Copy Number .....

UK PDR Phase 1 Final Report

P6957-11-001 Issue 1

Written by

.....
C Waugh (RRL)

Reviewed and Approved by
Group Leader

.....
B Hillam (RRL)

Authorised by
Project Director

Date .....

.....
C P Ash (RRL)

DERA Portsdown
Portsdown Hill Road
Fareham
Hants
PO17 6AD

Configuration Management	
CM Tool/Version	Visual SourceSafe v5.0
Library Location	Q:\P6957\Vss\…
Library Project	$/Library/11 Technical Reports/001 Final Report
Baseline	11_001_Issue 1
Element	sst9415.doc.doc

Modification History		
Issue	**Date**	**Description**
Draft A	31st March 2000	First draft issued for comments
Draft B	3rd April 2000	Updated with comments from E Willink & B Hillam
Draft C	4th April 2000	Updated with comments from G Dent and revised WDL concept diagram
Draft D	5th April 2000	Minor edits on Introduction
Issue 1	11th April 2000	Report Issued without M Harrington Section

Executive Summary

This report acts as an executive summary covering the work conducted during the UK Programmable Digital Radio (PDR) Phase 1, Waveform Description Language (WDL) programme (DERA contract CU009-0000002745). The PDR programme aimed to identify a methodology for encapsulating waveform descriptions, investigating software tools to aid encapsulating of a waveform and to conduct initial parameterisation of FM3TR and SATURN. A transition plan for the phase 2 programme was also required. The current work conducted on phase 1 has met the prime objectives of the programme and gone a significant way towards defining the constructs of a language in detail.

The use of a waveform description language will provide significant cost and time savings to MoD as it has the potential to reduce the development time for new waveforms and also for porting waveforms between different hardware platforms. The currently defined WDL aims to go beyond simply a tool to capture a waveform specification, but to provide the means of allowing a captured specification to be refined. This will allow Implementers to replace the specification leaf objects with their own implementations, which can then be used in an automatic code generation process.

The block diagram representation of the WDL concept is shown in Figure 3.1. Exploitation of WDL consists of two distinct aspects:

- A WDL specification, which utilises WDL to parameterise a waveform (performed by the waveform Sponsor) and allows the waveform parameterisation to be refined by the Sponsor or Implementer.
- A WDL compiler, which takes the Implementer refined parameterisation and creates a waveform bundle for download to the radio.

An example of usage to parameterise the FM3TR test waveform is shown in Section 3.2. The language has been developed to be hardware independent but use of it on JTRS hardware has been reviewed during the programme and no issues have yet been identified with the use of WDL for JTRS. In fact, WDL is complementary as it can specifiy the implementation for JTRS radios (or any other software radio platform).

A transition plan is shown in Section 4.4, which identifies activities to be conducted in phase 2 to further develop the language. Implementation of waveforms is proposed during phase 2 to validate the concepts.

A number of risks have been identified during phase 1. These are detailed in [11]. A 3 month de-risking activity is proposed at the outset of phase 2 to mitigate the risk associated with the WDL Compiler development.

Contents

Figures

Glossary

ADC	Analogue to Digital Converter
API	Application Program Interface
CESG	Communications-Electronics Security Group
COTS	Commercial Off The Shelf
Cd	Carrier detect
CM	Configuration Management
Dlc	FM3TR Data Link Control
DERA	Defence Evaluation and Research Agency
DSP	Digital Signal Processor
FIR	Finite Impulse Response
FM3TR	Frequency hopping test waveform
FSM	Finite State Machine
HCI	FM3TR Configuration and Operation Parameter layer
HTML	HyperText Markup Language
ISR	Interrupt Service Routine
Mac	FM3TR Medium Access Control Layer
MoD	Ministry of Defence
Nwk	FM3TR Network layer
PCSMA	Persistent Carrier Sense Multiple Access
PDR	Programmable Digital Radio
PHL	FM3TR Physical Layer
PIM	Programmable INFOSEC Module
RRL	Racal Research Ltd
Rx	Receiver
SATURN	NATO frequency hopping waveform
Tx	Transmitter
UCB	University of California Berkeley
WDL	Waveform Description Language
WP	Work Package
JTRS	Joint Tactical Radio System
SCA	JTRS Software Communications Architecture
SPW	Signal Processing Workstation
STANAG	STANdardisation AGreement
XML	eXtensible Mark up Language

References

[1] Q5246-001, UK Programmable Digital Radio Phase 1 Waveform Description Language, Volume 1, Technical Proposal, (Racal Raytheon UK PDR Consortium, 08 September 99)

[2] Q5246-002, UK Programmable Digital Radio Phase 1 Waveform Description Language, Volume 2, Commercial Proposal, (Racal Raytheon UK PDR Consortium, 08 September 99)

[3] DERA/CIS1(PDW)/04/1443/1, *UK Programmable Digital radio Phase 1 Requirement Waveform Description Language*, (DERA, June 1999)

[4] FM3TR/TWS/v2.0, FM3TR Test Waveform Specifications Issue 2.0, (FM3TR Technology Group, 9 March 1999)

[5] NATO Standardisation Agreement STANAG4372, Edition 3, 28 April 1999

[6] P6957-11-005, FM3TR Decomposition, (Raytheon, Racal UK PDR Consortium)

[7] P6957-11-012, SATURN Decomposition, (Raytheon, Racal UK PDR Consortium)

[8] P6957-11-013, WDL & JTRS, (Raytheon, Racal UK PDR Consortium)

[9] P6957-11-014, Waveform Description Language, (Raytheon, Racal UK PDR Consortium)

[10] P6957-11-004, PDR Library Primitives, (Raytheon, Racal UK PDR Consortium)

[11] P6957-11-017, Transition Plan, (Raytheon, Racal UK PDR Consortium)

1 Introduction

This report summarises the work conducted during the UK Programmable Digital radio (PDR) Phase 1, Waveform Description Language (WDL) (DERA contract CU009-0000002745) and is detailed in [1].

