University of Warwick

Engineering Doctorate: Executive Summary

Data Integrity:
An often-ignored aspect of safety systems

Eur Ing Alastair Faulkner CEng

November 2004

Contents

Acknowledgements 6

Author's declaration 7

Abstract 9

Abbreviations 10

Definitions 11

1 Introduction 13

1.1 The use of data by safety systems 13
1.2 Proposition: Data as a separate systems component 13
1.3 Aims and objectives of this research 14
1.4 The intended audience for this research 15
1.5 Structure of this document 15

2 The basis for this research 17

2.1 Data error as a causal factor in an accident 17
2.2 A theoretical air accident 17
2.3 Flight AA 965 – Cali Columbia – December 1995 18
2.4 Flight AA 965 is not an isolated incident 20
2.5 An initial search for guidance 20

3 The treatment of data in the existing literature and a survey of industrial practice 21

3.1 The research environment 21
3.2 The treatment of data in the existing literature 21
3.3 A survey of industrial practice 22
3.4 A conventional view of data used by safety systems 22

4 Defining data for use by safety systems 24

4.1 The use of an extended data dictionary to support data definition 24
4.2 The properties of data 24
4.3 A classification of data used in safety systems 25
4.4 An assurance model for data used in safety systems 28
4.5 Data definition in the data supply chain 29

5 Deriving data integrity requirements 30

5.1 The use of integrity requirements as a measure of criticality 30
5.2 Integrity requirements based upon numerical targets for failure rates 30

5.3 Adapting Functional Failure Analysis for use with data 30
5.4 A proposed set of guidewords to be used with Functional Failure Analysis 31
5.5 Apportionment of the system safety requirements to data 31

6 The use (and re-use) of data in large-scale systems 32

6.1 A requirement for a model to assist in the visualisation of large-scale systems 32
6.2 A layered model for large-scale systems 32
6.3 A layered model to express the role of one or more systems within a hierarchy of systems 35
6.4 The use of the layered model to expose the use (and reuse) of data 36
6.5 Rules for data exchange between systems within the layered model 37

7 Validating data for use by safety systems 38

7.1 Proposing a data validation policy 38
7.2 Data passed across the system interface 38
7.3 Data validation at point of data entry 39
7.4 Data validation as the responsibility of the consuming system 39
7.5 Data validation policy and its influence on the system safety case 39

8 Developing the data component of safety systems 40

8.1 Developing data-driven systems 40
8.2 Data development 40
8.3 A development lifecycle for data 40
8.4 The integration of the generic and application data lifecycle models within a system development 43
8.5 The logistics required to implement regular data updates 45
8.6 The execution of application dataset updates during the working lifetime of the system 47
8.7 Integrating application development with data provision 48

9 The design and assessment of a data supply chain 49

9.1 Data provision using a data supply chain 49
9.2 The DO 200A data supply chain 49
9.3 Generalising the DO 200A data supply chain 49
9.4 Defining a graphical representation for a data supply chain 50
9.5 A design and assessment method for a data supply chain 50
9.6 Common formations in the data supply chain 51
9.7 Abstraction in data provision 53
9.8 A data processing model for the data supply chain 53
9.9 Concerns over the stability of a data supply chain 54
9.10 Data ownership, liability for data errors and risk management 55

10 Discussion **57**

10.1 The treatment of data used by safety systems 57
10.2 A conventional view of data 57
10.3 A data paradigm - the treatment of data as a separate systems component 58
10.4 Defining data for use by safety systems 59
10.5 The determination of data integrity requirements 59
10.6 Identifying and managing the desirable properties of data 60
10.7 The selection of a data lifecycle model 60
10.8 Lifecycle models for generic and application data 61
10.9 The design and structure of the data component 61
10.10 Consideration of the safety arguments for generic and application data 63
10.11 Data provision using a data supply chain 65
10.12 Rules for the exchange of data between systems of differing integrity 65
10.13 The design and construction of the data supply chain 66
10.14 Use of software tools and applications to support the data component 66

11 Innovation, review and dissemination **68**

11.1 The treatment of data prior to this research 68
11.2 Demonstration of innovation 68
11.3 'Raising the profile' of data 70
11.4 Dissemination and peer review 70

12 Conclusions **71**

12.1 The stated aims of this research 71
12.2 The treatment of data as a separate systems component 71
12.3 Guidance for the management of the data component of safety systems 72
12.4 The dissemination of the research through published papers and presentation to professional forums 75

13 Recommendations **76**

13.1 A definition of data integrity 76
13.2 Data as a separate systems component 76
13.3 Techniques and measures to be used with the data component 77
13.4 The definition of data used by safety systems 77
13.5 The use of the layered model when considering the use of data within a systems hierarchy 79
13.6 Lifecycle models for generic and application data 79
13.7 Roles and responsibilities for those involved in the design, development and provision of data 80
13.8 The provision of data for safety systems 81

13.9 Data ownership, the cost of data, liability for data errors and risk 81

13.10 Strategies to be used in the construction of a safety argument for a data-driven system 82

References 83

Appendix A Structure of the Engineering Doctorate Portfolio 88

A.1 Overview of the submissions 88

A.2 A suggested order of reading the submissions 90

Appendix B A layered model for a hierarchy of systems 91

B.1 Introduction 91

B.2 Describing the layered model 91

B.3 Coupling and independence 95

B.4 System boundary issues 97

Appendix C The design and construction of a data supply chain 100

C.1 The design and construction of a data supply chain 100

C.2 The design of a data supply chain 102

List of figures

Figure 1: A segment of the navigation chart for Flight AA 965 approach to Cali 19

Figure 2: Data in the context of a control system 26

Figure 3: An operational railway as a large-scale system 32

Figure 4: A pictorial representation of an operational railway 33

Figure 5: The signalling function within a railway control centre 34

Figure 6: A layered model for a hierarchy of systems 36

Figure 7: Data passed across an interface at the system boundary 38

Figure 8: A generic development lifecycle for the data component 42

Figure 9: An application data lifecycle 43

Figure 10: A system development timeline 44

Figure 11: An idealised operational data provision lifecycle model 46

Figure 12: A more practical view of operational data provision 46

Figure 13: Providing data updates during the working lifetime 47

Figure 14: Provision of data by the developers and the data providers 48

Figure 15: Data passed across a boundary with supporting evidence 51

Figure 16: Data passed across a boundary without supporting evidence 52

Figure 17: A fragment of the system development lifecycle. 54

Figure 18: A fragment of the data supply chain to illustrate feedback. 54

Figure 19: A data supply chain crossing organisational boundaries 55

Figure 20: A layered model for a hierarchy of systems 92

Figure 21: Coupling within an operator workstation 96

Figure 22: Step 1 - Identify the data origins 103

Figure 23: Step 2 - Identify the boundaries 103

Figure 24: Step 3 - Identify the adaptations and processing of the datasets 104

Figure 25: Step 4 - Apportion the integrity requirements 104

Figure 26: Step 5 - Identify evidence requirements 105

Figure 27: Step 6 - Specify the corrective action process 105

List of tables

Table 1: Guidewords used in the consideration of possible data errors 31

Table 2: Data supply chain symbols 101

Acknowledgements

This doctorate could not have been completed without the support and patience of my wife Cheryl, and my children Eamon and Grace.

I would like to thank my colleagues at CSE International Ltd who have many years of experience of systems, software and safety issues. They have listened attentively to my presentations, read my submissions and after all that, provided challenging questions which have been both provocative and constructive. I would also like to thank Dr Phil Bennett for his support and encouragement. In addition this work has been supported by EPSRC without whose grant this work would not have been possible.

Finally my thanks to Dr Neil Storey and Ron Pierce, my academic and industrial mentors respectively, for their patience and understanding. They have encouraged, listened and provided support.

Author's declaration

Some of the material presented within this Executive Summary has previously been published in the following papers:

i) A. Faulkner, P. A. Bennett, R. H. Pierce, I. H. A. Johnston and N. Storey: "*The safety management of data-driven safety-related Systems*"; Lecture notes in computer science - Proceedings of the 19th International Conference SafeComp 2000, pp 86-95, ISBN 3-540-41186-0.

ii) A. Faulkner, "*The use of safety-related data within a railway command and control system*"; Proceedings of the Engineering Doctorate Conference 2001, IMC Univ. of Warwick.

iii) A. Faulkner and R. H. Pierce, "*Is it Software or Is it data*"; Proceedings of the 19th International Safety System Conference 2001, Huntsville. Alabama, USA. pp 323-329.

iv) N. Storey and A. Faulkner, "*The role of data in safety-related systems*"; Proceedings of the 19th International Safety System Conference 2001, Huntsville. Alabama, USA. pp 26-35.

v) A. Faulkner and N. Storey, "*The role of data in safety-related railway control systems*"; Proceedings of the 19th International Safety System Conference 2001, Huntsville. Alabama, USA. pp 793-800.

vi) A. Faulkner: "*Safer Data: The use of data in the context of a railway control system*"; Proceedings of the tenth Safety-critical Systems Symposium, pp 217-230 ISBN: 1-85233-561-0, Southampton, UK 2002.

vii) A. Faulkner, "*Data integrity: Assessing data used by safety-related control systems*"; Proceedings of the Engineering Doctorate Conference 2002, IMC Univ. of Warwick.

viii) N. Storey and A. Faulkner: "*Data Management in Data-Driven Safety-Related Systems*"; Proceedings of the 20th International Safety System Conference 2002, Denver. Colorado USA. pp 466-475 ISBN 0-9721385-1-X.

ix) A. Faulkner and N. Storey: "*Data: An often-ignored component of safety-related systems*"; Proceedings of the MoD *Equipment Safety Assurance Symposium ESAS02*, Bristol, UK. Oct. 2002.

x) A. Faulkner and A Harrison, "*Is asset information 'good enough' to be used by safety-related systems*"; Proceedings of Engineering Asset Management, London, 2002 ERA Report 2002-0409, ISBN 0 7008 0765 9.

xi) A. Faulkner, "*Data-intensive systems: Sourcing data of the required integrity, the problem of origination*"; Proceedings of the Engineering Doctorate Conference 2003, IMC Univ. of Warwick.

xii) A. Faulkner and N. Storey, "*Strategies for the Management of Data-Intensive Safety-Related Systems*"; Proceedings of the 21st International Safety System Conference 2003, Ottawa. Canada. pp 855-864, ISBN 0-9721385-2-8.

xiii) N. Storey and A. Faulkner: "*Data Requirements for Data-Intensive Safety-Related Systems*"; Proceedings of the 21st International Safety System Conference 2003, Ottawa. Canada. pp 865-874, ISBN 0 9721385-2-8.

xiv) N. Storey and A. Faulkner: "*The Characteristics of Data in Data-Intensive Safety-Related Systems*"; Lecture notes in computer science - Proceedings of the 22nd International Conference SafeComp 2003, pp 396-409, ISBN 3-540-20126-2.

xv) N. Storey and A. Faulkner: "*Data - The Forgotten System Component?*", "*Journal of System Safety*"; A publication of the System Safety Society, Volume 39, No 4 Fourth Quarter 2003, pp10-14.

All the work contained within this Executive Summary represents the original contribution of the author, unless stated otherwise.

Abstract

Data Integrity:

An often-ignored aspect of safety systems

Data is all-pervasive and is found in all aspects of modern computer systems, and yet many engineers seem reluctant to recognise the importance of data integrity. The conventional view of data, as simply an aspect of software, underestimates the role played by data errors in the behaviour of the system and their potential effect on the integrity of the overall system. In many cases hazard analysis is not applied to data in the same way that it is applied to other system components. Without data integrity requirements, data development and data provision may not attract the degree of rigour that would be required of other system components of a similar integrity. This omission also has implications for safety assessment where the data is often ignored or neglected. This position becomes self re-enforcing, as without integrity requirements the importance of data integrity remains hidden.

This research provides a wide-ranging overview of the use (and abuse) of data within safety systems, and proposes a range of strategies and techniques to improve the safety of such systems. A literature review and a survey of industrial practice confirmed the conventional view of data, and showed that there is little consistency in the methods used for data development. To tackle these problems this work proposes a novel paradigm, in which data is considered as a separate and distinct system component. This approach not only ensures that data is given the importance that it deserves, but also simplifies the task of providing guidance that is specific to data. Having developed this conceptual framework for data, the work then goes on to develop lifecycle models to assist with data development, and to propose a range of techniques appropriate for the various lifecycle phases.

An important aspect of the development of any safety-related system is the production of a safety argument, and this research looks in some detail at the treatment of data, and data development, within this justification. The industrial survey reveals that in data-intensive systems data is often developed quite separately from other elements of the system. It also reveals that data is often produced by an extended data supply chain that may involve a number of disparate organisations. These characteristics of data distinguish it from other system components and greatly complicate the achievement and demonstration of safety. This research proposes methods of modelling complex data supply chains and proposes techniques for tackling the difficult task of safety justification for such systems.

Abbreviations

AEL	Assurance Evidence Level
ATC	Air Traffic Control
ATM	Air Traffic Management
CAA	Civil Aviation Authority
CAP	Civil Aviation Authority Procedure
CENELEC	European Committee for Electro-technical Standardisation
COTS	Commercial Off The Shelf
DDR	Lifecycle Stage Gate: Detailed Design Review
DISC	Demonstration of Integrated Ship Control
E/E/PE	Electrical, Electronic or Programmable Electronic System as described by IEC 61508 [1].
EDA	Enterprise Data Architecture
EDPM	Enterprise Data Processing Model
EPSRC	Engineering and Physical Science Research Council
EngD	Engineering Doctorate
ESM	Engineering Safety Management (aka Yellow Book 3)
FAT	Lifecycle Stage Gate: Factory Acceptance Test
FFA	Functional Failure Analysis
FMS	Flight Management System
GPWS	Ground Proximity Warning System
IEC	International Electrotechnical Commission
ILS	Instrument Landing System
ISO	International Standards Organisation
NATS	National Air Traffic Service
NDB	Non Directional Beacon
OSI	Open Systems Interconnect
PDR	Lifecycle Stage Gate: Preliminary Design Review
SHARD	Software Hazard Analysis and Resolution in Design [2]
SAT	Lifecycle Stage Gate: Site Acceptance Test
SDR	Lifecycle Stage Gate: System Design Review
SIL	Safety Integrity Level [3]
SRR	Lifecycle Stage Gate: System Requirements Review
TRR	Lifecycle Stage Gate: Test Readiness Review
VOR	VHF (Very High Frequency) Omni-directional Radio-range

Definitions

Data driven systems

In this research, computer-based safety-related systems are considered as using data in the form of an extensive configuration dataset, as data exchanged across interfaces with other systems, or as a schedule that describes some planned use of the system. Perhaps one summary of these three categories covers both data provided to, and data produced by, computer-based systems. The word "extensive" in the earlier sentence is included because a simple configuration dataset consisting of only a few data values could be verified adequately by inspection and extensive engineering processes would not be required. However, a typical configuration dataset for a complex control system may contain hundreds of thousands of data items, often with complex relationships, and ensuring the integrity of all this data is then far from trivial [4].

