Preparing Life Usage Monitoring for the Next Decade

H.Pfoertner

C.Roß

Motoren- und Turbinenunion München GmbH

Summary

Continuous monitoring by an on-board engine monitoring system (EMS) is an attractive approach for economically feasible and safe engine life usage monitoring. EMS updates must be considered if financial benefits can be expected from improved accuracy of monitoring algorithms, if experience from engine operation or from accompanying tests suggest the introduction of new lifing concepts, or if changes occur in build standards, manufacturing procedures or in the engine air system. For a minimum-effort development of an EMS update a process chain with optimized interfaces is being developed at MTU. It includes all stages from detailed calculations to automatic generation of the EMS software. As an example the update of the OLMOS software is presented.

Eine kontinuierliche Überwachung durch ein on-board-System (EMS) ist ein attraktiver Weg einer wirtschaftlich sinnvollen und sicheren Überwachung des Lebensdauerverbrauchs eines Triebwerks. Updates des EMS müssen in Betracht gezogen werden, wenn finanzielle Vorteile aus verbesserten Algorithmen zu erwarten sind, wenn aufgrund operationeller Erfahrung oder begleitender Tests neue Lebensdauerkonzepte erstellt werden oder wenn Veränderungen in Bauteil-Standards, Fertigungsverfahren oder Luftsystemen vorliegen. Um EMS-Updates mit minimalem Aufwand zu realisieren, wird bei MTU eine Prozeßkette entwickelt, die bei optimierten Schnittstellen alle Stufen von detaillierten Rechnungen bis zur automatischen Software-Generierung umfaßt. Dies wird am Beispiel des Updates der OLMOS-Software dargestellt.

Table of Contents

1. Introduction 6

2. How the EMS Works 6

3. Reasons for EMS Updates 8

4. Process Chain for EMS Updates 10
   4.1. Overview 10
   4.2. General Approach for an Element in the Process Chain 11
   4.3. Process Chain: Establishment of Reduced Physical Model 12
   4.4. Process Chain: Software Design 13
      4.4.1. Program Generation 13
      4.4.2. Run Time Optimization of the Software 15
      4.4.3. Software Testing 16
      4.4.4. Reference Program for Ground Based Processing 17

5. Summary 17

6. List of References 18

7. Figures 19

1. Introduction

Continuous monitoring by an on-board engine monitoring system (EMS) is an attractive approach for economically feasible and safe engine life usage monitoring. EMS updates must be considered if financial benefits can be expected from improved accuracy of monitoring algorithms, if experience from engine operation or from accompanying tests suggest the introduction of new lifing concepts, or if changes occur in build standards, manufacturing procedures or in the engine air system. For a minimum-effort development of an EMS update a process chain with optimized interfaces is being developed at MTU. It includes all stages from detailed calculations to automatic generation of the EMS software. As an example the update of the OLMOS software is presented.

2. How the EMS Works

An engine monitoring system (EMS) has a lot of tasks the most prominent of which are

- life usage monitoring of critical areas on fracture-critical parts
- incident monitoring
- trend data acquisition.

For the TORNADO aircraft of the GAF/GNy the on-board engine monitoring system OLMOS (On-Board Life Monitoring System) was put into service starting in 1987.

"On-Board" indicates that all calculations required to fulfill the different monitoring tasks are performed during the flight. As a result life consumption values and information about e.g. incident occurrences (type, frequency etc.) during the flight can immediately be obtained after the flight. The information is available on different accounts which can be downloaded on ground by the service personnel by means of Hand-Held-Terminals (HHT).

Data evaluation or further diagnostics can be performed at the OLMOS Ground Station (OGS) from which a data link transfers the data to the central logistics unit. Details about

the overall handling can be found in [1].

An overview about the way the on-board part of the OLMOS system works is given in Fig.1.

Measured data obtained by probes are transferred either via aircraft computers or directly to the Data Acquisition Unit (DAU). In the DAU the measured data are checked for plausibility (absolute values, rates of change). Substitute values are calculated if some data are considered as implausible. Finally, only checked and corrected data are transferred to the lifing functions (separately processed for each engine) where the calculations required for the different monitoring tasks are performed.

