HEALTHWATCH-2 SYSTEM OVERVIEW

Eric Barszcz[1]
QSS Group, Inc.
NASA Ames Research Center

Marianne Mosher and Edward M. Huff
Computational Sciences Division
NASA Ames Research Center

ABSTRACT

Healthwatch-2 (HW-2) is a research tool designed to facilitate the development and testing of in-flight health monitoring algorithms. HW-2 software is written in C/C++ and executes on an x86-based computer running the Linux operating system. The executive module has interfaces for collecting various signal data, such as vibration, torque, tachometer, and GPS. It is designed to perform in-flight time or frequency averaging based on specifications defined in a user-supplied configuration file. Averaged data are then passed to a user-supplied algorithm written as a Matlab function. This allows researchers a convenient method for testing in-flight algorithms. In addition to its in-flight capabilities, HW-2 software is also capable of reading archived flight data and processing it as if collected in-flight. This allows algorithms to be developed and tested in the laboratory before being flown. Currently HW-2 has passed its checkout phase and is collecting data on a Bell OH-58C helicopter operated by the U.S. Army at NASA Ames Research Center.

INTRODUCTION

Earlier studies at Ames Research Center involving the analysis of vibration data used a specially developed sensor suite and data acquisition system referred to as *Healthwatch* [1]. That system, which was flown on the U.S. Army's AH-1S Cobra and OH-58C helicopters located at NASA Ames Research Center, did not have several features that were later found to be necessary. Most important, it did not have the ability to perform robust in-flight data preparation or conditional metric computations.

The Healthwatch-2 (HW-2) software system developed at NASA Ames Research Center is specifically designed to aid research into damage detection algorithms. In particular, it was designed to facilitate rapid prototyping of advanced health monitoring concepts and algorithms. The system is written in C/C++ and executes on an x86-based computer running the Linux operating system. The HW-2 executive module has interfaces to collect data from various sources, for example, accelerometers, torque meters, tachometers, and global positioning system (GPS).

A special feature built into HW-2 is that time or frequency averaging can be performed in real-time based on specifications in a user-supplied, *configuration file*. During flight, averaged data are then passed through an interface to a user-supplied algorithm written as a Matlab function. This architecture allows the researcher convenient control when transitioning from the exploratory development of algorithms in the laboratory to in-flight data processing.

Specific research objectives and underlying motivations for developing the HW-2 system were as follows:

Determine the characteristics of undamaged machinery in flight and evaluate damage detection metrics published in the open literature. The thresholds for most metrics were developed based on static test stand conditions. Experience has shown, however, that the normal flight environment often exceeds the thresholds established under such conditions [2].

1. Determine the degree of cyclostationary of various flight states or regimes. Experience has shown that data collected under what are thought to be identical maneuvering conditions have strong non-cyclostationary characteristics [1-3].

2. Develop and evaluate signal separation algorithms for planetary gear systems. Most approaches to signal separation of planetary gears require long sampling periods, which are subject to non-cyclostationary influences [4, 5].

3. Collect and archive high quality raw flight data. As novel data preparation and damage detection algorithms arise, it is most efficient to evaluate them

[1] This research was funded by NASA CICT program.

first on the ground using standard archival data from various flight conditions.

4. Process archived data on the ground. In addition to its in-flight capabilities, HW-2 software is capable of reading archived flight data and processing it as if in-flight. This allows algorithms to be developed and tested in the laboratory on a generic x86-based PC running the Linux operating system.

Currently, HW-2 has completed its checked out phase and is being used to collect data on the same Ames OH-58C helicopter used in earlier studies [2, 6-8]. Since this system will be the basis for several future studies, the remainder of this paper provides details on the specific hardware and software implementation on the OH-58C.

HW-2 HARDWARE ON THE OH-58C

The U.S. Army Aviation and Missile Command operate several research helicopters at Ames Research Center. One of them, a Bell OH-58C, is instrumented with the Healthwatch-2 system. The locations of the major components of the HW-2 system are shown in Figure 1.

Sensors

Four accelerometers are attached to bolts on the oil pump housing of the main transmission. The bolts are numbered 1-14, with bolt one located at the front of the transmission and the count incrementing in the clockwise direction when viewed from above. Three Endevco 7259A-10 single-axis accelerometers are located on bolts 2, 6 and 13. Figure 2 shows two of the single-axis accelerometers. An Endevco 7253C-10 triaxial accelerometer is located near the input pinion on bolt 10 (Figure 3). The accelerometer number, type and bolt locations are summarized in Table 1. The last column in the table labeled "OH-58C (g's)" shows the g-ranges experienced on the OH-58C in flight. They are used to set the gain for each of the accelerometers. Triaxial values are for the x-, y- and z-directions, respectively.

The three single-axis accelerometers are oriented radial to the transmission. For the triaxial accelerometer, the orientation is x tangent to the transmission, y parallel to the main rotor shaft and z radial to the transmission, (Figure 3).