Safety system

A safety-related system (or safety system) is required to implement the safety functions necessary to achieve a required risk reduction (or, more generally, to control risk such that it is tolerable). In addition the safety-related system is expected to achieve the necessary safety integrity to support these required safety functions. A failure of a safety-related system may give rise to a hazard that may lead directly to harm.

Data

Data is an abstraction. Common usage would regard numbers, characters and perhaps images as data. In more precise terms the association of context gives these numbers, characters and images meaning. Therefore we may infer that the intended use of the data provides the data with meaning. The use of data can range from representations of objects from the real world to abstract descriptions.

Data integrity

The term *data integrity* is perhaps overloaded. This term is used in a variety of circumstances and in each circumstance takes on a subtly different meaning. In this research ***data integrity*** *is taken as a measure of data quality* [5]. The term *data integrity* used within this research is derived from Engineering Safety Management (ESM) [6] and Sandhu [7] (who cites Courtney and Ware [8]).

From the ESM definition for Safety Integrity (see below), ESM states that it follows for software that:

> *"The software safety requirements will specify the **behaviour** of the software and its **safety integrity,** which is a measure of the confidence that the software will behave safely."*

Using the above definitions this research proposes that once data is recognised as a separate systems component, these definitions may be adapted for the data component so that *the data safety requirements will specify the **properties** of the data and its **safety integrity,** which is a measure of the confidence that the data will possess the desired properties* [5].

Safety integrity

ESM [6] defines safety integrity as:

> *"The likelihood of a system, product or other change satisfactorily performing the required safety functions under all the stated conditions within a stated period of time"* [6].

1 Introduction

1.1 The use of data by safety systems

Many systems adopt a data driven approach (see definitions) specifically because of the apparent ease of modification of the data component. A data-driven approach provides considerable flexibility as a generic system may be used in a number of different applications. This flexibility also allows the system to be easily adapted in response to changes in the configuration of the system or its environment. In this way data-driven systems offer the opportunity to modify the data component and hence influence the behaviour of the system without changing the hardware and software components. However, this ease of modification is commonly based upon assumptions concerning the design, structure and modularity of data.

The system developer might assume that the safety of the system is independent of the data and that in some way the verification of the software has validated the system for any data, and therefore changes to data are *independent* of the safety of the system and hence require only limited re-verification. These assumptions arise, in part, from the treatment of data as simply part of the software component. In addition there is a corresponding belief that data inherits the properties of good software design. In *high integrity* applications the apparent advantage to be gained through ease of modification of data is illusory, since the effort involved in re-validating the system following modifications to the data is often much greater than would be required for simple software changes. Unfortunately, unless the system has been specifically designed with the objective of simple re-validation in mind, the assumption that changes to data are *independent* of the safety of the system is almost certainly unjustified.

1.2 Proposition: Data as a separate systems component

This research proposes, a data paradigm, that data be treated as a separate system component. The research demonstrates that when data is treated as a separate systems component, this allows the formal apportionment of system safety requirements to data. A consequence of this allocation of safety requirements demands that data is treated with the same rigour as the hardware and software components of the system.

1.3 Aims and objectives of this research

1.3.1 Aims

This research was based upon the author's experience and the perception that data was not being treated with the 'appropriate' rigour. It is the author's experience that the potential influence of data errors on the behaviour of the systems that consume this data is often understated if not completely overlooked.

The aim of this research is to 'raise the profile' of data; to initiate, and contribute to, the debate as to the treatment of data in safety systems. The author has contributed to this debate through publication of a number of papers and presentation to professional forums. It is also intended to provide those safety professionals who assess and analyse safety systems with guidance.

1.3.2 Objectives

The objectives of this research were:

i) To develop a practical and theoretical basis for the treatment of the data component to show:

   a) By synthesis, that one or more data errors may give rise to a hazard that subsequently leads to an accident;

   b) That one or more data errors has contributed to at least one real life accident; and

   c) That the author's perceived view that data is indeed poorly addressed in standards, literature and industrial practice is valid.

ii) To propose guidance for the management of the data component of safety systems;

   This guidance will:

   a) Describe and classify data used by safety systems;

      This will include the development of a data taxonomy and subsequently lead to a statement of the requirements for the definition of data used by safety systems.

   b) Propose a method to derive data integrity requirements;

   c) Propose a model to express the use (and re-use) of data within a systems context;

   d) Propose a generic data and application data lifecycle model;

e) Describe the process of data provision;

The process of data provision will include the identification of the requirements of a generic data supply chain. This research will also propose a method for the design and assessment of the data supply chain.

iii) To disseminate the research through published papers and professional forums.

This dissemination could contribute to any revision of a standard, such as IEC 61508 [1]. A revision of a relevant standard would require practicing engineers to address the data component when reasoning about safety systems.

1.4 The intended audience for this research

The intended audience for this research are:

i) Developers and Practitioners responsible for reasoning about safety systems;

ii) Approval and acceptance bodies – so that they may appreciate the risks associated with data (and data errors) and therefore ensure that data is treated appropriately in submissions for their approval (or acceptance);

iii) Accident investigators – so that they consider data as a separate systems component, and in their search for causal mechanisms may attribute and subsequently record data as a causal factor in accidents; and

iv) Standards setters including those who contribute to good practice guides.

1.5 Structure of this document

The basis for this research is established in section 2. This basis is supported by the description of the Cali Air Accident in section 2 and by a literature search and a survey of industrial practice that are presented in section 3.

Section 4 then considers the definition of data used by safety systems and proposes a generic classification of data used by safety systems.

Section 5 presents a method for deriving data integrity requirements.

Where data is shared amongst a hierarchy of systems the influence and propagation of data errors is described through the use of a layered model. Section 6 describes the use and re-use of data in large-scale systems employing the layered model.

The required data should be validated before use. The Data validation policy may have a significant influence on the structure and complexity of the overall system safety case. Two data validation policies are described in section 7.

The treatment of data as a separate systems component requires that data be treated with the same rigour as the hardware and software components of the system. Therefore data development should be the subject of a lifecycle model. The issues associated with a data lifecycle model are presented in section 8.

The research has also considered the issues associated with data provision. This includes the development of a generic description of a data supply chain, and a graphical representation and a design method for such a data supply chain. The design and assessment of a data supply chain described in section 9.

The discussion is presented in section 10.

Section 11 describes the innovation, review and dissemination of this research.

The conclusions and recommendations are developed in sections 12 and 13.

Appendix A contains a brief description of the Engineering Doctorate Portfolio, together with a suggested order of reading.

Appendix B contains a more detailed description of the layered model.

Appendix C contains a more detailed description of the design and assessment of a data supply chain.

2 The basis for this research

2.1 Data error as a causal factor in an accident

This research is based upon the author's perception that there is a problem with the treatment of data used by safety systems. This perception is based upon the changing nature of these safety systems, in particular the increased reuse of components and in some cases complete applications. Although the safety community has debated the use of Commercial Off The Shelf (COTS) components [9 to 12], this debate has not extended to the use of data by safety systems.

The use of highly configurable systems, whether they reuse components or not, requires that these systems are provided with data that describes the real world. In some cases these systems also use data that describes the use of the system either in the form of constraints or parameters or as descriptions of the service to be provided. Where such a system relies upon the correctness of the data, data errors may subsequently give rise to a hazard that may subsequently lead to an accident.

To support the assertion this section describes a possible mechanism where data error may lead to an accident. The brief case study of Flight AA 965 provides a graphic illustration of the truth behind the assertion that data (and data error) can and does contribute to accidents and resultant deaths.

Additional support is also provided to show that Flight AA 965 is not an isolated incident and therefore the use of data and the consequence of data error are an often ignored aspect of safety systems.

2.2 A theoretical air accident

It is common for the general public to travel by air. Airlines provide frequent flights to many destinations at low cost. This accessibility masks the nature of air travel. A flight is largely described by data derived, in part, from the physical mapping of the terrain and the design and implementation of the description of the airspace. Using this data description, a single aircraft flight will be described by the flight plan. The flight plan will comprise a number of data constructs such as the departure airport, in-flight navigation waypoints and destination airport in addition to aircraft type and capabilities. In modern aircraft the flight plan is entered into the Flight Management System (FMS) and used to automatically direct the aircraft.

If such a flight plan contained data errors it would be conceivable that the data error may give rise to a hazard. In this example the aircraft may fly towards the terrain and if no other mitigating actions took place subsequently crash into the terrain.

2.3 Flight AA 965 – Cali Columbia – December 1995

2.3.1 An otherwise uneventful flight

On the evening of December 20, 1995, American Airlines Flight AA 965 from Miami, Florida, U.S.A., was approaching Aragon Airport, in Cali, Colombia. This was a scheduled flight, the final stages of which were completed at night. Although delayed while on the ground at Miami, the early part of the flight had been uneventful. The Captain had flown 13 times into Cali; the First Officer was on his first flight into Cali [13].

The approach to Cali is along a deep valley between two mountain ranges. Had the flight been completed during daylight hours the crew would have been able to appreciate the proximity of the high mountains on either side of the approach. Cali's Aragon Airport is approximately 3000 feet above sea level, and the mountain ranges on the east of the approach range from 5000 to 6000 feet, but to the east the terrain rises rapidly to almost 13,000 feet [14].

There are two types of instrument approaches into Cali's only north-south runway. One is a VOR-DME approach that is utilised when flights are landing towards the south on Runway 19 (R19). The other, an ILS precision approach for flights landing north on Runway 01 (R01). It was this north approach and landing that Flight AA 965 planned when it left Miami. This meant that their flight path would have taken them over the airport at a safe altitude, then southbound toward the Cali VOR for a few miles before reversing course and flying inbound on the ILS course for a landing on Runway 01 [14].

Upon contacting Cali Control (ATC) approximately 65 miles north of Cali, Flight AA 965 was cleared to descend to and maintain 15,000 feet and report passing Tulua. Tulua is a VOR navigation fix located 34 miles north of the airport, and is the IAF (initial approach fix) or beginning point for the VOR-DME approach to Runway 19. ATC offered Flight AA 965 the opportunity for a direct approach to Runway 19. This was a change to the flight plan previously programmed in the FMS. This was also the initiating event for the accident sequence that was to follow.

Of the 155 passengers, 2 flight crew, and 6 cabin crew on board, only 4 passengers survived the accident.

2.3.2 The accident sequence of Flight AA 965

Having accepted the opportunity for a direct approach to Runway 19, the flight crew had also accepted a high workload to prepare for landing. Part of this workload was to reprogram the FMS for the direct approach to Runway 19.

To program the direct approach the flight crew selected 'R' (for ROZO on the navigation chart) as the entry into the FMS. As a result of this entry the FMS instructed the plane to turn left towards ROMEO NDB. The fight crew detected the turn and started to correct after 90°. A Ground Proximity Warning System (GPWS) alarm was triggered. However, despite a quick response from the flight crew the aircraft crashed into the terrain [16].

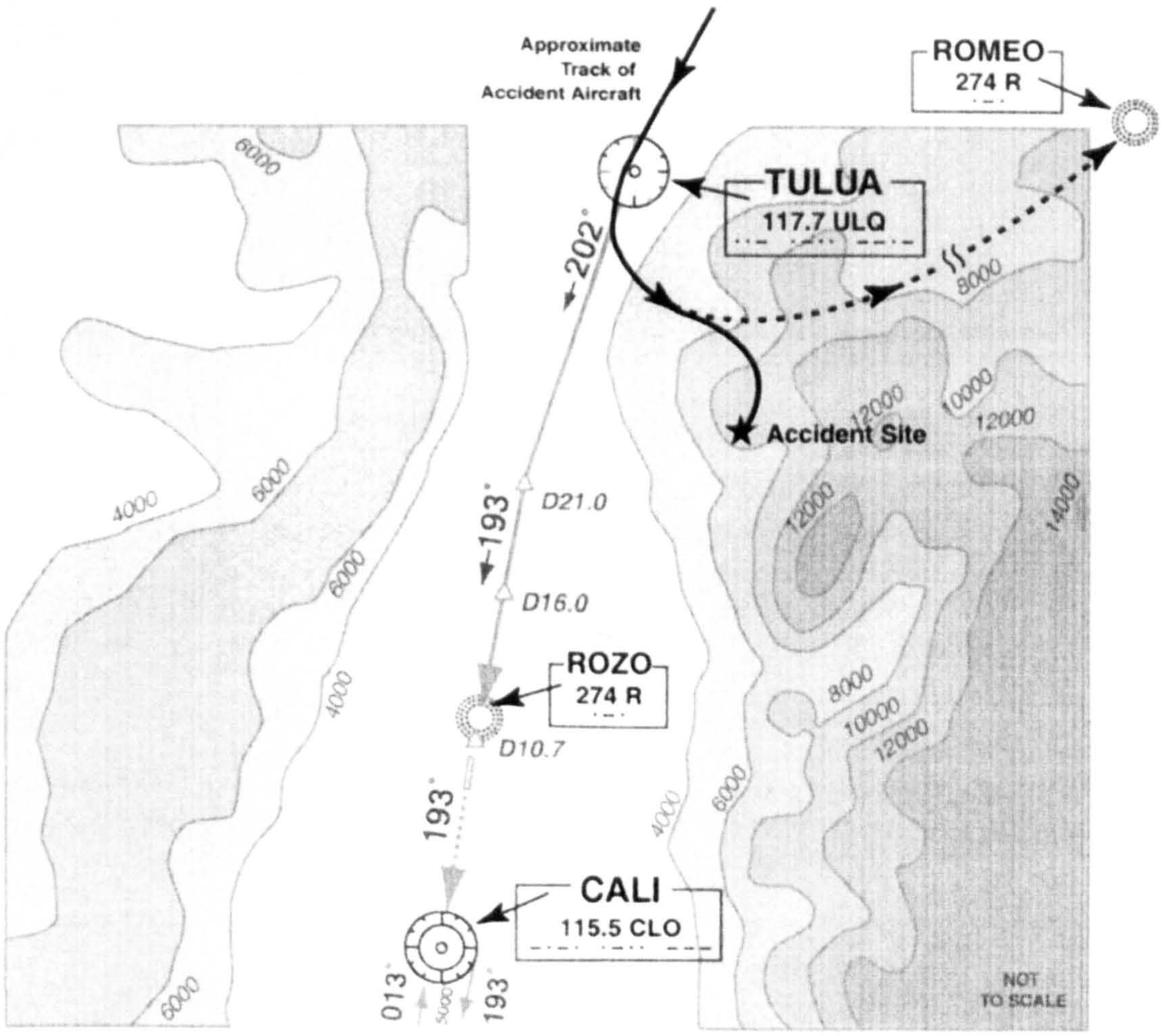


Figure 1: A segment of the navigation chart for Flight AA 965 approach to Cali
Source: Flight Safety Digest [15].