The life usage monitoring algorithms consist of the following steps which use checked input data and are executed for each module separately:

- Calculation of performance data (e.g. temperatures in the gas path, torques)
- Calculation of temperature distribution within the module
- Calculation of stresses at each critical area
- Application of a rainflow algorithm to the temperature-corrected stress history of each critical area. Thus stress cycles are obtained.
- Assessment of extracted cycles with regard to their damage

The result at the end of the flight is an accumulated damage value for each critical area. In the OLMOS system the damage values invariably are calculated as multiples of the damage under reference conditions. If the damage values pass some additional plausibility checks they are stored on accounts. Otherwise substitute values are calculated and stored together with a warning for the ground staff.

A more detailed description of the OLMOS system and the experiences gained so far can be found in [2,3]. A comprehensive description of the user's requirements for an EMS is given in [4].

3. Reasons for EMS Updates

As the overall lifetime of an engine is in the order of 30 years or more it is obvious that there will always be changes in the engine design due to several reasons, some of which will be discussed below. Changes in the engine design will mostly lead to a necessity of an EMS update, at least in the intermediate term. Having realized this fact it seems to be indispensable and of considerable financial importance to design the EMS from the very beginning to the demand of easy and low-cost updates.

Some reasons for EMS updates are listed in the following paragraphs .

The first group of reasons is due to changes in the engine design which may be caused by

- module re-design (e.g. for improved performance)
- changes in the engine air system which influence the thermal behavior of the modules (e.g. improved air seals, changed bleed air mass flows, changed aerodynamics of stators)
- experience from operational usage which indicates that the distribution in the life consumption within a module is different from the initially assumed values
- experience from operational usage and accompanying tests which indicate the necessity of considering additional damage mechanisms (e.g. crack propagation)

These points are essentially engine design related. An analysis of the changed situation will mostly lead to a re-analysis in some area of the engine lifing environment (e.g. performance calculations, design, life statements, lifing philosophy) which in turn must be accompanied by a respective adaptation of the EMS software. In some cases, an update of the EMS hardware may be recommended, too.

A completely different reason for an EMS update may occur from further developments in the monitoring algorithms which promise financial benefits from implementation. An example is the consequent introduction of transient temperature algorithms for the IP- and

HP-modules in the OLMOS system.

The temperature distribution versus time within a module is a process which is significantly influenced by transient situations. The assumption of stationary temperatures which only vary with some polynomial function of the spool speed is too crude for the HP modules and, to a lesser extent, for the IP modules, too. Therefore transient temperature algorithms were introduced where the current temperature distribution is a function of both current engine operating parameters (e.g. spool speed) and of the temperature distribution at the previous time step. Thus the transient behavior of the temperature distribution within a module can be modeled by far more accurate than with stationary temperatures only. This in turn influences thermal stresses which are a function of temperature gradients. There are critical areas where thermal stresses have a significant influence on the overall stress. In these cases an accurate model of the temperature development within the module versus time has a significant impact on the accuracy of the calculated stresses and, as a consequence, on the calculated values of the accumulated damage.

A simpler procedure, e.g. using stationary temperatures only, requires some conservatism as not to underestimate the thermal stresses. The result often is a significant overprediction of thermal stresses and, finally, of the damage during the flight. In this situation, introducing transient temperature algorithms increases flight safety and, normally, decreases the calculated damage values. This allows a longer usage of the individual modules which may have an immediate impact on the number of spare parts required.

Summarizing, it shall be stated that the introduction of more accurate algorithms increases flight safety and, in most cases, the time of usage. This coincides with financial benefits as fewer spare parts are required. If the costs for the EMS update (from development to introduction into service) are low enough, an EMS update just on the basis of more accurate algorithms may return significant pay-off.

In practice, both "external reasons" as engine re-design and "internal reasons" as development of more accurate algorithms occur simultaneously. Mostly an EMS update

is triggered by external reasons. More accurate algorithms underline the necessity and financial worthiness of the update.

4. Process Chain for EMS Updates

4.1. Overview

It has been shown in the previous section that EMS updates will invariably occur. For maximum customer satisfaction and minimum-effort for the update it seems to be obvious to take into account the occurrence of EMS updates at the earliest possible stage of the EMS design, ideally from the very beginning.

