A DATA TREATMENT TOOL FOR A SATELLITE THRUSTER TEST FACILITY

Geraldo L. da S. Ribeiro
geraldo-ribeiro@uol.com.br
D. Bastos-Netto
demetrio@lcp.inpe.br
National Institute for Space Research - INPE, Cachoeira Paulista, Sao Paulo, Brazil

This work introduces and discusses CATPA (Computer Aided Thruster Performance Analysis), a user friendly data treatment software specially developed for INPE's Satellite Thruster Test Facility. This facility is a ground-based structure that can emulate the space environment for testing and qualifying satellite thrusters and satellite propulsive loops. A structure like this requires the prompt availability of data treatment tools which must be extensible to encapsulate the particular needs of the customers. The original data treatment package supplied by the test bench development team required some effort to customize the available features. However several constraints were imposed on the overall system such as: a) support for low amount of data; b) large time slices were required for processing the acquired data; c) poor final output quality; d) specific hardware requirements; e) non portability; f) specific language used for coding the scripts; etc. The CATPA came up to override these constraints and to make data treatment easier and much more powerful. It uses modern concepts like design patterns and standard template library, making the reuse of code and the rapid application development easier to be reached.

Introduction

The BTSA[1] (INPE's Satellite Thruster Test Facility with Altitude Simulation) was delivered with a toolset of analysis scripts. This work started out with the intention of using the supplied scripts to treat the acquired measurements. However, after some testing it was concluded that these scripts were not fast enough nor flexible enough for existing needs. This led to the coding of a new treatment tool called CATPA.

This Facility depends on several kinds of equipments for its control, command, data acquisition and data treatment. This work will explain only the data treatment procedures.

Data Acquisition System

The data acquisition system is composed by two workstations HP-9000 running the Scopix 5.1 software coupled with a set of acquisition cards and signal conditioners. This system can reach a scan-rate about 50 kHz in its faster measure channels. This high scan-rate generates large files which are very difficult to treat.The Scopix software is a data acquisition tool with real time visualization capabilities. The applicability of this software tool at test benches was due to its convenient features. Unfortunately the data stored into Scopix disk must be transfered to another machine for external treatment.

Data Transfer Technique

The first problem to be solved was: How to transfer the acquired data to another machine?

Step number one was to define *"another machine"*. It was chosen the other machine to be a host which runs under an unknown operating system and a well specified database engine.

Step number two was to define *"what to transfer"*. To solve this step a complete and well substantiated know-how about the Scopix files format were needed. Once the operator possesses the informations above he is able to code a transfer driver.

To decompose and absorb all features of the Scopix configuration files and data files several tests were made. These tests consisted of a sequence of data acquisition procedures for ranging all kinds of measures to which the Bench can be exposed at. A complete set of validating tests were confronted with the prototypes of the importing drive until the full compatibility was reached.

The importing driver is divided in two parts (as shown in figure): a Scopix reliant part and a database engine reliant part. To supply the first part it was coded a `C++` class called `ScopixFireTest`. This `C++` class handles all features of a Scopix fire test. The `ScopixFireTest C++` class works as an interface element between the Scopix files and the importer routine. This `C++` class can be fully reused for coding importing routines from other database engines without any modification. The second part of the transfer driver inherits `ScopixFireTest` capabilities adding

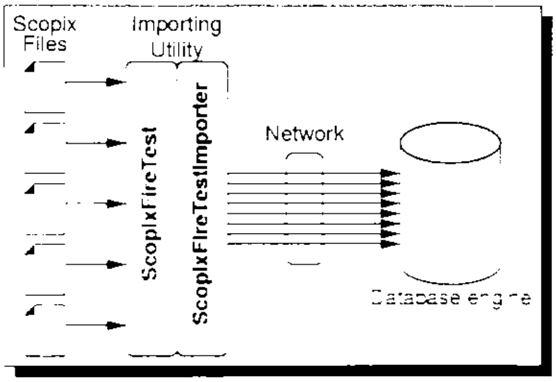

Figure 1 The importing scheme.

The import function

The import function reads the information from the ScopixFireTest C++ class and puts it into the database engine using its application programmable interface (API). The import function was coded using a specific database engine set of commands. This approach is needed because the import function demands a strong number of database engine calls. The change of the database engine implicates in the re-coding of the import function. This re-coding is needed to keep up the *high performance* of the application. This approach was chosen due to the strong loop needed to import all the measurements records. But all other database engine calls are made using a generic set of SQL commands thus keeping the code portability.