This accident sequence has many contributory and causal factors, but foremost amongst these is the data duplicate for identification of the ROZO and ROMEO NDBs.

2.3.3 Navigational data duplicates in the vicinity of Cali

On air navigation charts each NDB is identified by name and letter. In addition via a radio frequency transmission of the Morse code for identifying letter of the NDB is used. In this case ROZO and ROMEO shared the same first letter 'R' and radio frequency of 274 Khz. Furthermore the man machine interface of the FMS presented the flight crew with 12 other 'R' options, none of which were ROZO [16].

ROZO and ROMEO NDB's are about 150 nautical miles apart from each other.

The navigation data used by Flight AA 965 contained a number of errors; firstly that ROZO NDB was not presented on the FMS display– an error of omission; and secondly that many of the NDBs used 'R' as their first letter - a systematic error and an error of commission as the flight crew was presented with more than one 'R'. The multiple use of 'R' makes the cognitive task of selecting the 'right' NDB more difficult than necessary.

2.4 Flight AA 965 is not an isolated incident

Many accident sequences occur a number of times, only rarely do they result in an accident. This was the case for the Flight AA 965 accident. In a presentation Captain Bertrand de Courville [16] describes a series of other incidents, one of which is based upon a data error in a similar approach sequence after the Flight AA 965 Cali accident.

Data quality, and hence data integrity, remains an issue within the aeronautical navigation domain. Data quality is an issue in other domains as well.

2.5 An initial search for guidance

An initial review of the available literature and a limited selection of standards confirmed the initial view that data is indeed poorly addressed. Theoretical analysis shows that data can give rise to hazards that may subsequently lead directly to accidents. Flight AA 965 confirms that data (and data error) has been a significant causal factor in at least one real life accident and not just in theory.

3 The treatment of data in the existing literature and a survey of industrial practice

3.1 The research environment

The author has conducted this research in parallel with the execution of the full time duties and responsibilities of a Chartered Engineer employed in a number of senior engineering roles. This industrial setting has provided the author with access to other practising engineers.

3.2 The treatment of data in the existing literature

A literature search using the Inspec and Compendex search engines initially only revealed a single paper in this area, this being by Welborne and Bester [17]. Welborne and Bester acknowledge the limited literature in this area and are primarily concerned with the configuration, setting to work, and maintenance of plant and equipment within a nuclear plant. Their focus is on implementation rather than design. Many combinations of searches failed to identify other significant items of literature. In part, this is due to the variety of terms used to reference data and issues associated with data integrity. Some time later a casual remark in a structured interview led to the identification of DO 200A [18].

DO 200A addresses the supply of aeronautical navigation data. DO 200A contains much useful guidance particularly on the identification of scalar data properties and a description of an aeronautical data supply chain.

The literature (and standards such as IEC 61508-3 [19]) contains a considerable array of guidance for the design and development of the software component. This guidance often takes the form of recommended techniques and measures to be used in each stage of the software development lifecycle. In contrast, although data is mentioned several times, IEC 61508 does not recognise the role played by data in many safety-related systems, being concerned only with the data required by algorithms. This restricted view of data may also be underestimating the influence of data errors on the behavior of the system.

An example of this treatment is illustrated by the IEC 61508-3 statement that *"...the design method chosen shall possess features that facilitate ... the expression of ... data structures and their properties"*. While this statement is welcomed it is not consistent with the treatment of the hardware and software components by the standard. In addition the standard says nothing of the methods that should be used to generate or verify data, and it does not identify specific data requirements. This can, in part, be attributed to the systems technologies available at the time IEC 61508 was drafted.

The treatment of data in the standards (and literature) leads directly to the conventional view of data as merely an aspect of software. This conventional view of data has a significant influence on the way data is treated, and may lead directly to poor data integrity.

3.3 A survey of industrial practice

As part of this research a number of structured interviews and reviews were undertaken, primarily in the rail and to a lesser extent the aviation domains, to establish the state of practice. The interviews have identified that data may be drawn from a single source, a situation that would cause significant discussion if the hardware or software components of the system were similarly exposed to such a potential for common cause errors. Based upon the literature review [5], and now supported by this record of industrial practice [20], we can have little confidence that data used by safety-related systems will possess the required integrity.

Practising engineers often required persuasion to encourage them to participate in the survey. On a number of occasions participation was agreed only on the basis of anonymity. Although many more individuals were contacted, once the area of interest was explained many of prospective candidates declined to participate. This was partly due to recognition of the issues concerned and a reluctance to expose these issues within their own work environments or projects.

The results of the survey are also supported in a number of incident and accident reports from a range of industrial sectors. In these reports the author notes that the integrity of the hardware and software components of the system appeared to be adequate. In some cases a lack of data integrity has been a significant factor that has led to harm (people have been killed [13, 21 and 22] and many others remain at risk [23 to 25]).

3.4 A conventional view of data used by safety systems

A conventional view of data tends to pay little attention to the design, production or verification of data. The survey of industrial practice [20] has demonstrated that many engineers concentrate on the *processing* of data, focusing upon the management of error within the algorithms and the software that implements them.

In this conventional view of data, the system integrity requirements are apportioned to the software component, and the system developer may not recognise the influence of data errors on the integrity of the overall system. Without separate data integrity requirements it is difficult to see how a system developer can reasonably demonstrate that their data-driven system will provide confidence that the target failure rates can be achieved.

It is unsurprising that when reporting upon the factors leading to an accident many investigation authorities take the conventional view of data [13]. These reports may simply attribute a data causal factor to *software*. It therefore becomes problematic to find direct references to data in these accident reports without significant re-examination of each accident.

Perhaps the most striking result to come from the structured interviews was the total lack of any uniform approach to the development or maintenance of data. In many cases engineers had not considered data in any detail and had no specific strategy for dealing with the particular problems that it presents.

Treating data simply as an aspect of software leads directly to a self reinforcing argument so that:

i) Data is often not subjected to any systematic hazard or risk analysis.

ii) Data is often not given any specific safety requirements.

iii) Data is often not assigned any specific integrity requirements.

iv) Data is often poorly structured, making errors more likely and harder to detect.

v) Data is often not subjected to any form of separate verification.

The key finding from the literature review and industrial practice survey suggests that data within data-driven systems is often *not* being developed in the same way as application software. It also suggests that in many cases the data is not being developed and managed in a way that would satisfy the requirements outlined in IEC 61508, or other standards, for the development of software.

4 Defining data for use by safety systems

4.1 The use of an extended data dictionary to support data definition

It is good practice for large systems to use a data dictionary to manage the definition of the data component, however as these data definitions become more extensive, the management of complex dependencies within these data dictionaries becomes more difficult to control. This research recognises that although data may be defined within the development arena, this data will be provided by people and processes in the real world.

4.2 The properties of data

An important factor in determining the integrity of a system is the quality of the data that it uses. One of the few standards to consider this topic is DO 200A [18], which is concerned with aeronautical data. According to DO 200A data quality consists of accuracy, resolution, assurance level, traceability, timeliness, completeness and format. It is notable that data described within DO 200A are predominantly scalar quantities. Data may also represent binary quantities and references between single data items, collections of data items or between internal and external references in data models.

It is common when developing large information systems to create a data dictionary, particularly where the proposed system uses one or more data models. The data dictionary usually contains a description of the data model, its structure and the data elements and their attributes. The data element description, its 'type definition' and indeed each data instance maybe owned by separate parties. The extensive use of data from both within, and externally to, the development arena requires additional controls for the definition of data and aspects of data such as ownership, usage, update and validation. To address these additional requirements it is proposed by this research that these data-driven (and data-intensive) systems require an extended data dictionary to include:

i) The origin of each data element, relationship and attributes (this may also provide a description of the requirements for the data supply chain);

ii) The owner of each data element, relationship and attribute (see section 9.10);

iii) A *register of interest* of all those systems which will use (consume) each data element, relationship and attribute together with the integrity requirement of each consuming system;

iv) The system responsible for the updating of each data element, relationship and attribute;

v) A set of rules for the validation of each data element, relationship and attribute; and,

vi) A set of default values to be used in the event of failure to acquire data of the appropriate integrity for each data element, relationship or attribute for each system which will consume the data.

The general requirements of a data dictionary must also be maintained. The data dictionary should be complete and unambiguous. A key objective of the data dictionary is to capture information only once; in such a way that it is available for all. The use of the data dictionary provides a single point in which the data requirements for all the systems may be managed. The data dictionary should also contain an explicit statement of interrelations and data dependencies within the overall system.

4.3 A classification of data used in safety systems

4.3.1 The classification proposed by Welborne and Bester

Welborne and Bester [17] identify the following classes of data:

i) Calibration data

Calibration data, used to define, for example, the characteristics of each input signal such as electrical range, signal range, plant units, alarm or trip levels.

ii) Configuration data

Configuration data, used to define, for example, such items as display screen formats, termination data, plant locations, communications protocols, memory maps, where (the) calibration data are (located).

iii) Functionality data

Functionality data, used to define, for example, what logic operations are performed, what timing considerations apply, which signals are operated for a function, control states to be taken up at failure.

Welborne and Bester used their classification on the installation and commissioning of a large UK Nuclear plant. As such the classification reflects the requirement to combine existing plant and equipment into a process control system. The separation of Calibration Data reflects the need to address time based properties, such as drift or changes in physical effects due to ageing.

However the author feels that this Calibration Data is more suited as a subset of Configuration data. Indeed the locations of these data items are already included within the Calibration Data class.

The Welborne and Bester Functionality class contains elements that are more suited to a treatment as limited variability software, and as such are already addressed in standards associated with the process industry (IEC 61511 [26]).

Welborne and Bester are concerned with a system that has a 'single' use. It would be undesirable to implement nuclear process systems as 'agile' reconfigurable systems. As such Welborne and Bester describe a largely fixed system. A more generic approach needs to address a data class that contains a description of some future use of the system as a schedule or timetable. In addition a generic system should address the requirements of data derived not only from local systems but also data derived from external information systems.

4.3.2 The classification proposed in this research

A classification of data is required to provide a means by which the role of data may be described and by which potential data errors may be analysed. A useful tool in the classification of data is the 'data box model'. This facilitates the expression of the data integrity requirements as data passes across the boundary of the system or system component. The author believes that the box model presented in Figure 2 is applicable to a wide range of systems.

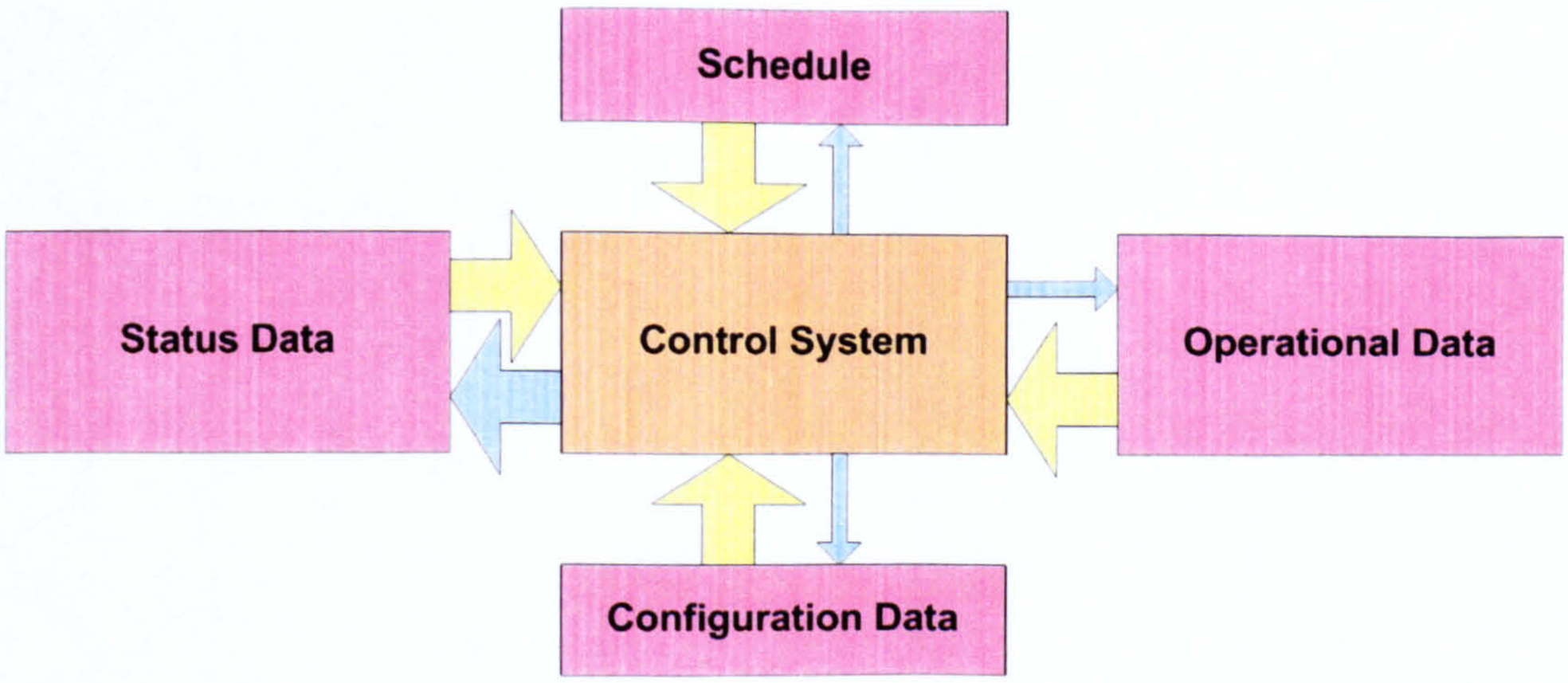


Figure 2: Data in the context of a control system

The box model identifies the system as a single box. Data that crosses the system boundary (in either direction) may contain errors and subsequently give rise to hazards. This data classification consists of four categories of data (configuration data, operational data, status data and schedule data).