An understanding of the process stages that are involved in an EMS update (and in the initial EMS development, too) is a fundamental prerequisite for such a design.

The first step is a detailed temperature and stress analysis followed by a step establishing a reduced physical model of the module for those critical areas which shall finally be monitored. These steps can be summarized as "establishing the physical model" and are typically performed in the performance and stress departments. The latter also defines in detail the monitoring algorithms and provides the coefficients which are necessary for the software design.

The next step is done in the software department where the software requirements are converted into software which can be used directly on the processor of the monitoring unit.

Finally the HW/SW-integration takes place where the software is integrated into the on-board environment and is finally downloaded to the storage medium (e.g. EEPROM) that shall be used in the monitoring system.

An EMS software update may be accompanied by a hardware upgrade. This is the case

for the OLMOS where a RAM extension and additional EEPROMs with software loading function are incorporated into the already existing board. Fig.2 shows the upgraded board which is made by DORNIER.

As there are several steps and several departments involved in the complete process it is evident than an optimal data flow from the first to the last step is mandatory. To achieve this goal MTU is establishing a process chain comprising all these steps. The basic requirements and the MTU approach are described in the next subsection.

4.2. General Approach for an Element in the Process Chain

The basic requirements for the process chain are to maximize data flow, flexibility and user comfort and to minimize paper work, man induced errors, response time after changes and cost for EMS update.

A typical element in the process chain between two consecutive steps in shown in Fig.3.

Step i provides output data (e.g. stresses from Finite Element (FE) analysis). This output data is read by an interface program which is controlled by an easily understandable control file. The output of the interface program can be used by step i+1 as input data. By simply changing some values in the control file (e.g. selecting some additional critical areas from the detailed FE analysis) a new run of the interface program creates new input data which can immediately be used by the program in the next step.

Some effort has to be put into the design of the interface program (modularity, flexibility, user comfort). A clear concept increases user acceptance and helps to save a lot of money.

It shall be pointed out that a strict separation of the different steps in the whole process must be maintained. It is not useful to mix different steps or to interconnect them by e.g. tuning the program of step i+1 to the result of a certain program in step i. If a new program is added in step i (e.g. as an alternative to the already existing program, which

may happen with the introduction of a new FE package) the program of step i+1 has to be changed, too. This situation should be avoided as these "core programs" are designed to fulfill a certain task independently of the source of the input data.

The link between two consecutive steps shall always be done by an interface program. In some rare cases the interface program might be left out and the output of step i can be used directly as input for step i+1.

4.3. Process Chain: Establishment of Reduced Physical Model

The first step in the process chain is a detailed temperature and stress analysis, typically performed by FE analyses. Using the results of these detailed analyses potentially critical areas of the module are selected on the basis of temperature and stress histories during the design mission. Damage mechanisms are selected and the damage at the potentially critical areas is assessed with the aid of the material data base.

The output of the first step (FE analyses) are normally large files containing data about the behavior (temperature and stress) during a design mission for the complete module. Furthermore potentially critical areas have been selected with the aid of a post-processing routine for life assessment. Now the interface program extracts the required data of the potentially critical areas from the FE output files and generates the input data for the next step, the establishment of the reduced models.

As it is not possible to transfer the detailed FE model to the monitoring system, a reduced physical model for temperature and stress behavior at the potentially critical areas has to be established. This is done on the basis of the underlying physics by means of optimization procedures, where parameters of the reduced model are optimized in such a way that the behavior in the reduced model fits the original FE data in an optimal manner. The results of the optimization step are reduced models for temperature and stress behavior at the critical areas.

These reduced models are now applied to a set of mission data which can be either

tasks the aircraft has to fulfill. An analysis of these missions yields life consumption values for all potentially critical areas. On the basis of a comparison between the predicted life values and the life consumption values those critical areas are selected which are the most critical ones and shall be considered during on-board monitoring.

After the completion of the reduced models the next interface program uses the output and some control information to write all the required data (mainly coefficients for the different models) in a format that can be read by the software generator.