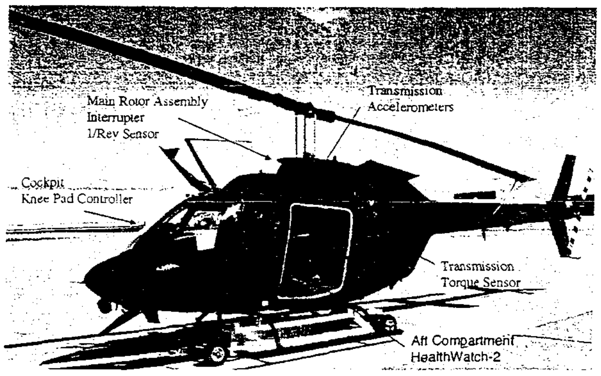
Main Rotor Assembly Interrupter 1/Rev Sensor
Transmission Accelerometers
Cockpit Knee Pad Controller
Transmission Torque Sensor
Aft Compartment HealthWatch-2

Figure 1. OH-58C with HW-2 component locations.

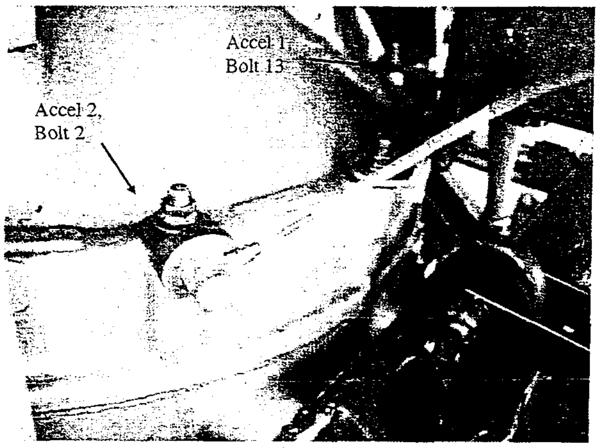

Figure 2. Two single-axis accelerometers on the transmission casing.

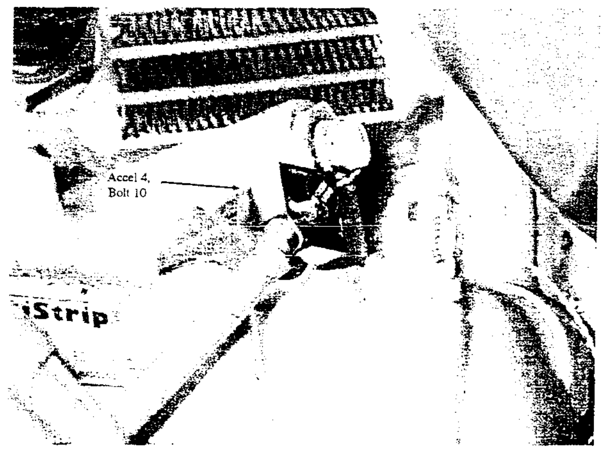

Figure 3. Triaxial accelerometer on transmission casing.

Table 1. Endevco accelerometer model and location.

Accel	Model	Bolt	OH-58C (g's)
1	7259A-10	13	±72
2	7259A-10	2	±72
3	7259A-10	6	±72
4	7253C-10	10	±140, ±140, ±72

Accelerometer gains were set by swapping resistors on a hand fabricated signal conditioning board. Based on reference sensitivity calibrations provided by the manufacturer and tolerance on resistor values, accuracies for the accelerometers are expected to be within 5%. Tap tests were performed on the mounted accelerometers to look for resonances. The accelerometers varied widely with regard to the number and location of resonant peaks, but none were judged to be of sufficient magnitude to warrant immediate concern. Nonetheless, signals from different channels may generally be expected to differ because of resonant response variations.

A torque sensor based on engine oil pressure is located in the ceiling at the back of the aft compartment (Figure 4). A graph of results from the torque calibration test is shown in Figure 5. The percentage of nominal torque (as displayed on the pilot's cockpit gauge) is linearly related to the counts recorded by the sensor. Note that readings were taken while monotonically increasing torque level and then monotonically decreasing torque level.

A once per revolution sensor for the main rotor shaft is shown in Figure 6. An interrupter (not seen) sweeps past the magnetic pickup sensor once every rotation of the main rotor shaft. The signal is sampled at 50 kHz, which is the same rate at which the vibration and torque signals are sampled.

Data Acquisition Hardware

A block diagram of the HW-2 data acquisition hardware is shown in Figure 7. Since the OH-58C does not have a 1553 bus, pilot command inputs and aircraft attitudes are not available. Accordingly, an inertial measurement unit (IMU) and global positioning system (GPS) are currently being integrated into the HW-2 system although they are not represented in Figure 7.

All analog signals pass through an 8-pole elliptic anti-aliasing filter (AAF). Signal attenuation is very high above 20 kHz, which prevents aliasing during A/D conversion. The sampling frequency on all channels is 50 kHz. There is also a high pass filter with a corner frequency of 70 Hz. This filter attenuates any low frequency signals under 70 Hz coming from the accelerometers. Note that these accelerometers do not have a DC response. The frequency response of the AAF is shown in Figure 8.