These categories provide the basis of an analysis to establish the data integrity requirements by consideration of the types of failure that each of these forms of data might exhibit. These four data categories are described below:

i) Configuration data

Configuration data is a description of the infrastructure and its capabilities and constraints. In the rail industry, the infrastructure includes, in general, both fixed assets and mobile equipment such as vehicles. This description is most conveniently regarded as "*static*" data in that it represents the entities in the real world, which change only in response to the action of maintenance or modification of these entities.

Static configuration data is also used to configure the hardware and software components and may also provide the characterisation of these standardised components including parameterisation that describes how these components are to operate. For example, configuration data may be required to describe the physical connections between hardware components, and configuration data may also be required to describe the logical software connections, devices and services.

ii) Status Data

Status Data is provided through interfaces to external reporting systems and direct status information from connections to local sensors and other inputs.

iii) Operational Data

Operational Data represents the individual operational conditions perhaps communicated to the control system via manual input. In the rail industry, the operator receives these operational conditions through human communication interfaces such as telephone and fax. The set of operational conditions represents persistent restrictions on the use of the infrastructure. These operational conditions may arise through adverse weather conditions, equipment failures or reports of accidents.

iv) A Command Schedule

A Command Schedule is used to describe the required use of the infrastructure. For example a railway control system would use a train schedule to describe the planned movements of multiple trains across the rail infrastructure and an ATC system a set of Flight Plans.

4.3.3 Using a data classification within a systems context

It is common for an application to cooperate with other applications and subsystems within an overall system. In such a systems context the role of an individual application, data structure or data element may be obscured. To aid the visualisation of the role section 6 describes a layered model.

Where data is shared by several systems it becomes important to classify data and to separate out other elements such as limited variability software. Such limited variability software often acts as 'glue' interposed between elements within the system. This 'glue' seeks to adapt the imperfections and shortcomings of the juxtaposition of differing (often pre-existing) applications within this system. Welborne and Bester include these 'glue' software elements within their definition of data.

This research recognises these software elements as software and requires that they be treated as simply an aspect of the software component. In addition this research recognises that some data driven systems consume data in the form of a schedule. It also recognises that this schedule data describes some future use of the system and as such manipulation of the elements within this schedule data may affect the safety of the overall system. Manipulating schedule data may influence the probability (or likelihood) of an event through either an increase or decrease in the time separation of schedule events. Where such events represent the movement of trains or airplanes these time separations may be equated to distances and hence possible accidents.

This research asserts that the classification of data presented in section 4.3.2 can be applied to a range of applications (and systems) contexts.

4.4 An assurance model for data used in safety systems

When considering data definition, provision should be made to facilitate a demonstration that the data integrity requirements have been, or indeed could be met. Therefore alongside the data definition should be a description of an assurance model.

The assurance model described below is based upon the concepts and terminology used in the UK CAA document Air Traffic Services Safety Requirements (CAP 670) section SW01 [27]. The relevant software concepts from CAP 670 SW01 have been adapted for data to give:

i) The concept of Assurance Evidence Level (AEL) to express the level of confidence that a data component will possess the integrity according to its specification on the basis of the strength and depth of the available evidence;

ii) The five fundamental safety objectives which a safety-related system must fulfil, namely *requirements validity*, *requirements satisfaction*, *traceability*, *configuration consistency* and *non-interference* with safety functions by non-safety functions;

iii) The concept of *direct* and *backing* evidence; and

iv) The requirement to define the integrity of a data component in terms of a defined set of attributes.

The assurance model should be developed for all aspects of data used by safety systems including each of the phases of the data supply-chain (discussed in section 9). The Assurance Evidence Level as defined in SW01 is not a measure of reliability, but a measure of confidence that a component satisfies its requirements. A safety-related system will have specific safety reliability requirements, which may be expressed in the form of numeric failure rate targets, or as a Safety Integrity Level (SIL).

4.5 Data definition in the data supply chain

It is proposed by this research that requirements specifications are required not only for the data but also for each element of the data chain, and the tools that are used to manipulate or verify data representations. These data requirement specifications form the data definition. The process of data definition should be used to identify any common data elements, including their respective quality and integrity requirements. See section 9 for a description of a data supply chain.

5 Deriving data integrity requirements

5.1 The use of integrity requirements as a measure of criticality

Integrity requirements are often used as a measure of the criticality of a system. In addition IEC 61508 [1] uses a classification of integrity requirements as a basis for the recommendation of techniques and measures to be used during the development of the system. This section presents a method for deriving data integrity requirements.

5.2 Integrity requirements based upon numerical targets for failure rates

Integrity requirements are commonly expressed as numerical targets for failure rates for each function of the system or sub-system. To attain the required integrity level the system should not only be developed using the techniques and measures recommended by standards such as IEC 61508, but this integrity level should also be demonstrated by the system whilst in service. This demonstration will be through the achievement of the appropriate target failure rates while in service, when supported by appropriate maintenance procedures.

In basing the integrity requirements upon minimum failure rate targets, standards such as IEC 61508, take no account of the scale or size of the system. The larger the system to be developed, the greater the amount of effort required to achieve the minimum failure rate. Therefore the development of larger systems requires improvements in both process and design (architecture) to achieve these minimum failure rates.

5.3 Adapting Functional Failure Analysis for use with data

To establish the integrity requirements for a given system it is important to identify the functional behaviour of the system and the mechanisms by which the identified functionality may fail. As each failure may give rise to a hazard, the potential harm should be established for each system hazard. An example of a technique that might be used to determine integrity requirements is Functional Failure Analysis (FFA).

The basic FFA process (proposed by this research, as it applies to data errors) is to:

i) Identify the functions;

ii) For each identified function, suggest failure modes (based upon data errors), using the guidewords (see section 5.4);

iii) For each failure mode, consider the effects of the failure on the system (this may require the development of a number of operational scenarios); and

iv) Identify and record any actions necessary to improve the design.

5.4 A proposed set of guidewords to be used with Functional Failure Analysis

This research has developed sets of guidewords, adapted from SHARD [2], and work by Harrison and Pierce [28] to be used as prompts in the consideration of possible failure modes (based upon data errors).

Table 1: Guidewords used in the consideration of possible data errors

Data Classification	Guidewords
Configuration data	Omission, Spurious data (commission), Positioning, Topological, Addressing, Type, Labelling, Value [28]
Status data (connected equipment)	Mode, Sequential, Combinational, Propagation, Timing, Volume
Status data (external systems)	Existence, Reference, Availability, Inconsistent, Timely
Operational data	Existence, Reference errors, Type errors, Inconsistent, Timely, Mode error, Sequential errors, Combinational errors, Propagation errors
Schedule data	Existence, Reference, Availability, Inconsistent, Timing, Propagation

5.5 Apportionment of the system safety requirements to data

Integrity requirements may be expressed in the form of an allowable failure rate and this can be considered as a failure budget for the system. The definition of the system architecture will have played a large part in the determination of the integrity requirements and is used to apportion the failure budget between random and systematic failures. The hardware component will be apportioned the random failure rate budget.

While integrity requirements are more than just a set of target failure rates, these targets *are* of importance. In an arrangement consisting of several component parts in series (in other words, all of which are necessary for the correct functioning of the system), the overall number of system failures will be equal to the sum of the system failures produced by each component. Therefore, if a system consists of hardware, software and data elements, the overall target failure rate may be apportioned to provide separate failure rates for each element. This implies that in a data-intensive system of a particular SIL, one aspect of the data integrity requirements should be that data errors should not produce system failures at a rate greater than that allocated to the data component. It also implies that the data will require a target failure rate that is *lower* than the figure given for the corresponding SIL.

6 The use (and re-use) of data in large-scale systems

6.1 A requirement for a model to assist in the visualisation of large-scale systems

Large-scale systems tend to be complex. A major constraint in the visualisation of such systems is the variability of often hybrid architectures. This variability leads to a requirement for standardisation, not least in the visualisation of such systems. This research proposes the use of the layered systems model presented in section 6.2. A further refinement is the adaptation of this layered model to show the use (and re-use) of data in such large-scale systems.

6.2 A layered model for large-scale systems

6.2.1 A railway as an example of a large-scale system

An operational railway is a complex system. Even the most brief of examinations of Figure 3 reveals the interrelated nature of a range of engineering disciplines, that all need to come together to form the operational system. These engineering disciplines are track, Civil Engineering Works, electrification (traction power) and plant, rolling stock and signalling, to name those that are most readily identified in Figure 3.

Figure 3: An operational railway as a large-scale system

When considered on a national scale, an operational railway is also a large-scale system. In order to reason about such large-scale systems we need to create a framework (or model) to express the role of the constituents of the system. Perhaps, by way of introduction, the best way to explain such a framework is to start with an explanation of the components of a railway.

A railway, in its simplest form, is used to transport people and goods from one geographic location to another. A pictorial representation of the railway is shown in Figure 4.

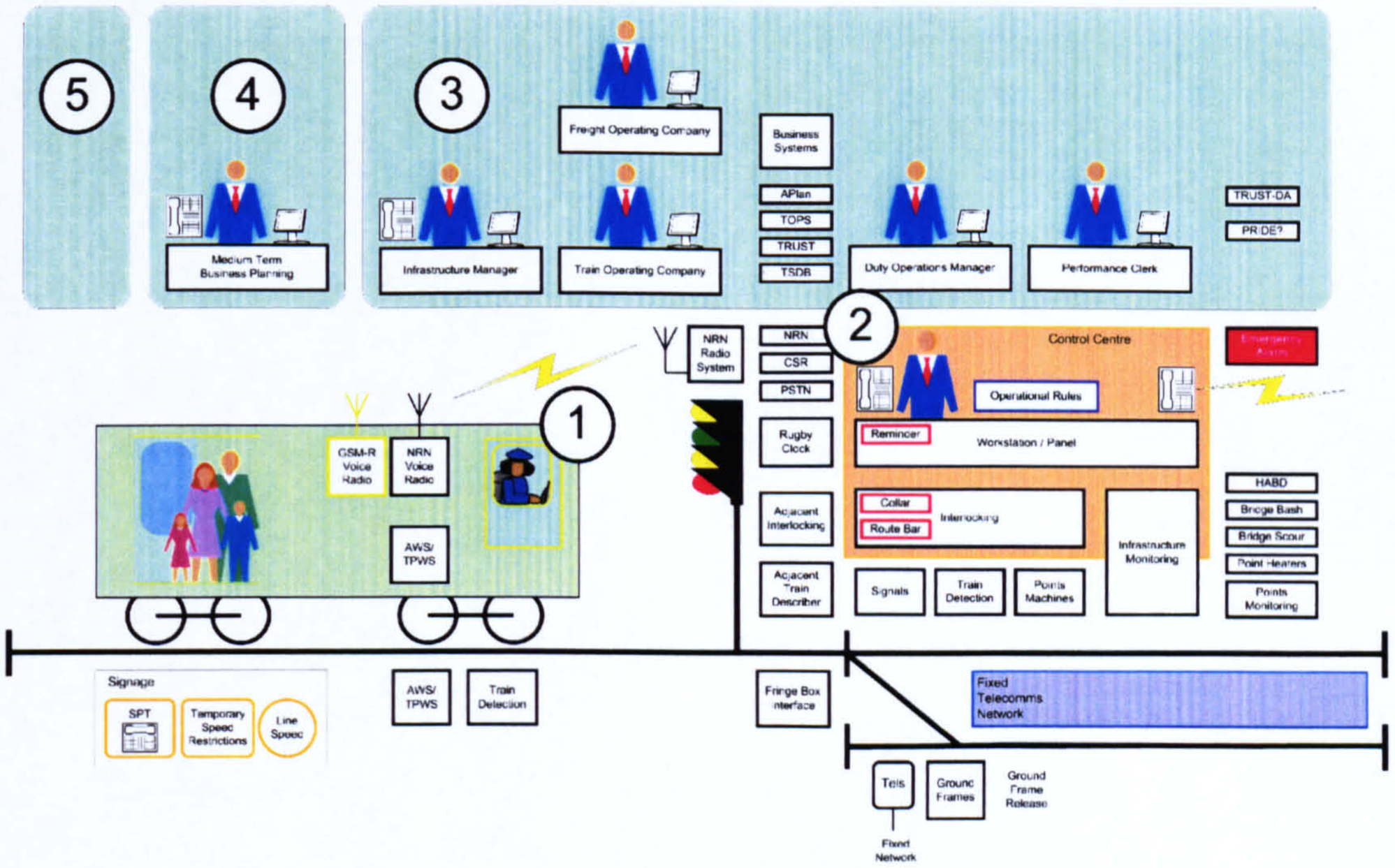


Figure 4: A pictorial representation of an operational railway

The railway may be classified as a guided rail system, where the train (1) passes over the rails as directed by the signals controlled by the signalling control centre (2). The signalling control centre, amongst other functions, executes route setting and the 'regulation' of trains across the rail network. The signalling control function manages a control area and is managed by regional (or zone) organisations (3). The regional organisations execute the strategy and policy set out by the enterprise (4). In a highly regulated environment governmental bodies such as the Office of the Rail Regulator (ORR) (5) may determine some of these policies and strategies.

Each of these 'layers' exchange information either as data or as control actions. To further describe these control actions consider the signalling control function within the signalling control centre.