4.4. Process Chain: Software Design

The next step in the process chain is the conversion of the mathematical algorithms for the description of thermal and mechanical behavior of the monitored components into the on-board software and a functionally equivalent program for ground-based processing. The MTU approach to optimize this step and some problems that arise in this context (especially for an integer arithmetic processor as used in the OLMOS system) is discussed in the following section.

4.4.1. Program Generation

A first step in the conversion of the algorithms for the description of thermal and mechanical behavior into a program for integer arithmetics is the examination of the range of input data by analysis of many available flight recordings, engine specifications, test bed data and also of plans for engine modifications in the future. By using more or less sophisticated mathematics, interval analysis, debugging and tracing techniques for programs and, to tell the truth, sometimes also by trial and error all intermediate results have to be checked for sufficient accuracy and avoidance of overflow taking into account all reasonable combinations of input. This process is illustrated in Fig.4, showing the steps necessary for the determination of ranges of variables. It has to be guaranteed by the input signal checks and filters that only parameter combinations within the valid range are fed into the program representing the mathematical engine model.

After establishing a prototype program with checked and proven ranges for input and model parameters the development of a corresponding generator program has been found to facilitate future work. This program removes the tedious and error prone tasks of computing scaling factors, shift increments, writing down long tables or repeated control structures.

The ultimate goal of the MTU efforts is to support automatic performance of most of the intermediate steps in software development. The vision is an immediate automatic generation of source code for the on-wing software from the design documentation. Existing software engineering tools have significantly improved the documentation process, which is now also governed by comprehensive rules, e.g. [5]. However, the programmer's expertise in the conversion of mathematically demanding algorithms into efficient and safe programs cannot be totally replaced by CASE tools.

Of course there is a trade-off between the effort to create and maintain program generators and the effort of manual program changes. The greatest benefit can be obtained with using automatic program generators, if the structure of the mathematical model remains stable and only coefficients of the model are revised. Also to be considered is the reduced amount of required testing for automatically created programs.

A direct generation technique for program code has been selected for the RB199 Engine Life Consumption Monitoring Program (ELCMP, written in the ANSI C language) because processor load is considered as the major bottleneck for the updated ELCMP, whereas program size was considered as uncritical after the introduction of a memory upgrade.

The main advantages of direct program generation are the elimination of trivial operations on special operands (0, 1, etc.), a greater flexibility in the scaling of intermediate results leading to improved accuracy, an easy treatment of modified algorithms for single areas and a reduction of program control overhead.

Problems are the complexity of conversion programs, additional documentation, an increased probability of introducing program errors at places of manual intervention and the

creased probability of introducing program errors at places of manual intervention and the readability of machine generated code.

4.4.2. Run Time Optimization of the Software

A limitation of processor load and memory usage in a newly developed system is quite common in airborne computer systems. The resulting reserve gives the system a growth potential which may later be exploited for improving existing system functions or for introducing new functions.

During the lifetime of an airborne computer system the reserve capacity becomes smaller. If limits of processor capacity or memory cannot be circumvented by hardware changes, optimization of the software may push the limits somewhat further. Examples for areas of optimization are the elimination of time consuming control structures, inlining of short functions to reduce call overhead, the loop unrolling or reordering of instructions to allow the processor an optimal usage of its facilities (e.g. register usage).

In real time systems there is the additional constraint of fulfilling the requirements that the maximum time needed to perform certain functions must fit into some given time interval. The nature of the algorithms used to compute life consumption of engine components is basically different from algorithms used in control systems, where most of the tasks are repeated identically each time step. Due to the algorithms used for cycle extraction there is a high probability that more than one cycle is found for a critical area at the same time step leading to a very high peak processor load at single time steps. Balancing of processor load between time steps is now accomplished by algorithmic modifications of the cycle extraction procedure.

The last chance to reduce computing time is the conversion of time consuming functions into assembler code for the target processor. This should be kept to the minimum amount possible since portability and maintainability is negatively affected by using too much assembler code. In the newly developed algorithmic part of the RB199 ELCMP only a few assembler functions will be used, e.g. for some mathematical functions, for the computa-

tion of matrix elements for heat flux or for the formula for thermal stresses.

