a partial simulation of cognitive waveform elements to study Cognitive Radio as shown in Figure 6.

In Figure 7, we show a mockup using our own DataSoft GUI to create and use a Cognitive Radio Simulation. Using DSSP, data can be retrieved from the Target system where portions of a waveform may be running in a realistic

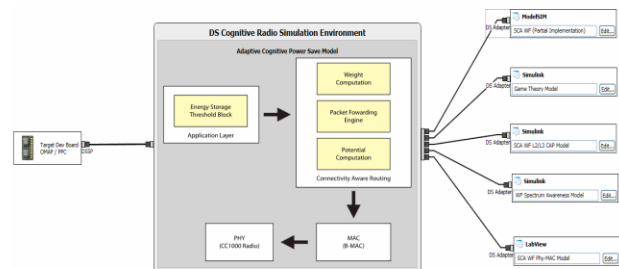
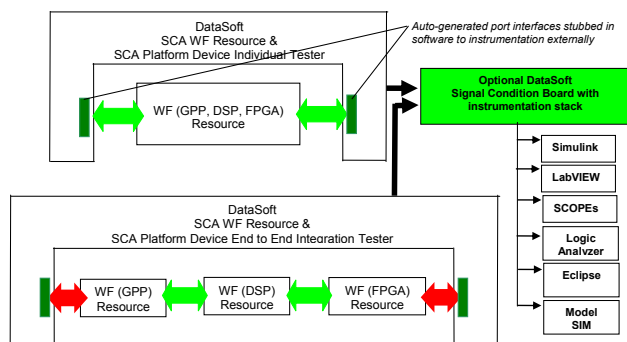


Figure 7: GUI and DSSP in Cognitive Radio Environment





**Figure 8: SDR Diagnostic Waveform Component Tester Tool**

environment, and data from them sent to various environments, such as Simulink, LabVIEW and ModelSIM, where cognition experiments have been built to work with it. This mechanism allows a developer to rapidly prototype new cognitive radio techniques and provides the ability to test them in a real run-time environment using a physical target.

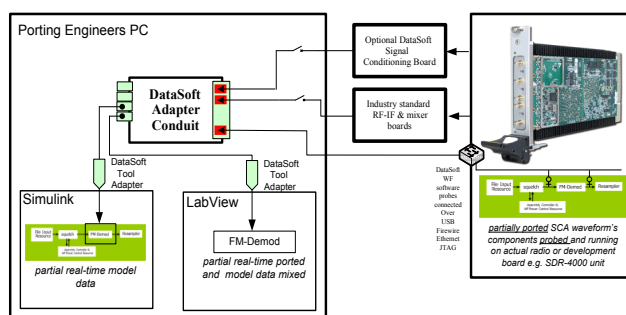
An example may be in testing out some theory with a practical environment for example taking an existing SCA waveform (running on target environment) and adding to it a Game Theory model (in MATLAB) that has been developed for predictive protocol policy adaptation and so has a partial implementation of the protocol also simulated – this being done in LabVIEW. Our tools allow the designer to take pieces of a waveform from existing real environments and add to it simulated pieces for cognition from other environments to build an experimental test bed to test out various theories of CR as well as to gather data to build more complex experiments.

### 3. SIMULATIONS

Figure 8 shows how the DSSP diagnostic waveform component tester tool interfaces with the Beagle Board signal conditioning board for tracing, combining all tools into one centralized delivery platform. The tester acts as a pseudo core framework and brings up a ported SCA resource and connects its ports to dummy ports through which test vectors are injected and outputs compared for analysis and checking that the porting is giving the desired result. Note that this tester also helps with WF integration, and potentially requirements fulfillment.

Figure 9 shows how the our approach permits the waveform porting engineer, if provided models of the waveform, to check individual components of a waveform for test vector response or boundary condition checking.

This is possible because the DataSoft probes on the actual radio interface to the adapter conduit on the engineer's PC and then can be hooked to the right diagnosing tool such as Simulink, LabVIEW, Agilent scopes, or Synplicity, TI, Xilinx plug-ins that run in Simulink and can integrate the waveform data.



**Figure 9: SDR Diagnostic Waveform Component Tester Tool**

Our approach is novel and unique in that it:

- (1) Allows the WF porting engineer to check ported and built components on the target quickly and individually without having to generate tedious XML on a new CF and new radio hardware. This is due to the DataSoft test harness allowing them to launch the single resource in its own little SCA microcosm quickly to test it.
- (2) Allows the WF porter to piece meal integrate and test incrementally, allowing for greater confidence of port quality.
- (3) Permits a probe to be non-intrusive at the SCA level so minimizing the porting engineer to have to learn new tools – minimizing time and cost to port.
- (4) Is totally visual allowing the porting engineer to see what they are connecting to what via MATLAB, Simulink, LabVIEW thus aiding understanding.
- (5) Allows real-time on the radio SCA WF resources to feed data into simulated elements or vice versa to test integration – thus leading to diagnostic debugging in SCA at a new level.
- (6) Optionally uses a custom signal conditioning board that allows even the lowest latency probing to be possible, for example in FPGAs, by providing VHDL and adaptive decimation on the Beagle platform.
- (7) Allows industry standard RF-IF and other boards to be accommodated using the right adapter and probe combination, thus the solution is standards compliant (MHAL, CORBA, SCA, and others) open, non-proprietary and extensible to other industry sectors also if needed. Overall it saves investment and total cost of long term ownership of the approach.