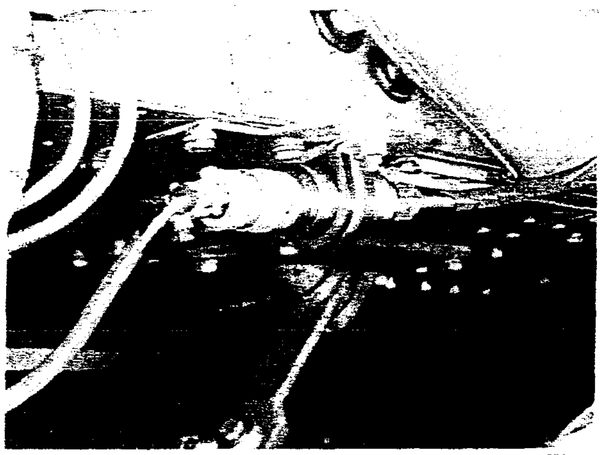

Figure 4. Oil pressure based torque sensor on ceiling of aft compartment.

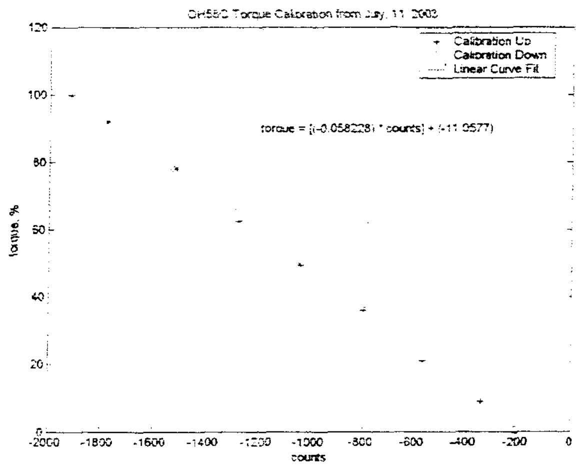

Figure 5. OH-58C torque sensor calibration.

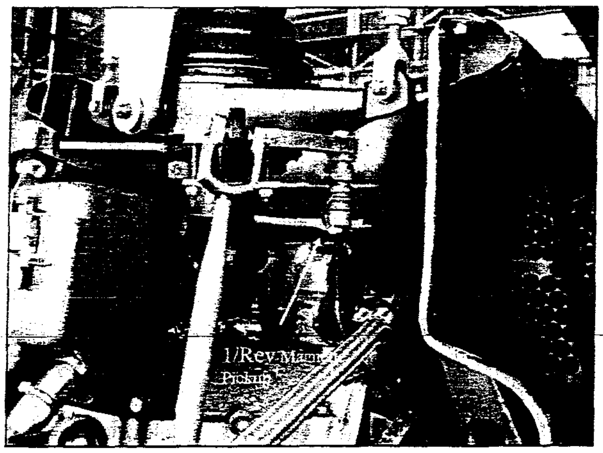

Figure 6. Main rotor once per revolution sensor.

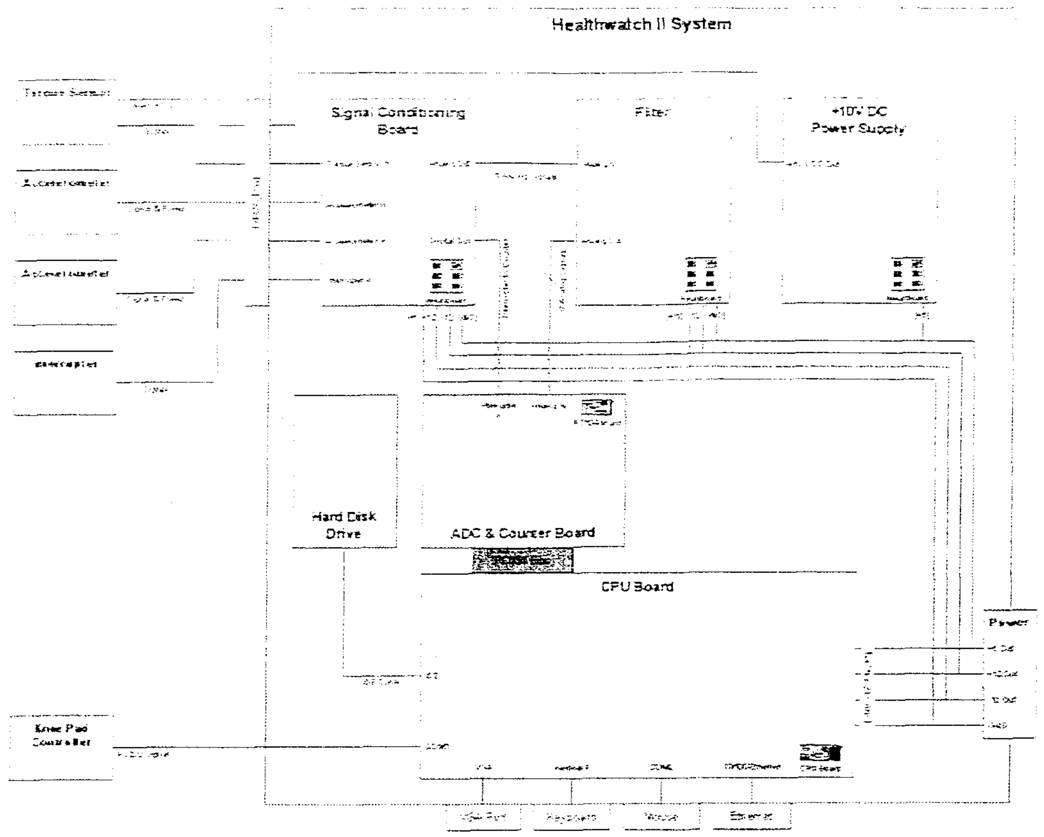

Figure 7. Block diagram of Healthwatch-2 data acquisition system.