The aim of PDR Phase 1 is to:

- Identify a methodology for encapsulating waveform descriptions
- Investigate software tools to aid encapsulation of a waveform
- Conduct initial Parameterisation of two waveforms: FM3TR [4] and SATURN [5]
- Identify a transition plan for a migration to an implementation of SATURN and FM3TR.

The report hierarchy for this project is as shown in Figure 1.1. This report (P6957-11-01) acts as an executive summary, summarising the lower level reports. The Work Package (WP) breakdown for the activities conducted during Phase 1 are also summarised below.

Also included in this report is an assessment of the benefits of this programme to MoD.

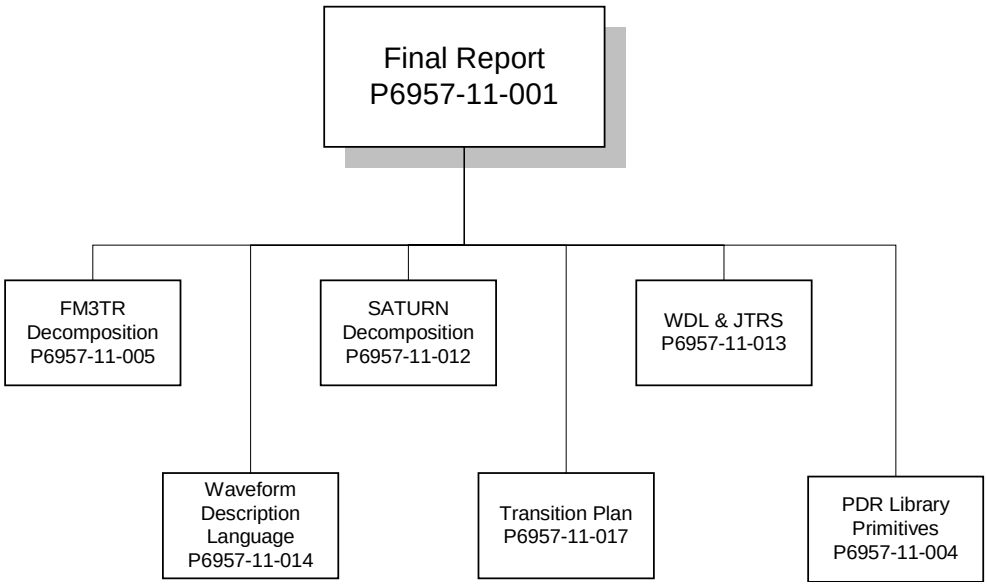

Figure 1.1 Document Hierarchy

Section 2 covers the benefits of this programme to MoD.

Section 3 provides a brief overview of the WDL.

Section 4.1 summarises the work conducted on the Waveform Description Language.

Section 4.2 summarises the object library functions & services and Application Program Interfaces (APIs) available from the US Joint Tactical Radio System (JTRS) programme.

Section 4.3 covers the work conducted to perform an initial parameterisation of FM3TR and SATURN. This has been done using the COTS software tool Visio.

Section 4.4 points the reader to the report covering the transition plan for phase 2.

Section 5 provides a brief overview of the tool used to demonstrate waveform decomposition.

Section 6 points the reader to the report detailing the programme risks.

Section 7 briefly summarises the further work needed on phase 2 to complete the language.

Section 8 contains the main conclusions from the PDR Phase 1 programme.

2 Benefits to MoD

This section will be written by Mark Harrington

3 Waveform Description Language

The aim of WDL is to develop a language and tool set that will allow a waveform to be captured, allow the captured waveform to be refined and allow automated code generation to build the waveform to run on multiple platforms. It has long been apparent that STANAGs and textual specifications for waveforms produce ambiguities, resulting in different implementations by different manufacturers. During the course of the work on the language and parameterisation of waveforms, WDL has highlighted this view. As a minimum, a language that could remove ambiguity and speed the implementation process for a waveform could significantly reduce the cost and risk in any future development.

The WDL aims to go beyond simply a tool to capture a waveform but to provide the means of allowing a captured specification to be refined to a point where manufacturers replace the specification leaf objects with their own implementations, which can then be used in an automated code generation process. If a manufacturer has a well-established library of elements available then minor variants of a waveform may only take a few weeks to implement against many months of work using current techniques. This could significantly reduce the cost to MoD to implement similar waveforms on different platforms.

It is likely that the first use of the WDL tools will incur effort in capturing existing waveforms. New implementations, however, should take less time than using the current STANAG process, as the WDL will remove ambiguities that normally require debate by the approvals committee.

Phase 1 of UK PDR has focused on the initial language development and ensuring that the WDL is compatible with at least the JTRS radio concepts (it is perceived at this stage that JTRS will provide COTS software radios). However, the language concepts are platform independent and the WDL can be used on any software radio platform.

The remainder of this section provides an overview of the WDL concept and initial language use.

3.1 Overview

The block diagram representation of the WDL concept is shown in Figure 3.1. WDL consists of two distinct aspects:

- A WDL capture tool, which utilises the WDL language to parameterise a waveform (provided by the waveform Sponsor) and allows the waveform parameterisation to be refined by the Sponsor or Implementer.
- A WDL compiler, which takes the Implementer-refined parameterisation and creates a waveform bundle for download to the radio.

The initial parameterisation of a STANAG or specification would be done using a schematic editor tool or text entry. The WDL language has two interchangeable formats. One is free format like C or ADA and is suitable for reading and manual editing. The other is an eXtensible Mark-up Language (XML) dialect that will be used for persistent storage and to support access by third party tools such as Xeena or Spy for direct XML maintenance, or Diva for schematic entry.