4.4.3. Software Testing

A set of test cases (test procedures, test data and methods for the evaluation of the test results) is provided to perform unit tests, integration tests and qualification tests. Typically these tests include a combination of synthetic data and flight-recorded data to cover all aspects of software function. Using a comprehensive and mature set of tests will reduce the time needed for software updates.

Tests can be improved by feedback of user experience, especially by provision of flight recordings of unusual flights. Examples of such flights are operations at limits of flight envelope, engine shutdown during flight, occurrence of failure incidents (surge or stall, sensor failures, malfunctions of the control system, mechanical defects of the engine, blade ruptures, bearing failures etc.)

All functions of the EMS have to be designed to cover the full possible operating range of the engine, not only the range currently used by the customer, including a reasonable margin to cover also known or probable failure conditions.

It is an MTU requirement for the lifing part of the EMS that all failures occurring with some reasonable probability will not lead to a breakdown of the life monitoring algorithms. There is, however, a trade-off between safety against spurious input failures under normal conditions and the continuation of life usage monitoring in cases of severe engine malfunctions.

It is MTU policy to set check limits for signals influencing life usage monitoring comparatively narrow, because failure conditions inhibiting the correct accumulation of life usage are nearly always a strong hint on necessary maintenance actions. The benefit gained from early information on incipient failures in the engine or its sensors justifies the additional effort to perform a substitute calculation for flights with muted lifing results.

4.4.4. Reference Program for Ground Based Processing

The development of the monitoring software for the airborne system is accompanied by the development of a functionally equivalent program for the processing of flight-recorded data. This program can either be used for continuous processing of all flights of an aircraft fleet (or of selected aircraft) or to process recorded flights from special purpose campaigns.

This program has options which give access to information normally not available in the on-wing system (e.g. plotting of input data, output of all cycles contributing to damage of selected critical areas, performing statistics).

There are several possible strategies to develop such a program for ground based processing, each of which has some advantages and disadvantages. For the OLMOS system a strategy was chosen where the on-board software is used as "kernel" with a customized user interface (input/output, flight data formats). Therefore only one version of the algorithms is required (simple configuration control) and identical results for on-board and off-board processing are obtained.

5. Summary

As the necessity for an EMS update naturally arises during the engine lifetime, is is reasonable to account for this fact by developing a "modification-tolerant" environment for EMS updates.

In the MTU approach a process chain is being developed where the different steps in the modification process from detailed FE analyses via establishing a reduced model to automatic software generation is covered. The aim of the process chain is to minimize the effort (and therefore the costs) for an EMS update by using standardized procedures, clearly defined interfaces and, during the last step, a software generator program which reads output from previous steps and generates the on-board software automatically. This system has been used in the current OLMOS update and shall be developed further to

achieve even shorter response time and lower costs which are vital for customer satisfaction.

6. List of References

[1] Schulz, U. Bott, J. Integrated Engine Control and Monitoring with Experiences Derived from OLMOS. Proceedings of the 16th AIMS Symposium Munich, pp.287-310, 1991

[2] Broede, J. Pfoertner,H. Advanced Algorithm Design and Implementation in On-Board Microprocessor Systems for Engine Life Usage Monitoring. Proceedings of the 15th AIMS Symposium Aachen, pp. 241-260, 1989

[3] Broede, J. Design and service experience of engine life usage monitoring systems. Proceedings of the 5th European Propulsion Forum, EPF-95-25, Pisa 1995

[4] Barnes,O.R. Engine health and usage monitoring. A military viewpoint. Proceedings of the 5th European Propulsion Forum, Keynote address, Pisa 1995

[5] Military Standard MIL-STD-498: Software development and documentation, US DoD, 1994

Acknowledgement

The development of the OLMOS system and the RB199 ELCMP software is financed by the German Ministry of Defence.

