Brunel UNIVERSITY WEST LONDON

School of Engineering and Design

PhD Systems Engineering

A Generic Predictive Information System for Resource Planning and Optimisation

SIAMAK TAVAKOLI

Supervised by:

Dr. ALI MOUSAVI

Professor MALCOLM IRVING

September 2010

A thesis submitted in fulfilment of the requirements for the award of the degree of Doctor of Philosophy in Systems Engineering

ABSTRACT

The purpose of this research work is to demonstrate the feasibility of creating a quick response decision platform for middle management in industry. It utilises the strengths of current, but more importantly creates a leap forward in the theory and practice of Supervisory and Data Acquisition (SCADA) systems and Discrete Event Simulation and Modelling (DESM). The proposed research platform uses real-time data and creates an automatic platform for real-time and predictive system analysis, giving current and ahead of time information on the performance of the system in an efficient manner.

Data acquisition as the backend connection of data integration system to the shop floor faces both hardware and software challenges for coping with large scale real-time data collection. Limited scope of SCADA systems does not make them suitable candidates for this. Cost effectiveness, complexity, and efficiency-orientation of proprietary solutions leave space for more challenge.

A Flexible Data Input Layer Architecture (FDILA) is proposed to address generic data integration platform so a multitude of data sources can be connected to the data processing unit. The efficiency of the proposed integration architecture lies in decentralising and distributing services between different layers.

A novel Sensitivity Analysis (SA) method called EvenTracker is proposed as an effective tool to measure the importance and priority of inputs to the system. The EvenTracker method is introduced to deal with the complexity systems in real-time. The approach takes advantage of event-based definition of data involved in process flow. The underpinning logic behind EvenTracker SA method is capturing the cause-effect relationships between triggers (input variables) and events (output variables) at a specified period of time determined by an expert. The approach does not require estimating data distribution of any kind. Neither the performance model requires execution beyond the real-time. The proposed EvenTracker sensitivity analysis method has the lowest computational complexity compared with other popular sensitivity analysis methods.

For proof of concept, a three tier data integration system was designed and developed by using National Instruments' LabVIEW programming language, Rockwell Automation's Arena simulation and modelling software, and OPC data communication software. A laboratory-based conveyor system with 29 sensors was installed to simulate a typical shop floor production line.

In addition, EvenTracker SA method has been implemented on the data extracted from 28 sensors of one manufacturing line in a real factory. The experiment has resulted 14% of the input variables to be unimportant for evaluation of model outputs. The method proved a time efficiency gain of 52% on the analysis of filtered system when unimportant input variables were not sampled anymore. The EvenTracker SA method compared to Entropy-based SA technique, as the only other method that can be used for real-time purposes, is quicker, more accurate and less computationally burdensome. Additionally, theoretic estimation of computational complexity of SA methods based on both structural complexity and energy-time analysis resulted in favour of the efficiency of the proposed EvenTracker SA method.

Both laboratory and factory-based experiments demonstrated flexibility and efficiency of the proposed solution.

ACKNOWLEDGEMENT

I would like to acknowledge the contributions of the following groups and individuals to the development of my thesis:

- My supervisors Dr. Alireza Mousavi and Professor Malcolm Irving who never hesitated to lead me throughout this thesis in every aspects of it.
- The Engineering and Physical Sciences Research Council (EPSRC) of the United Kingdom for their financial and continuous support.
- Mr. Peter Broomhead, lecturer at School of Engineering and Design, Brunel University, who allowed me brainstorm and receive feedbacks on many aspects of this research and development including the final draft.
- Dr. Bernard Krzykacz-Hausmann, member of committee on Safety Margins Action Plan (SMAP) Nuclear Energy Agency (NEA), who introduced me to his methodology on Entropy-based Sensitivity Analysis.
- Managers and staff of Bahman-Philver refrigerator manufacturing plant in Tehran, who keenly allowed me access to the production line.
- Technical staff of National Instruments UK branch, who devotedly try to find answer to questions on LabVIEW development tool.
- Members of System Engineering Research Group (SERG) at School of Engineering and Design, Brunel University who maintained a great atmosphere of passion and joy at work.
- Administrative staff of School of Engineering and Design, Brunel University without whose support accomplishment of this work could not be completed.

DEDICATION

This thesis is dedicated to my father, Abdolvahab Tavakoli, who taught me that even the largest task can be accomplished if it is done one step at a time. It is also dedicated to my mother, Iran Zanjan, who taught me that one can find an answer if he truly loves his questions and lives them.

This thesis is also dedicated to my beloved wife, Saghar Masihi, who passionately encouraged and fully supported me throughout completion of this work.

Finally, this thesis is dedicated to my beloved brothers Ebrahim, Bahram, Saeed, and Aryan who proved to me that one can do much more when he does more than what he knows he can do.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

LIST OF ABBREVIATIONS

Abbreviation	Stands for
AD	Algorithmic Differentiation
ANOVA	ANalysis Of VAriance
AS	Analysis Span
CIM	Computer Integrated Manufacturing
CNC	Computed Numerically Controlled
CR	Cut-off Ratio
CT	Cut-off Threshold
DAQ	Data AcQuisition
DAU	Data Acquisition Unit
DDL	Data Definition Layer
DES	Discrete Event Simulation
DESM	Discrete Event Simulation and Modelling
DID	Data Identification
DIL	Data Integration Layer
DMS	Database Management System
DMU	Data Management Unit
DPL	Data Processing Layer
DPU	Data Processing Unit
DVM	Derived Variable Method
EA	Energy Analysis
ED	Event Data
EPC	Event-driven Process Chain
ERP	Enterprise Requirement Planning
ESA	Entropy-based Sensitivity Analysis
ET	Event Threshold
FAST	Fourier Amplitude Sensitivity Test
FDILA	Flexible Data Input Layer Architecture
FIFO	First-In-First-Out
FMCG	Fast Moving Consumer Goods
FMS	Flexible Manufacturing System
GSISD	Global Sustainable Information System Development
HOOD	Hierarchical Object Oriented Design
I/O	Input / Output
ICT	Information and Communication Technology
IT	Information Technology
IVS	Input Variable Selection
JIT	Just-In-Time
KPF	Key Performance Factor
KPI	Key Performance Indicator
LHS	Latin Hypercube Sampling

MPSG	Message-based Part State Graph
MRP	Material Requirement Planning
OAT	One-At-a-Time
OPC	OLE for Process Control
OVM	Original Variable Method
PCA	Principal Component Analysis
PDL	Performance Definition Layer
PI	Port Input
PLC	Programmable Control Logic
R3M	Real-time Model Matching Mechanism
RFID	Radio Frequency IDentification
SA	Sensitivity Analysis
SAM	Sensitivity Analysis Method
SCADA	Supervisory Control And Data Acquisition
SDL	Scenario Definition Layer
SI	Sensitivity Index
SIMS	Semi-Integrated Manufacturing System
SPE	Simplified Polynomial Expansion
SS	Search Span
TD	Trigger Data
TT	Trigger Threshold
VDL	Variable Definition Layer
VI	Variable Input
VID	Variable Identification

1. Introduction

In this chapter, an overview of the context of the problem to be solved will be given. Based on the identified gap in the existing solutions, aim and objectives of this research which lead to the proposed solution will be defined. Finally, the essentials and organisation of the research work will be discussed.

In the current climate of economic challenges and environmental constraints, corporate middle management is under enormous pressure to maintain efficiency and effectiveness on the shop floor. Multiple performance factors have to be optimised (e.g. utilisation, customer satisfaction, waste, efficiency, and environment) and quick response decisions have to be made on minute-by-minute basis.

To date a plethora of powerful Supervisory Control And Data Acquisition (SCADA) (e.g. Rockwell, Siemens, ABB, Honeywell), Enterprise Requirement Planning (ERP) organisational integrators (e.g. SAP, Oracle), and custom-built decision support systems have been introduced into the market place with a degree of success (ABB, 2010; Honeywell International Inc.; Oracle, 2010; OSIsoft Inc., 2007; Rockwell Automation Inc., 2010; SAP AG., 2009; Siemens, 2009).

According to Mitchell (1991), up to 50 percent of the cost of a manufacturing facility can be raised from activities and resources associated with information integration and processing. Moreover, industrialists seem to be reluctant to adopt expensive and unnecessary information and monitoring system (Cecelja, 2002). They often want to see feasibility on the cost of equipment and training before they sacrifice added value of one-to-one contact with the production process (Cecelja, 2002).

On the other hand, manufacturing control solution providers are neither keen nor ready to offer a generic decision making tool which can be configured to work in more than one manufacturing field by an end user who is not as expert as the tool developer (Tavakoli, 2007; Tavakoli, 2008; Tavakoli, 2009; National

Instruments UK and Ireland, 2007; National Instruments UK and Ireland, 2008).

1.1. Manufacturing decision making tools

Figure 1-1 shows two ends of the spectrum of applied information systems for decision making in a typical manufacturing plant.

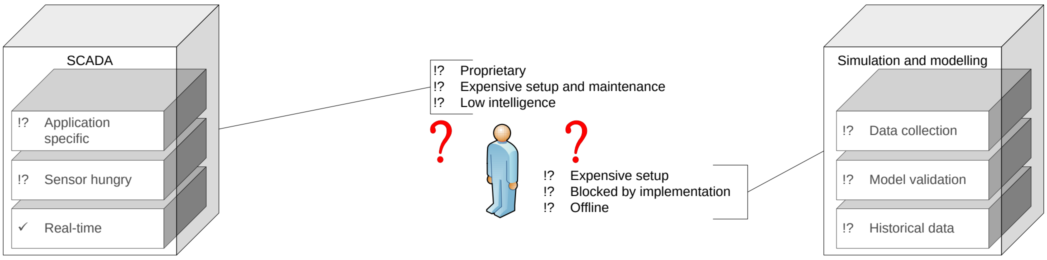


Figure 1-1 SCADA and simulation modelling as two ends of applied information system spectrum

As it could be seen in Figure 1-1, at one end of the spectrum SCADA systems provide real-time access to the sensory data of the shop floor. They are designed for presenting and controlling shop floor events in specified applications. At the other end of the spectrum, simulation systems represent data and processes of the shop floor based on a validated model. Data collection and validation could be a daunting task in simulation modelling .

According to literature, survey (interview) of industrialists, and our experience, especially the end users e.g. manufacturers, logistics, supply chain integrators, etc. have shown that SCADA tools and technologies that are designed to aid in these instances suffer from (Tavakoli, 2007; Tavakoli et al, 2007a; Tavakoli et al, 2007b; Tavakoli et al, 2007c; Tavakoli, 2008a; Tavakoli, 2008b; Tavakoli, 2009; Tavakoli et al, 2009a; Tavakoli et al, 2009b; Tavakoli et al, 2010a):

1. Costly installation and large after sale complexities and interruptions to the running system,
2. Integration of legacy or multi-supplier data acquisition equipment, database management system, IT infrastructure and machine tools,

3. Generation of large number of parameters and data with little timely post analysis and emphasis on interdependencies of information and processes,
4. Changes to plant layout or any modifications to equipment and machinery which requires major rewrite and recalibration of the SCADA/ERP system which normally requires expensive intervention by the supplier or third party contractors – often costly, disruptive and frustrating for the client,
5. Little flexibility with traditional SCADA systems in adapting to organisational and shop floor changes, which in effect curtails the confidence to propose new initiatives for improvements in the shop floor,
6. Little to no capability to translate vital data into meaningful information for quick response decision making,
7. Obstructed implementation at shop floor level due to impracticalities of the solutions, despite complex production planning and scheduling commands that are generated from ERP systems, thus reverting to conservative modular scheduling rather than on-fly real-time production planning.

The basic argument is that current technology is well capable of producing data and in most cases overwhelming decision makers with data. What is missing is the way to capture the complexity of decision making when faced with multiple often competing and contradictory factors.

1.2. A production management scenario

Consider a state-of-the-art manufacturing plant that bottles a product. The SCADA is highlighting discrepancies at the filling station i.e. some of the bottles are being overfilled (product is being given away free) and some are being under-filled i.e. ejected from the system. This quality issue is raised with the line manager. A quick diagnosis and identification of the fault reveals that the fillers need to be recalibrated and requires 2 hour attention. The highly regarded ERP system has succinctly planned and outlined the shift's production rate and plan to meet targets.

The line manager needs to make a decision to either shut down the line and deal with the line efficiency issue (yield and waste) or continue with the production to meet the deadline and customer demand (customer satisfaction).

Which of the two options will be the best decision?

The orthodox principle, dictates following the conventional policies, revert to similar decisions in the past or gut feeling – usually defendable at the board of directors but impossible to quantify the long term costs associated with such decision making.

To date there has been no method to quantifiably help us to verify such decisions that might not in short-term reveal their costs and threat to the organisation.

The main purpose here is to capture and visualise the randomness of events on the shop floor with minimum reliance on detecting devices. To extract information regarding fluctuation of demand, flow of material, availability of resources, changes in quality, reliability, work-in-process, queues and waiting times. Unless a fully automated factory is equipped with the latest ERP and SCADA equipment (often rigid) that can detect all such events, currently it is impossible to have a time critical and accurate (Cecelja, 2002) picture of the shop floor performance.

1.3. Human's nervous system and data acquisition and modelling – an analogy

Will Cuppy, the American writer (08/23/1884 – 09/19/1949) said: "*Aristotle was famous for knowing everything. He taught that the brain exists merely to cool the blood and is not involved in the process of thinking...*". Nowadays, we know that it is not impossible for brain to control body functions and consequently reduce blood circulation and from that body temperature by thinking.

The basic concept of this research is inspired by how the human's nervous system functions. Our nervous system is equipped with millions of sensory cells (DAQ) which continuously collect data from our surroundings. These cells combine information into five general sensory systems i.e. sight, hearing, taste, smell, and touch. They all send signals to the brain through a highly specialised information system architecture (the nervous system shown in Figure 1-2).

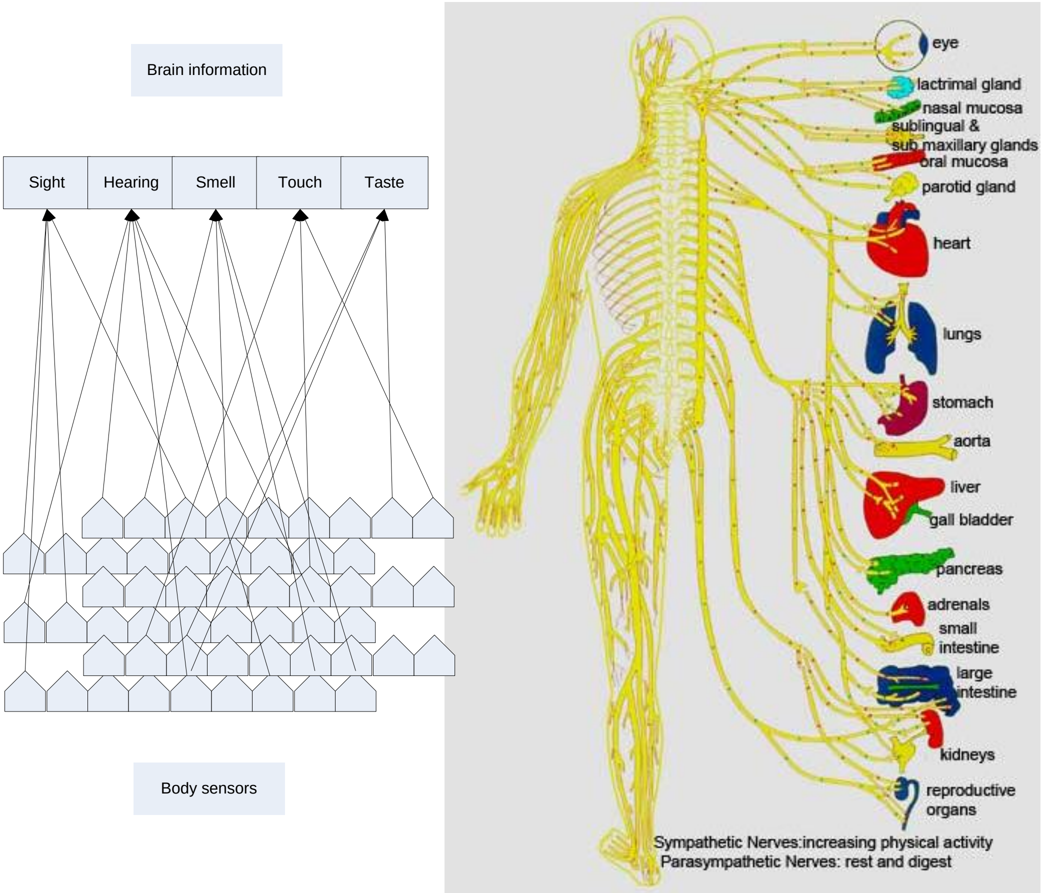


Figure 1-2 Analogy between human's nervous system and interpretation of data to information

The data is then translated into specific information e.g. brightness, colour, loudness, temperature, pain and so on. The conversion of millions of data points into a very small number of critical information points enables us to quickly respond to situations. The quality of decisions normally depends on how effectively we translate the data and how we relate them to the critical information. For example, a gymnast, when in action (Figure 1-3), may need to process received information from sight, hearing and touch sensors more effectively than the other sensors of his or her body.

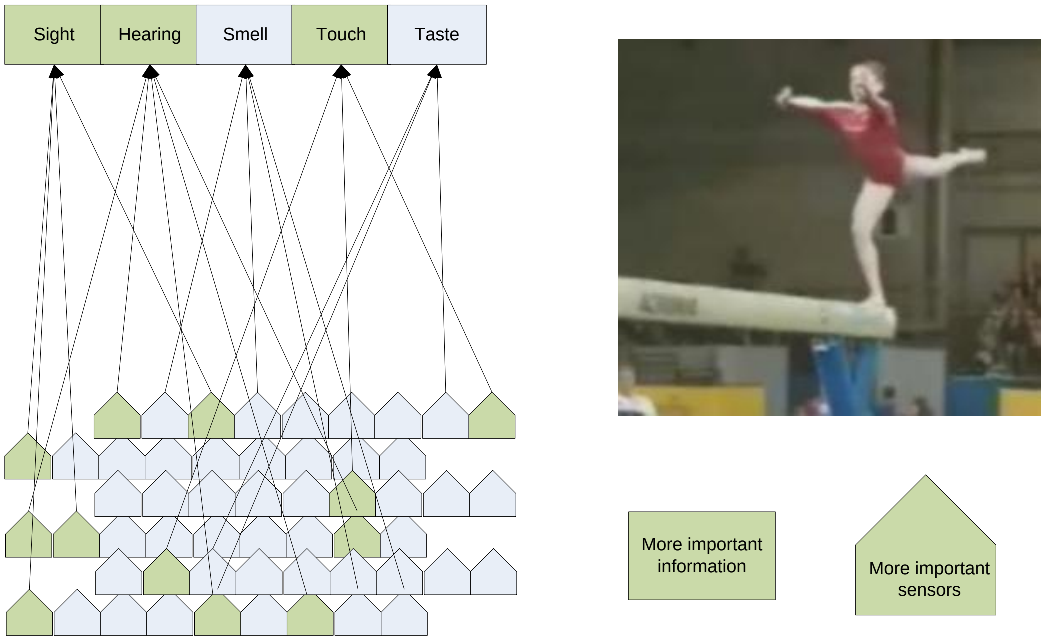


Figure 1-3 Reception of some data could be more important than some other

The research presented here is inspired by this concept.

1.4. The impact

The solution proposed in this research will allow for translation of key data into Key Performance Indicators (KPI). It will provide the middle management the opportunity to visualise and intervene on the daily activities of their operations and reduce the hidden cost of *Orthodox Thinking and Approach.* The predictive feature of the proposed solution will allow initiatives to be aired and assessed freely, with no actual disruption to the operations. Confidence in management will increase and the risks will be reduced, adaptation will turn into aggressive prediction, learning and positive change. This will lead to the mutation-evolution of existing man-made complex systems to viable systems.

1.4.1. Global Sustainable Information System Development (GSISD)

In the global approach to providing solution, analysis, and economic decision making, a common understanding of the underlying issues would not be possible without having access to the “relevant” knowledge and the basic data in real time (Choucri, 1999). Therefore, both connectivity to the real-time data and selectivity on the relevant data have significant impact on the effectiveness of a global and sustainable solution. Global and sustainable have been two ambitious targets in many development projects with environmental resource utilisation cost effectiveness in their agenda. The reason for this lies in the invariable complexity of global problems. By sustainable development it is meant: "*The process of meeting the needs of current and future generations without undermining the resilience of life-supporting properties or the integrity and cohesion of social systems*" (Choucri, 1999). At challenges to attain sustainable solutions, implementing the views of stakeholders about what is "real" and what is "important" (Choucri, 1999) is extremely important. Economic production is one of the dimensions of sustainability (Choucri, 1999).

1.4.2. Trapped in technology

In a changing environment, an unsustainable measurement infrastructure which cannot take in new coming measurement tools that are capable of satisfying new and more complicated problems could cause vital crisis. For example, in existing track and trace technology, large airports sometimes suffer from a lack of the coverage by their ground radar system in keeping track of hundreds of moving planes (Dumiak, 2008). Such airports need fitting with a new magnetic sensory system to fill the gaps of coverage. If the existing data acquisition system cannot provide sufficient capacity for the integration of new data sources into the existing ones, the airport security system will be "trapped in technology", required to revamp its measurement infrastructure.

In this sense, the biggest impacts of "technology trap" could be mentioned to be old weapons. However, the scope of this thesis is limited to generic and sustainable solutions that will work throughout generations of decision making tools and technologies.

1.5. The aim and objectives of the thesis

The aim of this research is to reduce the cost of orthodox decision making by being able to capture, visualise and assess the impact that such decisions have on the current and future performance of a system / organisation. In order to do so, this thesis presents a suite of technology and analysis tools to help with real-time decision making in typical industrial system. The system is tested in two experiments; firstly, in a laboratory demonstration and secondly in a refrigeration manufacturing plant.

Objectives

1. To capture and to allow for visualisation of the randomness of shop floor day-to-day operations,
2. To introduce a flexible and efficient in/output data layer that is capable of integrating multiple platform ICT infrastructure and equipment, thus overcoming cross platform anomalies in the data management systems,
3. To introduce and validate a novel sensitivity analysis methodology that is capable of producing sensitivity indices of system's Key Performance Factors (KPFs) with respect to system's input variables in time-constraint applications.
4. To test and validate the proposed system with a laboratory size production line simulator to prove the capability and the advantages of the proposed solution against existing SCADA, organisation integration platforms and traditional systems modelling and simulation techniques.

1.6. The scope and limitations of the project

In this section the route towards the identification of the dimensions of the problem is discovered. After discussing the impact of a flexible and predictive data integration solution, it is required to identify and outline the materials and efforts that could lead towards the definition of the problem and proposal of the solution. Manufacturing scenarios were selected as one of the major popular contexts of the potential data integration problems. Therefore, the following will cover a more detailed overview of flexible manufacturing systems, computer integrated manufacturing. Objectives of the associated literature review are as follows;

- To identify the extent that the current methodologies of decision making tools have taken advantage of new technologies, in other words, how updated the current decision making tools are,

- To provide a clear indication of the gap in the current decision making tools,
- To review those areas of knowledge base which will be covered by the proposed methodology,

1.6.1. Flexible manufacturing systems, computer integrated manufacturing, and simulation

A Flexible Manufacturing System (FMS) is referred to a manufacturing system in which standardised production is supposed to facilitate operations and therefore reduce the overall cost of manufacturing (Joshi et al, 1994). Standardised manufacturing system could cost more than a customised one due to its capacity for high volume production which is not always the case (Underwood, 1994). Therefore, implementation of an information system that can support manufacturing systems using cost effective solutions is the key requirement of manufacturers worldwide.

Manufacturing systems comprise of large and complex interrelated processes and human inspired initiatives. Continuous transformations take place inside these enterprises at intervals. By transformation it is meant changes in system parameters and the rules that govern these systems. Later on in this thesis they are referred as ‘events’. The associated data including functional data – company’s knowledge base how to function, product data – generated data about function, operational data – plans and instructions, and performance data required better coordination and organisation (Hannam, 1996). Response to this requirement has been the introduction of more computers in the company and computer controlled equipment on the factory floor. This has led to the concept of Computer Integrated Manufacturing (CIM) (Jones et al, 1990) and simulation modelling amongst other technologies and tools for analysing and controlling these complex systems. But how and to what extent can CIM benefit manufacturing is the subject of next section.

1.6.2. Scope of control and analysis problems in FMS

An important perspective in the attempt to find solutions for information system problems in FMS is the change in high number and variety of linked functions and operations due to their flexible and integrated nature (Hannam, 1996). Joshi et al (1994) divided control problems for an FMS into three categories based on their life cycle and range of the hierarchical structure of the control system; long-term planning, which concerns production strategies and capacities and extends throughout the whole company, medium-term planning, which covers off-line control tasks and decisions about the system, and finally short-term operation, which involves real-time control tasks and scheduling of machines and material handling equipments.

Continuous awareness for change reduces associated risks in volatile environments (Viswanadham et al, 1992). For example, in the Just-In-Time (JIT) approach to manufacturing (Bauer et al, 1994) production smoothing is a technique used to help production respond effectively to short term variations in market demand. In this technique, adaptation to periodic market demands is achieved through periodic preparation of average production level of each process based on an aggregate longer periods and demand forecast (Bauer et al, 1994). This implicates continuous analysis and control over each process. Therefore, dynamic monitoring of short term process parameters takes the largest proportion of analysis effort and control tasks in FMS.

1.6.2.1. Dynamic system analysis

System analysis activities in FMS are centred on either validation of the existing model (static analysis), mainly ensuring that information resources are available for processing activities, or on the actual processing and determination of performance indicators of the system (dynamic analysis). Figure 1-4 shows a conceptual diagram of a typical dynamic system analysis activity in FMS (Hannam, 1996). As is shown in the figure, the flow of real-time data passes through the data management and the data processing components. Four basic components form the computing model of dynamic data analysis; data management, data processing, data communication, and user interface (Bauer et al, 1994).

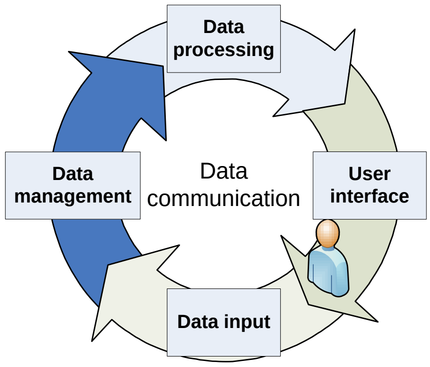


Figure 1-4 Activities of a typical dynamic system analysis

1.6.2.2. Flexible data integration

As an important aspect of information technology in FMS, flexibility brings with itself decentralisation and distribution of functionalities and decision making responsibilities (Bauer et al, 1994). This necessitates exploration of those areas of data management opportunities which lead to the controlling the cost of monitoring a changeable system. An optional component of the dynamic system analysis has been automatic selection of more important input data points into the analysis system, or Input Variable Selection (IVS) (Guyon et al, 2003). This area has been explored but lacks sufficient emphasis on the dynamism and randomness of changes to system and the input variables.

1.6.2.3. Simulation and modelling

Dynamic data processing in highly complex manufacturing systems where multiple product parts, sequence dependent setup, with a variety of scheduling and operational choices are integrated could be handled via simulation based decision support solutions (Gupta et al, 2002). Simulation models could generate feasible schedules with ability to reschedule the system when sudden changes occur.

1.6.2.4. Look-ahead scenario execution

A product manager, when predicting the cost of the system based upon a 'what-if' scenario, may use his or her experiences on the behaviour of the system. Due to lack of knowledge about future events and randomness of errors, this judgment may easily be made upon some assumptions which are not the same between different product managers, in other words, subjective probabilities (Mousavi et al, 2007).

Preparing a look-ahead indicator of the system status must be based on the long run relative frequency interpretation of probabilities of events and data in the system. In other words, characteristics of the system components at fast-forward run must closely match the real system. This necessitates proper use of the real-time data for fitting a model of the behaviour of the system and its components.

Identification of probability distributions and evaluation of goodness-of-fit to the chosen distributions which are going to represent input data to the system in fast-forward mode of simulation is introduced as an established approach by (Banks, 2001). Such approach could be computationally automated within the deterministic part of data integration layers.

1.6.2.5. User Interaction

Due to the complexity and continuous evolution of FMS, intervention in structures in the plant is inevitable. System reconfigurations and extensions like adding machinery, sensors, or maintenance and upgrading components of the information system are a regular necessity. Current solution providers deal with this problem as an expensive specialty service to their end users.

1.7. Thesis structure

The problem of dynamic and flexible decision making solution tool is explored with respect to subject areas; (1) real-time data integration systems, (2) discrete-event simulation modelling, and (3) input variable selection.

In the following three chapters a review of existing literature will be reported and appraised. Design of the architecture of the solution and an implementation of a testing platform will then be described in detail. Finally, the improvements made by the proposed system and the usefulness of the proposed approach (i.e. how much the proposed work resolved the problem) in comparison to the alternative solutions will be discussed. This will lead to conclusion of the work on delivering better solution in this area of research and development. A series of appendices support infrastructural knowledge or complimentary comments where seemed necessary.

The dissertation and its constituents, as shown in Figure 1-5, were prepared under three themes and ten chapters;

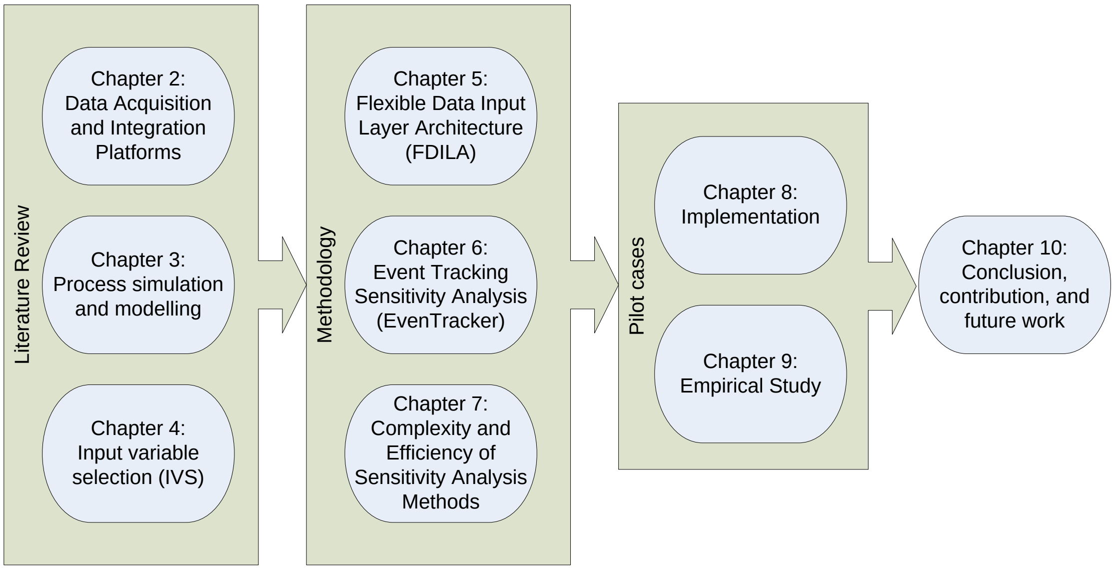


Figure 1-5 Logical flow of the thesis

1.7.1. Literature review theme

This theme covers three chapters on design approaches and implementation issues of data acquisition and integration, simulation modelling, and input variable selection. Gaps of the current methodologies and technologies are discussed.

1.7.1.1. Chapter two

Draws attention to the components and properties of existing data acquisition and data integration platforms with more focus on the manufacturing scenarios. Discussion will be made about the borders and efficiency parameters of data integration systems. Finally, off-the-shelf data integration systems and their advantages and disadvantages will be discussed as the basis to propose a novel solution.

1.7.1.2. Chapter three

Discusses how discrete-event simulation and modelling could, as part of a decision making tool, complement the proposed solution as a strong component for decision making tool. For this, this chapter delves into how simulation and modelling has been useful in the context of the defined problem. Properties of application of simulation in manufacturing will be discussed. Dimensions of real-time and discrete-event simulation will be explained. Finally, criteria for selection of a simulation modelling tool will be introduced in the proposed solution as a productive platform.

1.7.1.3. Chapter four

Input Variable Selection (IVS), as methodology to help with the reduction of the computational cost of data integration and simulation solutions is explained with emphasis on the difference between variable and feature as well as between selection and information construction. Description of variable selection problem is then classified into two categories of “derived” and “original” variable methods. Finally, sensitivity analysis is introduced as a technique to evaluate the importance of input variables.

1.7.2. The methodology theme

This theme includes three chapters about the approach taken to provide a solution to the problem and the proof of its efficiency. It therefore covers the proposed data integration architecture, the proposed sensitivity analysis technique, and the proposed method for evaluation of various sensitivity analysis methods against the proposed method.

1.7.2.1. Chapter five

Proposes an architecture that addresses complexity and efficiency issues of data integration in design, implementation, and execution aspects.

1.7.2.2. Chapter six

Puts forward a methodology that helps with cost reduction of computations involved in data integration by focusing on the use of input data when studying and measuring the level of their influence on system output.

1.7.2.3. Chapter seven

Extends the comparison of the proposed sensitivity analysis methodology with other major sensitivity analysis methods. Appropriate computational complexity metrics of sensitivity analysis methods will be provided and applied to the comparison.

1.7.3. The Pilot cases theme

This theme introduces two main tests and implementation scenarios of the proposed methodology, each for different purpose; one for understanding the outcome of data integration methodology proposed in chapter eight, and the other for analysis of the output of sensitivity analysis method proposed in chapter nine. Both cases are experimental; one on laboratory-based and the other in factory-based platform.

1.7.3.1. Chapter eight

To help with proof of concept of data integration architecture proposed in this study, in chapter eight implementation of a data integration system from design and development to execution on a laboratory based platform is described. The suite of hardware and software which was designed, developed, and implemented demonstrated the versatility and flexibility in the input data definition as well as real-time data acquisition and simulation.

1.7.3.2. Chapter nine

Completes the proof of concept by implementing the proposed sensitivity analysis method and demonstrating its efficiency in comparison to other conventional sensitivity analysis methods. This is achieved by a case study conducted in a real world factory-based set of experiments.

1.7.3.3. Chapter ten

Concludes the research work and discusses the potential for future and development work.

2. A Review of Current Data Acquisition and Integration Mechanisms, Approaches, and Technology Solutions

This chapter draws attention to the properties of existing data acquisition and data integration platforms with more focus on manufacturing scenarios. A broad analysis of key features, boundaries, and efficiency parameters of existing data integration systems will be discussed. Finally, off-the-shelf data integration solutions and their advantages and disadvantages will be discussed as support for the proposed solution.

In today's enterprise engineering and business development, installation and support for a platform that can handle information about the progress of the processes involved in the performance assessment of enterprises is inevitable. For example, in manufacturing, a virtual enterprise is an extension of a manufacturing company that interprets and reports key performance parameters that assure competency of operations (Goossenaerts et al, 1997). The information system is considered to be the major bottleneck in the formation of the virtual enterprises (Goossenaerts et al, 1997). The issue of incompetency of information system in manufacturing has been tackled from two perspectives; first, infrastructural (Mills et al, 1997; Reid et al, 1997; Kosanke et al, 1997), and second, system extendibility (Benadjaoud et al, 1997; Goossenaerts et al, 1997). The malaise persists on different levels from globally distributed manufacturing networks (Katzy et al,1997; Mertins et al, 1997) to work-cells on shop floors (Takata et al, 1997; Goossenaerts et al, 1997).

Data integration, like a passageway, is known to be the collection of activities for acquiring the raw data from the actual business and preparing it for the other parts of the information system such as data processing (Mills et al, 1997). Thus, the data integration system is usually connected to the network of sensors and data sources at one end and to the data processing system at the other end. In order to meet the expectations for data integration, it is important to understand the attitude and relationship between data integration and the sensor network with respect to the requirements of data processing.

2.1. Static sensor network and dynamic data integration

In many data integration applications, it is not feasible to change the configuration of data sources (i.e. the type and position of the sensors that are installed). The type, number, and location of sensors in most of the applications are not subject to day-to-day change.

For example, in air traffic control, ground radar systems may not cover the tracking of hundreds of moving planes which may cause deadly consequences like the 1977 jumbo jets collision on the ground. To avoid this, many small sensors were installed to fill in the blind spots (Dumiak, 2008).

As another example, collapse of the I-35W Mississippi River Bridge in Minneapolis, Minnesota, United States on August 1st 2007 raised questions on the efficiency of the applied inspection systems. The installation of many piezoelectric sensors at potential failure points with wireless connection to monitoring systems that could warn early enough about the bridge structure condition, was considered as one potential solution by Subramanian (2008).

Although in the two examples above efficiency in the configuration of sensors is an important aspect to ensure data is collected properly of the installation of sensors, but no mention of how modification can be made to the original setup if conditions change. Maintaining the efficiency of such information systems remains highly dependent on the performance of data gathering efforts. For example, in the air traffic control scenario, if some blind spots are temporarily used by other equipment and not used for traffic, a proper task in the data integration may decide to ignore sampling data from sensors which cover the idle spot. Identification of the components of data integration and their performance issues is a prerequisite to the feasibility study of the application of such a 'sensor selectivity' task.

2.2. Components of data integration

Data acquisition and data processing are the main challenges of any data integration system. They include data collection equipment and devices, communication standards and protocols, sampling rules, multiple value structures, storage, and data restructure for effective use in higher levels of information systems.

Among the five building blocks in the architecture of the production activity control introduced by Bauer et al (1994), the term ‘monitor’ was applied to the block which handles the reception of data from shop floor devices, and provides status data for dispatching, performance measures for scheduling, and high level information for factory coordination. In terms of functionality, the major components of the monitor block are data capture (or data acquisition), data analysis, and decision support (Bauer et al, 1994).

On the other hand, Bauer et al (1994) introduced four basic components for a computing model; data management, data processing, data communication, and user interface (Figure 2-1).

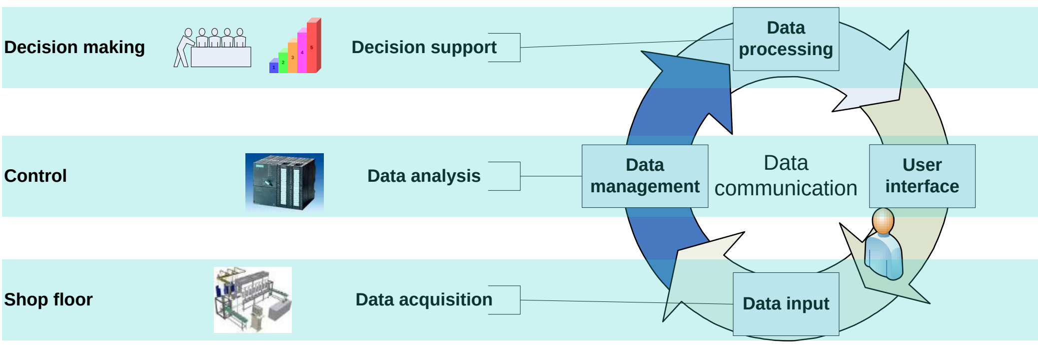


Figure 2-1 Computing model of data integration and its components

The components are explained further within the following sections.

2.2.1. Data acquisition

Data acquisition, defined by Taylor (1997) as '*the branch of engineering dealing with collecting information from a number of analogue sources and converting it to digital form suitable for transmission to a computer, printer, or alphanumeric display*' is known as the key element for reliable, fast, and accurate shop floor monitoring (Bauer et al, 1994). Hannam (1996) mentions a number of performance measures which could be monitored by capturing data from the shop floor; they include 'machine monitoring' as direct monitoring and diagnosis of production parameters, 'work-in-progress tracking' as understanding of the actual progress of production, 'time and attendance recording' as record of presence of staff and material, 'shipping and receiving recording' as understanding of actual progress of goods and orders, and 'inventory control' to monitor utilisation of resources.

In addition to monitoring shop floor performance, Hannan (1996) mentions that data acquisition could help manufacturers with monitoring the whole manufacturing operation. For example, quality, cost of product and cost of production requires frequent measurement of product related parameters as well as production environmental parameters. These include capturing data for temperature, pressure, weight, flow rate, dimensions to mention a few examples.

Provision of a complete data acquisition system requires attention to both hardware and software issues such as sensors, signal multiplexing and sampling, memory and interfacing (Taylor, 1997). Although commercial data acquisition systems provide an assembled and ready to use version of all these tasks and issues, they come with expensive and extra costs (Taylor, 1997). To avoid the cost issue while maintaining efficiency, Taylor (1997) suggests understanding the exact requirements by spending experts' time.

Hannan (1996) mentions Supervisory Control And Data Acquisition (SCADA) systems as potential solutions to the problem of efficient data collection. Definition and limited scope of applications of SCADA systems are discussed in the following subsection.

2.2.1.1. Supervisory control and data acquisition (SCADA)

Supervisory control and data acquisition (SCADA) system is defined by Boyer (1999) as “*the technology that enables a user to collect data from one or more distant facilities and/or send limited control instructions to those facilities*“. SCADA covers data acquisition, data communication, data processing, and data visualization as well as generating control commands (Cobus, 2003). Data processing in SCADA includes tasks which measure performance parameters, relate system inputs to the performance measure, and finally determine the setpoints that maximise the value of the performance measure (Murrill, 1988).

The scope of the application of SCADA, however, is the processes that need relatively simple control and monitoring scenarios, such as opening and closing valves of small hydroelectric stations according to the requested demand, or regular meter readings of oil and gas production facilities, or similar (Boyer, 1999).

Expressed as a shortcoming of SCADA systems, Boyer (1999) points out that “high risk” situations – when both the probability and the consequences of system failure are high and serious – especially in remote conditions, must be excluded from implementation of SCADA systems. Chiesa et al (2007) points at a similar issue under different terms which express worries about the high cost of securing and patching a SCADA system against IT-related flaws.

2.2.2. Data analysis

A core component in managing acquired data is its interpretation in order to ensure that the next component - decision support - receives the ideal required information efficiently (Kirianaki et al, 2002). In general, the task of data analysis and management is lowering the cost of providing the applicable data (Kirianaki et al, 2002). This cost could come from the both data acquisition and data processing ends.

On the data acquisition end, non-ideal conditions like sensitivity of sensor signals to environmental changes and component tolerances, or proprietary signal data formats used in some smart sensing devices may require efforts for signal conditioning and compensation (Kirianaki et al, 2002). For example, barcode scanners may scan two extra characters at the beginning and at the end of the actual barcode (Hannan, 1996).

On the data processing (decision support) end the intensity of sequence of data entities are important (Kirianaki et al, 2002). This means both signal sampling rate and the number of signals can be controlled by a data analysis task, so that the data processing task applies less effort on more meaningful information without losing performance. For example, the presence of one item on the production line may be represented by the simultaneous setting of two binary data bits from two light switch sensors. Conversion of these two binary data to one resulting from logical AND function between them is a transformation that saves further computational efforts in storage and delivery of two data sets to data processing.

In terms of data intensity, it would help with efficiency of computation if a data entity which is, for some reason, totally unnecessary to the data processing, could be identified and made redundant. A criterion is required to decide on redundancy of unnecessary data entities. A number of these criteria relate input data entities to their impact and importance on the output of the system. The important issue of selecting important input data will be discussed under ‘Input Variable Selection’ in a separate chapter (chapter 4). The following will discuss another important issue in providing efficient data, and it is the effect that knowledge about usage of data in data processing could have on the way data is treated at higher levels of information management.

2.2.2.1. Data and event

Data entities provided from the presence of an item or a situation, inform about an event, and their capture helps with understanding certain causal relationships in the system. Some performance measures mentioned in the previous section (2.2.1) are mainly provided by capturing data about presence and identification of items or situations. For example, staff availability could be measured using RFID sensors that provide identification data when a person holding an RFID tag is near an RFID reader at the work place. The identification data therefore depends on ‘when’ this proximity occurs. In counting scenarios as another example, the number of items is counted as they pass a certain point with an RFID reader, barcode identification, or simple light switch. Therefore, the event data is not available at a defined time and has a stochastic nature.

Many sensor output data that represent analogue values of physical phenomena are chosen to be sampled at any rate as according to necessity. This perspective could affect the behaviour of data capture methodologies; if event data is captured and prepared for analysis only at instances of events, then computational efforts of data storage and data processing for performance modelling could shrink to those instances of events only. Such a solution has been set and trialled by researchers like Ploetner (2007) who proposed an event-triggering-based data acquisition system to reduce the cost, power, and computational requirements of video sensor acquisition. Kruger et al (2009) propose solutions for event-based monitoring to cope with the hardware and software restrictions of battery powered wireless sensor networks.

As a basic example, as shown in Figure 2-2, treating the RFID reader as data capture requires memory allocation and frequent decision (check) tasks at counting stage regardless to availability of ‘tag’ information or ‘nothing’. Instead, if when RFID reads a tag, data capture occurs, the number of processing tasks is limited to count only.

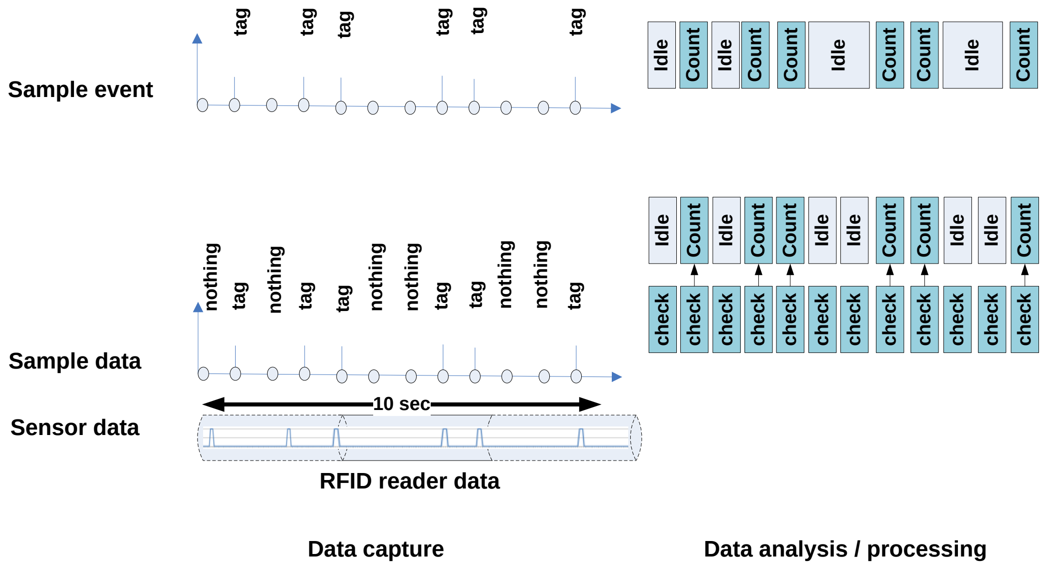


Figure 2-2 Intensity of computational tasks on sample data and sample event

While treating event data helps with lowering computational cost of data storage and data processing, the tasks of data storage itself could be performed in different ways depending on the efficiency and data retrieval.

2.2.3. Data storage

Storage of data in memory so that it could be shared between different computational activities is inevitable. Efficient data storage helps more efficient computation of data processing tasks. Access to data in terms of time and effective use is an important objective of data storage methodologies (Bauer et al, 1994). Both data model and communication with the stored data, i.e. database structure, must be considered for effective data manegement (Rembold, 1993).

2.2.3.1. Data model

Both Bauer et al (1994) and Rembold et al (1993) considered three different data models which lead to three different data storage systems; hierarchical, network, and relational.

In hierarchical and network data models data entities branch to one (hierarchical) or more (network) other data entities in a tree structure. Of course, access to each data entity has to be made through first accessing the data entity at the root and other nodes before the enquired data entity (Rob, 2008). For example, as shown in Figure 2-3 data about the equipment and sensors of a manufacturing machinery are represented based on their relationship with the machinery.

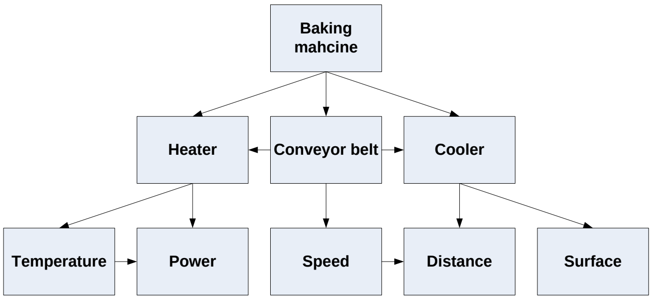


Figure 2-3 Network data model (Hierarchical data model without horizontal links) representation of a baking machinery

The tree approach to access each data entity makes a complex data structure in terms of design and development (Bauer et al, 1994), as well as implementation and restructuring (Rembold et al, 1993) particularly in data integration problems for FMS where the association of equipment and sensors are subject to frequent change.

In relational data model, each data entity has a unique address in the format of a table, i.e. a pair of rows and columns (Haritsa et al, 2008). Some data entities may share either row or column depending on their relationship, as shown in Figure 2-4.

Name	Current	Range	Equipment	Machinery
Temp1	78	0-100	Heat1	Baking1
Temp2	23	-50-50	Cool1	Baking1
Temp3	200	100-1500	Heat2	Welding1
temp4	330	0-450	Heat3	Cooking1

Figure 2-4 Relational data model representation

Querying a data entity is therefore provided by its row-column address, i.e. querying. Although this inherits slower data access time and more memory space, it also brings ease of adaptability and flexibility in design and implementation (Rembold et al, 1993).

2.2.3.2. Database structure

Additional to structure of access to one data entity, other impacting features in data storage regard the overall structure of the whole space of data entities of a system (Rembold et al, 1993). For example, redundancy of storage causes one data entity to be stored in more than one memory space. Other features of interest include access speed, updating, and maintaining (Rembold et al, 1993). Four database structures are mentioned by Rembold et al (1993) to be different in manufacturing operations; loose connection of independent databases, centralised, interface, and distributed database.

In loose connection of independent databases, however old fashioned for its little or no consideration for integrity and relationships between data entities of different parts of information system, yet due to their fast implementation and high access speed, are used in specific applications.

A centralised database has the highest degree of consistency as well as lowest degree of redundancy and so is commonly known as the fittest structure for generic data models. However, high integrity seeks accurate maintenance after any change of data (Clement et al, 1995). Therefore, access and manipulation for instance updating and maintenance could be quite slow for its multi-user nature (Rembold et al, 1993).

Interfaced and distributed databases are similar in their domain-centred structure with the difference of not having any access between domain databases in distributed database structure. The advantage of these semi-centralized databases over centralised ones is in their faster access speed (Rembold et al, 1993).

The above given perspective shows opportunity to the data modelling and database structuring methodologies that aim at overcoming time constraints in data integration applications to compare and trade-off data querying efforts of centralised databases with distributed and logically integrated data collections (Cecelja, 2002).

2.2.4. Data communication

Similar to the discussion of the earlier section on data storage, data communication is a fundamental part of data integration (Hannan, 1996). For the variety of data communication networks at least in the area of computer integrated manufacturing (CIM), data communication has also a fundamental impact on the efficiency of data integration systems (Hannan, 1996). According to Rembold et al (1993), the key role in this efficiency is played by compatibility of communication interfaces between data integration components in issues such as; communication medium, data format, and the most important, communication protocol.

Wireless, wired (usually copper), and fibre-optic conductors are three major media for data communication with differences in range and cost depending on the signal transmission capabilities (Tanenbaum, 2002). The most popular medium among the three is still wired communication.

The data format indicates the amount of byte overhead which is required for transmission of each value of data entity (Tanenbaum, 2002). For example, an integer value takes fewer of bytes than a floating point value.

Data communication protocol has a wide variety briefly covered by (Tanenbaum, 2002). By increasing the scale of complexity in connectivity between devices and central controllers (Kamen, 1999), the importance of networked communication in CIM has increased (Bauer et al, 1994), and as a result, direct communication between two individual components (Rembold et al, 1993) has faded out. For a number of advantages discussed in the following subsection, this review covers network-based industrial data communication protocols.

2.2.4.1. Data communication networking

The advantages of networked communication mentioned by Hannan (1996) include; resource sharing due to availability of data to all components, fault tolerance as recovering data losses are possible via backed up contents by other component(s), increased economy because of cheaper processing power of communicating processors, better communication between computing workgroups, and finally flexibility in linking more local computers, extending and (portioned) upgrading of the network.

Common industrial data communication networks that connect sensors and actuators to controllers, analysers, graphic terminals etc., include three levels of networks; the first for sensor devices, the second for control, and the third for plant level, as shown in Figure 2-5 (Kamen, 1999).

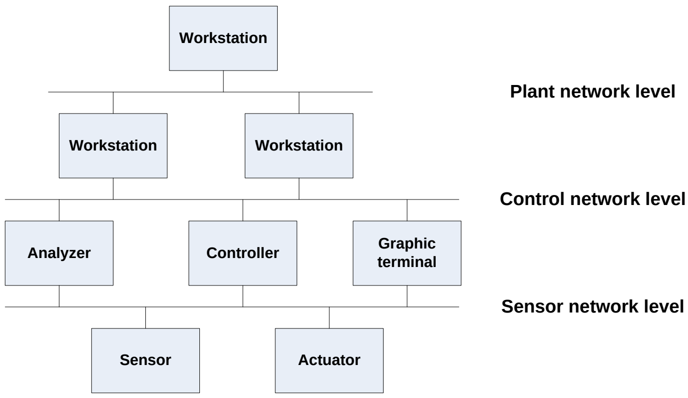


Figure 2-5 Three levels of industrial data communication networks

Although at each level different networking technologies have been defined and used for many years, such as Profibus (Siemens, 2005), and ControlNet (ODVA Inc., 2008), to name a few, Ethernet (IEEE SA, 2008) has dominated among other contenders in communication technology application domains (Kamen, 1999). Ethernet, or IEEE standard 802.3, defines how to pack information in signals and how to use a network medium (usually wired) to send/receive signals. In other words, Ethernet manages transmission of digital content between computing devices in networks of multiple nodes. Ethernet has proved its utility in performance, flexibility, ease of installation and administration (Hannan, 1996) between network protocols.

Another procedure which is not covered by network communication protocols like Ethernet is architecting the digital information, or how to format actual data taken from a device and the address of destination device before it is sent by network communication protocol (Kamen, 1999). The OPC protocol manages this data interfacing in a globally standard way so that implementation network devices, e.g. controllers, computers, databases servers, from different vendors could simply and flexibly use it to handle data communication without having to adapt with different definitions and architectures of different data sources (OPC Task Force, 1998). As shown in Figure 2-6, OPC and Ethernet form two standard layers for data communication.

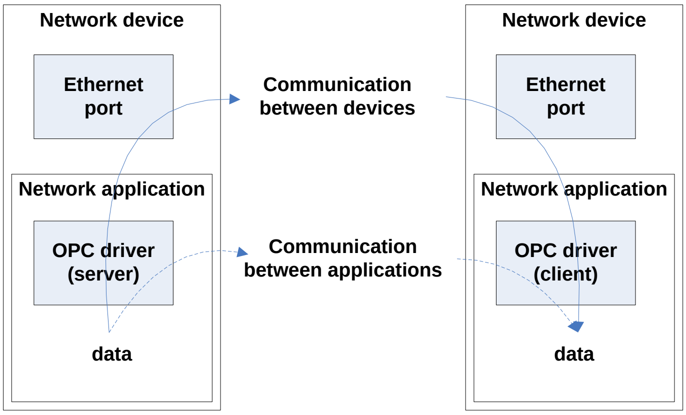


Figure 2-6 Ethernet and OPC layers, the actual path for data communication is through Ethernet port

2.2.5. User interface

Bauer et al (1994) mentions that '*A good user interface aims to facilitate synergy between man and computer*.' In data integration systems this could be achieved by handling necessary interaction between users and system (Bauer et al, 1994). Two groups of users usually have interaction to a data integration system (Rembold, 1993); users who set up the system and prepare its execution, and users who try to understand and describe the context of the problem of the system.

2.2.6. Data processing and decision support

This final component of data integration feeds information to the business model so that it could produce the Key Performance Indicators (KPI), which in turn provide a better insight into the system, enhance the confidence about the system, make decisions, and control the work flow. Therefore, modelling performance factors should effectively help the process of system monitoring and decision making (Viswanadham et al, 1992). For example, understanding the spare capacity in the system to undertake other jobs, or prediction of the effect of changing resources or their conditions are among important decision making tasks for a product manager (Viswanadham et al, 1992).

Viswanadham et al (1992) further classifies performance evaluation methods into two classes; performance measurement, that is carried out on existing and operational systems by means of frequent data collection and analysis to monitor KPIs, and performance modelling, which could be either analytical or simulation.

Analytical models feature mathematical formulae or procedure that could be solved in closed form or by using numerical techniques. This could on one hand support the adequacy of performance results due to the possibility of creating a specific and accurate model of the real system, and on the other hand put disadvantage of expertise cost and intractability if model is too detailed (Viswanadham et al, 1992).

Simulation modelling, however, has been popular within manufacturing monitoring applications due to the easier implementation of details of manufacturing processes and estimates of performance measures. The benefit and issues of simulation modelling methodologies will be discussed in more details in the next chapter.

2.3. Efficiency of the data integration systems

Both available technology and objectives of a data integration solution play important role in the deployed methodology and the measure of efficiency of data integration systems. For example, an information integration system in multi-sensor systems was proposed by (Iyengar et al, 1994) which focused on the architecture aspects of distributed sensor networks with regarding to sensor data clock synchronisation, communication costs, and fault tolerance. Data Integration was also addressed by (Ibrahim et al, 2005) who proposed a query processing algorithm as a semantic solution in wireless smart sensors. In both solutions processing of entered sensor data necessitates inefficient method of depositing the data into a database accompanied by all the acquired data attributes which significantly increases computational cost. Other researchers have focused on the security of data over sensor networks (Ozdemir, 2007), efficiency of data transmission (Zhou et al, 2006), dependability of sensor networks (Thuraisingham, 2003), independence of sensor data (Mueller et al, 2007) etc amongst others.

The need for a robust and agile tool that helps production managers make quick decisions in Fast Moving Consumer Goods (FMCG) industry has been discussed in (Mills et al, 1997) and (Mousavi et al, 2007). (Bauer et al, 1994) put consistency, precision and timeliness as the three prior properties of information on which an informed and accurate decision making process could rely. (Bauer et al, 1994) also added simplicity and convenience of monitoring operations for manufacturing personnel to the efficiency factors of a monitoring system.

The focus in this thesis is to provide a flexible and easy to use data integration environment which will facilitate quick decision making. In the following subsections, a brief description of how these efficiency factors of data integration system could be measured is given.

2.3.1. Promptness

In reality, several features in the architecture of a data integration system may affect the speed of access to data. Frequent availability of data is required in many monitoring and decision making applications. Centralised architecture, data structure, and scale of the application are among the reasons which could slow down the transition of the data from source to the buffer and from buffer to the client. Promptness of data retrieval mechanism between the stages of a data integration system could be measured by attaching a timestamp to the acquired data at each stage and evaluating the differences between timestamps, so called ‘loganalysis’.

Being real-time, a data integration system is meant to “*pertain to the performance of a computation during the actual time that the related physical process transpires*” (Boyer, 1999). In other words, the series of computation processes throughout the components of the data integration system starting from the data acquisition up to the data apparatus could occur with such a speed so that missing the fluctuation of acquired parameters of the physical system does not lower the performance of the desired measures of the system below a desired level. In this sense, real-time remains a relative property associated with the system. For example, in data integration for transportation systems, collection of location data from the cargo fleet on the road for measures related to the mode of transportation could be considered in real-time if the rate of sampling location data is every 30 seconds (Byon et al, 2009). This rate is however too slow for measurement of critical parameters in an air compressor testing system (Sudha, 2009).

Nevertheless, the important fact is about finding out exactly what is happening on the work flow to be able to improve the performance of the work (Cecelja, 2002).

2.3.2. Accuracy

Transition of values of data from sensor outputs towards data storage and data processing components without losing the required resolution of the original signal is one aspect in the accuracy of a data integration system. The other aspect is in the functionality of the components to generate services as expected. These two aspects are generally considered and usually satisfied by today's data integration system solutions. However, an amount of computational overhead and therefore complexity of the system may be traded off for achieving accuracy depending on both dynamic data structure and database architecture.

2.3.3. Communication

Scale of data integration systems in most of the applications necessitates use of data communication between several data handling modules which run in parallel and need data from each other. Passing messages which wrap the data within some metadata is an established method to simplify data communication between concurrent software executions on different processors (Folino et al, 1998). Depending on the data communication technology used, one complete data transfer may involve two messages; one for request and the other for response. Communication media and speed, security and compatibility of the communication protocol with regard to the application environment affect the effectiveness of the data communication.

2.3.4. Complexity

In literature a number of measures and criteria, which are discussed at the following, have been defined as indicator of the computational complexity of computation problems (Aimin et al, 2008; Bansiya et al, 1999; Roca, 1996; Harrison, 1992) (Penzes et al, 2002). Aimin et al (2008) mention the complexity of software architecture of the information system, to be a contributing factor to the computational complexity. Roca (1996) mentions the deficiencies in the execution code as a potential contributor to computational complexity. In both cases structural complexity which is determined by the structural characteristics of either the overall problem or the actual algorithm are the main culprits for computational complexities (Harrison, 1992).

Bansiya et al (1999) mention the relationship between complexity of comprehension and implementation of the computational solution as a determinant of computational complexity. This approach suggests an algorithm oriented measurement.