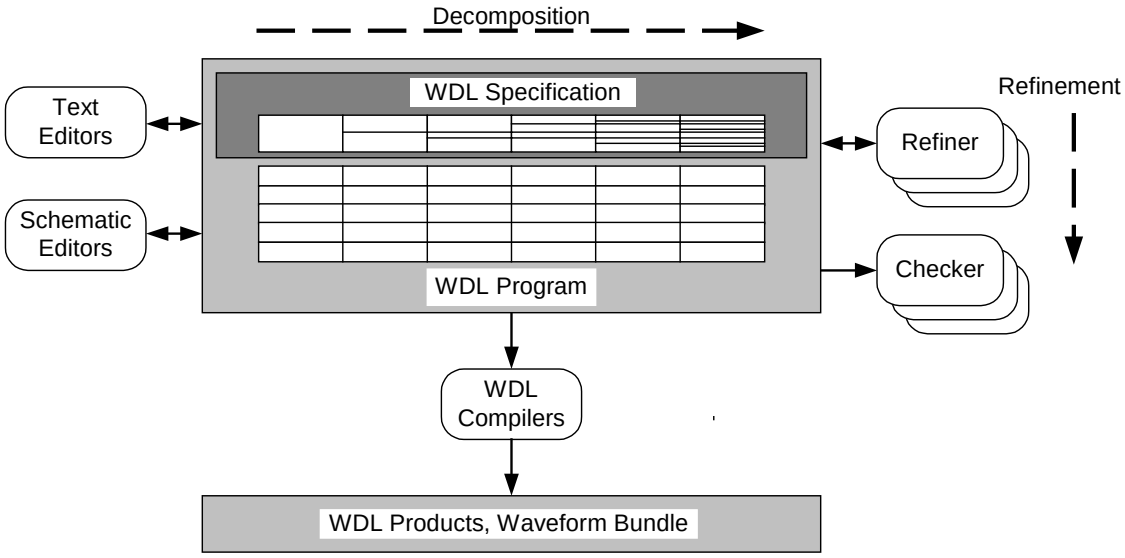

Figure 3.1 WDL Overview

WDL supports the capture of a system specification using decomposition to express the required behaviour as a hierarchy of sub-specifications. Decomposition proceeds until the specification of a pre-existing sub-system can be re-used, or until it is appropriate to define the behaviour using mathematical expressions.

The WDL further supports the progressive refinement of a specification into a program that may then be compiled and executed. This refinement may involve relatively minor addition of detail to support a reference model, or potentially wholesale changes to adapt the specification to exploit pre-existing component libraries.

The refinement process may be assisted by special purpose refinement tools that assist in the resolution of detailed system design issues such as filter implementations.

The integrity of a design may be validated by further tools that check system properties such as processing loads or implementation losses.

