Multi-Sensor Process Tomography System Design: Part 2 – Data Processing and Systems Software Design

B S Hoyle[1], X Jia[3], H McCann[2] , F J W Podd[1], H I Schlaberg[1], R M West[3] and R A Williams[3]

1. Institute of Integrated Information Systems, School of Electronic and Electrical Engineering, University of Leeds, Leeds, LS2 9JT, UK,
email: b.s.hoyle@leeds.ac.uk, f.j.w.podd@leeds.ac.uk, h.i.schlaberg@leeds.ac.uk
2. Department of Electrical Engineering and Electronics, UMIST, Manchester, M60 1QD, UK, email: t.a.york@umist.ac.uk
3. Particle and Colloid Engineering Group, School of Process, Environmental and Materials Engineering, University of Leeds, Leeds, LS2 9JT, UK,
email: x.jia@leeds.ac.uk, r.m.west@leeds.ac.uk, r.a.williams@leeds.ac.uk.

All authors are members of the Virtual Centre for Industrial Process Tomography

Abstract *- This paper deals with the second part of the systems engineering of a multi-modal process tomography system, with a primary focus on the data processing needs and software solutions in this design. It deals in particular with the conceptual design of an integrated process tomography data acquisition system, which supports a multi-modal system offering a 'plug and play' capability. It deals with the design constraints of the software system: the need to support differing front-end sensors for each modality in a generic fashion; the need to communicate with front end data pre-processing and packing systems; the need to integrate the data to allow data fusion; and finally to enable successful interpretation.*

Keywords: multi-modality tomography, data fusion

1. SYSTEM OVERVIEW

This is the second paper which deals with the various aspects of the specification and design of a multi-sensor and multi-modal process tomography system. The first paper [1] provides a detailed overview of the hardware options and design choices for the system, including the various programmable modules.

The design concept aims to embrace a spectrum of possibilities: from a flexible system designed for experimentation, to an embedded industrial system tuned to a specific application.

The experimental trial prototype constructed by the joint authors has demonstrated the concept. This second paper deals with the data processing and software components.

2. SOFTWARE TOOLSUITE CONCEPT

Just as the hardware sub-systems benefit from careful modular design, in which components are based wherever possible on existing standards, the software units are easier to construct and maintain if they use common and reliable tools and methods.

The core aim of the data processing and software modules, in line with the requirements of the overall system, is to facilitate the deployment of multiple sensors, and allow their individual data to be collected efficiently, combined effectively, and interpreted to suit a monitoring or control need.

The various services, particularly in the system designed for experimental prototype use, offer a *software toolsuite*, in which data streams and processes may be configured to suit an experimental requirement.

2.1 Logical toolsuite elements

The toolsuite relies upon two key logical components:

- Data Layers: to provide a standard link between processing elements, and allow other external applications to display or post-process the data at that point in the processing stream.
- Processing programs: to perform a specific processing operation upon the data stream at that point.

2.2 Software development standards

For simplicity and portability the core software uses some of Posix extensions to the C language.

For the design of graphical user interface features, which can simply build configuration files for use by real-time modules, C++ and Java are used in conjunction with graphical class libraries.

2.3 Physical data and process links

The basic standards defined so far intentionally make no reference to the detailed form in which the data is passed between processes. This is to permit flexibility in the detailed design.

The system designed for experimental use takes a particular approach that allows further flexibility. This facilitates the channels between process as inter-process *pipes*, allowing the underlying operating system to deal with their efficient implementation. This re-directs the standard input and output of the two executable modules to the process pipe. This allows users to add software modules without their need to understand the underlying communication complexities.

A further feature which allows extensive processing power is the connection of the processing modules via logical stream *sockets*. Such sockets can link processes within a given processor, but interestingly can be implemented via a network. In general this is realised with appropriate system drivers using the TCP/IP protocol over a network using an Ethernet communications link. This allows separate processors to run in parallel where appropriate.

Where single processes can be segmented into independent units, these units can be executed in parallel as separate threads to enhance processing efficiency.

Table 1 below provides a comparative checklist of these possible methods and their features. Comms. speed provides a qualitative indication of the data transfer rate of the communications channel.

In order to obtain some level of real-time functionality and also to fine tune the software it is useful to use an operating system that provides various priority levels. This enables the various software processes to run at different rates. Both the Unix operating system and Microsoft NT have this feature: Unix has seven priority levels available and NT has three.