7. Figures

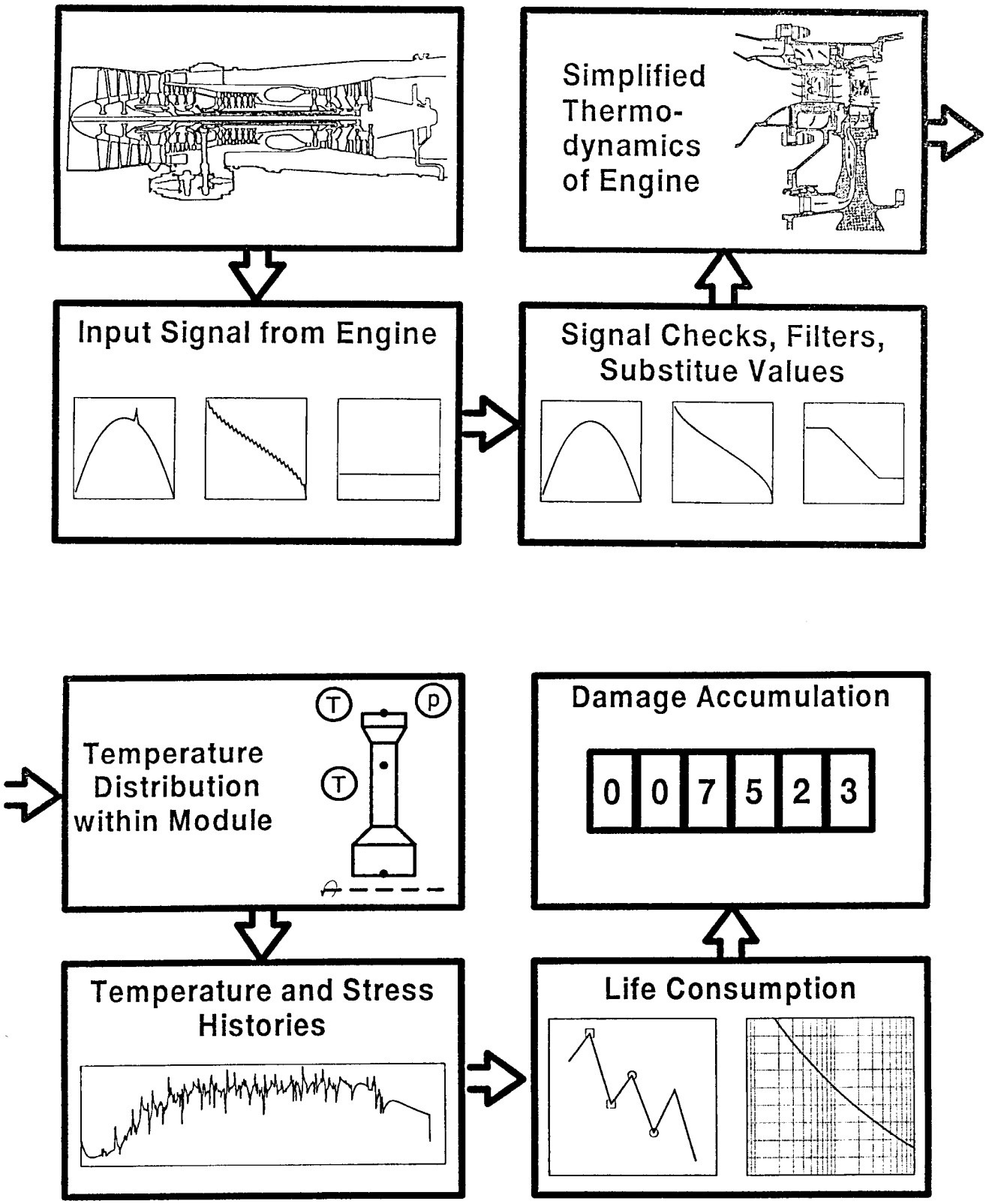

Fig.1 Overview about life usage monitoring in OLMOS