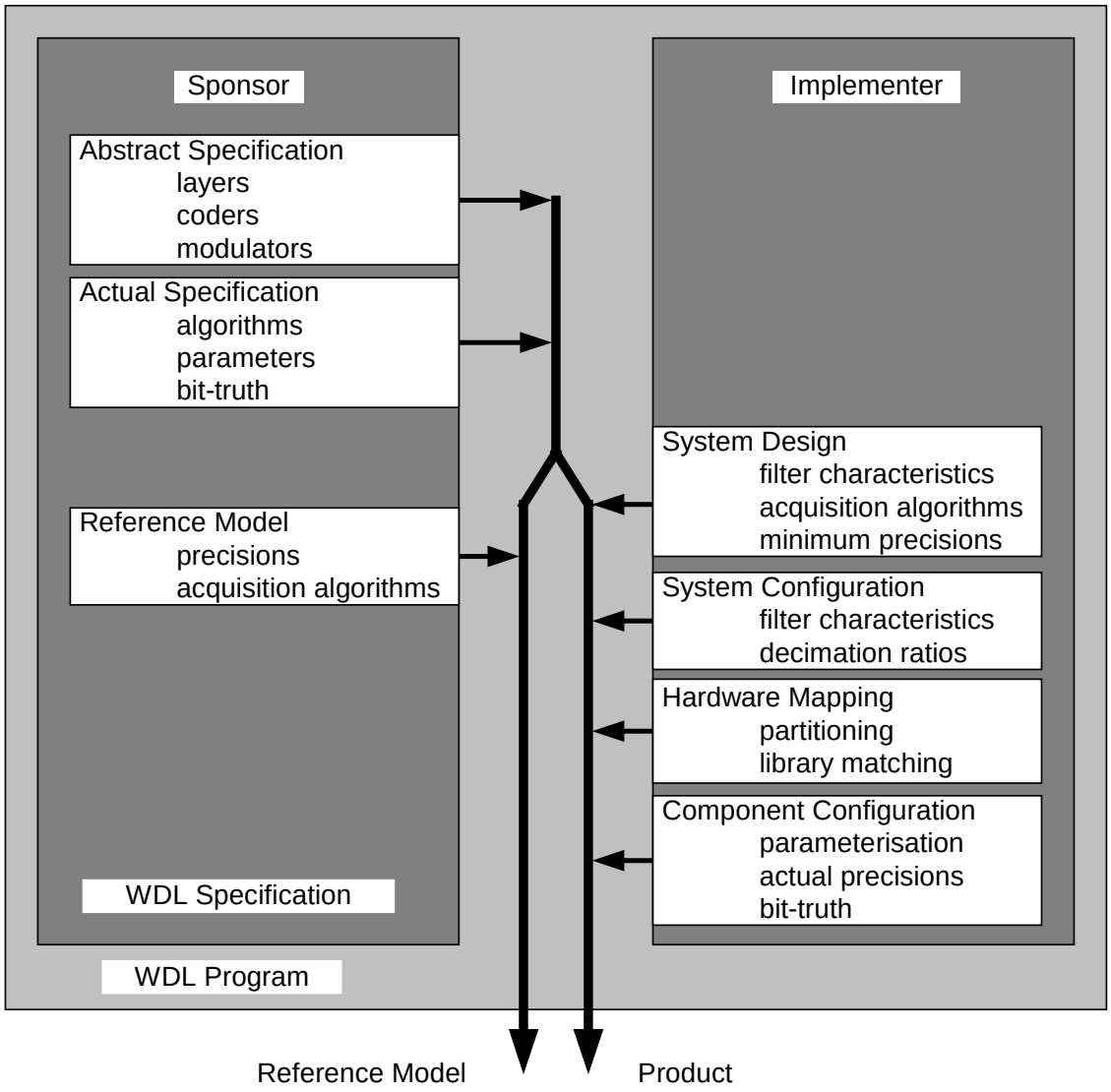

Figure 3.2 WDL Refinement

The refinement activities are shown in more detail in Figure 3.2. The specification may be written in more than one level, with a relatively abstract specification perhaps only defining concepts such as the need for OSI layers and data coding, leaving an actual specification to define protocols and algorithms.

The waveform sponsor may perform a minor refinement to produce a reference model in which example implementations of non-mandatory algorithms may be provided.

The more major refinements are performed by the vendor, who will be concerned to tune the design to minimise computational requirements and match the implementation to particular hardware platforms for which optimised component implementations may already be available. Exploitation of these resources requires the WDL to be refined to describe the required hardware partitioning and to define the interfaces of available components.

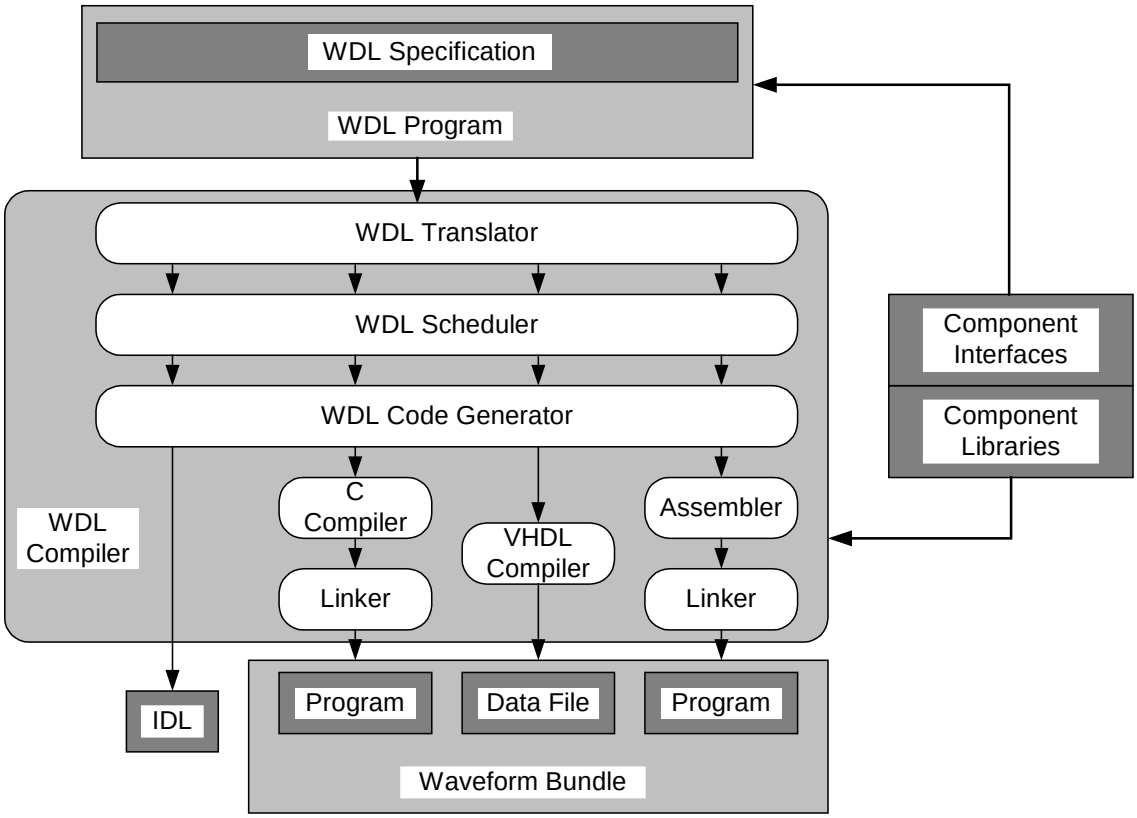

Figure 3.3 WDL Compilation

Direct compilation of WDL would involve an unacceptable degree of software tool development to support a wide variety of target platforms. The compilation process is therefore broken down into stages as shown in Figure 3.3 so that COTS compilers and the Ptolemy system can be exploited. Ptolemy is a freeware product developed by the University of California at Berkeley, it supports scheduling of DSP simulations using a wide variety of computational models, and can be adapted to support code generation. The WDL Scheduler and WDL Code Generator are therefore based upon Ptolemy. The WDL translator presents a resolution of the WDL refinement process and WDL abstractions in a form suitable for use by Ptolemy.

Existing or hand-optimised component libraries are exploited by incorporating their code with the appropriate phase of COTS compilation or linking, and by providing a WDL definition of the interface as part of the refinement process.

The UK PDR phase 1 activities involve investigating the first level parameterisation (the Abstract and Actual Specification in Figure 3.2), reviewing COTS tools to implement the WDL and investigating issues associated with the WDL Compiler.