6.2.2 The systems hierarchy within the signalling function

The systems hierarchy within the signalling function is depicted in Figure 5. Within this systems hierarchy status information from the plant (1) is passed via the plant interface to the 'interlocking' (2). The interlocking sets points and signals to enable the safe passage of trains and is a rule based protection device that may contain features, usually implemented as tokens, to disable parts of the interlocking and hence restrict trains from access to parts of the rail network.

These tokens may take the form of collars and may be placed by the signalman or 'route bars' entered through the technician's terminal (for more persistent tokens).

Status information is then passed to the signalmen's panel or workstation (3). The signalman (4) may use reminder devices such as notes as an additional reminder of operational restrictions.

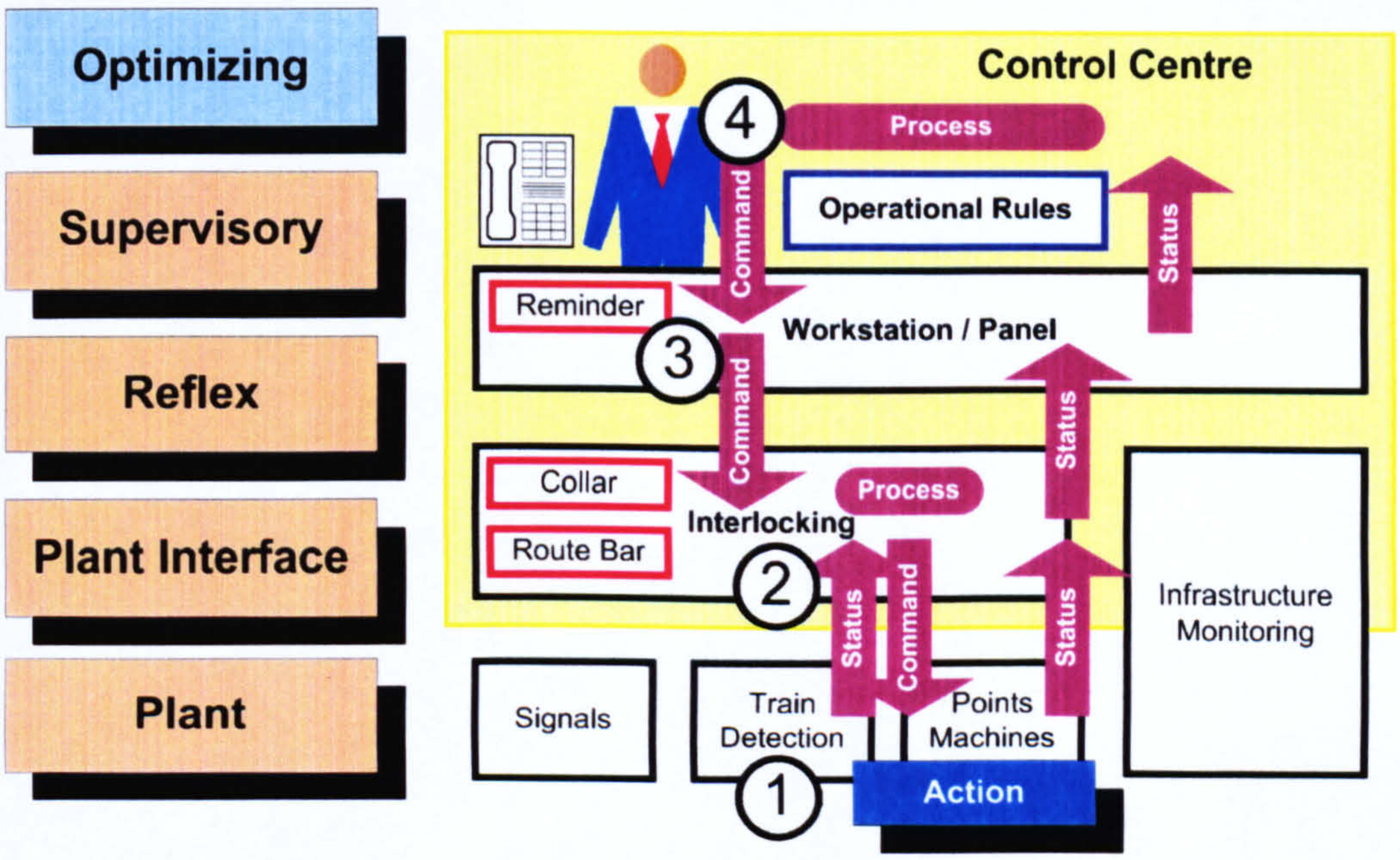


Figure 5: The signalling function within a railway control centre

The signalman regulates trains through a combination of judgement, competence, experience and the use of the workstation (or panel) to issue a command to the interlocking. The interlocking checks its rules before accepting the command and issuing one or more commands via the plant interface to the plant, in this example commanding a points machine (switch) to move.

The layered model presented on the left of Figure 5 presents the signalling function in a number of layers [29]. A more detailed explanation of the role and function of these layers is contained within Appendix B

The author considers that this layered model is generic and that may consequently be applied to other functions and systems. The author concurs with Richard Allan [29] in considering that the extent of any safety function should be limited to the first four layers (plant, plant interface, reflex and supervisory). If this limit is used then the signalman can optimise the delivery of the train service by using whole safety functions. In addition the use of whole safety functions may reduce the complexity of any safety argument concerning the signalling function to a reasonable minimum.

6.3 A layered model to express the role of one or more systems within a hierarchy of systems

The form of distributed systems varies tremendously; one can gain an insight into the interaction between system elements by considering a typical distributed control system. For this purpose, section 6.2 has introduced the very specific example of a large-scale railway control system, though it should be noted that not all systems would follow this scheme. The model is described in some detail in Appendix B

This abstraction into layers allows the development or replacement of the underlying layers based upon a respect for the services provided by each layer, and the preservation of the interfaces at each layer boundary. A system possessing low coupling and high cohesion will generally possess the desirable design properties of resilience and stability. Low coupling and high cohesion also minimises the effects that changes to one component have on other components of the system. The value of these properties has been considered self evident for both good software and good systems design, appearing in many texts. However anecdotal evidence suggests that the value of these design properties has not been recognised in the design of data used by data-intensive systems. Data, in common with hardware and software, is a system component and broad parallels can be drawn as to the desirable properties for good design of the data component of a system.

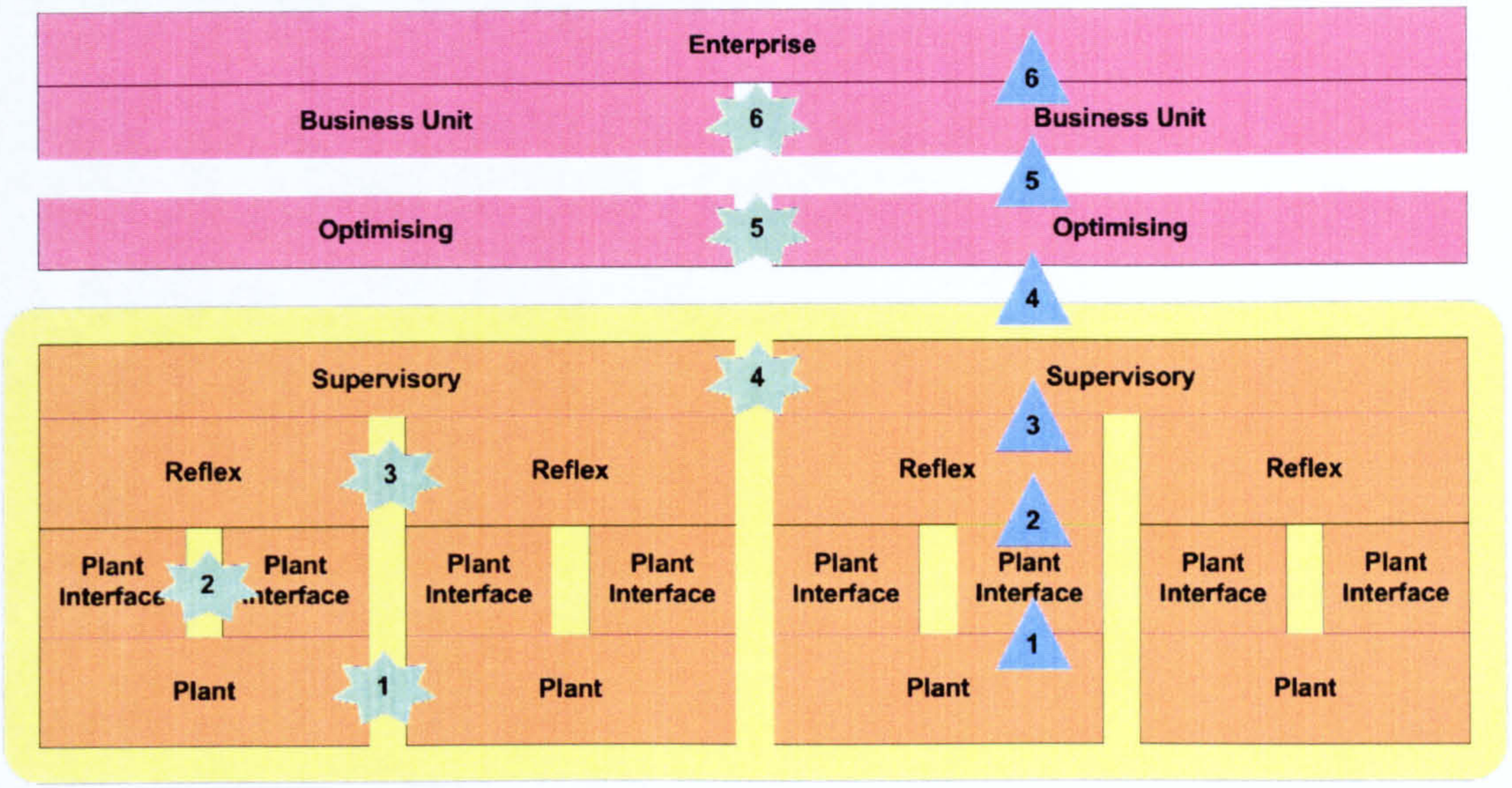


Figure 6: A layered model for a hierarchy of systems

The model, in Figure 6, depicts a control system as a hierarchy based upon the role and nature of the functional layers within the system.

A typical control system may require many human operators. Each operator will usually be responsible for one aspect of the system such as one or more control areas. An example is Air Traffic Management where an aircraft crosses many control sectors (areas) during its flight. The delivery of the planned services requires the provision of a seamless control of the infrastructure when handing over control from one control area to the next. These operators may also be supervised. In this context, Figure 6 could illustrate two operator positions, which may control the infrastructure within the same '*control centre*' or in separate '*control centres*'. The intention of Figure 6 is to illustrate both horizontal and vertical coupling within a realistic example system.

This research identifies vertical coupling via the numbered triangles; horizontal coupling via the numbered star shapes. Both vertical and horizontal coupling may be applied to services provided at each layer and between peer components within the layered model. In the context of data driven systems this research proposes that vertical and horizontal coupling may be applied to data used and produced by applications or complete systems within the layered model.

6.4 The use of the layered model to expose the use (and reuse) of data

In the context of this research the layered model is used in conjunction with the data classification (see section 4.3.2). Each system, plant or equipment will contain varying amounts of each of the classes of data. Those components at the lower levels of the hierarchy are less likely to use the schedule data. The

components in the mid and higher parts of the hierarchy are more likely to use schedule data.

Within this systems hierarchy data is exchanged across external interfaces, and across internal interfaces. In such a hierarchy data may be used within several layers. Each system component may use this common data in a different context and hence re-used data may be also attributed subtly different meaning for each usage. The layered model allows the visualisation of the extent of the influence of these data structures, data elements or data items. This visualisation may identify a requirement for validation at an interface to preserve the integrity of one or more safety functions. One aspect of the interpretation of these visualisations is the use of rules for the exchange of data between systems.

6.5 Rules for data exchange between systems within the layered model

The rules for data exchange between systems are derived from practices common to many *safety-related* standards. The requirements presented below are adapted and extend from Petersen's DISC report [30]. Data should only be shared amongst systems when the data integrity requirements of each consuming system are fully satisfied.

The requirements are that:

i) The data integrity requirements of all sub-systems or applications within the system are documented;

ii) Data may be passed from a higher integrity system to a lower integrity system (provided that the data from the higher integrity system exceeds the data integrity requirements of the lower integrity system for each of the data elements passed across the interface, including error rates and error modes);

iii) Data may be passed between systems of the same integrity requirements if and only if these data integrity requirements are compatible, including error rates and error modes;

iv) Data may <u>not</u> be passed (without verification) from a lower integrity system to a higher integrity system unless data integrity requirements are compatible, as this low integrity data, by definition, may contain a data error rate greater than that required by the high integrity system; and

v) The hardware and software components of these systems meet the integrity requirements for each system.

7 Validating data for use by safety systems

7.1 Proposing a data validation policy

Common sense dictates that data is validated before use. This section presents two validation policies and notes their influence upon the structure and complexity of the safety arguments associated with the data component.

7.2 Data passed across the system interface

The extent and nature of resources required to validate data are strongly influenced by the integrity of the source data as well as the integrity of the processes used to transport this data to the systems which will consume it. This is a multi-facetted problem; on one hand are small-scale systems whose data can be adequately managed through data entry and delivery of a validated dataset. At the other extreme are large-scale systems drawing data from a number of sources. This data is processed, transformed, consolidated, transported and finally delivered to one element of the overall system. An example of such a system is Air Traffic Management (ATM), where a number of control systems share common data such as aircraft type or 'adaptation data' describing the airways. A second facet is the maturity of the application. New systems may require completely new datasets, whilst new implementations of existing systems may re-use existing data.

Irrespective of the scale of the system, data may be passed across an interface at the system boundary. This data is likely to be transformed from the external representation into its internal representation. This research proposes that this data should be verified either through an automated (but not automatic) process. This automated process is likely to require supervision and where necessary manual intervention and is depicted in Figure 7.