The importing driver was implemented with command line behavior to be used in batch processing mode. Its usage is shown at figure . . If the verbose option was used an NCURSES interface is shown displaying the operation performed and its progress.

Taking the import routine external to the CATPA main application several data sources can be used without re-coding. This approach allows the operator to use different data sources by changing the import driver only.

The CATPA main module has a push button which checks for non imported tests and launches the importer utility for each one. For machines with more memory and high speed processors, the CATPA can launch several processes at the same time performing

```
$ catpaScopixFireTestImporter

Usage: catpaScopixFireTestImporter [options]

where [options] are:

  -c --campaign      Campaign name
  -t --test          Test name
  -u --user          Username
  -p --password      Password
  -v --verbose       Verbose mode
  -h --help          Print this help
```

Figure 2 The catpaScopixFireTestImporter command usage.

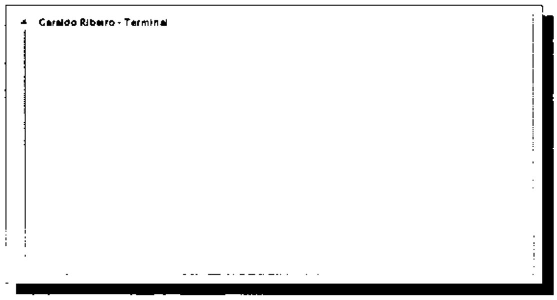

Figure 3 The catpaScopixFireTestImport verbose window.

a parallel mode import strongly increasing the performance.

Development architecture

The development architecture is an AMD K6-II 300 MHz machine with Debian GNU/Linux 3.0 operating system running a MySQL database engine version 4.0. This architecture was chosen because no hardware restriction could be imposed on the final user. Therefore the cost of this new system is a fraction of the replaced system with much more speed and flexibility.

The MySQL database engine was chosen because it has a well-deserved reputation for being a very fast database server that is also quite easy to set up and use. The current release of the CATPA suite only supports the MySQL database engine. But it can be easily extended to a large amount of database engines.

CATPA is developed under Linux, but it is set-up to be highly portable. As a result, it runs on most other Unix flavors as well. For graphical user interface it was used a multi-platform C++ GUI application framework called Qt. It provides application developers with all the functionality needed to build applications with state-of-the-art graphical user interfaces. Qt is fully object-oriented, easily extensible, and allows true component programming.

For a better use of the available hardware on the test facility the CATPA software was written to be **modular**. This feature enables it to get raw data from the Scopix workstation at Host A and to put it into a database server located at Host B. The CATPA main application (located at Host C) opens a data stream between hosts B and C, performs the calculation and puts the results in another machine (Host D). The opposite approach is also possible, i.e., the concentration of all operations in a single machine.

Hard disk usage

CATPA has no data size constraint, extending the old one Giga-byte limitation to several Tera-bytes of data files. The old application used the same file for handling all fire tests. The opposite approach is employed by the CATPA enabling the developer to use up to date technologies to improve the speed analysis.

The use of one separated table (file) for each acquisition rate group provides a lot of benefits for tasks such as incremental backups, data exchange and data conversion which are made much simpler.

User Interface

[illegible]

[illegible] as shown at figure [illegible], consists basically of a log area (LogWindow C++ class) and a toolbar (MasterToolbar C++ class). The LogWindow C++ class handles the output messages by scrolling its history. The LogWindow C++ class has a lot of useful capabilities like coloring the messages by its status, easy retrieval of all messages on the database, status icon and non storable messages support (useful for progress status, intermediate step calculation, etc). The logMessages table holds all messages generated by the CATPA, enabling the user to inspect the action history. All stored messages can be accessed at any time by scrolling a slider on the CATPA interface.

Each message is composed by:

- uniq id;
- description text;
- date;
- time;
- level;
- user name.

These informations can be retrieved by using level filters, e.g., by selecting the warning messages only.