3.2 Example Parameterisation

The concept that is adopted to parameterise a waveform is to progressively break down the waveform specification into smaller more focused sub-specifications, using state and message flow diagrams, until leaf entities are identified that can mathematically define the specification. In order to define the decomposition process, the semantics and syntax of a language need to be developed. The first pass at defining the language is detailed in [9]. Examples of the language used to parameterise FM3TR and SATURN, are given in [6] and [7]. As a summary, the following sections briefly shows example parameterisation diagrams for FM3TR, illustrating the basic WDL concepts and use.

3.3 FM3TR

FM3TR is an unclassified parametric test waveform, which has been prepared to test interoperability between two different Programmable Digital Radio platforms. The waveform has been written to address the issues of:

- hop rate programmability
- extended frequency band coverage over 30-400 MHz.

The FM3TR waveform has been used extensively throughout the phase 1 work as it provides a good example of a modern waveform, which has exercised the language currently under development. As the final WDL is expected to be issued as an open standard, a tutorial example of a waveform parameterisation is required. FM3TR will fill this role.

3.3.1 FM3TR Parameterisation

The first task in parameterisation of FM3TR is to identify the main specification structure. Figure 3.4 shows the main ports identified at the top level entity for the specification.

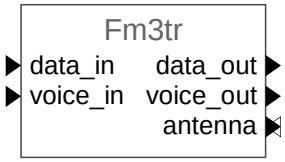

Figure 3.4 FM3TR

The next layer of decomposition identifies the FM3TR protocol layers (as shown in Figure 3.5) and, hence, the main structure of the specification. The FM3TR radio specifies a realisation of the lower three layers of an OSI protocol stack using four layers: Network (Nwk), Data Link Control (Dlc), Medium Access Control (Mac) and Physical (Phl). The FM3TR specification identifies a variety of configuration and operation parameters, but does not specify their interface. An Hci entity is therefore introduced to terminate these unspecified interactions.

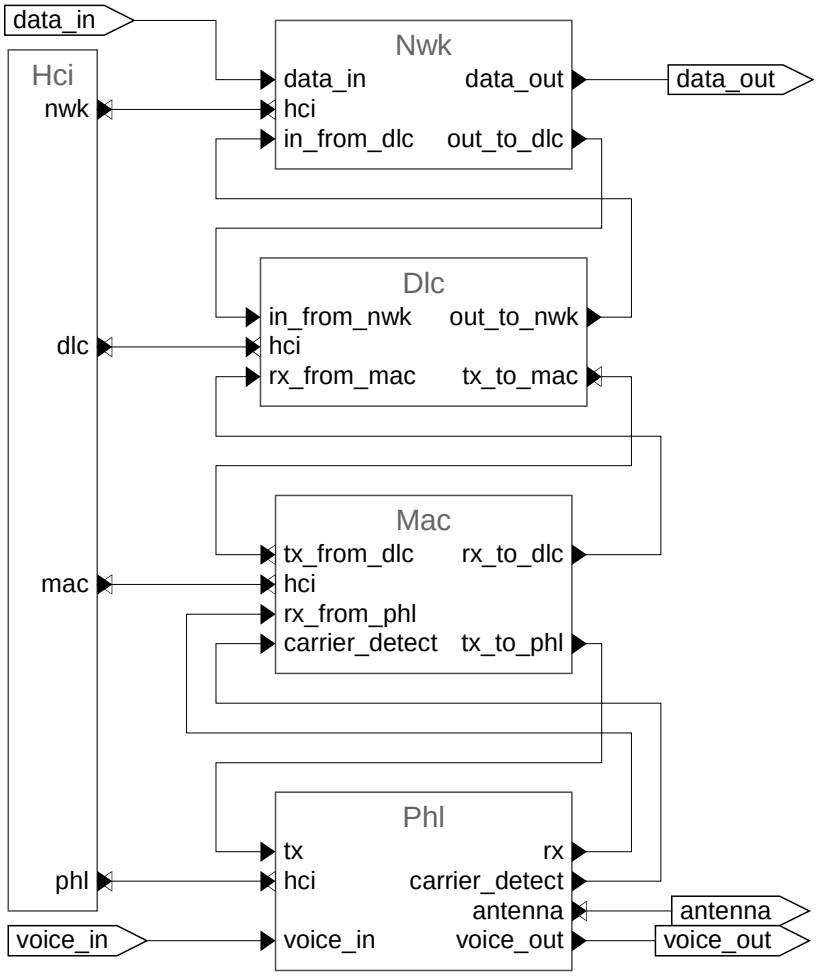

Figure 3.5 FM3TR Structure

Further decomposition of the physical layer entity (Phl in Figure 3.5) identifies the entities shown in Figure 3.6.

The Phl layer supports transmission of either the CVSD `voice_in` or `tx` packets and corresponding reception to produce `voice_out` or `rx` packets. Configuration of the layer is provided through an `hci` message interface that the Hci entity decodes to provide control of carrier detect response and synchronisation robustness. Transmit and receive statistics may be interrogated.

A packet interface is provided by the Rx and Tx entities, with the Cd entity providing a `carrier_detect` signal for use by the pCSMA algorithm in the MAC layer.

Multiplexing of transmit and receive signals to the `antenna` is provided by the Radio entity.

Debounce filtering of the Push To Talk input is provided by the Ptt entity.

The TransSec entity is not specified by FM3TR, but it must clearly supply information to the Fsm that switches between the different operational modes of the physical layer.