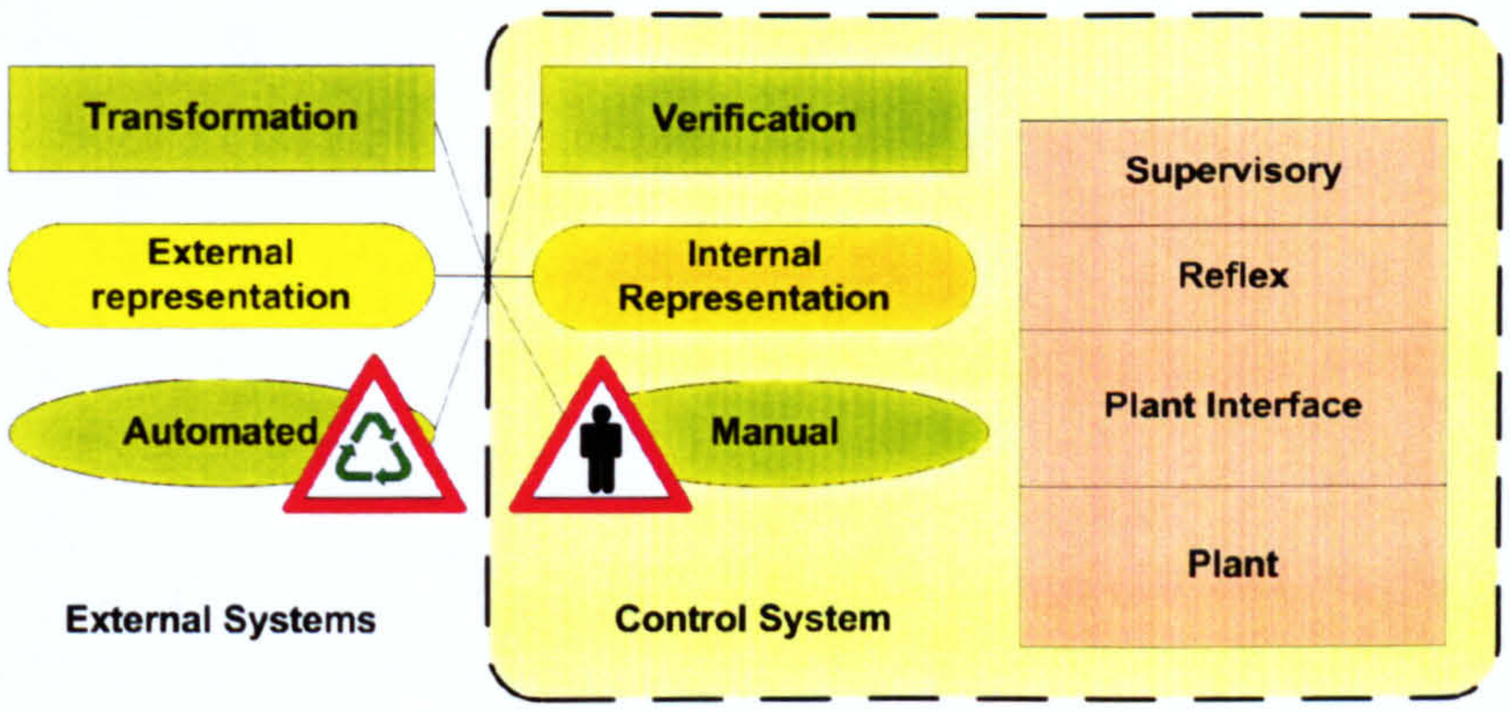


Figure 7: Data passed across an interface at the system boundary

This research notes that the point at which data validation takes place is likely to have a significant influence on the structure and complexity of the safety argument for the data component of the overall system.

7.3 Data validation at point of data entry

Data may be extracted from existing data sets or created at data entry. In this research the term 'data origination' is used to describe the origin of data.

Tillotson [31] observes an approach where the responsibility for data integrity may rest at the data source with data entry. However this may be three, four or five systems away from where the data is used.

7.4 Data validation as the responsibility of the consuming system

An alternative policy is proposed by DO 200A [18] where the responsibility to ensure that the data integrity requirements have been met rests with the user of the data.

7.5 Data validation policy and its influence on the system safety case

The system safety case should consider all aspects of the system, its behaviour and the processes used in its creation, operation and maintenance. In a system where a conventional view of data is taken, the safety case will address the hardware and software components of the system. Where data is treated as a separate component, the safety case will also explicitly address the data component. The processes for the design and development of the data component will have many similarities with those processes used to design and develop the hardware and software components of the system. However this is where the similarities end. Elements of the data component may be subject to continual change, being updated or modified as a normal part of the operational system. The system safety case should address this aspect of the data component.

A data validation policy, which only considers validation at data entry, extends the effective system boundary to include the processes, procedures and people that execute data entry. This extended system boundary significantly increases the complexity of the safety argument for the data component. This complexity is further increased where the safety argument must also consider the rules for the exchange of data between systems, developed in section 6.5.

In contrast, this research proposes a data validation policy where the responsibility to ensure that the data integrity requirements have been met rests with the user of the data encourages those responsible for the system to identify and define a boundary. In defining the boundary, this allows evaluation of the complexity of any proposed safety argument for the data component.

8 Developing the data component of safety systems

8.1 Developing data-driven systems

The design, development, implementation and maintenance of data-driven systems are complex undertakings. These complexities come about in part through the shared use of common data definitions and datasets. The system designer is faced with an extensive array of options when selecting a solution to any particular problem. In many instances the designer is constrained by the problem context, which may also contain existing solutions.

8.2 Data development

The treatment of data as a separated systems component requires that data be treated with appropriate rigour. This requires the developer to address data development. Perhaps one approach is to examine the provision of data for safety systems through the development lifecycle and then to examine data provision outside the development arena.

8.3 A development lifecycle for data

8.3.1 Data developed within the system development arena

The conventional view of data tends to pay little attention to the design, production or verification of data. It is argued elsewhere in this executive summary that data should be treated as a separate system component, and from this argument it follows that this data component should be developed with the same rigour as the other components of the system. Logic then dictates that the data component should also be subject to a development lifecycle model. This research recognises that data may be drawn from sources external to the development lifecycle and may also be provided either through a data production process or extracted from external information systems. Once obtained this data may be processed within a data supply chain to create one or more application datasets. These application datasets will then be installed to form the complete build for one or more instances of the system. In addition data-driven applications may be used (and re-used) in many installations and therefore the design of the data component may be separate from its implementation.

This research recognises that the size, scale and complexity of modern systems development demands that the series of processes and activities required to create these systems are set out so that effective control may be applied to the system development. This research uses the 'V' lifecycle model as the basis for

the discussion of the lifecycle issues affecting the development of data. This data development is not only concerned with the data architecture but also with the means of data provision. The design, implementation and provision of data may require extensive co-ordination of activities, processes and personnel far beyond that which can be shown in a simple 'V' lifecycle model.

The 'V' lifecycle model is used to describe how a development project moves from requirements, through system design, component design, component implementation, component integration (assembly) into system integration and finally system test. A particular feature of the 'V' lifecycle model is the linking of the left hand and right hand sides of the 'V' (as shown in Figure 8 below). In the 'V' lifecycle model the requirements are stated at the commencement of the project at the top-left-hand of the 'V' lifecycle model. The activities associated with the demonstration that the requirements have been satisfied are concurrently defined at the top right hand side of the "V" lifecycle model. In this way the top-left-hand activities of the 'V' lifecycle model are linked to the top-right-hand activities of the 'V' lifecycle model.

As the project progresses down the left-hand side of the 'V' lifecycle model, each level of the 'V' is linked with activities on the right hand side. For example system design would be linked to system test, subsystem design would be linked to subsystem integration and component design linked to component test. Design reviews will be held at identified points on the 'V' lifecycle model to provide assurance that the development process have been executed with appropriate rigour and that the project deliverables are available for examination. The design reviews on the left-hand side of the 'V' are also linked to review points on the right hand-side to demonstrate that the requirements associated with each level of the development have been delivered.

This research has developed the following generic and application data lifecycle models. In following the pattern set out by the software 'V' lifecycle model, similar hazard and risk analyses are carried out throughout the data development. This requires an explicit statement of the data safety requirements, the data architecture, the data design, and the data integrity requirements. The data architecture and data design are therefore significant factors in the determination of the data integrity requirements, as would be the case for the software and hardware components of the system.

A possible development lifecycle for the data component is shown in Figure 8 below.

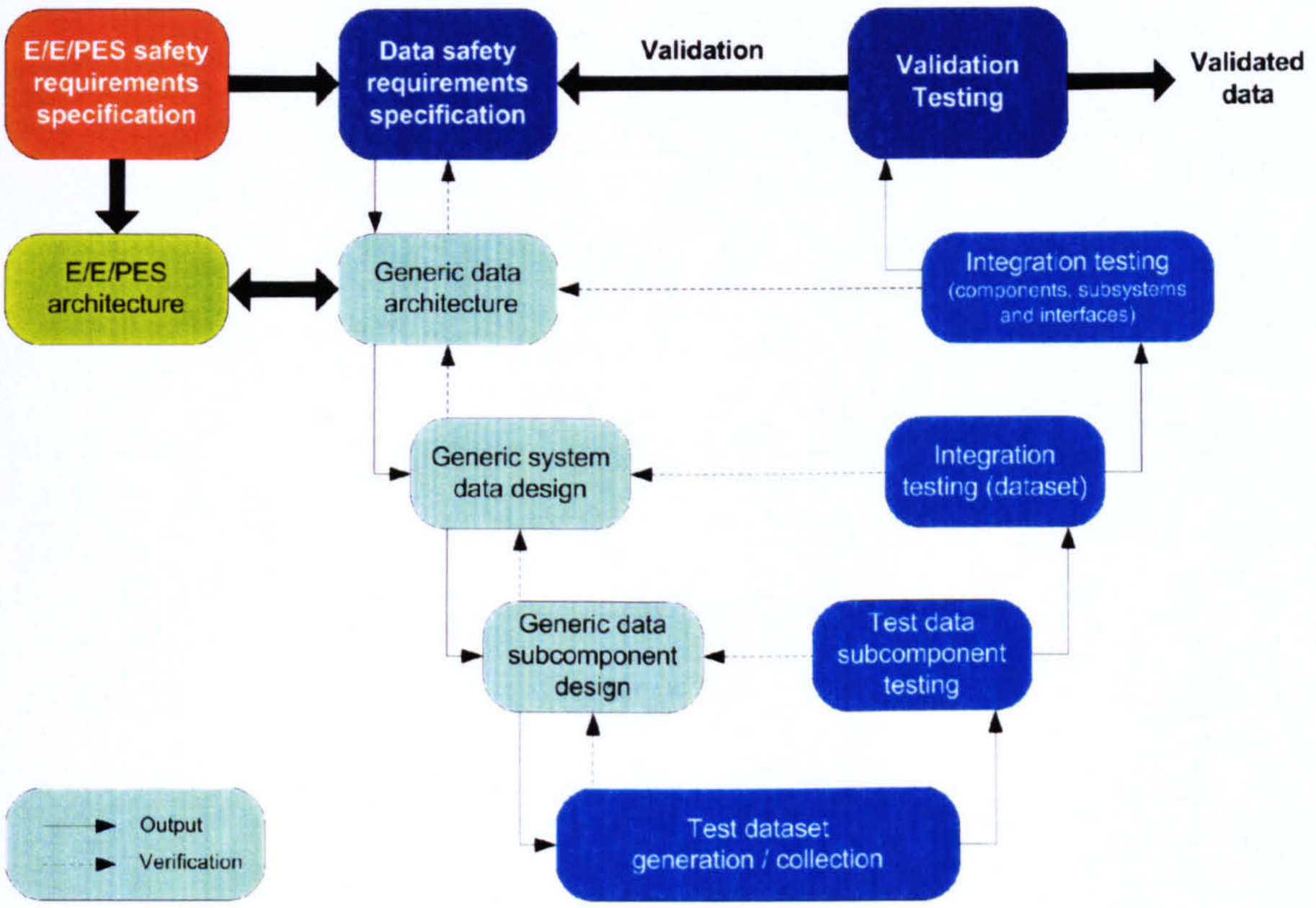


Figure 8: A generic development lifecycle for the data component

8.3.2 Data generated (or collected) outside the system development arena

Implicit in the data lifecycle, in Figure 8, is the consideration that the data will be generated (or collected) at the same time as the system development. While this may be true for some data components, a majority of the application data is likely to be produced or collected outside the development environment. This is based upon the assumption that multiple instances of a data-driven system may be employed in order to reuse the hardware and software design (component or complete application), either because the system is designed as a product for multiple customers, or through informal re-use. Therefore the same group of development products may be customised through the use of data to the specific requirements of an individual application.

To accommodate these data requirements demands that the data development lifecycle be modified to accommodate this change in intended use. The modified lifecycle should recognise two distinct phases of system development; firstly the development of the generic system and secondly the application development, as shown in Figure 9.