Energy consumption has also been connected to the computational complexity. Martin (2001) proposed a complexity measure comprising of both energy and time efficiencies of computation based on the physical model of implementation of elementary operators or gates. They proved that $\mathrm{E} \times \mathrm{t}^2$ metric (E: computation energy, t: computation delay) is a reliable measure. Both energy and time are calculated based on the electrical features of the circuit which handles the computation.

Peng (2008) applied Landauer’s theorem (Landauer, 2000) to measure the consumed energy of the computational problem by means of the entropies of the problem’s initial state and final state. This method gives an indication of the lower bound of energy consumption for solving a computational problem without using the knowledge about the implemented algorithm.

The scope of this thesis sought an architectural and methodological view of the computational complexity analysis. Different algorithms may be used to achieve the same goal under the same architecture. Therefore, algorithm-oriented methods were disregarded.

2.4. Proprietary data integrated systems

In the following sub-chapters data integration structures which have been adapted in existing data integration systems will be discussed. In this section an overall presentation and analysis of their infrastructure and architecture is made. In addition, the efficiency of data integration systems and structures will be discussed. Furthermore, the advantages and disadvantages of the proprietary data integrated systems that are currently adapted in industry will be discussed.

2.4.1. Honeywell

Information Management, as an operations application within Honeywell's process solutions, was introduced to facilitate transformation of data to knowledge by providing interfacing to the modern data sources, and a formula management user interface (Information Management, 2009), a centralized formula management infrastructure (Advanced Formula Manager, 2009) in hand with a centralised database server (Uniformance PHD, 2009) which accommodated calculation results. However, a robust and consistent near real-time access to the stored historical data required integration of Information Management with Honeywell's Experion® Process Knowledge System (PKS) (Honeywell, 2009). Such massive set of integrated solutions was not yet prepared to monitor operating events of the process plant as an Event Monitoring solution was required (Event Monitoring, 2009).

2.4.2. Siemens

Plant Intelligence was offered by Siemens to link the two levels of data between SCADA, i.e. machine level, and the manufacturing execution system (MES), i.e. corporate level (Siemens, 2009) which by definition limited the scope of data integration to what each SCADA could cover. Scalable integration solutions required communication between SCADA level modules and archived data through server points (Siemens, 2009). Although monitoring the entire plant was also offered by an alternative approach, the link to corporate level was as limited as a dedicated interaction with the ERP systems (Siemens, 2009).

2.4.3. Rockwell Automation

Through machine, line, or plant performance solutions of Rockwell Software's information solutions (Rockwell Automation Inc. 2010), real-time data was introduced to be available to the monitoring and analysis tools. However, efficiency via an optimised number of data input points was only introduced to be available through their product called Value Map in Manufacturing Assessment and Planning (MAP) assessment services which were operational consultancy and process efficiency identification tasks (Rockwell Automation Inc. 2010). An automated computational efficiency measurement tool which could control the amount of computational overhead of the overall information system by introducing the less important real-time data was not introduced.

2.4.4. ABB

System 800xA Information Management Historian, an information management solution within ABB's control systems, was introduced to be able to reduce data storage resolution in order to increase storage space (ABB, 2010). However, no intelligent and efficiency-aware mechanism was mentioned to support this configuration.

2.4.5. OSIsoft and SAP

The use of an integrated suite of information management solutions between SAP's NetWeaver Master Data Management (SAP AG., 2009) and OSIsoft's PI System (OSIsoft Inc., 2009) was introduced to lead to improvements in business decision making based on the access to the real-time data from the plant floor. This however did not enable any functionality in the coalition to help avoiding dump of useless data in centralised databases. Instead, extra efforts were sought in the integration to reduce number of duplicate data entries (OSIsoft Inc., 2007).

2.4.6. Oracle

Oracle Data Integration Suite was offered as the main component of the Oracle Fusion Middleware data integration solution to manage data movements and data profiling (Oracle, 2010). However, low impact real-time data integration and continuous data availability solution was not guaranteed in the suite. On the other hand, two Oracle's conjoint products; GoldenGate (Oracle, 2010) and Product Data Quality (Oracle, 2010) helped with offering real-time access to data and enterprise level data transformation, respectively. In order to achieve fully real-time and quality assured data integration, the combination of the three aforementioned products seemed demanding. High volumes of expertise, effort, and expenditure could be sought in the implementation of such massive integration projects.

2.4.7. Advantages of proprietary data integration systems

- Comprehensiveness; a wide range of services and functionalities were introduced and offered by proprietary data integration products. Some of the offered products may cover future data integration plans as well as overcoming the current problems.
- Rapid development; some of the proprietary data integration products may be chosen to be implemented for the feasibility to achieve quick solution developments with a lot of ready to use materials and functionality.
- Training and support; All proprietary data integration solution providers offered training services for development. Almost all of them offered consultancy services for definition of the original problem. Further support and training for maintaining the product and services also were offered at some costs.

2.4.8. Disadvantages of proprietary data integration systems

- General purpose; featuring services and functionalities more than could be required in many information automation applications, it would not be easy to find the cost effective product and then the correct route between many options in one product. A straight forward environment which draws the focus of the implementation on connections to data points and definition of the performance model i.e. key performance indicators would speed up easier product selection and quicker fits.

- Computational overhead; many of the solution providers - including PLC makers - started from SCADA level automation, and then built up more comprehensive information management systems on top of the legacy automation solutions. The final product was therefore not an optimised architecture of components which means a surplus cost of implementation and execution for no valuable output. An up to date architecture which is built based on the scalability of data integration and efficiency of execution could reduce computational cost by employing less components and data.
- Computationally sensitive; it did not appear in any proprietary data integration solution, that computationally more expensive data influenced the determination or estimation of the importance of data in an attempt to monitor or control its computational overhead. Appending such service to data integration system, if this new service itself is not computationally costly, could help elimination of potentially many data integration efforts.
- Proprietary DAQ firmware; NI's Labview needs Labview Driver software to be installed as a run-time engine (National Instruments Corp., 2007). Microstar Laboratories (ML)' DAP acquires data from only ML's DAQ hardware (Microstar Laboratories Inc., 2009). A generic method of introducing sensor data to DAQ software with no need to particular manufacturer's DAQ firmware would help applications to save on keeping their legacy data acquisition hardware.
- Too technical; some existing solutions needed high level of expertise in dynamic specifications of data collection and connection to data sources, which may not cover the fields of expertise of a product manager. A high level user interaction which avoids complicated data

connection definitions and resolves them by automatic detection, typical settings, or other approaches would attract a wider area of applications.

2.5. Design approaches of data integration system architecture

2.5.1. Layered architecture

The layered approach to define architecture for an information system can be described as defining different groups for the elements of the information system in which, elements of each group are capable of providing different functionalities and services, and could communicate with the elements of the other groups through defined communication behaviour (Bauer et al, 1994). Such description carried the key concepts of the layered architecture as layer, entity, service, function, and protocol more comprehensively described by (Bauer et al, 1994).

The main advantages of a data integration system built based on the layered architecture appear in its flexibility and modularity (Bauer et al, 1994) which brings with itself efficiency and effectiveness of integration through standard services and protocols.

Bauer et al (1994) address the standardisation of an information technology reference model for shop floor control by definition of a layered architecture based on the shop floor layout i.e. factory, cell and workstation layers, and the main shop floor tasks i.e. production environment design, scheduling, dispatching, monitoring, producing, and moving.

A three-layer metamodel has been suggested by Mannarino et al (1997) in which building blocks of each layer were constructed based on the lower layer. In their proposed architecture, Mannarino et al (1997) defined generic constructions of the production environment, such as functions, resources, and tasks were concluded in the lowest layer, called metamodel layer. This only first layer covers all components of the environment for which this thesis suggested a solution for data integration. Although, the scope of the metamodel-based solution proposed by Mannarino et al (1997) was the entire information system of enterprise integration problem in organisational level, the benefits of the layered architecture appeared in its openness and accessibility to the attributes of the primitive building blocks of the production environment.

2.5.2. Hierarchic object-oriented design

In software design methodology, HOOD, standing for or Hierarchic Object-Oriented Design, is "*an architectural design method, helping a designer to partition the software into modules with well defined interfaces that can either be directly implemented or further partitioned into modules of lower complexity*" (Rosen). Both object-oriented and functional design approaches are supported by HOOD. In summary, HOOD is a hierarchical design approach that incorporates the notions of object-oriented design into an industrial process. It includes a notation and a design process. The formalism is supported by a set of rules which are enforced by tools.

2.6. Conclusion on data acquisition and integration platforms

In this review it has been concluded that due to flexibility and reconfiguration issues, a data integration system which gathers and transforms input data seems more feasible for efficient structure and action than the sensor network which provide input data. Therefore, the components of data integration system have been explored and efficiency issues of their functionalities discussed.

It has been discussed that data acquisition as the backend connection of data integration system to the shop floor faces both hardware and software challenges for coping with large scale real-time data collection. It has been pointed out that the limited scope of SCADA systems do not make them suitable candidates for this.

Data analysis component of data integration has been introduced as an interface between raw data and high level information processing. A big challenge here is to identify events in event-driven environments and providing accurate and sufficient information to deal with least computational cost.

For efficient data storage, both the data structure and database structure have been discussed to play role in terms of access speed and memory space. A relational data model in a decentralised database is more responsive in time-constrained applications. However, query style demand of data from relational data model could waste time.

An Ethernet network and OPC service are concluded to be good candidates for efficient data communication. However, in a generic solution for data integration, other means of data networking must be supported.

After a review on the efficiency parameters of data integration systems, some of the most popular proprietary data integration systems were briefly studied and challenged for their suitability in terms of adaptation to the objectives of this thesis. It has however been concluded that cost effectiveness, complexity, and efficiency-orientation of proprietary solutions leave space for more challenge.

Since, performance factors generated at the data processing stage were identified as playing a role in capturing and analysing event data, further studies are required to explore this relationship. Therefore, this thesis will continue with a review of issues and ‘how’s of input variable selection. But prior to this, there will be a review of performance modelling by using discrete-event simulation and modelling techniques.

3. Process simulation and modelling

The aim of this chapter is to understand how simulation and modelling could support the proposed solution in this research as a part of a decision making tool. For this, the chapter delves into how simulation and modelling has been useful in the context of the defined problem. Properties of application of simulation in manufacturing will be discussed. Dimensions of real-time and discrete-event simulation will be stressed. Finally, criteria for selecting a simulation modelling tool will be introduced.

3.1. Simulation in manufacturing

Setting up an environment which can help to demonstrate the behaviour of a model through a certain time interval can be called simulation. The simulation environment provides a means to interact with the model. Justifications for when simulation is and when it is not the appropriate tool are given by Banks (2001). The advantages and disadvantages of using simulation are also discussed by the same reference.

It is clear that simulation has been used most extensively in manufacturing than probably in any other field. Onut et al (1994) showed how simulation was integrated into a complete shop floor control system for a Semi-Integrated Manufacturing System (SIMS). They developed a framework that interfaced the simulation system with a Material Requirement Planning (MRP) system, a host computer, a Database Management System (DMS), a shop floor control system and a supervisory input system. This greatly enhanced the effectiveness and control of the manufacturing operations.

Gupta et al (2002) proposed shop floor scheduling with simulation based proactive decision support in a highly manufacturing complex system where multiple product parts, sequence dependent setup, moulding machine specifications, mould restriction’s etc with a variety of scheduling and operational choices are integrated. Gupta et al (2002) developed a simulation model that generates a feasible schedule and has ability to reschedule the system when sudden changes occur. The ahead of time system parameters provided the scheduler with the opportunity to find best schedules efficiently.

Potoradi et al (2002) also developed a simulation-based scheduling system to maximise demand fulfilment in a semiconductor assembly facility. They used simulation as an engine to generate schedules and to control various machines at execution time and also to plan for the start of materials. The schedule adapts to "unforeseen" changes on the shop floor by the use of online data availability. However, their data entry from the shop floor and planning system is not fully automated, hence the model-update is quite slow and requires an expert and is not done frequently. From the above review, it will be noticed that simulation plays a vital role in the understanding, control and improvement of complex systems. Additionally, it is observed that there is basically not much difference in the approach to applying simulation in scheduling. However, the details of the integration and framework are custom made to suit the particular environment.

Manivannan and Banks (1991) also provide a brief review of earlier attempts to build intelligent controllers for managing operations in a manufacturing cell with and without simulation. In an earlier work by Wu et al (1988), they observed that significant improvements could be made by using a simulation model to determine the future course of a manufacturing system.

In all the above cases, the gain in system performance as a result of the use of simulation has been noticeable. The area of shop-floor control and scheduling demonstrates an aspect of the capability or power of simulation. But there is more to it. Shop floor control and simulation could be performed in real-time. A number of these cases are discussed in this chapter after the weaknesses of "non-real-time" simulations is briefly explained in the next section.

3.2. Weakness of traditional simulation

By traditional simulation it is meant the approach to systems modelling and simulation that follows the methodology of Banks (1998) and which Son et al (2001), described as "throw-away" tools. This approach normally requires a simulation expert to build the model and is heavily dependent on historical data. The main drawbacks of this approach are evident in numerous published works in simulation and are summarised below with particular reference to simulation projects in healthcare undertaken by Komashie et al (2005, 2008). The main shortcomings of traditional simulation are:

3.2.1. Time consuming

Traditional simulation often requires the manual collection and analysis of input data. Mining the data and preparing them for use in a model is always time consuming. Sometimes data that is seemingly available may not be in the format usable for a simulation study. In healthcare for instance most processing time data and proportion of patients at various branches in the system are unavailable and have to be collected. Apart from being a time consuming exercise, the data obtained and processes defined are also time dependent and subject to change in short cycles.

3.2.2. Time dependent

Due to the fact that traditional simulation is heavily dependent on historical data, as the input data gets older the results of the model also become less reliable. This is a critical issue in dynamic and complex systems like healthcare and manufacturing. In some cases, by the time the modelling project is completed, the input data that was collected for the model may be obsolete and may cause doubt in usability of the results. When the input data to a model is not as reliable, then it becomes difficult to use the model to predict future events accurately.

3.2.3. Potential inaccuracy in prediction

Simulation models help to understand the operations of a system and serve as a cost effective, risk free platform for testing different configurations of the system. In addition, a well validated model of a system may be a useful tool for predicting future events. However, this is not a reliable exercise with traditional simulation which is dependent on historical data. This has been one of the incentives for researchers and practitioners to conduct research on real-time data acquisition and control systems.

3.2.4. Costly

With the expertise required and the time it takes to build and run good simulation models, the cost of keeping a traditional simulation model up to date and fit for predictive analysis would be prohibitive. To reduce this cost and providing a reliable platform for system managers to predict future events more accurately is part of the motivation for the proposing a generic framework in this research.

3.3. Why run real-time simulation?

Joshi et al (1994) compared analytical models as computational procedures which evaluate system performance and decision variables with simulation models as descriptors of logical relationships between system events and dynamics. The concern of this research is justification of using real-time simulation, i.e. a simulation which runs based on the values of the parameters of the shop floor in real-time.

Real-time provision of an indication of the cost of a running system is at the heart of the driving force for implementation of simulation in today's industry.

3.3.1. Real-time VS one-off simulation

For several years, simulation has been applied to the long-term planning, design and analysis of manufacturing systems. These models have been termed "throw away models" because they are seldom used after the initial plans or design is finalised (Son et al., 2001; Smith et al., 1996; Harmonosky, 1995). Over the past decade, however, researchers and practitioners have taken advantage of the power of simulation technology to develop simulation models that can be fully integrated into complex manufacturing systems and run in real-time. The ability to automatically generate simulation models for certain application has also been achieved (Son et al., 2001).

Harmonosky (1995) conducted a review of simulation based real-time scheduling and found the need to further explore the concept with look-ahead and what-if capabilities. More recent attempts to use real-time simulation modelling in the control and analysis of manufacturing systems may be found in (Mullarkey et al., 2000; Rabbath et al., 2000; Lee et al., 2002; Dangelmaier et al., 2006).

3.3.2. Production cost and product cost

Over the past 25 years, huge advances have been made in the use of techniques such as computational fluid dynamics and finite element analysis to maximise product performance at the design stage (Wheatley, 2009). The use of simulation to model – and improve – production and logistics processes is a much more recent phenomenon. It has come to the attention of producers that designing a device that works but then costs too much to manufacture is a new problem to be addressed by simulation (Wheatley, 2009).

3.3.3. Applications wider than manufacturing

In recent years, simulation is finding applications in several non-manufacturing environments including healthcare (Komashie et al, 2005). The labour intensive nature of healthcare systems however has made this application more challenging. Subsequently, a real-time application of simulation in healthcare is non-existent (Komashie et al, 2009). In this thesis, a framework is proposed for accomplishing real-time simulation of a manufacturing system which is adaptable to non-manufacturing systems as demonstrated by a healthcare application in (Tavakoli et al, 2008).

3.4. Application-specific simulation

In a volatile and flexible production environment where the cost of adaptation of the information system to the application is of major concern, generic approaches to producing simulation models seem to be more appealing than application-specific ones. As opposed to the generic approach to modelling and simulation which uses definition of events and processes to describe details of the modelled systems, an alternative approach has been taken to compensate for the complexity of mapping specifications of complex systems to events and processes (Askin, 1993). In such an application-specific approach, for each object such as station, buffer, material handling device, and process plan, a class would be defined with its attributes matching the specification of the corresponding object. This approach seems more straightforward in terms of the efforts for model definition (Askin, 1993). However, like in any other application specific approach, maintenance and deviation to similar applications would not be expected to be as easy and flexible as in generic approach.

3.5. Discrete-Event Simulation (DES)

In discrete-event simulation, the model is affected by its environment after certain fractions of time (Banks, 2001). The new status of the model is worked out and demonstrated based on the new collection of model inputs after each time interval. The following components are defined within the concept of DES;

- System State; State of the system could be defined as the “collection of variables necessary to describe the system at any time” (Banks, 2001). Of course, this is not necessarily including all existing variables of the system, and depends on the objectives and expectations from the system model.
- Event; Based on the description of the system state, an event in the system could be defined as “an instantaneous” occurrence that may change the state of the system” (Banks, 2001).
- Entity; An entity in a system could be referred to any object in the system which is considered and defined with some specifications which is somehow useful in the description of the system. Those specifications which are connected to the entity could be referred to as attributes.
- Activity; A collection of some events of the system in a specific time period could be referred to as system activity.

3.6. DES in real-time control

Real-time systems differ from traditional data processing systems in that they are constrained by certain non-functional requirements e.g. dependability and timing constraints or requirements. Vaidyanathan et al (1998) developed a discrete event simulation model as a daily scheduling tool. They employed a hybrid approach that integrates a scheduler and a simulation model. The simulation model plays the role of modifying the output from the scheduler, and the two together become a tool for day-to-day production scheduling.

An efficient simulation of real-time system requires a model that satisfies simulation objectives and timing constraints (Lee et al, 2001). Son et al (2001) developed a structure and architecture for automatic simulation model generation for very detailed simulation models intended to be used for real-time simulation based shop floor control. They identified two essential stages to be automated for automatic simulation model generation: System specification and the associated model construction. In this work, Son et al (2001) proposed a methodology for generating a simulation model (based on Rockwell Software Arena) from a resource model (in MS Access 97) and a Message-based Part State Graph (MPSG) based shop floor control model. This was made possible because the Arena simulation software supports Visual Basic Application (VBA), which enables application integration and automation. Lee et al (2001) undertook the development of a modelling methodology to efficiently model real-time systems to satisfy given simulation objectives and to achieve arbitrary timing requirements. Mousavi et al (2008) proposed the framework and implementation architecture of a combined real-time shop floor data collection (monitor) and DES. They highlighted the practical implementation and potential benefits of using predictive (multipass) simulation in combination with real-time data acquisition (DAQ) in dealing with the complexity of manufacturing environments.

The above review provided some evidence of the extent of the application of real-time simulation modelling in the manufacturing industry. Evidence from comprehensive reviews conducted by Brailsford (2007), and Eldabi et al (2007), indicate that the application of real-time discrete event simulation is yet to be extensively adopted in industry.

3.7. Model development approaches based on DES

The structure of information which can be represented in DES allows a model developer to look and construct the model taking one of the two different approaches (Banks, 2001);

3.7.1. Event scheduling

In this approach the focus is on events and their effect on system states. Model behaviour is defined the way so that it would accept certain values at certain times (i.e. events) as input and runs through the procedures which work out the new status of the system.

3.7.2. Process interaction

One may focus on developing entities with a number of attributes for them which may change the status of the model as they exist in the model. The life cycle of the entities and the whole model is then affected in an exchangeable way.

3.8. Simulation modelling tool capabilities

To be able to evaluate the performance of complex manufacturing systems in sufficient detail in real-time or fast-forward operation modes, a series of features are expected in forming the simulation modelling tool. These essential features are discussed at the following sections.

3.8.1. Object-oriented modelling and simulation

While creation and modification of simulation models - including writing, testing and debugging the simulation programs - are time consuming tasks, object-oriented modelling approach helps with the code development, model management, and reuse of the developed code (Joshi et al, 1994). The justification and advantages of object-oriented programming under four key concepts of encapsulation, data abstraction, dynamic binding, and inheritance is well-established (Masini, 1991).

3.8.2. Real-time connectivity

Aforementioned examples in section 3.6 have introduced some situations in which the simulation model is connected to an external data source. For the target of this thesis, it is important to identify and implement approaches that can guarantee connection between the simulation model and the real-world in real-time. Two issues are very important in the establishment of real-time connectivity;

3.8.2.1. Data repository

It seems unavoidable in computational terms to forward a sampled value of an input variable straight onto the simulation model and without storing it in an intermediate memory space as could be an ideal real-time data connection. Increased number of data necessitates use of organised memory spaces or so called databases. Database servers are applications which help with prompt provision of space for allocation and restoration of data. However, they bring extra computational overhead for execution and querying services (Gennick, 1999). Particularly when long term data storage is not necessary, more temporary in-memory data repositories like queue structures are helpful.

3.8.2.2. Data communication

Depending on the development environment of data acquisition and simulation model, appropriate hardware and software to design and develop code for communication with external resources is required. The embedded messaging service of the proprietary operating systems is easy to use. Data querying services and compatibility with database servers may appear as a delay-concerned issue.

In terms of communication channel, wired Ethernet is rather fast and reliable (Tanenbaum, 2002) in trade-off between cheaper and less secure communication media and protocols such as wireless Bluetooth (Ferrigno et al, 2005).

3.8.3. Fast-forward simulation

One important feature in a simulation model is the ability to execute the sequence of processes of a faster pace than in real-time situation. This feature helps with using the simulation model for real-time control (Joshi et al, 1994), future prediction and decision making. Joshi et al (1994) describe a recursion model and hybrid approach as two of the main approaches that researchers have taken towards facilitating fast-forward simulation. In general, reduction of the total simulation run-time must not ideally result in sacrificing the desired statistical accuracy. This may be achieved by taking advantage of parallel architecture of computer hardware and well developed concepts in probability theory, stochastic processes and statistics (Joshi et al, 1994).

3.8.4. Simulation language capabilities

File manipulation, event calendar maintenance, statistical calculations, built-in animation constructs are among the important capabilities that are expected from a programming language for design and development of simulation (Harmonosky, 1990). Although these features could be manually coded using general purpose languages, they usually are included in the simulation language. However, among people of different disciplines who need to program simulation models, only industrial engineers may have knowledge and expertise to use simulation languages (Harmonosky, 1990). This is a trade-off with the fact that important features like real-time connectivity may not be provided by all simulation languages (Harmonosky, 1990).

3.9. Model conceptualisation and abstraction

In model conceptualisation, as one of the important steps in a simulation study, the ability to abstract the essential features of the system is of fundamental importance as it initiates the enrichment and elaboration of the model (Banks, 2001). The task involves efforts to enhance the quality of resulting model as well as to increase the confidence of the model users in the application of the model (Banks, 2001). Such efforts can be costly when at the same time they may never reach the expected quality. Having attracted the attention of many researchers, this issue has been tackled from several angles including data mining techniques which can eliminate interaction of the model with those features which are not essential.

3.10. Summary of process simulation and modelling

In this chapter, after reviewing the existing approaches on generation of simulation models, it has been comprehended that real-time discrete-even simulation could be the appropriate method to represent the status of a system.

Although it has been figured out that efforts could be spent to create a simulation model that can generate and indicate system performance in real-time, no evidence has been found to indicate that computational efforts in simulation can be fine tuned through efforts of the previous stages in data integration. The next chapter reviews input selection approaches which may be able to link these issues.

4. Input Variable Selection (IVS)

In this chapter the existing concepts and methodologies which could help with the reduction of the computational cost of data integration and simulation solutions discussed in the previous two chapters is introduced.

The main concerns of Fast-Moving Consumer-Goods (FMCG) manufacturing industry are product quality, delivery time, and overall production costs. Consequently, companies that are faced with challenges of the ever more competitive global markets have realised the importance of accurate account of the key performance measures. This demand could involve reception and processing of a large amount of data from shop floor activities which in turn could add burden to the computational efforts in terms of both computational cost, and performance of decision making. This effort may be reduced by identifying and filtering a number of lesser important inputs.

Thus, the outcome of this chapter complementarily helps with understanding that how a data integration solution can be developed to encompass all efforts of data acquisition and aggregation from the entire data sources in real-time while at the same time featuring quantified measures of the importance of the data sources. The term “important” in the era of variable selection has been translated into two separate notions of “usefulness” and “relevance” (Kohavi et al, 1997; Blum et al, 1997). Such formalisation is directly influenced by the classification of the types of measures that variable selection methods use. Therefore, existing works and concepts under the name of Input Variable Selection (IVS) and feature selection are analysed; the scope of the varieties of IVS methods is compared against the requirements of the problem. Sensitivity analysis is also described as a tool for variable importance estimation.

4.1. IVS and feature selection

The role of IVS and feature selection could marginally differ. This section aims to clarify these two roles and understand their concerns so that use of IVS in the scope of this research work could be justified.

It is helpful to distinguish the difference between the problem of IVS and feature selection before continuing with the varieties of IVS problems.

Feature selection is a well known problem addressed by a large number of research and literature (Guyon et al, 2003; Unler et al, 2010; Chen et al, 2010; Gao et al, 2010; Bunke et al, 2010; Gunasekaran et al, 2010; Huang et al, 2010; Zhang et al, 2010). The objective of this thesis is to propose techniques for decide which input data sources are crucial to report on system status. Thus, the aim of the IVS in this thesis is to deal with individual input data sources. This allows space for discrepancy between the two problems of IVS and feature selection from at least two perspectives; one is the nature of input variable and feature, and the other is their selection methodologies. Both aspects are explained in the following two subsections.

4.1.1. Input variable and feature

Although in many literature the terms "*variable*" and "*feature*" are used and treated equally, with marginal differences input variable differ in nature from feature on the basis that the former is generally referred to as a piece of information about the system which is later used to continuously represent the model of the system, whereas the latter is a piece of knowledge about a series of data in the system. Feature may be locally and temporarily created to help with decision making not necessarily about the performance of the system but for understanding some specific behaviour in the system. Hand et al (2001) highlights the fact that although non-sequential data may be sufficient for a given data mining task, sequential information of input variables could be critical in certain applications. This conceptual difference is shown in Figure 4-1.

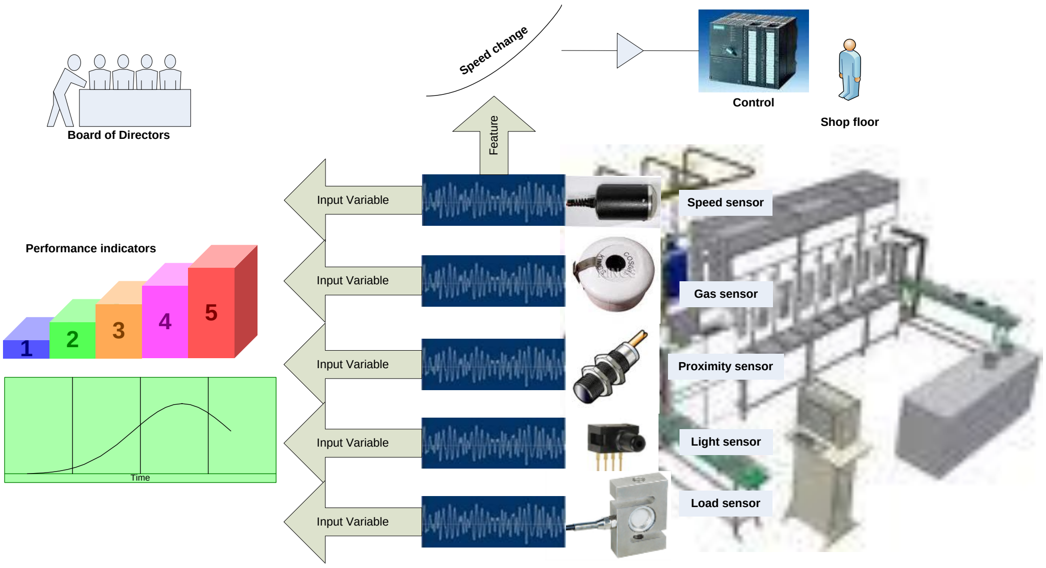


Figure 4-1 Input variables are used differently compared to features

Input variable is a direct result of aggregation of raw input data from data sources. Feature, however, may be extracted by mining of data which is obtained from a collection of input variables. Conversely, features may be constructed by application of acknowledged functions to the input variables in order to generate new variables or features. In this sense, features are created to overlay input variables for decision making process. The two feature creation scenarios are explained through examples in the following subsections.

4.1.1.1. Features derived from input variables (feature construction)

Features of a system under monitoring may be deviated from the initial input variables of that system that are built upon the raw sensor data, similar to the intermediate level data fusion in categorisation of data fusion levels by Acheroy (1999). For example, Kwak (2010) described input variables with the term initial feature candidates which are collected based on the literature and represent the shop floor status and job characteristics. He then used transformation and combination of these original and primitive features to extract the main features.

Moreover, in order to detect fault in rechargeable battery, Park et al (2009) defined two variables. These two variables are capacity and cycle life that measured by amount of charge and the number of complete charge/discharge cycles before fall of the nominal capacity below a certain value. However, to shorten the measurement cycles and therefore accelerate decision making, they devised new variables by combining the derivatives of the two variables with different orders.

Blue arrows shown in Figure 4-2, symbolise the extra efforts spent for conversion from input variables to features. Depending on the type of this process, computational overhead can be expected, which as well as computational cost may affect the decision making process particularly from the real-time processing point of view.

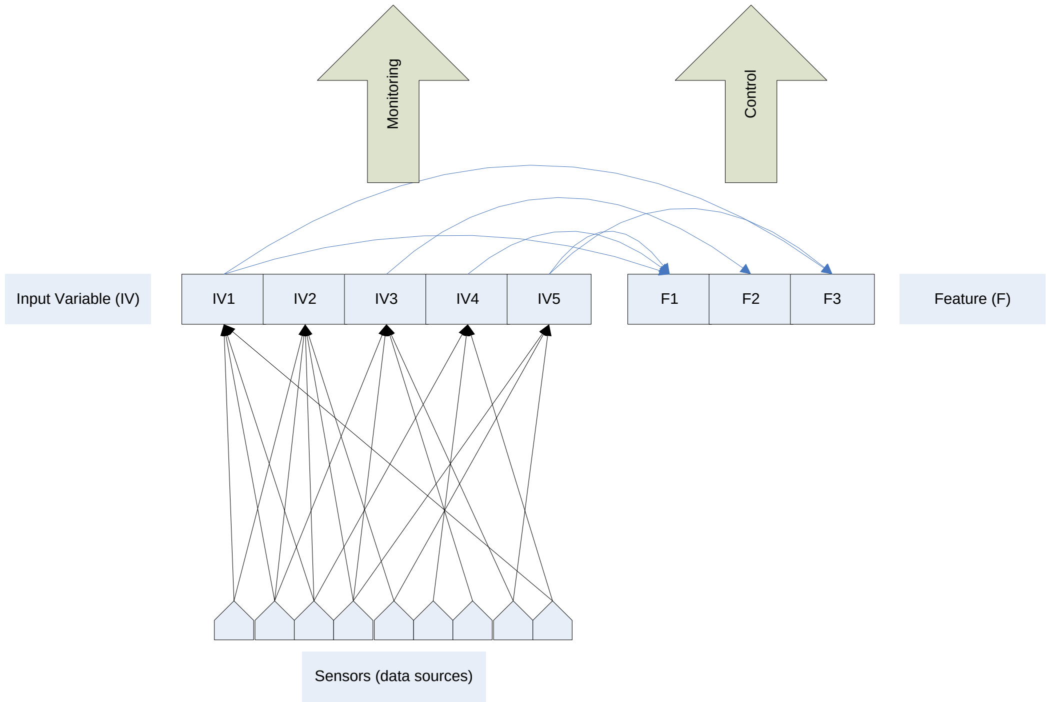


Figure 4-2 Feature construction from input variables

4.1.1.2. Features based on data mining (feature extraction)

Studying a series of acquired data of appropriate size can assist practitioners to figure out features of the data sources which are not previously known and actionable like input variables. In order to understand complex characteristics of bioprocesses and enhance production robustness, Charaniya et al (2010) applied descriptive (e.g., frequent pattern discovery, clustering) and predictive (e.g. classification, regression) pattern recognition methods. Charaniya et al discovered significant trends in process data sourced from archived temporal records of physical parameters and production scale process data.

Kang et al (2009) developed a Virtual Metrology (VM) system to predict every wafer's metrology measurements based on production equipment data and metrology results. They collected four summary statistics, such as mean, variance, minimum, and maximum value from each sensor of the two etching processes of a Korean semiconductor manufacturing company.

As shown in Figure 4-3, data mining and feature extraction require additional efforts for data warehousing (represented by database) and historical data processing (represented by blue arrows) because they are heavily reliant on historical data. This brings extra computational costs and is likely to affect the real-time aspect of monitoring.

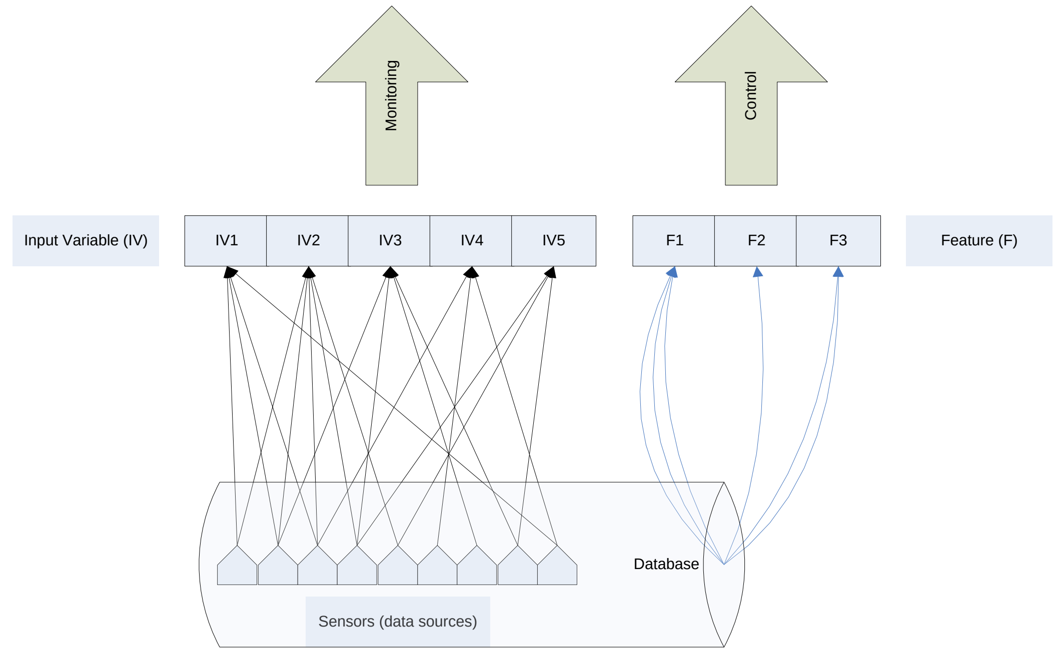


Figure 4-3 Feature extraction from raw data

4.1.2. Input variable selection and variable transformation

In the last two sections, it has been reviewed that there are cases that producing input variables which help making sense of the initial (or raw) measurements did not seem enough for variable selection and decision makers tended to spend extra effort to construct new variables (or features) upon the original variables. One important case is dimensionality reduction (Buchenneder, 2007; Dias et al, 2009; Qi et al, 2006; Mladenić, 2006; Zaman et al, 2009). This was regardless to the temporal nature of the process and measurement. As a matter of fact, in cases that the measured data was time dependent, stored data in data warehouse was used for variable (feature) construction.

As one of the key objectives of this research work, being to maintain the data integration for decision making to real-time conditions, it is feasible to study and work on IVS methodologies that tend to allow for the least computational overhead. If by using an IVS method reducing the number of input variables causes creation of new variables then the contributed computational effort may compromise the savings the IVS causes. Therefore, IVS methodologies which do not generate new subsets of variables but select among the existing input variables are studied with more interest.

In the next sections, the main methodologies which address the problem of IVS are introduced under two groups of Derived Variable Methods (DVMs) and Original Variable Methods (OVMs) and their shortcomings with particular emphasis on computational costs are discussed.

4.2. Derived variable methods

Some of the IVS methods measure performance that can be achieved by using derivations from the original inputs to select and build new variable subsets are reported in (Brodersen et al, 2010; He et al, 2010; Kim et al, 2010; Lavrač et al, 2010). The term “derived variables” has been used for the same purpose in data mining literature including (Hand et al, 2001) that suggest regression-based techniques including Projection Pursuit Regression and Principle Component Analysis for variable transformation. The former technique is one of the common multivariate regression techniques, and the latter is a special case of the former. Regression is thus known to be behind many dimensionality reduction methodologies.

Another group of methods that aim at lowering number of data entities by replacing a group of similar ones with a representative data entity are known as cluster analysis (Jain, 2010; Mirkin, 2005). Suitability of both regression and cluster analysis methodologies are discussed in the following two sections.

4.2.1. Regression

The task of estimating a map from a number of independent variables to a dependent variable whose value is dependent to the values of independent variables is defined by Hand et al (2001) as regression. In this thesis independent variables are input variables to the model, and dependent variables are interpreted as model’s performance outputs. Regression methods are adapted in two situations;

1. either requirement to predict the value of dependent variable on new values of independent variables when no predictive model between independent variables and dependent variable exist,
2. or when a new set (usually fewer in number) of independent variables are expected to replace the original set with the same effect.

Under the generic assumption of this thesis, the heterogeneous nature of distribution of data that represent input variables prevents making assumption about the structure of relationship between the independent variables and dependent variable. This leads to considering nonlinear and non-parametric regression methods. Banks (2003) reports that by growing the dimension of input variables the number of possible regression structures increases faster than exponentially. This issue contributes to extreme unreliability of regression methods.

(Uysal et al, 1999) reviewed six of the most popular non-parametric regression techniques with different characteristics. Their comparison was based on the following five properties;

1. Memory-based; methods with this property do not actually use data for training a function. Instead, they simply store all data values - which are used for training phase – in memory so that in prediction phase the nearest value of the dependent variable could be worked out by some sort of similarity criteria (Uysal et al, 1999). Obviously, these methods are computationally costly as for they normally have large number of dimensions and sample sizes that require continuous access to a large memory space during data processing. Another pitfall of these methods is the lack of a set of variables to replace the original set.
2. Recursive partitioning; methods with this property divide the space of training data set to partitions in a recursive manner with one selected feature dominating each partition (Uysal et al, 1999). The new predicted value is then provided by examining the new data set in accordance

with the selected features of the same recursive order. Effort is required to decide on the feature and its associated parameters in order to reach an optimum number of partitions (Uysal et al, 1999). This could be very application specific.

3. Interpretability; this property is about ease of verification of extracted information when required by expert (Uysal et al, 1999). This is a generally useful property that could be applied to any other algorithm out of area of regression methods, too.
4. Adaptive; methods with this property are sensitive to the actual values of data of independent variables (Uysal et al, 1999). In other words, they adjust their functions and parameters according to the local values they examine. This property does not show indication of contribution to the computational cost of the method.
5. Incrementality; this term is suggested by Uysal et al (1999) and refers to the property of regression methods that could process data without loading all of the data into memory. In terms of computational overhead, this is a useful property to notice. Only one third of examined regression methods, which were not reported adaptive, were reported incremental.

Banks et al (2003) compare the performance of regression techniques by conducting simulation experiment on ten prominent regression methods. They considered change of six factors in their experiment; (1) regression method, (2) the embedded functional relationship between the data of independent and dependent variables, (3) the number of variables (dimension), (4) sample size, (5) added noise to the sample data, and (6) the portion of involved variables in the function (model sparseness). Banks et al (2003) do not recommend any of the ten examined regression methods to be applicable to all conditions. They report Recursive Partitioning Regression (RPR) to be able to cope with high numbers of dimensions (12 variables) and when all variables were involved (explanatory). They however, recommend analysts to engage in the process of selection of regression method by trying portions of data on each method that seem to reasonably fit.

From above analysis and overview, one can conclude that no particular regression method is capable to cover the scale and heterogeneity of variables in volatile industrial systems whilst keeping low computational cost. Although hints are taken about the required IVS method e.g. not to be memory-based but can be incremental, it is vital to explore other areas of dimensionality reduction method, like clustering. This discussion follows.

4.2.2. Cluster analysis

In the cluster analysis method, provided that a group of data values are "similar" according to a "similarity criteria", they can be either replaced by a new value representing the group (clumping) or assigned a unique type of label (partitioning) (Jain, 2010). As a basic example, K-Means clustering (Mirkin, 2005) divides a set of data entities to K non-overlapping clusters of similar data and each cluster is represented by the mean value of its data (called centroid). It is obvious that the choice of number of clusters (K) and similarity criteria are two main challenges in this approach.

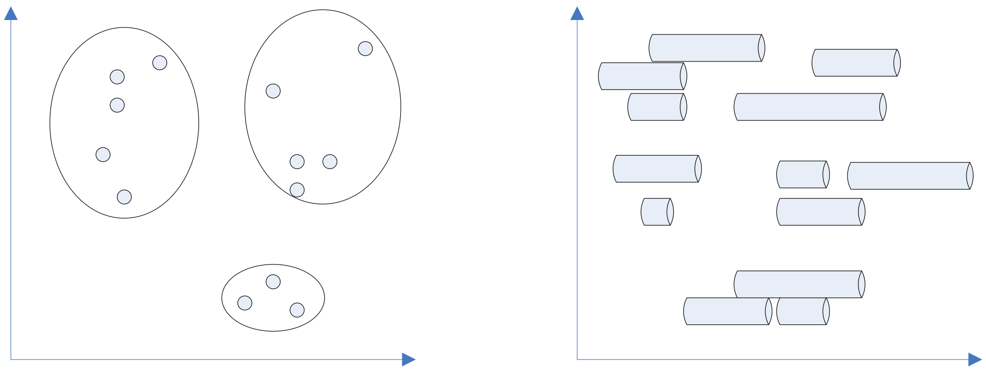
As a type of clustering method with specific similarity criteria and automatic selection of number of clusters, Principal Component Analysis (PCA) replaces a number of input variables that are correlated by a smaller number of variables which are not correlated (principal components) and which at the same time keep the same variability of the original input variables (Joliffe, 2002). Given a fixed set of input variables PCA always produces a unique set of new variables independent to the analysis of model performance factors. The issue here is at the option of using one of the actual (genuine) entities among each group for representation of the group instead of generating an artificial one - like mean value for example – (Mirkin, 2005).

Jain (2010) highlights the fundamental challenges associated with clustering as;

1. Data representation; deals with the question about the nature of the cluster. Jain (2010) points that quality of data representation affects the construction of clusters and therefore, the choice of similarity method. Although Jain rejects to accept that there is a universally good data representation, Jain admits that domain knowledge could help with clustering process. In the case of this thesis where input variables are assumed to hold heterogeneous data series in the time domain, there

cannot be a clear choice of changing and selecting a particular type of data representation.

2. The purpose of grouping; which according to Jain (2010) is related to the end goal of the user. This is true in the sense that in the case of this thesis the dimension of data of input variables is higher than the usual dimension of data required for the clustering methods. As it is shown in Figure 4-4, the added dimension which appears in the time domain, raises two potential problems; the first problem is that each data entity (to be clustered) is replaced by an input data variable in a time series. Therefore, similarity between entities becomes meaningless unless either one common indicator could be defined to represent time series of input variables like shown in Figure 4-5, or alternatively, a similarity criteria could be applied between two time series, as shown in Figure 4-6. This last alternative highlights the second problem which lies inside each time series and imposes limitation on the choice of similarity criteria between the time series. Due to different sampling rates of different input variables, the size of the collected samples are different and not necessarily time related between different samples of input variables.



(a) **(b)**

Figure 4-4 Each data entity in (a) is singular while in (b) is a time series of data

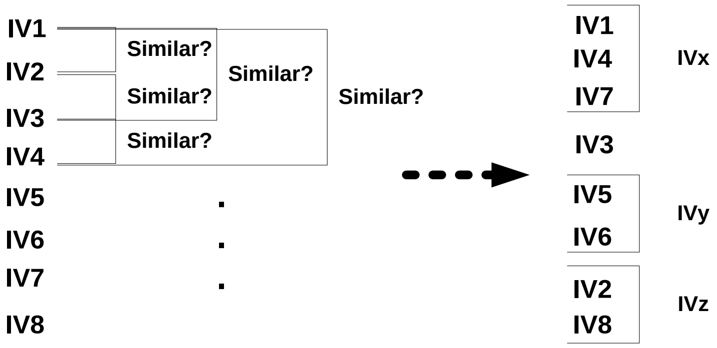


Figure 4-5 Similarity indicators represent similar data series

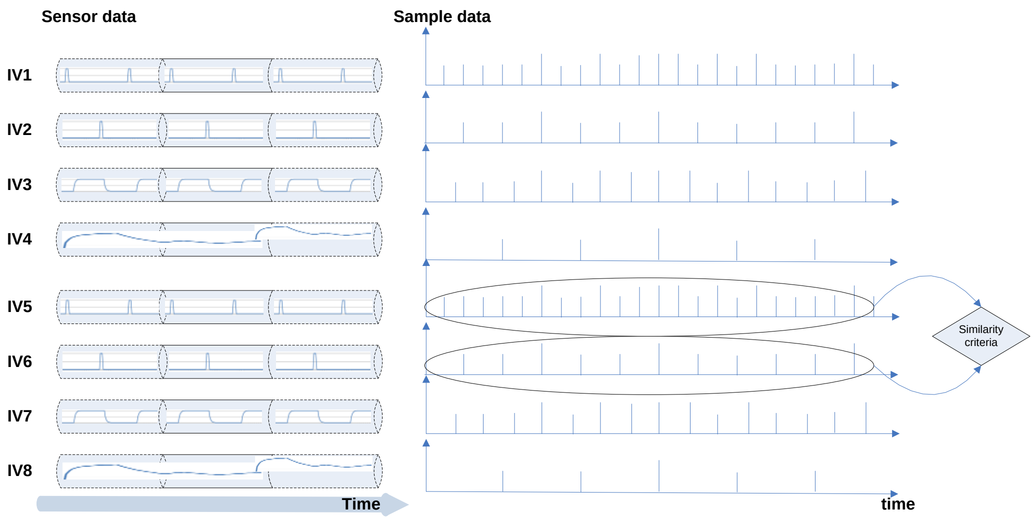


Figure 4-6 A similarity criteria decides which data series could be grouped

Solving these issues could significantly contribute more on top of the computational effort required for solving clustering problem on the input variables.

3. Cluster validity; and close relative notation “cluster stability” (Jain, 2010), deal with constructed clusters provided they fit the clustering algorithm and do not vary over different input data samples. Validity evaluation task is recommended to take place preferably after one full clustering cycle is completed (Jain, 2010). This is because cross validation between multiple cluster structures is wider used and more respected evaluation method. Stability evaluation task is suggested to take place frequently because the rate of reaching to an asymptotic stability with respect to the number of samples is reported a more useful measure.

4. Comparing cluster algorithms; Jain (2010) reports that metrics have been suggested to identify algorithms that generate similar cluster structures irrespective of the data or in spite of minor changes in the parameters of functions involved. However Jain adds that due to the unknown prior knowledge about the structure of data no best clustering algorithm exist and a diverse set of approaches could be tried before determining an appropriate algorithm for clustering at hand.

It could not yet be decided on the appropriate IVS method after reviewing the trends in regression and cluster analysis. Review of the shortcomings of the major methodologies for DVM-based IVS methods would be completed by an analysis of competitiveness of OVM-based IVS methods with respect to accuracy and computational cost. The analysis is in the following section.

4.3. Original variable methods

Some of the IVS methods - presented in this thesis as original variable methods (OVM) - find the most appropriate set of input variables according to some criteria that may or may not consider the output performance of the system. Gaweda et al (2001) and Xing et al (2003) introduced a variable selection criteria based on numeric data based fuzzy modelling, which was explored by (Takagi et al, 1985). The former performed an input-output sensitivity analysis to remove input variables holding the least maximum normalised sensitivity index in a backward elimination fashion. The latter proposed an Input Variable Selection Criterion (IVSC) function, which estimates the importance of each input variable numerically (most important input variables are kept by forward selection scheme).

In DVM-based IVS methods, for evaluation of newly (artificial) created variables, values of original variables are required to be acquired as often as the new variables are evaluated. This adds burden to real-time data integration systems that wish to employ an IVS method of low computational overhead. OVM, in contrast to DVM, avoids this burden. This is expected to result in reduction of the computational cost of data integration as it leads to collecting and processing less important data less frequently whilst it does not trade in another type of computational overhead. In order to support and invalidate this expectation, it is important to explore the major IVS methods of OVM nature.

4.3.1. Variable ranking

Variable ranking has proved to be a sound approach due to its simplicity and scalability in many variable selection algorithms (Guyon et al, 2003). In variable ranking, a scoring (or scaling) function is used to compute and assign a quantitative (Mirkin, 2005) score (or scale) for each input variable in relation with a target class e.g. correlation coefficient (Quevedo et al, 2007), and then variables are sorted based on their scoring criteria (Guyon et al, 2003).

Quevedo et al (2007) present a simple ranking algorithm that seems to be superior to more complex state-of-the-art ranking algorithms. The powerful computation time performance of their ranking method mainly comes from the explicit and sequential implementation of two modules (Quevedo et al, 2007) in a cycle of selection and elimination tasks; at first, a correlation-based criterion examines and ranks input variables with respect to output variables. Then an orthogonalisation module applies redundancy detection and variable elimination by normalising input variables according to the top ranked variable at the same cycle. This sequence is shown in Figure 4-7.

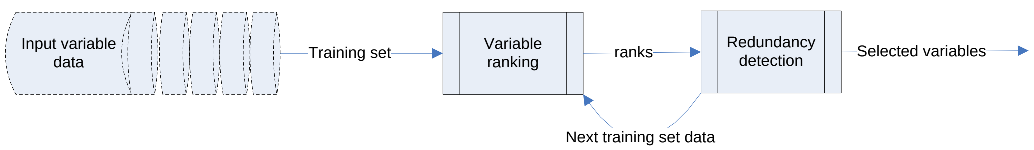


Figure 4-7 Cyclic ranking and selection of variables based on sequential data sets

The loop stops when either all input variables are processed, or when the number of training data values is smaller than the number of input variables.

Quevedo et al (2007) used Simplified Polynomial Expansion (SPE) as a sufficiently good approximation of general nonlinear models that map input variables to output variables. Therefore, an initial computational effort is required at the beginning of each cycle (embedded in variable ranking module) to accomplish SPE between output variables and remaining input variables. Quevedo et al (2007) experimented SPE-ranker algorithm on artificially generated data sets. The stoppage criterion, however safe, encourages long runs for systems with high number of input variables, since for example, for a system with 100 input variables the algorithm needs at least 100 execution cycles for processing 100 samples. With typical sampling rates of up to 5 samples per second, this takes at least 20 seconds before one full decision on the elimination (or change of acquisition settings) of input variables could be taken.

4.4. Variable subset performance assessment methods

Another root to the computational overhead of IVS methods could be explored in the overall approach that is taken to measure the performance of a selected set of variables. The effect of using one approach instead of the other could appear in the accuracy of the resulting selected variable set as well as the computational cost. As could be seen in Figure 4-8, approaches differ in the way variable selection module and selection validation modules interact, i.e. if they loop, are embedded, or are one-off. They also differ in the number of involvements of data sets from input or output variables, i.e. once, none, or multiple uses of data sets for training or test purposes.

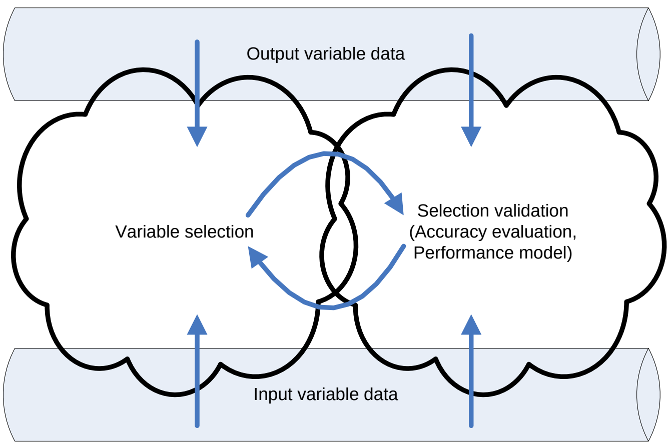


Figure 4-8 Different approaches could exist for interaction between variable selection task and selection validation task

Any of the aforementioned variable selection methods may take wrapper, embedded or filter approach for this. The compromise between these approaches will be discussed in the introduction given to the architecture of each approach at the following subsections.

4.4.1. Wrapper method

The wrapper approach (Kohavi et al, 1997) measures and compares the usefulness of different subsets of input variables with respect to decision making parameters. To achieve this, as shown in Figure 4-9, the approach selects the input variable and measures their influence on the performance model.

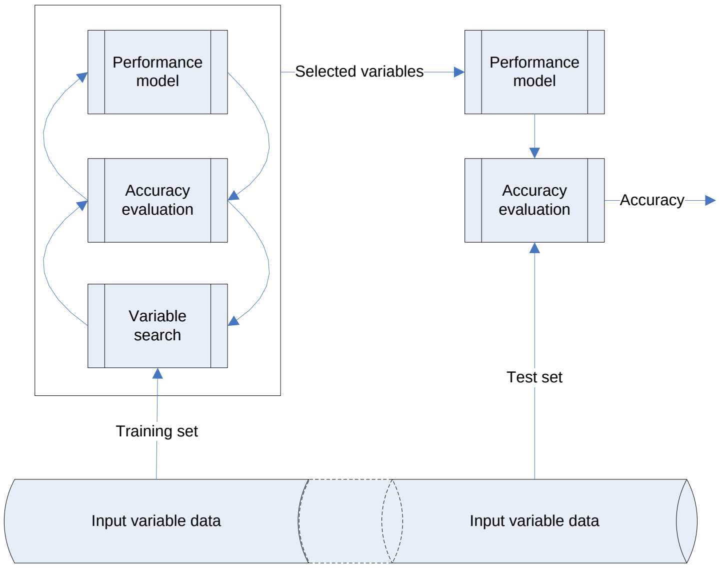


Figure 4-9 Conceptual representation of wrapper approach to variable selection

Therefore, similar to a learning machine, execution of iterative runs of the performance model using different variables (training set) at each run (Quevedo et al, 2007), and finally examination of selected variables with a different set of variable data (test set) leads to computational overhead (Talavera, 2005). Lemaire et al (2006) introduce a wrapper method for measuring importance of input variables built on predictive models, and probability distribution of input variables. In wrapper methodology, at occasions when computation is too exhaustive for large number of variables, heuristics such as backward elimination and forward selection (Norvig et al, 1995) may reduce but not necessarily eliminate the overall computational cost (Xing et al, 2003).

4.4.2. Embedded method

In embedded method, similar to wrapper method, an iterative learning mechanism is applied to determine the importance of the subsets of input variables. However, in contrast to wrapper method, embedded method does not use a closed learning mechanism which works independent of the actual variable selection functionality. Instead, like shown in Figure 4-10, embedded method incorporates the variable selection in the learning mechanism, which of course accelerate the finding of solutions as well as leading to more accurate outcome (Guyon et al, 2003).

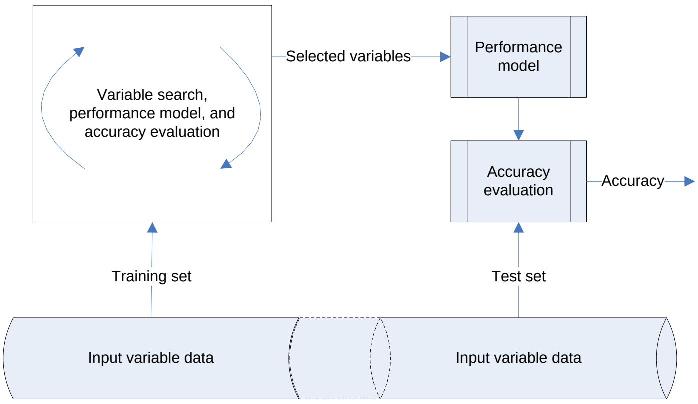


Figure 4-10 Conceptual representation of embedded approach to variable selection

Nevertheless, the embedded method, similar to the wrapper method, requires iterative executions of the performance model as well as historical variable datasets for training and test phases. From this point of view, the associated computational overhead is still of concern.

4.4.3. Filter method

Filtering of irrelevant and redundant variables, if accurate enough, could help with lowering the computational complexity of data acquisition (Quevedo et al, 2007) and search space (Ragg et al, 2004). In filter method, data and its properties are assessed for relevance using an independent criterion function. Therefore, details of algorithms and model which governs output variables (system performance data) make no difference on the variable selection. Therefore, filter method, as shown in Figure 4-11, in spite of learning-based wrapper method, seems like a one-step, and therefore low computationally complex process (Talavera, 2005). For the same reason of not implementing a learning process, however, filter methods do not necessarily find the most accurate and useful subset of variables (Guyon et al, 2003).

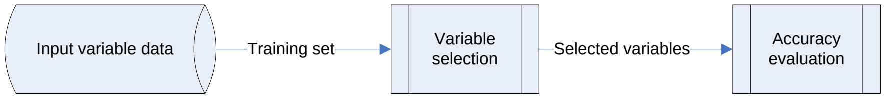


Figure 4-11 Conceptual representation of filter approach to variable selection

4.5. Perspectives of IVS methodologies

An overview to the applied methodologies for input variable selection could help with acquiring an appropriate angle to tackle IVS problem in this thesis. Answering the questions on the initial assumptions about the nature of the search space if it includes dependent variables as well as independent variables or if data sets are divided for different purposes, the relationship between input variables and model performance factors if it is deterministic or stochastic, generalities and specificities of the methods for fulfilling varieties of situations such as scale, time and accuracy of result, are among the points of interest required to be covered by an appropriate perspective.

The problem of IVS, both DVMs and OVMs, could be broken to two main sub-problems; first the method to derive the new subsets of variables, and second to decide how to assess the performance of the variable subsets. Solution to these two sub-problems was supplied from different perspectives which are discussed at the following. The last four perspectives were originally introduced by (Mirkin, 2005) for covering two main issues of clustering methodologies - being the nature of clusters and cluster validation approach.

4.5.1. Heuristic approach

Scale of the problem and time constraints on the IVS as well as the discrete nature of its search space suggest application of a heuristic approach (Askin et al, 1993). By heuristics it is meant to distinguish the search strategies which use problem-specific knowledge to find the solution (Norvig et al, 1995). The way heuristic can help with solving complex problems is through ignoring some aspects of the problem without performing computational processes. Therefore, in using heuristic approach for IVS some input variables to a problem that are known to be less important and more computationally expensive, could be ignored in the process of ranking or redundancy detection. Thus, the heuristic function which evaluates the cost of the input variable enquires extra information about the computational method applied to the input variable throughout the system.

Coding and maintaining heuristic rules are easier than optimisation procedures (Askin et al, 1993). However expert knowledge of the factors that determine rules of heuristics makes this approach limited to well-defined and known problems (Askin et al, 1993).

4.5.2. IVS through Optimisation and Simulation

It may be argued that input variable selection is similar to optimisation problem. The opening to this argument is the possibility to rephrase the objective of IVS – at least in this research - to “minimisation of computational overhead caused by sampling, processing, and storage of unnecessary values of inputs to a system while not losing the accuracy in the values of the key performance factors”. The amount of computational overhead is understood to be directly proportional (linear or non-linear) to the number of input variables (Quevedo et al, 2007). Therefore, the objective turns to be minimising the number of input variables, or more actually, minimising their sampling rates while keeping a limit to the inaccuracy in the values of the model’s performance variables.

The difficulty in taking this approach for IVS comes out of the stochastic nature of the values of objective i.e. model’s performance variables. The accuracy in these values could not be evaluated exactly due to its non-deterministic nature. Banks (2001) introduced some remedies to the issues of estimated values of stochastic parameters in a simulation based on the execution of multiple replications and / or longer runs of the simulation. Such solution does not comply with the time constraints of real-time analysis situations.

In other words, optimisation via simulation seeks building a prescriptive model (answering how to set the input variables) from an existing descriptive model (system simulation) (Askin et al, 1993). Prescriptive models can grow in size and become non-linear very quickly when details are incorporated, making them virtually impossible to solve to optimality (Askin et al, 1993).

4.5.3. Statistic perspective

By using statistic approach one extracts the properties of data as a sample of a probability distribution (Mirkin, 2005). For input variables, the approach replaces each input variable with the probabilistic distribution that can represent the data of the associated input variable with close enough accuracy (using an error estimation method). For selection of input variables, the allocated probability distributions may be clustered or otherwise shortlisted according to the result of the analysis against the degree of their influence on the performance of the model.