Figure 10 shows the Simulink integration in which we read in I/Q samples from a file. Using Simulink's S-Function capabilities, we linked in CORBA code to make pushPacket() calls directly on the FM-Demod component running on the other machine. We then finished the rest of the processing chain, primarily resampling, right in Simulink and were able to play the processed sound on the Simulink machine. Using this technique, it is easy to see the benefit of

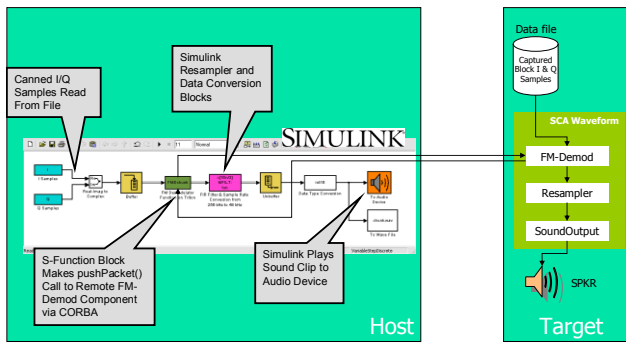


Figure 10: Simulink Simulation Setup

being able to remotely probe components running an external machine without having to make any modifications to the remote system.

A similar approach was followed for the LabVIEW simulation as shown in Figure 11. In this case, LabVIEW was able to make direct DLL calls to make the CORBA pushPacket(..) calls to push data to the remotely running FM-Demod component. LabVIEW blocks were used to perform the resampling and to play the data in the sound out.

#### 4. CONCLUSION

The DataSoft STS approach provides dual benefit in the process of porting an SCA waveform to new JTRS radio platforms, as illustrated in Figure 12.

First, DSWFAT analyzes WF source uniquely in a manner to characterize SCA port complexity. It does this by applying accepted industry software complexity metrics, for example McCabe Halsted, but with the constraints and in the contextual framework of accumulated SCA domain knowledge as embodied in the NEDT&E Waveform Portability Guidelines to produce a workable methodology. This helps to produce a near optimal solution in the JTRS SCA sense.

Second, DSSP arms the waveform porting engineer with a series of real-time diagnostic debug probe tools to study SCA waveform component resources at macro and microscopic levels. It enables a divide and conquer approach to waveform porting. Waveform elements can be

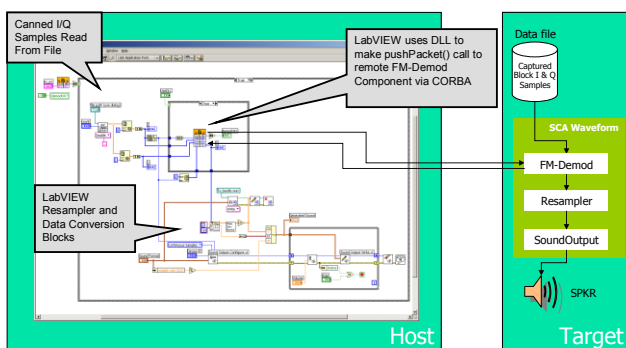


Figure 11: LabVIEW Simulation Setup

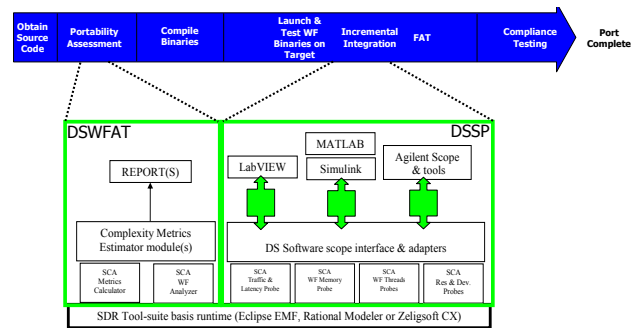


Figure 12: DataSoft SDR Tool-Suite Use in WF Lifecycle

incrementally ported and integrated on the target in a traceable and efficient manner yielding a standard process to porting waveforms. It supports an engineer in the synthesis of good test vectors using industry standard tools and then seamlessly injects them into WF components on real target platforms without the need for SCA CF and XML infrastructure. Using these, engineers can feed and extract complex test data to varying degrees and at various times in the waveform integration process. Finally it allows for an open framework to integrate an SCA WF component with standard industry signal processing workbench tools, such as MATLAB-Simulink and LabVIEW, which enable the engineer to check fidelity of I/O and boundary conditions sooner in the integration process, yielding greater productivity in the long term.

#### 5. ACKNOWLEDGEMENTS

This research was supported through Army contract W15P7T-09-C-5451. The opinions presented are those of the authors and do not necessarily reflect the views of the sponsors. The authors would also like to thank Zeligsoft for their support.

#### 6. REFERENCES

- [1] T. McCabe, "A Complexity Measure," IEEE, 1976.
- [2] L. Anderson, "JTRS, Network Enterprise Domain, Test & Evaluation, Waveform Portability Guidelines," 28 May 2008.
- [3] S. Henry and D. Kafura, "The evaluation of software systems structure using quantitative software metrics," Software Practice and Experience, 14(6), 561-573, 1984.
- [4] E.J. Weyuker, "Evaluating software complexity metrics," IEEE, 14(9), September, 1357-1365, 1988.
- [5] M.B. Pursley and T.C. Royster, Protocols for Adaptation in Cognitive Radio, Cognitive Radio Technology, Academic Press, 2009.
- [6] J. Neel, J.H. Reed, and A.B. MacKenzie, Cognitive Radio Performance Analysis, Special focus is the use of Game Theory, Cognitive Radio Technology, Academic Press, 2009.