	Sockets	**Pipes**	**Threads**
Platform	Unix/NT mix over Ethernet	Unix or NT	Unix or NT
Scalability	multiple processors multiple computers	multiple processors	multiple processors
Method	connected executables	connected executables	all modules within one program
Data transfer speed	slowest	medium	fastest
Advantages	allows multiple platforms	communicat-ion with binary progs.	fast for multiple processors
Dis-advantages	comms overhead, limited number	difficult to create a web of modules	difficult to support one large program

Table 1: Process communication/control options

3. PROCESSING IMPLEMENTATION

The toolsuite software is compartmentalised into various sub-systems or layers. The following sections provide further complementary details of the data layers and programs that form the toolsuite.

3.1 Sensor data communications

For simplicity the software toolsuite should communicate with each sensor-head in a standard way irrespective of the modality of the sensor

Some commands are common to all sensor types (such as interrogation of what device is connected to particular link), while others will have to be special to a particular S-S.

An example of typical generic and sensor-specific commands are given below.

Generic commands examples

- **1. Get Status** (Identify attached device)
  1: System Type: (1 byte)
  most significant 4 bits define type: 1-ECT, 2-ERT, 3-US
  least significant 4 bits define version: 0-15
  2: Sensing planes: 0-64K (2 bytes)
  3: Sensors per plane: 0-64K (2 bytes)
  4: Measurement resolution (number of bits): 0-255 (1 byte)
  5: Software Date (when the S-S software was last updated): dd-mm-yyyy (4 bytes)

- **2. Get Single Measurement** (Only perform measurement on selected sensors and return value)
  ECT, ERT system (Arguments: Sensor 1, Sensor 2)
  1: Measured value (1-2bytes)
  US system (Arguments: Emitter)
  1: No. of data points (2 bytes)
  Data points in following format:
  a: Emitter (1 byte)
  b: Receiver (1 byte)
  c: Time of flight (1-2 bytes depending on defined resolution)

Specific commands for e.g. the ultrasound modality

- **21 Send Statistics** (Return values of how many echoes each transducer has received and generated)
  1: Reflected echoes received (No of sensors x 1-2 byte)
  2: Reflected echoes generated (No.of sensors x 1-2 byte)
  3: Transmitted echoes received (No.of sensors x 1-2 byte)
  4: Transmitted echoes generated (No.of sensors x 1-2 byte)

- **22 Fire** (Trigger one transducer without initiating the acquisition process)
  (Arguments: Transducer No.)

Further commands categories are derived to control the special functions of other sensor subsystems types. Additional commands for the ECT subsystem begin at index 40 and others specific to the ERT sensor subsystem commence at 60.

This interface in effect implements the 'plug and play' sensor feature described in the first paper. Each S-S is connected to the DA-S via the SC-S and is interrogated on initialisation. A typical (but abbreviated) status information return is:

```
TFC-VCIPT-1.0
CONTACT = F.J.W.Podd@Leeds.ac.uk
#SENSOR_INTERFACE
 DATA_EXCHANGE_VERSION = 0
 SENSOR_MODALITY = UST
 HARDWARE_VERSION = 0
 SOFTWARE_VERSION = 0
 SYSTEM_ID = 1
 COPYRIGHT = Virtual PT Group
 MANUFACTURER= VCIPT Systems Group
 NUMBER_OF_PLANES = 1
 NUMBER_OF_SENSORS_PER_PLANE = 16
#END
```

3.2 Pre-processing and reconstruction modules

There are many types of pre-processing modules, most of them modality dependent. For example an algorithm used for the ultrasound modality sieves the data, removing unwanted (out of bounds) echoes.

Simple but flexible reconstruction algorithms are provided for ERT, ECT and UST modalities. These algorithms are based on the back-projection method. More accurate, but less adaptable, model based algorithms are used for specific applications.

3.4 Raw images and data fusion

Off-line investigation of image-based data using other standard tools is made permissible by providing image format conversion routines for standard formats, eg AVS. Where possible the routine employs a header to identify the experimental test, the time link in the test and the general format, number of pixels, bits per pixel, colour planes etc.

A software system to integrate sequences of images through a given algorithm. This includes the ability to 'shift' one set of images relative to another (to account for transport delays); and to derive statistical parameters. Model based data fusion [2] can also be accommodated.

3.6 Interpretation/feature extraction

Data at this layer will be offer a set of 'user data' values designed to interface to process monitoring and control systems.

This data is very application specific e.g. the air-core diameter and position within a hydrocyclone.

3.5 Configuration/control program.

This key software program provides a variety of services depending upon the implementation.