The analog-to-digital converter (A/D) card is a Real Time Devices DM7520HR-8 card. On the OH-58C, the A/D card is configured as 8 differential channels with 12 bits of precision being sampled at 50 kHz each. The card is capable of 1.25 MHz throughput. The public domain library, Linux Control and Measurement Device Interface (COMEDI) [9], was used to develop a device driver for the card.

The HW-2 system runs on a PC-104 computer with a 450 MHz AMD-K6 processor, 512 MB of DRAM, and a 20 GB IBM ATA hard disk. A keyboard, monitor and mouse can be plugged directly into the chassis. For file transfer, a laptop is connected via 10/100 Mb Ethernet. Communication with the pilot is through the Knee Pad Controller (Figure 9) connected to a serial port.

The operating system is Red Hat Linux with a 2.4.18 kernel. Most of the services are shut down during flight to free resources for data acquisition. In addition to minimizing the number of services, the operating system is configured to automatically load the device drivers for the A/D card and then start the HW-2 Executive.

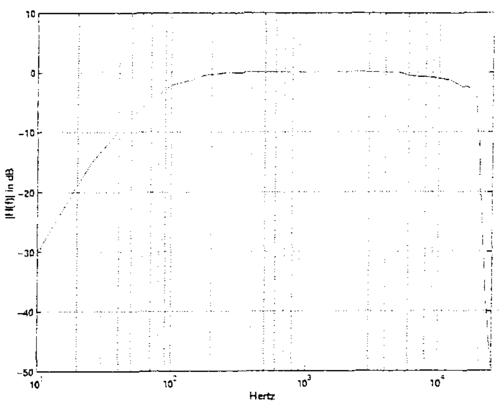

Figure 8. Frequency response of the 8-pole anti-aliasing filter.

The source code for the Linux line printer daemon was modified to count the number of interrupts seen by the once-per-revolution sensor (tachometer) during a flight. This feature allows discontinuous synchronous averaging (DSA) to be performed on shafts other than the main rotor shaft when data are taken across recording epochs within a flight. DSA is described in greater detail in the *Data Averaging* portion of the Configuration File section below.

Pilots interact with the HW-2 system through the Knee Pad Controller (KPC). It is a Sharp OZ-730 Wizard with a keyboard and an 8-line by 40-character display (Figure 9). It runs on two AA batteries that are changed before each 2.5 hour flight. Velcro glued to the bottom of the KPC secures it during a flight. The KPC itself is programmed to communicate with the HW-2 computer and display simple text messages. The text messages are either informational or queries that the pilot can respond to with a few key strokes.

The type of pilot interaction is specified in the HW-2 configuration file. However, a typical flight has the following pattern. When turned on, the KPC displays the boot status of the HW-2 system. After booting, the pilot's initials are requested. Thereafter, the system waits for the pilot to initiate the data acquisition process. Once initiated, the system cycles through acquisition, processing, and data storage stages (i.e., a recording epoch), based on values specified in the configuration file. During the execution of each such data acquisition cycle the KPC displays the current state. While data are being saved, the record number is also displayed, and may be used by the pilot for taking written or oral notes. At any point in time, the pilot is allowed to halt the system. However, except for the first one, the pilot's input is not required to initiate each recording epoch.

If data are to be acquired under specific flight conditions known only to the pilot, the configuration file can be modified so that the pilot uses the KPC to initiate each recording epoch. In this case, the pilot is also records the flight condition and sample record number on a flight card to allow post-flight correlation of flight condition with record number.

Since HW-2 is a research tool and not designed to assist maintenance technicians, there is no formal ground station. To transfer files, a keyboard and monitor are attached directly to the HW-2 chassis, and the system booted into multi-user mode. At this point, a laptop computer is attached via Ethernet and files downloaded. After downloading, the files are deleted from HW-2 to free disk space for the next flight. When collecting raw data, a 2.5 hour flight generates approximately 4 GB (giga-bytes) of data.

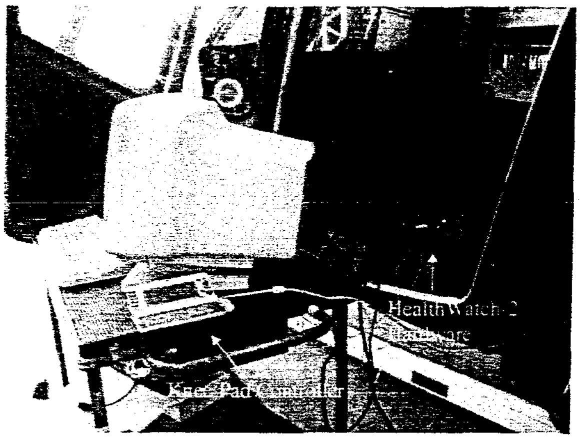

Figure 9. Healthwatch-2 in aft compartment and Knee Pad Controller.