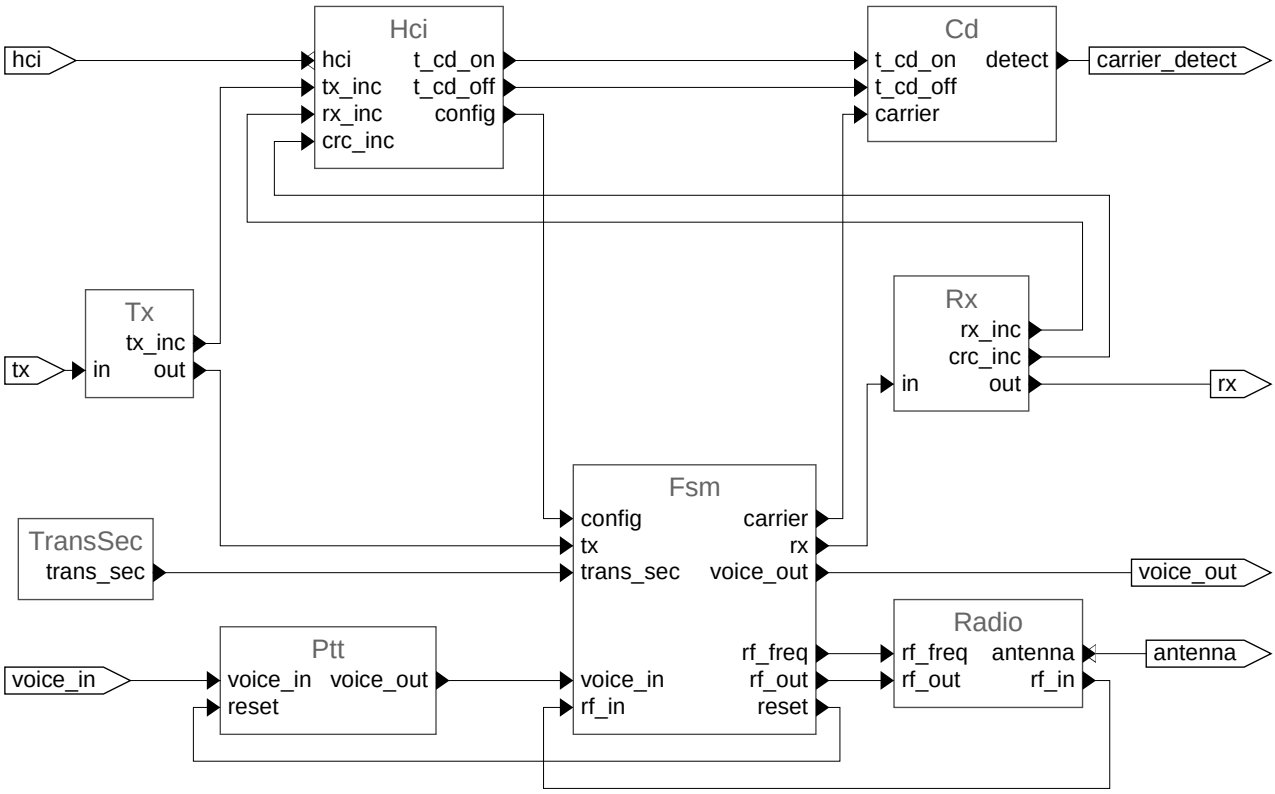

Figure 3.6 Decomposition of the Phl layer in Figure 3.5

The Fsm block in Figure 3.6 represents the entity for the main state machine for the physical layer and its decomposition is shown in Figure 3.7.