The suitability of statistic perspective on the input variables comes from two features of input variables; one is the expected uncertainty about the data series of the input variables due to the generality of the objective IVS approach in this thesis. Nothing is assumed to be known about the distribution of data sets. The second is the unlimited nature of the data series that enables statistic approach to look at them as a sample of a larger distribution (Mirkin, 2005).

The difficulty with using statistic approach for IVS, however, comes from the limitation on the number of distributions that could be applied to fit the sampled input variable data.

4.5.4. Machine learning perspective

Application of machine learning necessitates repetitive use of consecutive values of data in order to set some prediction parameters which could be used to categorise newly created data (Mirkin, 2005). Enough samples from each input variable should be tried against sufficient number of samples of performance parameters. This should occur variable-by-variable against parameter-by-parameter. Exhaustive number of executions is involved.

Supervised machine learning, use of performance parameters in the analysis, plays an important role in wrapper and embedded-based variable selection methods. The major disadvantages of learning mechanism are its requirement to accumulated data of input variables and repetitive performance model executions.

Unsupervised learning machine approach has the same on-by-one and repetitive execution issues as supervised. However, in terms of use of training data, it is more of curve fitting which was discussed in previous section or data mining which will be discussed in the next section.

4.5.5. Data mining perspective

Unlike statistic perspective, in data mining perspective concern is finding patterns and regularities (Mirkin, 2005) which are natural and unknown (Jain, 2010) within the available data. The approach does not focus on providing a consistent distribution - as in statistic - or function - as in machine learning. Instead, provision of a consistent pattern or cluster is targeted (Mirkin, 2005).

In general, by using data mining methods focus remains on the similarity of the input variables without considering their importance or influence on the model performance. Therefore, although group of similar input variables can be identified, extra effort would be needed to help with determination of their importance with respect to the output variables and from there to help with decision about their acquisition attributes.

4.5.6. Classification / knowledge-discovery perspective

Mirkin (2005) mentions that although the term classification is treated as a data mining concept - which, in contrast to the clustering concept, uses pre-specified categories to find and group data – classification perspective could differ to the one of data mining. The difference comes from the special settings that are expected to be explored and maintained between the structures of data (Mirkin, 2005).

In the context of IVS problem, input variables are considered inputs to a model which generate performance parameters. Therefore, the cause-effect relationship between the input variables and performance parameters could be considered as knowledge about the system to be discovered. Input variables with different levels of cause-effect relationships could be classified in different categories. From a discriminative viewpoint (Hand et al, 2001), some sort of functionality is required to maximise some measure of separation between the variables. Such discriminant function is explored through studying the level of impact that each independent variable has on the dependent variable.

The following section opens discussion to this issue and its available options under the term of sensitivity analysis.

4.6. Sensitivity analysis (SA)

Sensitivity analysis has been discussed by (Cloke et al, 2008; Durkee et al, 1998; Cukier et al, 1978; Borgonovo et al, 2007; Saltelli, 2002) as a technique to minimise the computational overhead by eliminating the input variables that have the least impact on the system. Sensitivity analysis techniques can help with focusing only on the most valuable information that has significant impact on behaviour of systems. Sensitivity indexing is the systematic way to express the impact of any input variable on the output parameters of a system (Krzykacz-Hausmann, 2001). From the same perspective, sensitivity analysis is a systematic approach for expressing the relationship between inputs and outputs of a system.

The measurement of the true impact of an input on the output of a system becomes challenging due to the epistemic uncertainty of the relationship between the two variables (Krzykacz-Hausmann, 2001). Selection of an appropriate method for sensitivity analysis therefore, depends on a set of factors and assumptions. These factors can be listed as follows:

4.6.1. The analytical relationship between the input and the output

Majority of sensitivity analysis methods tend to demonstrate the impact of change in one variable on the other by means of the mathematical equation that describes the relationship between them. Methods such as differential analysis (Isukapalli, 1999), Green's function (Yang, 2003), and coupled/decoupled direct (Faghihi et al, 2004) are classified among the analytical sensitivity analysis methods by Isukapalli (1999). However, the non-linear and non-monotonic relationship between input and outputs of a given system may not necessarily lend themselves to the use of such analytical methods (Krzykacz-Hausmann, 2001). The following paragraphs explain the reasons for this unfitness.

4.6.1.1. Differential analysis

In differential analysis the impact of an independent variable on the dependent variable is assessed by identification of the perturbation behaviour of the dependent variable due to the changes of the independent variable (Ambrosetti, 2007). This is achieved by finding the coefficients of the differential equation that explains the relationship between the independent and dependent variables (Ambrosetti, 2007). Methods like Neumann expansion (Lallemand et al, 1999) and perturbation method (Buonomo et al, 2010) could help with extracting these coefficients through approximating the differential equation. However, it is never guaranteed that the often complex and nonlinear relationship between system variables could be approximated with differential equations with sufficiently low error margins (Isukapalli, 1999).

4.6.1.2. Green's function method

When differentiating model equations is difficult due to their nonlinearity, use of Green's function could act like a catalyst to help achieving the sensitivity equations (Isukapalli, 1999). Effectively, in this method, differentiation operation is replaced by the sequence of finding impulse response of the model (Duffy et al, 2003), and the subsequent integration operations.

The concept of Green's function stems in the fact that one could formulate the total output of a linear time-invariant (James, 2004) system by a summation term that adds up all outputs of the system for all single points (Beylkin, 2008). In other words, each continuous function could be replaced by the infinite sum of delta functions with nearly zero distances, as shown in Figure 4-12. The output of the system for a single point is described by the Green's function.

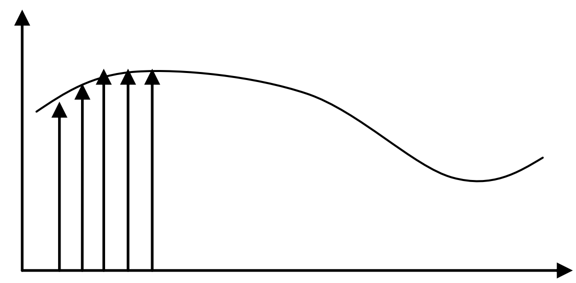


Figure 4-12 Representation of a continuous function by infinite sum of delta functions

The mathematical view of Figure 4-12 gives; (Duffy et al, 2003)

$$f(x) = \int \delta(x - z) f(z) dz \qquad \textbf{(4-1)}$$

Now the Green's function $G(x)$, is defined in $\mathscr{L}[G(x)] = \delta(x)$ with the linear differential system $\mathscr{L}$ that governs the model equation i.e. $\mathscr{L}[v(x)] = f(x)$. If $G(x)$ could be obtained, then it could be used in; (Beylkin, 2008)

$$v(x) = \int G(x, z) f(z) dz \qquad \textbf{(4-2)}$$

to solve the differentiation problem of $\mathscr{L}[v(x)] = f(x)$ with nonlinear $f(x)$.

Having introduced the auxiliary function, it is important to note that a linear and time-invariant system could only benefit from the situation. Another disadvantage of the application of the Green's function method is its ability to work only with the ordinary type of differential equations that govern dependent variables with respect to independent variables. In real applications it is difficult to separate relationships of independent variables with dependent variable. Additionally, working one variable at a time for high dimensional systems could be computationally expensive.

4.6.1.3. Coupled/decoupled direct

In coupled direct method after differentiation of model equations, the subsequent sensitivity equations are solved together with the original model equations (Isukapalli, 1999). In decoupled direct method they are solved separately (Isukapalli, 1999). This gives the impression that decoupled direct method is advantageous in terms of computational cost. Although decoupled direct method is reported more efficient than Green's function method (Isukapalli, 1999), similar to the other analytical methods, knowledge of model equations is required. This tags the two features model-oriented and expertise-hungry to the analytical methods, making them disadvantageous over the sensitivity analysis methods that do not require model equations.

4.6.1.4. Sampling-based methods

Where there is no mathematical equation defined between the model variables, or when it is not preferred to work on the existing model equations, some other sensitivity analysis methods which do not care much about the computational overhead tend to establish one by identification of some statistical features in the distribution of data series of the two variables. The general shortcoming of these methods such as Fourier Amplitude Sensitivity

Test (FAST) (Cukier et al, 1978)(McRae et al, 1982), Morris (Jin et al, 2007; Braddock et al, 2006), Monte-Carlo (Sobol, 2001) and Latin Hypercube (De Pauw et al, 2008) is in their heavy reliance on historical data. This dismisses them from being good candidates for time-constrained applications.

As one example, Cloke et al (2008) applied their model to 1280 sample values of 20 input parameters. Each cycle of sample generation and model execution took between 130 minutes to 52 hours per set of samples. The overall execution cycles took almost 46 days. This example reflects the significant impact of sampling based analysis on the computational overhead. The following subsections explain major sampling based methods.

4.6.1.4.1. Monte Carlo and Latin Hypercube methods

Random sample generation, as the main characteristic of Monte Carlo method, provides the required values of independent variables from which dependent variables are produced based on the execution of the model on sampled input data (Robert, 2004). The random sampling scheme occurs either in no particular manner, or with some criteria that could help with the efficiency of computation (Isukapalli, 1999). For example, in Latin Hypercube Sampling (LHS) method (De Pauw et al, 2008), the range of each input parameter is divided into intervals of equal probability. In each set of samples of input parameters, each input parameter takes a random value from one of its intervals with no repeat of the same interval for one full sampling cycle (De Pauw et al, 2008). This way, a better chance to all segments of data in the distribution and therefore more informative distribution parameters for generated output could be achieved in shorter period (Isukapalli, 1999).

Nevertheless, in a general overview of Monte Carlo method, as could be seen in Figure 4-13, first from the stream of available data, the probability distribution of input variables is estimated, i.e. the curve fitting blocks. Then based on these distributions, random sample generation occurs, i.e. sampler blocks. After the model is applied to the generated samples, the produced output values are processed for estimation and extraction of their distribution attributes (Shonkwiler, 2009).

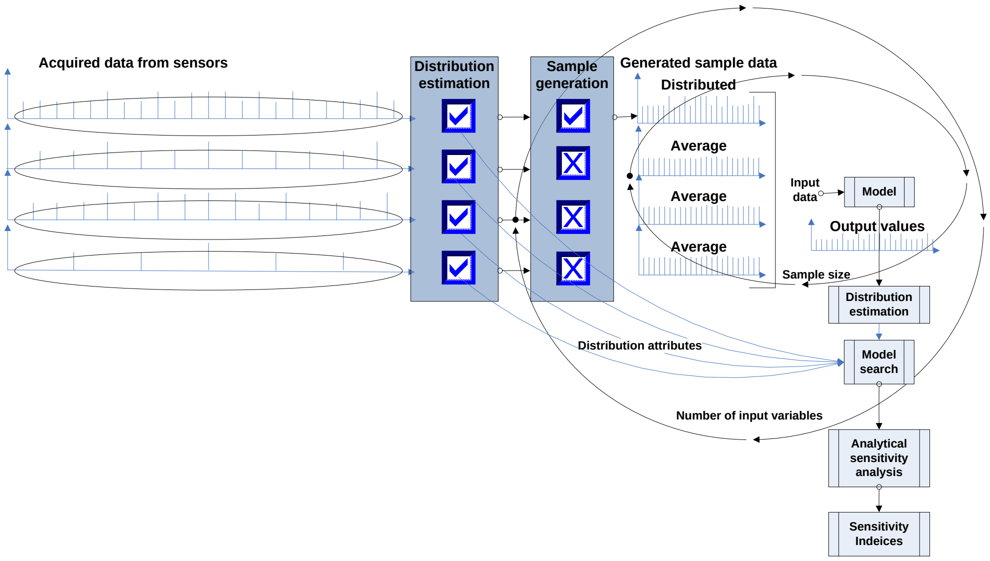


Figure 4-13 General view of Monte Carlo method for sensitivity analysis

One rather big issue with Monte Carlo methods for real-time applications is the effort required to estimate the distribution of the input variables prior to sample generation.

For sensitivity analysis purposes based on Monte Carlo sampling method, in order to infer the impact of each input variable on the output variable, data samples of only one input variable (the checked box in Figure 4-13) is generated at a time while the other input variables are set at a fixed – for example average - value (the cross-marked boxes in Figure 4-13). This cycle repeats per each input variable. Krzykacz-Hausmann (2006) called this feature '*double-loop nested sampling procedure*' which could be computationally very expensive, particularly with high dimension of input variables.

4.6.1.4.2. Morris method

In Morris method, as a parameter screening method (De Pauw et al, 2008), changes in the value of output variable is measured per changes in each input variable. Changes of only one input variable (θ) is applied to the equation $EE_i(\theta) = \frac{y(\theta+\Delta)-y(\theta)}{\Delta}$ to calculate values of elementary effect (EE_i), with input change step dictated by Δ (Jin et al, 2007). The produced set of EE_i is then processed for distribution estimation. This implies that each cycle of output distribution estimation takes $M = 2rn$ model executions if r is the number of required output values for estimation of a stable distribution and n the number input variables (De Pauw et al, 2008). Even though more economical extensions of Morris method could reduce the total number of cycles - for example by using each generated model output in more than one calculation (Braddock et al, 2006), a typically low value for M is as high as 21000 executions (1000 output values and 20 inputs applied to $M = r(n+1)$ in an improved Morris method (De Pauw et al, 2008)). Morris method thus could not satisfy sensitivity analysis in time-constrained applications.

4.6.1.4.3. Analysis of variance (ANOVA) methods

To overcome the issue of One-At-a-Time (OAT) processing of influences of input variables on output, which does not support detection of influences of interactions between multiple input variables (i.e. second order and higher) on the output, a series of sensitivity analysis methods measure and decompose variance of output distribution to the elements that could separately represent these interactions (Saltelli, 2002). ANOVA based sensitivity analysis methods are in general computationally more efficient for this reason (Ravalico et al, 2005).

The fundamental idea of variance decomposition (Sobol, 2001) could be shown by equation;

$$V(y) = \sum_i V_i + \sum_i \sum_{j>i} V_{ij} + \cdots + V_{12\ldots n} \qquad \textbf{(4-3)}$$

in which, the left hand side shows the total variance of model output y and the right hand side is a sequence of summation terms of first order influences of input variables, second order influences, and so on with finally $V_{12\ldots k}$ as the portion of variance of output for interaction of all n input variables together (Saltelli, 2002). Based on this variance decomposition, sensitivity index of output with respect to each input variable is defined as (Saltelli, 2002);

$$S_i = {}^{V_i}/_{V(y)} \qquad \textbf{(4-4)}$$

To achieve the decomposition elements of equation 4-3 and from there sensitivity indices, when no explicit relationship exists between inputs and output (i.e. when analytical approach is not possible), numerical approach that is mainly based on sample generation (Monte Carlo) could be adopted (Borgonovo et al, 2007). The amount of computational overhead, in terms of the number of model runs (for producing output values per each input sample set) could be worked out from equation;

$$M = N \times \sum_{i=0}^{n} \frac{n!}{(n-1)!i!} \quad \textbf{(4-5)}$$

with N as the sample size, and n as the number of input variables (Saltelli, 2002). For example, with 10 input variables and 1000 samples, the number of model executions would make 1024000 which is significantly high. Thus, the basic idea is not attractive to real-time applications unless improved. The following subsections cover this improvement and the overall drawback of all sampling based methods.

4.6.1.4.4. Fourier amplitude sensitivity test (FAST)

Fourier Amplitude Sensitivity Test (FAST) (Cukier et al, 1978) and its extended version (Xu et al, 2008) are examples of improvements in computational efficiency of ANOVA-based sensitivity analysis methods. FAST (and extended FAST) is distinguished between other ANOVA methods by its input data sample generation scheme, in which, samples for each input variable is generated according to a periodic function within the limits of the input variable (De Pauw et al, 2008). In other words, in FAST method the data distribution of input variables cannot be estimated from the acquired historical data. Instead, all distributions of input variables are considered to be uniform and within a range which should be specified. The subsequent generated samples on this range follow a periodical function as (Saltelli, 1999);

$$x_i(s) = \frac{1}{2} + \frac{1}{\pi} \sin^{-1}[\sin(w_i s)] \quad \textbf{(4-6)}$$

where s is a scalar value changing between 0 and 2π, and w_i an integer frequency assigned uniquely to each input variable.

The periodic nature of sample generation scheme (i.e. change of s) causes the model output values (for each i) to be periodic in terms of s. Therefore, by using numerical Fourier analysis on the values of output, magnitude of Fourier spectrum at each frequency w_i represents the sensitivity index of the corresponding input variable. Components of this process are shown in Figure 4-14.

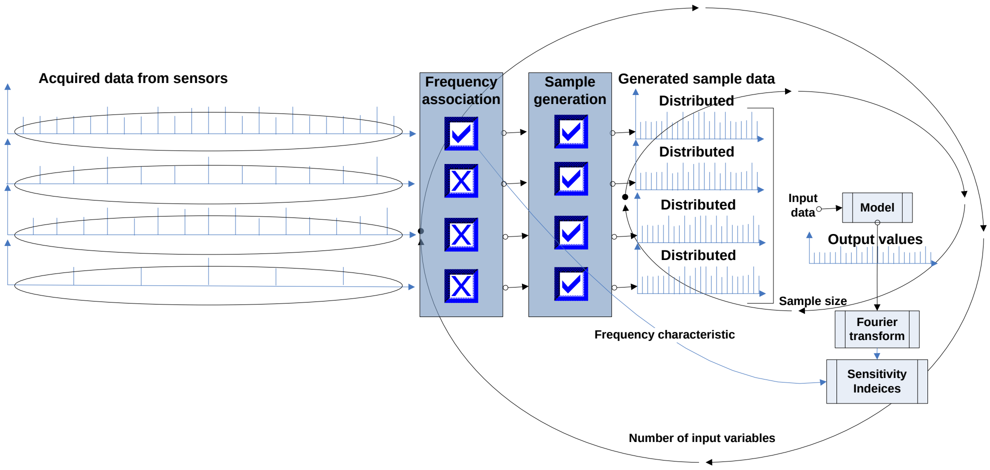


Figure 4-14 General view of FAST method for sensitivity analysis

As shown in Figure 4-14, some aspects of computational cost that existed in the Monte Carlo methods – i.e. distribution estimation - is omitted in FAST method and replaced with simple tasks of boundary detection and frequency association. Furthermore, the output value distribution estimation is also replaced by numerical Fourier Transform (FT) method for finding Fourier spectrum. In order to explicitly identify the power coefficient associated to the frequency of each input variable, proper choice of the unique frequencies (w_i) is required. For this, the range of frequencies w_i is divided into high and low ranges. A high frequency is assigned to the input variable subject to power spectrum coefficient identification and the rest of input variables are assigned a frequency from the low range. This way the distance between the high frequency and all other low frequencies on the spectrum allows clear identification of the coefficient, or sensitivity index. In Figure 4-14 checked box in frequency association module shows that the input variable number 1 is assigned a very different (high) frequency for sample generation comparing to the frequency of the others (with crossed boxes). As a result the power coefficient of the first frequency would be detectable with sufficient confidence (see Figure 4-15).

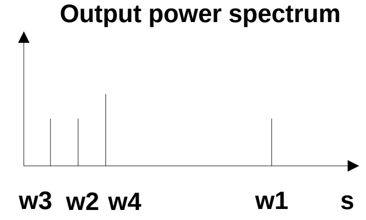


Figure 4-15 Power spectrum of the model output

This type of frequency association adds a new loop for sample generation and model execution process to the analysis. Using FAST, the number of model runs could be obtained from equation;

$$M = Nn(8w_{max} + 1) \qquad \textbf{(4-7)}$$

where N is the sample size, n the number of input variables and w_{max} the largest among the assigned frequencies w_i (Saltelli, 1999).

The number of model executions in FAST does not seem low comparing to the alternative sampling-based SA methods, as it still features the '*double-loop nested sampling procedure*' according to Krzykacz-Hausmann (2006). However, the computational overhead could be lower due to the simpler tasks included in the nested loops – sample generation and Fourier transform in FAST is usually less computationally costly than the collection of sample generation, distribution estimation, and distribution-based function fitting (model search).

4.6.1.4.5. Time uncertainty between generations of sample data

A major drawback of all sampling-based SA methods is in the concurrency in use of generated sample data by the model which is not always the case in real systems. The simultaneous entities of data and their connection to the model for producing the next corresponding output value is a rare scenario in systems with stochastic nature of events and non-deterministic behaviour of responses.

For example, Li et al (2006) in the application of stochastic simulation for statistical analysis and characterisation of Resin Transfer Molding (RTM) processes. For calculating the mold pressure profile and resin advancement progress, they consider measurement of time as sources of uncertainty among the other measurements such as viscosity, pressure, displacement, surface density, compressing variation, stacking sequence, race tracking, and human error.

As an industrial example, it is difficult to ascertain when an item on a conveyor belt passes a certain measurement point, and therefore, the exact time that scanned RFID tag of the item or the measured temperature of the surface of the item enters the system is not known. In sample generation for measured data series, only magnitude of input variables is concerned to follow a fit distribution. The time of generating each new data obeys a fixed frequency. When no value of a data series has actually entered the system, in sample generation, data entity of zero or low value is generated based on the fitted probability distribution. This argument is shown in Figure 4-16.

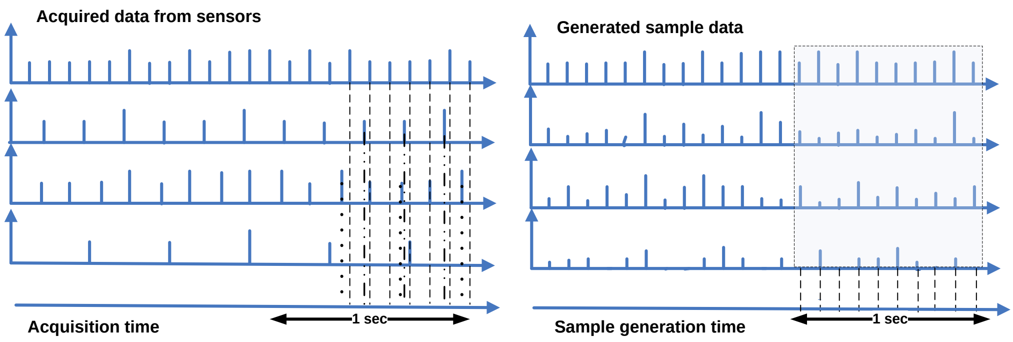


Figure 4-16 Sample generation may generate more data entities than the original distribution

In the industrial example shown in Figure 4-16, four data time series from four sensors are acquired with sampling rates of 10, 4, 6, and 2 samples per second. In one second, therefore, 22 data entities enter the system. However, sample generation with generation rate of 10 samples per second, generates 40 data entities in one second. Hence, it seems very likely that a portion of computational overhead of sampling-based sensitivity analysis methods comes from generation of extra data. To avoid this one may consider realisation of randomness or frequency of the data entering the system, and simulating that by applying the realised frequency or random function to the rate of sample generation.

Although researchers who are interested in obtaining more realistic mathematical models, tackle the problem of stochastic modelling by adding some randomness to the model equations (Øksendal, 1998), a similar practice has not been reported in sampling-based sensitivity analysis methods. Even though if applied, it adds up to computational efforts of distribution estimation and sample generation tasks, and the trade-off between this added computational overhead and the reduction of generated sample data must be investigated.

One sensitivity analysis method despite being conceptually based on relationships between dependent and independent variables similar to ANOVA-style decomposition methods, used an estimation method for sensitivity measurements that took original samples into account. This method which helped with reduction of computational overhead was proposed by Krzykacz-Hausmann (2001) and is introduced at the following.

4.6.1.5. Entropy-based epistemic sensitivity analysis

Krzykacz-Hausmann (2001; 2006) tackled the issue of computational cost of '*double loop sample generation strategy*' and restrictive conditions of evaluation of dependent variables based on independent variables in sampling-based SA methods by proposing an approximation approach that measures the entropy of variable distributions from original samples. The method used the same decomposition equation as in equation 4-3 in section 4.6.1.4.3 with the only difference of working on entropy instead of variance of sample data distributions.

According to Krzykacz-Hausmann (2001) and the following equations, for a dependent variable Y values of entropy $H(Y)$, conditional entropy $H(Y|x)$ i.e. when the value of an independent input variable is known, and expected conditional entropy $H(Y|X)$ i.e. mean value of all values of conditional entropy when the value of an independent input variable is known, help with determination of $H(Y) - H(Y|X)$ as the difference between the entropy and expected conditional entropy which indicates the sensitivity index $SI(Y,X)$ of dependent variable Y with respect to independent variable X.

$$H(Y) = -\int f(y) ln f(y) dy \quad \textbf{(4-8)}$$

$$H(Y|x) = -\int f(y|x) ln f(y|x) dy \quad \textbf{(4-9)}$$

$$H(Y|X) = -\int H(Y|x) f(x) dx = -\int\int f(y|x) ln f(y|x) dy dx \quad \textbf{(4-10)}$$

$$H(Y) - H(Y|X) = \int\int f(x,y) ln \frac{f(x,y)}{f(x)f(y)} dxdy \quad \textbf{(4-11)}$$

$$SI(Y,X) = \frac{H(Y)-H(Y|X)}{H(Y)} \quad \textbf{(4-12)}$$

(f is probability density function of Y)

The method replaces time consuming sample generation of X and evaluation of Y by '*Simple-Random Sampling (SRS)*' using the following piecewise uniform density function estimations;

$$f^*(x) = \sum {n_{i.}}/{n_{..}} (a_i - a_{i-1})^{-1} I[a_{i-1}, a_i)(x) \quad \textbf{(4-13)}$$

$$f^*(y) = \sum {n_{.j}}/{n_{..}} \left(b_j - b_{j-1}\right)^{-1} I\left[b_{j-1}, b_j\right)(y) \quad \textbf{(4-14)}$$

$$f^*(x,y) = \sum\sum {n_{ij}}/{n_{..}} (a_i - a_{i-1})^{-1}\left(b_j - b_{j-1}\right)^{-1} I[a_{i-1}, a_i)(x) I\left[b_{j-1}, b_j\right)(y) \quad \textbf{(4-15)}$$

where $a_0 < a_1 < \cdots < a_k$ and $b_0 < b_1 < \cdots < b_m$ are partitions of the ranges of X and Y respectively into k and m disjoint subintervals,

n_{ij} is the number of sample points (x_k, y_k) falling into the rectangle $[a_{i-1}, a_i) \times \left[b_{j-1}, b_j\right)$,

$n_{i.}$ is the number of x_k values falling into the interval $[a_{i-1}, a_i)$,

$n_{.j}$ is the number of y_k values falling into the interval $\left[b_{j-1}, b_j\right)$,

$n_{..}$ is the total number of sample points,

$I[a_{i-1}, a_i)$, $I\left[b_{j-1}, b_j\right)$ are the indicator functions of the intervals $[a_{i-1}, a_i)$, $\left[b_{j-1}, b_j\right)$.

Replacement of above piecewise uniform density functions into equation 4-11 gives the estimated sensitivity index as;

$$H^*(Y) - H^*(Y|X) = \int\int f^*(x,y) ln \frac{f^*(x,y)}{f^*(x)f^*(y)} dxdy = \sum\sum {n_{ij}}/{n_{..}} \ln\left(\frac{n_{ij}n_{..}}{n_{i.}n_{.j}}\right) \quad \textbf{(4-16)}$$

The approach is shown in Figure 4-17.

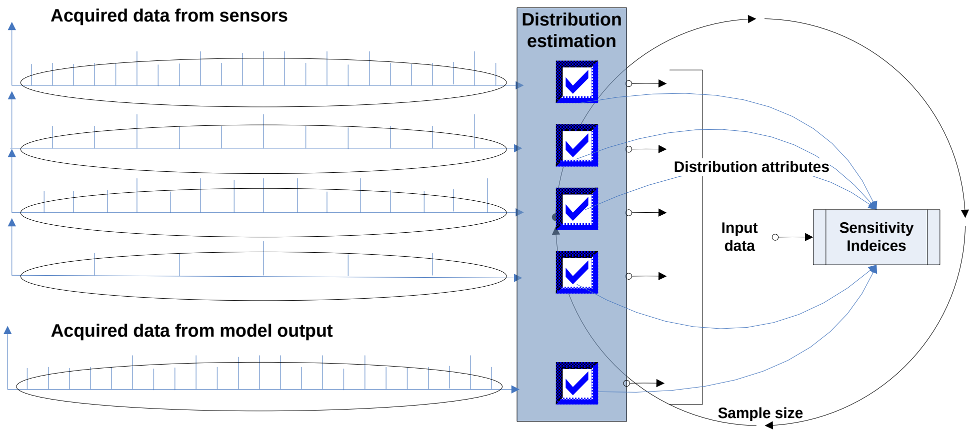


Figure 4-17 General view of Entropy-based method for sensitivity analysis

As it could be seen in Figure 4-17, only one sample size execution is sufficient for obtaining samples to accomplish approximation of sensitivity indices. Krzykacz-Hausmann (2006) demonstrated feasibility of the estimation approach in a test case with fifteen independent and two dependent variables. Reasonable results were provided with far less computational cost. However, obtaining the appropriate indicator functions for each independent variable requires state of knowledge of their distribution probabilities (Krzykacz-Hausmann, 2006).

4.6.2. The Statistical distribution of input variables

The second factor involved in the selection of sensitivity analysis method is characteristics of data distributions of input and output variables. The sensitivity indices are normally influenced by the distribution of the corresponding data series. For example, nonlinear relationships between input and output series of a model cannot be recognized by correlation-based sensitivity analysis methods (Annis, 2008). Variance-based and Entropy-based indices are expected to be more sensitive to heteroscedastic data (Krzykacz-Hausmann, 2001), whilst homoscedasticity of data series can be higher among discrete signals and much higher between binary signals.

In the case of this thesis, the assumption of generality of type and characteristics of data discourages use of those SA methods with interest on specific types. For example, correlation-base SA cannot be considered as a suitable approach. However, variance-based or entropy-based SA methods show to be feasible from the point of view of data heterogeneity.

4.6.3. The Computational overhead of SA methods

Sensitivity analysis is a computation hungry process. In domain-wide (global) sensitivity analysis methods, large batches of input variables are captured at each time interval and levels of sensitivity is measured based on historical data analysis. For example, sampling-based methods need to generate new and rather large sizes of sample values of both output and input data regardless of the original sample sizes.

The amount of resources needed by the SA algorithm and its associated data can be compared with the amount of savings that may occur as a result of the applied algorithm. Correlation-based methods (Annis, 2008) need equal sizes of data batches for input and output series of the model. Therefore, sampled data series need either interpolation or extrapolation to maintain equal sizes; subsequently adding extra computational load onto the system. ANOVA based SA methods save computational efforts on the analytical analysis side, but instead contribute to computational cost by sample generation efforts.

It has been reviewed in earlier sections (under 4.6.1) that FAST performs better than the other ANOVA-based SA methods due to its independence from detection of input variable distribution characteristics. Ravalico et al (2005) introduces FAST to perform most efficiently between all SA methods. However, having investigated the ability of several global SA methods - including Morris, correlation, and FAST- they also stress that none of the existing SA methods maintain competency with increasing number of input variables.

It is obvious that there is wide scope for exploring methodologies for sensitivity analysis that could perform as efficient as they could serve time-constrained scenarios like real-time applications.

4.7. Chapter conclusion

Input variable selection, as a solution to the problem of dimensionality reduction, partially shares methodologies with feature selection. Partially because features are created for a purpose more particular than input variables. Solution to the problem of Input Variable Selection (IVS) is by itself computationally intensive. The amount of computational complexity that may be added as a result of variable selection must also be measured and used for evaluation and selection of the IVS method.

From the reviewed literature, it could be concluded that based on their effect on the structure of the original set of input variables, IVS problems can be classified into two distinct groups of methods; primary - or original - variable methods (OVM) and secondary – or derived - variable methods (DVM). The major distinction between OVM and DVM lies in the issue of keeping the variables intact and only deciding on their redundancy, or in contrast, transforming to new subsets of variables. A general disadvantage of DVM regards to the data storage and processing of the new variables. These issues contribute to the computational overhead of DVM-based IVS methods which makes them less attractive.

It seems feasible to find an IVS method which does not generate extra computational effort to construct new variables. It is an important feature to realise that in this thesis; firstly a real-time low-cost computation data integration, and second, an assumption of setting up predetermined connections between input variables to the system is pursued. Such IVS method could make use of classification and separation approach by taking into account some concepts of relationships, for example cause-effect, between input variables and model's performance parameters.

The method for measurement of separation criteria of input variables based on their influence on system outputs, so called sensitivity analysis, could be selected based on preferences and priorities on the relationship between input and output variables, data distribution of variables, and more importantly, computational cost of the method. Based on those attributes, yet there is no sensitivity analysis method that competently works with complex system in terms of heterogeneity and large number of input variables as well as time constraint.

The opportunity to bring new light to the problem of input variable selection will be explored in chapter 7 and measurements of computational efficiency will be introduced in chapter 8. Implementations and their results will be discussed in further chapters.

5. Design and development of the Flexible Data Input Layer Architecture (FDILA)

Based on the issues learnt from the reviewed literature in chapters 2-4, in this chapter the foundations of architecture of this thesis is proposed to address the complexities of data integration in design, implementation, and execution aspects. The computational overhead of pre-processed and post-processed data depends on the architecture of integrated data extracted from sensor network (Iyengar et al, 1994). Computational complexity and overhead increases exponentially when the number and variety of sensors increase.

While in a modern manufacturing system the main processes are locally controlled, still many data points are networked to have their data acquired and gathered for a further central process (Cecelja, 2002). A typical shop floor data collection system may be designed to gather data about for example process yield, machine performance, operation times, order status, inventory, product traceability, product/process quality, and personnel (Cecelja, 2002).

According to Isukapalli (1999), the potential problems and their causes of systematic input data collection in shop floors include erroneous problem definition, lack of clear objectives, system complexity, poor data access, difficulty in identifying available data sources, and limited data handling capability. Isukapalli (1999) developed a reference data model that links parts and resources in a production flow. However, his proposed system falls short of explaining the methods that relate sensory input data to their proposed input data model.

5.1. Data life cycle within data integration systems

Prior to introducing the main components of the proposed data integration system, the sequence of transformations of data throughout integration and processing is discussed by an imaginary example. Figure 5-1 shows a schematic of the main areas of data transformation in an imaginary manufacturing line.

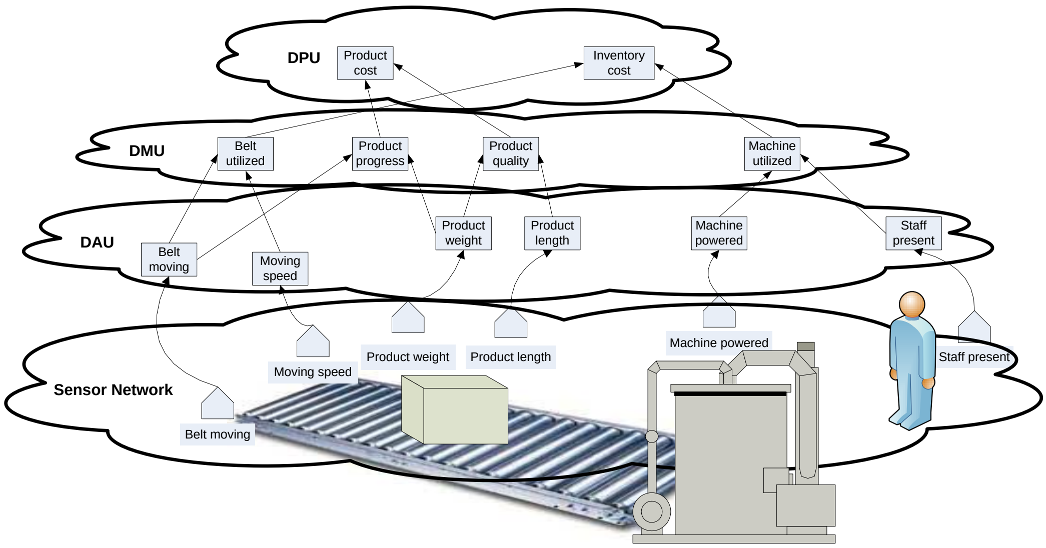


Figure 5-1 Schematic of the main areas of data transformation in an imaginary manufacturing line

5.1.1. Data acquisition unit

A series of hardware and software form Data Acquisition Unit (DAU) that is in charge of collection of raw data from sensor output (curved arrows). This is achieved either directly by connection between the analogue signal of sensor output to the data acquisition hardware or through controller devices and their associated data networks e.g. OPC and Ethernet. Acquisition of signals occurs frequently with specified sampling frequency. For each sampled signal one parameter is defined in the memory space to hold the latest sampled value. Sampled values are prepared to be used by modules that define more meaningful information from raw data. Preparation includes computational signal conditioning to translate the signal emanated from field data acquisition devices. For example, values of weak signals may be scaled up to fit a specific range, or an identification tag may be cleared off extra prefixed characters.

5.1.2. Data management unit

A series of software computational methods form Data Management Unit (DMU) which transforms of raw data into pre-defined parameters, or variables. This task which could be called sensor data aggregation is achieved by frequent access to the memory spaces where sampled raw data is stored; implementing functions, and storing the result of functions to new memory spaces, called input variables. Figure 5-2 below illustrates an example of aggregating sensor data into variables.

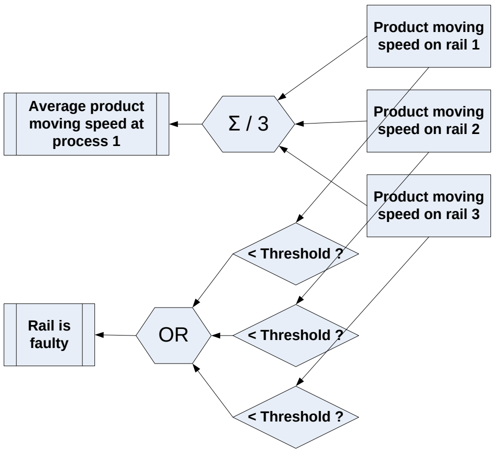


Figure 5-2 Aggregation of sensor data into variables

5.1.3. Data Processing Unit (DPU)

Values of input variables are used by another series of computational methods that work out the performance factors of the system being monitored. The number of Key Performance Indicators KPIs is usually a few and they represent parameters usually useful for decision making about reconfiguration of processes and layouts in favour of improvement of KPIs. Usually, a fewer (perhaps one or two) number of KPIs are set to report the overall cost of the processes being monitored. Reduction of this final cost parameter is set as target for decision makers.

A generic paradigm for computation and evaluation of KPIs would be difficult to implement using analytical modelling. To be able to define a performance modelling tool that could cope with complexities of process definition and stochastic events, Discrete-Event Simulation (DES) and modelling has been reported to be a better candidate (Harmonosky, 1995; Son et al, 2001; Komashie et al, 2005, Tavakoli et al, 2008, Komashie et al, 2009). This technique is used for performance modelling and performance evaluation.

5.2. Layered architecture

The hierarchical nature of the relationships between DAU, DMU, and DPU in terms of data types and data transformations, together with the generic nature of the problem in hand are addressed by taking layered approach in design of the data integration system. It has been reviewed in Chapter 2 that the layered architecture, as a well known approach to gathering and preparing data for different types of volatile systems allows definition of different groups for the elements of the information system in which, elements of each group are capable of providing different functionalities and services, and could communicate with the elements of the other groups through a defined communication behaviour (Bauer et al, 1994).

To start with the layered approach, first the overall functionality and data flow diagram of data integration system is divided into two main layers; one layer for data and related functions before performance modelling, and the other layer for performance modelling and functions afterwards. Figure 5-3 shows the overall layered view given to the elements of the data integration system. The Data Integration Layer (DIL) and Data Processing Layer (DPL) are assumed to be capable of covering all activities with the data throughout the information system.

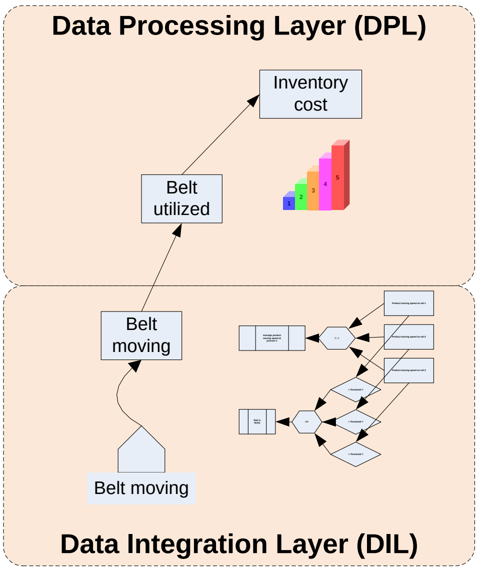


Figure 5-3 Overall layered view given to the elements of the data integration system

In the proposed view, the DPL was used to apply the modelled processes of the real system on the input variables prepared by the DIL. The DIL is therefore responsible for collecting data and preparing variables. The DPL consists of data processing and presentation. In addition, DPL is used to generate deterministic and potentially stochastic information. These major responsibilities of the two layers were thought to fit within separate sub-layers.

According to Bauer et al (1994), International Standardization Organization (ISO) expresses that the selection of a best layering solution may not be easy. Bauer et al (1994) in their book introduced four principles which could be considered in order to achieve good layered reference architecture; the number of necessary layers, the entities within each layer, the services provided by each layer, and the protocol used to communicate between entities. These principles are focused on separate layers to be influenced by manifestly different, easily localised, and minimised interactive functions (Bauer et al, 1994). Considering these principles helps with easy integration and redesign of layers as well as cost effective interactions between them (Bauer et al, 1994). Design of the layered architecture is described according to Bauer et al (1994) as follows;

5.2.1. The number of necessary layers

To ascertain the number of layers in DIL and DPL one needs to take into consideration the nature of the data flow and operation mode. Figure 5-4 shows the main layers of DIL and DPL. In DIL, collection of raw data occurs in Data Definition Layer (DDL), and preparation of the information - or variable – in Variable Definition layer (VDL). In DPL simulation of the process based on the real-time information occurred in the first layer, Performance Definition Layer (PDL), where prepared information arrived from the VDL. The second layer in the DPL, Scenario Definition Layer (SDL), dealt with the analysis of information for fast-forward simulation. Data flow was considered upwards from the lower layer in the diagram towards the upper layer. The contents and services of each layer are discussed at the following sections.

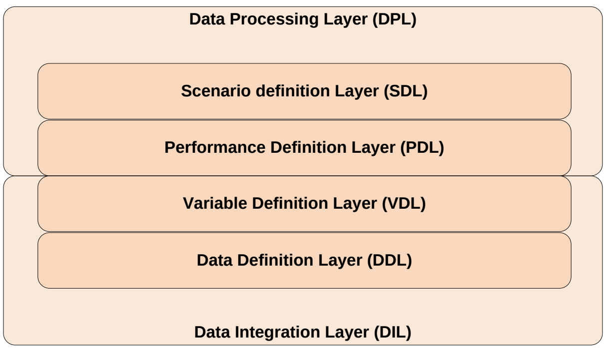


Figure 5-4 Four layers form the layered structture of the data integration system

Details of the four layers DDL, VDL, PDL, and SDL are expanded and described at the following sections under the remaining three principles of layered architecture design.

5.2.2. The assembly of entities within each layer

Each layer consisted of data entities as well as functional units. Depending on the layer, the way to deal with the available data is different. This influences the structure and attributes of data at each layer. Functionalities at each layer include definition of data entities and communication with the adjacent layers according to the defined structure and technology. The following sub-sections describe each entity of each layer in more details.

5.2.2.1. The entities of data definition layer

The main resident of this layer was the raw data which was collected from the sensory network. For data to be collected connections to the real system were essential. Therefore, connection units made up the data acquisition system in this layer. Flow of data between the sensory network and the DDL occurs under the specified data structure which includes type of data and acquisition attributes of each data connection. Of course, a buffering entity holds the collected values to be taken by the DDL. Flexibility in the definition and number of data connections necessitated facilities for their creation and modification by the user of the information system. Above entities are summarised in Figure 5-5 below.

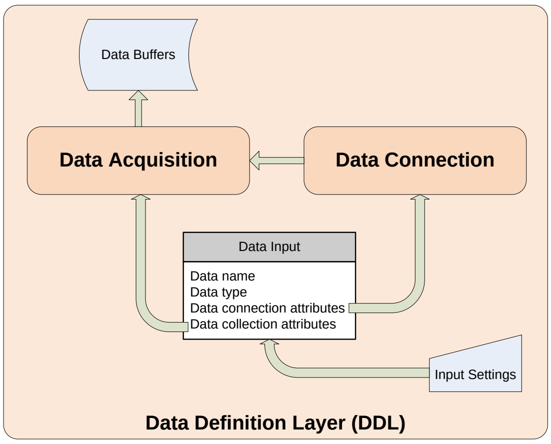


Figure 5-5 Entities of data definition layer

5.2.2.2. The entities of variable definition layer

Similar to the DDL, the main entity of this layer was considered to be the information which was to be prepared by combining some data inputs. This information, called as variable, require some attributes to help with computing their value. These attributes, which included the list of data inputs and the formula between them, together with the evaluation unit were the other entities of this layer. Preparation of information also included some filtering functionality which could select the variables with more importance for the next layer and influence the collection attributes of the variable. Similar to the flexibility requirements in DDL, facilities to the definition of the variables based on the data inputs was considered in this layer. Lastly but not less important, a data buffer was thought to hold the prepared values available to the performance definition layer. Above entities are summarised in Figure 5-6 below.

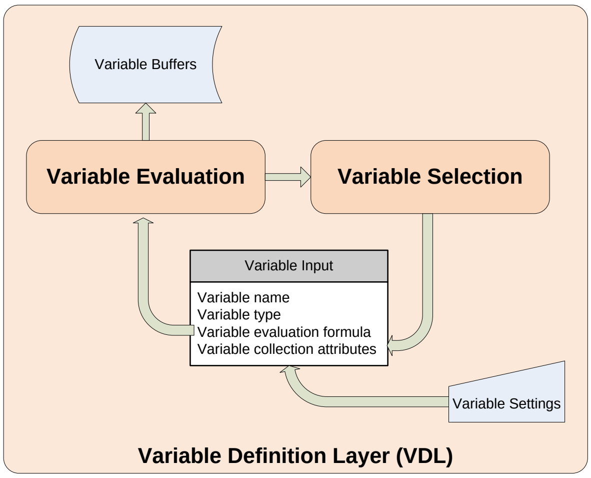


Figure 5-6 Entities of variable definition layer

5.2.2.3. The entities in performance definition layer

The performance definition layer, or PDL, held those entities of the data processing module facilitates the evaluation and presentation of system key performance indicators. The Performance Model module consisted of the models of the system processes and their relationships as well as models of system data structures. Another entity in the PDL interfaced with the VDL for reception of information (value of variables) and relating them to the Performance Model. This mechanism takes place in real-time. Additionally, the

Performance Model had facilities to specify the key performance indicators of the system. Figure 5-7 shows the entities of the PDL.

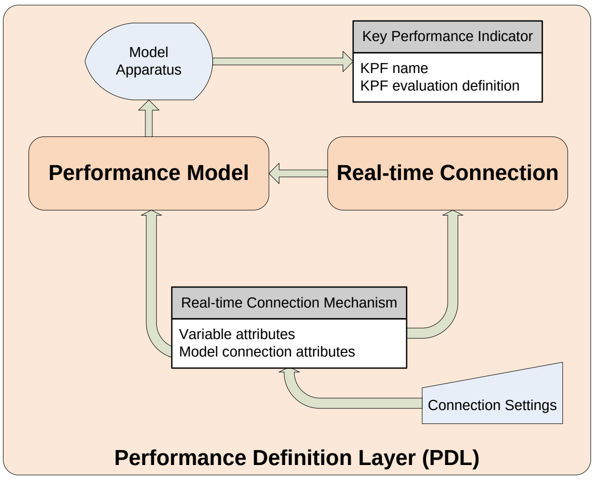


Figure 5-7 Entities in performance definition layer

5.2.2.3.1. Performance model – a DES approach

Although modelling and simulation could require a large effort for the steps involved including data gathering, input analysis, model building, verification and validation (O'Hara et al, 2010), and it is used to analyse complex systems, for three reasons modelling and simulation was chosen for performance modelling in this work;

1. This work targets rather complex systems - as methodologies to eliminate data integration complexities are suggested,
2. Less-complex systems bring smaller efforts on modelling and simulation,
3. Modelling and simulation helps with performance analysis, data apparatus, and decision analysis, i.e. 'what-if' scenario.

For simulation of manufacturing process, first building of the model of the process has to be accomplished.

5.2.2.3.2. Modelling and simulation – stochastic VS deterministic

In general, modelling methods can be classified into three major types; mathematical models, graphical models, and physical models (Rembold, 1993). Due to the nature of their applications, neither mathematical models nor physical models can be used for real-time measurement of the behaviour of a manufacturing system. Only graphic models could symbolically represent the manufacturing operations. Therefore, graphic modelling was a suitable candidate for the purpose of this research.

The majority of manufacturing processes are discrete events and must be described as such (Rembold, 1993). Simulation of discrete events facilitates the stochastic behaviour of flow of products or information within the process. Moreover, availability of high-level DES tools integrated in the graphic modelling tools supported selection of graphic DES simulation method.

5.2.2.4. The entities in scenario definition layer

Entities of the Scenario Definition Layer (SDL) must help with the definition and execution of environment in which the Performance Model in the PDL could run a defined scenario other than the one of real-time based on the processed information received from the real-time. Therefore, the entities are in close relationship with the Performance Model entity and the received information. Fast-forward information which fed the Performance Model and a curve-fitting functionality which formed fast-forward information from real-time information are the main entities of the SDL. In order to maintain the flexibility of the solution, the facility to define fast-forward information settings and execute fast-forward operation is also among the entities of this layer. In Figure 5-8 the entities of the SDL are shown.

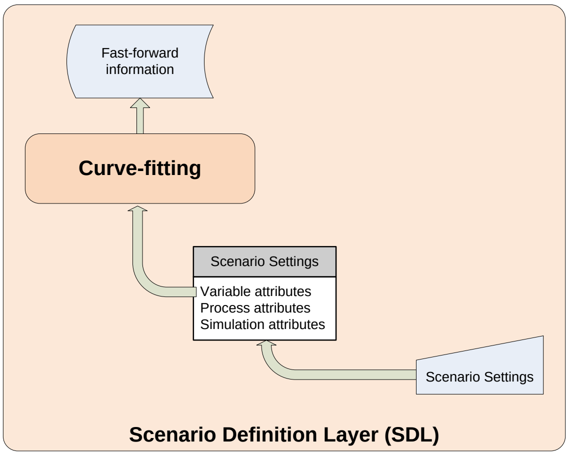


Figure 5-8 Entities in scenario definition layer

5.2.3. The services provided by each layer

The main functionality of each layer with respect to its adjacent layers was about the structure and flow of data. The collection of these services was focused on the proper use of data and time. Starting from the DDL, the service for the VDL was ensuring that each data input holds a valid value on the data buffers. The VDL provided the same service for the PDL. The VDL also provides collection attributes for variables via its variable selection module. The variable collection attributes are also used by the DDL to influence its data collection attributes at data connection entity. The main service of the PDL, in addition to the provision of data apparatus for user, is provision of Key Performance Factors (KPFs) for both user and VDL layer. The SDL served the PDL by providing the set of fast-forward information.

5.2.4. The protocol used to communicate between entities

Certain rules and formats are proposed between the entities of each layer as well as between the services of the adjacent layers. In terms of input definition at each of the two data integration layers (i.e. DDL, VDL), a generic input data model that deals with acquiring data of different types on a regular basis is implemented. In terms of flow of data or variable between the layers, the computational cost dictates the simple structure of {value, ID, time} format, i.e. values with their unique identification and timestamp. In both situations, i.e. data/variable input definition and data flow, the type of access to the data is of navigation type (Rembold, 1993) in contrast to relational type. This is because after the definition of data and variables in the system, use of each data value takes only a certain route and only once. This route starts from Data Connection, Data Acquisition and Data Buffer entities in DDL, and continues in Variable Evaluation and Variable Buffer entities. Once a data value passed this route, it is replaced by a new value in the next iteration in the same way. This solid access to certain memory spaces is formed after the definitions of data points and variables are completed in the system. Query-based access via a database server does not fit the temporary and real-time data storage requirements within DDL and VDL. Time constrained data acquisition, limited bandwidth of communication channels, and pre-processing efforts on the sampled data contribute to the computational overhead. Simple data structure with low amount of overhead information helps balanced utilisation of computational resources.

The defined rules apply limits to the services to ensure data is integrated and connected robustly, i.e. data format and data flow destinations remain valid even though if the service does not complete accurately. The following sub-sections describe the data formats at each layer and the rules by which the services run between the entities.

5.2.4.1. Data Input format

As shown in Figure 5-9, the format of the Data Input is in a data table consisting of four main elements;

- Data name; was the name of the Data Input. This piece of information remained unique throughout the definition of data inputs.
- Data type; as the type of the Data Input, defined the amount of computational memory which was allocated to the Data Input.
- Data connection attributes; these values defined the type and specification of the physical connection between the data acquisition module and the sensory device of the real world. Variety of definitions of these values, as well as the data type, supported flexibility of the data integration system.
- Data collection attributes; a set of values which supported the rules of data acquisition, for example, limits on the sampling rate, or special signal conditioning parameters, and a unique identification code which supported the communication of the data values between DDL and VDL.

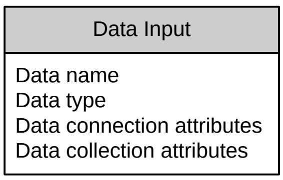


Figure 5-9 Elements of input data structure

Data input format, once defined, should not change throughout the execution of the data integration system.

5.2.4.2. Data Buffer format

The acquired data is buffered in Data Buffer based on one space per each Data Input. This implies that the value of the Data Inputs is overwritten on the same memory space on the acquisition of a new value. This rule provides an image of the real-time output of the sensory device on the era of the data integration system and available to the fusion at VDL. The structure of this real-time data consisted three pieces of information;

- Data ID (DID); this identification code was defined as a part of the data collection attributes. DID helped with tracking and collection of the values of Data Inputs from Data Buffers by Variable Evaluation module in the next layer (VDL).
- Data timestamp; helped with the tracking the timeliness of the Data Acquisition task on the particular data input, which in turn, helped with the estimation of the amount of computational effort needed for its acquisition.
- Data value; was the sampled value of the sensory output updated regularly.

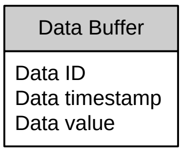


Figure 5-10 Elements of data buffer structure

The allocated memory space for DID and data timestamp are fixed. However, the allocated memory space of the data value depended on the data type, which is a factor in the estimation of the computational cost of the data acquisition.

5.2.4.3. Variable Input format

As shown in Figure 5-11, the format of the Variable Input resides in one data table consisting four main elements;

- Variable name; was the name of the Variable Input. This piece of information remained unique throughout the definition of variable inputs.
- Variable type; as the type of the Variable Input, defined the amount of computational memory which was allocated to the Variable Input.

- Variable evaluation formula; this field defined the analytical relationship between the data inputs and the variable input. Variety of available mathematical functions in the definition of this field, as well as the data type, supported flexibility of the data integration system.
- Variable collection attributes; a set of values which supported the behaviour of the variable evaluation, for example, limits and typical sampling rate, and a unique identification code which supported the communication of the variable values between VDL and PDL.

Variable Input
Variable name Variable type Variable evaluation formula Variable collection attributes

Figure 5-11 Elements of input variable structure

Variable input format, once defined, should not change throughout the execution time.

5.2.4.4. Variable Buffer format

The evaluated variable is buffered in Variable Buffer. Memory allocation mechanism in a first-in-first-out (FIFO) queuing mode is placed to store the evaluated values of the Variable Inputs until they are collected by the next layer (i.e. PDL). This rule guaranties connection and association of events in the Performance Model of the processing layer (PDL) to the consistent series of sampled data so that the Variable Selection module in PDL works accurately. The structure of the Variable Buffer consists of three pieces of information;

- Variable ID (VID); this identification code was defined as a part of the variable collection attributes. VID helped with tracking and collection of the values of Variable Inputs from Variable Buffer by Real-time Connection module in the next layer (PDL).
- Variable timestamp; helped with the tracking the timeliness of the Variable Evaluation task on the particular variable input, which in turn, helped with the estimation of the amount of computational effort needed for its evaluation.

- Variable value; was the actual evaluated value of the variable input updated regularly.

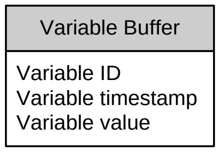


Figure 5-12 Elements of variable buffer structure

The allocated memory space for VID and variable timestamp are fixed. However, the allocated memory space of the variable value depended on the variable type, which was a factor in the estimation of the computational cost of the variable evaluation.

5.2.4.5. Real-time model matching mechanism (R3M)

For real-time simulation, a connected mechanism is created in order to take care of the real-time association of evaluated variables to their destination on the simulated model. In this mechanism, called Real-time Model Matching Mechanism (R3M) values of evaluated variables are sent to the corresponding variables in the performance model according to the attributes of Variable Buffer. Performance model variables further assist the model in any of the three ways, like schematically shown in Figure 5-13;

- Creating entity in the model; new entity starts flowing in the simulation model,
- Triggering model process; the waiting process continues, in this case a real-time event flow is established,
- Changes in the status of an entity or resource, model variable represented a parameter of the modelled system.

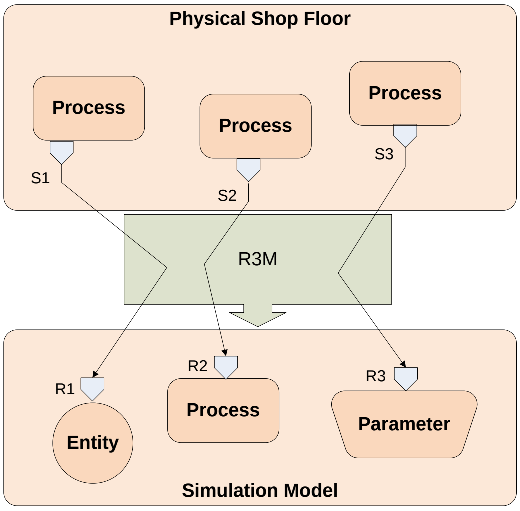


Figure 5-13 R3M establishment between variables inputs S1, S2, S3, and model variables R1, R2, R3

The process flow on the simulated model runs continuously and receives discrete events from shop floor activities occurring at certain points. Creation and progress of each entity or batch of entities depends on the contents of the message which is sent to the Simulation Model.

5.2.4.6. Real-time connection settings

Connections between the Variable Inputs and the Performance Model were established in Real-time Connection entity of the PDL based on the definitions of the Variable Buffer and the location of the process or data structure in the Performance Model.

Real-time Connection Settings
Variable attributes Model connection attributes

Figure 5-14 Elements of real-time connection settings

The attributes of variables and the model connection support and guaranty two tasks in the establishment of the real-time connection;

1. Identification of the model reaction; an attribute of the model connection is coded in accordance with the type of the reaction in the simulated model. This attribute is extracted first and decided upon if the reaction was entity creation, process progress, or parameter setting.

2. Extracting key information; depending on the type of the reaction, VID and value of variable input are assigned to the independent model variables or model entity attributes. The combination of VID and variable value guaranty the uniqueness of this assignment.

5.2.4.7. Fast-forward simulation

Four conditions need to be met in order to run a fast-forward simulation;

1. Isolation; interaction with the real world need to be disconnected. This is achieved by Real-time Connection entity.
2. Curve fitting; a separate entity in the scenario definition layer (SDL) fits and associates parameters of model behaviour according to the collected model parameters during the real-time run up to the time of starting fast-forward simulation. Only those parameters are curve fitted that are supplied from the real-time information, i.e. values of Variable Buffer. This could occur whilst the real-time simulation is running.
3. Scenario definition; it is defined which values of the Variable Buffer to be modified before supplying the Simulation Model.
4. Fast-forward information supply; defined and curve fitted parameters were loaded by the Simulation Model and assigned to the corresponding model variables as in real-time.

5.3. Summary of the proposed architecture

Concepts of a layered system architecture, FDILA, was proposed in this chapter to address generic data integration platform through facilitating flexibility in definition of data input sources and data connections between pre-processing and processing stages.

Efficiency in promptness of the proposed integration architecture was provided by decentralising and distributing services in different layers and temporary buffering of data in first-in-first-out queues between each layer. This way, no centralised database provides data for each service upon enquiry.

Distinct execution of services with one type of data transformation at each separate layer provided accurate and clear account of functionalities of data integration.

Connection setting between Data Definition Layer (DDL) and data sources is not limited to specific type of communication protocol or hardware to keep with generality and flexibility of the solution. OPC service via Ethernet, however, is a recommended type of access to remote data sources. DDL and Variable Definition Layer (VDL) are executed in the same software tool to speed up data communication. Between VDL and Data Processing Layer (DPL), as they may execute in two separate software tools (data acquisition and simulation), Ethernet protocol is recommended over the other slower protocols.

Since definition of services, functionalities, and layers are data oriented, no execution of a service occurs at ‘no-data’ situation. Data queuing strategy between layers avoids waiting services. Overwriting of unnecessary data avoids extra efforts on garbage data space consumption. Event flow nature of connection between variables and performance model avoids extra execution of performance model for repeated values of acquired data. All above help with consumption of less energy and time of computation, thus, less complex data integration.

The proposed conceptual architecture took dynamics of functionalities on data into the account of **efficiency of data integration**. The next attempt of this research provides means to decide on the **efficiency of integrated data** itself.