A future direction is to attach HW-2 to the local area network (LAN) in the hanger to allow direct uploading of files to the flight data archive.

HW-2 SOFTWARE

HW-2 software is written in C/C++ and runs on any PC compatible running the Linux operating system. Fault detection algorithms are written as Matlab functions [10]. The functions are compiled and called by the HW-2 executive during flight. See the *MATLAB Compiler User's Guide* [11] for details on using Matlab with C++.

An unanticipated interaction between the HW-2 executive and Matlab revolved around the public domain FFT library known as FFTW [12]. HW-2 uses FFTW to compute the FFT. Matlab also uses some of the FFTW routines to compute FFT's resulting in a conflict between the Matlab runtime support library and the FFTW library. See the Mathworks web site [13] for a list of the specific FFTW functions used by Matlab. The issue was resolved by renaming the functions in the FFTW library.

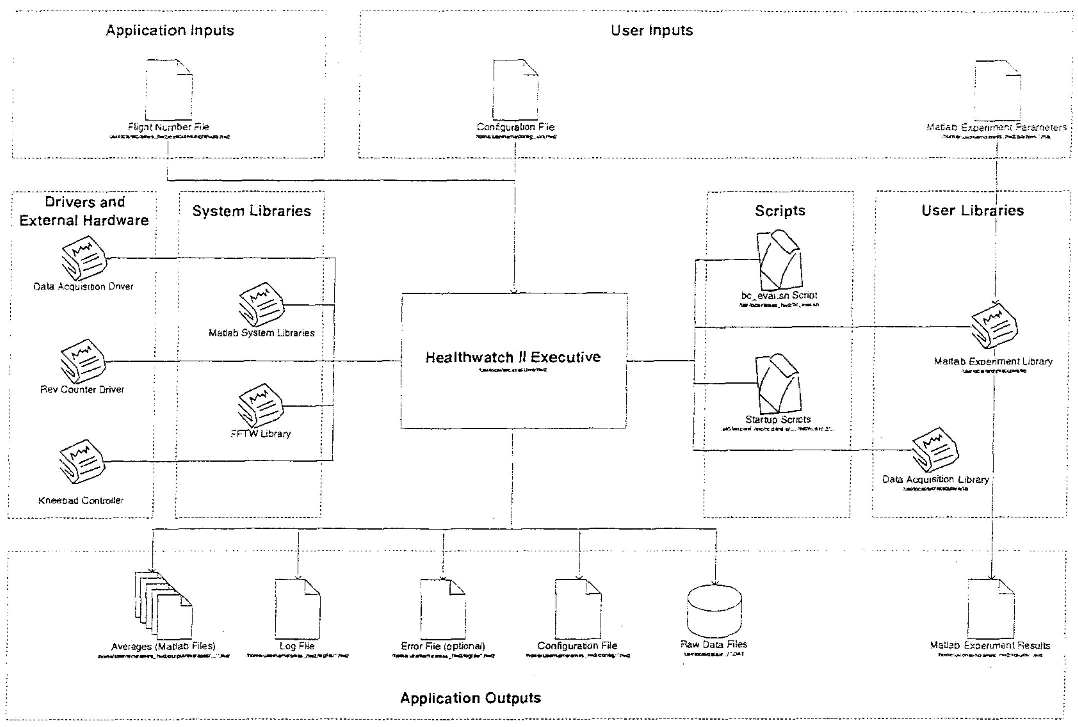

Figure 10. HeathWatch-2 software architecture block diagram.

Software Architecture

A block diagram of the HW-2 software architecture is shown in Figure 10. Of particular interest to users are the configuration file and Matlab experiment parameters file. The configuration file controls overall execution and is described below. The Matlab experiment parameters file allows parameters of a fault detection algorithm written in Matlab to be changed without recompiling the function. For example, a fault detection algorithm may employ a threshold when determining the presence of a fault. The threshold value may be set in the parameters file rather than inside the algorithm itself.

Communication with the pilot is conducted via the KPC. As mentioned above, at each stage the status is displayed on the KPC, and the pilots may submit a request for the HW-2 system to shut down at any time. However, the HW-2 software only checks for requests from the pilot at the end of each data acquisition cycle. If the system needs to be shut down immediately, as in an emergency, the pilot has control authority over the power to the HW-2 system.

Pseudo code for the overall execution of the HW-2 software is listed in Figure 11.

```
1)  read configuration file
2)  prepare to acquire data
3)  communicate status to pilots via the KPC
4)  wait for pilots to initiate data collection
5)  while not done [
6)        acquire data
7)        if saving raw data, write it to disk
8)        process data as specified in the configuration file
9)        save processed data to disk
10)       pass data to Matlab function
11)       wait for Matlab function to terminate
12)       check if done collecting data
    ]
13) indicate HealthWatch-2 is terminating
14) perform housecleaning
15) exit
```

Figure 11. HW-2 executive execution flow.