In the simplest case of an embedded system this would simply offer a set of pre-set timed operations to specific and constrained sensor operations. Some of the data layers may be activated to deliver continuous, or sampled, output of 'intermediate' or 'final' data. Such output would then be used for process monitoring or closed loop control applications.

In the laboratory (PC based) system the program offers an 'experimental user interface'. This provides a set of menus from which the configuration of sensors in use, data acquisition options, reconstruction options, and parameters for other modules can be set.

In its most sophisticated mode the system collects multiple sensor data, reconstructs each data stream, and performs on-line data fusion. When in 'Run' mode the system provides visual (window) displays of selected data layer channels. Thus a 'rolling' sensor output could be displayed, and its on-line reconstructed image (similar to a current single mode system). When in 'Configure' mode a simple set of options is offered. The range of sensors would be available (sensed directly from the sensor-heads and its status information).

These features can be delivered by a modern operating system with appropriate application development software. Although a number of operating system could deliver this functionality, Microsoft Windows-NT was selected. Although on-line display of images is not a strategic requirement for *fast* PT systems, a current PC (eg Pentium with PCI bus) is able to support multiple image display with a reasonable refresh rate.

An example graphical user interface for the visualisation of data and hardware informaion is shown in the figure 1 below.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the support of the UK Government Office of Science and Technology, through its Foresight Challenge programme; and the support and generous co-operation of their industrial partners: British Steel plc, BP plc, DuPont Ltd, ECC plc, ICI plc, Malvern Instruments Ltd, Pilkingtons plc, Process Tomography Ltd, Schlumberger Cambridge Research Ltd, Unilever plc, Zeneca plc.

REFERENCES

[1] B.S. Hoyle *et al*, Multi-sensor process tomography system design: part 1 – systems and hardware engineering.

[2] R.M. West and R.A. Williams, Opportunities for data fusion in multi-modal tomography.

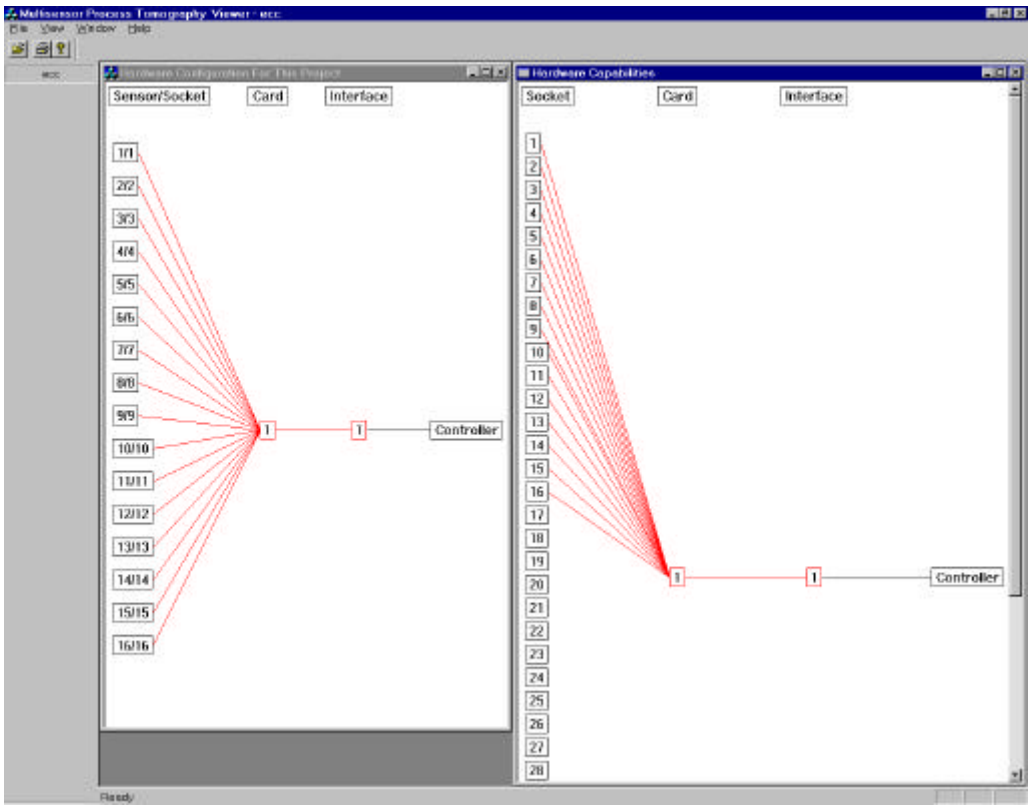

Figure 1