6. Event Tracking Sensitivity Analysis (EvenTracker)

After proposing a generic and flexible data integration system architecture, this research puts forward a methodology that helps with cost reduction of computations involved in data integration by focusing on the use of input data as events when studying and measuring the level of their influence on system output. Such methodology demands a universal selection criterion for key system inputs.

Mousavi et al (2007) provided a platform that can incorporate the information based on a cost analysis format to help decision maker make quick decisions on daily activities. They focused on food manufacturing industry and analytical cost models as performance measurement of systems. The generic nature of the solution in this thesis is assumed to cover a wider range of industries. Also the model which provides performance factors is discrete-event simulation based and executes in real-time. Therefore, the main challenges here come from minimal data type assumptions and time constraint.

In principle, it is highly desirable to gain maximum benefit from the least number of data sources. It must be decided to what extent a model output is driven by each of its linked data variables. This type of activity is called Sensitivity Analysis (SA) (Saltelli, 2002).

Competency of sensitivity analysis techniques was reviewed in chapter 4 and understood to decline by increasing size and complexity of integrated models, large number and heterogeneity of the model input data series (Ravalico et al, 2005). A suitable technique for sensitivity analysis in generic situations was recommended to be as model-free as possible (Saltelli et al, 2004).

In order to overcome the shortcomings of the existing SA methods, an effective and efficient way for sensitivity analysis of data in two time series is introduced in this chapter. In the following sections detailed description of event-driven data types and their impact on the sensitivity analysis is provided. The proposed event tracking sensitivity analysis (EvenTracker) and its application in a typical manufacturing case are then explained.

6.1. EvenTracker mechanism

EvenTracker defines an input and output occurrence matrix [+, -] at pre-specified time intervals. This matrix subsequently describes the relationships between causes that trigger events (trigger data) and the actual events (event data) enabling to construct a discrete event framework for sensitivity analysis.

A short description of discrete event system, together with the definition of trigger data, and event data are provided in the following subsections.

6.1.1. Discrete event systems

As opposed to continuous system, Discrete-Event System (DES) is defined by disparate occurrence of events in a specified time span (Banks, 2001). In other words, the system state change is determined by the changes of the input variables at specified intervals. Each state transition of the system is called an event. Therefore, in DES, only the attributes that represent the occurrence of an event are considered. These attributes are discussed in the following section.

6.1.2. Trigger data and event data

Based on the definition of DES, any input variable to a DES whose value transition is responsible for an event in the system is defined as a Trigger Data (TD). Accordingly, the series of data which represent the state of the system is defined as Event Data (ED). It is possible that the number of EDs and TDs in a system to be different. For example, a number of TD series may be responsible for changing a single ED series. However, various TD series could have different impact on an ED series.

$$ED \because \{TD_1, TD_2, \ldots, TD_n\} \qquad \textbf{(6-1)}$$

6.1.3. An example of a baking process

An example here would help to explain the underpinning rationale for the proposed sensitivity analysis method.

One of the methods to detect system state transitions is to detect and track the changes that occur with an input variable. Figure 6-1 is a simplified illustration of a baking machine with a single heater. Two light reflector sensors (S01 and S02) and a temperature sensor (T01) are installed on the machine. The sensors send signals to the EvenTracker software model. Sensors S01 and S02 provide the data about the entry and exit of components from the baking machine. Their signal data therefore carries either a no-voltage (i.e. binary value of 0) showing the heater in sleep mode, or a pulse of voltage (i.e. binary value of 1) as an indication of the heater being on for the duration of baking (Baking Time) i.e. observed variable. The third sensor (temperature), T01, provides the data about the temperature of the component during and after the heating process. Its value is therefore an analogue value fluctuating in a range between 22° and 70° C.

The combination of the data provided by the three sensors will be used to measure two production process performance factors i.e. latent variable. These performance factors are for example instantaneous Resource Utilisation (RU) and Product Quality (PQ) based on temperature properties of the baking dough.

The baking machine utilisation (i.e. RU01) can be defined as the ratio of the number of times that the heater is occupied to the capacity of the baking machine (Kelton, 2009) (capacity is one in this example).

Product Quality (PQ01) is defined as the average fluctuation of temperature from centre to the surface measured by a temperature probe on a sample. The quality of product is determined by the average instantaneous reading of the temperature.

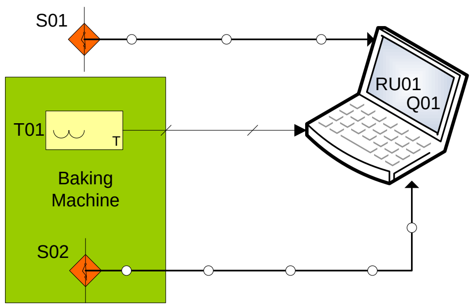


Figure 6-1 A hypothetical Baking System with three sensors

Figure 6-2 shows the relationship between each event triggered by S01 and S02 with the changes in RU01. Any change within a pre-specified range (e.g. 1%) to the RU01 in a given time span can be expressed as an event and the positive level transition (binary 0 to 1 or positive edge) of both S01 and S02 as triggers, then RU01 can be defined as event data (ED). Both S01 and S02 can therefore be considered as trigger data (TD).

Similarly, value of PQ01 changes with T01 (Figure 6-3). Definition of an event will be when a temperature change exceeds a defined threshold representing an ED (e.g. 3%). The changes equal or above a certain threshold on T01 is considered as a TD.

$$\begin{aligned} &if\,(T01_t - T01_{t-1}) \geq \theta \xrightarrow{\text{(Trigger)}} TD_t \\ &if\,(PQ01_t - PQ01_{t-1}) \geq \psi \xrightarrow{\text{(Event)}} ED_t \end{aligned} \qquad \textbf{(6-2)}$$

where $T01_t$ is the temperature at time t, θ is the temperature change threshold, $PQ01_t$ is the product quality at time t, and ψ is the quality change threshold.

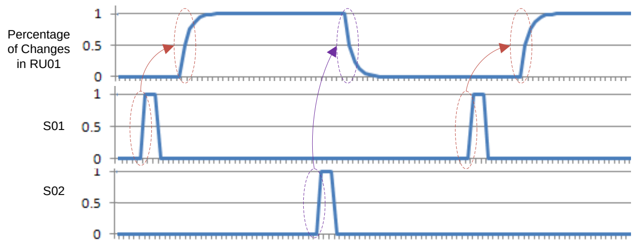


Figure 6-2 Causal relationship between two switch signal data S01, S02, and the performance factor RU01

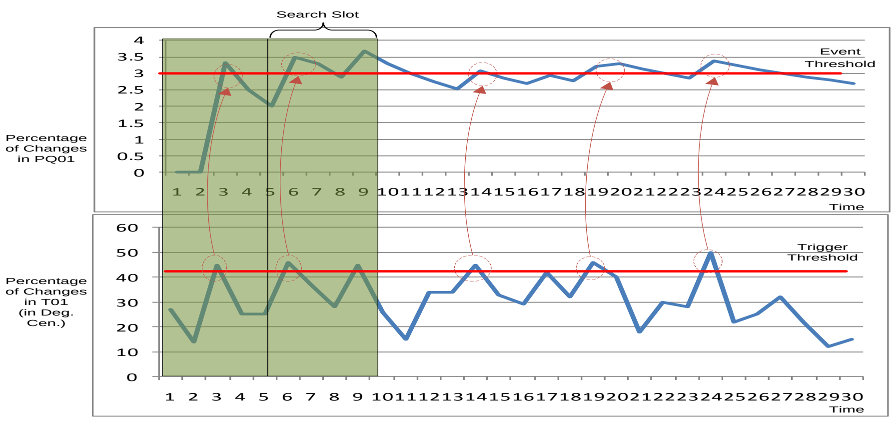


Figure 6-3 Causal relationship between Temperature sensor signal T01 and the performance factor PQ01

6.2. Methods and parameters for EvenTracker

There are four parameters that make EvenTracker functional. Two parameters, Search Slot (SS) and Analysis Span (AS), are about the time span of transition detection and the overall system state analysis, Event Threshold (ET) and Trigger Threshold (TT), are about tracing the values of the acquired data series, and the other two parameters. These parameters are defined at the following, the assumptions and logic of the method is discussed in later sections.

6.2.1. Search slot

The SS is the fixed time span that batches of TD_i and ED_i are captured. It can also be described as the scan rate. The SS is shown as the width of the green rectangles in Figure 6-3.

6.2.2. Analysis span

The AS is the time span in which a period of sensitivity analysis takes place. AS comprises of a number of consecutive SS_i. The number of TD and ED observation will then be used to apply sensitivity indices at the end of an Analysis Span. This will be a new weight to be allocated to the new TD. This means at any AS there is the possibility of weight of the TD_t to be different from TD_{t+1}.

6.2.3. Event threshold

Each transition between subsequent values of Event Data (ED) series is examined by Event Threshold (ET). This value, as shown by a horizontal red line in the upper diagram of Figure 6-3, is a proportion of an overall range of values of the ED series over the Analysis Span (AS). It is therefore expressed in the form of a percentage.

6.2.4. Trigger threshold

TT is a value against which the values of Trigger Data (TD) series are examined. In Figure 6-3, TT is shown by a horizontal red line on the lower chart. TT, like ET, is a proportion or percentage of an overall range of values of TD series over the analysis span.

ET and TT determine whether a signal represents a real change in the system state that can be interpreted as an event.

6.3. The assumptions of the proposed method

There are a number of assumptions that need to be made before implementing the EvenTracker. These assumptions can be listed as:

6.3.1. Assumption 1: Triggers and events

The assumption here is that only those fluctuations of data series that are interpreted as trigger (for TD data series) and as event (for ED data series) are considered for further processing. The base for this interpretation is the threshold (ET and TT) settings (Tavakoli et al, 2010).

6.3.2. Assumption 2: Thresholds

Thresholds are the pre-specified short term range of signal fluctuation for every data series. These defined ranges are introduced as ET and DT in earlier section and are addressed once in each Search Slot. ET and DT are evaluated only post Analysis Span period as the assumption is that a fluctuation of data series is likely to occur during this period. Therefore, a trigger or an event occurs when the difference between the maximum value and the minimum value of a data series within a Search Slot meets or exceeds the threshold defined for that data specific series.

6.3.3. Assumption 3: Homogeneity of data series

The value of threshold for each data series remains fixed within the Analysis Span (AS). This implies that in all search slots of a single data series the range of possible values of transition that occur is assumed to be the same. In other words, each data series is assumed to have the same probability distribution over all Analysis Span.

6.4. EvenTracker algorithm

The EvenTracker algorithm is designed to respond quickly. The algorithm's life cycle is equivalent to an AS. This life cycle is divided into several SS intervals. Within each SS interval, the method considers two batches of time-bound data that were captured from two time series and provides a value which is translated into sensitivity index according to the logic introduced at the following subsections. A sensitivity index is added to the indices of subsequent search slots. At the end of each SS, the sensitivity indices of all data series are normalised. The main functions of EvenTracker algorithm are shown in Figure 6-4 and Figure 6-4. The main steps of the algorithm are:

6.4.1. Stepwise scan

A First-In-First-Out queue memory space is allocated for every batch of data in a search slot. The size of the queues is unlimited. The content of the queues are flushed out at the end of the search slot.

The data is then passed to the EvenTracker detection and scoring algorithm. The next search slot continues to fill the queue immediately. Using this technique no data would be lost. Figure 6-4 shows three stepwise scans and their analysis operations in three consequent search slots.

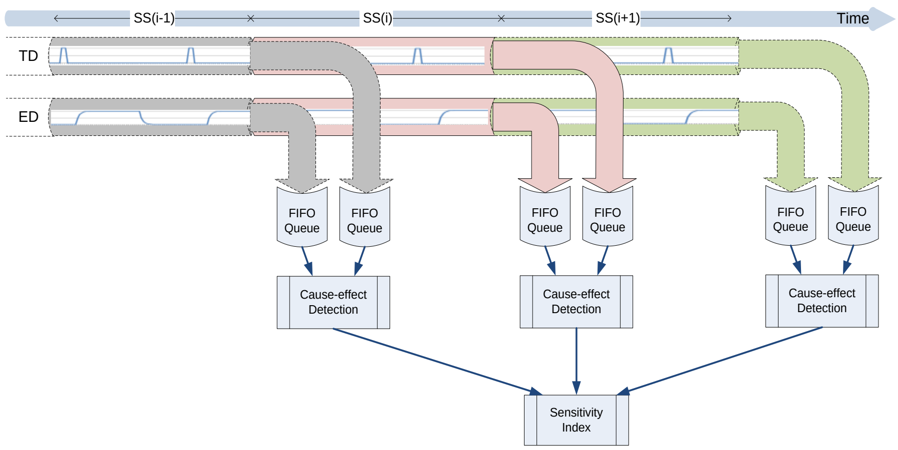


Figure 6-4 Overall functionality diagram of EvenTracker algorithm

6.4.2. Trigger-Event detection

As shown in Figure 6-5, within each search slot a pair of two data batches is examined for evidence of trigger and event. For this, the data batch of TD is searched for fluctuations larger than the specified trigger threshold, and data values of ED are checked for changes larger than Event Threshold. This functionality results a true value if at least one of these changes is found in the batch.

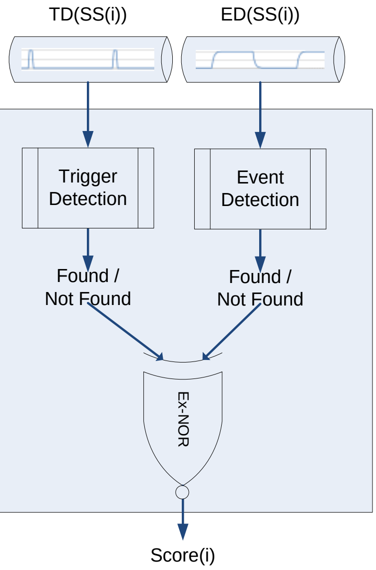


Figure 6-5 Trigger-Event Detection functionality on each Search Slot

6.4.3. Two-way matching score

In each search slot simultaneous existence or non-existence of a change in each pair of data batches is scored as +1, otherwise the score would be -1. This operation is similar to a weighted logical Exclusive-NOR as shown in Table 6-1. Based on this approach, the impact of the inputs on a given output is extracted.

Table 6-1 Weighted Exlusive-NOR functionality

Input 1	Input 2	Output
0	0	+1
0	1	-1
1	0	-1
1	1	+1

6.4.4. Summation of two-way matching scores

The +1 and -1score for each search slot is added up to the overall score as in equation 6-3 which is in fact, the sensitivity index of the measured ED with respect to the measured TD after time t, or in discrete form, after search slot n. Where n is the number of search slots after time t. Sensitivity Index (SI) can therefore be calculated as:

$$SI_{(t)} = \sum_{1}^{n} Search\ Slot\ Scores \qquad \textbf{(6-3)}$$

6.4.5. Normalisation

At the end of each search slot the magnitudes of sensitivity indices are linearly scaled to the unit range. In other words, given a lower bound l and an upper bound u for the set of all indices in the search slot, each final value of sensitivity index is transformed to a value in the range of [0,1]; thus:

$$\tilde{S} = \frac{SI - l}{u - l} \qquad \textbf{(6-4)}$$

The linear scaling transformation helps with easier ranking of the indices while maintaining their order intact. A summary of the algorithm is demonstrated in Table 6-2. In this table the flow of matching scores and sensitivity indices (SI1, SI2, SI3) of one event data (ED1) with respect to three trigger data (TD1, TD2, TD3) during 10 search slots could be seen. Star symbols in Table 6-2 represent a detected event or trigger in the values of ED1, TD1, TD2, or TD3 on each search slot. Each value of Score1, Score2, or Score3 is -1 or +1 depending on the exclusive match between ED1 and TD1, TD2, or TD3 respectively. Normalised sensitivity indices SIn1-SIn3 are the normalised values of SI1-SI3 according to the equation 6-4.

Table 6-2 Production of sensitivity indices SI1, SI2, and SI3 and normalized sensitivity indices SIn1, SIn2, and SIn3 for event data ED1 with respect to trigger data TD1, TD2, and TD3 in 10 search slots by using EVENTRACKER method. Star mark shows a match and S1

Search Slot	0	1	2	3	4	5	6	7	8	9	10
ED1	*	*		*		*	*	*		*	*
TD1			*			*			*	*	*
S1	-1	-1	-1	-1	1	1	-1	-1	-1	1	1
SI1	-1	-2	-3	-4	-3	-2	-3	-4	-5	-4	-3
SIn1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
TD2	*				*	*	*	*		*	
S2	1	-1	1	-1	-1	1	1	1	1	1	-1
SI2	1	0	1	0	-1	0	1	2	3	4	3
SIn2	1.00	1.00	1.00	0.67	0.33	0.33	0.67	0.75	0.80	0.80	0.75
TD3		*		*		*		*		*	
S3	-1	1	1	1	1	1	-1	1	1	1	-1
SI3	-1	0	1	2	3	4	3	4	5	6	5
SIn3	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

The normalised sensitivity indices (SIn) in Table 6-2 show that ED1 is the most sensitive to TD3 and the least sensitive to TD1 in almost all search slots. Diagram in Figure 6-6 shows the values of SIn.

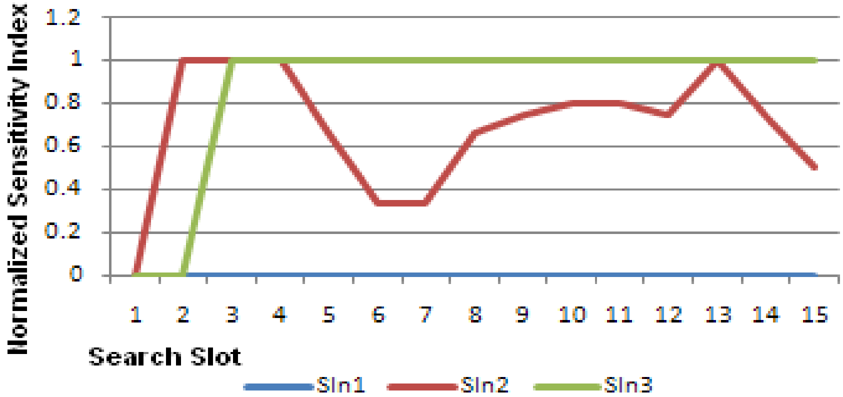


Figure 6-6 Normalised sensitivity indices as in Table 6-2

The overall average values of normalised sensitivity indices as shown in Figure 6-7 indicate the lateral convergence of the indices towards a specific value analogues to steady state.

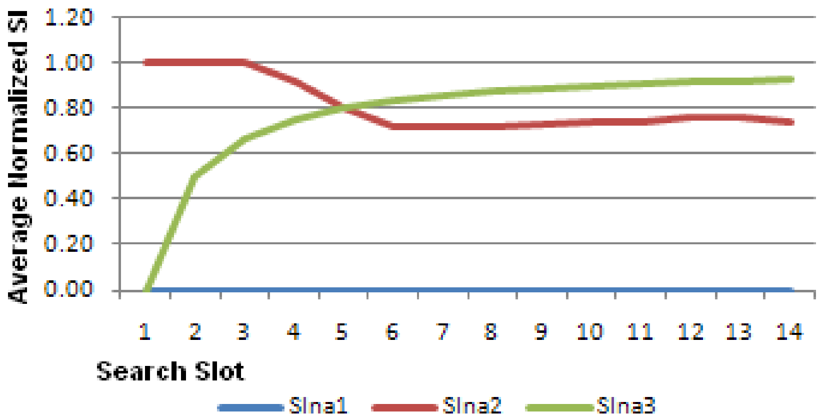


Figure 6-7 Averaged normalised sensitivity indices as in Table 6-2

In cases when the normalised indices do not truly represent the values of sensitivity index, one can instead adopt the instantaneous value. Using current normalised values of sensitivity indices or their historical average values can be decided by the system analyst.

6.5. Conclusion on EvenTracker sensitivity analysis

EvenTracker method has been described in this chapter as a solution to sensitivity analysis problem in high complexity systems with time-constraint. The approach took advantage of event-based definition of data involved in process flow. The underpinning logic behind the event-tracking-based sensitivity analysis (EvenTracker) method is the capturing of the cause-effect relationships between triggers (input variables) and events (output variables) in a specified period of time. The approach does not require estimating data distribution of any kind (Figure 6-8). Neither the performance model requires execution more than in real-time.

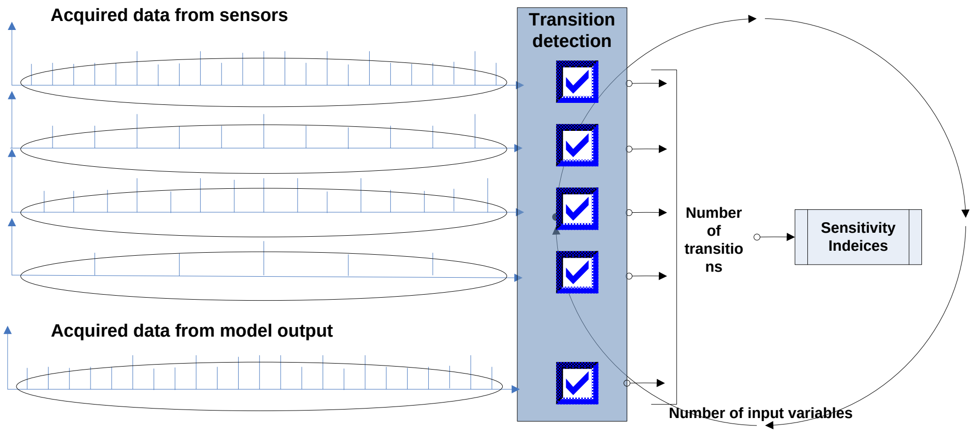


Figure 6-8 General view of EvenTracker method for sensitivity analysis

The key feature of the proposed method is the quick information about unimportant data that at times may overwhelm the data processing platforms. It may be safe to claim that with regard to the time domain, EvenTracker method may be classified as a Local Sensitivity Analysis method. Moreover, to estimate sensitivity indices, EvenTracker method does not require any prior knowledge about the analytical relationship between input and output variables. EvenTracker, in this sense, can be considered as a Global Sensitivity Analysis method or better say model-free method. The advantages and the application of EvenTracker in systems analysis are discussed in the next chapter.

7. A Framework for Measuring the Complexity and Efficiency of Sensitivity Analysis Methods

The purpose of this chapter is to suggest a platform for comparing EvenTracker introduced in chapter 6 to the other major Sensitivity Analysis Methods (SAM). Selection of a SAM in an application requires the analyst to consider a series of criteria as discussed by Tavakoli et al (2010). They include: the analytical relationship between the system's inputs and the outputs, the statistical distribution of input variables, and computational overhead.

An important question to address in the process of evaluation of a computational method is to establish the appropriate computer model for estimation and the appropriate computation parameters (Cook, 1983). The principles of complexity measurement and comparison adopted in this thesis are implemented through the following three steps:

- Step 1: Design a generic computational framework that allows for cross comparison of various SAM.
- Step 2: Propose and embed in the proposed framework a number of criteria that allows for measuring the computational complexity of different SAM.
- Step 3: Run tests to measure and compare the computational complexity of each SAM in accordance with the set evaluation criteria defined in step 2.

7.1. The Generic sensitivity analysis method framework

In this chapter the design of a generic platform for comparing different SAM is proposed.

Firstly, according to Cobham (1965) the key "steps" for the computation process in a SAM will be outlined. In its most generic form the function of any SAM is to generate sensitivity indices for a given output variable with respect to the system input variables. SAM conducts its function based on system model equations, distribution estimation of input data, input data sample generation, evaluation of the resultant outputs or a combination of these tasks.

Finally, SAM produces a quantitative sensitivity index value that is presented in Figure 7-1.

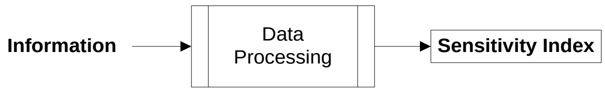


Figure 7-1 The most generic structure of a SAM

Although Figure 7-1 suggests a common structure which describes all sensitivity analysis processes as single-step operation, this view seems to be too simplistic for differentiating the capabilities and efficiencies between different sensitivity analysis methods. In this chapter a selection of the most popular SAM computational processes are appraised and in specific, comparisons are made with the EvenTracker technique.

Conventional SAMs fall into four categories: sensitivity testing, analytical methods, sampling-based methods, and computer algebra-based methods (Isukapalli, 1999).

7.1.1. The Computational structure of sensitivity testing methods

Sensitivity testing process consists of two steps. The first step is to produce an iterative production of model outputs based on the manipulated changes of the model inputs one-at-a-time (Isukapalli, 1999). The second step is to calculate sensitivity indices by applying a series of algebraic matrix operations on the manipulated model inputs and the generated output values after having had produced all the necessary and expected output values.

The sensitivity testing process can be shown in Figure 7-2 as a two-process task; the first occurring multiple times and the next process only once.

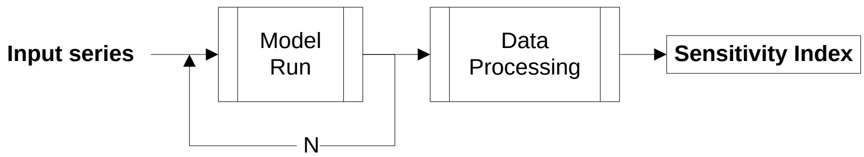


Figure 7-2 Overall structure of Sensitivity Testing method

7.1.2. The computational structure of analytical methods

The common feature of most of the important analytical SAMs is in their access to the system's model equations (Papoulis, 1991; Yang, 2003) (Faghihi et al, 2004). Some analytical SAMs expand the model equations and find the model's inverse equations i.e. Differential Analysis methods (Isukapalli, 1999), or solve matrix equations i.e. Spectral Based Stochastic Finite Element method (Papoulis, 1991). Some analytical SAMs differentiate system's model equations and solve sensitivity equations using auxiliary differential equation i.e. Green's Function (Li, 1998; Yang, 2003), and Coupled/Decoupled Direct method (Faghihi et al, 2004; Dunker et al, 2002).

Figure 7-3 shows the overall functionality of analytical SAMs.

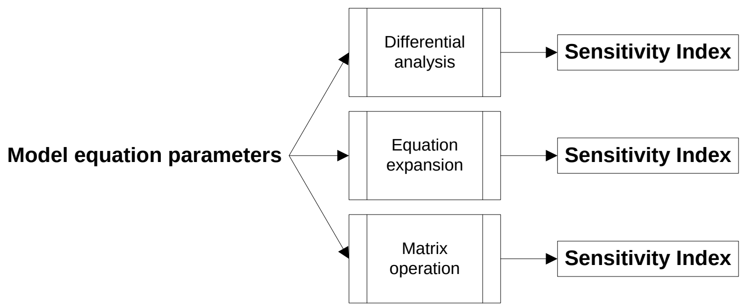


Figure 7-3 Overall structure of Analytical SAM

7.1.3. The computational structure of sampling-based methods

In sampling based SAMs, samples of input data series are used to derive a number of system attributes by mapping the distributions of the data series using statistical analysis methods (Xu et al, 2008; Cukier et al, 1978; Jin et al, 2007; Braddock et al, 2006; Sobol, 2001). These attributes, in turn, are used to synthesis new data samples. Multiple runs of sample generation are then used to generate output data series. The output data series is then used to infer the attributes of its distribution. Finally, sensitivity indices are provided based on the attributes of input and output data series. Figure 7-4 shows a version of the above scenario.

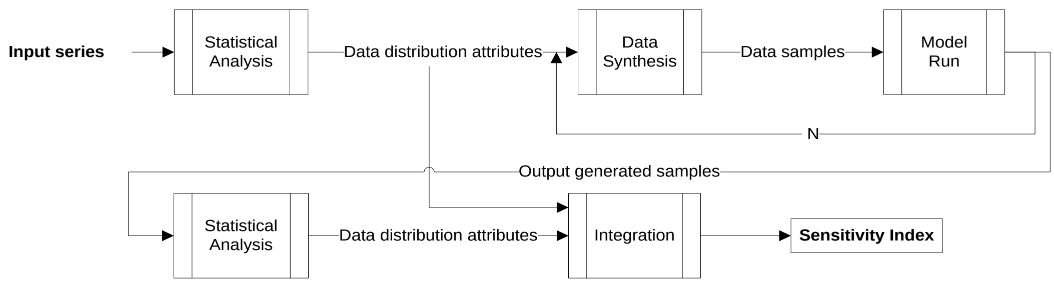


Figure 7-4 Overall structure of Sampling based SAM

As shown in

Figure 7-4, the statistical analysis block is used to produce the statistical distribution parameters using the input data series.

7.1.4. The computational structure of computer algebra-based methods

Computer algebra-based SAM or the so called Automatic or Algorithmic Differentiation (AD) is based on the direct manipulation of the computer code of the system's model. The first step in this method is to process the source code of the model by traversing the chain rule (Apostol, 1974) forwards or backwards so that a 'derivative code is generated (Griewank et al, 2008). In the second step the derivative code is seeded once per each input variable to accumulate the changes in model outputs and estimate indices of sensitivity.

Computer algebra-based SAMs in general require having access to the source code that describes the behaviour of the model encompassing input and output data that are used for the sensitivity analysis. Figure 7-5 Overall structure of Computer Algebra based SAM shows the process of computer algebra-based sensitivity analysis.

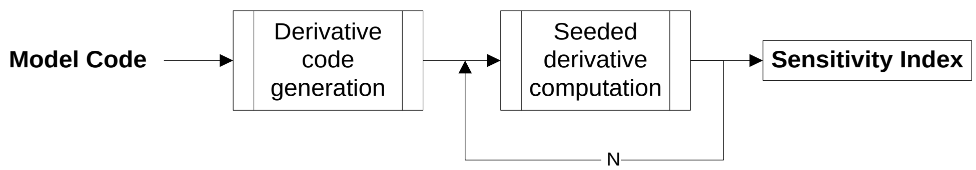


Figure 7-5 Overall structure of Computer Algebra based SAM

7.1.5. The computational structure of EvenTracker SAM

EvenTracker SAM, like sensitivity testing method and sampling-based method, is a causal statistical analysis method, but unlike the two aforementioned methods, it is not an experimental method, i.e. EvenTracker method does not manipulate data for statistical measurements. In other words The EvenTracker method does not fall into the empirical methods and does not rely on historical data. Instead, EvenTracker method performs statistical measurements on the changes of the actual real-time generated data which are produced by system input sources. Figure 7-6 shows the overall process of EvenTracker sensitivity analysis method.

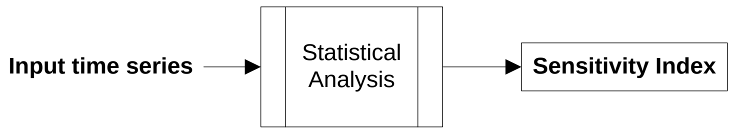


Figure 7-6 Overall structure of EvenTracker SAM

7.1.6. The common building blocks of SAMs

From the discussions in previous sections one can declare that the basic building blocks for a generic SAM evaluation platform should be able to account for the following features: (1) model data entry, (2) data pre-processing, (3) sample generation, (4) iterative data processing, and (5) data post-processing units. This common architecture for SAMs looks like the diagram in
Figure 7-7.

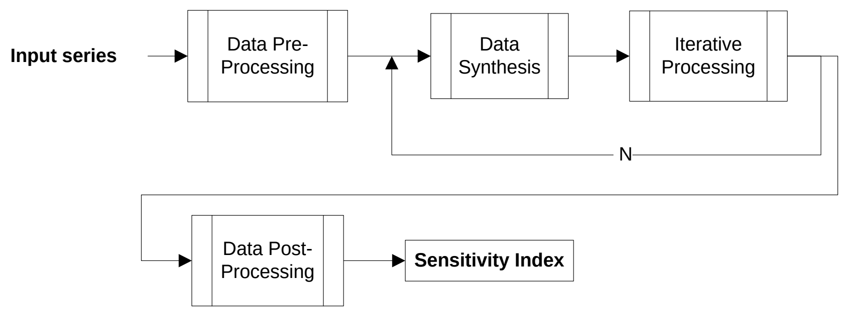


Figure 7-7 The common building blocks of SAMs

In the following section we will map the five mentioned sensitivity analysis methods (SAMs) onto the proposed common evaluation framework.

7.2. Mapping SAMs onto the generic platform

As can be seen from the mapped SAMs in Figure 7-8, for some of the discussed SAMs, some blocks in the common skeleton are not essential. For example, sensitivity testing method starts with iterative run of the model whereas in all the other categories of SAMs, a major data processing occurs immediately after receiving input data from the model.

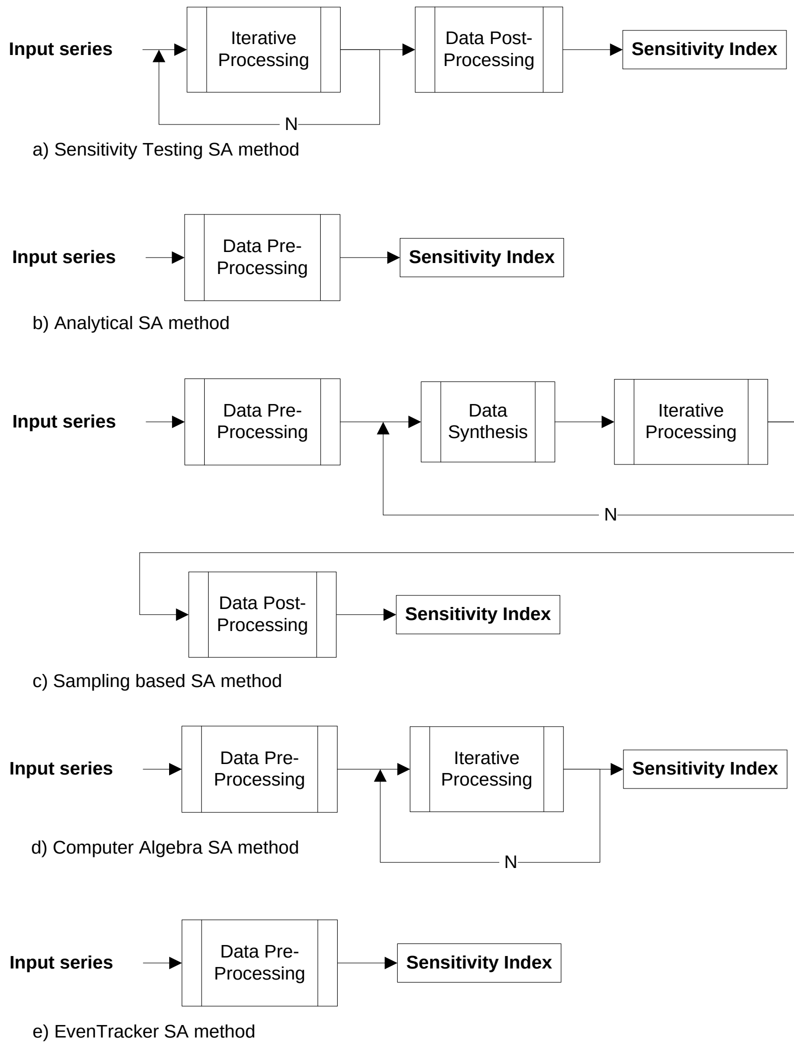


Figure 7-8 Mapped SAMs onto the common skeleton

In Figure 7-8, the number of iterative processes is denoted as N on each mapped SAM. N is different per SAM and depends on the method implementation.

The key advantage of the proposed generic platform for testing various SAM mentioned in this chapter is the capability of the platform to capture the details of the computational complexity of various SAMs. The building blocks of the common skeleton are therefore dealt with by the evaluation criteria as atomic members of a computation structure (Aimin et al, 2008).

7.3. The computational complexity

In this thesis a hybrid approach was taken for evaluation of computational complexity. This was achieved by combining the introduced measures of the computational complexity which do not depend on the details of the algorithm and instead take the overall structure and functionality of the computation problem into account. Therefore, among the complexity metric introduced in chapter 2, structural complexity, energy-time analysis, and entropy change are used to complete the evaluation of the five aforementioned sensitivity analysis methods.

A computation complexity indicator is defined here as a function of three metrics; $\mathrm{E} \times \mathrm{t}^2$, entropy change, which we call $\nabla \mathrm{Q}$, and structural computation complexity (SC).

$$\text{Complexity} \because \{\mathrm{E} \times t^2, \nabla \mathrm{Q}, \mathrm{SC}\} \qquad \textbf{(7-1)}$$

Therefore, evaluation of a given SAM would be a three-tuple measurement. Each measurement has its own approach which will be introduced in the following sections.

Figure 7-9 presents the evaluation criteria.

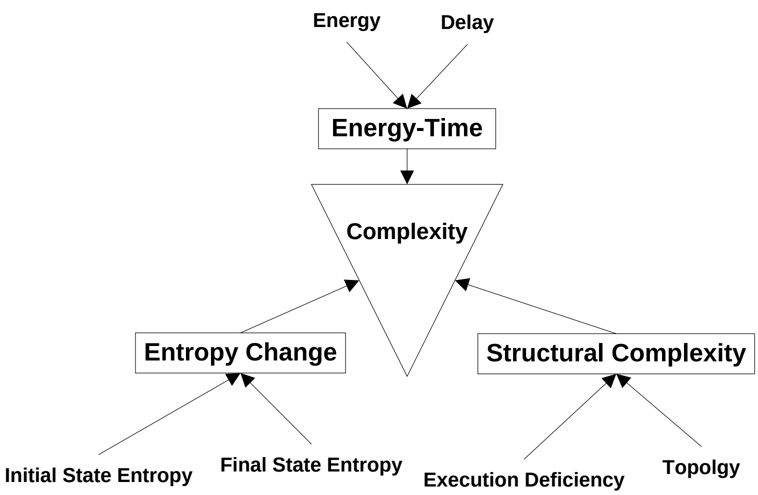


Figure 7-9 SAM complexity evaluation triangle and its inputs

7.3.1. Energy-time metric

In (Martin, 2001) the complexity metric of computing problem F is denoted by Θ and defined as in equation 7-2.

$$\Theta \stackrel{\text{def}}{=} E \times t^2 \qquad \textbf{(7-2)}$$

where E is the energy and t is the time of computing the problem F shown in Figure 7-10.

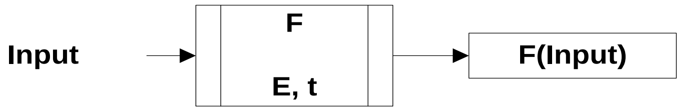


Figure 7-10 A single problem F with its associated consumed energy E and delay t

We use this Energy-Time metric to measure the lower bound of energy-delay efficiency of computation for the considered SAMs. Two basic compositions are proposed:

7.3.1.1. Sequential composition

Let's assume that the computing problem F consists of the sequential composition of N computing problems $\mathrm{F_s}$, i.e. each subsequent problem runs after completion of the previous problem. Let's assume again that all sub-problems have similar energy ($\mathrm{E_s}$) and computing time ($\mathrm{t_s}$) (Figure 7-11). Complexity of computation problem F can be calculated using equation 7-3 (Penzes et al, 2002).

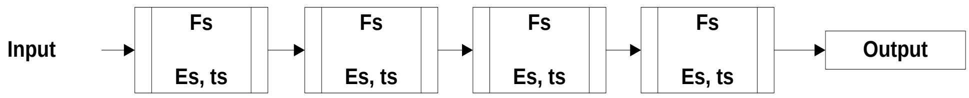


Figure 7-11 Four sequential similar problems with their associated consumed energy and delay time

$$\Theta_\mathrm{s} = 3 \times N \times \mathrm{E_s} \times (\tfrac{3}{2} \times N \times t_s)^2 \qquad \textbf{(7-3)}$$

7.3.1.2. Parallel composition

Now we assume that the computing problem F consists of parallel composition of N computing $\mathrm{F_p}$ problems with energy $\mathrm{E_p}$ and delay time $\mathrm{t_p}$ (see Figure

7-12). In this composition, complexity of computing problem F can be expressed using equation 7-4 (Penzes et al, 2002).

$$\Theta_p = 3 \times N \times E_p \times (\tfrac{3}{2} \times t_p)^2 \qquad \textbf{(7-4)}$$

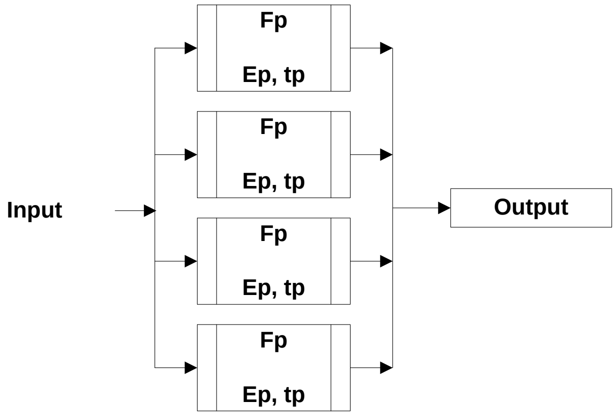


Figure 7-12 Four parallel similar problems with their associated consumed energy and delay time

If $\mathrm{E_p} = \mathrm{E_s}$ and $\mathrm{t_p} = \mathrm{t_s}$, whilst there is no communication overhead (ideal state), then the ratio of Θ_p and Θ_s would be:

$$\frac{\Theta_p}{\Theta_s} = \frac{1}{N^2} \qquad \textbf{(7-5)}$$

This is as expected if similar problems are once set in parallel and another time set in series. The complexity of computing series problems is larger than the parallel problems and one of the reasons is that it takes longer to solve.

To apply equations (7-3) and (7-4) in the complexity measurement, we assume that the building blocks of the skeleton of SAMs are atomic problems which cannot be broken into smaller problems.

7.3.2. Entropy Change

By using the Energy Analysis (EA) method the computational complexity is calculated only by initial state and final state of the problem (Peng, 2008). Therefore, EA method helps with decoupling the lower bound from details of the implementation i.e. the algorithm, allowing us to concentrate on the way each SAM defines the problem of sensitivity index estimation. Each building block of the generic SAM platform is viewed as a separate computational problem. To be able to combine the results of measuring the computational complexity of each building block of a SAM, we define properties of the measures used in the EA method.

7.3.2.1. Entropy change in sequential dependent problems

In sequential dependent problems the input to each problem is the output of the previous problem. The energy consumption of a computing problem which comprises of a number of sequential dependent computation problems depends on the initial state of the first computing problem and the final state of the last computing problem. This property of the EA is expressed by equation 7-8. For example, in the case of two sequential dependent problems is shown in Figure 7-13.

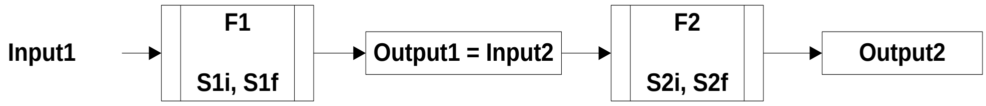


Figure 7-13 Two sequential dependent computing problems with their associated initial state and final state entropies

$$\nabla Q_1 = T(S_{1_f} - S_{1_i}) \quad \textbf{(7-6)}$$
$$\nabla Q_2 = T(S_{2_f} - S_{2_i}) \quad \textbf{(7-7)}$$

where ∇Q_1 is the F_1 problem's energy consumption, S_{1_i} is the initial and S_{1_f} the final state entropies of F_1 problem.

$$\nabla Q = T(S_{2_f} - S_{1_i}) \quad \textbf{(7-8)}$$

where ∇Q is the energy consumption of both F_1 and F_2.

7.3.2.2. Entropy change in sequential independent problems

In sequential independent problems the input to each problem is different to the output of the previous problem. The energy consumption of a computing problem which comprises of a number of sequential independent computational problems is the sum of all energy consumption for all problems. This property of the EA is expressed by equation 7-9 for two sequential independent problems shown in Figure 7-14.

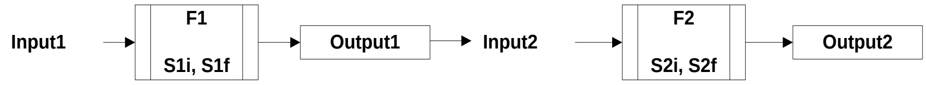


Figure 7-14 Two sequential independent computing problems with their associated initial state and final state entropies

$$\nabla Q = \nabla Q_1 + \nabla Q_2 \quad \text{(7-9)}$$

7.3.2.3. Entropy change in parallel dependent problems

In parallel dependent problems the input to each problem is a portion of the original input to the set of parallel problems, and the output of each problem to be a portion of the final output of parallel problems. The energy consumption of a computing problem which comprises of a number of parallel dependent computation problems is the sum of all energy consumption for all problems. This property of the energy analysis is expressed by equation 7-10 for two parallel dependent problems shown in Figure 7-15.

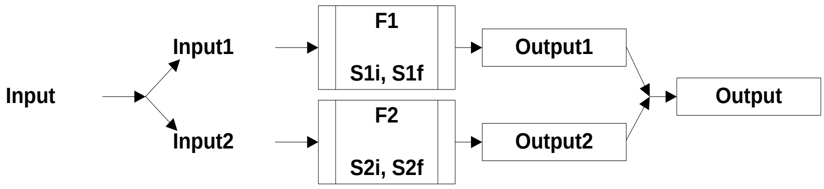


Figure 7-15 Two parallel dependent computing problems with their associated initial state and final state entropies

$$\nabla Q = \nabla Q_1 + \nabla Q_2 \quad \text{(7-10)}$$

7.3.2.4. Entropy change in parallel independent problems

In parallel independent problems the input for each problem is same as the original input to the set of parallel problems. The output of each problem is also same as the final output of parallel problems. The energy consumption of a computing problem which comprises of a number of parallel independent computation problems depends on the initial state of the original input and the final state of the final output of the problem. This property of the energy analysis is expressed by equation 7-11 for two parallel independent problems shown in Figure 7-16.

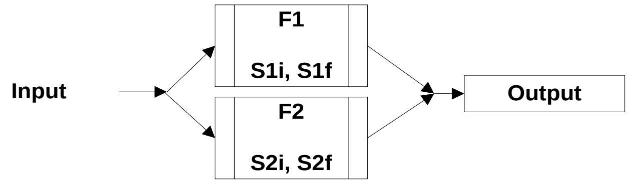


Figure 7-16 Two parallel independent computing problems with their associated initial state and final state entropies

$$\nabla Q = \nabla Q_1 = \nabla Q_2 \qquad \textbf{(7-11)}$$

In using EA method to measure the computational complexity of SAMs, the iterative problems may be described to be within the four mentioned categories of topologies pending the flow of data between inputs and outputs. This will be discussed later in this chapter.

7.3.3. Structural complexity

Among several proposed methods to measure the complexity associated with the structure of a computing solution (Aimin et al, 2008; Bansiya et al, 1999; Roca, 1996; McCabe, 1976; McCabe et al, 1989; Woodward et al, 1979), the approach taken by Roca (1996) is adopted here as it is based on the topological properties of the graph associated with the structure of the computing solution. In this method, first the direct graph model of the computing program is constructed and then it is transformed into the ‘reduced graph’ including three basic structures ‘sequence’, ‘branch’ and ‘loop’. The entropy of random uniform response function of the graph is then derived based on the entropy of random response function of the three basic structures and their combination rules.

Similarly, we transform each mapped SAM structure to a ‘reduced graph’ and then work out the entropy random response function based on the three basic structures within the graph. Here, we introduce the entropy of random response function of the three basic structures and their combination rules. For definition of the random response function and calculation of its entropy, reader is recommended to refer to (Roca, 1996).

7.3.3.1. Sequence structure

Evaluation of entropy H_s (bits), for a sequence of computing paths is calculated as (Roca, 1996):

$$H_s = -\int_0^1 \ln(N \times P^{N-1}) \cdot dp \quad \textbf{(7-12)}$$

where P is the probability of successful execution of a path, and N is the number of paths with similar probability of P as shown in Figure 7-17.

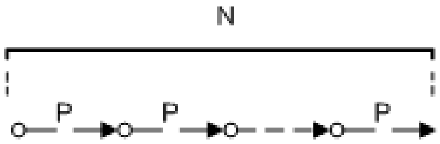


Figure 7-17 Sequence of N similar paths of computing

7.3.3.2. Branch structure

Evaluation of entropy $\mathrm{H_b}$(bits) for a branched computing path is calculated as follows (Roca, 1996):

$$\mathrm{H_b} = -\int_0^1 \ln[(N+1) \times B \times P^N] \cdot dp \quad \textbf{(7-13)}$$

where B is the frequency of execution of the branch path and N is the number of paths with similar probability of P in the branch as shown in Figure 7-18.

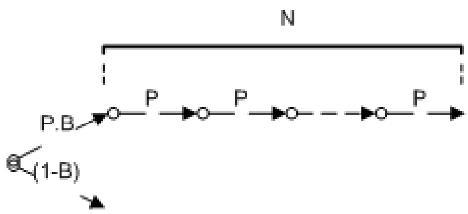


Figure 7-18 A branch of computing with N similar paths

7.3.3.3. Loop structure

Evaluation of entropy $\mathrm{H_l}$ (bits) for a loop of computing path is calculated as (Roca, 1996):

$$\mathrm{H_l} = -\int_0^1 \ln[\frac{M \times P^{M-1} \times [L \times (1-P^{N+1}) + 1] + (N+1) \times L \times P^{N+M}}{L \times (1-P^{N+1}) + 1}] \cdot dp \quad \textbf{(7-14)}$$

where L is the number of loop executions, N is the number of paths with similar probability of P in the true route of the loop, and M is the number of paths with similar probability of P in the false route of the loop as shown in Figure 7-19.

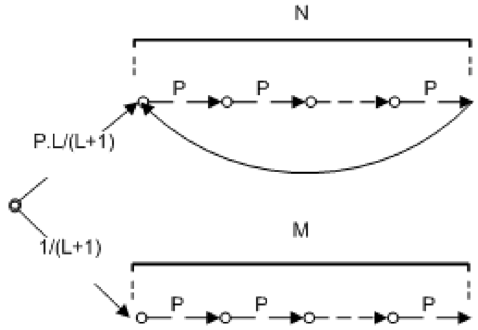


Figure 7-19 A loop of computing with N similar paths

7.4. Complexity analysis of sensitivity analysis methods

This section focuses on the analysis of the five categories of sensitivity analysis methods based on the proposed framework of the Evaluation Triangle (Figure 7-9). On each item of the evaluation criteria, all categories will be discussed and ranked against each other.

Throughout the evaluation process it is assumed that the number of inputs and outputs of the system under analysis are T and E respectively.

When applying each complexity metric, we assume that the attributes of the building blocks for each mapped SAM are identical whenever the differentiation needs knowledge about details of the actual algorithm. For example, the consumed energy and the delay time of all blocks are assumed to be equal.

Iterative tasks in the mapped SAMs are considered sequential computing problems unless it is possible to assume that the iterations can be executed in parallel.

7.4.1. Energy-Time metric of SAMs

Each SAM is considered to be a computing problem subject to the complexity metric measurement defined in equation 7-2. For further simplification, the building blocks for each computing problem are assigned an identical energy consumption value of (E) and an identical delay time of (t).

Figure 7-20 shows the composition and attributes of the sub-problems for each SAM. Iterative sub-problems are considered in sequential connection. In

Figure **7-20**-c, each pair of the sub-problem in N series represents the pair of Data Synthesis and Iterative Processing tasks within the mapped SA structure shown in Figure 7-8-c. Equations (15-19) show the calculation of the Energy-Time complexity of the SAMs.

a) Sub-problems of Sensitivity Testing SA method for Energy-Time metric

b) Sub-problems of Analytical SA method for Energy-Time metric

c) Sub-problems of Sampling based SA method for Energy-Time metric

d) Sub-problems of Computer Algebra SA method for Energy-Time metric

e) Sub-problems of EvenTracker SA method for Energy-Time metric

Figure 7-20 Sub-problems of each SAM for Energy-Time metric

$$\Theta_{ST} = \frac{9}{2} \times (N+1)^3 \times E \times t^2 \quad \textbf{(7-15)}$$

$$\Theta_{A} = E \times t^2 \quad \textbf{(7-16)}$$

$$\Theta_{S} = 36 \times (N+1)^3 \times E \times t^2 \quad \textbf{(7-17)}$$

$$\Theta_{CA} = \frac{9}{2} \times (N+1)^3 \times E \times t^2 \quad \textbf{(7-18)}$$

$$\Theta_{ET} = E \times t^2 \quad \textbf{(7-19)}$$

where $\Theta_{ST}, \Theta_{A}, \Theta_{S}, \Theta_{CA}, \Theta_{ET}$ are complexity measurements for the five SAMs i.e. Sensitivity Testing, Analytical, Sampling-based, Computer Algebra, and EvenTracker SAMs, respectively.

7.4.2. Entropy change of SAMs

For a sound application of entropy change measurement on each SAM, the initial set up of the sensitivity analysis problems must be determined. This initialisation would be based on flow of data between the set of inputs and outputs for each problem.

7.4.2.1. Entropy change in Sensitivity Testing SAM

The type of iterative processing in the first block (Figure 7-8-a) of Sensitivity Testing SAM is considered to be a sequential independent computing problem. This is due to the fact that regardless of details of the algorithm in Sensitivity Testing SAM, a function generates a series of values of outputs using a series of different values of inputs. In this process, each output generation is independent of other generated outputs.

To explain the relationship between the iterative processing block and data post-processing block, one could mention that all the output data from the iterative block together with the input data block are used by the post-processing block. Hence, an independent sequential type of computing problem is constructed.

Therefore, the overall entropy change of Sensitivity Testing SAM could be calculated as (7-20).

$$\square Q_{ST} = (N + 1) \times \nabla Q_s \qquad \textbf{(7-20)}$$

7.4.2.2. Entropy change in Analytical SAM

This change of entropy depends on the type and number of input series to the data pre-processing block as well as the number of output sensitivity indices. An overall entropy change could be calculated for this method using equation 7-21.

$$\square Q_A = \nabla Q_s \qquad \textbf{(7-21)}$$

7.4.2.3. Entropy change in Sampling based SAM

The type of relationship between the iterative processing block and the data synthesis block of sampling-based SAM, (Figure 7-8-c), is considered as a sequential dependent computing problem, because the synthesised data is immediately used for generation of output of iterative processing block.

With regard to the overall iteration of the two blocks, the relationship is sequential independent. Therefore, the overall entropy change of the sampling-based SAM could be calculated as 7-22.

$$\nabla Q_{\mathrm{S}} = (N+2) \times \nabla Q_s \qquad \textbf{(7-22)}$$

7.4.2.4. Entropy change in Computer Algebra SAM

Similar to the Sensitivity Testing SAM, the relationship between the first block of the Computer Algebra method, i.e. data pre-processing, and the iterative processing block is independent in terms of states of inputs and outputs. This also applies to the iterative tasks in the iterative processing block.

Therefore, the overall entropy change of the Sampling based SAM could be calculated as 7-23.

$$\square Q_{CA} = (N+1) \times \nabla Q_s \qquad \textbf{(7-23)}$$

7.4.2.5. Entropy change in EvenTracker SAM

The change of entropy in data pre-processing block depends on the type and number of input series and number of output sensitivity indices. An overall entropy change could be calculated for this method by equation 7-24.

$$\nabla Q_{\mathrm{ET}} = \nabla Q_s \qquad \textbf{(7-24)}$$

7.4.2.6. Structural complexity of SAMs

Each SAM common skeleton is transformed into its minimal path. For simplicity purposes, the probability of successfully executing each path is considered to be equal to P for all paths.

7.4.2.7. Structural complexity of Sensitivity Testing SAM

The composition of the reduced graph corresponding to the common skeleton in Figure 7-8-a is a loop structure with the loop attributes $\mathrm{M} = 1, \mathrm{N} = 1$ shown in Figure 7-21.

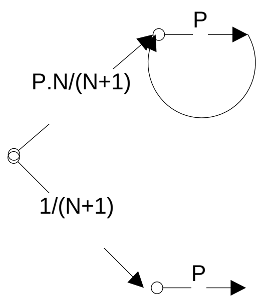


Figure 7-21 Reduced graph equivalent for Sensitivity Testing SAM

The structural complexity of Sensitivity Testing SAM could therefore be calculated by equation 7-25.

$$\mathrm{H_{ST}} = -\int_0^1 \ln[\frac{N(1+P^2)+1}{N(1-P^2)+1}] \cdot dp \qquad \textbf{(7-25)}$$

where N is the number of inputs to the Sensitivity Testing SAM.

7.4.2.8. Structural complexity of Analytical SAM

The reduced graph of the common skeleton of Analytical SAM comprises of one path, and therefore, a sequence of one computing path ($\mathrm{N} = 1$) is considered. The structural complexity is therefore equivalent to -1.

$$\mathrm{H_A} = -1 \qquad \textbf{(7-26)}$$

7.4.2.9. Structural complexity of Sampling based SAM

For this purpose a sequence of two structures are included as the reduced graph of the Sampling based SAM. As shown in Figure 7-22, the first structure is a one path sequence, and the second structure is a loop with attributes: $\mathrm{M} = 1, \mathrm{N} = 2$.

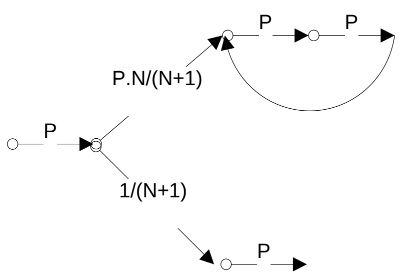


Figure 7-22 Reduced graph equivalent for Sampling based SAM

Therefore, the structural complexity of the Sampling based SAM could be calculated by equation 7-27.

$$\mathrm{H_S} = -\int_0^1 \ln[P + 1 + \frac{3NP^3}{N(1-P^3)+1}] \cdot dp \qquad \textbf{(7-27)}$$

7.4.2.10. Structural complexity of Computer Algebra SAM

Similar to the Sampling based SAM, a sequence of two structures completes the reduced graph of the Computer Algebra SAM. Like shown in Figure 7-23, the first structure is a one path sequence, and the second structure is a loop with attributes: $\mathrm{M} = 0, \mathrm{N} = 1$.

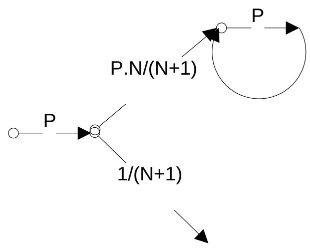


Figure 7-23 Reduced graph equivalent for Computer Algebra SAM

Therefore, the overall structural complexity of the Computer Algebra SAM could be calculated using equation 7-28.

$$\mathrm{H_{CA}} = -\int_0^1 \ln[P + \frac{2NP}{N(1-P^2)+1}] \cdot dp \qquad \textbf{(7-28)}$$

where N is the number of inputs to the Sensitivity Testing SAM.

7.4.2.11. Structural complexity of EvenTracker SAM

The reduced graph of the common skeleton of EvenTracker SAM comprises of one path sequence ($\mathrm{N} = 1$), resulting the structural complexity equivalent to -1.

$$\mathrm{H_{ET}} = -1 \qquad \textbf{(7-29)}$$

7.5. Overall evaluation of computation complexity of sensitivity analysis methods

The ranked score of evaluated computational complexity of different sensitivity analysis methods are reported in Table 7-1. The higher the evaluated complexity of a SAM in one criterion, the lower its rank and the lower the score value would be. Each of the three evaluation criteria is separately scored and an overall score is associated with each method. Some scores were equal among SAMs, for example between EvenTracker and Analytical, in all three criteria. The final score is the sum of the three scores for each SAM. Table 7-1 summarises the individual and the final scores.

Table 7-1 Ranked scores of sensitivity analysis methods

	Θ	∇Q	H	Total
Sensitivity Testing	3=	3=	3	9
Analytical	1=	1=	1=	3
Sampling-based	5	5	5	15
Computer Algebra	3=	3=	4	10
EvenTracker	1=	1=	1=	3

A correlation could be observed between the ranks of each SAM on the three criteria. One can see that the higher levels of complexity of SAMs the more complex is the implementation and data exchange, therefore, contributing to higher utilisation of computation resources.

From the final scores it could be concluded that EvenTracker and then analytical methods feature the least computational complexities. Moreover, sampling-based SAM, show the highest computational complexity. Analytical SAM, working based on the parameters of model equations, lacks the advantage of data processing with real-time data as is the case for EvenTracker SAM.

7.6. Summary of complexity analysis of SA methods

This chapter evaluated computational complexity of sensitivity analysis methods (SAMs) regardless of their initial assumptions and applications. A three-tier evaluation methodology was discussed and adopted to evaluate computational complexity of SAMs in terms of their Energy-Time metric, change of entropy, and structural complexity.

The results of the evaluation showed that the proposed EvenTracker sensitivity analysis method has the lowest computational complexities compared to other popular sensitivity analysis methods which could work with real-time data. Using the proposed evaluation platform we also concluded that sampling-based sensitivity analysis methods have the highest computational complexity.

In the upcoming chapters of this thesis we will experimentally demonstrate how the accumulation of computational efforts per dimension of the system inputs is avoided by implementation of different SAMs.

It is worth to add that although the evaluation strategy in this work was focused on the context of SAM, the proposed evaluation strategy may be feasible to evaluate other computational methods for comparing their strengths and weaknesses.

8. Implementation

In this chapter the design and development of a data integration system prototype to help with providing the concept of this study is discussed. The suit of hardware and software which was designed and developed is meant to demonstrate the versatility and flexibility in the input data definition as well as real-time data acquisition and simulation. Therefore, it features the major challenging issues of this study, which were added to the system in few consecutive development and implementation stages. A simplified section of a chocolate manufacturing line is introduced in this chapter.

8.1. Main features of the implemented system

The main features which have been considered in the functionality of the implemented system are outlined and explained. Functionality of these features are later discussed with evidence and introduced within the architecture of the system.