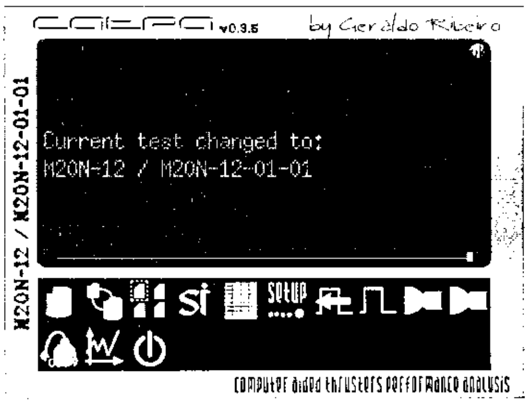

Figure 4 The CATPA main window.

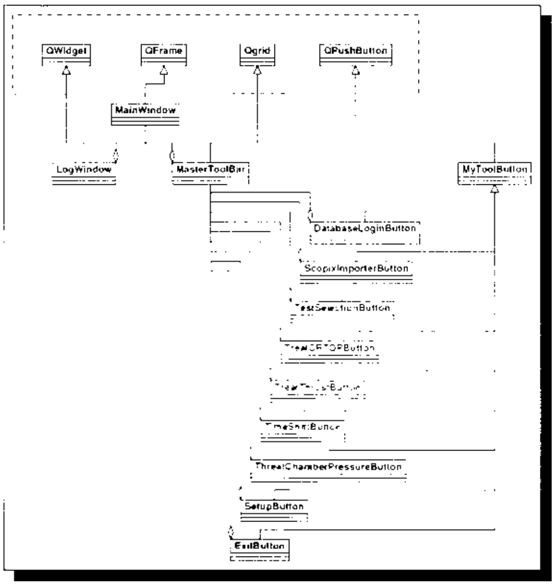

Figure 5 Partial C++ class diagram of CATPA.

New treatments

All new (customized) treatment routines must be placed into the MasterToolBar by inheriting MyToolButton C++ class. The CATPA supplies a C++ class called TreatDefaultButton which is a template for new treatment routines. Figure shows the tool buttons relationship. The C++ class MyToolButton provides a complete set of check routines, conditional enable/disable, independent time lines, communication with the LogWindow C++ class, and so on.

The output "factory"

This application supplies templates for coding and adding new features. The report output format is controlled by a factory design pattern allowing the developer to code drivers for any other formats. The C++ class Report handles the CATPA output features. This class chooses the output format by reading a resource file, then creates an object which inherits the C++ class OutputFormat capabilities. The creation of a new output format is very easy. The user must derive his C++ class from the OutputFormat and overload its member functions. The C++ class Report is implemented as a Singleton[2] keeping just one instance loaded in memory.

Each tests campaign has an independent output directory enhancing the file system organization.

Default data treatment

The standard data treatment is composed by:

- SI conversion;
- Time adjustment;
- Pulse description;
- Mean thrust calculation;

- Mean combustion chamber pressure calculation;
- Ignition time delay calculation;
- Data reduction and data export.

SI conversion

All kinds of reports and exported data generated by CATPA are delivered under SI units. The CATPA supplies an smart SI conversion utility which enables the user to adjust all measurements to the International System of Units automatically.

The unit of the measurements which are acquired by using the Scopix software depends on the Scopix configuration only. However this setting represents particular user needs (e.g., pressure acquired in bar). The CATPA has conversion factors for several common ASCII representations of the unit such as "`cm^3/s`", "`bar`", "`°C`", "`C`", etc. If the Scopix user sets the unit of a measure to a non usual string (e.g. "`Celsius`", "cc/s") the CATPA will ask for a conversion expression which must be placed into the CATPA resource file.

Time adjustment

At the beginning of the firing test two delays occur: a) the acquisition system pre-trigger; and b) the control-command pre-trigger. These delays are needed only to supply the starting conditions. But its time slice must be suppressed for the correct data analysis to take place.

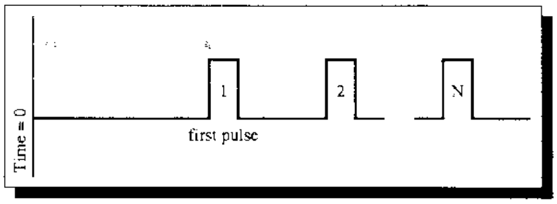

Figure 6 Time adjustment scheme.

By using the PLC (Programmable Logic Controller) command signal, the CATPA searches for the starting of the first pulse and then it decrements the time scale of all measurements. This decrement puts the real test start at time zero. A scheme for the time adjustment procedure is shown at figure .