Configuration File

A sample configuration file is shown in Figure 12. A configuration file is composed of sections, blocks and statements. Each section is labeled and delimited by braces. A section can be composed of one or more blocks and statements. A block is labeled and delimited by braces and composed of statements. A statement has a "C-like" syntax with a parameter label on the left hand side followed by an equal sign and then either a string enclosed in double quotes, numeric value, numeric range, or a keyword. A statement is terminated by a semicolon. Certain labels are expected to have a number appended to indicate which case it refers too. For example, "AccelerometerName.2" in Figure 12 indicates that the associated string is the name of the second accelerometer channel.

The *Session Description* section informs HW-2 which sensor package to use by declaring the aircraft and whether the data are coming from sensors in flight or from the data archive. If data come from the archive then a directory path is included. Currently, the sensor packages on a particular aircraft do not change often, so the aircraft designation informs HW-2 what sensors are available.

Also included in the *Session Description* section are parameters relating to the Matlab function, where to look for the Matlab parameters file, and the maximum allowed time for each call to the Matlab function. If the Matlab function has not finished processing within its allotted time, the HW-2 executive terminates it. However, there is a provision to allow the Matlab process to continue until completion. In either case, the Matlab function terminates before starting the next data acquisition cycle.

The *Session Milestones* section in the configuration file determines how the session starts, how each data acquisition cycle is initiated, and how the session will be terminated. A session can start automatically when the HW-2 system boots or after the pilot provides an input. The *SoftTriggerMethod* parameter determines whether each acquisition cycle is triggered manually by the pilot or by a timer. Termination of the session is signaled by the pilot or after some number of recording epochs.

The *Data Acquisition* section assigns names to each accelerometer channel. This section also defines the length of time (in milliseconds) for each sampling epoch through the *DataAcquisitionTime* parameter.

```
SessionDescription {
  System = OH58C_FLIGHT;
  PrincipalInvestigator = "Marianne Mosher";
  Comments = "Collect Raw Data from Proscribed Flights";
  RecordPilotInitials = YES;
  ExperimentParamsDirectory = ".";
  ExperimentTestNumber = 3;
  ExperimentTimerValue = 10000;
}

SessionMilestones {
  SessionStartMethod = KPC;
  SoftTriggerMethod = TIMER;
  SoftTriggerTimerValue = 0;
  SessionEndMethod = KPC;
}

DataAcquisition {
  AccelerometerName.1 = "Sensor 1";
  AccelerometerName.2 = "Sensor 2";
  AccelerometerName.3 = "Sensor 3";
  AccelerometerName.4 = "Sensor 4, X Channel";
  AccelerometerName.5 = "Sensor 4, Y Channel";
  AccelerometerName.6 = "Sensor 4, Z Channel";
  DataAcquisitionTime = 34000;
}

DataScaling {
  ADCAccelScaling.1 = 0.03516484;
  ADCAccelScaling.2 = 0.03516484;
  ADCAccelScaling.3 = 0.03516484;
  ADCAccelScaling.4 = 0.06837607;
  ADCAccelScaling.5 = 0.06837607;
  ADCAccelScaling.6 = 0.03516484;
  ADCTorqueScaling = -0.058228;
  ADCTorqueOffset = 189.9035;
}

DataAveraging {
  CriteriaSet.1 {
    ADCInterrupterRPM = 250, 450;
  }
  Gear.1 {
    Name = "Carrier";
    SamplesPerEpoch = 8192;
    EpochsPerAverage = 50;
    GearRatioEquation = "1";
    NumberOfTeeth = 99;
    TSA = YES;
    PSA = YES;
    PSAWindowFunction = HANNING;
  }
}

DataSaving {
  SaveRawData = YES;
  BaseDirectory = "/hw2output/$DV$_$TN$$FN$/";
  ADCDataFiles = "adc/$DV$_$tn$$fn$$rn$.dat";
  GPSFile = "gps/$DV$_$tn$$fn$$rn$gps.hw2";
  SystemLogFile = "log/$DV$_$tn$$fn$log.hw2";
  SystemErrorFile = "log/$DV$_$tn$$fn$error.hw2";
  FinalConfigurationFile = "config/$DV$_$tn$$fn$config.hw2";
  AveragesFile = "average/$CI$$GI$/$DV$_$tn$$fn$$ci$$gi$$rn$.mat";
  ResearchResultsDirectory = "results/";
}
```

Figure 12. Sample configuration file.

Table 2. Criterion set metrics for the OH-58C.

Accelerometer	ADCAccelerometerVariance.1
	ADCAccelerometerVariance.2
	ADCAccelerometerVariance.3
	ADCAccelerometerVariance.4
	ADCAccelerometerVariance.5
	ADCAccelerometerVariance.6
Torque	ADCTorqueMean
	ADCTorqueVariance
	ADCTorqueCD
RPM	ADCInterrupterRPM

The *Data Scaling* section specifies accelerometer gains and offsets. The values are used to convert from raw counts to engineering units. On the OH-58C, the first six channels contain vibration data. The next is torque and the last is the tachometer signal. Vibration data are scaled by the specified values. Raw data values for torque are scaled according to the following equation: $torque = (rawCount + offset) \times gain$. Note that this is not the same equation for percentage torque as given in Figure 5, so the torque offset values differ between the calibration graph and the sample configuration file.