1. <u>Demonstration of flexibility in the definition of inputs to the DAQ system</u>; As a major challenge in this work, different types of interfacing with sensors and their data types were adopted at input definition layer. An automated mechanism has been designed and developed which dynamically prepares the structure of input points at the root of DAQ system based on the attributes defined by the user. With advances in sensor data communication media and technology this feature requires to expand and update with the new types of connections and communication protocols.
2. <u>Provision of sensory data in real-time</u>; Overcoming challenges of real-time data acquisition is among important subjects of this work. Connection to and data retrieval from historical data is not a recommended practice in this research. Therefore, once definition of data connections to sensor network is completed functionality is provided to serve frequent and quick access to the sensor signals.
3. <u>Demonstration of real-time connectivity between the data acquisition system and simulation model</u>; Real-time simulation requires real-time

connectivity and a representation of the process using descriptive model. Therefore, Real-time Model Matching Mechanism (R3M) has been designed and developed to facilitate the communication between data acquisition system and the simulation model of the process.

4. Demonstration of fast forward run of the model as well as real-time; Fast-forward run has been designed and developed in order to allow the user run the simulation models with respect to the most recent data.

8.2. Architecture of the implemented system

The building blocks of the integrated hardware / software solution are defined in a layered structure according to the four layers introduced in section 5.2. Acquired data from sensor network transferred between the layers. Each layer received the data from its lower layer and prepared it in the proper structure for the next layer. A distributed and queue-based real-time database structure takes care of the data storage and data communication between the layers. This prevents the layers to wait for each other when one is not prepared to receive data. In hardware-oriented words, like a pipelined arrangement of the tasks (National Instruments Corp., 2009), each processing core of the CPU could continuously handle an assigned task without being interrupted for communication with the other processing core (i.e. multi-threading).

The four layers of the architecture are shown on the colour-shaded diagram in Figure 8.1. The nature of the main components of the implemented system, the itinerary of the data and the overall functionality of the layers are described at the following paragraphs.

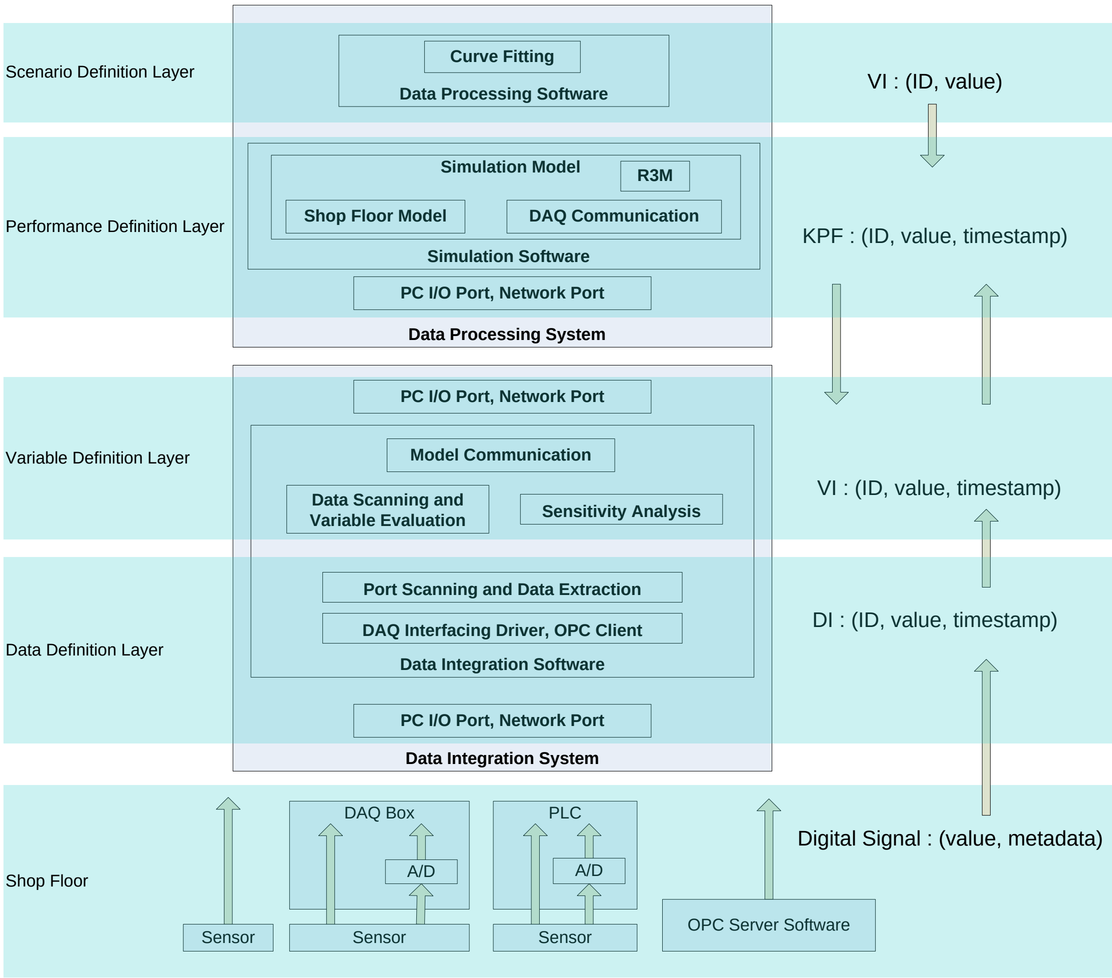


Figure 8-1 Implemented system architecture

8.2.1. Implementation of the data definition layer

In the Data Definition Layer (DDL), the main concern is the continuous access to the input ports of the data acquisition hardware. This necessitated the operation of software which could drive data acquisition hardware and which could be programmed for scanning specifications. Therefore, the hardware-aware, or driver-equipped, software uses some user-defined specifications to extract the data from each data connection. Scanned data is then accompanied with timestamp and a unique identification tag associated to the scanned connection. The triplet Data Input, or DI, is then located in the specific memory locations (addressable by a unique DI name) for access by the Variable Definition Layer (VDL). Therefore, per each sensor output, as shown in Figure 8-2, one corresponding memory space for one DI is allocated. This space is overwritten on each scan.

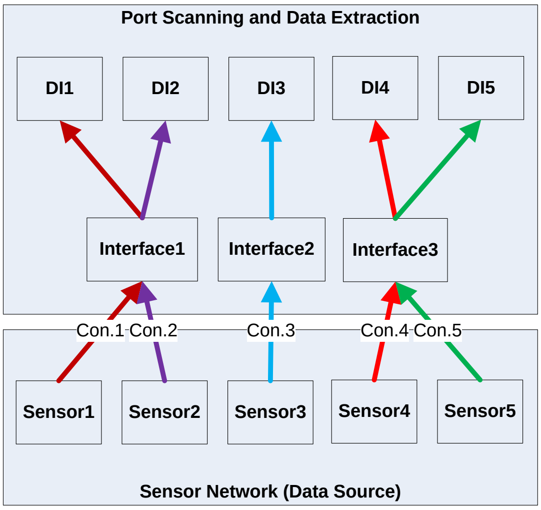


Figure 8-2 The main task in data definition layer is interfacing and sampling data sources, (different colours show different instances of the same task component)

8.2.2. Implementation of variable definition layer

Three components are implemented in the VDL to accomplish its tasks. The first component is a software program which regularly scanns and combines the values of DIs based on the user-defined formula. The resulting Variable Input or VI was wrapped up together with a timestamp and a tag unique to the type of variable. This triplet piece of data is then buffered for collection by the next layer (PDL). This buffer has a queue structure, as shown in Figure 8-3, so that older evaluated values are not overwritten by the new ones. The number of instances of Data Scanning and Variable Evaluation software component that execute at the same time depends on the number of VIs.

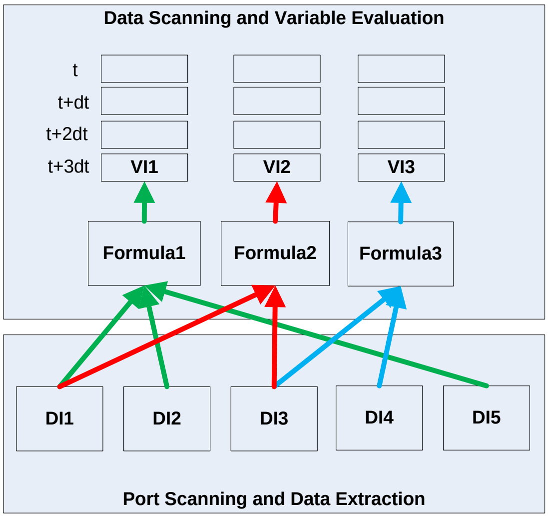


Figure 8-3 One task in variable definition layer is converting sampled data to variable information, (different colours show different instances of the same task component)

The second implemented building block in VDL is a single programmed firmware which handles the data communication between VDL and PDL for the functionality requirements in VDL. The Model Communication task collects the buffered values of VIs and sends them to the established data connection between the two layers, as shown in Figure 8-4.

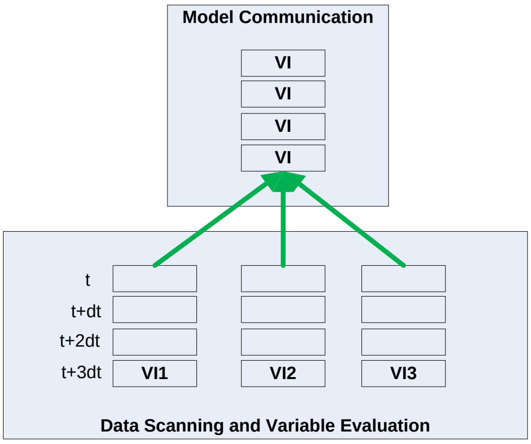


Figure 8-4 The second task in variable definition layer is queueing up all variable information to send to the model

The third task is to measure the sensitivity index of the generated Key Performance Factor (KPF) values in PDL with respect to the values of VIs. EvenTracker algorithm was programmed to collect KPF values and VI values from their corresponding buffers, and produce a series of sensitivity index values for each pair of {KPF,VI}.

8.2.3. Implementation of performance definition layer

Implementation of PDL involves software simulation model design and hardware settings for data communication between VDL and PDL. This communication link is the same as the link uses by the Model Communication in the VDL. DAQ Communication firmware block used a messaging service for both receiving VI values from VDL and sending KPF values to VDL (KPF values to be used by EvenTracker algorithm). R3M software mechanism is developed as a part of the simulation model. Figure 8-5 shows two task components in PDL.

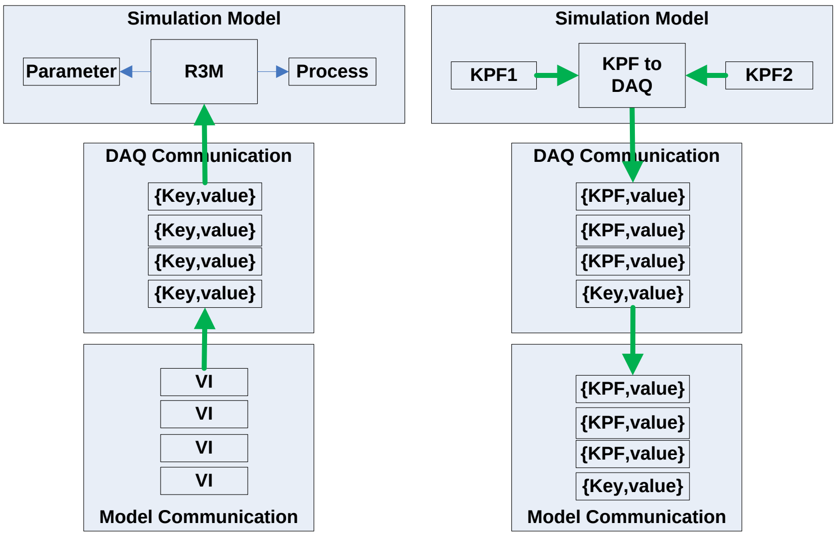


Figure 8-5 In performance definition layer, one task receives variable information and diseminates them in model (left-hand-side), and the other gathers model performance values (e.g. KPF1, KPF2) and sends them to the DAQ system (right-hand-side)

One component task of PDL receives DAQ messages and interprets them as pairs of {Key, Value}. The Key is used by R3M to associate the Value to the unique process, or parameter of the model. As shown in Figure 8-5, simulation model holds various processes and parameters depending on the system being modelled.

The outcome of the Simulation Model is a series of performance factors and possibly charts which are defined at simulation modelling design time. Another task component was designed in the model to take the values of performance factors regularly and format them in pairs of {KPF, Value} and send the pairs to the Model Communication in VDL where they are processed by EvenTracker task component.

8.2.4. Implementation of scenario definition layer

In the SDL two tasks take care of preparation of data and use of data for fast-forward execution. While in real-time execution values of Variable Inputs VIs flow from VDL to the PDL, each VI data series is also taken by a curve fitting task that estimates the best statistical data distribution and its associated attributes. The generated attributes are stored in memory blocks and updated every often. Figure 8-6 shows three instances of distribution estimation task in charge of curve fitting for three VI data series. Multiple instances of curve fitting task component could run in parallel at the same time, one for each VI.

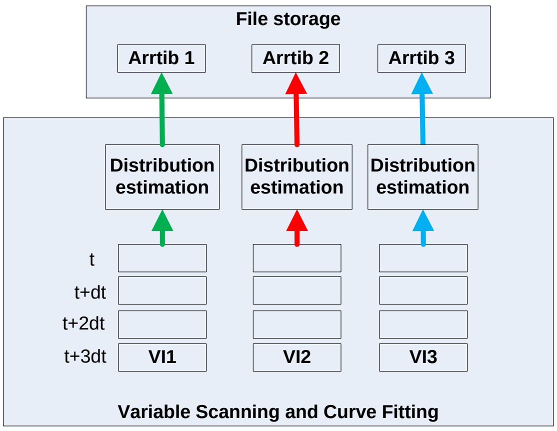


Figure 8-6 One task in scenario definition layer is curve fitting from input variables

Another task in SDL has facilities for user to execute fast-forward simulation using the stored attributes of VIs series. Each attribute is used on the same process or parameter that the corresponding VI was connected through R3M. This task has a single execution and is implemented in simulation model (Figure 8-7).

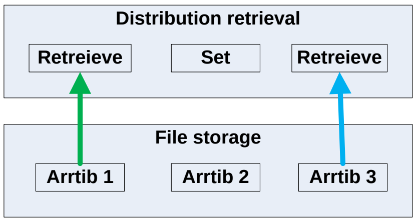


Figure 8-7 In fast-forward mode of simulation, stored attributes of variable distributions are retreieved and used in the model

8.3. Criteria for selection of development tools in the implementation

The system to be implemented will consist of a significant amount of hardware and software in terms of both strengths and number of components (e.g. sensors) so that the efficiency of the functionalities could be compared with existing ones. The major criteria based on which the components of the system are defined and implemented are explained at the following. Further, the components of the system and their implementation are introduced and described.

8.3.1. Distributed versus centralised

Single or multi-core PC based computers with Windows XP operating systems and connections to programmable logic controllers (PLC) and data acquisition cards are among the usual computer systems which are used for implementation of data integration systems (Tavakoli, 2007; Tavakoli, 2008a; Tavakoli, 2008b; Tavakoli, 2009). With this processing power available in most cases, depending on the size of the data acquisition and system and simulation model, it may be too much to implement both data acquisition and simulation on one PC system. It is preferrable to implement a multitier system on these occasions, so that data acquisition, management and presentation can run on separate machines. This guarantees more accurate measurements on the computational overhead of each sensor data acquisition (since all running threads and I/O interfaces belong to the data acquisition and not simulation), when at the same time it would be a more feasible example of industrial data acquisition and monitoring system.

8.3.2. Communication media

This applies to both data communication between sensors and the DAQ system, and communication between the DAQ system machine and simulation machine. Having to split the machines for different roles, a communication media with sufficient data transfer rate between machines uses Ethernet network connection. Sensor data communication, on the contrary should demonstrate variety and flexibility. Different possible types which may include external processor based systems as well are thought of. Connection to sensors via PLC and OPC server can provide a flexible and comprehensive support.

8.3.3. Software programming language

At least three programming environments may be considered as major candidates for development of such systems. SUN's Java language and its associated open source library codes (Oracle Corp., 2010), Microsoft .NET programming environment (Microsoft, 2009), and National Instruments LabVIEW software (National Instruments Corp., 2010). Pros and cons in using each environment which are outlined below, justified use of NI LabVIEW.

8.3.3.1. SUN Java

Open source has always two ends. On one end there is free access to a pool of library codes capable of much functionality according to their authors. On the other end, open source leaves no guarantee for supporting documents especially on new versions and extensions. This could endanger the timeline of project development.

8.3.3.2. Microsoft .NET

Although the development environment plus some more special library codes are available at a charge (in non-express versions), this one-off charge may be worth access to the available further technical support. Similar to Java, most of the data acquisition and user interface codes as well as established mathematic algorithms must be written by programmer. In addition, management of computer resource such as multi-core and port access in run-time takes extra effort, which could surpass the scope of this work.

8.3.3.3. NI LabVIEW

Programmer in this data-driven environment enjoys a visual and modular programming tool which was also claimed to take care of resource allocation when more than one thread of code needs processing or input/output access. The data acquisition and data management orientation of the environment facilitates the initial effort to setup connections to different communication media. At the same time almost all modules can be customised and programmed to be reconfigured from a user interface.

Provided all above programming facilities, drawbacks may appear on the running speed when pieces of thread codes are numerous and memory hungry. However, by creating efficient multithreaded code and parallel programming strategies, like explicit threading and assigning code to processing core (National Instruments Corp., 2010), taking advantage of each processor and consequently deterministic real-time for critical I/O could become possible in LabVIEW environment. Although some technical references including (Moore, 2008) claim that adding cores slow data-intensive application, they add that this disadvantage rises after adding about eight cores. On the other hand, today's high performance personal computers utilise four cores, and single core machines are fading out, therefore this disadvantage does not impact use of LabVIEW which works well with multi-cores.

8.3.4. Modelling and simulation software

There are several packages which can satisfy process level modelling and simulation including WITNESS (Lanner, 2010), Flexsim (Flexsim Software Products Inc., 2010), AutoMOD (APPLIED MATERIALS, 2010), Arena (Rockwell Automation Inc., 2010), SIMUL8 (SIMUL8 Corp., 2010) to name a few. Rockwell Software's Arena software (Rockwell Automation Inc., 2010) is a good candidate, available, capable enough in terms of modularity and versatility of modules as well as facilities for communication with other software applications via an Ethernet connection.

8.4. Components of the implemented system

The implemented system is a four-tier system with three tiers as PC machines and one tier as a combination of sensors and data acquisition devices (e.g. PLC). Figure 8-8 shows these components and their relationships.

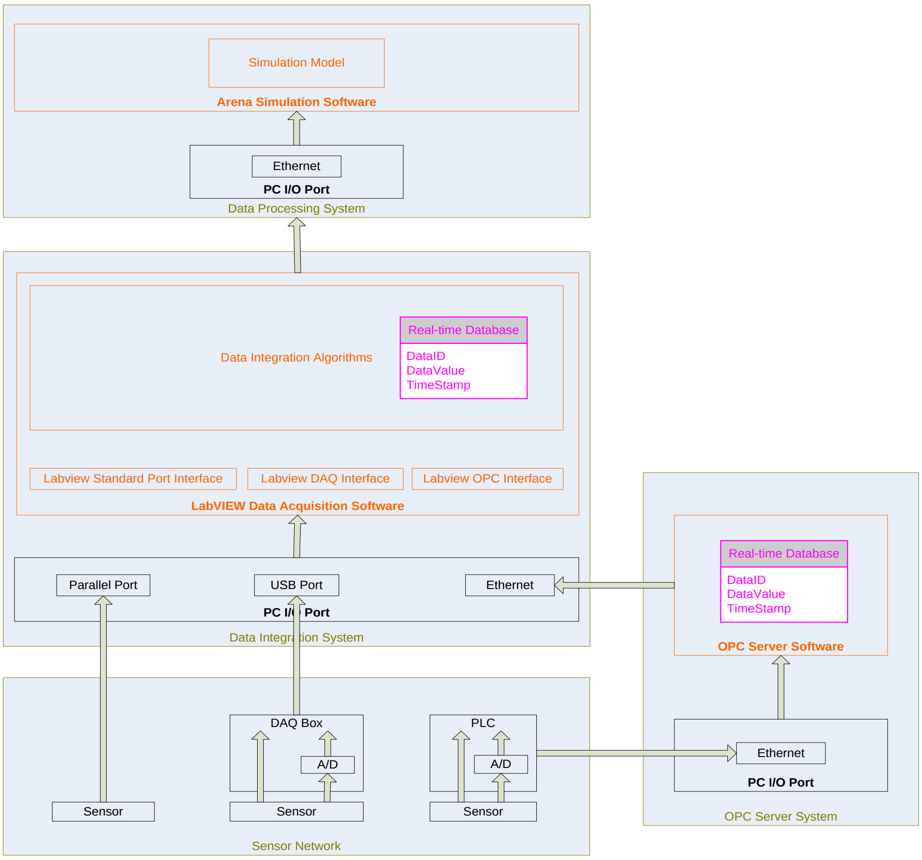


Figure 8-8 Implementation system components and data flow

A lab prototype demonstrator was created at the Systems Engineering Research Laboratory to demonstrate the proposed solution. The interrelationship between the four tiers of the system is explained in the following sections.

8.4.1. Sensor network

A series of sensors were mounted around a conveyor belt system (Figure 8-9) used to carry small objects. The sensors generate raw input data from the processes. The sensors deployed in the demonstrator are listed in **Error! Reference source not found.** in Appendix A and described here;

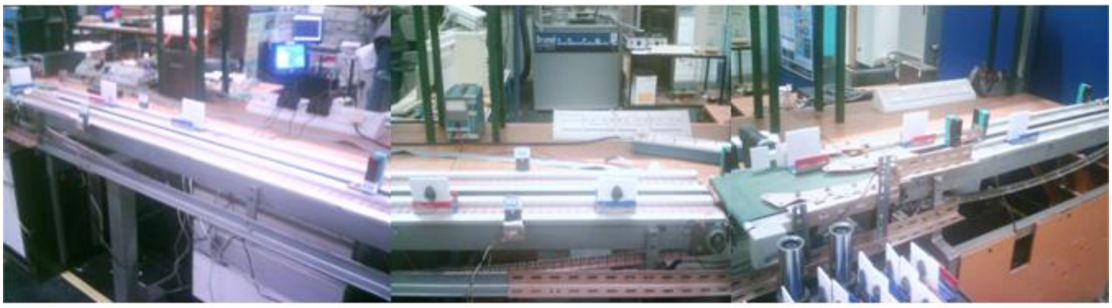


Figure 8-9 Panoramic view of the conveyor belt system setup for implemented system

Four proximity RFID readers were installed using four USB serial ports. Each RFID reader sends a multiple-byte data which is the identification serial number (ID) of the passive tag held near the reader (Figure 8-10).

Figure 8-10 Proximity RFID reader and its associated passive tag

One range RFID reader was connected to one USB serial port. The range RFID reader sends a multiple-byte data indicating the ID of the active tag held around the reader (Figure 8-11).

Figure 8-11 Range RFID reader and its associated active tag

One data acquisition box with fourteen sensors was connected to another USB port (Figure 8-12). This data acquisition box provides data from fourteen sensors deployed in the demonstrator. The sensors range from microphone, thermocouple temperature sensor, luminance sensor, and five manual potentiometers - and six manual on/off switches. Data from the eight analogue sensors were available in the range of 0-100 integer values, and data from the six switches were available as single-bit values 0 or 1.

Figure 8-12 National Instruments' Data acquisition demo Box

Eight light switch sensors were connected one-by-one to the eight bits of one parallel port (Figure 8-13). Each light switch sends a one-bit data (0 or 1) indicating cross passing an object in front of it.

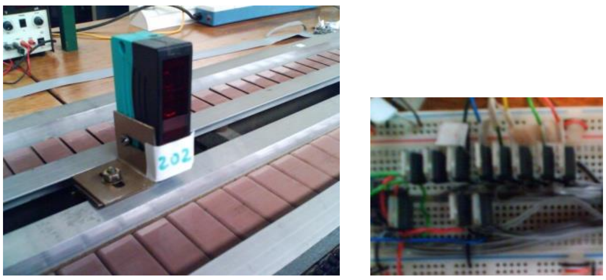


Figure 8-13 Light switch sensor and its associated voltage converter circuit board

Five sensors were connected one-by-one to five OPC real-time database variables through five PLC connections. The five sensors included one infra-red temperature sensor, one load sensor, and three light switches (Figure 8-14, Figure 8-15).

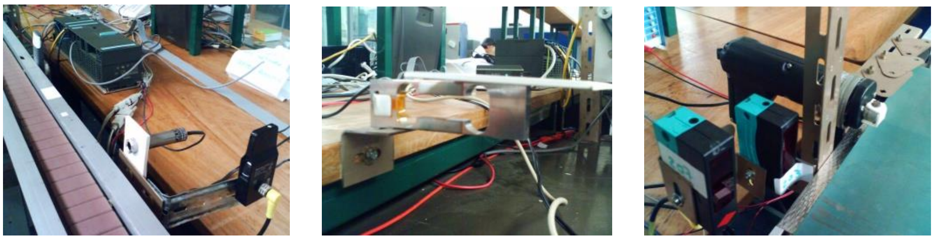


Figure 8-14 Five connected sensors to the PLC

Figure 8-15 Siemens S7-300 PLC device

The PLC device was used to handle simple control tasks of the defined manufacturing scenario as well as collection and distribution of the connected sensor data.

8.4.2. OPC server system

One PC is responsible for providing a real-time access to sensor outputs for DAQ system from those sensors which are connected via PLC. This PC acts as an OPC Server and since it could serve multiple device connections at the same time, it was considered as a separate tier. PLC provides sensor output values available on the Ethernet LAN connection which is then taken by OPC Server software from Ethernet port and arranged as an OPC item. These item values are again left available by OPC Server software on the LAN connection for any OPC Client software which is connected to the same LAN network.

8.4.3. The data acquisition (DAQ) system

One PC computer was assigned to handle the task of real-time data acquisition and management. Data acquisition software system prepared to read and organise raw sensor data available from the variety of PC I/O connections. A number of interface connections were made available and used in the implementation. They included one parallel port, two serial ports, up to eight USB ports, and Ethernet port. Sampling rate for each sensor was defined by DAQ system. A database sub-system stored sampled data in a time-based queue structure (i.e. First-In-First-Out) prepared for immediate transfer to the simulation system. Data acquisition system featured multi-threaded acquisition software with each thread responsible for reading from a different I/O connection. Multi-core characteristics of a PC machine for DAQ system together with using LabVIEW software for acquisition development helped with access speed and memory management.

8.4.4. The simulation system

A PC machine handles the process of performance modelling and data apparatus for the process flow. Based on a defined manufacturing scenario the modelled processes were designed and developed in Arena software and wherever real-time data was required to trigger the process, R3M mechanism is implemented. Of course, R3M mechanism took care of the data communication between DAQ system and simulation system on both ways (i.e. sending real process variable values from the DAQ system to the simulation system, and sending model-oriented performance values from the simulation system to the DAQ system). Scenario definition interface and fast-forward mechanism were also implemented on the developed model.

Figure 8-16 shows the three machines which OPC Server system, DAQ system, and simulation system run on.

Figure 8-16 view of three PC machines for - from left to right - OPC server system, simulation system, and DAQ system

The following use case diagrams in Figure 8-17 and Figure 8-18 demonstrate the interaction of the user with the three components as well as with the PLC device at both design time and run time. Details of the design and development tasks and functionalities of the components will be described at the following step by step.

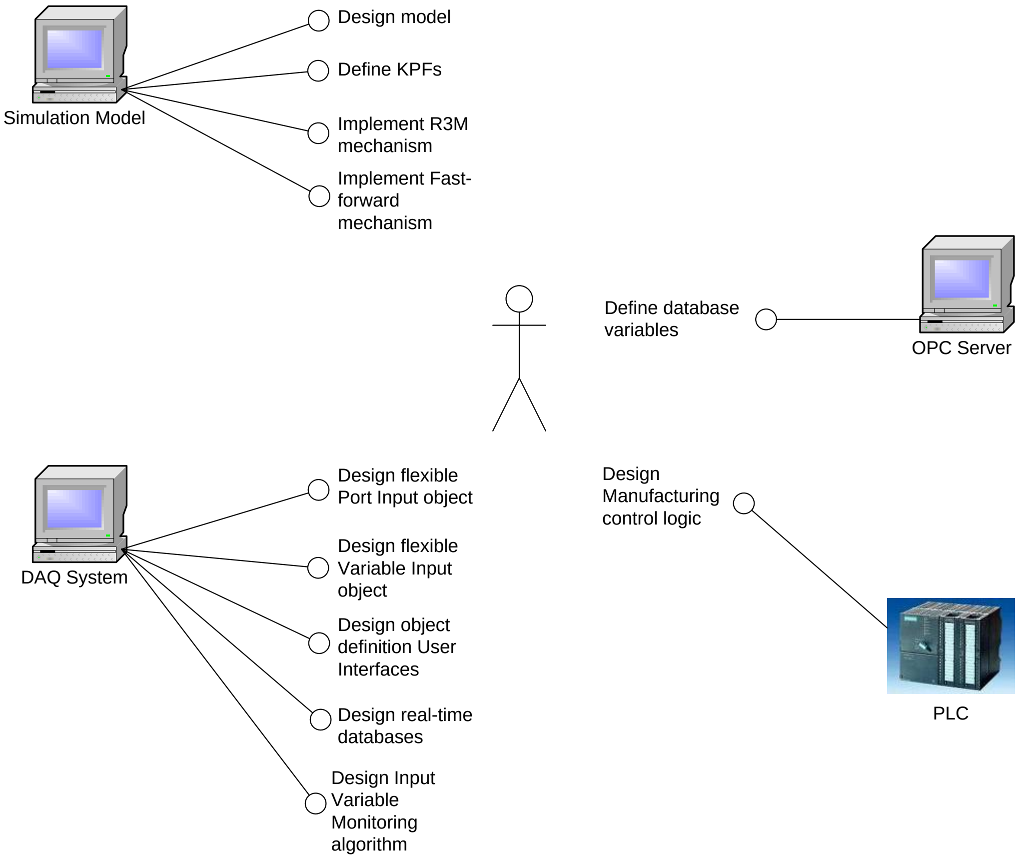


Figure 8-17 Implemented system use case – design time

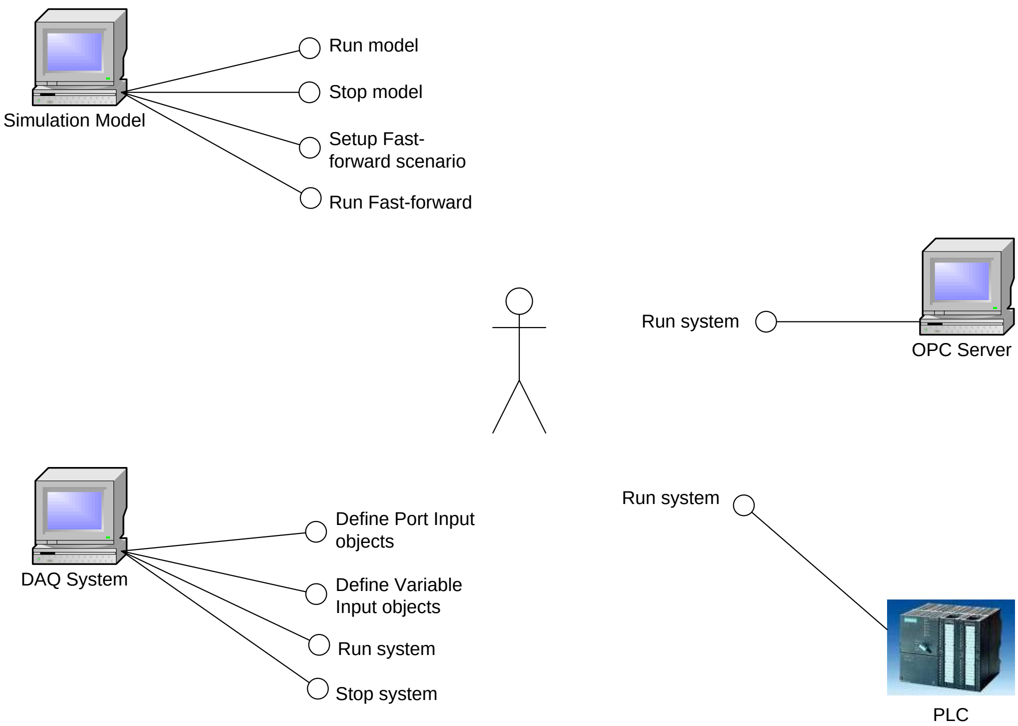


Figure 8-18 Implemented system use case - run time

8.5. Implementation steps

Implementation of the firmware system has been performed feature by feature on a priority basis and according to the milestone diagram of Figure B-1in Appendix B. As can be seen in this diagram, the implementation started from design and development of a real-time single sensor data acquisition and real-time simulation system. By growing the system up to involving more sensors for a more comprehensive shop floor scenario the type of simulation software has remained the same - as Rockwell Software's Arena, whereas the data acquisition system development environment has changed from Microsoft's C# .NET to National Instruments' LabVIEW software.

Prior to the design and development of the platform for multi-sensor data acquisition, a simulation model of the shop floor operations was designed and developed. Generated data from this virtual shop floor was then prepared for connection to the second simulation model that represents shop floor and receives real-time data from data sources. This virtual laboratory-based step provided confidence in the operation of the designed mechanism for real-time connection between the physical sensor data acquisition and the simulation model (Tavakoli et al, 2008).

Adding Fast-forward model simulation in the next stage has also brought up curve fitting algorithms in the data acquisition system software. Ultimately, sensitivity analysis has been developed and added to the data integration system software.

8.6. Implementation system design

After components of the pilot system were identified and prepared, three major developments were sought for design. At hardware stage in the sensor network, adaptation of the voltage level either from analogue to digital, or on simpler cases, from high DC levels to lower levels were among the essential signal conditioning tasks. Major parts of the work lied on the software stage with PLC programming, DAQ software development with LabVIEW and simulation modelling with Arena. Design issues and development ideas are introduced and explained at the following. Before all, a set of imaginary processes were defined as the basis for the simulation model as well as the definition of the locations and roles of the sensors. The defined manufacturing scenario is introduced below prior to the other design aspects.

8.6.1. Manufacturing scenario of implementation

A part of an imaginary chocolate manufacturing shop floor scenario was defined. The green blocks of the flow chart in Figure 8-19 show the processes of the virtual chocolate manufacturing scenario. The other symbols in Figure 8-19 show the location of the connected sensors. The defined processes in connection with the role of the associated sensors are described at the following. Production items move on the conveyor belts in order to travel between the processes.

- Arrivals; virtual raw materials entered a conveyor belt per portion of one chocolate bar and moved towards the first process. Each portion was located on a separate container with a unique RFID tag attached to it. A proximity RFID reader at the entrance of the conveyor belt read the tag of the passing container. The scanned tag was scanned by the DAQ system.
- Baking; containers of raw material entered the Baking process one after the other. At the entrance of the baking process, a light switch generated one pulse signal per passage of each container. At the exit of the process, a proximity RFID reader read the tag of the passing container.
- Inspection; at this station the temperature of the material of each container was read by the thermometer on the demo box. Additionally, a member of staff manually checked the size and the weight of each container. After this investigation the member of staff either returned the container on the conveyor belt or disposed it and switched off a switch on the demo box. A light switch generated a pulse signal per passage of each returned container.
- Covering; containers entered the covering process one by one passing first by a proximity RFID reader. A thermometer read the temperature of each passing container. At the exit of the Covering process, a light switch generated pulse signal per passing each container.

- Wrapping; containers entered the wrapping process one by one passing first by a light switch which generated a pulse signal per passage of each container. At the exit of the wrapping process, a light switch generated pulse signal per passing each container.

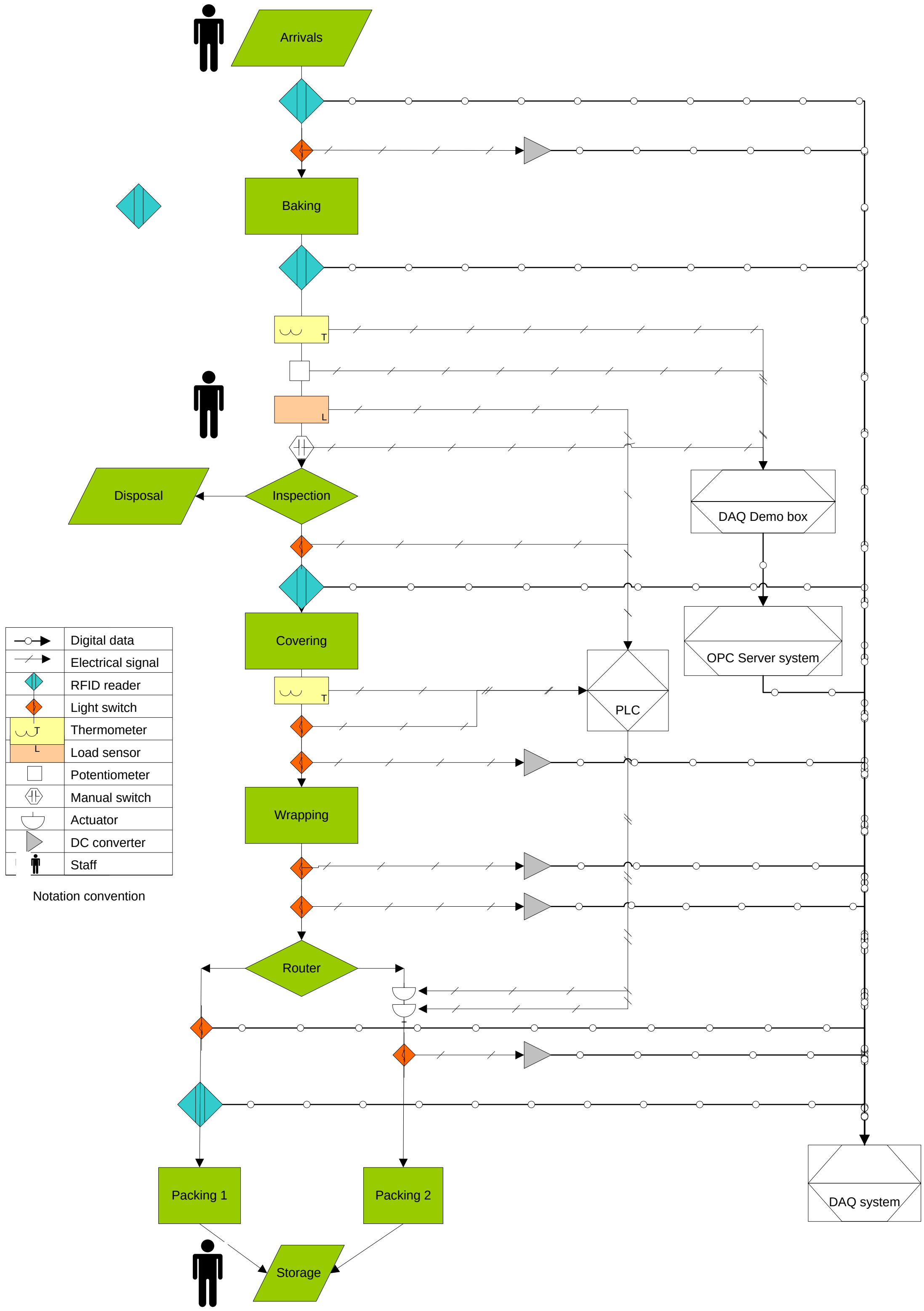


Figure 8-19 Hypothetical partial chocolate manufacturing shop floor production scenario and connected sensors and components

- Router; at this stage, based on the size of the container, they were directed to one of the two packing conveyor belts. Redirection was performed by means of an actuator and two light switches attached on a small conveyor belt. The two light switches which were connected to the PLC - as well as to the DAQ system - generated pulse signal and sent it to the PLC if the container blocked them. The actuator either received a signal from the PLC and pushed the container towards the route to the packing2 process, or alternatively, received no signal in which case, the container remained on the route to the packing1 process.
- Packing1; containers entered the packing1 process one by one. At the entrance a switch was trigged and consequently a pulse generated and sent to the DAQ system. At the exit of the process, a proximity RFID reader read the tag of the passing container.
- Packing2; containers entered the packing2 process one by one. At the exit, a switch was trigged and consequently a pulse generated and sent to the DAQ system.
- Storage; all containers after passing the Packing1 and Packing2 processes, enter the storage process.

As illustrated in Figure 8-20, sensor outputs are electrical signals in many occasions which cannot be directly connected to the Input / Output port of the DAQ system PC. Modification of the electrical attributes of some of the electrical signals is therefore necessary in order for the DAQ system PC to understand them as digital data. Upcoming paragraphs will explain this part of the design task.

8.6.2. Sensor signal conditioning

Three categories of signals were identified among sensor output signals and proper signal conversion is provided for them;

- Analogue signals; the analogue current flow on the output of the infra-red thermometer (sensor number 28 in Table A-1 in Appendix A) had an output fluctuation range of 4-20 mA (for temperature range of -20ºC

– 100ºC) which is high enough to be sensed by analogue current input port of the PLC. Therefore, it has been directly connected to the current input port of the PLC. The analogue voltage on the output of the load sensor (sensor number 29 in Table A-1 in Appendix A), however, had too low range of variations (100mV) to be sensed directly be voltage input of the PLC. It was therefore fed into the input of a differential amplifier circuit (Figure 8-20) which could leverage the variations to 5V. The output of the amplifier circuit was then connected to the analogue voltage input of the PLC. Other analogue sensors including thermocouple thermometer, microphone and luminance meter, and manual potentiometers (sensors number 6 - 13 in Table A-1 in Appendix A) were connected to the converter circuits in the Demo box. Their output fluctuations were therefore sensed and converted to digital signals by means of the Demo box circuits.

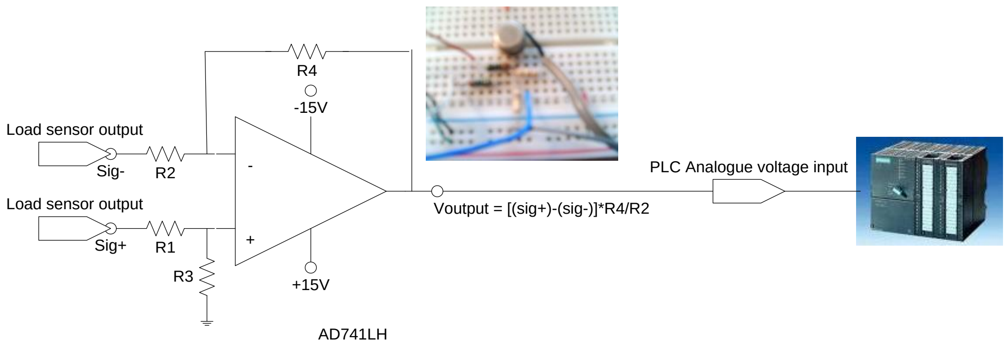


Figure 8-20 Differential Amplifier circuit for load sensor output amplification

- Two level high voltage DC signals; switch sensors which generate an output between 0V and their supplied DC voltage (usually a level between 10V–24V) required a conversion circuit at their output if they were connected to the parallel port of the computer directly. In cases where their outputs were connected to a PLC digital input port, there was no need to convert the level of DC voltage. The DC conversion circuit, as shown in Figure 8-21, used a voltage regulator IC which pulled down the 24V DC to 5V and did not change the 0V level.

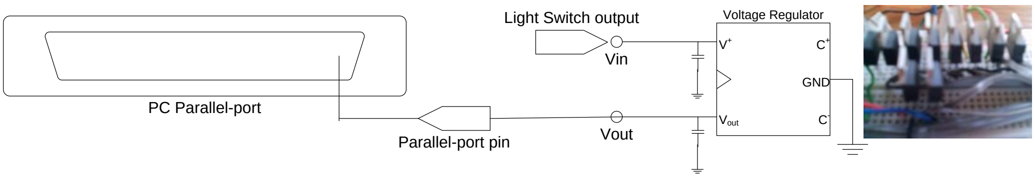


Figure 8-21 DC voltage converter circuit

- Digital signals; RFID tag readers (sensors number 1 - 5 in Table A-1 in Appendix A) enjoy their serialized RS232-based output stream of bytes on their serial output connection which could be directly connected to the serial port of the PC computer. Due to the lack of serial ports on today's computers, USB-to-RS232 converter devices were used in order to accommodate the five RFID readers on the I/O port interface of the DAQ system PC (Figure 8-22).

Figure 8-22 USB-to-RS232 converter cables connected to RFID readers

By implementing above design and development tasks, digital data was provided for being received by the DAQ system PC. However, design and development of packing route control software in the PLC as well as preparation of the actuator signal was also performed. These are explained here before starting the description of DAQ software design.

8.6.3. PLC software

Rather simple control software was designed, developed, and deployed into the processing unit of the PLC. This software was responsible for recognising if the passing container is of one of the two defined types. The two types defined based on the size of the container representing the size of the chocolate bar being produced. Recognition of the size was made possible by implementing two light switches in parallel and close enough to each other, as in Figure 8-23. Passing containers of large types caused both light switches to generate pulse signal simultaneously. Whereas passing small type containers caused the two light switches to generate pulse one after the other. Upon recognition of the large type, the software in the PLC generated three pulse signals with defined durations on three output pins. One signal was connected to the conveyor belt so that it stopped rolling. The second and the third signals helped with the generation of two pulse signals for the actuator with different polarities one after the other. The first generated signal caused the actuator to push the stopped container sideway. And the next signal caused the actuator to pull back before the conveyor belt starts again. This simple control scenario is demonstrated in flow chart below in Figure 8-23. The associated designed software code is available as attachment to this thesis.

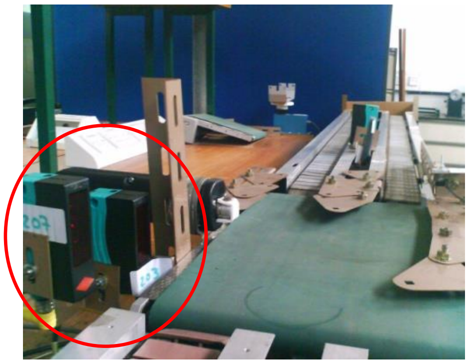


Figure 8-23 Two light switches in parallel and closely attached

As shown in the flow chart in Figure 8-24, the PLC software starts with combining the two input ports in a binary AND function and checking its output. Unless the two port pins are both at high logic level, the function outputs zero. When the two port pins are high, software runs towards the sequence of setting and resetting output pins as well as considering some defined delay in between them. This way each stage of the actuation could be completed and container located properly.

An electronic circuit was designed and built to convert the two PLC output pins to two signals of different polarity for the actuator. This circuit consisted of one electronic IC of type L293D acting as a push-pull driver or H-Bridge circuit (Horowitz, 1989). Specification of the component L293D is available in (STMicroelectronics, 2003). Figure 8-25 shows how the PLC signals (CW and CCW) were converted and connected to the actuator signals.

With the PLC software and its separate processing unit, no control algorithm on the manufacturing scenario was expected from the DAQ software. DAQ software therefore was designed to only acquire data from sensor network and to manage them for further processing. Details of data acquisition software design are explained as follows.

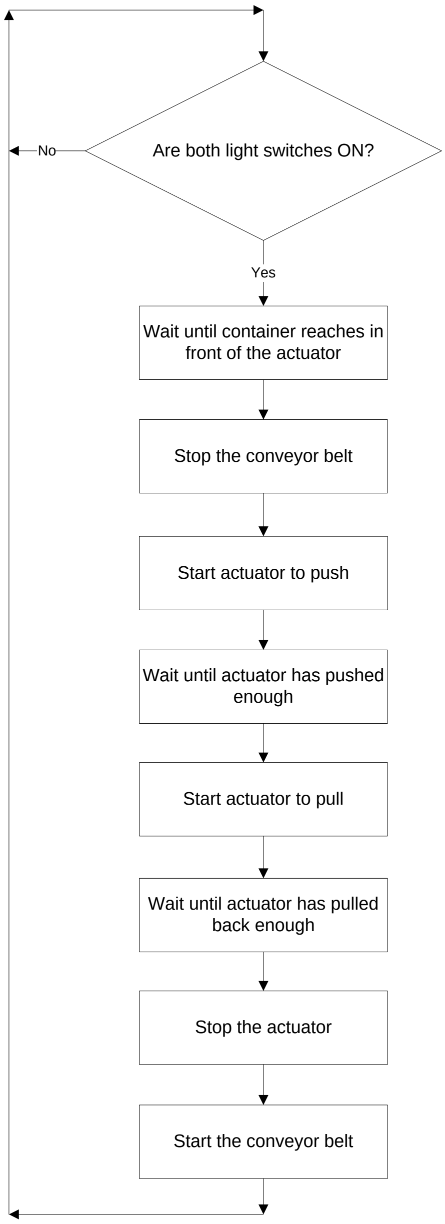


Figure 8-24 Sensing and actuation control software in the PLC

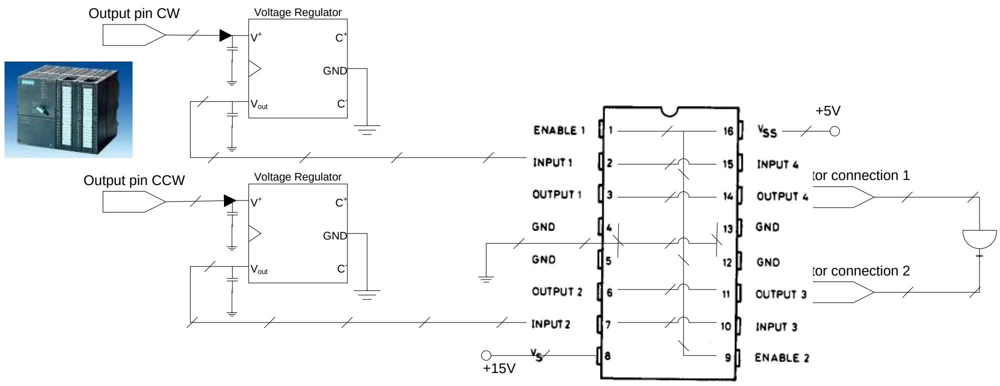


Figure 8-25 Connection configuration between PLC signals and actuator

8.7. DAQ software implementation

A data acquisition software system was designed and developed to manage data acquisition from data points defined in 8.4.1. Important features of the DAQ software are described as the following;

- Input data point definition; Data Inputs and their acquisition attributes could be defined through user interaction. The type of the hardware interface to the data input point, the type of the data input being acquired, and the rate of sampling the data input are among the key attributes which were required by the DAQ software before it could establish a connection and start sampling data. Additionally, input variables could be defined through user interaction with available defined data inputs points and mathematical functions.
- Input / Output interfacing; creation and maintenance of connections to the data input points through PC interfaces using PC operating system (and LabVIEW software) hardware interface resources.
- Sampling; frequent reading of the available data from connected data input points according to the specified sampling rates.
- Curve fitting; a distribution estimation module estimates the attributes of a fit distribution to the Variable Input data for use in Fast-forward mode.

8.7.1. DAQ software components

A block diagram view of the overall organisation of the data acquisition software is illustrated in Figure 8-26.

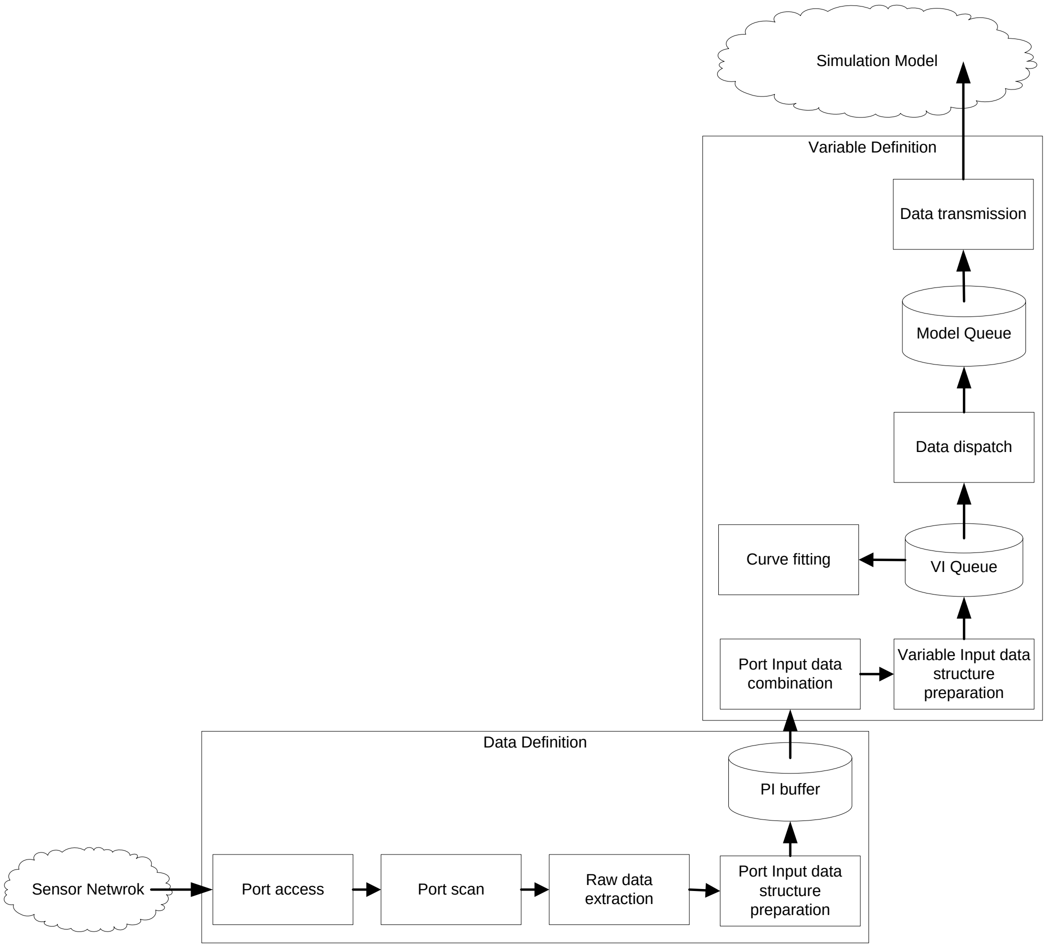


Figure 8-26 DAQ software block diagram

Components of the DAQ software could be summarised in the functionalities and memory spaces which are defined within the responsibilities of the two layers; Data Definition Layer, and Variable Definition Layer. Components of the two layers and data flow between them are separately explained at the following.

8.7.1.1. Port access

Provided the type of communication with the sensors in the sensor network was defined, three types of hardware interface were identified and prepared for definition with parametric attributes. They included parallel-port, USB port, and Ethernet connection.

Connection to four types of interfaces was provided. To establish each connection a set of attributes were defined and sent to the corresponding module which could open the connection. Among the common attributes, are the unique address of the port and the amount of waiting time before the open module cancels opening after failed attempt. Some open modules have more attributes than some other depending on the type of interface and communication protocol. Error handling procedures were also implemented wherever required to repeatedly waiting for a connection.

A reference to the opened connection was sent to another module in charge of reading from the corresponding port. This reference data remains valid until the opened connection is closed. Exceptionally, a parallel port connection opens and reads data and immediately closes connection, so that no reference to the port is provided. Details of each type of port connection will be introduced at the following paragraphs.

8.7.1.1.1. USB port access

A USB connection sends and receives data over a one bit channel in a serialised fashion. For more details and technical information about USB connection reader can refer to standards defined by USB Implementers Forum (USB Implementers Forum Inc., 2008). Schematics of a USB access port is illustrated in Figure 8-27. Attributes needed by the USB port access module are defined in below outlines;

1. Port Name; a unique name associated with the hardware address of the serial port on the computer,
2. Baudrate; the rate of transmission of data on the serial port,
3. Data Bits; the number of bits in the incoming data,
4. Parity; the parity used for every frame to be transmitted or received,
5. Stop Bits; the number of stop bits used to indicate the end of a frame,
6. Flow Control; the type of control used by the transfer mechanism,
7. Enable Termination Character; a Boolean flag to prepare the serial port to recognize the termination char.

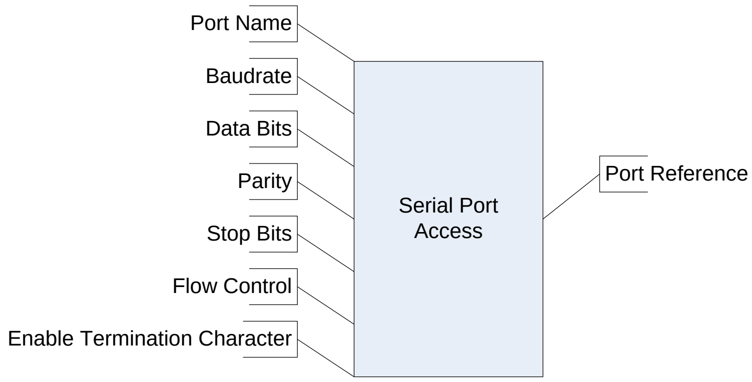


Figure 8-27 Serial port access module inputs and output

More details about the above serial communication definitions are out of scope of this document. Reader can refer to a number of resources including (Park, 2003) for further details.

Once above attribute data are available, a port can be accessed. The access module outputs the Port Reference data to the opened port so that the serial port reader module can find the opened port.

8.7.1.1.2. OPC connection

An OPC connection provides an Ethernet channel through which a fixed memory location, as a data item, could be accessed on a LAN network (OPC Task Force, 1998). Details about OPC technology are provided by (OPC Task Force, 1998), according to which, only two pieces of data are required for establishment of an OPC connection. They are the unique network address of the OPC item, and the type of access whether it is for read or write. A piece of data as the Connection ID is provided by the OPC connection module (as shown in Figure 8-28).

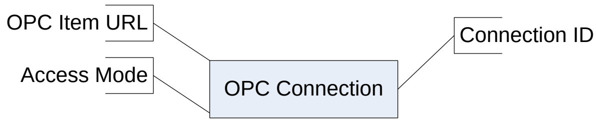


Figure 8-28 OPC connection module attributes

8.7.1.1.3. Parallel port access

Data on the parallel port can flow as 8-bit bytes at once. Complete detail about parallel port communication is available at (Park, 2003). Like in OPC connection module, a parallel port connection requires two pieces of information before it can access the port. They are port address and access type. Exceptional to the other port access types, once a parallel port access is opened and data on the port is read, the connection closes until it is opened again for the next read, like in Figure 8-29.

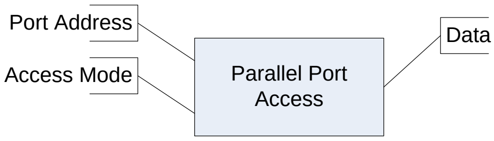


Figure 8-29 Input output attributes of the Parallel access module

8.7.1.1.4. Demo-Box serial port access

Connection to the National Instruments' Demo-Box was provided via USB serial port and Modbus communication protocol. Definition of this protocol is outside the scope of this work, and available at (Modbus Organization Inc., 2010). The associated connection module supported Modbus communication protocol and therefore required only port name information to initialise a connection. Like in Figure 8-30, it then provided a reference data for Demo-Box's read or write module.

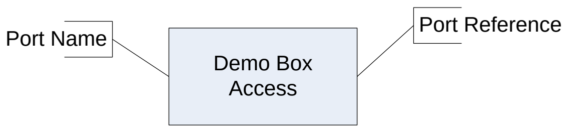


Figure 8-30 Attributes of the Demo-Box access module

8.7.1.2. Port scan

Corresponding to each of the defined hardware interfaces a read action was defined which took the port address details and accomplished the transfer of the data from port towards the computer processing unit.

The Port Scanner or Port Reader modules required few other attributes regarding the type and amount of data to read. In this section the flow of input and output data for each of the four port-scan modules are introduced.

8.7.1.2.1. USB port scan

The serial port reader module took two pieces of data before they could read; the number of scanning bytes as well as a reference to the serial port. A series of characters representing the series of scanned bytes were available at the output of the module once they scanned the port (Figure 8-31).

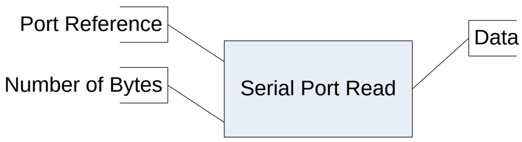


Figure 8-31 Inputs and output of the Serial port reader module

8.7.1.2.2. OPC item scan

An OPC read module read the OPC item once three pieces of data were provided; the Connection ID to the OPC item, the type of data to be read, and the amount of time to wait for a value update to become available in the connection buffer (Figure 8-32).

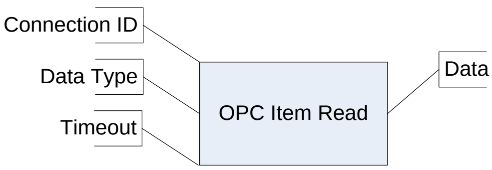


Figure 8-32 Attributes of the OPC item read module

8.7.1.2.3. Parallel port scan

As already mentioned in Parallel Port Access data flow description (i.e. 8.7.1.1.3), in software aspects, scanning from the Parallel Port occurs at the time of accessing the port. Therefore, no data is flown apart from the scanned byte at the output of the Parallel Port Access block in Figure 8-29 repeated here in Figure 8-33 for convenience.