Pulse boundaries

The most important treatment step is to locate the pulses boundaries. Figure displays a schematic pulse boundaries description. These limits are needed to correctly select the sampling region.

All analysis uses the boundaries found at this step to get the measurement values for the calculations.

Analysis span

For short pulses, some assumptions are made: The solenoid valve starts its opening process upon receiving the ON signal from the PLC within a very short time delay (circa 1 ms). The solenoid valve is assumed to

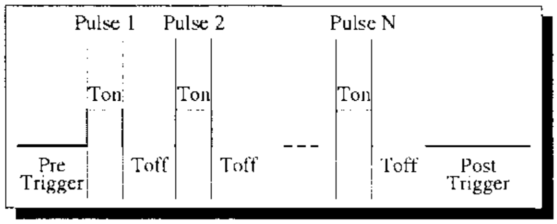

Figure 7 Pulse boundaries.

be fully opened with *70%* of the total allowed current. In this case the best region for the data analysis is the second half of the pulse, as shown schematically in figure . The viability of this choice can be observed in figure which displays an actual example.

The way to specify the analysis span is to choose the second half of ON-mode of the pulse, so that a quasi steady state can be assumed for calculation purposes. In the case of long pulses the analysis span is limited by a resource variable called `Treat/AnalysisSpan` (three second duration by default).

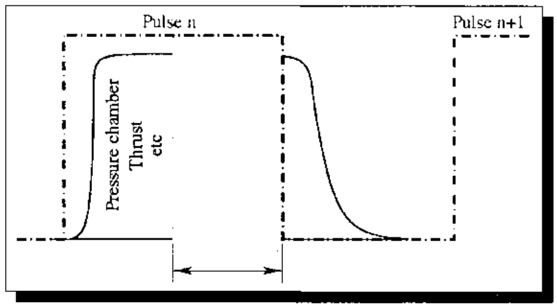

Figure 8 Analysis span.

Mean values

The mean values of thrust, chamber pressure, propellant flow rate, ..., used for calculation of several performance parameters, are estimated by using data sampled from the analysis span, as shown in figure . The user can change this range by changing the `Treat/AnalysisSpan` resource variable.

Ignition time delay

Figure shows the PLC command, the propellant solenoid valve current and the combustion chamber

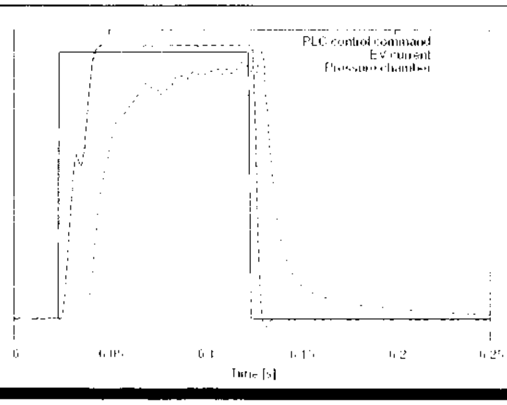

Figure 9 Ignition time delay.

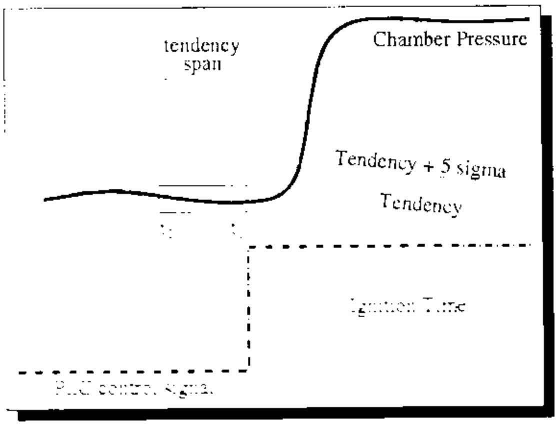

Figure 10 Ignition time calculation scheme.

pressure behavior. Notice the delay between the starting of the PLC pulse and the starting of the pressure pulse.

The ignition time delay is considered to be the time interval between the PLC starting command and a sensible change in the combustion chamber pressure. To estimate the ignition time a tendency curve is required. This tendency curve is calculated using the sampled data between t_0 and t_1 (figure). t_1 is the starting time of the PLC pulse. The t_0 value is given by $t_0 = t_1 - t_{ts}$, where t_{ts} is the value of the `Treat/TendencySpan` variable.