The *Data Averaging* section controls discontinuous signal averaging (DSA) activities, so called because data from shaft revolutions entering into a particular average are not required to be continuous in time. Rather, the DSA procedure first allows signals during each revolution of the main rotor shaft to be examined, to determine if one or more *a priori* selection criteria are met. If so, that revolution's data will be selected for use when computing discontinuous averages of either of two types: TSA refers to time-synchronous averages computed in the time domain; PSA refers to power-spectral averages computed in the frequency domain.

The selection criteria are specified in the configuration file as criterion sets, along with appropriate instructions that specify which gear averages to compute with the data. This latter information is referred to as the "gear block." A criterion set contains one or more statements each of which specifies a min-max range for designated metrics that are calculated by carrier revolution. If a metric's value lies within the stated range then it satisfies the criterion, i.e., $minValue \leq Value < maxValue$.

Each criterion set has an associated label of the form *Criteria.#*. Metrics that can be tested on the OH-58C are shown in Table 2, and are currently being expanded to include GPS and IMU metrics. Two or more criterion sets may contain tests for the same metric, and data from each carrier revolution are tested against all criterion sets.

Table 3. File name substitution parameters.

Code	Description
\$DV\$	Device
\$TN\$, (\$tn\$)	Test Number
\$FN\$, (\$fn\$)	Flight Number
\$PN\$, (\$pn\$)	Processing Number
\$GI\$, (\$gi\$)	Gear Index
\$CI\$, (\$ci\$)	Criterion set Index
\$RN\$, (\$rn\$)	Record Number

A gear block designates a particular gear in the transmission (i.e., by target shaft speed relative to output carrier shaft, and the number of teeth), target number of interpolation points, and which type of averaging to perform, i.e., TSA or PSA. Like criterion set blocks, each gear block has an associated label, *Gear.#*.

Once a carrier revolution has passed a criterion set, the target gear revolution boundaries are determined using the revolution counter and relative target shaft speed given in the *GearRatioEquation*. Each target gear revolution is interpolated to the number of sample points specified by *SamplesPerEpoch*. If PSA is desired, the FFT is computed from the interpolated data and the power spectrum added to the PSA accumulator. For TSA, the interpolated data is simply added to the TSA accumulator. Once *EpochsPerAverage* target gear revolutions have been accumulated, the averages are computed from the accumulators.

The criterion set applies to the signal for a complete rotation of the main rotor shaft, not to each revolution of the target gears. Hence, it is possible that averages for the input pinion, which rotates approximately 17 times faster than the main rotor shaft, will include data that do not satisfy the criterion set. Although such data should be very close to passing since a carrier rotation takes about a fifth of a second and the computed metrics do not change very rapidly, this feature will be improved upon in the future.

The *Data Saving* section determines whether HW-2 will save raw input data, and where the various files are to be stored. Unique file names are automatically generated for each flight by including various substitution parameters into the filename in the configuration file. The list of substitution parameters is given in Table 3. Note where there are upper and lower case substitution parameters, the file name will include the first letter of the substitution parameter in upper or lower case respectively. For example, for flight 128, '\$FN\$' in the file path will be replaced by 'F128'. Whereas '\$fn\$' is replaced by 'f128'.

Processing Data

Processing the raw data to compute averages is the longest and most computationally intensive part of HW-2. The logic for this step is shown in Figure 13. The data from each sampling epoch are processed through several loops. Within these loops, the data are interpolated, averaged, and placed into data structures in preparation for being passed to the compiled Matlab function.

```
convert Raw Data to internal format
determine Main Rotor Revolution boundaries
for each Main Rotor Revolution do [
  calculate Criteria Set metrics
  for each Criteria Set do [
    if Main Rotor Revolution meets Criteria Set do [
      for each Gear do [
        calculate phase of the Gear relative to the Main Rotor
        determine Gear Revolution boundaries
        for each Gear Revolution do [
          resample Gear Revolution to specified number of samples
          add Gear Revolution to accumulators
          if numRevsInAverage == revsPerAverage do [
            compute Averages (TSA/PSA)
            calculate statistics
            save Averages and statistics
            construct MATLAB data structures
            call MATLAB function with data structures
            wait for MATLAB function return
            reset accumulators
          ]
        ]
      ]
    ]
  ]
]
```

Figure 13. HW-2 data processing pseudocode.

Stages in the compute-intensive portion of the HW-2 code have been profiled. The top five activities are shown in Table 4. Together, they account for more than 92% of the total compute time. It is not surprising that cubic spline interpolation and power spectral computations account for nearly 75% of the time. Cubic spline computations are about 3 times slower than linear interpolation, and changing to linear interpolation would speed up the overall computation by about 50%. However, a comparison between the two methods showed that cubic splines retain more of the energy in the original signal. It was decided for research purposes that additional accuracy was more important than computing speed. Moreover, since the hardware will be replaced by significantly faster systems in the future, the issue will become irrelevant.