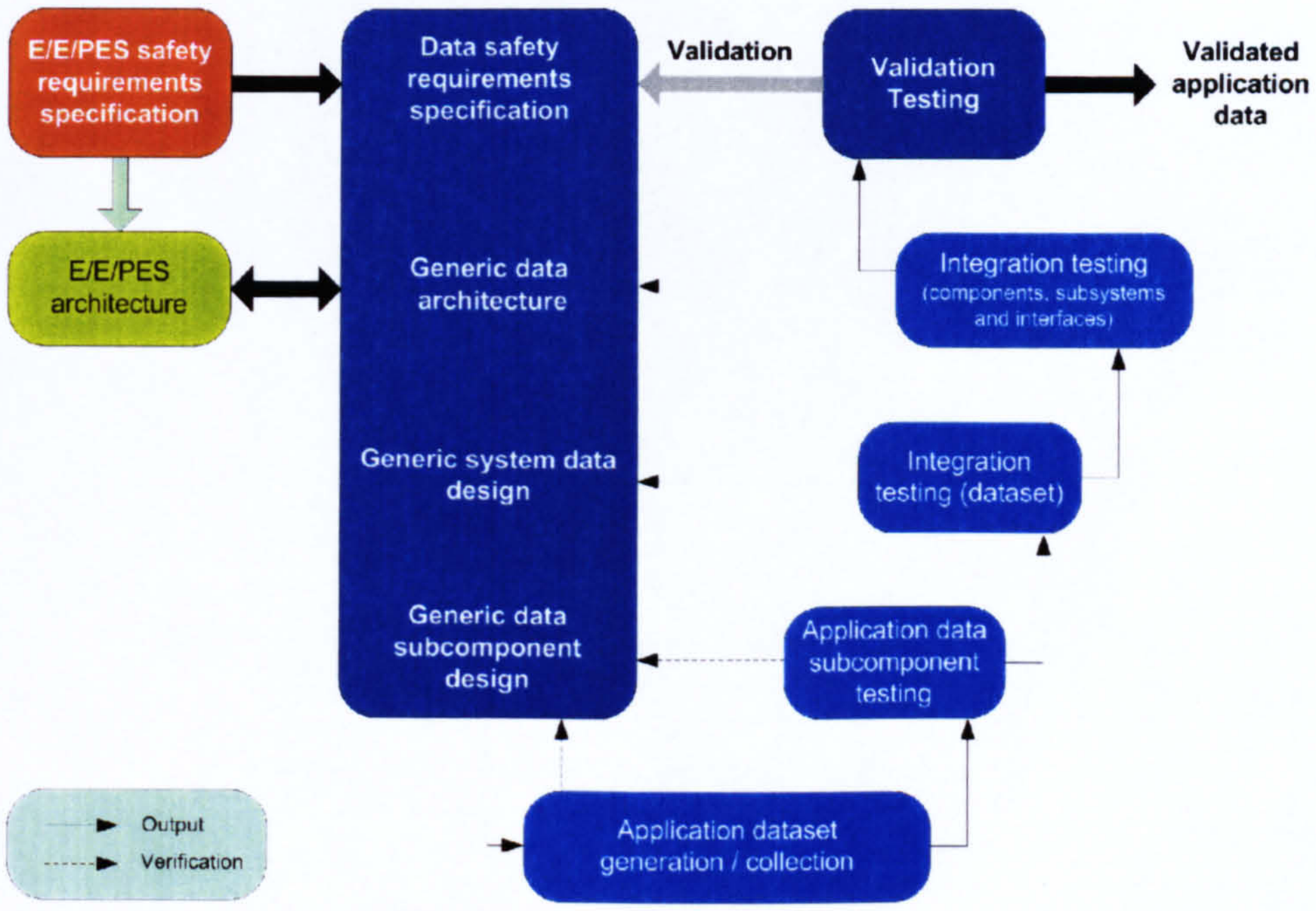


Figure 9: An application *data lifecycle*

This application data lifecycle highlights a number of difficulties with this approach. The application development assumes that the system development deliverables are available and contain all the required information as to the data required, including any sensitivity to data error, omission or commission. If the circumstances of the application development identify additional hazards or changes in the frequency of occurrence or severity of outcome of any possible accident, then the manner in which changes to the system or its hardware, software or data components are to be affected are unclear.

8.4 The integration of the generic and application data lifecycle models within a system development

The 'V' lifecycle model is essentially a linear model, which assumes that development progresses in an orderly fashion, each stage being largely complete (or more commonly expected to be complete) before the next stage begins. The 'V' lifecycle model over-simplifies the development and production of data. Additional data requirements may also emerge from the design as requirements of the development of the hardware and software components of the system. This is in part due to the use (and re-use) of generic hardware and software components that require configuration data.

This research has developed a timeline model to illustrate the integration of the hardware, software and data components with the development of the overall system. In Figure 10, the development of the overall system is contained within the Systems Engineering activities. This research proposes that the system and

the data architectures are created as part of the systems engineering and application development activities. This facilitates review of the overall system and data architectures as part of an integrated development process. These reviews are shown as the Preliminary Design Review (PDR) and Detailed Design Review (DDR) along with other lifecycle stage gates in Figure 10.

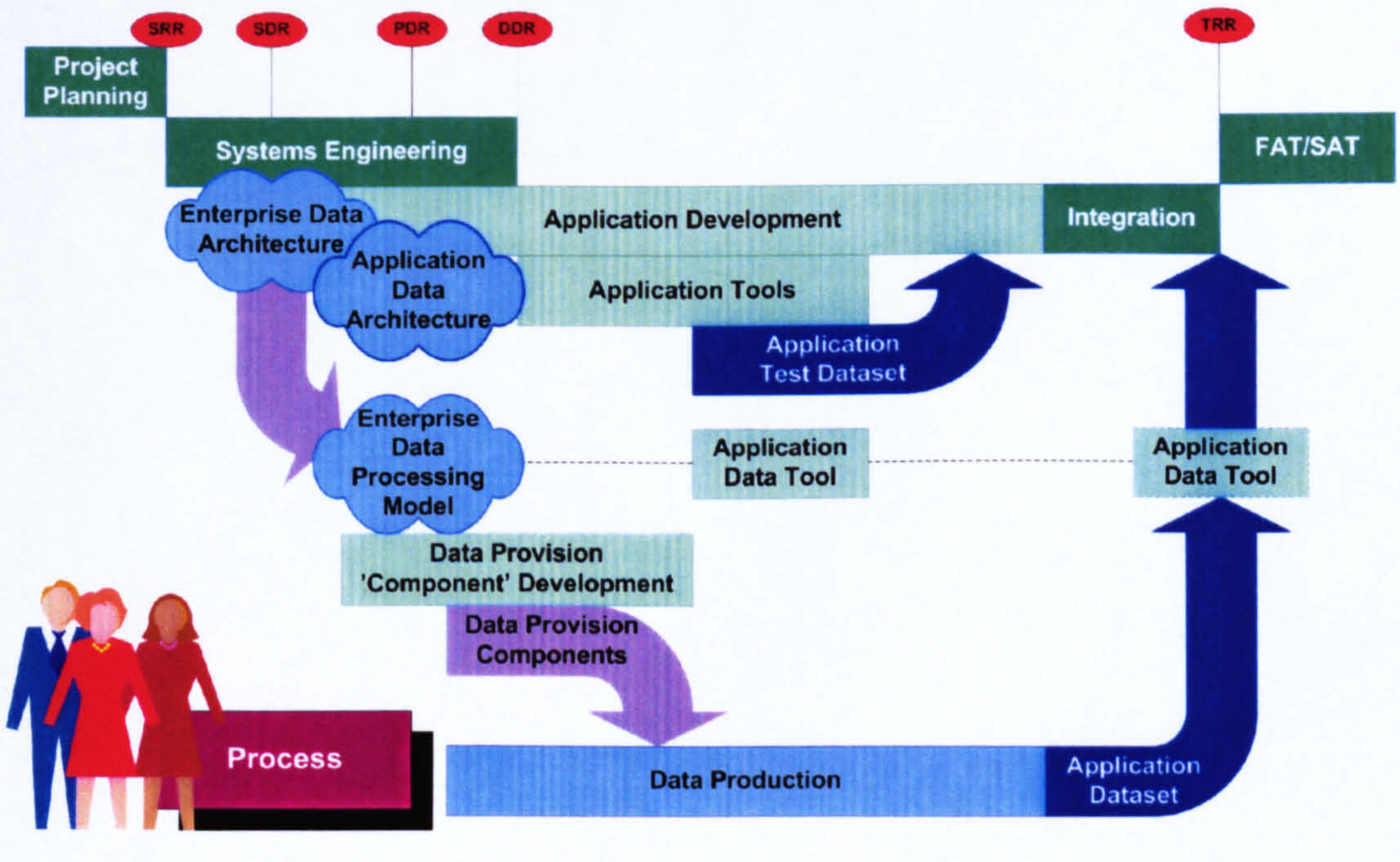


Figure 10: A system development timeline

The successful completion of the Systems Engineering PDR allows for the initiation of the application development. In a particular enterprise there may be many such applications, but for clarity only one application development is shown. This research proposes that Data Provision be initiated at this time. To support Data Production, Data Provision requires tools and infrastructure elements to facilitate the formation of the data supply chain. The tools and infrastructure elements are likely to be software applications supported by people and procedures. These software applications will be developed as part of the software development lifecycle. Data Production may employ many data provision tools to form and manage the data supply chain. The arrangement of these data provision tools is informed (and may be directed) by the Enterprise Data Processing Model (EDPM). The purpose of Data Provision and Data Production is the delivery of the required application dataset to the application system. The use of an EDPM is proposed in DO 200A [18].

The Enterprise Data Architecture (EDA) is a strong influence upon the development of the applications, data provision and data production. The structure and nature of the EDA is also a significant influence upon the ability of the enterprise to adapt. In the circumstance where the EDA is based upon fixed structures, interfaces or legacy systems, the EDA restricts the ability of the enterprise to change and therefore the EDA further reinforces, and to some extent entrenches, the organisation. Conversely where the EDA contains adaptable elements, then the enterprise may employ this adaptability to evolve based upon business needs.

This adaptability also requires careful management, as the service (or product) offered by the enterprise is the basis for existing and future revenue. In such organisations the EDA will contain elements to facilitate the design of the service (or product), and therefore this *data design* will be enacted through *data production.* The Air Traffic Management (ATM) domain is an example of such a system. The *design house* is the national air transport authority (in the UK this airspace design (in terms of overall air route structures and categories of airspace) is in the hands of a joint CAA/MoD department). The design house creates the description of the airways and the procedures required for their operation (these are rules that describe the limitations of each airway as well as rules for the connection of different airways). In addition National Air Traffic Services (NATS) requires additional configuration information such as sectorisation to add to its ATM systems. In this ATM system it is desirable to create the design of the service provision that may be loaded into the system at regular intervals (every 28 days in the case of the ATM AIRAC cycle). It is therefore undesirable and expensive to validate the software and hardware components of the system every time the new dataset is loaded. Therefore the application *data design* is required to be independent of the hardware and software components of the system, within a pre-defined set of criteria. The data design is supported by data production.

8.5 The logistics required to implement regular data updates

Significant logistics may be required to ensure that the operational dataset is brought into effect on a certain date (and time of day), and that all those operational systems use the active dataset (or derived dataset), until such time as the dataset is superseded. The lifecycle should include requirements to support strong configuration management and design reviews at the lifecycle stage gates to ensure that data integrity is maintained. Stage gates may also be convenient landmarks in the development environment to enforce

Configuration Management baselines. An idealised circular data update lifecycle is depicted in Figure 11.

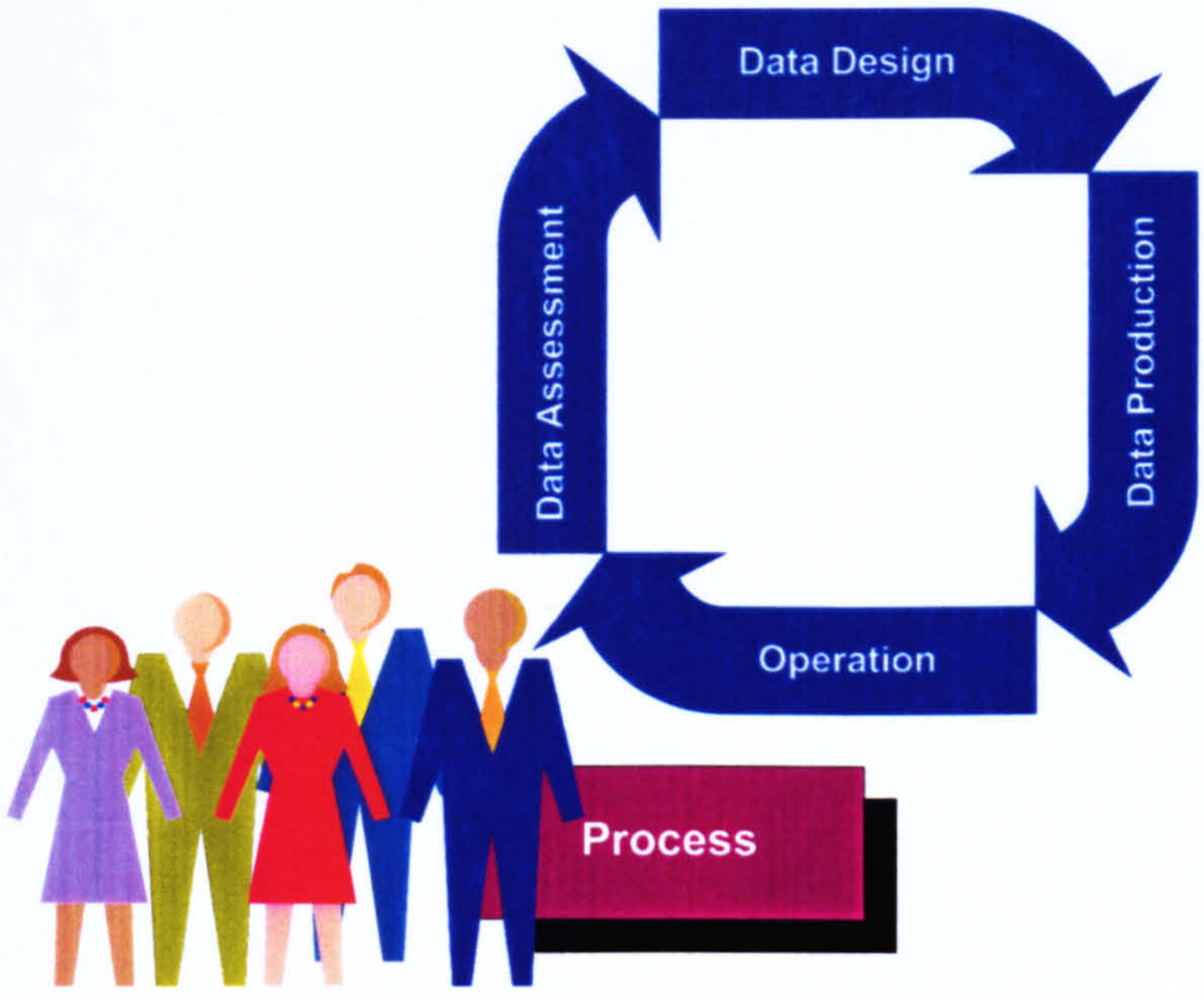


Figure 11: An idealised operational data provision lifecycle model

The data assessment may not be performed upon the current dataset until one or more data lifecycles are completed. To demonstrate a continuous series of circuits of the data lifecycle the model has been laid out to present a number of the data cycles so the relationship between cycle #N and its predecessors cycles may be shown. A series of data cycles are depicted in Figure 12.