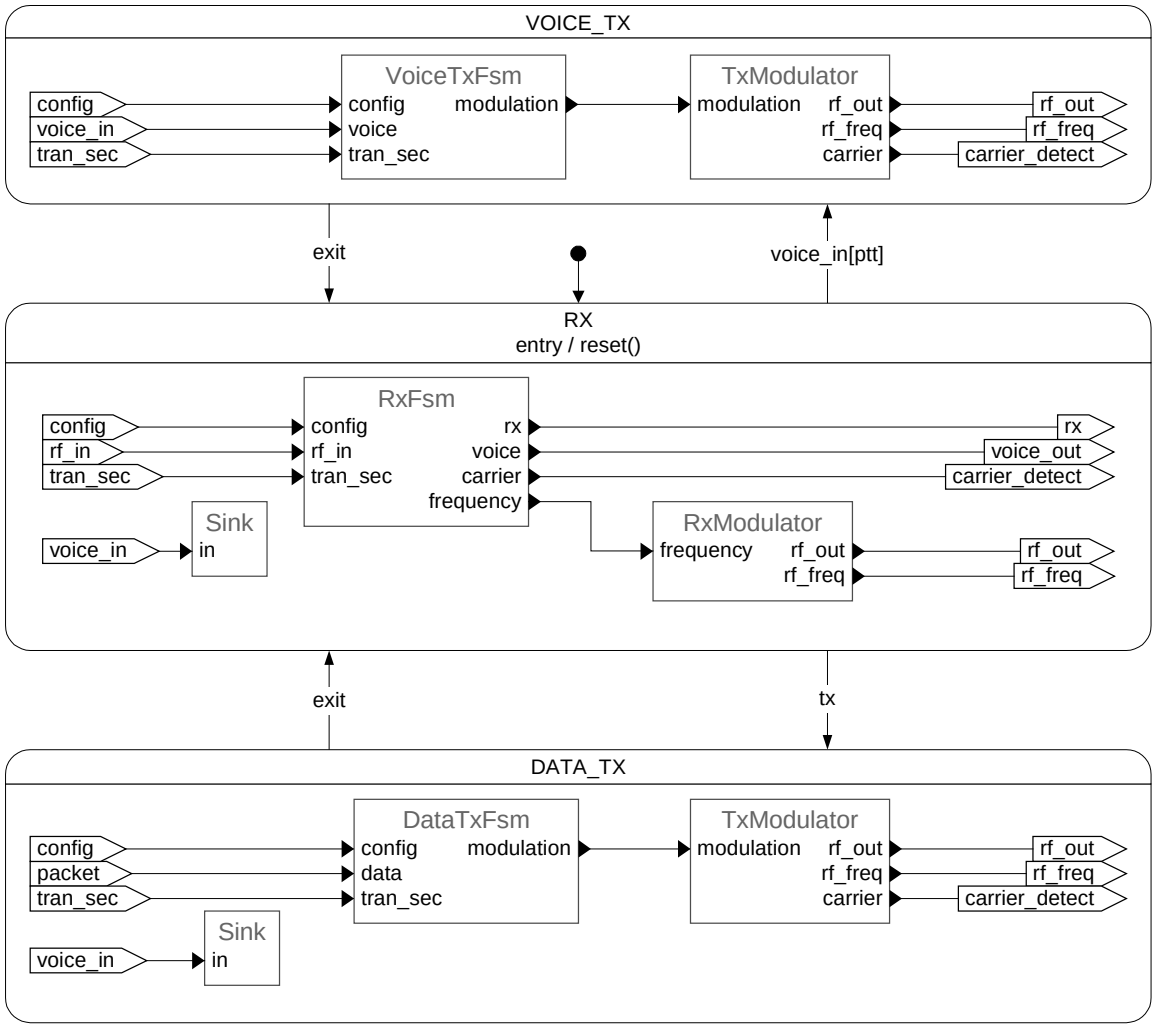

Figure 3.7 Main Physical Layer States (Fsm in Figure 3.6)

Assuming half duplex operation, the FSM idles in the Rx state, entering either VOICE_TX or DATA_TX in response to an appropriate transmit request. Entry to RX issues a reset to the PTT monostable. Each state specifies distinct receive and transmit activity.

The decomposition process continues until leaf entities are identified.

Leaf entities at the bottom level can consist of two fundamental types. Those defined in the WDL Primitives Library [10], which are generic entities likely to be used by other specifications, or entities specific to the current specification. Leaf entities will contain a mathematical expression.

A completed parameterisation would be contained within the WDL tool set, with auto generated HTML based text providing a full description of the parameterisation. Details of all waveform specific leaves would also be included in the parameterisation documentation.

4 Phase 1 Work Packages

The following sections briefly summarise the work conducted on Phase 1. A detailed description of the findings from this work can be found in the lower level documents as detailed in Figure 1.1.

4.1 WP200 Waveform Description Script

This work package focused on the development of WDL, which is detailed in [9]. This is supported with the PDR Library Primitives report [10]. [9] discusses the background to the language, which consists of two distinct and exactly equivalent syntaxes:

- Wdl is a conventional textual language like C or Ada.
- WdML is an XML dialect.

IDL was used as a starting point for the number and type concepts within WDL. However IDL specifies an exact implementation, whereas WDL seeks only to specify the minimum requirements to be satisfied by an implementation. Very little of IDL remains directly visible in WDL, however once a compiler has chosen implementation types for each specified type, translation of WDL to IDL should not prove to difficult.

MoML (the XML dialect used by Ptolemy II) provided some inspiration for the message flow syntax and the overall syntactical structure. However, MoML leaves most syntax to be resolved from unspecified text strings, which is unsatisfactory for a complete language. The 10 or so language elements of MoML therefore become 100 or more, once types and statecharts and expressions are all fully defined.

A WDL specification uses the ideal behaviour of an infinitely fast implementation as a reference, and requires practical implementations to satisfy tolerances with respect to this reference.

- inertia
- runaway for calculation delays
- latency for clock jitter
- spurious energy for signals

Practical implementations may use any algorithm or computation order that satisfies the tolerances on the ideal observable behaviour.

The report defines in detail the language definition covering number definitions, types, language constructs, hierarchy, bit-truth mapping and grammar. A further work section outlines the work required to complete the language definition.

4.1.1 Language Approaches & Tools

WDL supports hierarchical decomposition of a specification into state machines, data flows and leaf behaviours, using whichever of a schematic, statechart or text editor is most appropriate for maintenance. During phase 1 a number of tools were investigated that could help implement the WDL concepts.

WDL also supports progressive refinement of a specification from the abstract form provided by a waveform sponsor to a concrete form required by a reference implementation (simulation) or a vendor's actual implementation. A variant of XML will be used to represent WDL.

The reference model should be as easy to program as possible and as portable as possible. Java has a greater breadth of functionality than anything else, provided speed is not a prime requirement. In particular, the ability to compile and run code within a program is a facility not readily available in compiled languages such as C++.

It is far from coincidental that this assessment coincides with a very similar decision by UCB (Berkeley) to redevelop Ptolemy in Java. Ptolemy II is clearly the most flexible DSP system and comes closest to satisfying the requirements for rapid automated realisation of the reference model. Ptolemy is free, and has formed the basis for most research on DSP scheduling, which continues as part of the Heterogeneous Modelling And Design project. Other tools such as Cossap or SPW are expensive, support a very limited variety of computational models and are unable to handle continuous time, state machines, process networks or user types.

Contributions to compile-time programming should use a single language. It would be unsatisfactory if some library entities required a Basic interpreter, other library entities required an Ada environment and yet other library entities required Lisp. Java is again the best compromise with a very high degree of portability and a substantial breadth of functionality.

4.1.2 JTRS Compatibility

One of the likely sources of COTS software programmable radios is the US JTRS programme. Part of the work on the WDL is to ensure that it can be used for implementing waveforms for JTRS radios. The current work on this and other work packages has not identified any issues that would prevent WDL being used on JTRS.

4.1.3 WDL Syntax and Semantics

As part of Phase 1 an initial definition of the language syntax and semantics needs to be supplied. [9] details the currently documented status of the language and includes details of further work. Section 3.3 also provides an example use of the language for FM3TR parameterisation.