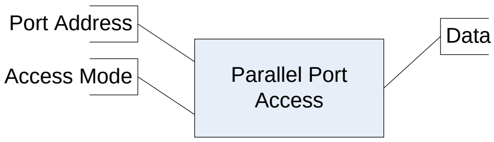


Figure 8-33 Input output attributes of the Parallel access module

8.7.1.2.4. Demo-Box serial port scan

A Demo-Box reader module reads from separate groups of inputs on the Demo-Box at separate times. On each read cycle, the series of analogue sensors were read immediately after the series of binary switches were scanned. Each reader module similarly needed six pieces of data before it scanned input data; Modbus Command, as a cluster of three attributes included the access type (being read or write), the starting address of the series of inputs in the Demo-Box, and the number of inputs to be read. Serial Parameters to specify the encoding scheme of the scanned bytes, being ASCII or RTU (Simply Modbus, 2008), Port Reference, and finally Timeout (Figure 8-34). Once one read function was operated, an array of scanned values were available at the output of the read module, which in turn required separation before each single input could become available. The task of separation will be described in the Raw Data Extraction section which comes next.

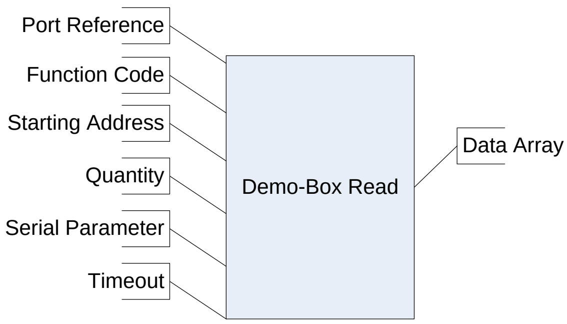


Figure 8-34 Attributes associated with the Demo-Box read module

8.7.1.3. Raw data extraction

Depending on the type of Data Input, scanned data required to be prepared for the actual value of the sensor output. Occasionally, data might be corrupted and unfeasibly out of the range that should be under any circumstance. Such inconsistencies were checked and verified at this stage and the healthy portion of data passed to the next stage.

After a series of bits or bytes were scanned from a port, depending on the type of sensor connected to the port, the actual sensor data required to be extracted and the rest of the scanned data discarded. For example, a binary switch data was actually one bit included within a scanned byte. Therefore, few attributes were required to justify the method of data extraction. This is explained for each combination of sensor data type and port in below paragraphs.

8.7.1.3.1. RFID tag extraction on the USB port

The RFID readers which were used on this implemented system, when reading an RFID tag, generate few characters extra than the actual tag ID and attach them as suffix and prefix to the tag ID before sending them to the port. Therefore, Tag ID itself required to be separated from these extra characters before it could be useful. Such data extraction module received three adjustment parameters and scanned data itself as input, and delivers the tag ID as output. Figure 8-35 shows inputs and output of this functionality.

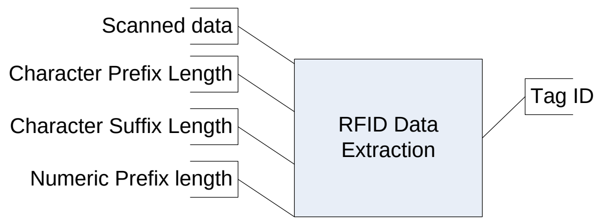


Figure 8-35 Inputs and output of the RFID data extraction module

8.7.1.3.2. Array member data extraction

Regardless to the type of port interface, if the scanned data is an array of bytes or bits including the byte or bit value representing a sensor data, it required an extraction function to be able to assign the single value to the associated Port Input data. The single value could be a binary switch sensor data on the output of the OPC item data, or Parallel Port data, or the Demo-Box. Alternatively, it could be a single analogue value on the array of analogue values scanned on the Demo-Box. Such array indexing functionality required index information to point at the location of the bit within the array. The symbolic view of the functionality, together with its inputs and output is shown in Figure 8-36.

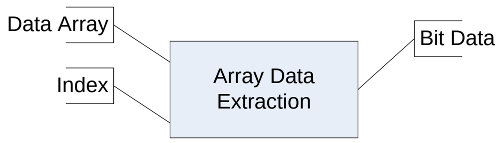


Figure 8-36 Inputs and outputs of the Array Data Extraction module

8.7.1.4. Port Input data structure preparation

After the value of the sensor output was deterministically prepared, it was transferred into the proper structure which helped with evaluation of the value at Variable Definition Layer. This structure included the value of Data Input entity, and the timestamp. This data structure would not require a unique identity to the Port Input data since it was then transferred to the unique buffer spaces corresponding to the Port Input (explained in below paragraph).

Once the actual data entity was extracted out of the scanned Port Input data, it is attached to a timestamp value representing the time of the attachment, and was left available to the next layer which constructed Variable Input data out of Port Input data. This function required only Port Input data value and timestamp value to give out a cluster of the two input values (Figure 8-37).

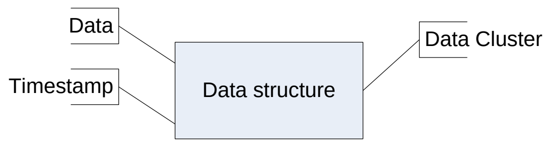


Figure 8-37 Connections of the PI structure module

8.7.1.5. PI buffer

As a part of the real-time database structure, this memory space took care of the Port Input data between the two layers. One PI buffer is assigned to each Port Input. A PI buffer was either written by its corresponding Port Input data from Data Definition Layer, or alternatively read by Variable Definition Layer. In the second case, it was read to be combined and build up a Variable Input data.

The module responsible for writing the value of the Port Input cluster into the buffer required a reference to the buffer which was supposed to be accessible before the reference was left available to this module. Like in Figure 8-38, the PI cluster and the buffer reference enter this module.

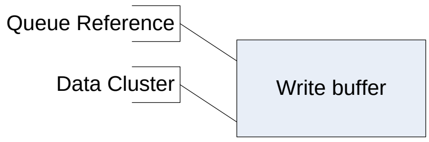


Figure 8-38 Buffer writing module

8.7.1.6. Variable Input evaluation

At the early stage of the Variable Definition Layer the values of defined Port Inputs were combined to each other to evaluate the value of the Variable Input using the defined mathematical functions.

Available Port Input data in the corresponding buffers were subject to scanning by modules in charge of constructing the Variable Input data. The Variable Input Evaluation module scanned only the values of the Port Input queues which were involved in its construction. It therefore required a series of references to their queues as well as a reference to another queue structure opened for storage of the Variable Input data value. The module also required a reference to the formula which dictated the evaluation method of the Variable Input data out of the corresponding Port Input data. The only output of this module was the value of the Variable Input data. Figure 8-39 shows the associated inputs and output of this module.

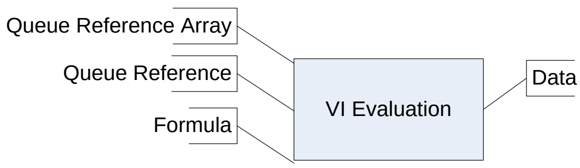


Figure 8-39 Variable Input Evaluation module attributes

8.7.1.7. Variable Input data structure preparation

Variable Input data values were organised in structures similar to the Port Input data, which is a cluster comprising the value of the Variable Input, and the timestamp of evaluation of the value.

Once the value of Variable Input data was evaluated, it was clustered together with a timestamp value. The output cluster was left available in a queue structure for dispatching towards the simulation system. The structure preparation procedure was similar to the Port Input data structuring.

8.7.1.8. VI queue

The two-piece Variable Input (VI) data was queued up in queue-based databases and left available to the Data Dispatch module before it could be sent to the simulation model. Similar to the Port Input (PI) Buffers, VI Queue was uniquely assigned to each VI.

Again similar to the PI buffers, each VI had one corresponding queue-based database opened and accessible by the Write Queue module. For dispatching and all other further purposes these queue databases hold updated VI data.

8.7.1.9. Data dispatch

This module is in charge of preparing information which was to be sent towards the simulation model. It integrates Variable Inputs in one single queue (Model Queue). Member of this queue were then available to the module that transmitted them over to the simulation system. Integrated Variable Inputs at this stage were structured according to the format which helped the R3M connectivity in the simulation model. Timestamp was attached to a unique variable identification code and the actual value of the VI.

Values of VI data were collected by Data dispatch module in a continuous loop and buffered in the Model Queue. This module did not output anything and required the set of references to the VI Queues as well as the reference to the Model Queue (Figure 8-40). The three-piece cluster of VI ID, VI value, and timestamp was structured in this module before buffering. This module employs Queue Writing module which also worked for both PI and VI buffering tasks.

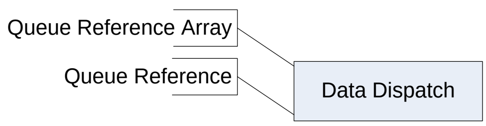


Figure 8-40 Data dispatch module inputs

8.7.1.10. Model queue

The queue-based database was common between all Variable Inputs. Therefore, a queue with first-in-first-out (FIFO) storage scheme was selected in order to take care of the time order of the creation of the VIs.

8.7.1.11. Data transmission

An Ethernet (TCP/IP) socket was opened at the beginning of the data acquisition execution and left accessible for both transmission and reception of data between the DAQ system and the simulation model. Stored Variable Inputs were frequently collected from the Model Queue one-by-one and sent over to the simulation model. This functionality terminates the series of performances required to run as fast as possible so that the real-time simulation was least affected.

Model Queue was accessed by Data transmission module; its members were taken off one-by-one frequently and sent to the simulation system. This module required two references; one to the opened Ethernet connection, and the other reference to the Model Queue (Figure 8-41).

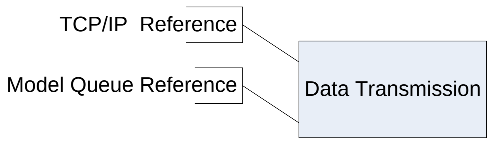


Figure 8-41 Data Transmission module inputs

In the next section a demonstration of the performance of the DAQ software functionalities on the targeted main features will be given. The four main features mentioned as Customization, I/O Interfacing, and sampling.

8.7.1.12. Curve fitting

The module takes series of data values by accessing references to their array. It then applies data series to an error measurement function that includes several probability distributions and generates as output a reference to the fittest distribution among those available. This reference together with the generated attributes of the fit distribution are then stored in a file associated to the original data series.

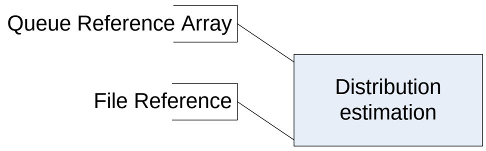


Figure 8-42 Curve fitting module inputs

8.7.2. DAQ software user interfaces

Figure 8-43 to Figure 8-46 show how parameters for definition of Port Inputs and Variable Inputs were entered by designed user interfaces. In one window-based panel depending on the selections of some parameters, other sub-panels appeared and let the user to complete definition of one Port Input at a time. Parameters of a defined Port Input could be amended or entirely deleted. In Variable Input definition user interface, a series of defined and available Port Inputs helped user to define an arbitrary mathematical function between one Variable Input and an arbitrary number of defined Port Inputs. For example, in Figure 8-43 a Port Input was given the name p2 and ID 2. This Port Input was assigned to an RFID sensor connected via USB port with Serial Port settings seen in the figure. RFID tags are to be extracted through the parametric settings seen in the same figure. In Figure 8-47 a Variable Input was defined with the name v2 and ID 5. This VI was built based on the Port Inputs P1, P4, and P7 according to the formula given in the same figure.

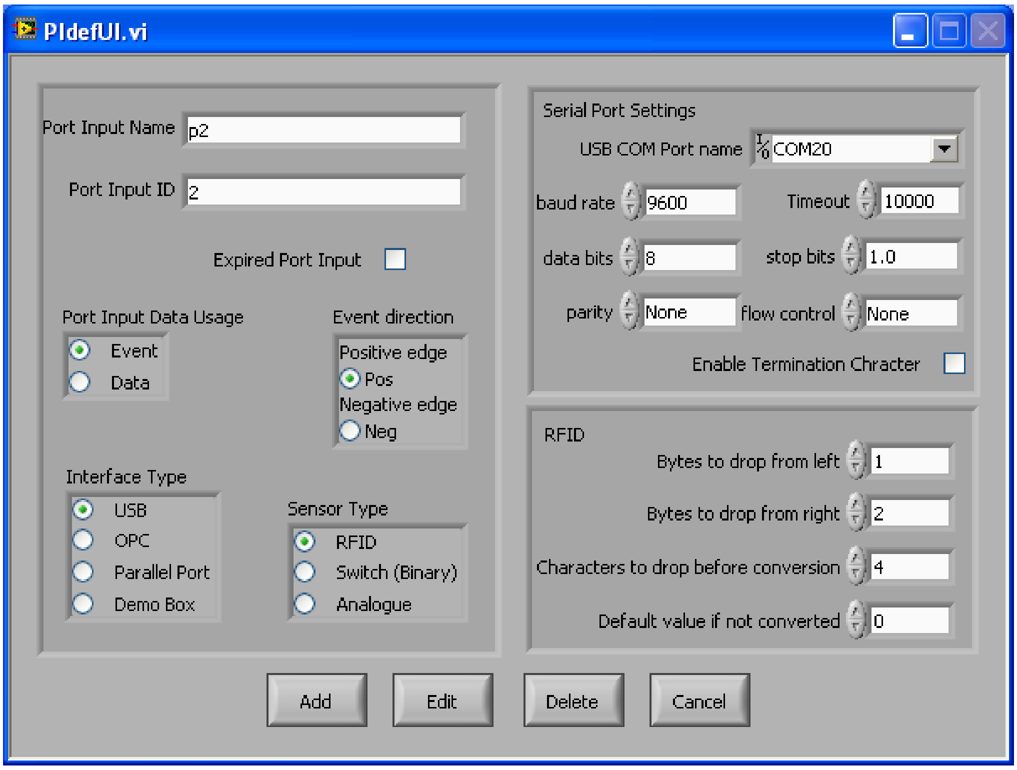


Figure 8-43 USB Port Input definition user interface

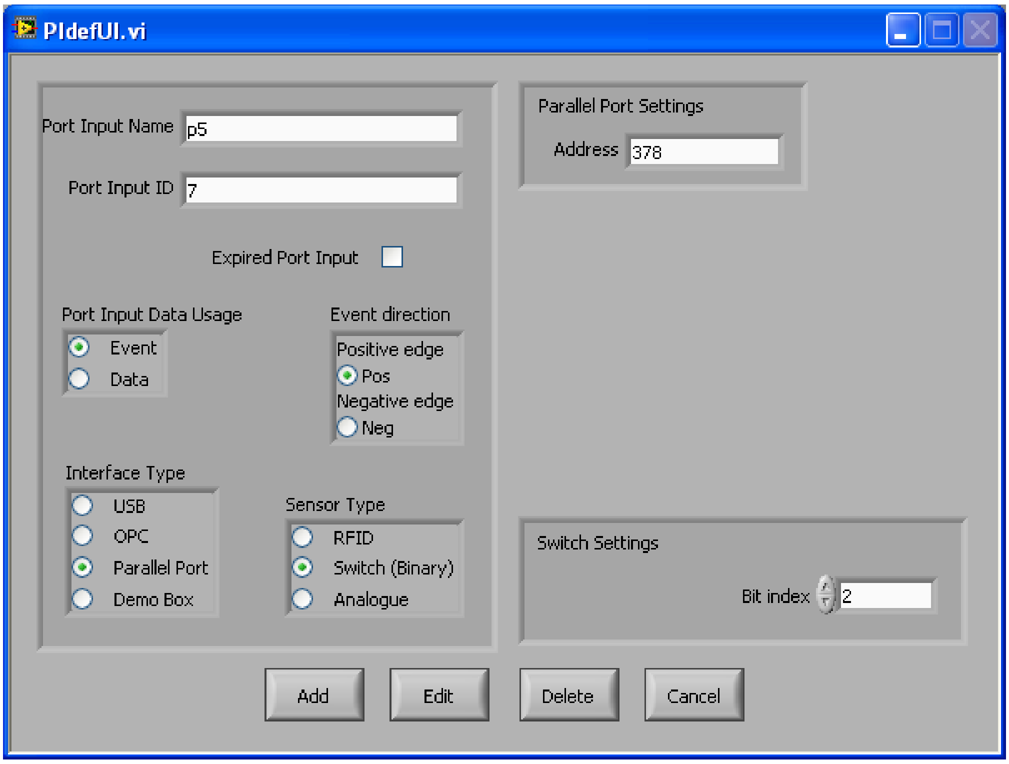


Figure 8-44 Parallel-port Port Input definition user interface

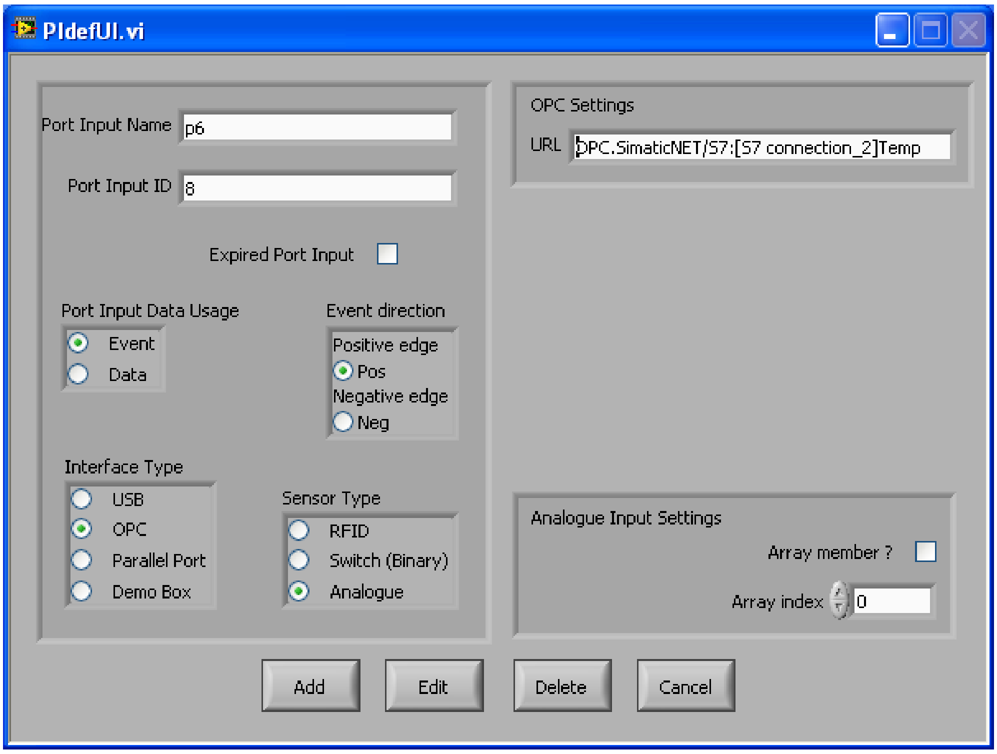


Figure 8-45 OPC Port Input definition user interface

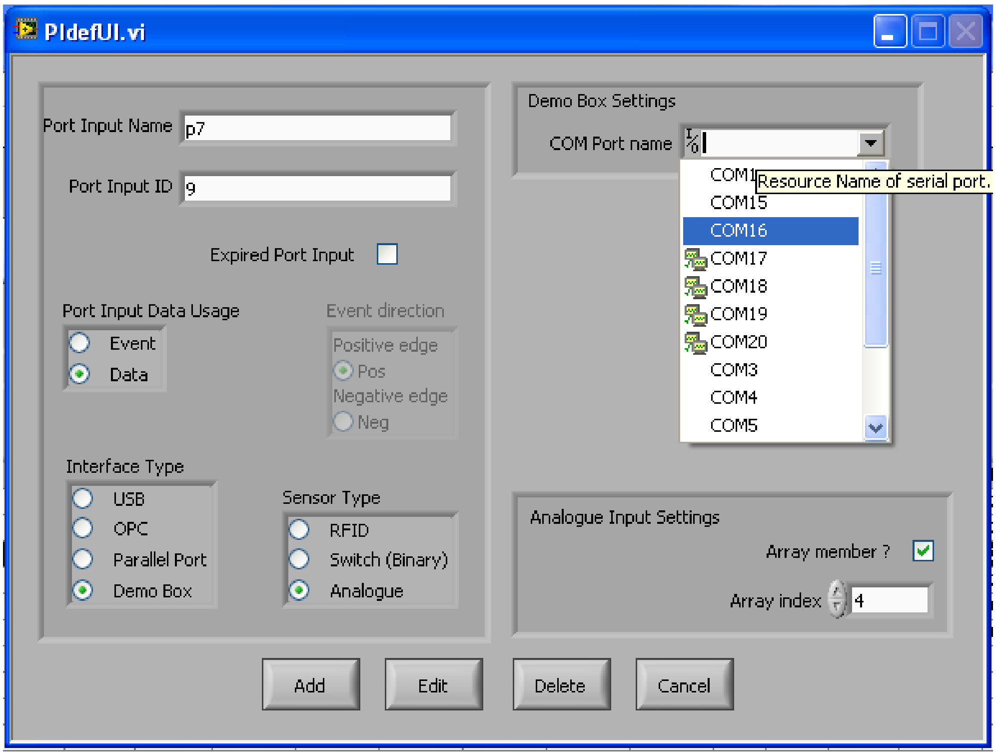


Figure 8-46 Demo-Box Port Input definition user interface

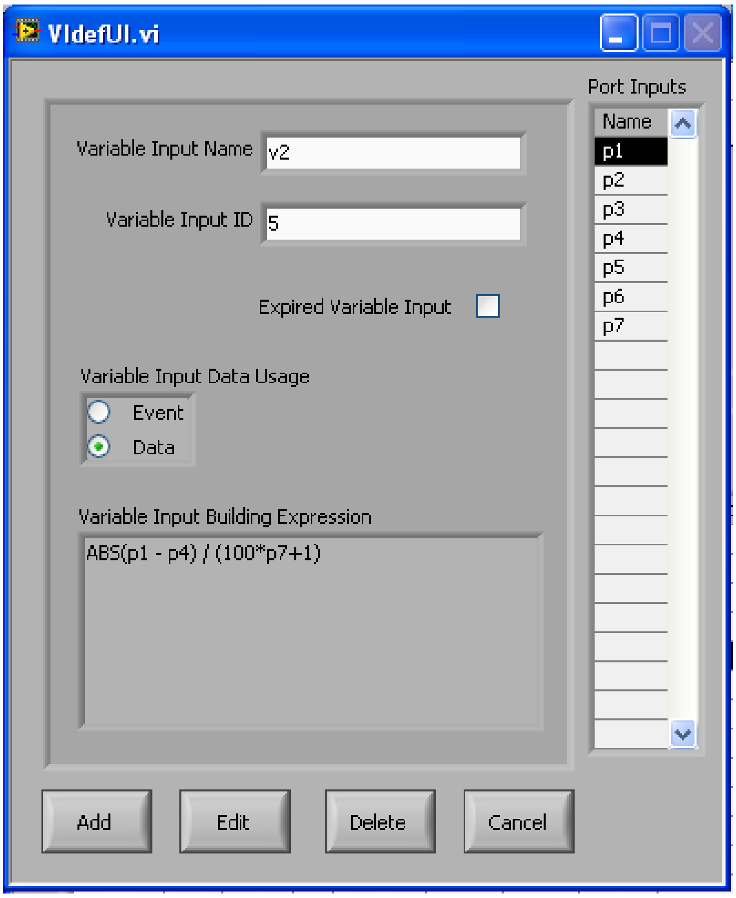


Figure 8-47 Variable Input definition user interface

Defined Port Inputs and Variable Inputs were listed in tables in Figure 8-48 and Figure 8-49. As it could be seen in the figures, Port Inputs r1 to r5 are assigned to RFID sensors connected via USB ports. P0 to p7 represented eight light switches connected to the Parallel Port connection. d1 to d6 were six binary switches on the Demo-Box and a1 to a8 assigned to the eight analogue sensors on the same Demo-Box. Finally, o1 to o6 represented six OPC items. On Figure 8-49 it could be seen that VIs v1 to v9 and u1 to u4 are defined equivalent to one of the PIs.

All Port Input and Variable Input definitions could be edited or deleted through the definition user interfaces at any time before the actual data acquisition started.

Port Inputs

Name	ID	Usa	Dir	Interface	Sensor	Expi	EnTe	Timeo	Flo	Sto	Par	Da	Baudrate	USB Port	URL	Pport add	DBox Port	Byte	Byte	Cha	Def	Bit inx	Anlg arr.	Arr
r1	5 101	0	0	0	0	0	0	10000	0	10	0	8	9600	COM17				1	1	4	0	0	0	0
r2	5 102	0	0	0	0	0	0	10000	0	10	0	8	9600	COM20				1	1	4	0	0	0	0
r3	1 1	0	0	0	0	0	0	10000	0	10	0	8	9600	COM18				1	1	4	0	0	0	0
r4	5 103	0	0	0	0	0	0	10000	0	10	0	8	9600	COM19				1	1	4	0	0	0	0
r5	3	0	0	0	0	0	0	10000	0	10	0	8	9600	COM5				0	0	0	0	0	0	0
p0	5 200	0	0	2	1	0	0	0	0	0	0	0	0			1020		0	0	0	0	0	0	0
p1	5 201	0	0	2	1	0	0	0	0	0	0	0	0			1020		0	0	0	0	1	0	0
p2	5 202	0	0	2	1	0	0	0	0	0	0	0	0			1020		0	0	0	0	2	0	0
p3	5 203	0	0	2	1	0	0	0	0	0	0	0	0			1020		0	0	0	0	3	0	0
p4	5 204	0	0	2	1	0	0	0	0	0	0	0	0			1020		0	0	0	0	4	0	0
p5	5 205	0	0	2	1	0	0	0	0	0	0	0	0			1020		0	0	0	0	5	0	0
p6	5 206	0	0	2	1	0	0	0	0	0	0	0	0			1020		0	0	0	0	6	0	0
p7	5 207	0	0	2	1	0	0	0	0	0	0	0	0			1020		0	0	0	0	7	0	0
d1	50007	0	0	3	1	0	0	0	0	0	0	0	0				COM4	0	0	0	0	0	0	0
d2	50008	0	0	3	1	0	0	0	0	0	0	0	0				COM4	0	0	0	0	1	0	0
d3	50009	0	0	3	1	0	0	0	0	0	0	0	0				COM4	0	0	0	0	2	0	0
d4	50010	0	0	3	1	0	0	0	0	0	0	0	0				COM4	0	0	0	0	3	0	0
d5	50011	0	0	3	1	0	0	0	0	0	0	0	0				COM4	0	0	0	0	4	0	0
d6	50012	0	0	3	1	0	0	0	0	0	0	0	0				COM4	0	0	0	0	5	0	0
a1	40001	1	0	3	2	0	0	0	0	0	0	0	0				COM4	0	0	0	0	0	1	0
a2	40002	1	0	3	2	0	0	0	0	0	0	0	0				COM4	0	0	0	0	0	1	1
a3	40003	1	0	3	2	0	0	0	0	0	0	0	0				COM4	0	0	0	0	0	1	2
a4	40004	1	0	3	2	0	0	0	0	0	0	0	0				COM4	0	0	0	0	0	1	3
a5	40005	1	0	3	2	0	0	0	0	0	0	0	0				COM4	0	0	0	0	0	1	4
a6	40006	1	0	3	2	0	0	0	0	0	0	0	0				COM4	0	0	0	0	0	1	5
a7	40007	1	0	3	2	0	0	0	0	0	0	0	0				COM4	0	0	0	0	0	1	6
a8	40008	1	0	3	2	0	0	0	0	0	0	0	0				COM4	0	0	0	0	0	1	7
o1	5 207	0	0	1	1	0	0	0	0	0	0	0	0		opc:/			0	0	0	0	0	0	0
o2	5 208	0	0	1	1	0	0	0	0	0	0	0	0		opc:/			0	0	0	0	0	0	0
o3	5 209	0	0	1	1	0	0	0	0	0	0	0	0		opc:/			0	0	0	0	1	0	0
o4	50210	1	0	1	2	0	0	0	0	0	0	0	0		opc:/			0	0	0	0	1	0	0
o5	50211	1	0	1	2	0	0	0	0	0	0	0	0		opc:/			0	0	0	0	1	0	0
o6	5 210	0	0	1	1	0	0	0	0	0	0	0	0		opc:/			0	0	0	0	0	0	0

Figure 8-48 User interface showing the table of defined Port Input points

Variable Inputs

Name	ID	Usage	Expired	Buidling Expression	Port Input names	Port Input Indices
v1	5 101	1	0	r1	r1	0
v2	5 102	1	0	r2	r2	1
v3	1 1	1	0	r3	r3	2
v4	2 2	1	0	r3	r3	2
v5	5 103	1	0	r4	r4	3
v6	5 200	0	0	p0	p0	5
v7	5 201	0	0	p1	p1	6
v8	5 202	0	0	p2	p2	7
v9	5 203	0	0	p3	p3	8
u1	5 204	0	0	p4	p4	9
u2	5 205	0	0	p5	p5	10
u3	5 206	0	0	p6	p6	11
u4	5 207	0	0	p7	p7	12

Figure 8-49 User interface showing the table of defined Variable Input points

After the definition of PIs and VIs was done, the actual data acquisition was started. For that to happen, a number of instances of data definition layer equal to the number of defined PIs were automatically loaded with associated parameters and executed. Similarly, Variable Definition task components were automatically called and fed with the set of parameters per each corresponding VI.

8.7.3. Input / output interfacing and data sampling

Both data and variable definition layers had some modules which ran continuously. The stream of data from sensor network towards the real-time buffers and queues was therefore established.

A recorded time-limited snapshot of the execution of the system is available as media file on the attachments to this thesis. Buffered data at each of the PI buffer, the VI Queue, and the Model Queue were logged together with their timestamps so that the delay between buffering stages could be measured. Figure 8-50 shows the average time delay on the block of logged data. It could be seen that the overall time delay never exceeded 300 milliseconds. Such delay for the scenario running on this implemented system is considered short enough to keep up with the concept of real-time data acquisition and real-time simulation.

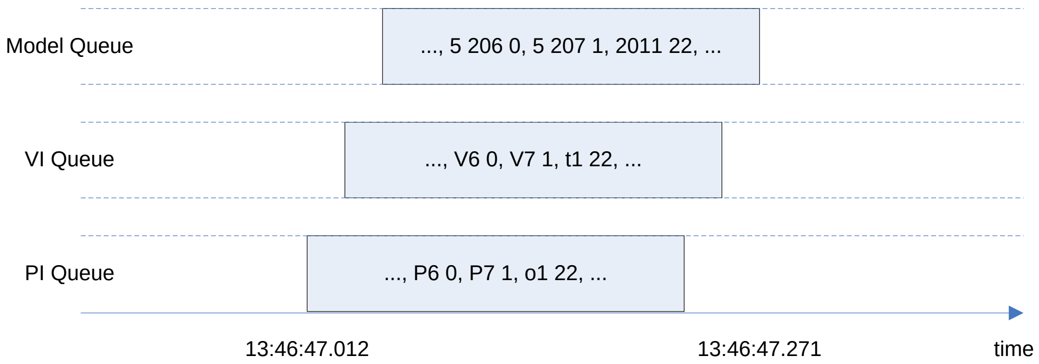


Figure 8-50 Data transfer delay between different stages of the DAQ system

In Figure 8-50 it is important to associate corresponding values of PI, VI and model messages to each other. For example, according to the settings of the implemented system, Port Input P6 is used to define Variable Input V6, similar definition applies to P7 and V7, O1 is required for evaluation of t1 and finally generation of model message ‘2011 22’.

The generated time delays are due to the non-deterministic nature of the relationship between independent software modules including acquisition, evaluation, and dispatch. Of course the higher the number of inputs and their data sizes, the higher the overall time delay.

8.8. Simulation model implementation

A simulation model has been designed and developed as one of the two software tiers of the implemented suite of components. This model was built based on the chocolate manufacturing scenario defined in the implemented system and previously described in section 8.6.1. The model features the capability to be connected to the DAQ system of the implemented system in real-time and receive data from it. The model architecture is described at the following section.

8.8.1. Simulation model architecture

The overall architecture of the simulation model highlights use of real-time model matching mechanism (R3M) at every point where real-time data should affect the flow of entities of the model. This can be, as could be seen in Figure 8-51, change of the value of a parameter, or start and end of a process, or effectively any other occurrence of event in the discrete-event simulation environment.

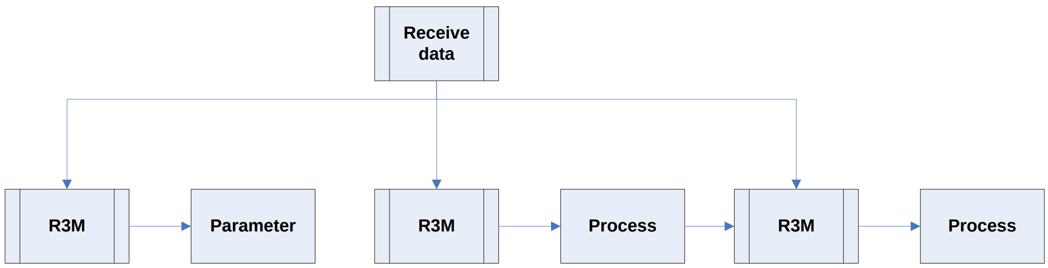


Figure 8-51 Overall architecture of the simulation model

In general, based on the entity-driven nature of the simulation model, flow of entities of the model in different blocks caused change of status and thus, the execution of the model scenario. According to the architecture, for every received data from DAQ system, an entity, a dummy messenger entity, took responsibility to adhere the data to the correct R3M connection. Upon arrival of the messenger entity to the R3M connection and use of accompanying data by the corresponding R3M block, a change in the status of the model will occur depending on the type of block associated to the R3M block. The main entities of the simulation model continue flowing until they reach another R3M connection.

Arena simulation software has been used for designing and developing this model. At the following, major components of the model together with their design steps and relationships will be described. The actual software of the model is available as attachment to this thesis.

8.8.2. Simulation model components

Since this model was supposed to be connected to the DAQ system for receiving real-time data on the flow of products and events of the processes, certain parts of the model relied on the messages exchanged with the DAQ system. Therefore implementation of R3M mechanism was spread over the model wherever a real-time connection to the model was thought of. Since real-time connection of the main entry of the flowing products in the model is associated with the data received from the DAQ system, there would not be any creation process in the simulation model to create product entities automatically and without an influence from the DAQ system. Messaging mechanism over TCP/IP is also used for sending model parameter values over to the DAQ software. Due to the nature of the manufacturing scenario of the implemented system, staff existence detection was among the real-time tasks of the model.

The following paragraphs will explain the building blocks of the simulation model designed and developed for the implemented system.

8.8.3. Message reception

Being the prerequisite of the R3M implementation, this component was in charge of receiving messages and disseminating their parameters throughout the model. This was made possible by ripping the message off to some parameters and assigning those parameters to the model variables and / or entity attributes. Of course, a decoding procedure was defined in order to interpret the contents of which received message was meant to fulfil which part of the model. Arrivals of the products into the first part of the simulation model, start or end of a process by an entity, updating the value of a sensor variable were among the main consequences of this component depending on the type of the contents of the messages. Figure 8-52 shows these three possible paths after a received message is decoded.

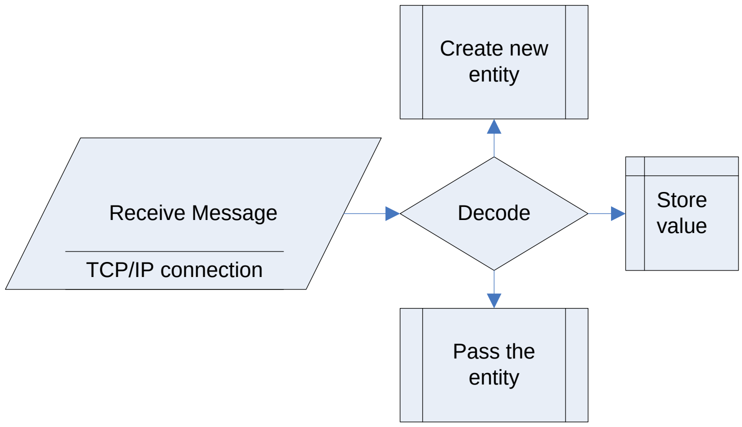


Figure 8-52 Message reception component of the simulation model

8.8.4. R3M components

Two major tasks played role for R3M in the model. The first component generated a process code or so called process passport ID and kept it in a queue buffered memory until it was detected by an entity looking for exactly this passport ID. Once this ID code was found, the memory space is released. This is symbolised in Figure 8-53. The process passport ID was generated using both serial number of the entity and the location ID of the process to guaranty its uniqueness.

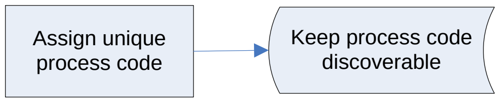


Figure 8-53 Buffering a passport ID until it is hunted

The second part of the R3M implementation was used whenever an entity was to pass a certain point in the model based on a real event. This logic is designed according to the blocks in Figure 8-54.

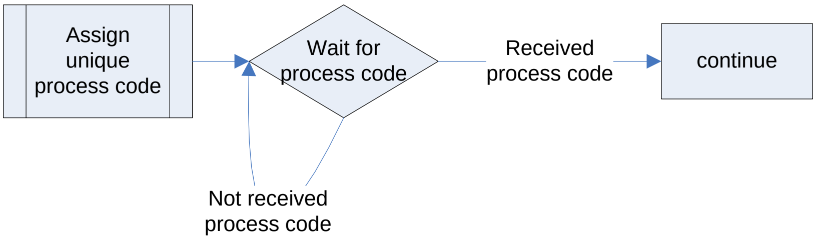


Figure 8-54 ID check and continue component of the simulation model

8.8.5. Status consistency detection

This component featured a continuous check to keep track of existence of one or a series of similar particular situation (Figure 8-55). The status or situation in this implementation is existence of staff on certain areas of shop floor.

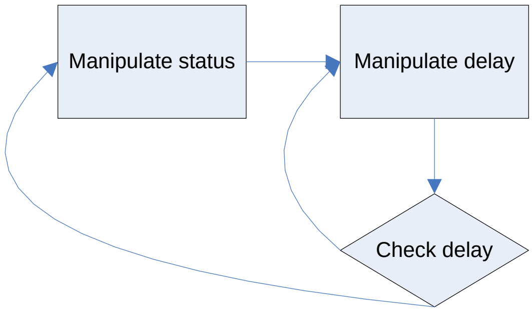


Figure 8-55 Status consistency detection component

8.8.6. Fast-forward execution

The task reads attributes of variable data distributions from memory storage and defines model parameters of each associated process or parameter to the particular variable. Simulation model entities will then be created based on the dictated distributions (Figure 8-56).

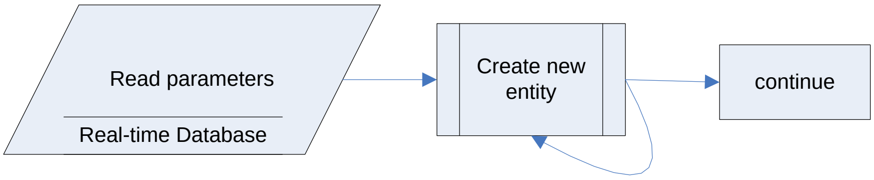


Figure 8-56 Entity creation based on stored distribution parameters

8.9. Simulation model design

Rockwell Software's Arena software was used to design and develop the simulation model for this implementation. According to the implemented system's manufacturing scenario, except in the packing stage, other stages were sequentially followed by each product sequentially with the product moving through. The fundamental entity which travelled along the modules of the model corresponded to the product or in the case of this implementation, a container of chocolate bar. Eight processes and their associated resources were in charge of the flow of products in the model. Major components of the model were identified and their interrelations were defined. At the following, software components and their connections will be described.

8.9.1. Product arrival

The fundamental building block of this task is about message reception from DAQ system (via TCP/IP). As it could be seen on Figure 8-57, each entity was created according to a corresponding message. The created entity, or the modelled product, held some customised attributes which received their values from the message. These attributes included the unique serial number of the created entity (or product), its type index, and finally the product's unique process code associated with the location from where the entity was expected to continue the process.

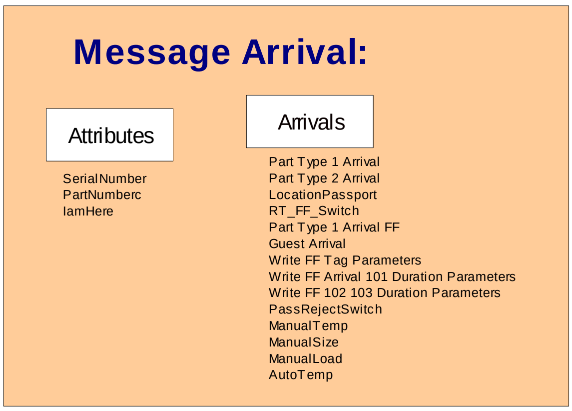


Figure 8-57 Message types and customised entity attributes in Message Arrival component

8.9.2. Connection establishment

An Ethernet connection between the machine on which simulation model runs and the machine which holds the DAQ component is sought. Only one connection is established between the two components even if both components run on the same machine. Simulation model component receives messages from DAQ component.

8.9.3. Entity handling

When the content of the message is to create or change the status of an entity, two tasks must be accomplished in order to control the access to the entity:

1. Triggering the correct entity inside the model; once all information is ready for the reflection on the receiver process, if the reaction belongs to creation or change of status of an entity, or a bunch of entities, the targeted entities must exist at the corresponding receiver point. Each entity on the simulated model, at the moment of creation, is given a unique identifier. Carrying the serial number of the entity with the message and checking it against the serial number of the entity on the simulated model guaranties the concurrency of the simulated model with the physical environment.
2. Handling entity queues and timings; at certain receiver points where entities require to be checked against a message before they could progress, arrival entities build up a queue. Upon receiving a message on the receiver point, the queue is searched for the particular entity. The entity which is found is de-queued and released to progress to the process.

8.9.4. ID check in / check out

Flowing entity should be heading towards the first process block which corresponded the baking process. Entering this process or any of the further processes was due to a wait-and-check procedure until the corresponding sign was received by the message reception task and prepared for checking.

Similarly, prepared signals from message reception task waited for being checked by the corresponding waiting entity.

In Figure 8-58 the actual modules which have been used to develop this task are shown. For each location a new buffer was prepared in the design of this component.

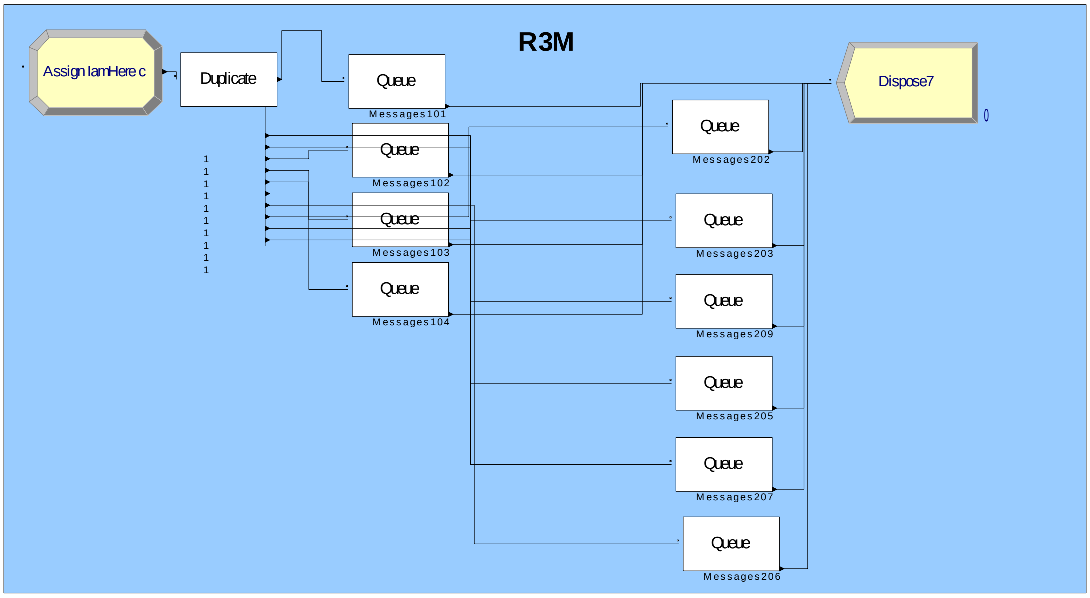


Figure 8-58 Buffers keep passport IDs until they are found

The actual ID generation and look up procedure is shown in Figure 8-59.

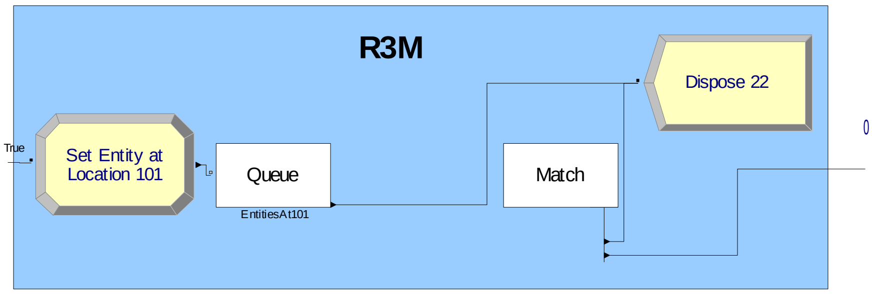


Figure 8-59 Continuation of an entity based on the existence of a corresponding process code

8.9.5. Staff existence check

Implementation of status consistency detection component targeted frequent checking upon those parameters of the model which were updated by means of messages received from the DAQ system and in turn from the active RFID tag reader device. Figure 8-60 shows how this loop was actually designed and developed.

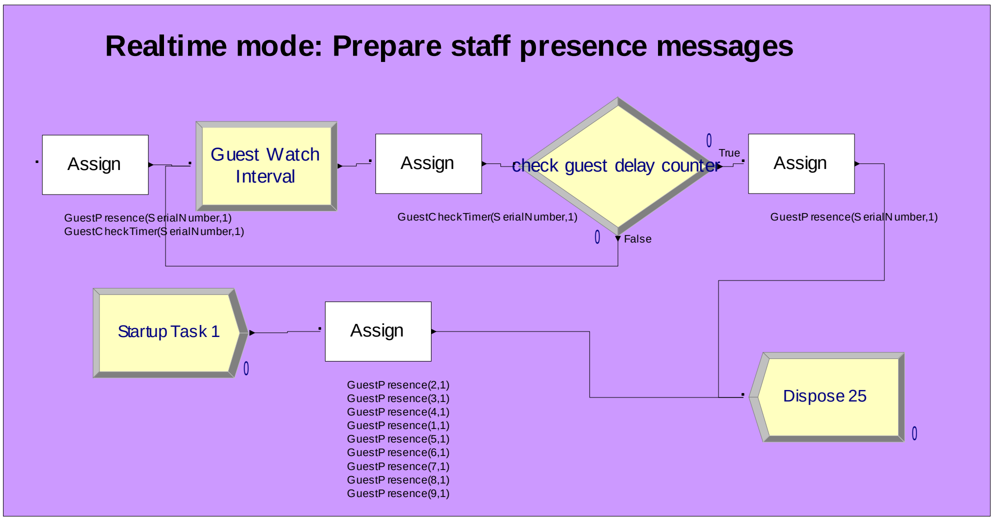


Figure 8-60 Staff existence detection model

8.9.6. Fast-forward data retrieval

Switching from real-time mode to fast-forward mode of execution stopped creation of entities that ran through check-in/check-out route. Therefore, wait-and-check procedure did not happen and entities which were now created according to dictated distributions handled the flow of processes. Parameters of distribution were acquired from storage memory storage at switching time. Figure 8-61 shows the actual model blocks responsible for these actions.

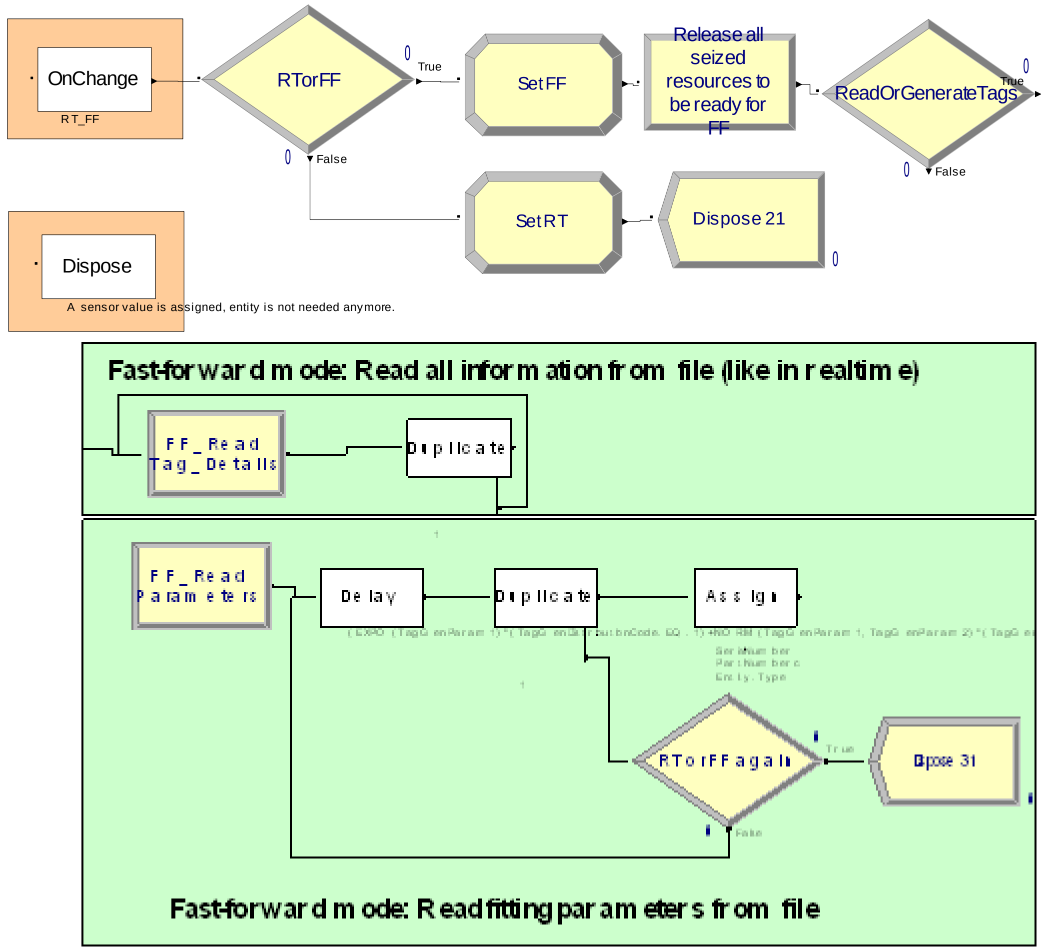


Figure 8-61 Fast-forward data retrieval components

8.10. Summary of implementation

This chapter covered a series of activities that demonstrated the flexibility of data definition, real-time data acquisition, real-time and fast-forward simulation features of the data integration system proposed in this thesis. A platform for connection to sensor network from one end and to simulation model from the other end has been designed and developed. The platform ensured definition of a variety of data sources as input data to system and definition of latent variables from input data by using mathematical functions available options. This was demonstrated in user interfaces and values of acquired data and variables based on the definitions.

For sensor network, a series of sensors and data acquisition drivers and technologies was associated to it. Sensor network included proximity and range RFID readers with digital output on RS232 data communication connection, on/off light switches, temperature, load, sound, and luminance sensors with analogue outputs connected either directly to controller device or through an electronic amplifier circuit. OPC data communication service has been established between sensor output data connected to controller and acquisition system. PLC controller device has been programmed to run a simple sensing and actuating logic as well as gather data for OPC service.

For simulation model, an imaginary manufacturing scenario was defined to use data from sensor network. Connection of data from acquisition system to the processes and / or parameters of the simulation model were established by using real-time model matching mechanism R3M. Real-time connection and execution of simulation model based on real-time data as well as fast-forward information were demonstrated in several laboratory-based seminars. A recorded media file is available as attachment to this thesis.

The next chapter will demonstrate the implementation of EvenTracker sensitivity analysis method proposed in this thesis in the application of the implemented platform on a real factory-based manufacturing plant.

9. Empirical Study

The purpose of this chapter is to introduce proof of concepts proposed in this thesis by implementing EvenTracker sensitivity analysis method and demonstrating its efficiency in comparison with other capable sensitivity analysis methods. This – as will be shown - was achieved by providing a factory-based experiment.

9.1. Task overview

This research received the attention of a refrigerator production plant which manufactures refrigerators and freezers. The empirical study on this factory started with the collection of data from one of their several manufacturing sections which makes side-panels for both fridges and freezers. The tasks and components of this section have been identified and modelled in terms of sequential processes with their associated sensors and actuators. The collection of data from signal flow on the sensors and actuators of each process has been accomplished on time intervals so that chunks of data was available to the data acquisition system designed and developed on the implementation. In order to take full advantage of the empirical data, the DAQ system must send variable data to simulation model and receive performance information from it. Like the laboratory platform of the implementation, a simulation model of the identified manufacturing section has been built and connected to the DAQ system for a complete run of the implemented system. The result of this empirical study consists of the collected data and simulation model of one of manufacturing sections of the factory as well as the outcome of testing and verification of the concepts of this work based on those data.

At first, the environment within which the empirical study has been accomplished will be described. Then the collected data and its role in test and verification of EvenTracker sensitivity analysis method will be introduced. Next, the designed simulation model will be described. At the end of this chapter, the outcome of the test and verification of EvenTracker method will be discussed and compared with the one of another capable sensitivity analysis method. At the following sub-sections side-panel production line together with their signals are described.

9.2. Side-panel manufacturing line

A side-panel manufacturing line at a refrigerator production plant has been used as implementation platform. The overall operations and machineries in this section were grouped into six sequential operations. The first four task groups, outlined at the following, were subject to this study (TecnoTeam, 2000). Installed sensors and actuators of each of the four task groups are shown in Figure 9-1. Table C-1 provided in appendix C summarises the role and data type of each signal. A brief description of the production process is explained here:

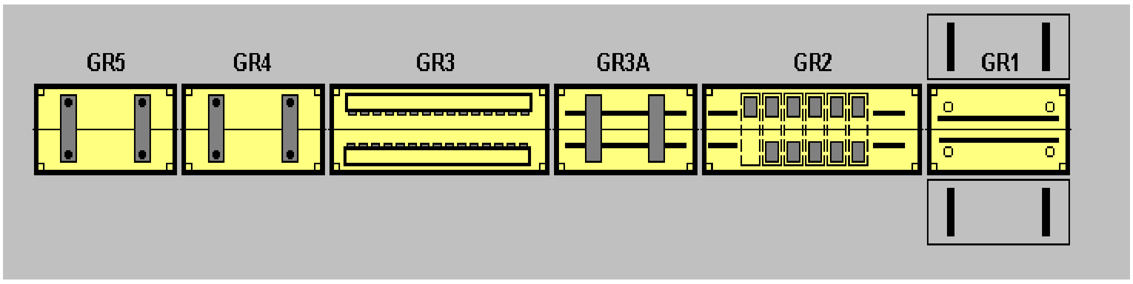


Figure 9-1 Side-panel manufacturing line layout

9.2.1. GR-1 or Loader

An automatic loader with two trolleys feeds the tinplate sheets to the line. The sheet is picked up by suckers operated by a pneumatic cylinder. The suckers are controlled by a Venturi-type valve which assures the vacuum for the piece to be picked up and the air blow for the piece to detach. After the sheet has been picked up, permanent magnets make the sheet hang from polyurethane belts and hold it in place after it has detached from the suckers. The sheet is then fed into the GR-2 group through the belts, when the station is free, another piece is picked up. Figure 9-2 shows a side view of the main frame of the Loader machine.

Figure 9-2 GR-1 or Loader process

Four sensors and three actuators were installed on the loader machine (Figure 9-3). Sensor signals were used to ensure about the location and situation of the metal sheets during loading operation. Actuators signals were used to command the actual loading performances including sheet grabbing and sheet relocation. In table C-1 (Appendix C) a summary of the signals and their roles and data types in loading operation is held.

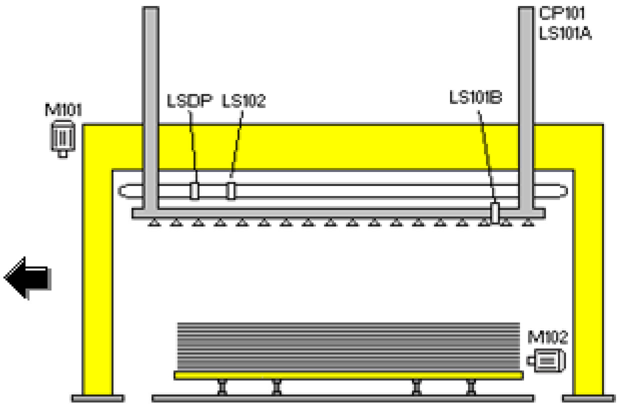


Figure 9-3 Sensors and actuators of GR-1 or Loader

9.2.2. GR-2 or Shearing Unit

The sheet coming from the loader is fed to the positioning lock and centred by the pneumatic centring devices. It is picked up by the mechanical hand controlled by CNC and the shearing cycle starts according to the PC-program and the model to be manufactured. When the shearing process is completed, the mechanical hand will take the piece to a fixed position which will be the same for all the models where the piece will be released. Then the mechanical hand will reset and wait for another piece to be sheared. The conveyor belts will convey the finished piece to the following station. Figure 9-4 a,b show the Shearing unit from entry and exit ends respectively.

(a) (b)

Figure 9-4 GR-2 or Shearing unit from a) entry and b) exit side

A number of sensors and actuators on this operation were fixed regardless to the shearing program. Each added punching device added two sensors and one actuator signal control to the operation. Figure 9-5 gives an illustration of number and location of sensors and actuators of the shearing unit. Table C-1 (Appendix C) summarises all fixed signals and a sample of the similar signals. Listing all manual blanking die actuator signals (like CO201) and their associated couple of sensor signals (like LS201A, LS201B) are avoided as they are fully similar.

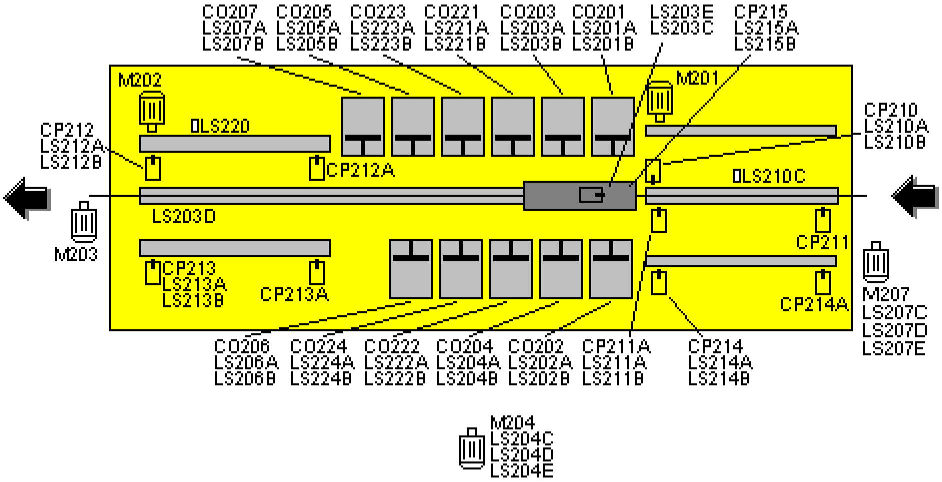


Figure 9-5 Layout of sensors and actuators on the GR-2 or Shearing unit

9.2.3. GR-3A or Tilter

The tilter allows the piece to be reversed after the shearing process has been completed so that the panel can be roll-formed, the border upwards. The tilter is made of a steel structural work, on which a belt conveyor and the magnet needed for the transport of the pieces are assembled (Figure 9-6). The reversal of the piece is controlled by an a.c. motor with inverter. Once the piece is reversed, the conveyor belt feeds it to the roll forming machine.

Figure 9-6 GR-3A or tilter machine

Few sensors and actuators which generate the signals of the operation are mentioned in table C-1 (Appendix C) and their layout demonstrated in Figure 9-7.

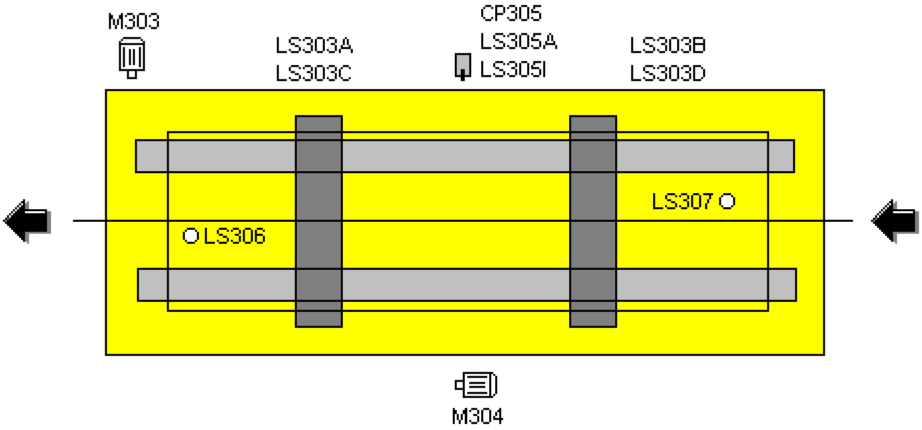


Figure 9-7 GR-3A or tilter machine layout and its sensors and actuators

9.2.4. GR-3 or Roll forming machine

The machine consists of a steel structural work and roll forming heads. The heads are clamped to the structural work on one side (stationary standard) and to a movable plate (movable standard) on the other side. The movable plate is fixed on ball guides and can be adjusted through an a.c. motor with inverter. Such an adjustment allows manufacturing two different models of panels. Figure 9-8 gives a picture of the roll forming machine from entry side.