What is surprising is that computing the checksum and reshaping from a multiplexed format to vectors accounts for over 6% of the time. These routines involve memory operations and are not compute intensive. The time taken becomes understandable, however, when the total amount of data collected each sampling period is considered. Each sampling epoch lasts for 34 seconds. Sampling eight 12-bit channels at 50 kHz for 34 seconds generates 1.7 million 2-byte samples per channel for a total of 26 MB per sampling epoch. Thirty-four seconds is roughly 200 main rotor shaft revolutions. Under these conditions, any function that touches all of the data will take a significant amount of time. The update statistics routine, which does some computation in addition to accessing all of the data, accounts for 12% of the time. This lends support to the hypothesis.

Table 4. Top five compute-intensive activities.

Function	% Time
Cubic Spline	49.11
Power Spectrum	24.88
Update Statistics	12.08
Reshape Raw Data	5.09
Checksum	1.58

COMMENTS AND FUTURE DIRECTIONS

The first version of HW-2 system was officially flight tested on the OH-58C in September 2003 at Ames Research Center. Since that time, it has been used to collect and process a significant amount of OH-58C data in-flight (12 flights), and also to flesh out a legacy database for the evaluation of future metric and data preparation methods.

Because GPS and IMU capabilities were added to the aircraft after the initial version of HW-2 software was tested, the code is currently being updated. Version HW-2a will allow the inclusion of GPS derived measures (e.g., ground speed, climb rate, etc.) in the selection criterion sets for DSA applications. Thereafter, version HW-2b will include aircraft attitudes, as well as measures that reflect pilot control inputs.

Building upon the recent installation of the Army's Vibration Monitoring Enhancement Program (VMEP) [14] capability on a UH-60L aircraft, located at Ames Research Center, a Healthwatch system will soon be under development. This is envisioned to greatly expand HW-2 capabilities, since extensive 1553 data will be available — including pilot stick input activity and flight controller states.

In this regard, it is also noteworthy that the real-time capabilities being incorporated into various Healthwatch systems for DSA purposes are also completely compatible with usage monitoring schemes for maintenance applications, and regime-state recognition for component parts replacement. These avenues of future research are also being actively explored.

ACKNOWLEDGEMENTS

HW-2 software was subcontracted to Sigpro, Inc. and written in accordance with functional specifications provided by the U.S. Government. Software validation and further refinement of the HW-2 code occurs under contract to QSS Group, Inc.

The authors would like to express gratitude to Mr. Larry Cochrane, Sigpro, for highly professional aircraft instrumentation and data collection support. Thanks is also extended to Mr. Munro Dearing, III and Major David Arterburn, who piloted the aircraft through flight tests, and in general to the staff of the Army/NASA Flight Projects Office headed by Major Arterburn.

REFERENCES

[1] Huff, E. M., Tumer, I. Y., Barszcz, E., Dzwonczyk, M., and McNames, J., "An Analysis of Maneuvering Effects on Transmission Vibrations in an AH-1 Cobra Helicopter," *Journal of the American Helicopter Society*, pp. 42-49, 2002.

[2] Mosher, M., Pryor, A. H., and Huff, E. M., "Evaluation of Standard Gear Metrics in Helicopter Flight Operation," presented at 56th Meeting of the Society for Machinery Failure Prevention Technology, Virginia Beach, VA, 2002.

[3] Huff, E. M., Barszcz, E., Tumer, I. Y., Dzwonczyk, M., and McNames, J., "Experimental Analysis of Steady-State Maneuvering Effects on Transmission Vibration Patterns Recorded in an AH-1 Cobra Helicopter," in *56th Annual Forum*. Virginia Beach, VA: American Helicopter Society, 2000.

[4] McFadden, P. D., "A Technique for Calculating the Time Domain Averages of the Vibration of the Individual Planet Gears and the Sun Gear in an Epicyclic Gearbox," *Journal of Sound and vibration*, vol. 144, pp. 163-172, 1991.

[5] Samuel, P. D. and Pines, D. J., "Vibration Separation and Diagnostics of Planetary Geartrains." Virginia Beach, VA: American Helicopter society, 2000.

[6] Huff, E. M., Tumer, I. Y., and Mosher, M., "An Experimental Comparison of Transmission Vibration Responses from OH-58 and AH-1 Helicopters," presented at American Helicopter Society 57th Annual Forum, Washington D.C., 2001.

[7] Huff, E. M., Mosher, M., and Barszcz, E., "An Exploration of Discontinuous Time Synchronous Averaging Using Helicopter Flight Vibration Data," presented at American Helicopter Society 59th Annual Forum,, Phoenix, AZ, 2003.

[8] Tumer, I. Y. and Huff, E. M., "Analysis of Triaxial Vibration Data for Health Monitoring of Helicopter Gearboxes," *ASME Journal of Vibration and Acoustics*, vol. 125, pp. 120-128, 2003.

[9] "http://www.comedi.org."

[10] *Using MATLAB*: The MathWorks, Inc., Natick, MA, 2002.

[11] *MATLAB Compiler User's Guide*: The MathWorks, Inc., Natick, MA, 2002.

[12] "http://www.fftw.org."

[13] "http://www.mathworks.com/support/solutions/data/33683.html."

[14] Grabill, P., Berry, J., Grant, L., and Porter, J., "Automated Helicopter Vibration Diagonstics for the US Army and National Guard," presented at 57th Annual Forum of the American Helicopter Society, Washington D.C., 2001.