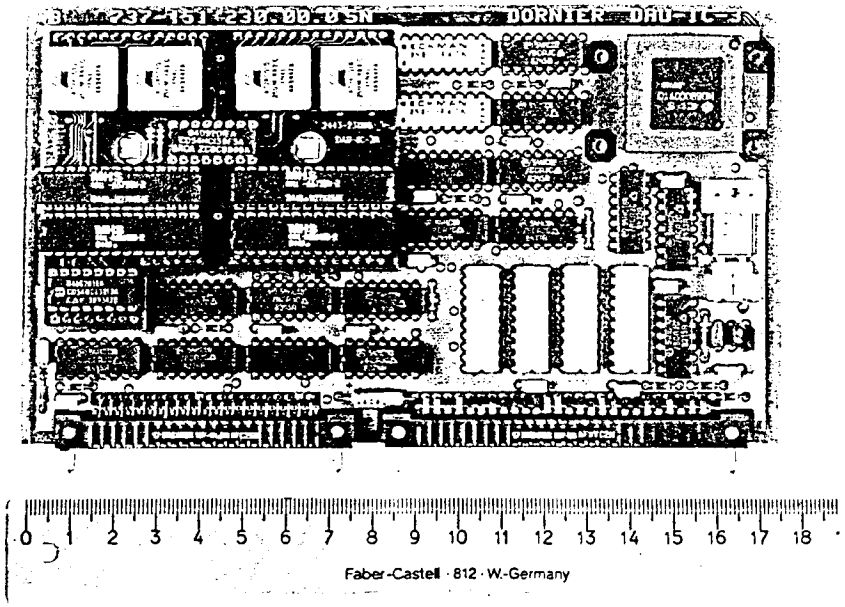

Fig.2 Upgraded board

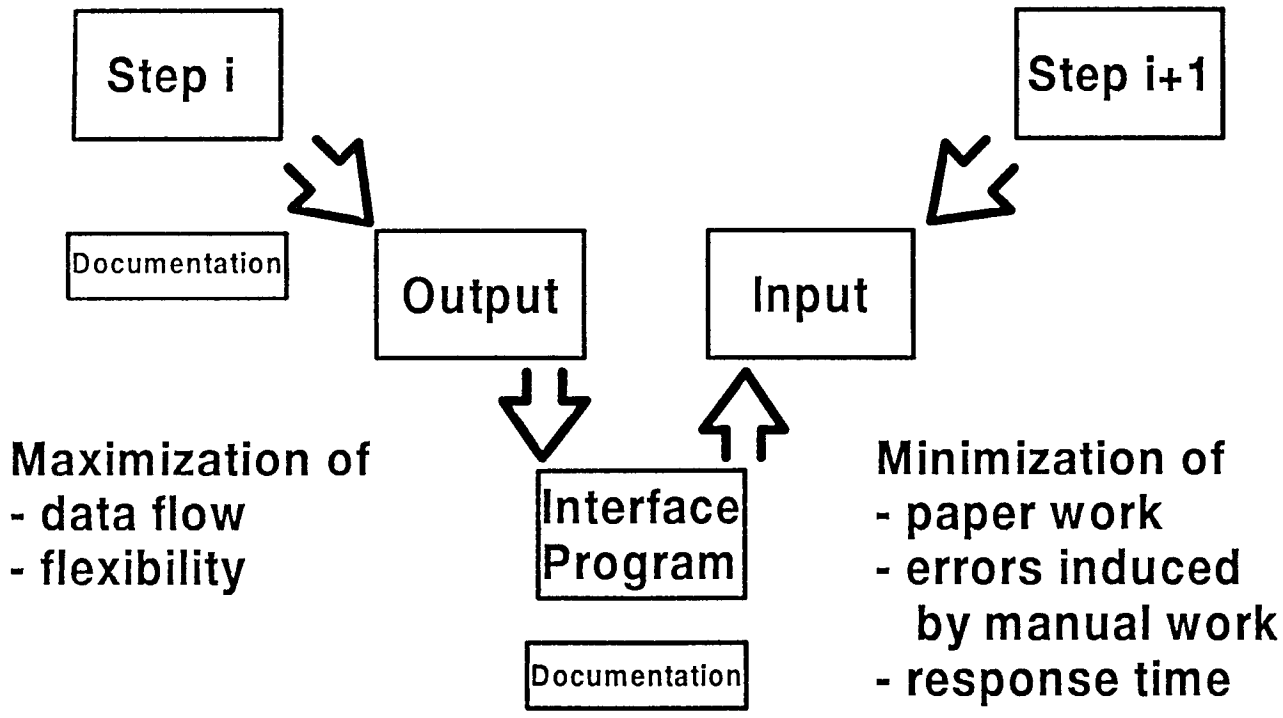