4.2 WP300 & WP500 Object Library Functions and Services & APIs

Work packages WP300 and WP500 were originally intended to be two distinct packages of work. Current work on JTRS has produced the JTRS Software Communications Architecture (SCA) specification, which covers much of the Application Program Interfaces (APIs) and object library functions and services. As a consequence, these two work packages have been merged and the work conducted during phase 1 is covered in [8].

The SCA establishes a framework with baseline requirements for the development of Joint Tactical Radio System (JTRS) software configurable radios. These requirements are comprised of interface specifications, application program interfaces (APIs), behavioural specifications and rules. The goal of this specification is to ensure the portability and configurability of the software and hardware, and to ensure interoperability of products developed using the SCA.

Reference [8] contains a summary of the SCA document, along with the latest version in an Appendix. Also included is an example of the top layers of FM3TR implemented on the JTRS. The implementation is extended to show how the waveform can be mapped to the Raytheon JTRS hardware. As part of this process, necessary WDL waveform refinements are required and these are discussed.

The work conducted during this work package has not yet identified any issues associated with the use of the WDL language for JTRS. In fact, WDL is complementary as it can do the implementation for JTRS radios (or any other software radio platform).

4.3 WP 400 Database of Waveform Specification parameters

The initial view of this work at the outset of the programme was that phase 1 would complete a database of components that would be required to parameterise a waveform and initial parameterisations of FM3TR and SATURN would be done. The waveform parameterisation were also expected to be in the form of a database. The development of the language, however, has progressed further than anticipated and the parameterisation closely resembles the expected WDL final format. The database of components has been replaced with PDR Library Primitives [10]. This contains the currently identified leaf entities that would be used on a number of waveforms. The language supports the creation of waveform specific leaf entities within a parameterisation.

The work conducted on this work package is covered in three reports:

- P6957-11-004, PDR Library Primitives, (Raytheon, Racal UK PDR Consortium) [10]
- P6957-11-005, FM3TR Decomposition, (Raytheon, Racal UK PDR Consortium) [6]
- P6957-11-012, SATURN Decomposition, (Raytheon, Racal UK PDR Consortium) [7]

4.3.1 FM3TR Decomposition

FM3TR has been used extensively to aid the development of the WDL. It is unclassified, which makes it ideal as a tutorial guide covering the use of the language. An extensive decomposition of FM3TR has been done and this is detailed in [7]. An example of the decomposition is shown in Section 3.3. During the decomposition of this waveform a number of errors in the original specification were identified. These have been passed to the specification authors. This has prevented completion of the FM3TR decomposition.

4.3.2 Saturn Decomposition

The SATURN waveform is of key interest to MoD as this waveform will enter service over the next few years. During this phase of work an initial decomposition of this waveform has been done and is detailed in [7]. A full decomposition could not be completed due to the late approval of release of the STANAG to the Phase 1 Consortium. Full parameterisation will be completed in Phase 2.

The current parameterisation provides the first few stages of decomposition to identify the specification structure. The waveform was then decomposed in two key areas.

4.4 WP600 Transition Plan

A transition plan is required as part of the phase 1 activities, identifying the route from Phase 1 to a complete language and implementation demonstrations. The full plan is detailed in [11]. The main activities involve continuation of the language definition, with the further work sections in the reports to be completed, along with definition work on the WDL compiler. A number of implementations are also proposed.

5 Demonstration of Waveform Decomposition

Part of the phase 1 activities involved the development of a demonstrator. The Consortium turned this to their advantage by developing a tool set to facilitate the decomposition of a waveform. All of the decomposition diagrams created in [6] and [7] and the summary diagrams shown in Section 3.2 were all created using the WDL Visio tool set.

Visio is a PC based complex drawing and schematic tool that has a Visual Basic interface, which allows it to be customised. The Visio tool has also been used to create a template of PDR Library Entities, which can be used on any parameterisation.

The demonstration for phase 1 is not a demonstration of the tool, but a demonstration of the output from the tool use on the SATURN and FM3TR decomposition.

6 Programme Risks

The currently identified risks on UK PDR Phase 2 are summarised in [11]. A plan of how these can be controlled is also detailed in the Transition Plan.

7 Further Work

UK PDR Phase 1 is essentially a feasibility phase to identify the language approach and transition plan for a more complete development. UK PDR is a research programme addressing issues and tools that are state of the art. Each of the reports detailed in Figure 1.1 has a further work section, where relevant, which details the work that needs to be completed. A key issue identified in phase 1 is the risk associated with development of the WDL Compiler. The timescale on phase 1 did not allow this issue to be addressed fully and a 3 month investigation programme will be required at the start of phase 2 to identify and plan an adequate risk mitigation strategy.

The transition plan addresses the further work that is needed to complete the language on Phase 2.

8 Conclusion

The UK PDR phase 1 programme has conducted a feasibility study on the development of a software language to parameterise and implement waveforms. The findings from the programme, including further work, are detailed in the reports shown in Figure 1.1. Significant further work is required to complete the language and there are a number of risks associated with this development. However, it is perceived at this stage that such a language would permit considerable reductions in the development time for new waveforms and would simplify the task of porting waveforms between different software radio platforms. This could make a considerable cost saving in future procurements for MoD.

Key benefits from different levels of WDL development are:

- Language alone
  - Formalised specifications- reduced development time & costs
- Language + Checker
  - Formalised, consistent specifications- reduced development time & costs
- Language + Checker + Compiler
  - Faster turnaround- weeks instead of months for waveform changes, significant time & cost savings
  - Portability- waveforms could easily be built for different hardware platforms, significant time & cost savings

The programme for phase 2 needs to include an initial 3 month phase to identify the key risks associated with the WDL Compiler development and to propose a detailed de-risking plan in this area.