Figure 9-8 GR-3 or Roll forming machine

The main role of the few sensors and the only actuator in this operation is about rolling the tinplates forward to the other end of the machine. Table C-1 (Appendix C) summarises this role. Layout of sensors and actuators on the roll forming machine is shown in Figure 9-9.

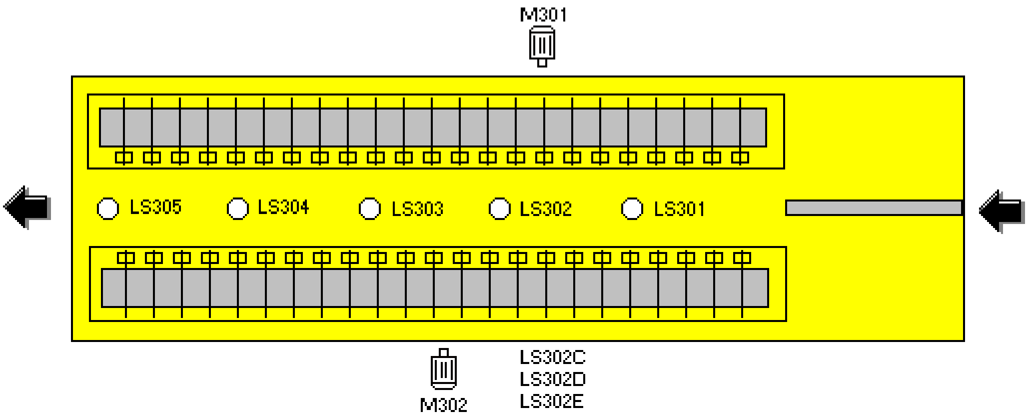


Figure 9-9 layout of sensors and actuators of GR-3 or Roll forming machine

All signals on the side-panel manufacturing line carry binary data of either one or two bits. All 2-bit signals represent the direction of the corresponding operation, for example, if a conveyor belt is moving forwards or backwards. Within the scope of this study, either direction triggers the same event which is activating an actuator. The following section describes how the signals are interpreted into triggers and descriptions of production process.

Twenty eight signals are used as input variables. National Instruments LabVIEW software tool was used to develop the data acquisition platform to collect the sample signal data acquired from the shop floor. The acquired data was then fed into a discrete event simulation software package (Arena™ Rockwell Automation). This integration allows for direct translation of multiple signals to production performance analysis tool (Tavakoli et al, 2008). R3M mechanism described in chapter 6 (5.2.4.5) and in (Tavakoli et al, 2008) was implemented so that the relationship between signals and processes is well-defined within the structure of the model. For example, the model is capable to relate signals from LS301 and LS305 to the starting and ending of the forming process shown in Figure 9-9. It is however important to note that this awareness will not be used in the estimation of the sensitivity indices in

EvenTracker method. However, it will help with the validation of the results of this work in the way that each more important TD which is reported less important will be regarded as a ‘false negative’. Margins of performance of EvenTracker method will be discussed under the assumption of not having any false negatives.

9.3. Discrete-event model of side-panel manufacturing line in a refrigerator production factory

A real-time discrete-event simulation model of the production process of side-panel manufacturing line has been designed and developed to help with measuring production performance parameters based on the same type of architecture and category of components that have been introduced in the implementation chapter (chapter 8). The proposed scenario coupled with several sensor and actuator signals through R3M mechanism. Therefore, the simulated model would require the real-time flow of the signals to receive from the DAQ system in order to run. The following Event-driven Process Chain (EPC) diagrams of Figure 9-10 to Figure 9-13 show modelled processes and their constituent operations. It also shows signals that trigger commencement and the end of each task (i.e. an event). The parameter chosen for production performance analysis is the Instantaneous Resource Utilisation.

9.3.1. Side-Panel manufacturing scenario

In the proposed manufacturing scenario, each operation group was considered as one process which concluded few consecutive tasks or processes. The below EPC diagrams show the signals and their location used in connection to the model. Each diagram shows a major process and its component tasks and functions. It also shows signals which, like and event, trigger the start and end of each component task. In the implementation of these processes temporal order of the events, indicated as arrow connections between the events, dictates the order of the processes. Of course, the first event of each diagram started the major process and the final event ended it.

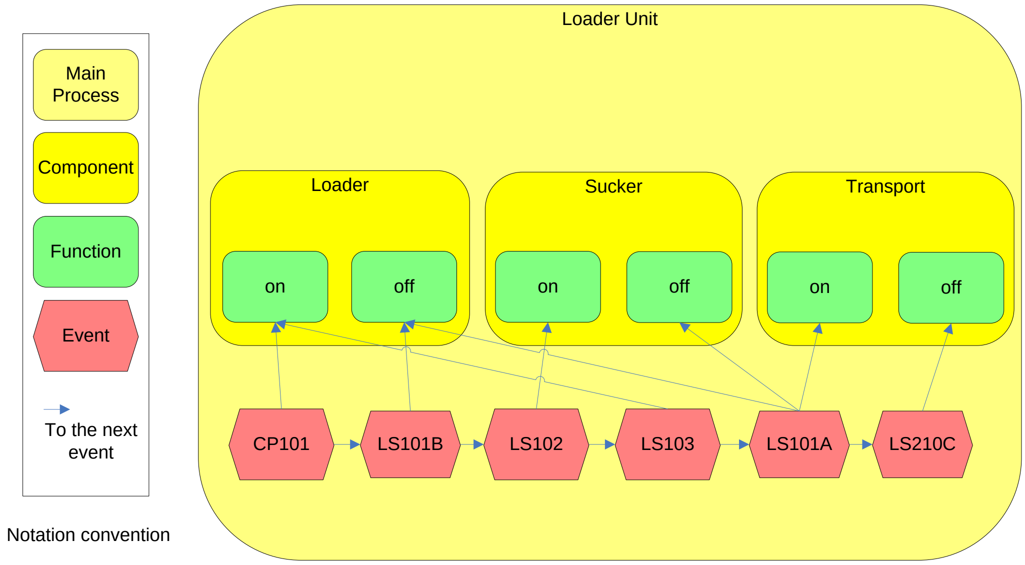


Figure 9-10 Loader process components and flow of events

As it could be seen on EPC diagrams, some signals triggered more than one function of the process. Wherever a function required more than one signal to operate, logical AND combination of the multiple signals triggered the function.

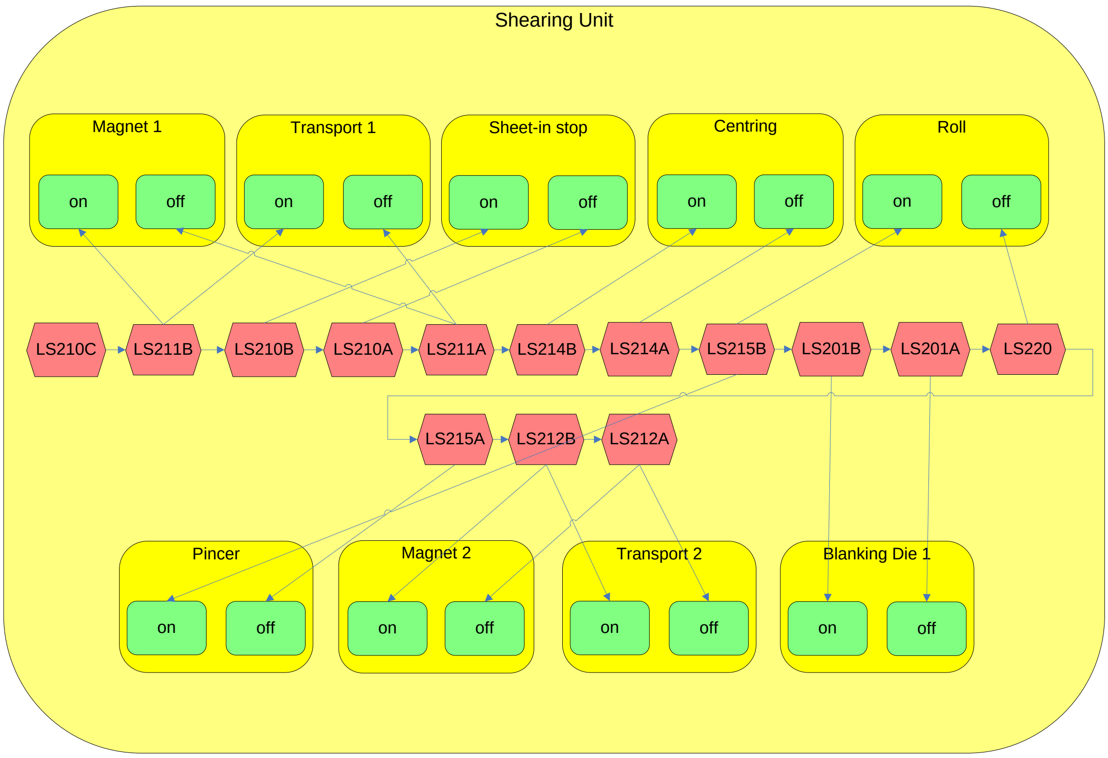


Figure 9-11 Shearing Unit process components and flow of events

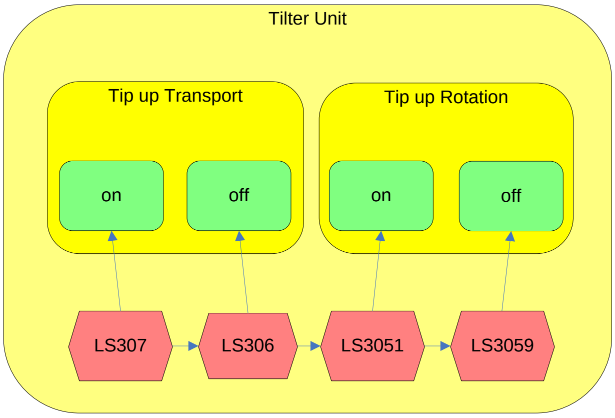


Figure 9-12 Tilter Unit process components and flow of events

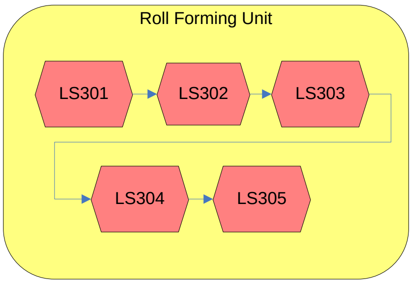


Figure 9-13 Roll forming Unit process components and flow of events

9.4. Real-time data streaming into the model

The DAQ system designed and developed in the implementation of this work has been used to connect to the central control system of the refrigerator factory's side-panel manufacturing line and collect data from the operation signals. The central control system for this manufacturing line was consisted of three PLC devices only one of which included the processor unit and the other two were containing input/output connections (Figure 9-14). The processor part of the PLC was a Siemens S7-315 processor unit and its associated power supply and industrial network communication processor. The Input / Output connection modules were two E200 devices of Siemens family. Since all sensors and actuators of all operation groups were connected through one PLC system, OPC connection was used as the only type of data point for data acquisition. Figure 9-15 shows a schematic of the configuration of hardware and software components for data collection.

Figure 9-14 PLC in charge of side-panel manufacturing central control

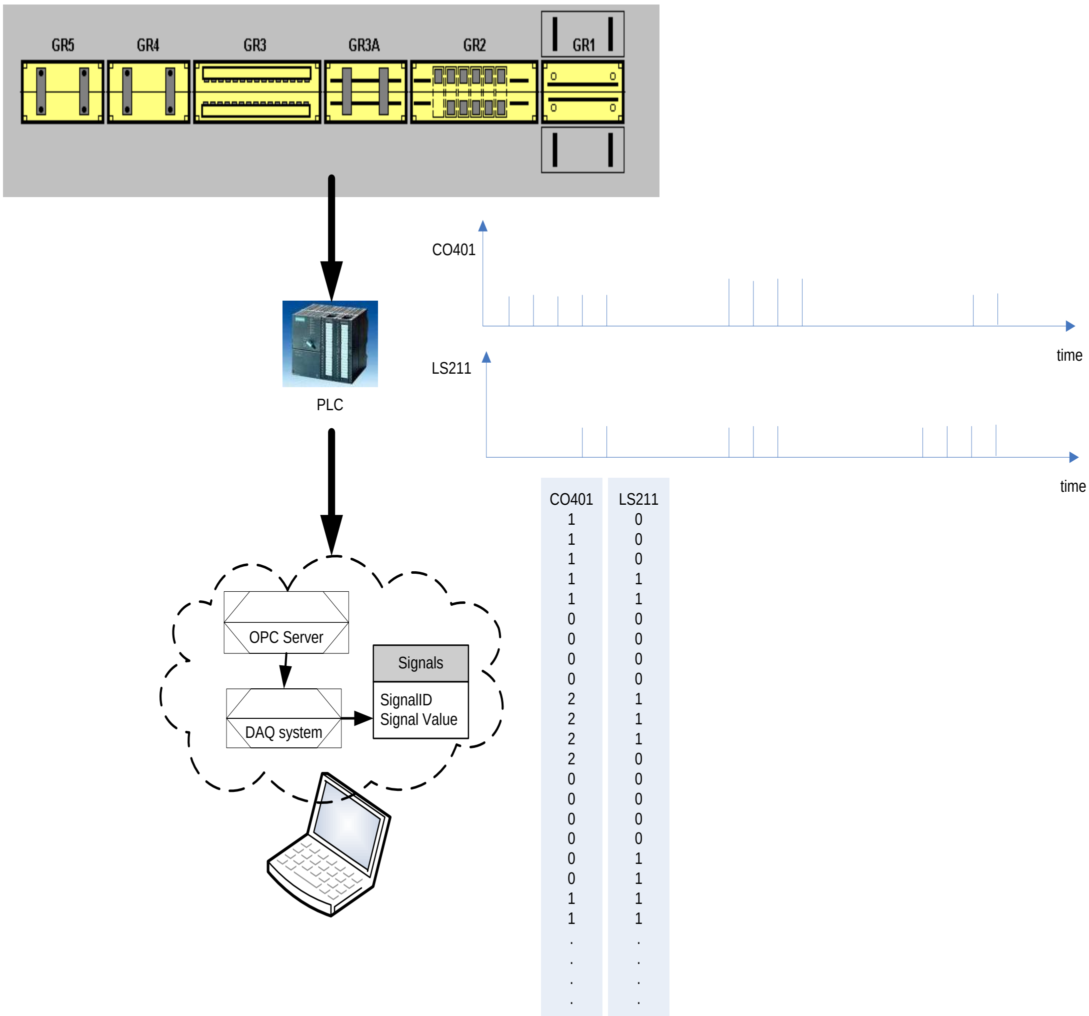


Figure 9-15 Connected components for data collection from side-panel manufacturing line

The data from the side-panel manufacturing line were acquired and fed into the model by DAQ system at the same rate as they were collected during a period of 2 minutes (i.e. 500 data points at 5 samples per second). The event data to be collected were to measure resource utilisation of machines to perform the four tasks in the production process. In conjunction with the 28 TD values for four event data series were acquired and logged.

As it can be observed in the table of machinery signals (Figure C-1 in Appendix C), all sensor signals are acquired in binary two-state mode, and all actuator signals are also generated in binary two-state mode. There is no single analogue value entering or exiting the central control system. Moreover, many signals particularly input signals, have a nature of an event, meaning their value changes only when there is a detection state and all other times they have a steady 'background' value. This shows the importance of events among the two defined types of entry data (i.e. event and data). An almost 20-second snapshot of the signal data, or PI values is shown in Figure 9-16.

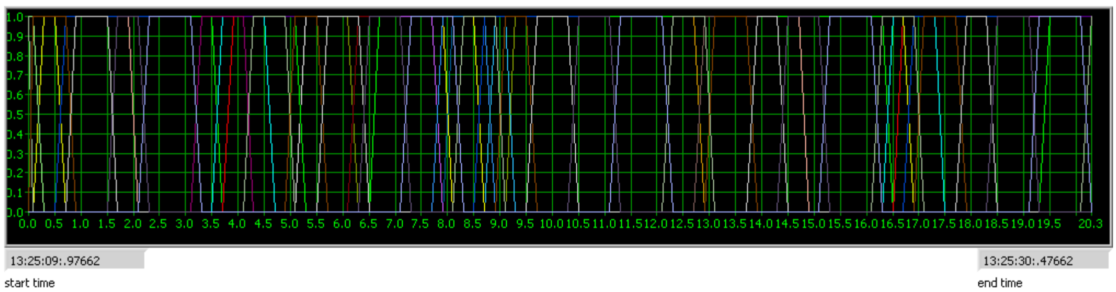


Figure 9-16 A 20-second snapshot of sampled signals

9.5. Implementation of EvenTracker sensitivity analysis

EvenTracker sensitivity analysis method has been designed and developed in LabVIEW software and implemented on DAQ system in conjunction with the components of Variable Definition Layer. Blue shaded blocks of Figure 9-17 show the position and relationship of EvenTracker components of the DAQ software system.

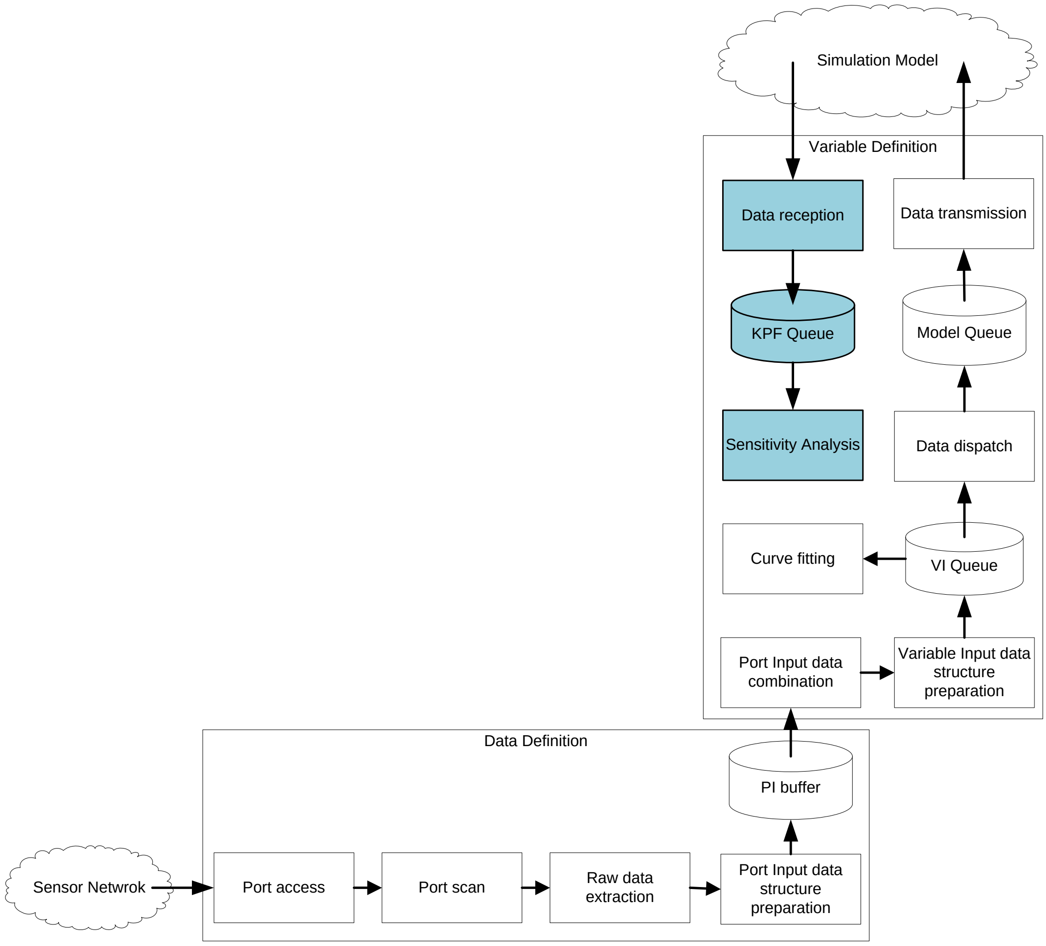


Figure 9-17 EvenTracker sensitivity analysis implementation components

However, requirement of EvenTracker method to the values of simulation model outputs (values of Key Performance Factors or KPF) necessitated implementation of a functionality that could manage transmission of KPF values towards DAQ system where sensitivity analysis executes. This functionality was designed and developed in the simulation model with details described at the following.

9.5.1. KPF transmission to DAQ

Simulation model parameters are updated on a regular basis and could be accessed for presentation purposes or for further analysis or delivery to other destinations inside or outside model. A message departure component model was designed to frequently read from the values of defined model outputs e.g. Key Performance Factors (KPF). Figure 9-18 shows the schematic of the actual model blocks that implement the functionality. Selected parameters were sent to the DAQ system through the same TCP/IP connection as used for message arrival component introduced in the design of product arrival component in chapter 8 (8.9.1).

Each value of a selected KPF was packed in a message accompanied by the KPF name so that it could be identified in DAQ system. Due to the format of available blocks in modelling environment (Rockwell Software's Arena), one block performed the task of transmission, and a number of blocks, one for each KPF, prepared the message.

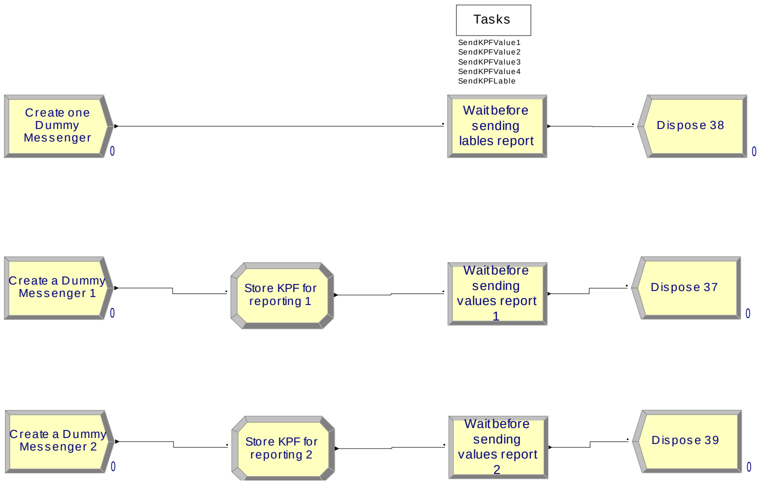


Figure 9-18 Every departing parameter meets a send request

9.5.1.1. Data reception

The same TCP/IP connection which was opened for data transmission module in Variable Definition Layer was used for data reception from model. The TCP/IP connection was left open at the beginning of the data acquisition execution and left accessible for both transmission and reception of data between the DAQ system and the simulation model. The received data from the simulation model was queued up in a buffer memory space to be decoded for the value of KPF sent from model. This module needed two references; one to the opened TCP/IP connection, and the other reference to the KPF Queue (Figure 9-19).

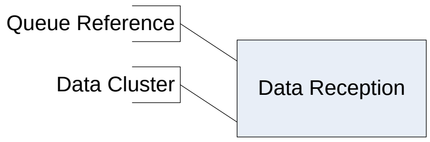


Figure 9-19 Elements of data reception module

9.5.1.2. KPF queue

Queued KPF messages were frequently accessed and decoded for identification of the performance parameter in the message. After it was identified which performance factor was sent in the message, the decoded value was queued up in the corresponding queue as the performance parameter.

9.5.1.3. Sensitivity analysis

This module retrieves values of KPF and Variable Input data series in terms of event data (ED) and trigger data (TD), respectively, from their queue storage and performed an analysis of sensitivity of each KPF to each Variable Input. Execution of EvenTracker method, as shown in Figure 9-20, generated a two dimensional array of sensitivity indices in time domain. The cycle was repeated for different values of Event Thresholds and Trigger Thresholds.

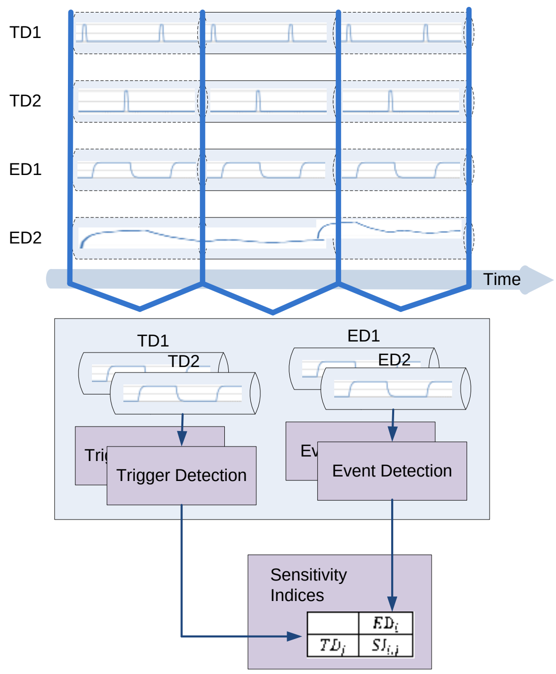


Figure 9-20 Implementation of EvenTracker for multiple input variables (trigger data series TD1, TD2) and output parameters (event data ED1, ED2)

9.6. Validation of threshold parameters in EvenTracker

The approach to measure the efficiency of EvenTracker sensitivity analysis method was based on increasing the speed of computation and reduction in computational overhead without compromising the quality of analysis. The key objective of EvenTracker is to inform DAQ system in real-time of the least important trigger data so that they could be filtered out by DAQ system and the lower number of input variables can be transferred to the data processing layer (simulation model).

To validate the proposed threshold parameters used in EvenTracker the following was conducted:

- Step 1: Establish the maximum number of least important trigger data and eliminate them from the analysis.
- Step 2: Find the optimum for search slot, event trigger, and trigger threshold.
- Step 3: In order to validate the optimisation process conducted in steps 1 and 2, compare the results with a scenario where the total number of trigger data has been included in the analysis (i.e. no reduction in data load).

To establish how to decide on the maximum number of the least important trigger data series, EvenTracker was implemented to read all 28 Trigger Data (TD) series and generate sensitivity indices for four Event Data (ED) series with respect to the trigger data series. Event Threshold (ET) and Trigger Threshold (TT) values of 50% and Search Slot (SS) period of 5 seconds are considered for this attempt. The results are shown in Figure 9-21 in four line charts representing the values of normalised sensitivity indices of four ED series with respect to each of 28 TD series. Normalised sensitivity indices are scaled according to the left vertical axis.

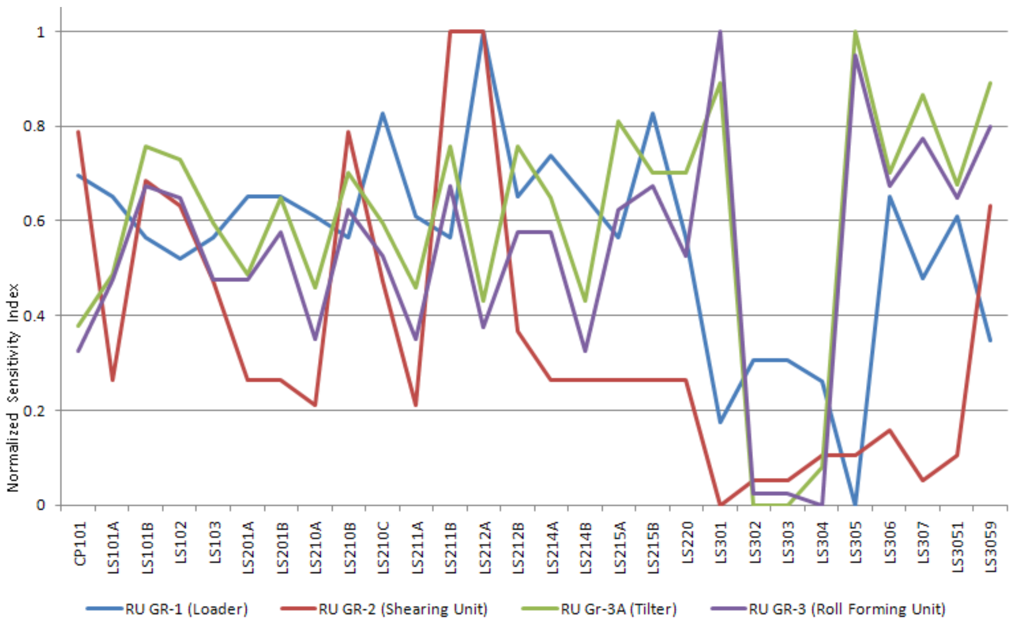


Figure 9-21 EvenTracker sensitivity indices of 4 EDs with respect to 28 TDs

By observing the results shown in Figure 9-21, the event data shows higher sensitivity towards a few number of TD from the list of 28 TDs used in this example. This process allowed elimination of the unimportant TD for the period of analysis.

For evaluation purposes, more important TDs were dominated by completely filtering out those TDs with lower value of sensitivity indices against a certain cut-off threshold value. The cut-off threshold (CT) is defined per series of indices of one ED series and its value is between the minimum and maximum values of the range of those index values as in equation 9-1.

$$CT = Min(SI_{ED}) + CR * (Max(SI_{ED}) - Min(SI_{ED})) \qquad \textbf{(9-1)}$$

where CR is called the Cut-off Ratio and $0 \le CR \le 1$. For example, if CR is 0.5, or 50%, then the values of four cut-off thresholds are all the middle of the range of their associated sensitivity indices. Figure 9-22 and Figure 9-23 show normalised sensitivity indices of ED series RUGR1 and RUGR3-A respectively. On both charts the minimum available value of normalised sensitivity index is 0, and the maximum is 1. Therefore, the value of CT on both series are 0.5 (the green dashed line). According to the chart in Figure 9-22, seven TDs are considered as the least important for the ED series RUGR1 (red bars). From Figure 9-23, ten TDs are considered least important for the ED series RUGR3A. Three TDs, i.e. LS302, LS303, and LS304, are commonly among the least important TDs for both ED series.

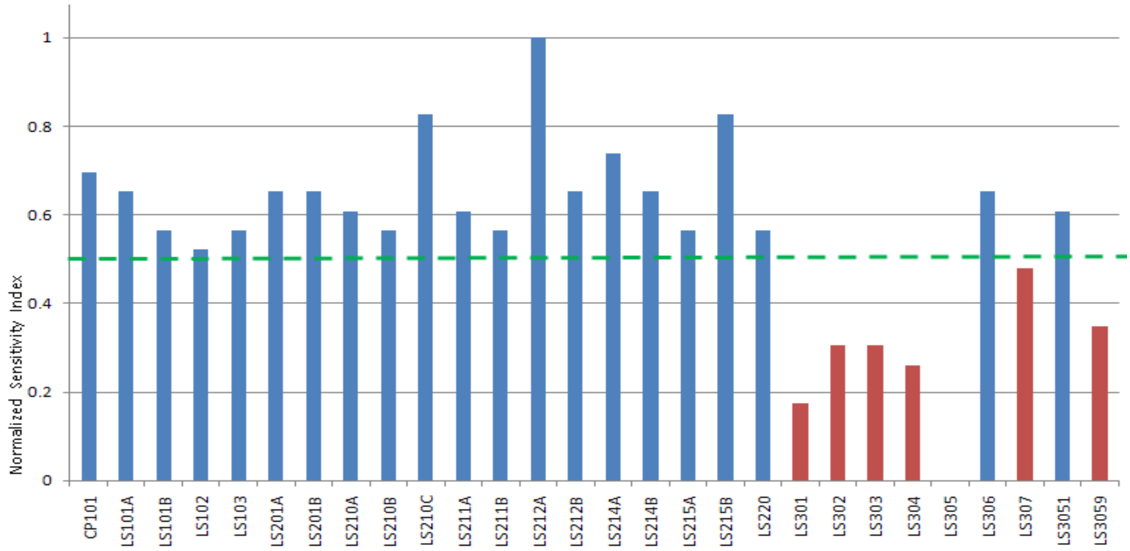


Figure 9-22 Normalised sensitivity indices of RUGR1 with respect to 28 TDs

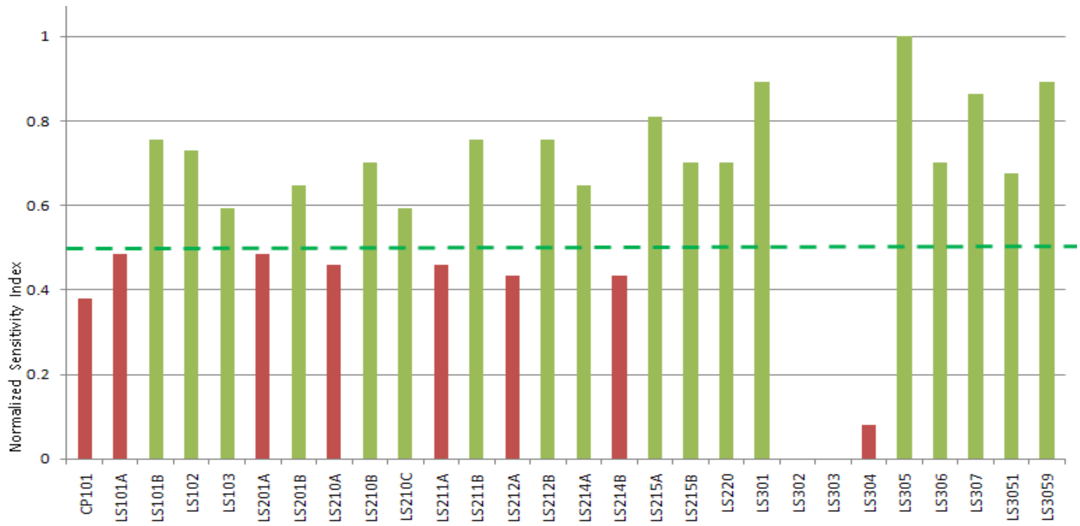


Figure 9-23 Normalised sensitivity indices of RUGR3-A with respect to 28 TDs

If a TD series fails examination of its sensitivity index against all ED series, then it is reported as a less important TD in the system and may be decided to be filtered out, or sampled and processed less often. Figure 9-24 shows the percentage of less important TDs based on different values of Cut-off Ratio (CR) which are shown as a percentage value in the first row of the horizontal axis. Obviously, the higher the CR, the more TDs are filtered out. This means that if it was decided to sample the filtered TDs less often than the other ones, lower amount of computational overhead would be expected.

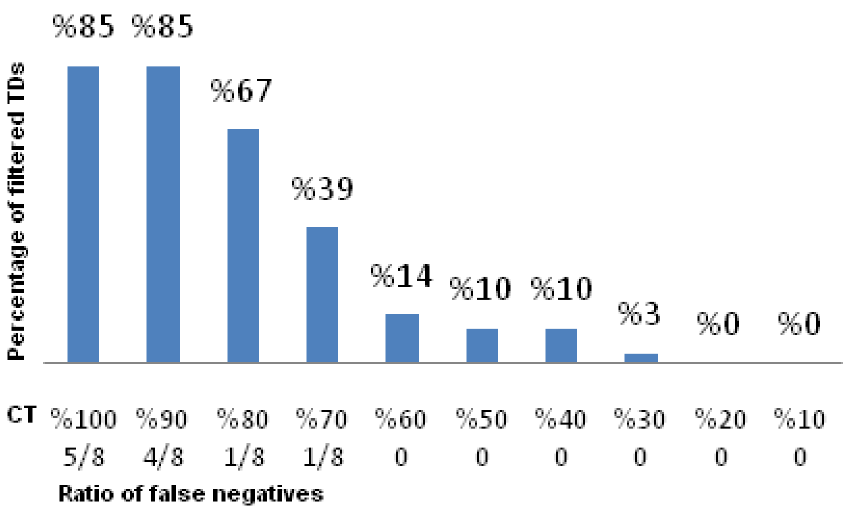


Figure 9-24 Proportion of reported less important TDs per CR and ratio of false negative on EvenTracker method

The lower row of numbers on the horizontal axis in Figure 9-24 represents the number of TDs which are reported less important (because they failed the cut-off threshold), but according to the structure of the model are more important for an ED. Therefore, these figures show the number of false negatives resulted from EvenTracker method. For example when 67% of TDs are filtered out, 1 out of 8 more important TDs is reported less important, or 12.5% false negative. Obviously this is not a very attractive resolution. Instead, at a safest case in terms of maintaining truly important TDs, when CR is 60%, 14% of the TDs are reported less important none of which are truly more important (i.e. zero false negative). The reported more important and less important TDs with 60% CR are listed in Appendix D.

9.6.1. Sensitivity of EvenTracker to the method parameters

In order to evaluate the degree of dependency of EvenTracker to the assumption parameters of the method, sensitivity indices resulting from different values of ET and TT as well as SS were compared. Figure 9-25 shows the percentages of less important TDs based on different values of ET and TT on different levels of CR. Figure 9-26 shows the percentages of less important TDs based on different values of SS on different levels of CR. It appears on Figure 9-25 that ET and TT values do not make a significant difference on the indices. However, SS values show influence according to

Figure 9-26. Focusing on the region of no false negatives in Figure 9-26, the three thicker line charts with maximum 30% CR, recommend use of SS size not shorter than 2 seconds of data and not longer than 8 seconds for achieving the highest savings in computational overhead.

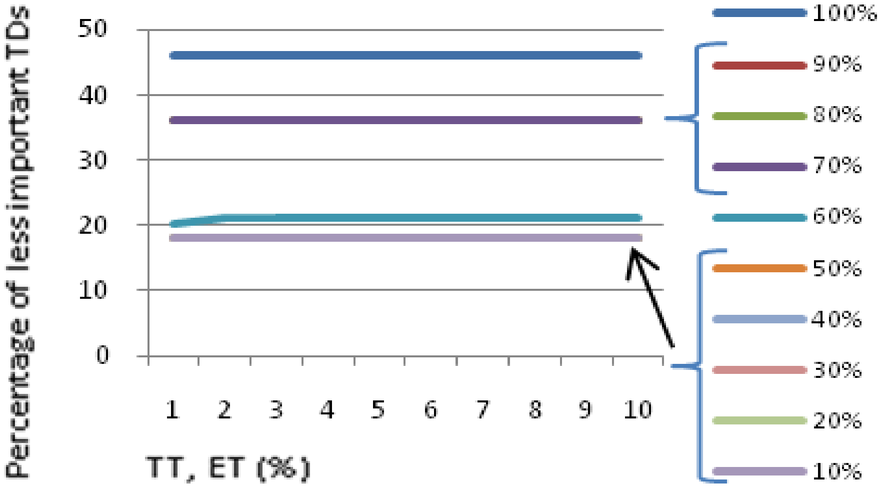


Figure 9-25 Percentages of less important TDs per percentage of ET and TT with different CR values

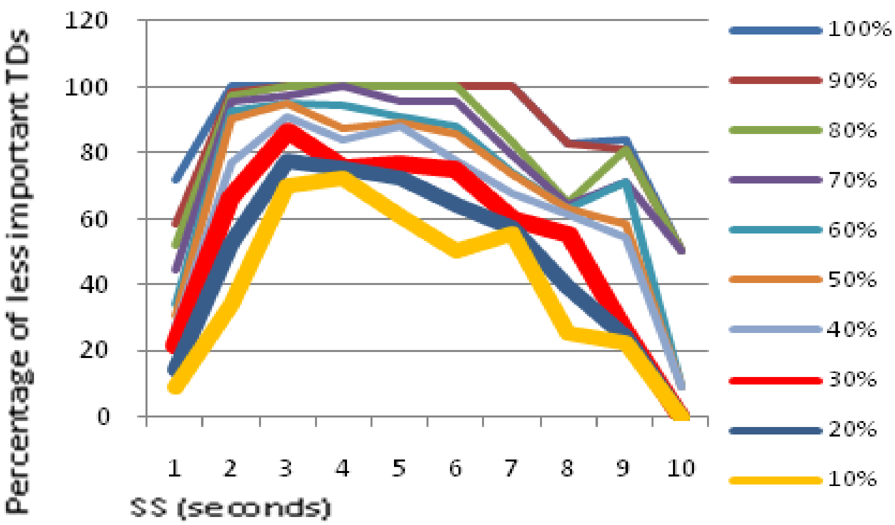


Figure 9-26 Percentages of less important TDs per percentage of TS with different CR values

9.6.2. EvenTracker after input variable elimination

24 reported more important TDs according to the list in Appendix D were selected to be sampled by the model for generating ED series. These generated EDs were compared against the previously generated EDs from the

model with all 28 sampled TDs. Figure 26 in Appendix E shows that no difference appeared between each pair of new and old ED data series. EvenTracker algorithm spent 6.875 seconds to analyze 28 TDs and 4 EDs, whereas it has spent 3.5625 seconds for 24 TDs and 4 EDs. The average CPU usage remained almost at the same level during the period of analysis. Therefore, a time efficiency of nearly 52% has been gained. Of course, by ignoring the samples of reportedly less important TDs, efficiency in the amount of data communication and data buffering could also be expected.

9.7. A Comparison between EvenTracker and Entropy-based Sensitivity Analysis (ESA) methods

To highlight the advantages of EvenTracker sensitivity analysis method a comparison between this method and Entropy-based Sensitivity Analysis (ESA) method was undertaken. For the reasons of ANOVA techniques reliance on historical data and homoscedasticity (Xu et al, 2008; Braddock et al, 2006; De Pauw et al, 2008), the applicable technique that can be compared with the proposed technique would be ESA. EvenTracker method handles large number of data points with increased computational efficiency without compromising accuracy and quality of sensitivity analysis.

An Entropy-based sensitivity analysis method had been proposed by Krzykacz-Hausmann (2001). In this method sensitivity index of a model output with respect to a model input is defined as the amount of reduction in the entropy of the output if the input does not have any uncertainty, i.e. its values are all known. Further details of the method could be found at (Krzykacz-Hausmann, 2001). Although this method, like ANOVA-based methods needs analytical determination of density functions of input and output series, Krzykacz-Hausmann (2001) proposed a method for estimation of the value of sensitivity index from samples. The ESA estimation method is implemented in this work for performance analysis in comparison to EvenTracker method.

ESA method has been adapted and applied to the same data series as used for EvenTracker algorithm. Results appeared in Figure 9-27 and Figure 9-28 with similar formats as Figure 9-21 and Figure 9-24, respectively. It appears from Figure 9-27 that on average ESA method filters out more TDs with different CR values. However, the ratio of false negatives on the same figure shows that ESA method produces more false negatives for most of CR values. EvenTracker and ESA methods are compared in Figure 9-29 based on the region of ‘no false negatives’. Where EvenTracker method reports up to 14% of TDs as less important without any false negative, ESA method produces 37.5% false negatives (i.e. 3 out of 8 or 3/8).

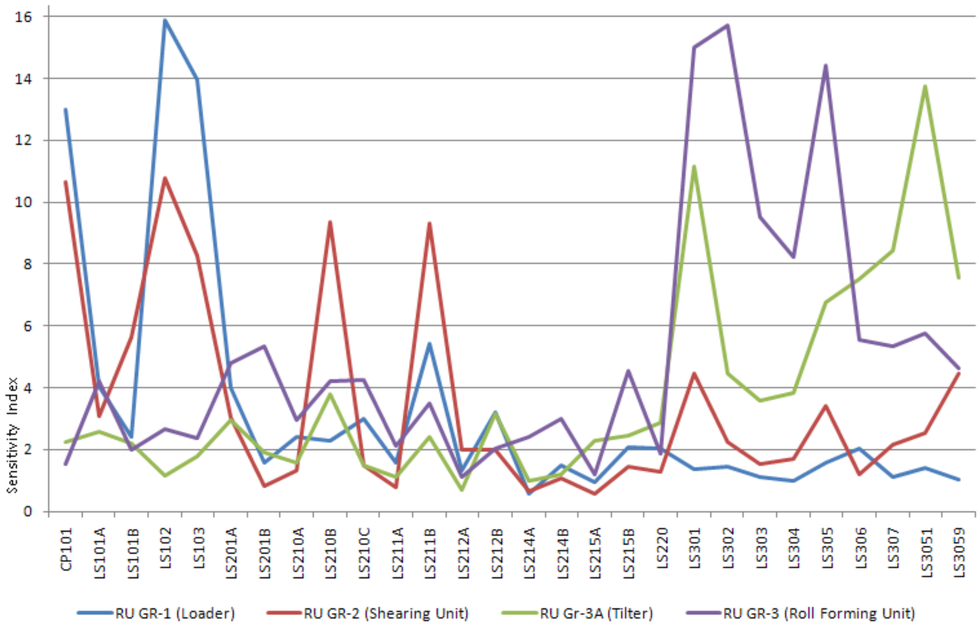


Figure 9-27 ESA sensitivity indices of 4 EDs with respect to 28 TDs

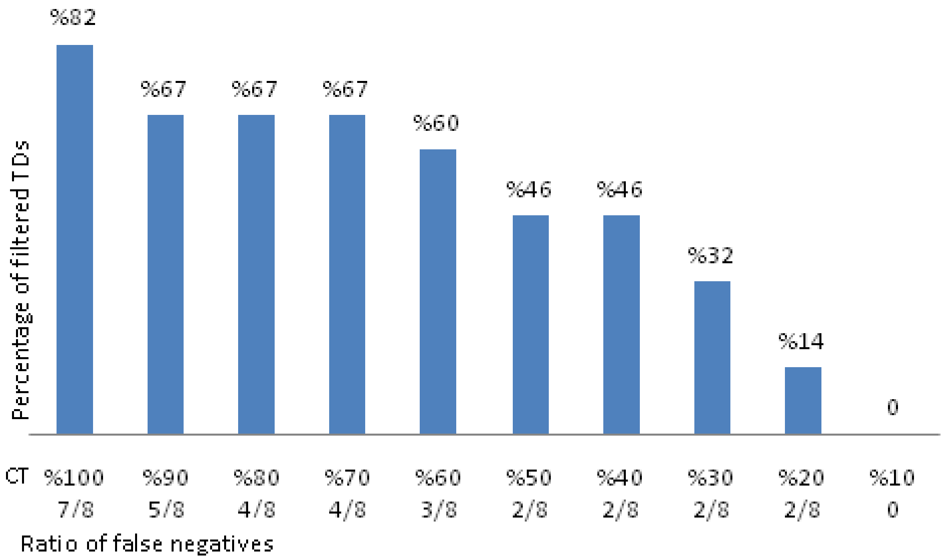


Figure 9-28 Proportion of reported less important TDs per CR and false negative ratio on ESA method

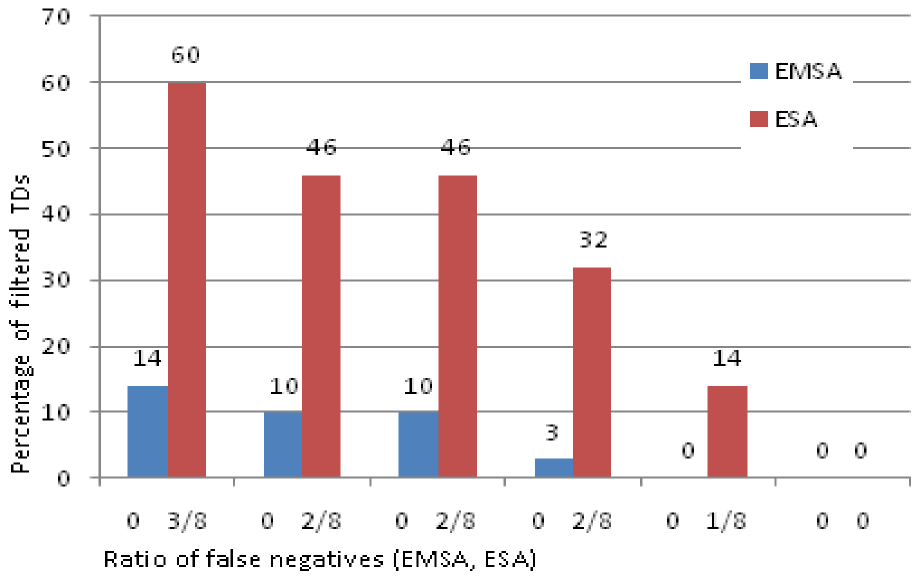


Figure 9-29 Comparison of proportion of less important TDs with low false negative ratios on EvenTracker and ESA methods

By comparing the levels of CPU usage between the two sensitivity analysis methods, it was observed that the ESA method continuously took an average of up to 50% of CPU effort for the duration of 1348.87 seconds. EvenTracker method utilised the CPU at nearly 55%, but for a much shorter time of 6.875 seconds.

With a typical sampling rates as high as five samples per second on ED and TD series, ESA method shows a lesser computational efficiency compared to EvenTracker in dealing with real-time analysis.

9.8. Summary of empirical study

The empirical study in this chapter discussed feasibility of novel EvenTracker approach for sensitivity analysis in real-time data acquisition and performance modelling systems. Application of EvenTracker on a production plant with 28 input variables and four performance factors as model outputs correctly resulted 14% of the input variables to be unimportant for evaluation of model outputs. The method proved a time efficiency gain of 52% on the analysis of filtered system when unimportant input variables were not sampled anymore.

Experiments on the sensitivity of EvenTracker to its own parameters showed that the length of search slot i.e. time interval of searching for data level transition has more significant role than the transition level. Time intervals between 2 to 8 seconds were recommended.

The method compared to entropy-based sensitivity analysis technique as the only other method that can be used for real-time purposes is quicker, more accurate and less computationally burdensome.

10. Conclusion, contribution, and future work

The purpose of this research work was to demonstrate the feasibility of creating a quick response decision platform for middle management in industry. It utilises the strengths of current, but more importantly creates a leap forward in the theory and practice of Supervisory and Data Acquisition (SCADA) systems and Discrete Event Simulation and Modelling (DESM). The proposed research platform uses real-time data and creates an automatic platform for real-time and predictive system analysis, giving current and ahead of time information on the performance of the system in an efficient manner.

10.1. Contribution to knowledge

In short, this research has contributed to the following features of data integration systems;

1. Flexible data source definition in data acquisition interface,
2. Flexible information evaluation from data source combination,
3. Real-time connection of shop floor data to the simulated model,
4. Efficient variable importance evaluation,

In brief, above contributions have been made by the following activities and achievements;

1. In this work the importance of a comprehensive environment for design and development of data interface units and data aggregation units was emphasised. The shortcomings of the existing data acquisition and integration systems in terms of data input management were highlighted. Flexible Data Input Layer Architecture (FDILA) was proposed to fill the existing gap in current shop floor data management system. The proposed architecture enables the system developers (line managers) to define a wide number and variety of data entry points and data types in a data integration system no longer requiring proprietary firmware system.

2. A generic framework for real-time discrete event simulation has been proposed. Starting with a review of existing applications of real-time simulation in manufacturing, the concept of this framework has been presented and all the major components of the framework have been

explained. Four drawbacks of traditional simulation have been identified which the proposed framework effectively addressed.

The solution consisted of four interrelated layers.

- First, the Data Definition Layer (DDL), where different types of signals from sensors and data sources are collected.
- Second, the Variable Definition Layer (VDL), where latent variables are evaluated from acquired data sources based on user defined functions.
- Third, the evaluated variables at VDL are then transferred to the Performance Definition Layer (PDL), where a simulation model represents the flow of processes and outputs performance indicators.
- Forth, 'what-if' scenario is defined in this layer and together with estimated distribution attributes of variables are applied to the simulation model.

The implemented platform was used as a framework for data integration and implementation of the novel EvenTracker sensitivity analysis method (SAM).

3. EvenTracker SAM has been proposed and introduced and the algorithm and the key assumptions of the method have been discussed. Optimum values for the parameters of EvenTracker method based on an industrial case study were suggested. EvenTracker method was compared against entropy-based epistemic SAM (Krzykacz-Hausmann, 2001) in terms of their execution time and overall volume of input variables that could be efficiently handled by each method. It has been declared that the major shortcoming of traditional SAMs is in their heavy reliance on historical data, heavy computational overhead, or expert input which makes quick response (i.e. real-time) decision making impossible.

10.2. Quick reference to objectives and the steps taken

(1) To capture and to allow for visualisation of the randomness of shop floor day-to-day operations:

How?

- Created novel techniques in data acquisition and models for fusing relative data and their associated performance functions,
- Designed experiments which caused little to no interruptions on to the daily activities of our industrial partners, validated and verified the models against the real system,
- Designed and developed the data modelling and information and communication system architecture which is flexible enough to cater for needs of the industry,

(2) Tested and validated a fully flexible data input layer architecture (FDILA) that was capable of integrating multiple platform ICT infrastructure and equipment, thus overcoming cross platform anomalies in the data management systems:

How?

- Developed the necessary and customised user interfaces, real-time databases, communication and data construction models on laboratory scale,
- Implemented standard communication routes and protocols to ensure seamless flow of signals/data/information along the software hardware architecture,

(3) Introduced a flexible and simple mobile systems monitoring tool, "*fitting in a laptop case*" that can be easily installed and adapted to the environment that is being used in with minimum reliance on system supplier:

How?

- Investigated and introduced off-the-shelf cost effective data acquisition equipment (where possible cost effective wireless enabled devices),
- Utilised the standard operation systems and software development toolkits to minimise development and maintenance conflicts,

(4) Tested and validated the system in a major fast moving consumer goods industry to prove the capability and the advantages of the proposed system against existing SCADA, organisation integration platforms and traditional systems modelling and simulation techniques:

How?

– Once the experiments were designed liaised with partner manufacturers to allow access to the sites for data collection and analysis,

– Tested and checked the actual results against the expected and made the necessary adjustments.

10.3. Future work

There is space for expansion of this work on almost every aspects including but not limited to;

- The EvenTracker approach in sensitivity analysis can be used to run large scale distributed data analysis in which local filters will reduce the number of systems analysis parameters and join the global solution for quick response to unexpected events. For example this approach can be used for meteorological and climate change analysis, large scale global manufacturing/logistics operations or interlinked financial systems.

- Code for distribution fitting has been designed and developed with a limited scope. A more sophisticated algorithm is foreseen for the future of this work.

- Producing industry based objective function models where Key Performance Indicators (KPIs) can be combined to provide overall performance of the system facilitating future optimization approaches.

- Although the proposed work focused on applications where raw data is acquired from shop floor level in second-scale real-time situations and high level information is a few performance factors, the generic platform may be applied to fill the gap between other levels of information management and time scale. For example, when satisfactory profits are combined and adopted as the organization's goal, one may implement the proposed platform to acquire material requirements planning (MRP) and enterprise resource planning (ERP) level data and translate them into factors such as profit stability, market share increase, product diversification, price stability, worker morale improvement, and so on (Hillier et al, 2004). As another example, with the advent of nanotech technology, it will not be so far from integration of millions of tiny sensors, data processing units and actuators in superhuman bodies (Jones, 2008). Energy efficiency requirements at this scale of processing applications will demand careful and quick selection of processing inputs.

- Provision of a shop-floor process data acquisition platform for other projects; The designed and developed system has provided a flexible and real-time data acquisition and simulation modelling platform for several other projects of postgraduate students (Wairatpanij, 2009; Purewal, 2009; Sanchez, 2008; Daniel, 2008; Riley, 2007) involved real-time simulation, track and traceability, system reliability, and measurement of process parameters of a model. This platform is sought to help postgraduate students of Advanced Manufacturing Systems course in their research and development work.

10.4. Overall benefits of the research outcome

This research solution will allow for primitive to complex systems to translate key data into performance functions. It will provide the middle management the opportunity to visualise and intervene on the daily activities of their operations and reduce the hidden cost of *Orthodox Thinking and Approach.* The predictive feature of the proposed solution will allow initiatives to be aired and assessed freely, with no actual disruption to the operations. Confidence in management will increase and the risks will be reduced, adaptation will turn into aggressive prediction, learning and positive change. This will lead to the mutation-evolution of existing man-made complex systems to viable systems.

References

ABB 2010, “System 800xA Information Management Historian” *[Homepage of ABB], [Online]. Available: http://www.abb.com/product/seitp334/263fd06e86e19ad18525719a0044237a.aspx?productLanguage=us&country=00&tabKey=2* [2010, 02/10].

Acheroy, M. 1999, 27/01/1999-last update*, “*Introduction to Data Fusion” *[Homepage of Signal and Image Centre (SIC), Electrical Engineering Department of the Faculty of Applied Sciences of the Royal Military Academy], [Online]. Available: http://www.sic.rma.ac.be/Research/Fusion/Intro/* [2009, 08/06].

Aimin, L., Li, H. & Xueshan, L. 2008, "Entropy-based complexity analysis for architecture of information system", *Proceedings of 2008 IEEE International Conference on Networking, Sensing and Control, ICNSC*, pp. 1734.

Ambrosetti, A. & Malchiodi, A. 2007, “Nonlinear analysis and semilinear elliptic problems”*, Cambridge University Press, Cambridge, UK ; New York*.

Annis, C. 2008, 01/01/2008-last update*, “*Correlation” *[Homepage of statisticalengineering.com], [Online]. Available: http://www.statisticalengineering.com/correlation.htm* [2009, 08/13].

Apostol, T.M. 1974, “Mathematical analysis”*, 2nd edn, Addison-Wesley*.

APPLIED MATERIALS 2010*, “*Applied AutoMod” *[Homepage of APPLIED MATERIALS], [Online]. Available: http://www.appliedmaterials.com/products/automod_2.html* [2010, 05/23].

Askin, R.G. & Standridge, C.R. 1993, “Modeling and analysis of manufacturing systems”*, Wiley, New York, London*.

Automation & Control Solutions, Process Solutions, Honeywell 2009, “Uniformance PHD Transforming Data into Knowledge”, *Honeywell International Inc., Phoenix, AZ, USA*.

Banks, J. 2001, “Discrete-Event System Simulation”*, 3rd edn, Prentice Hall, Upper Saddle River, NJ, USA*.

Banks, J. 1998, “Handbook of simulation: principles, methodology, advances, applications, and practice”*, Wiley, New York, USA*.

Banks, D.L., Olszewski, R.T. & Maxion, R.A. 2003, "Comparing methods for multivariate nonparametric regression", *Communications in Statistics Part B: Simulation and Computation,* vol. 32, no. 2, pp. 541-571.

Bansiya, J., Davis, C. & Etzkorn, L. 1999, "An entropy-based complexity measure for object-oriented designs", *Special issue on selected papers from ESEC/FSE'97,* vol. 5, no. 2, pp. 111-118.

Bauer, A.e.a. 1994, "Shop floor control systems: from design to implementation"*, 1st edn, Chapman and Hall, London*.

Benadjaoud, G.N. & David, B.T. 1997, "An extensible reference model for data integration", *The second IFIP TC5/WG5.3/WG5.7 international conference on Information infrastructure systems for manufacturing*, eds., Chapman and Hall, London, UK, pp. 206.

Beylkin, G., Kurcz, C. & Monzón, L. 2008, "Fast algorithms for Helmholtz Green's functions", *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences,* vol. 464, no. 2100, pp. 3301-3326.

Blum, A.L. & Langley, P. 1997, "Selection of relevant features and examples in machine learning", *Artificial Intelligence,* vol. 97, no. 1-2, pp. 245-271.

Borgonovo, E. & Peccati, L. 2007, "Global sensitivity analysis in inventory management", *International Journal of Production Economics,* vol. 108, no. 1-2, pp. 302-313.

Boyer, S.A. 1999, "SCADA: supervisory control and data acquisition"*,* 2nd edn, *Instrument Society of America, Research Triangle Park, N.C., USA*.

Braddock, R.D. & Schreider, S.Y. 2006, "Application of the Morris algorithm for sensitivity analysis of the REALM model for the Goulburn irrigation system", *Environmental Modeling and Assessment,* vol. 11, no. 4, pp. 297-313.

Brailsford, S.C. 2007, "Tutorial: Advances and challenges in healthcare simulation modeling", *Proceedings - Winter Simulation Conference,* pp. 1436.

Brodersen, K.H., Haiss, F., Ong, C.S., Jung, F., Tittgemeyer, M., Buhmann, J.M., Weber, B. & Stephan, K.E. 2010, "Model-based feature construction for multivariate decoding", *NeuroImage*.

Buchenneder, K. 2007, "Processing of myoelectric signals by feature selection and dimensionality reduction for the control of powered upper-limb prostheses*", Lecture Notes in Computer Science (including subseries*

Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-38449120756&partnerID=40&md5=d0035422173fb3941a3905e9e3e13d65.

Bunke, H. & Riesen, K. 2010, "Improving vector space embedding of graphs through feature selection algorithms", *Pattern Recognition*. *Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-77953078185&partnerID=40&md5=2fe02081a72afabb36b576444fb8e630*

Buonomo, A. & Lo Schiavo, A. 2010, "Nonlinear distortion analysis via perturbation method", *International Journal of Circuit Theory and Applications,* vol. 38, no. 5, pp. 515-526.

Byon, Y.-., Abdulhai, B. & Shalaby, A. 2009, "Real-time transportation mode detection via tracking global positioning system mobile devices", *Journal of Intelligent Transportation Systems: Technology, Planning, and Operations,* vol. 13, no. 4, pp. 161-170.

Casale, M., Zunin, P., Cosulich, M.E., Pistarino, E., Perego, P. & Lanteri, S. 2010, "Characterisation of table olive cultivar by NIR spectroscopy", *Food Chemistry,* vol. 122, no. 4, pp. 1261-1265.

Cecelja, F. 2002, “Manufacturing information and data systems : analysis, design and practice”*, Penton, London*.

Chang, T., Wysk, R.A. & Wang, H. 2006, “Computer-aided manufacturing”, 3rd edn, Pearson Prentice Hall, Upper Saddle River, NJ.