Fig.3 Element of process chain

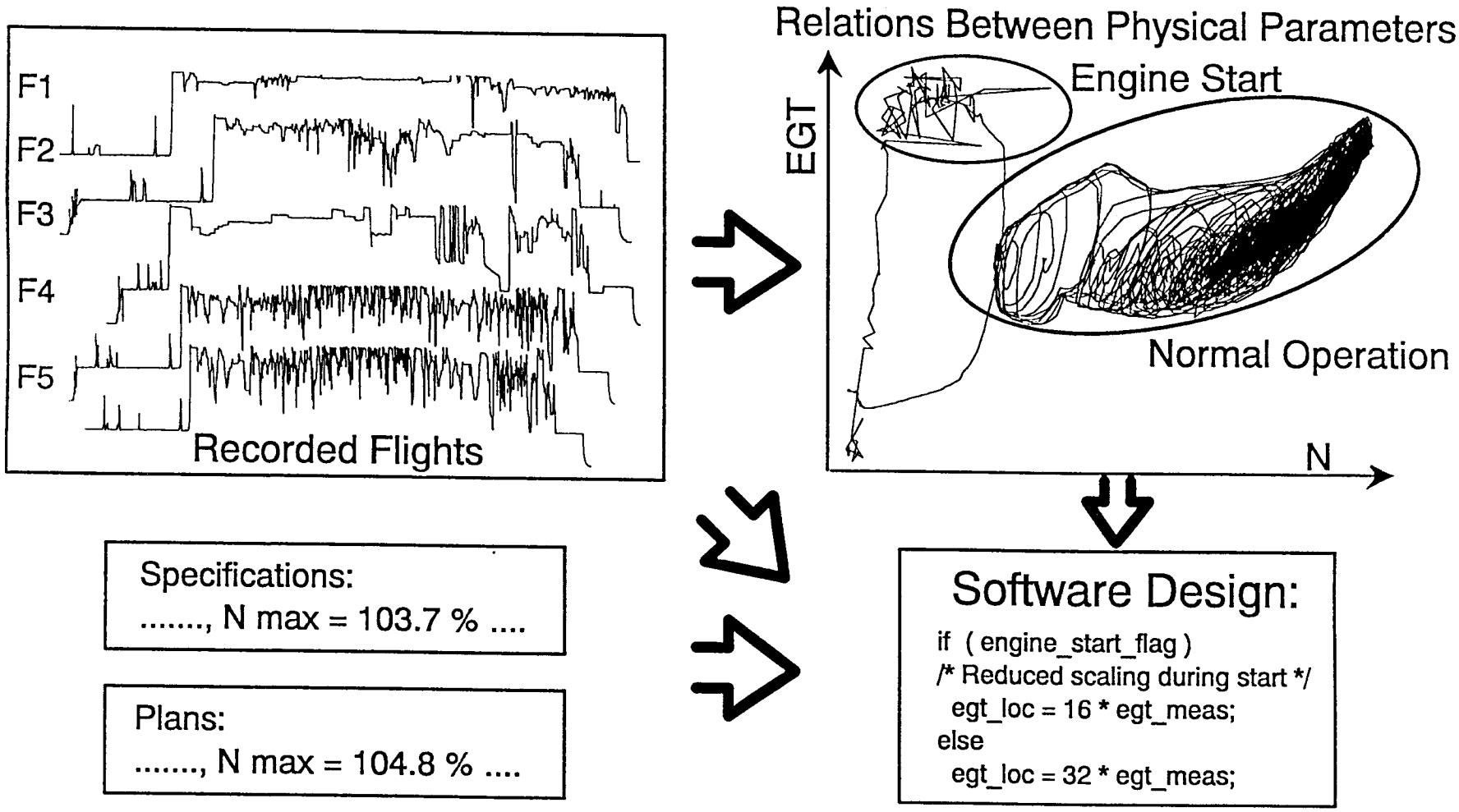

Fig.4 Steps for determination of ranges of variables