Data reduction/export

The main limitation in using common commercial spread sheets like Microsoft Excel are the number of records. At the BTSA-INPE some measurements can be acquired in a rate about 50 kHz. For a short duration test (less than 5 second firing time) the acquisition needs a pre-trigger (about 6 seconds) and a post-trigger (about 1 second) resulting in 600,000 records for just one measure. The limit of standard MS Excel is 32,768 rows.

CATPA can handle a large amount of data and export it through a user arbitrary step. The calculation of each new point is made by using an average over the point neighborhood. Figure shows the scheme of calculation of exported data.

The output of the export routine is in ASCII format for full compatibility with all kinds of operating systems and data manipulators. The CATPA exporting format does not use tabs or any other non printable characters. Depending on the customer needs other exporting formats can be written, e.g. CSV (Comma Separated Values), XML (Extensible Markup Language), and so on.

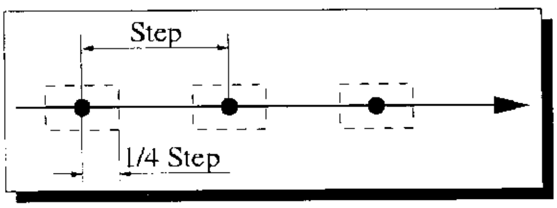

Figure 11 Data reduction scheme.

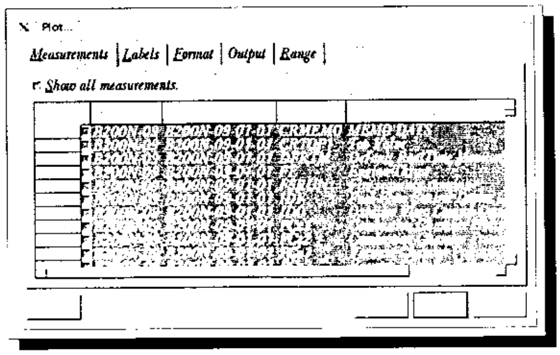

Figure 12 Main plotting screen.

Plotting

The CATPA has an enhanced plot utility which enable the user to compare measurements from different campaigns. This task is made by enabling the "Show all measurements" check box in the main plotting window (figure).

The current version of CATPA enables the user to create plotting files in the following formats:

- EPS (Enhanced postscript);
- HPGL (Hewlett-Packard graphic language);
- PNG (Portable network graphic).

CATPA has a smart file naming which uses the name of the measurements to suggest the name of the plotting file. This feature prevents the replacing of old files and generates a more comprehensible file list. Each campaign has an independent output directory enhancing the file system organization.

The plotting utility possesses an automatic ranging which fits the curve on the sheet. The label of each curve is set by the full name of the measurement, e.g. "Combustion chamber pressure" in contrast with the old application which printed only "Pc".

Thrust data adjustment

The load cell calibration procedure requires several conditions:

- the calibration process is done at atmospheric pressure (vacuum chamber opened to ambient);
- the propellant line pressure is kept low (about 200 kPa) during the calibration process for personnel safety;
- all vacuum pumps are stopped to reduce the vibration level;
- the load cell is aligned and pre-loaded until the minimum hysteresis is reached;

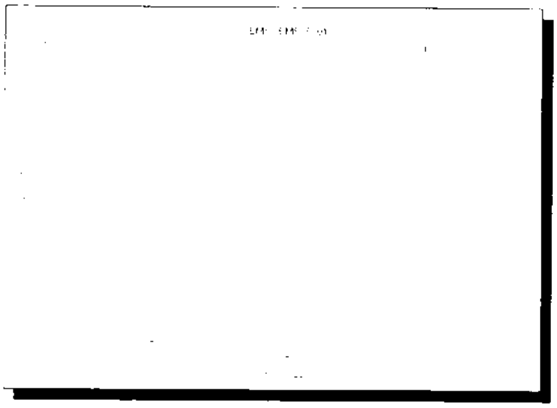

Figure 13 Initial thrust shift.

Therefore when all equipments (boiler, ejectors, vacuum pumps, etc) are running the thrust measurement displays a small shift as shown in figure . CATPA can detect and remove this thrust shift.