Charaniya, S., Le, H., Rangwala, H., Mills, K., Johnson, K., Karypis, G. & Hu, W.-. 2010, "Mining manufacturing data for discovery of high productivity process characteristics", *Journal of Biotechnology*.

Chen, F.-. & Li, F.-. 2010, "Combination of feature selection approaches with SVM in credit scoring", *Expert Systems with Applications,* vol. 37, no. 7, pp. 4902-4909.

Cheng, K. 2005, “E-manufacturing : fundamentals and applications”*, Wit, Southampton*.

Cheng, K., Webb, D. & Marsh, R. 2004, “Advances in e-engineering and digital enterprise technology – I” *: proceedings of the Fourth International Conference on e-Engineering and Digital Enterprise Technology (e-ENGDET), Leeds Metropolitan University, UK, 1-3 September 2004,* Professional Engineering, Bury St Edmunds.

Chiesa, R. & Pennasilico, A.L.R. 2007, “SCADA (in) Security: Hacking Critical Infrastructures*”*, *Critical Infrastructures Security Testing and Analysis Lab, Kualalampour, Malasia*.

Choucri, N. 1999, *“*Using GSSD” *[Homepage of Global System for Sustainable Development, Massachusetts Institute of Technology], [Online]. Available: http://gssd.mit.edu/GSSD/gssden.nsf/Default%20Frameset?OpenFrameSet&Frame=Right&Src=%2FGSSD%2Fgssden.nsf%2Fstructurecontentview%2FusingGSSD%3FOpenDocument%26AutoFramed* [2008, 02/12].

Clement, J., Coldrick, A. & Sari, J. 1995, “Manufacturing data structures: building foundations for excellence with bills of materials and process information”*, O. Wight Ltd. Publications, Essex Junction, VT*.

Cloke, H.L., Pappenberger, F. & Renaud, J.-. 2008, "Multi-Method Global Sensitivity Analysis (MMGSA) for modelling floodplain hydrological processes", *Hydrological Processes,* vol. 22, no. 11, pp. 1660-1674.

Cobham, A. 1965, "The intrinsic computational difficulty of functions", *International Congress for Logic, Methodology, and Philosophy of Sciences*Y. Bar-Hellel, Ed., North Holland, Amsterdam, pp. 24.

Cobus, S. 2003, “Practical electrical network automation and communication systems”*, Newnes, Burlington, MA, USA*.

Cook, S.A. 1983, "An Overview of Computational Complexity", *Communications of the ACM,* vol. 26, no. 6, pp. 408.

Cukier, R.I., Levine, H.B. & Shuler, K.E. 1978, "Nonlinear sensitivity analysis of multiparameter model systems", *Journal of Computational Physics,* vol. 26, no. 1, pp. 1-42.

Dangelmaier, W., Mahajan, K.R., Seeger, T., Klöpper, B. & Aufenanger, M. 2006, "Simulation assisted optimization and real-time control aspects of flexible production systems subject to disturbances", *Proceedings - Winter Simulation Conference,* pp. 1785.

De Pauw, D.J.W., Steppe, K. & De Baets, B. 2008, "Unravelling the output uncertainty of a tree water flow and storage model using several global sensitivity analysis methods", *Biosystems Engineering,* vol. 101, no. 1, pp. 87-99.

Dias, N.S., Kamrunnahar, M., Mendes, P.M., Schiff, S.J. & Correia, J.H. 2009, "Variable subset selection for brain-computer interface: PCA-based dimensionality reduction and feature selection", *Proceedings of the 2nd*

International Conference on Bio-Inspired Systems and Signal Processing, pp. 35.

Duffy, D.G. 2003, “Advanced engineering mathematics with MATLAB”, *2nd edn, Chapman & Hall/CRC Press, Boca Raton, FL*.

Dumiak, M. 2008, “Magnetic Field Sensors Could Help Halt Runway Crashes”, *IEEE, New York, USA*.

Dunker, A.M., Yarwood, G., Ortmann, J.P. & Wilson, G.M. 2002, "The decoupled direct method for sensitivity analysis in a three-dimensional air quality model - Implementation, accuracy, and efficiency", *Environmental Science and Technology,* vol. 36, no. 13, pp. 2965-2976.

Durkee, D.P., Pohl, E.A. & Mykytka, E.F. 1998, "Sensitivity analysis of availability estimates to input data characterization using design of experiments", *Quality and Reliability Engineering International,* vol. 14, no. 5, pp. 311-317.

Eldabi, T., Paul, R.J. & Young, T. 2007, "Simulation Modeling in Healthcare: Reviewing legacies and investigating futures", *Journal of the Operational Research Society,* vol. 58, pp. 262-270.

Eren, H., Ph. D. 2006, “Wireless sensors and instruments : networks, design, and applications”*, CRC, London, UK*.

Faghihi, F. & Hadad, K. 2004, "Numerical solutions of coupled differential equations and initial values using Maple software", *Applied Mathematics and Computation,* vol. 155, no. 2, pp. 563-572.

Ferreira, J.J.P. 2004, “E-manufacturing : business paradigms and supporting technologies”*, 18th International Conference on CADCAM, Robotics, and Factories of the Future (CARs & FOF), July 2002, Porto, Portugal,* Kluwer Academic Publishers, Boston ; Great Britain.

Ferrigno, L., Paciello, V. & Pietrosanto, A. 2005, "A bluetooth-based proposal of instrument wireless interface", *IEEE Transactions on Instrumentation and Measurement,* vol. 54, no. 1, pp. 163-170.

Flexsim Software Products, I. 2010*, “*Flexsim” *[Homepage of Flexsim Software Products, Inc.], [Online]. Available: http://www.flexsim.com/products/flexsim/* [2010, 05/23].

Folino, G., Spezzano, G. & Talia, D. 1998, "Performance evaluation and modeling of MPI communications on the meiko CS-2" in *High-*

Performance Computing and Networking Springer, Berlin / Heidelberg, pp. 932-936.

Gao, H., Mandal, M.K. & Wan, J. 2010, "Classification of hyperspectral image with feature selection and parameter estimation*", International Conference on Measuring Technology and Mechatronics Automation, ICMTMA,* pp. 783.

Gaweda, A.E., Zurada, J.M. & Setiono, R. 2001, "Input selection in data-driven fuzzy modeling", *IEEE International Conference on Fuzzy Systems,* pp. 1251.

Gennick, J. 1999, “Oracle SQL*Plus : the definitive guide”*, O'Reilly, Sebastopol, CA, USA*.

Goossenaerts, J. & Acebedo, C.M. 1997, "Information infrastructure services for small and medium size manufacturers: the MI2CI project", *The second IFIP TC5/WG5.3/WG5.7 international conference on Information infrastructure systems for manufacturing*, pp. 371.

Goossenaerts, J., Wortmann, J.C. & Kimura, F. (eds) 1997, “Information Infrastructure Systems for Manufacturing”, *1st edn, Chapman and Hall, London, UK*.

Goossenaerts, J.B.M. 1997, "A framework for connecting work and information infrastructure", *The second IFIP TC5/WG5.3/WG5.7 international conference on Information infrastructure systems for manufacturing*, pp. 193.

Griewank, A. & Walther, A. 2008, “Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation”*, 2nd edn, Society for Industrial and Applied Mathematic (SIAM)*.

Gunasekaran, S. & Revathy, K. 2010, "Content-based classification and retrieval of wild animal sounds using feature selection algorithm", *The 2nd International Conference on Machine Learning and Computing*, pp. 272.

Gupta, A.K., Sivakumar, A.I. & Sarawgi, S. 2002, "Shop floor scheduling with simulation based proactive decision support", *Winter Simulation Conference Proceedings*, pp. 1897.

Guyon, I. & Elisseeff, A. 2003, "An introduction to variable and feature selection", *The Journal of Machine Learning Research,* vol. 3, pp. 1157-1182.

Hand, D.J., Mannila, H. & Smyth, P. 2001, "Principles of data mining"*, MIT Press, Cambridge, Mass.; London*.

Hannam, R.G. 1996, "Computer integrated manufacturing: from concepts to realization"*, Addison-Wesley, Harlow*.

Haritsa, J.R., Kotagiri, R., Pudi, V. & SpringerLink 2008, "Database Systems for Advanced Applications", *Springer-Verlag Berlin Heidelberg, Berlin, Heidelberg*.

Harmonosky, C.M. 1995, "Simulation-based real-time scheduling: review of recent developments", *Winter Simulation Conference Proceedings*, pp. 220.

Harmonosky, C.M. 1990, "Implementation issues using simulation for real-time scheduling, control, and monitoring", *Winter Simulation Conference Proceedings*, pp. 595.

Harrison, W. 1992, "An entropy-based measure of software complexity", *IEEE Transactions on Software Engineering,* vol. 18, no. 11, pp. 1025-1029.

He, G. & Liao, W., 2010, "Feature construction method of combined subdivision surface", *Jisuanji Jicheng Zhizao Xitong/Computer Integrated Manufacturing Systems, CIMS,* vol. 16, no. 3, pp. 507-512.

Hillier, F.S. & Lieberman, G.J. 2004, "Introduction to operations research"*, 8th edn, McGraw-Hill, New York ; London*.

Honeywell International Inc., "Advanced Formula Manager" *[Homepage of Honeywell Internatonal Inc.], [Online]. Available: http://hpsweb.honeywell.com/Cultures/en-US/Products/OperationsApplications/InformationManagement/AdvancedFormulaManager/default.htm* [2009, 01/25].

Honeywell International Inc., "Event Monitoring" *[Homepage of Honeywell International Inc.], [Online]. Available: http://hpsweb.honeywell.com/Cultures/en-US/Products/OperationsApplications/OperationsManagement/EventMonitoring/default.htm* [2009, 01/25].

Honeywell International Inc., "Information Management" *[Homepage of Honeywell International Inc.], [Online]. Available: http://hpsweb.honeywell.com/Cultures/en-US/Products/OperationsApplications/InformationManagement/default.htm* [2009, 01/25].

Honeywell International Inc., *"*Uniformance PHD" *[Homepage of Honeywell International Inc.], [Online]. Available: http://hpsweb.honeywell.com/Cultures/en-US/Products/OperationsApplications/InformationManagement/UniformancePHD/default.htm* [2009, 01/25].

Horowitz, P. & Hill, W. 1989, "The art of electronics"*, 2nd edn, Cambridge University Press, Cambridge*.

Hsu, C. 1994, "Manufacturing information systems", *Handbook of Design, Manufacturing, and Automation*, eds. R.C. Dorf & A. Kusiak, John Wiley and Sons, New York, USA, pp. 737-766.

Huang, X. & Wang, W. 2010, "Electroencephalography based feature selection for multi-intelligence activity", *2nd International Workshop on Education Technology and Computer Science, ETCS 2010*, pp. 808.

Ibrahim, I.K., Kronsteiner, R. & Kotsis, G. 2005, "A semantic solution for data integration in mixed sensor networks", *Computer Communications,* vol. 28, pp. 1564-1574.

IEEE Standards Association 2008, 2010-last update*, "*IEEE 802.3: LAN/MAN CSMA/CDE (ETHERNET) ACCESS METHOD" *[Homepage of IEEE], [Online]. Available: http://standards.ieee.org/getieee802/802.3.html* [2010, 13/02].

Isukapalli, S.S. 1999, "Uncertainty Analysis of Transport-Transformation Models", *New Burnswick Rutgers, The State University of New Jersey, New Jersey*.

Iyengar, S.S., Jayasimha, D.N. & Nadig, D. 1994, "A versatile architecture for the distributed sensor integration problem", *IEEE Trans. Computers,* vol. 43, no. 2, pp. 175-185.

Jain, A.K. 2010, "Data clustering: 50 years beyond K-means", *Pattern Recognition Letters,* vol. 31, no. 8, pp. 651-666.

James, G. 2004, "Advanced modern engineering mathematics"*, 3rd edn, Pearson Prentice Hall, Harlow*.

Jin, Y., Yue, H., Brown, M., Liang, Y. & Kell, D.B. 2007, "Improving data fitting of a signal transduction model by global sensitivity analysis", *Proceedings of the American Control Conference,* pp. 2708.

Joliffe, I.T. 2002, "Principal component analysis"*, 2nd edn, Springer*.

Jones, A. & Saleh, A. 1990, "A Multi-level/Multi-layer Architecture for Intelligent Shop floor Control", *International Journal of Computer Integrated Manufacturing,* vol. 3, no. 1, pp. 60-70.

Jones, R.A.L. 2008, “Rapturing the Nanotech Rapture”, *IEEE, New York.*

Joshi, S.B. & Smith, J.S. 1994, “Computer control of flexible manufacturing systems : research and development”*, Chapman & Hall, London, UK*.

Kamen, E.W. 1999, “Introduction to industrial controls and manufacturing”*, Academic Press, San Diego, Calif. London*.

Kang, P., Lee, H., Cho, S., Kim, D., Park, J., Park, C. & Doh, S. 2009, "A virtual metrology system for semiconductor manufacturing", *Expert Systems with Applications,* vol. 36, no. 10, pp. 12554-12561.

Katzy, B.R. & Riggers, B. 1997, "Value system redesign: system oriented management and enterprise integration in globally distributed manufacturing networks", *The second IFIP TC5/WG5.3/WG5.7 international conference on Information infrastructure systems for manufacturing* , eds. pp. 221.

Kelton, W.D. 2009, “Simulation with Arena”*, 5th edn, McGraw-Hill Higher Education*.

Kim, E., Lee, S., Kwon, K. & KIm, S. 2010, "Feature construction scheme for efficient intrusion detection system", *Journal of Information Science and Engineering,* vol. 26, no. 2, pp. 527-547.

Kirianaki, N.V. 2002, “Data acquisition and signal processing for smart sensors*, J. Wiley, Chichester ; New York*.

Kohavi, R. & John, G.H. 1997, "Wrappers for feature subset selection", *Artificial Intelligence,* vol. 97, no. 1-2, pp. 273-324.

Komashie, A. & Mousavi, A. 2005, "Modeling emergency departments using discrete event simulation techniques", *Proceedings - Winter Simulation Conference*, pp. 2681.

Komashie, A., Mousavi, A. & Gore, J. 2008, "Using Discrete Event Simulation (DES) to manage theatre operations in healthcare: An audit-based case study", *Proceedings - UKSim 10th International Conference on Computer Modelling and Simulation, EUROSIM/UKSim2008,* pp. 360.

Kosanke, K., Vernadat, F.B. & Zelm, M. 1997, "CIMOSA process model for enterprise modelling", *The second IFIP TC5/WG5.3/WG5.7 international* *conference on Information infrastructure systems for manufacturing*, pp. 59.

Krüger, M. & Große, C. 2009, "Event-based data acquisition in wireless sensor networks for structural health monitoring", *Technisches Messen,* vol. 76, no. 12, pp. 568-577.

Krzykacz-Hausmann, B. 2001, "Epistemic Sensitivity Analysis Based on the Concept of Entropy", *Inter'l Symp. Sensitivity Analysis of Model Output (SAMO2001)*, pp. 53.

Krzykacz-Hausmann, B. 2006, "An approximate sensitivity analysis of results from complex computer models in the presence of epistemic and aleatory uncertainties", *Reliability Engineering and System Safety,* vol. 91, no. 10-11, pp. 1210-1218.

Kwak, C. 2010, "Architecture of a dynamic production controller in CIM enterprise environments", *International Journal of Production Research,* vol. 48, no. 1, pp. 167-182.

Lallemand, B., Plessis, G., Tison, T. & Level, P. 1999, "Neumann expansion for fuzzy finite element analysis", *Engineering Computations (Swansea, Wales),* vol. 16, no. 5, pp. 572-583.

Landauer, R. 2000, "Irreversibility and heat generation in the computing process", *IBM Journal of Research and Development,* vol. 44, no. 1, pp. 261-269.

Lanner 2010, "How WITNESS Works" *[Homepage of Lanner], [Online]. Available:* http://www.lanner.com/en/witness.cfm [2010, 05/23].

Lavrač, N., Fürnkranz, J. & Gamberger, D. 2010, "Explicit feature construction and manipulation for covering rule learning algorithms", *Studies in Computational Intelligence, Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-74049153719&partnerID=40&md5=f361c4a49bc45c810c35f311957bc100.*

Lee, S., Ramakrishnan, S. & Wysk, R.A. 2002, "A FEDERATION OBJECT COORDINATOR FOR SIMULATION BASED CONTROL AND ANALYSIS", *2002 Winter Simulation Conference*ed, pp. 1986.

Lemaire, V. & Clérot, F. 2006, "An input variable importance definition based on empirical data probability distribution", *Studies in Fuzziness and Soft Computing, Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-*

34047164934&partnerID=40&md5=7bd07bcd90ae8b4c8a078f6969716f16.

Li, J., Zhang, C., Liang, Z. & Wang, B. 2006, "Stochastic simulation based approach for statistical analysis and characterization of composites manufacturing processes", *Journal of Manufacturing Systems,* vol. 25, no. 2, pp. 108-121.

Li, K. 1998, "Fast computation technique of Green's function and its boundary integrals for multilayered medium structures by using Fourier series expansion and method of images", *International Journal of RF and Microwave Computer-Aided Engineering,* vol. 8, no. 6, pp. 474-483.

Lopez, J.L. 2010, "The Liouville-Neumann expansion in one-dimensional boundary value problems", *Integral Transforms and Special Functions,* vol. 21, no. 2, pp. 125-133.

Manivannan, S. & Banks, J. 1991, "Real-time control of a manufacturing cell using knowledge-based simulation", *Winter Simulation Conference Proceedings*, pp. 251.

Mannarino, G.S., Henning, G.P. & Leone, H.P. 1997, "Metamodels for information system modeling in production environments", *The second IFIP TC5/WG5.3/WG5.7 international conference on Information infrastructure systems for manufacturing* Chapman & Hall, Ltd, London, UK, pp. 103.

Marri, H.B. & Brunel University. Department of Systems Engineering 2000, “Implementation of computer integrated manufacturing in small and medium enterprises”*, Brunel University, Uxbridge*.

Martin, A.J. 2001, "Towards an energy complexity of computation", *Information Processing Letters,* vol. 77, no. 2-4, pp. 181-187.

Masini, G. 1991, “Object-oriented languages”*, Academic, London*.

McCabe, T.J. & Butler, C.W. 1989, "Design complexity measurements and testing", *Communications of the ACM,* vol. 32, no. 12, pp. 1415-1425.

McCabe, T.J. 1976, "COMPLEXITY MEASURE.", *IEEE Transactions on Software Engineering,* vol. SE-2, no. 4, pp. 308-320.

McRae, G.J., Tilden, J.W. & Seinfeld, J.H. 1982, "Global sensitivity analysis-a computational implementation of the Fourier Amplitude Sensitivity Test (FAST)", *Computers and Chemical Engineering,* vol. 6, no. 1, pp. 15-25.

Mertins, K., Schwermer, M. & Jakel, F. 1997, "Confidence in a network of cooperating companies—QM-documentation for certification", *The second IFIP TC5/WG5.3/WG5.7 international conference on Information infrastructure systems for manufacturing*, eds., pp. 231.

Microsoft 2009, "The .NET Framework" *[Homepage of Microsoft], [Online]. Available: http://www.microsoft.com/net/* [2010, 05/22].

Microstar Laboratories, I. 2009, "Data Acquisition Processor (DAP) Boards" *[Homepage of Microstar Laboratories, Inc.], [Online]. Available: http://www.mstarlabs.com/dataacquisition/hwgroups.html* [2009, 11/01].

Mills, J.J. & Brand, M. 1997, "The system integration architecture: a framework for extended enterprise information systems", *The second IFIP TC5/WG5.3/WG5.7 international conference on Information infrastructure systems for manufacturing*, eds., pp. 155.

Mirkin, B.G. 2005, "Clustering for data mining: a data recovery approach"*, Chapman & Hall/CRC, Boca Raton, FL*.

Mitchell, F.H. 1991, "CIM Systems: An Introduction to Computer Integrated Manufacturing"*, Prentice-Hall, Englewood Cliffs, NJ*.

Mladenić, D. 2006, Feature selection for dimensionality reduction, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-33745840855&partnerID=40&md5=eb339e5bc2b37fe281c7ed9fc517a8f0*.

Modbus Organization, I. 2010*, "*Modbus Specifications and Implementation Guides" *[Homepage of Modbus Organization, Inc.], [Online]. Available: http://www.modbus.org/specs.php* [2009, 03/06].

Moore, S.K. 2008, "Multicore Is Bad News For Supercomputers", *IEEE, New York*.

Mousavi, A., Hamdi, M. & Sarhadi, M. 2007, "Quick-response decision-making in the food processing industry", *International Journal of Industrial and Systems Engineering,* vol. 2, no. 1, pp. 99-122.

Mousavi, A., Broomhead, P. & Devagiri, R. 2008, "A generic platform to address event-based problems for universal applications: Supervisory control and simulation", *International Journal of Industrial and Systems Engineering,* vol. 3, no. 3, pp. 257-276.

Mueller, R., Alonso, G. & Kossmann, D. 2007, "SwissQM: Next generation data processing in sensor networks", *3rd Conf. Innovative Data Systems Research*.

Mullarkey, P., Gavirneni, S. & Morrice, D.J. 2000, "Dynamic output analysis for simulations of manufacturing environments", *Winter Simulation Conference Proceedings*,pp. 1290.

Murrill, P.W. 1988, "Application Concepts of Process Control", *Instrument Society of America, Research Triangle Park, N.C., USA*.

National Instruments Corp. 2010, "LabVIEW Under the Hood: A Decade of Running on Multiprocessor/Multicore Platforms" *[Homepage of National Instruments Corp.], [Online]. Available: http://zone.ni.com/wv/app/doc/p/id/wv-359* [2010, 05/23].

National Instruments Corp. 2010, "Multicore Programming Resources" *[Homepage of National Instruments Corp.], [Online]. Available: http://www.ni.com/multicore/* [2010, 05/22].

National Instruments Corp. 2010, "NI LabVIEW [Homepage of National Instruments Corp.", *[Online]. Available: http://www.ni.com/labview/optin/* [2010, 05/22].

National Instruments Corp. 2009, 31/07/2009-last update, "New Parallel Technologies - From Signal to Software" *[Homepage of National Instruments], [Online]. Available: http://zone.ni.com/devzone/cda/tut/p/id/9403* [2009, 11/09].

National Instruments Corporation 2007, 08/01/2007-last update, "LabVIEW Run-Time Engine 8.5 - Windows 2000/XP/Vista/Vista x64" [*Homepage of National Instruments Corporation], [Online]. Available: http://joule.ni.com/nidu/cds/view/p/id/861/lang/en* [2009, 11/01].

National Instruments UK and Irelend 2008, "Customer Case Study Booklet", *NIDays08 Worldwide Garphical System Design Conference, Newbury, Berkshire, UK*.

National Instruments UK and Irelend 2007, "Customer Case Studies", *NIDays07 Worldwide Garphical System Design Conference, Newbury, Berkshire, UK*.

Norvig, P. & Russell, S.J. 1995, "Artificial Intelligence: Modern Approach", *1st edn, Prentice Hall*.

O'Hara, S., McLean, C. & Lee, Y.T. 2010, "Modeling and Simulation for Emergency Management and Health Care Systems", *National Institute of Standards and Technology (NIST), US Department of Commerce, USA*.

ODVA Inc. 2008, "ControlNet - CIP on CTDMA Technology" *[Homepage of ODVA Inc.], [Online]. Available: http://www.odva.org/Portals/0/Library/Publications_Numbered/PUB00200R0_CIP_Adv_Tech_Series_ControlNet.pdf* [2010, 14/02].

Øksendal, B.K. 1998, "Stochastic differential equations : an introduction with applications", *5th edn, Springer, Berlin ; London*.

Onut, S., Gunay, F.G., Demirel, T. & Erdem, H.I. 1994, "Using simulation for shop floor control", *Yugoslav Journal of Operations Research,* vol. 4, no. 2, pp. 195-203.

OPC Task Force 1998, "OPC Overview", *OPC Foundation*.

Oracle, "Oracle Data Integration Suite" *[Homepage of Oracle], [Online]. Available: http://www.oracle.com/us/products/middleware/data-integration/059300.html* [2010, 01/29].

Oracle, "Oracle GoldenGate" *[Homepage of Oracle], [Online]. Available: http://www.oracle.com/us/products/middleware/data-integration/059240.html* [2010, 01/29].

Oracle, "Oracle Product Data Quality" *[Homepage of Oracle], [Online]. Available: http://www.oracle.com/us/products/middleware/data-integration/data-quality/index.html* [2010, 01/29].

Oracle Corp. 2010, "Java Technology" *[Homepage of Oracle Corp.], [Online]. Available: http://www.sun.com/java/* [2010, 05/22].

OSIsoft Inc. 2007, "Integrating manufacturing data from the plant floor into SAP", *OSIsoft Inc*.

OSIsoft Inc., "PI System" *[Homepage of OSIsoft], [Online]. Available: http://www.osisoft.com/software-support/what-is-pi/what_is_PI_.aspx* [2009, 12/01].

Ozdemir, S. 2007, "Concealed data aggregation in heterogeneous sensor networks using privacy homomorphism", *IEEE Int. Conf. Pervasive Services*IEEE Computer Society, pp. 165.

Papoulis, A. 1991, "Probability, random variables, and stochastic processes", *3rd edn, McGraw-Hill, New York, USA*.

Park, J.I., Baek, S.H., Jeong, M.K. & Bae, S.J. 2009, "Dual features functional support vector machines for fault detection of rechargeable batteries", *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews,* vol. 39, no. 4, pp. 480-485.

Park, J., Mackay, S. & Wright, E. 2003, “Practical data communications for instrumentation and control”, *Elsevier, Amsterdam; London*.

Peng, W. 2008, "Energy analysis method of computational complexity", *Proceedings - International Symposium on Computer Science and Computational Technology, ISCSCT 2008,* pp. 682.

Penzes, P.I. & Martin, A.J. 2002, "Energy-delay efficiency of VLSI computations", *The 12th ACM Great Lakes symposium on VLSI* Association for Computing Machinery (ACM), New York, USA, pp. 104.

Ploetner, J. & Trivedi, M.M. 2006, "A multimodal approach for dynamic event capture of vehicles and pedestrians", *Proceedings of the ACM International Multimedia Conference and Exhibition*, pp. 203.

Potoradi, J., Boon, O.S., Mason, S.J., Fowler, J.W. & Pfund, M.E. 2002, "Using simulation-based scheduling to maximize demand fulfillment in a semiconductor assembly facility", *Winter Simulation Conference Proceedings,* pp. 1857.

Qi, L., Kambhamettu, C. & Jieping, Y. 2006, "Integrating spatial and discriminant strength for feature selection and linear dimensionality reduction*", Conference on Computer Vision and Pattern Recognition Workshops*.

Quevedo, J.R., Bahamonde, A. & Luaces, O. 2007, "A simple and efficient method for variable ranking according to their usefulness for learning", *Computational Statistics and Data Analysis,* vol. 52, no. 1, pp. 578-595.

Rabbath, C.A., Abdoune, M. & Belanger, J. 2000, "Effective real-time simulations of event-based systems", *Winter Simulation Conference Proceedings*, pp. 232.

Ragg, T. & Granzow, M. 2004, "Normalization, variable ranking and model selection for high-dimensional genomic data – Design and Implementation of automated analysis strategies", *WORKSHOP ON STATE-OF-THE-ART IN SCIENTIFIC COMPUTING PARA'04*Springer.

Ravalico, J.K., Maier, H.R., Dandy, G.C., Norton, J.P. & Crokef, B.F.W. 2005, "A Comparison of Sensitivity Analysis Techniques for Complex Models", *Int'l Cong. Modeling and Simulation MODSIM'05*, eds. A. Zerger & R.M. Argent, pp. 2533.

Reid, S.P. & Banerjee, S.K. 1997, "A supporting enterprise infrastructure using STEP technologies", *The second IFIP TC5/WG5.3/WG5.7 international conference on Information infrastructure systems for manufacturing*, eds. J. Goossenaerts, F. Kimura & J.C. Wortmann, Chapman and Hall, London, UK, pp. 81.

Rembold, U. 1993, "Computer integrated manufacturing and engineering", *Addison-Wesley, Wokingham, USA*.

Richardson, J.M. & Marsh, K.A. 1988, "Fusion of multisensor data", *International Journal of Robotics Research,* vol. 7, no. 6, pp. 78-96.

Rob, P., Coronel, C. & Crockett, K. 2008, "Database systems : design, implementation and management", *International edn, Cengage Learning, London*.

Robert, C.P. & Casella, G. 2004, "Monte Carlo statistical methods", *2nd edn, Springer, New York*.

Roca, J.L. 1996, "An Entropy-Based Method for Computing Software Structural Complexity", *Microelectronics Reliability,* vol. 36, no. 5, pp. 609-620.

Rockwell Automation Inc. 2010, "Arena simulation software" *[Homepage of Rockwell Automation Inc.], [Online]. Available: http://www.arenasimulation.com/* [2010, 05/23].

Rockwell Automation, I. 2010, "Assessment Services Manufacturing Assessment & Planning" *[Homepage of Rockwell Automation, Inc.], [Online]. Available: http://www.rockwellautomation.com/services/assessment/planning.html* [2010, 02/11].

Rockwell Automation, I. 2010, "Information Solutions Manufacturing Performance Solutions" *[Homepage of Rockwell Automation, Inc.], [Online]. Available: http://www.rockwellautomation.com/solutions/information/performance.html* [2010, 02/11].

Rosen, J.P. "HOOD An Industrial Approach for Software Design", *Adalog, ARCUEIL, FRANCE*.

Saltelli, A. 2002, "Sensitivity analysis for importance assessment", *Risk Analysis,* vol. 22, no. 3, pp. 579-590.

Saltelli, A., Tarantola, S., Campolongo, F. & Ratto, M. 2004, “SENSITIVITY ANALYSIS IN PRACTICE: A Guide to Assessing Scientific Models”*, 1st edn, Wiley*.

Saltelli, A., Tarantola, S. & Chan, K.P., 1999, "A quantitative model-independent method for global sensitivity analysis of model output", *Technometrics,* vol. 41, no. 1, pp. 39-56.

Sanchez, E.R. 2008, “Industrial Engineering study of the Application of Real-time Simulation in Product Tracking and Traceability Especially in Food Industry”, *School of Engineering and Design, Brunel University*.

SAP AG. 2009, “SAP NETWEAVER MASTER DATA MANAGEMENT”, *SAP AG*.

Shonkwiler, R.W. & Mendivil, F. 2009, “Explorations in Monte Carlo methods”*, Springer, Dordrecht ; London*.

Siemens 2009, 10/07/2009-last update*, “*Plant Intelligence with SIMATIC IT” *[Homepage of Siemens], [Online]. Available: http://www.automation.siemens.com/mcms/mes/en/plantintelligence/intmeslevel/Pages/Default.aspx* [2009, 12/02].

Siemens 2009, 02/09/2009-last update*, “*Scalability with WinCC” *[Homepage of Siemens], [Online]. Available: http://www.automation.siemens.com/mcms/human-machine-interface/en/visualization-software/scada/simatic-wincc/wincc-scalability/Pages/Default.aspx* [2009, 12/02].

Siemens 2005, “PROFIBUS according to IEC 61158/EN 50170” *[Homepage of Siemens IK PI], [Online]. Available: http://www.koneksis.com/urunler/PLC/siemens/siemens1.pdf* [2010, 13/02].

Simply Modbus 2008, “Modbus ASCII vs Modbus RTU” *[Homepage of Simply Modbus], [Online]. Available: http://www.simplymodbus.ca/ASCII.htm* [2009, 03/12].

SIMUL8 Corp. 2010, “SIMUL8 Simulation Software” *[Homepage of SIMUL8 Corp.], [Online]. Available: http://www.simul8.com/?ga_sim&gclid=CNet9NHpi6MCFYiX2AodVmrBfA* [2010, 05/23].

Smith, J.S. & Peters, B.A. 1996, “Short Term Scheduling Using Discrete Event Simulation”, *Department of Industrial Engineering, Texas A&M University, College Station, TX, USA*.

Sobol, I.M. 2001, "Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates", *Mathematics and Computers in Simulation,* vol. 55, no. 1-3, pp. 271-280.

Son, Y.J. & Wysk, R.A. 2001, "Automatic simulation model generation for simulation-based, real-time shop floor control", *Computers in Industry,* vol. 45, no. 3, pp. 291-308.

STMicroelectronics 2003, “PUSH-PULL FOUR CHANNEL DRIVER WITH DIODES”, *STMicroelectronics, Italy*.

Subramanian, N. 2008, "I-35W Mississippi river bridge failure – Is it a wake up call?", *The Indian Concrete Journal,* , no. February 2008, pp. 29-38.

Sudha, B., “Creating a High-Channel-Count, PC-Based Test System for Compressor Testing Using NI LabVIEW” *[Homepage of Soliton Technologies Pvt. Ltd.], [Online]. Available:* *http://sine.ni.com/cs/app/doc/p/id/cs-650* [2009, 02/06].

Takagi, T. & Sugeno, M. 1985, "Fuzzy identification of systems and its applications to modeling and control", *IEEE Transactions on Systems, Man and Cybernetics,* vol. 15, no. 1, pp. 116-132.

Takata, M. & Arai, E. 1997, "The glue logic: an integrated programming/execution environment for distributed manufacturing work-cell control system", *The second IFIP TC5/WG5.3/WG5.7 international conference on Information infrastructure systems for manufacturing*, eds., pp. 181.

Talavera, L. 2005, “An evaluation of filter and wrapper methods for feature selection in categorical clustering”, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-33745202625&partnerID=40&md5=cc0d9fca9e79e5adeb4857661a5ed560*.

Tanenbaum, A.S. 2002, “Computer networks”*, 4th edn, Prentice Hall, Upper Saddle River, NJ, USA*.

Tavakoli, S. 2009, “Interview with industrial exhibitors including Dale Fittes, Technical Specialist “hitex (UK) Ltd”, “AmFax Test & Measurement

Solutions", "Simplicity AI", "SigmaQuest Inc.", *NIDays09 Worldwide Graphical System Design Conference, Savoy Place, London, UK*.

Tavakoli, S. 2008b, "Interview with Keynote Speaker: The Parallel Shift – Discovering the Power of Concurrent Technologies", *NI Technical Symposium, Newbury, UK*.

Tavakoli, S. 2008a, "Interview with industrial exhibitors including Steve Lees "ATE Solutions", Richard White "Kane Computing Ltd", "Peak Group", "Sundance", "Test Works", *NIDays08 Worldwide Graphical System Design Conference, Savoy Place, London, UK*.

Tavakoli, S. 2007, "Interview with industrial exhibitors including Peter Heyes "SDC Test Limited", *NIDays07 Worldwide Graphical System Design Conference, Savoy Place, London, UK*.

Tavakoli, S. & Mousavi, A. 2010a, "Event Tracking for Real-Time Unaware Sensitivity Analysis (EvenTracker)", *Transaction paper edn, submitted to IEEE Transactions on Knowledge and Data Engineering*.

Tavakoli, S. & Mousavi, A. 2009b, "Reliability of Flexible Data Input Layer Architecture in Security Monitoring for Olympic Games", *Presented to the Pro-Vice-Chancellor for External Relations, Advanced Manufacturing and Enterprise Engineering (AMEE) Laboratory, School of Engineering and Design, Brunel University, London, UK*.

Tavakoli, S. & Mousavi, A. 2009a, "Advantages of Flexible Data Input Layer Architecture", *Presented to the industrial partners from Harford Control, Rockford27, Brunel University Research Support and Development Office (RSDO), Advanced Manufacturing and Enterprise Engineering (AMEE) Laboratory, School of Engineering and Design, Brunel University, London, UK.*

Tavakoli, S. & Mousavi, A. 2007b, "RFID for Tracking and Traceability in Manufacturing Scenarios", *Advanced Manufacturing and Enterprise Engineering (AMEE) Laboratory, School of Engineering and Design, Brunel University*.

Tavakoli, S. & Mousavi, A. 2007a, "Real-time Data Acquisition for Real-time Simulation", *Presented to the industrial partner from Harford Control, School of Engineering and Design, Brunel University, London, UK*.

Tavakoli, S., Mousavi, A. & Komashie, A. 2007c, "Real-time and Fast-forward Discrete Event Simulation in Manufacturing and Healthcare", *Presented to the industrial partner from NHS Trust, Advanced Manufacturing and*

Enterprise Engineering (AMEE) Laboratory, School of Engineering and Design, Brunel University, London, UK.

Tavakoli, S. & Mousavi, A. 2010b, “A Generic Framework for Measuring and Comparing Complexity and Efficiency of Sensitivity Analysis Methods in Time Critical Applications”, *Transaction Paper edn, submitted to IEEE Transactions on Software Engineering*.

Tavakoli, S., Mousavi, A. & Komashie, A. 2008, "Flexible data input layer architecture (FDILA) for quick-response decision making tools in volatile manufacturing systems", *IEEE International Conference on Communications,* pp. 5515.

Tavakoli, S., Mousavi, A. & Komashie, A. 2008, "A generic framework for real-time discrete event simulation (DES) modelling", *Proceedings - Winter Simulation Conference*, pp. 1931.

Taylor, H.R. 1997, “Data acquisition for sensor systems”*, Chapman & Hall, London*.

TecnoTeam 2000, “Philver Operator’s Manual”, *TecnoTeam*.

Thuraisingham, B. 2003, "Dependable infrastructures and data managers for sensor networks", *Ninth IEEE International Workshop on Object-Oriented Real-Time Dependable Systems (WORDS'03F)* IEEE Computer Society, pp. 11.

Underwood, L. & Economist Intelligence Unit 1994, “Intelligent manufacturing*”, Addison-Wesley Pub. Co., Wokingham*.

Unler, A. & Murat, A. 2010, "A discrete particle swarm optimization method for feature selection in binary classification problems", *European Journal of Operational Research,* vol. 206, no. 3, pp. 528-539.

USB Implementers Forum, I. 2008*, “*Universal Serial Bus” *[Homepage of USB Implementers Forum, Inc.], [Online]. Available:* http://www.usb.org/home [2009, 03/05].

Uysal, I. & Gu̇venir, H.A. 1999, "An overview of regression techniques for knowledge discovery", *Knowledge Engineering Review,* vol. 14, no. 4, pp. 319-340.

Vaidyanathan, B.S., Miller, D.M. & Park, Y.H. 1998, "Application of discrete event simulation in production scheduling", *Winter Simulation Conference Proceedings,* pp. 965.

Viswanadham, N. & Narahari, Y. 1992, “Performance modeling of automated manufacturing systems”*, Prentice Hall, Englewood Cliffs, N.J*.

Wheatley, M. 2009, "Simulating the factory", *Engineering & Technology,* vol. 4, no. 14, pp. 64-65.

Woodward, M.R., Hennell, M.A. & Hedley, D. 1979, "A measure of control flow complexity in program text", *IEEE Transactions on Software Engineering,* vol. 5, no. 1, pp. 45-50.

Wu, S.-.D. & Wysk, R.A. 1988, "Multi-pass expert control system - a control/scheduling structure for flexible manufacturing cells", *Journal of Manufacturing Systems,* vol. 7, no. 2, pp. 107-120.

Xing, Z.-., Jia, L.-., Qin, Y. & Lei, T. 2003, "Research on input variable selection for numeric data based fuzzy modeling", *International Conference on Machine Learning and Cybernetics*, pp. 2737.

Xu, C. & Gertner, G.Z. 2008, "A general first-order global sensitivity analysis method", *Reliability Engineering and System Safety,* vol. 93, no. 7, pp. 1060-1071.

Yang, X. 2003, "Green's function and positive solutions for higher-order ODE", *Applied Mathematics and Computation,* vol. 136, no. 2-3, pp. 379-393.

Zaman, S. & Karray, F. 2009, "Features selection using fuzzy ESVDF for data dimensionality reduction", *Proceedings - 2009 International Conference on Computer Engineering and Technology, ICCET 2009*, pp. 81.

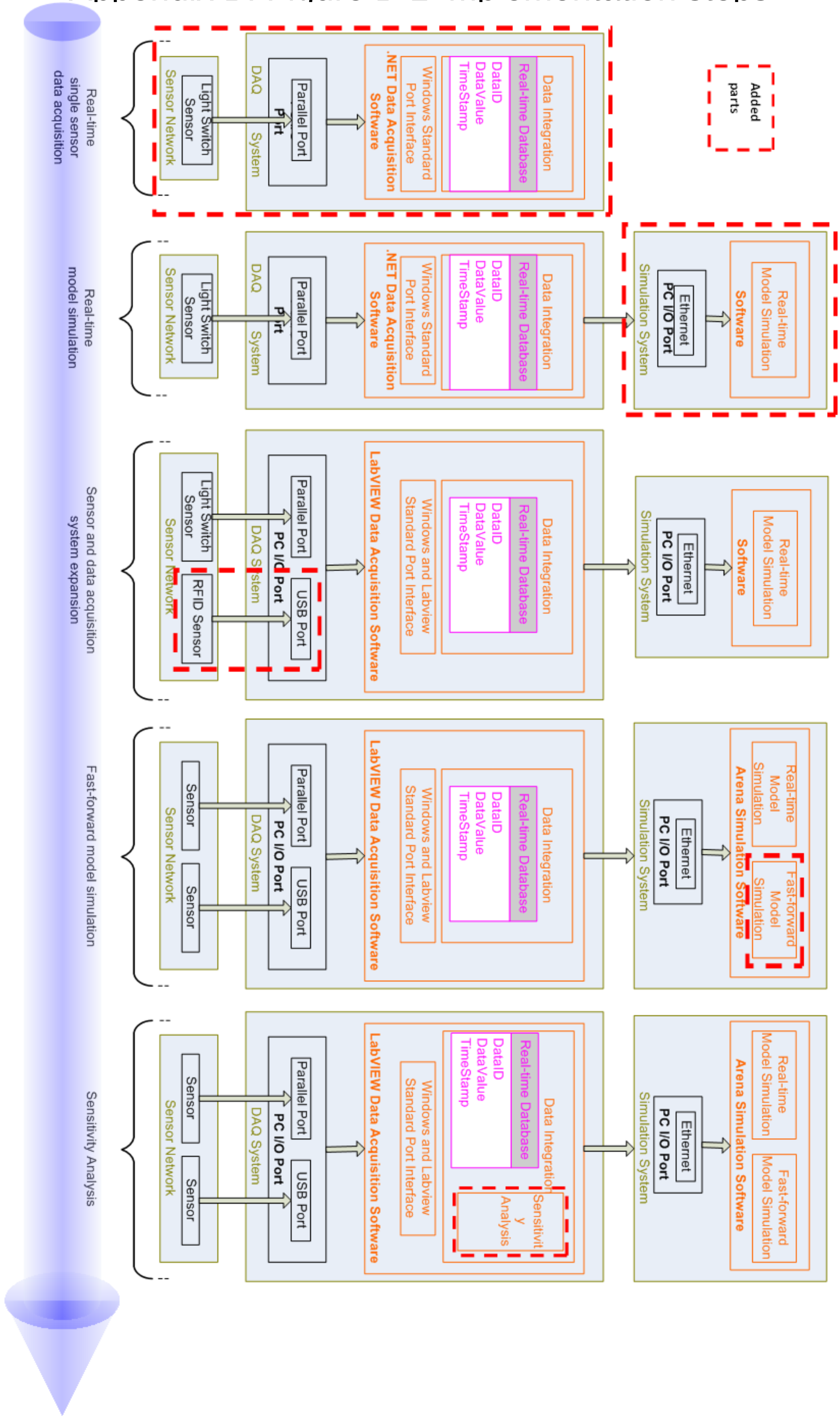
Zhang, H. & Hu, Y. 2010, "Study of feature selection for the stored-grain insects based on artificial immune algorithm", *CAR 2010 - 2010 2nd International Asia Conference on Informatics in Control, Automation and Robotics*, pp. 140.

Zhou, B., Ngoh, L.H., Lee, B.S. & Fu, C.P. 2006, "HDA: A hierarchical data aggregation scheme for sensor networks", *Computer Communications,* vol. 29, no. 9, pp. 1292-1299.

Appendix A: Table A-1 List of deployed sensors in the lab prototype demonstrator

	Input Tag	Sensor Type	Output Type	Data Type	Connection Type
1	100	RFID tag reader	Digital	Integer	USB Serial Port
2	101	RFID tag reader	Digital	Integer	USB Serial Port
3	102	RFID tag reader	Digital	Integer	USB Serial Port
4	103	RFID tag reader	Digital	Integer	USB Serial Port
5	104	RFID tag reader	Digital	Integer	USB Serial Port
6	40001	Manual Potentiometer	Analogue	Integer	USB Serial Port
7	40002	Manual Potentiometer	Analogue	Integer	
8	40003	Manual Potentiometer	Analogue	Integer	
9	40004	Manual Potentiometer	Analogue	Integer	
10	40005	Manual Potentiometer	Analogue	Integer	
11	40006	Luminance meter	Analogue	Integer	
12	40007	Microphone	Analogue	Integer	
13	40008	Thermometer	Analogue	Integer	
14	50007	Manual ON-OFF Switch	Digital	Boolean	
15	50008	Manual ON-OFF Switch	Digital	Boolean	
16	50009	Manual ON-OFF Switch	Digital	Boolean	
17	50010	Manual ON-OFF Switch	Digital	Boolean	
18	50011	Manual ON-OFF Switch	Digital	Boolean	
19	50012	Manual ON-OFF Switch	Digital	Boolean	
20	202	Light switch	Digital	Boolean	Parallel Port
21	203	Light switch	Digital	Boolean	
22	204	Manual ON-OFF Switch	Digital	Boolean	
23	205	Light switch	Digital	Boolean	
24	206	Light switch	Digital	Boolean	
25	207	Light switch	Digital	Boolean	
26	208	Light switch	Digital	Boolean	OPC
27	209	Light switch	Digital	Boolean	OPC
28	50210	Thermometer	Analogue	Integer	OPC
29	50211	Load Sensor	Analogue	Integer	OPC

Appendix B: Figure B-1 Implementation steps



Appendix C: Table C-1 List of signals in four task groups of the side-panel manufacturing line

Task Group	Signal	Sensor/Actuator	Role	Data Type
GR-1	CP101	Actuator	Manual loader up/down	2-bit digital
	LS101A	Sensor	Loader up	1-bit digital
	LS101B	Sensor	Loader down	1-bit digital
	LS102	Sensor	Sheet presence (align)	1-bit digital
	LSDP	Sensor	Double sheet	1-bit digital
	M101	Actuator	Transport forw/rev	2-bit digital
	M102	Actuator	Manual trolley forw/rev	2-bit digital
GR-2	M201	Actuator	Manual transport 1	2-bit digital
	M202	Actuator	Manual transport 2	2-bit digital
	CP210	Actuator	Sheet-in stopped up/down	2-bit digital
	LS210A	Sensor	Sheet-in stopped up	1-bit digital
	LS210B	Sensor	Sheet-in stopped down	1-bit digital
	LS210C	Sensor	Slowing stopped CP210	1-bit digital
	CP211	Actuator	Manual magnet 1 up/down	2-bit digital
	LS211A	Sensor	Magnet 1 CP211 up	1-bit digital
	LS211B	Sensor	Magnet 1 CP211 down	1-bit digital
	CP212-213	Actuator	Manual magnet 2 up/down	2-bit digital
	LS212A	Sensor	Magnet 2 CP212 up	1-bit digital
	LS212B	Sensor	Magnet 2 CP212 down	1-bit digital
	LS213A	Sensor	Magnet 2 CP213 up	1-bit digital
	LS213B	Sensor	Magnet 2 CP213 down	1-bit digital
	CP214	Actuator	Manual centring forw/rev	2-bit digital
	LS214A	Sensor	Centring forw	1-bit digital
	LS214B	Sensor	Centring rev	1-bit digital
	CP215	Actuator	Manual pincer open/close	2-bit digital
	LS215A	Sensor	Pincer open	1-bit digital
	LS215B	Sensor	Pincer close	1-bit digital
	LS220	Sensor	Out block	1-bit digital
	M203	Actuator	Manual roll machine forw/rev	2-bit digital
	LS203C	Sensor	Axel manilpulator forw	1-bit digital
	LS203D	Sensor	Axel manilpulator rev	1-bit digital
	LS203E	Sensor	Axel manilpulator home	1-bit digital
	M204	Actuator	Axel move guide forw/rev	2-bit digital
	LS204C	Sensor	Axel move guide forw	1-bit digital
	LS204D	Sensor	Axel move guide rev	1-bit digital
	LS204E	Sensor	Axel move guide home	1-bit digital
	M207	Actuator	Axel centring forw/rev	2-bit digital

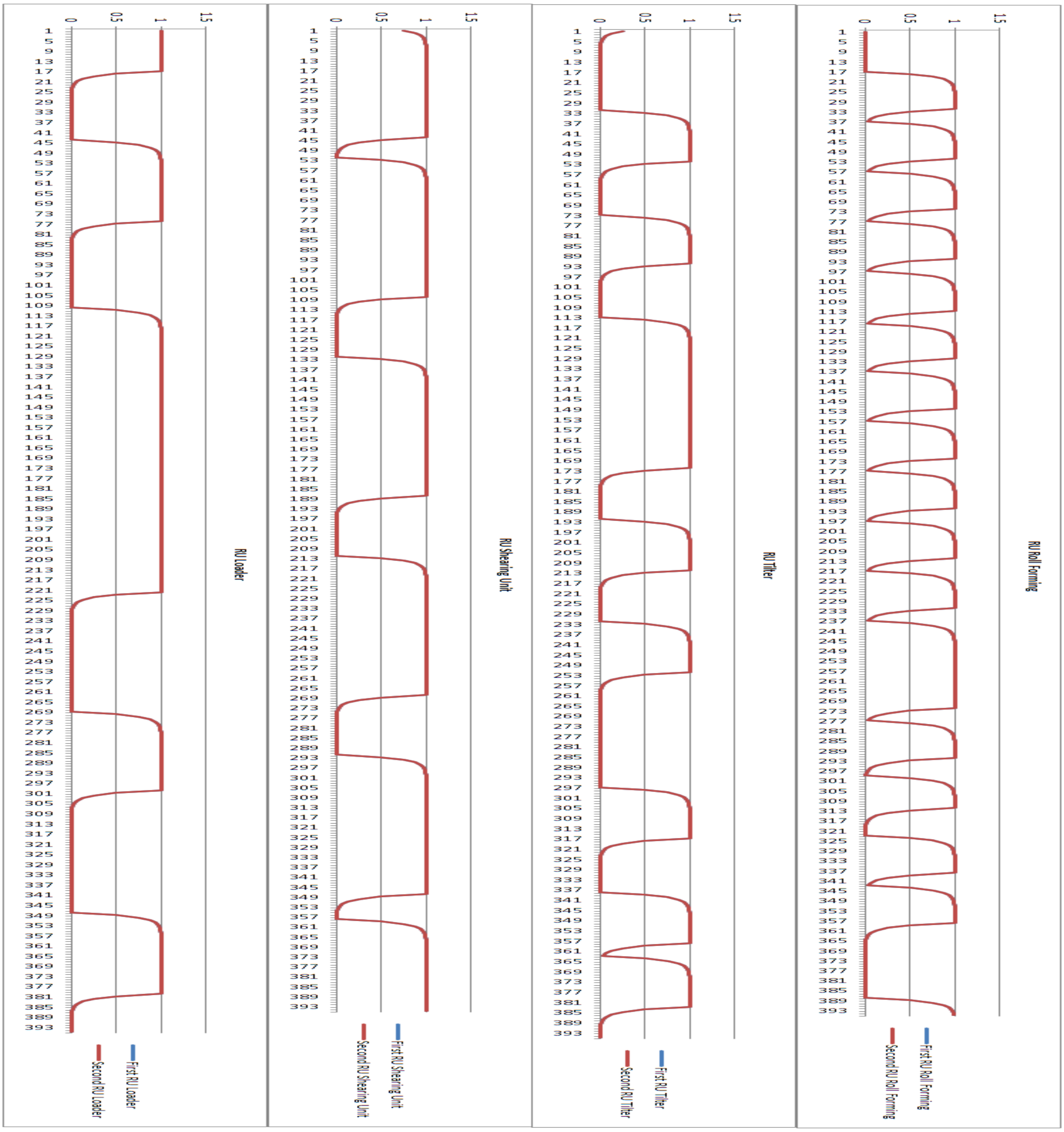
	LS207C	Sensor	Axel centring forw	1-bit digital
	LS207D	Sensor	Axel centring rev	1-bit digital
	LS207E	Sensor	Axel centring home	1-bit digital
	CO201	Actuator	Manual blanking die up/down	2-bit digital
	LS201A	Sensor	Blanking cylinder CO201 up	1-bit digital
	LS201B	Sensor	Blanking cylinder CO201 down	1-bit digital
GR-3A	M303	Actuator	Manual tip up transport forw/rev	2-bit digital
	M304	Actuator	Manual tip up rotation forw/rev	2-bit digital
	CP305	Actuator	Manual tip up block forw/rev	2-bit digital
	LS305A	Sensor	Tip up block forw	1-bit digital
	LS305I	Sensor	Tip up block rev	1-bit digital
	LS303A	Sensor	Tip up rotation up	1-bit digital
	LS303B	Sensor	Tip up rotation down	1-bit digital
	LS303C	Sensor	Tip up rotation slowing up	1-bit digital
	LS303D	Sensor	Tip up rotation slowing down	1-bit digital
	LS306	Sensor	Tip up stop outside	1-bit digital
	LS307	Sensor	Tip up stop inside	1-bit digital
GR-3	M301	Actuator	Manual roll machine forw/rev	2-bit digital
	LS301	Sensor	Roll machine location 1	1-bit digital
	LS302	Sensor	Roll machine location 2	1-bit digital
	LS303	Sensor	Roll machine location 3	1-bit digital
	LS304	Sensor	Roll machine location 4	1-bit digital
	LS305	Sensor	Roll machine location 5	1-bit digital
	M302	Actuator	Move guide forw/rev	2-bit digital
	LS302C	Sensor	Move guide forw	1-bit digital
	LS302D	Sensor	Move guide rev	1-bit digital
	LS302E	Sensor	Move guide home	1-bit digital

Appendix D: Table D-1 List of reportedly more important Trigger Data series (green cells) and less important Trigger Data series (red cells) after EvenTracker method analysed 28 pairs of {Event Data, Trigger Data} with 30% Cut-off Ratio.

Underlined numbers indicate truly more important Trigger Data series according to the model structure.

	RU Loader	RU Shearing Unit	RU Tilter	RU Roll Forming	
CP101	0.695652	0.789474	0.37838	0.325	More Important
LS101A	0.652174	0.263158	0.48649	0.475	More Important
LS101B	0.565217	0.684211	0.75676	0.675	More Important
LS102	0.521739	0.631579	0.72973	0.65	More Important
LS103	0.565217	0.473684	0.59459	0.475	Less Important
LS201A	0.652174	0.263158	0.48649	0.475	More Important
LS201B	0.652174	0.263158	0.64865	0.575	More Important
LS210A	0.608696	0.210526	0.45946	0.35	More Important
LS210B	0.565217	0.789474	0.7027	0.625	More Important
LS210C	0.826087	0.473684	0.59459	0.525	More Important
LS211A	0.608696	0.210526	0.45946	0.35	More Important
LS211B	0.565217	1	0.75676	0.675	More Important
LS212A	1	1	0.43243	0.375	More Important
LS212B	0.652174	0.368421	0.75676	0.575	More Important
LS214A	0.73913	0.263158	0.64865	0.575	More Important
LS214B	0.652174	0.263158	0.43243	0.325	More Important
LS215A	0.565217	0.263158	0.81081	0.625	More Important
LS215B	0.826087	0.263158	0.7027	0.675	More Important
LS220	0.565217	0.263158	0.7027	0.525	More Important
LS301	0.173913	0	0.89189	1	More Important
LS302	0.304348	0.052632	0	0.025	Less Important
LS303	0.304348	0.052632	0	0.025	Less Important
LS304	0.26087	0.105263	0.08108	0	Less Important
LS305	0	0.105263	1	0.95	More Important
LS306	0.652174	0.157895	0.7027	0.675	More Important
LS307	0.478261	0.052632	0.86486	0.775	More Important
LS3051	0.608696	0.105263	0.67568	0.65	More Important
LS3059	0.347826	0.631579	0.89189	0.8	More Important

Appendix E: Figure E-1 matching Event Data values before and after de-selection of reported less important Trigger Data series. Each diagram holds two identical data series



Appendix F: Awards, Publications and Seminar Presentations Resulting from This Research

AWARDS

1. Winter Simulation Conference 2008 (Florida, USA), PhD Colloquium Presentation Award.
2. Brunel University Vice Chancellor's Travel Prize for Research Students 2006/2007/2008.

PUBLICATIONS

Journals

1. **Siamak Tavakoli**, Ali Mousavi (2010), *A Generic Framework for Measuring and Comparing Complexity and Efficiency of Sensitivity Analysis Methods in Time Critical Applications***,** Submitted to the Transactions on Software Engineering.
2. **Siamak Tavakoli**, Ali Mousavi (2010) *Event Tracking for Real-Time Unaware Sensitivity Analysis (EvenTracker)*, Submitted to Transactions on Knowledge and Data Engineering.
3. **Siamak Tavakoli**, Ali Mousavi (2009), *Product Customisation: A Case for the Application of Real-Time Discrete Event Simulation for Traceability in Food Manufacturing (SIMTrack)*, International Journal of Mass Customisation, Vol. 3, No.1 pp. 18 – 37.
4. **Siamak Tavakoli,** Ali Mousavi (2007), *Input Variable Monitoring for Quick-Response Decision Making in Volatile Systems*, Research in Interactive Design - Vol.2 Springer Verlage.

Conferences

1. **Siamak Tavakoli** , Ali Mousavi (2008), *Adopting User-Interacted Mobile Node Data to Flexible Data Input Layer Architecture*, Proceedings of the Fourth International Conference on Intelligent Sensors, Sensor Networks and Information Processing, Dec 2008, Sydney, Australia.

2. **Siamak Tavakoli**, Ali Mousavi and Alexander Komashie (2008), *A generic platform for real-time discrete event simulation modelling,* Proceedings of 40th Winter Simulation Conference, 2008, Miami, FL, USA.
3. **Siamak Tavakoli**, Ali Mousavi and Alexander Komashie (2008), *Flexible data input layer architecture (FDILA) for quick-response decision making tools in volatile manufacturing systems,* Proceedings of IEEE ICC'08 Conference, China.
4. **Siamak Tavakoli**, Ali Mousavi (2006), *Input Variable Monitoring for Quick-Response Decision Making in Volatile Systems*, Proceedings of Virtual Concept 2006 , November 2006, Playa Del Carmen , Mexico.

SEMINARS

1. **Siamak Tavakoli**, Ali Mousavi, *Reliability of Flexible Data Input Layer Architecture in Security Monitoring for Olympic Games,* presented to the Pro-Vice-Chancellor for External Relations, Advanced Manufacturing and Enterprise Engineering (AMEE) Laboratory, School of Engineering and Design, Brunel University, 2009.
2. **Siamak Tavakoli**, Ali Mousavi, *Advantages of Flexible Data Input Layer Architecture,* presented to the industrial partners from Harford Control, Rockford27, and Brunel University Research Support and Development Office (RSDO), Advanced Manufacturing and Enterprise Engineering (AMEE) Laboratory, School of Engineering and Design, Brunel University, 2009.
3. **Siamak Tavakoli**, Ali Mousavi, *A Generic Platform for Real-time Discrete Event Simulation,* Winter Simulation Conference 2008, (an abstract presented at the WSC'08 PhD Colloquium in Florida, USA).
4. **Siamak Tavakoli**, Ali Mousavi, *A Generic Platform for Real-time Discrete Event Simulation,* a poster presented at the first School of Engineering and Design Research Student Conference, Brunel University, June 2008.
5. **Siamak Tavakoli**, Ali Mousavi and Alexander Komashie, *Real-time and Fast-forward Discrete Event Simulation in Manufacturing and*

Healthcare, presented to the industrial partner from NHS Trust, Advanced Manufacturing and Enterprise Engineering (AMEE) Laboratory, School of Engineering and Design, Brunel University, 2007.

6. **Siamak Tavakoli**, Ali Mousavi, *RFID for Tracking and Traceability in Manufacturing Scenarios,* presented to the industrial partners from Harford Control, Advanced Manufacturing and Enterprise Engineering (AMEE) Laboratory, School of Engineering and Design, Brunel University, 2007.
7. **Siamak Tavakoli**, Ali Mousavi, *Real-time Data Acquisition for Real-time Simulation,* presented to the industrial partner from Harford Control, School of Engineering and Design, Brunel University, 2007.

PERSONAL COMMUNICATIONS

1. **Siamak Tavakoli**, interview with industrial exhibitors including Dale Fittes, Technical Specialist “hitex (UK) Ltd”, “AmFax Test & Measurement Solutions”, “Simplicity AI”, “SigmaQuest Inc.”, *NIDays09 Worldwide Graphical System Design Conference,* Savoy Place, London, UK, Nov. 2009.
2. **Siamak Tavakoli,** interview with Keynote Speaker, *The Parallel Shift – Discovering the Power of Concurrent Technologies,* NI Technical Symposium, Newbury, UK, 2008.
3. **Siamak Tavakoli**, interview with industrial exhibitors including Steve Lees “ATE Solutions”, Richard White “Kane Computing Ltd”, “Peak Group”, “Sundance”, “Test Works”, *NIDays08 Worldwide Graphical System Design Conference,* Savoy Place, London, UK, Nov. 2008.
4. **Siamak Tavakoli**, interview with industrial exhibitors including Peter Heyes “SDC Test Limited”, *NIDays07 Worldwide Graphical System Design Conference,* Savoy Place, London, UK, Nov. 2007.