Chamber pressure adjustment

The CATPA has a smart correction routine which minimizes the drift effect on measurements. After running this more accurate values of the combustion chamber pressure can be achieved. Figure presents a zoom of the combustion chamber pressure measurement, showing the drift effect. This measurement was performed with a piezoelectric pressure transducer Kistler type 7001 coupled with a charge amplifier Kistler type 5007 sensor.

Drift[3] is defined as an undesirable change in output signal over time which is not a function of the measured variable.

Piezoelectric measuring systems are active electrical systems. That is, the crystals generate an electrical output only when they experience a change in load. For this reason, they cannot perform true static measurements. However, it is a misconception that piezoelectric instruments are suitable for dynamic measurements only.[4] Quartz transducers, paired with adequate signal conditioners, offer excellent quasi static measuring capabilities. There are countless examples of applications where quartz based transducers accurately and reliably measure quasi static phenomena for minutes and even hours.

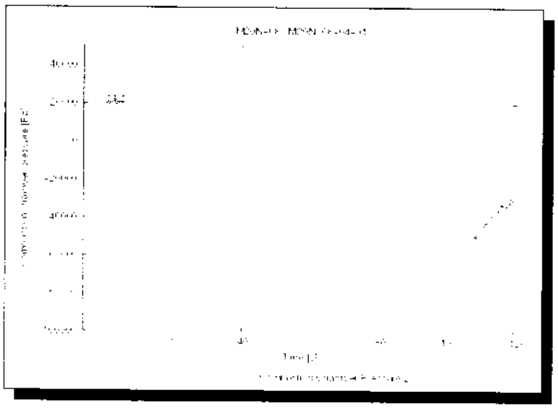

Figure 14 Measurement with drift.

Conclusions

CATPA is a tool to assist the user in his analysis task. Several trials with the CATPA show its viability. Table compares CATPA with its predecessor.

The old application prints graphics with solid lines in monochrome mode only. This feature disables the measurement identification in multi curves plots. On the other side, CATPA saves the graphic files in several formats and supports color and line styles. The files created with CATPA can be directly imported no scanning is needed by almost all current softwares thus reducing the report generation time.

The CATPA may run in low cost personal computers with Linux. The old application runs in HP-UX system only.

The CATPA exporting data format can be customized and the data can be reduced for use with external analysis routines.

Table 1 Comparing CATPA with the old treatment tool.

Old application	CATPA
Graphic printing only	Saves the graph files on several formats.
Monochrome graphs	Color graphs
Hardware dependent	Runs on almost any hardware architecture
Uses private database engine	Uses the most widely used database engine
Low speed due to encoding format (data size optimization)	High speed (no convertion is needed, i.e., a floating point number is stored as a floating point number)
Specific data format exporting	Customizable data format exporting
No data reduction support	Data reduction support
No automated routines	Treatment can be applied over each pulse

References

[1]Yribarren, J. P., Bastos-Netto, D., Pinard, J., Salles, C. E. R., and Paille, J. L., "INPE-SNECMA Cooperation in the Field of Satellite Propulsion," *51st International Astronautical Congress*, 2000.

[2]Câmara, G., Vinhas, L., de Souza, R. C. M., Paiva, J. A., Monteiro, A. M. V., de Carvalho, M. T., and Raoult, B., "The Design Patterns in GIS Development: The Terralib Terralib Experience," *III Workshop Brasileiro de Geoinformática*, 2001.

[3]Kistler Instruments Corporation, *Drift of charge amplifier -- cause and remedy*.

[4]Kistler Instruments Corporation, *The Piezoelectric Effect, Theory, Design and Usage*.

[5] Zawodny, J. D., "How to Write Efficient MySQL Applications," *Linux Magazine*, 06 2001.

[6] [illegible], S., "Installing, Configuring, and Using MySQL," *Linux Magazine*, 03 2001.

[7] [illegible] E. M., da Silva Ribeiro, G. L., de Freitas Mu[illegible] W. [illegible] Bastos-Netto, D., "Case Study of Satellite Engine [illegible] Behavior Under Simulated Altitude Working Conditions," XXII [illegible] *Brasileiro de Aplicações* [illegible]

[8] [illegible]

[9] [illegible] Addison-Wesley Pub. Co., 2nd ed., 1997.

[10] Stroustrup, B., [illegible] Addison-Wesley Pub. Co. 1997.