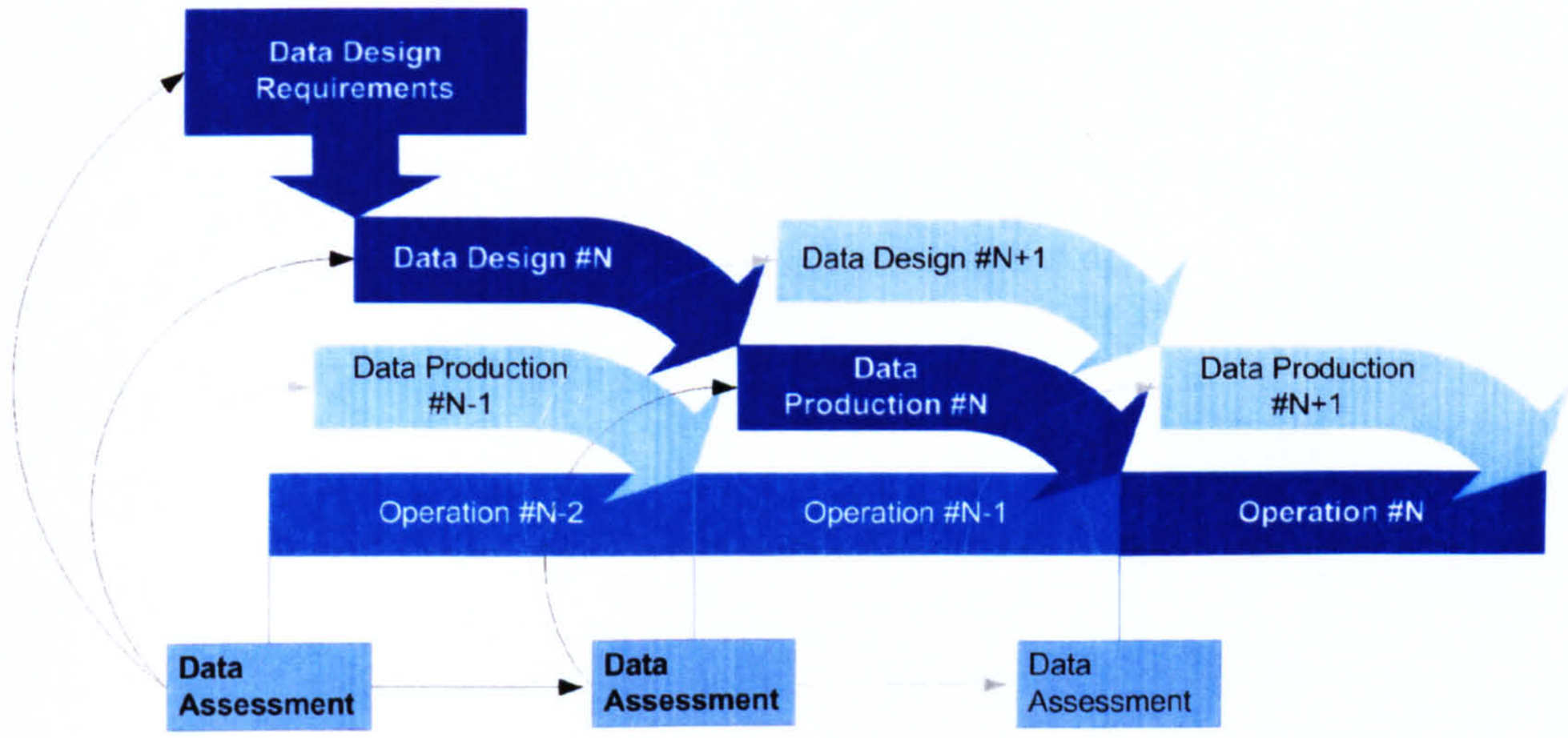


Figure 12: A more practical view of operational data provision

The data assessment may contribute corrective actions from one or more previous data cycles. These corrective actions are feedback into the design and

production phases of the cycle. This is a simplified view of the data cycle as prior to the data becoming operational the data should be subject to data assurance activities such as verification and validation. Where the data integrity requirements demand it, the data should also be used in a simulator with a range of operational conditions. These operational conditions should not only consider normal operation but also a variety of degraded (and emergency) modes.

8.6 The execution of application dataset updates during the working lifetime of the system

The development arena is likely to encompass the development of the system, the tools to support data provision and data production. It is also likely that development activity gives way to maintenance after successful completion of the Factory and Site Acceptance Testing (FAT / SAT) as the system enters operational service.

Once the system is deployed, data updates may be undertaken during the working lifetime of the system. Figure 13 depicts these data updates.

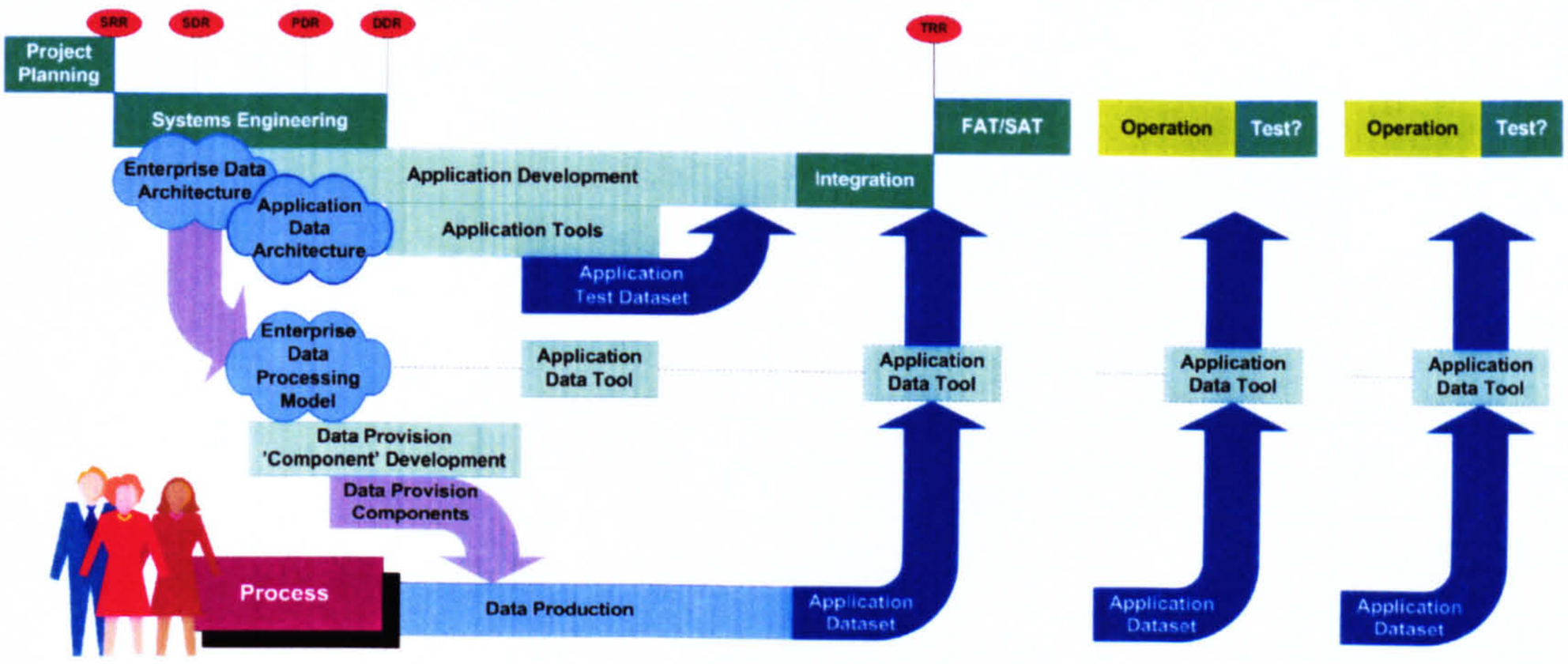


Figure 13: Providing data updates during the working lifetime

The process shown in Figure 13, delivers data to the operational system. This data is loaded into the application (or system) using the application data tool. These data updates are likely to take place outside the development arena and hence away from the rigour of the people and processes that created the system. This research observes that what is not clear is whether the application (or system) will be subject to test before being returned to operational service.

8.7 Integrating application development with data provision

An important consideration in the operational life of a data-driven system will be the ability to update the application dataset. At this update the application dataset is 'taken into use'.

The provision of data during the operational lifetime of the system will be undertaken using the application data lifecycle (shown in the bottom left hand quadrant of Figure 9). The contrast between the application development and provision of operational data is that these processes are likely to be executed by separate groups of people. This separation between these groups may lead, at best, to misunderstanding the data integrity requirements. An important part of this understanding is the sensitivity of the application to different types of data errors. These different groups are represented in Figure 14.

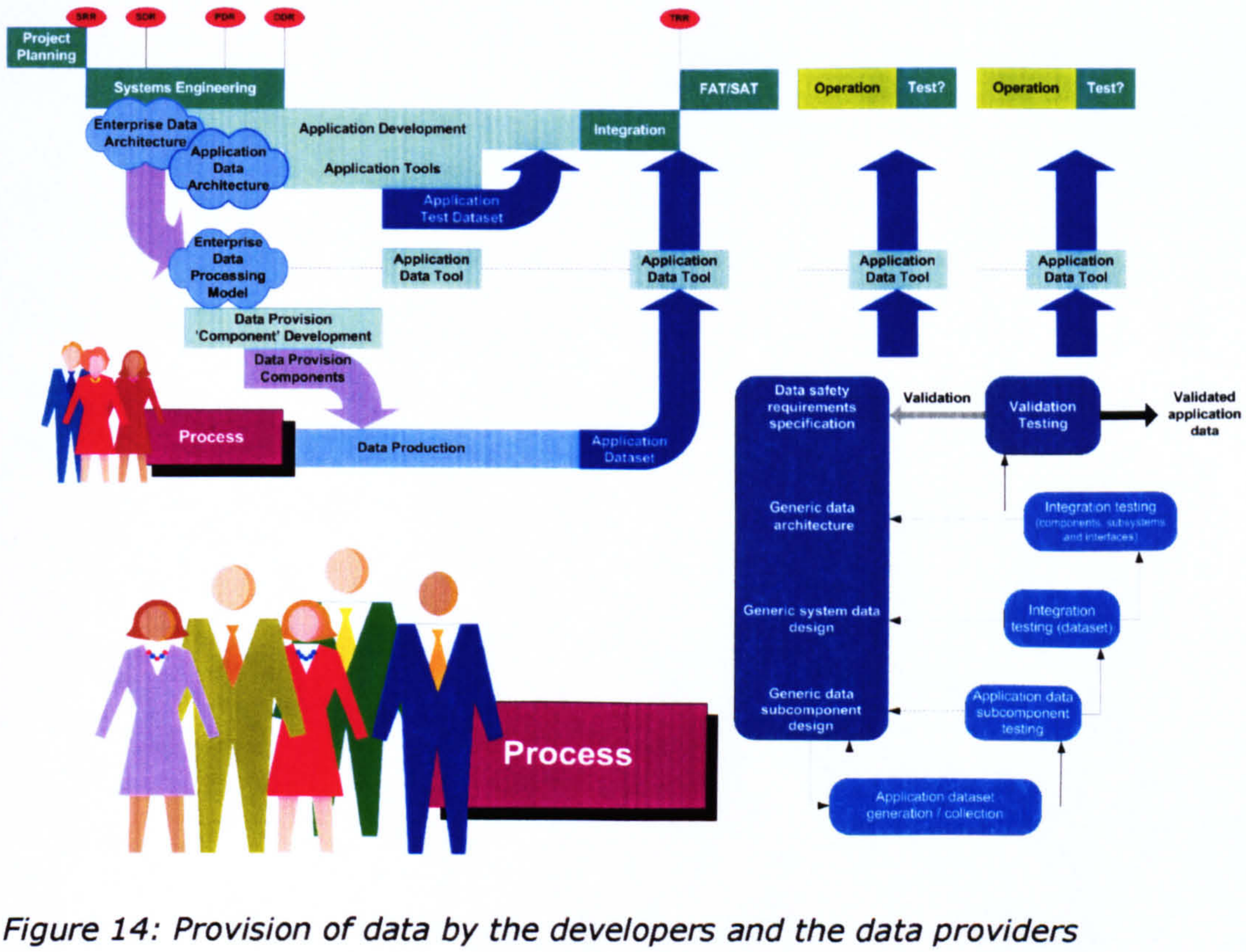


Figure 14: Provision of data by the developers and the data providers

One aspect of data provision is the use of a data supply chain. The requirements of the design and assessment of the data supply chain are addressed in the following section.

9 The design and assessment of a data supply chain

9.1 Data provision using a data supply chain

This research proposes the use of a data supply chain to satisfy the requirements of data provision. The data supply chain used in this research is based upon the aeronautical data chain described in DO 200A [18]. This has then been developed into a generic data supply chain. This research supports this generic data supply chain by proposing a design and assessment method.

9.2 The DO 200A data supply chain

The components of the data supply chain are described by DO 200A as phases. The DO 200A phases are receive, assemble, translate, select, format and distribute.

An example description of a data supply chain might be where data is received from the origin and assembled into a database. Data is then extracted from the database, translated into the application format and subsequently stored in the (application) database. Data is then selected from the (application) database into a tailored subset. This tailored subset is then formatted and stored. The formatted data is distributed to the application [18].

Each phase is executed to satisfy a set of specified criteria. Data is verified against these criteria and where the data fails to meet the criteria corrective action may be instigated and an error log recorded.

The author contributes the *consolidate* and *transform* phases to this description to complete the functionality required for a general-purpose data supply chain. A further contribution is the description of a method for the design of a data supply chain, including a graphical representation. When considering the design of a data supply chain it may become evident that combinations of phases are used several times. These phases may be collected together in formations to provide combinational re-use.

9.3 Generalising the DO 200A data supply chain

To generalise the DO 200A data supply chain the author considers that two further phases are required. These are the *consolidate* and *transform* phases.

The *consolidate* phase selects a data from a database based upon predetermined criteria, processing the data to produce a limited number of data elements based upon the selected data. An example of the proposed use of the consolidate phase would be to provide a data element representing a property of the selected data such as minima, maxima or average value.

The transform phase takes as its input a dataset, its purpose is to apply verification criteria in order to create evidence to justify the use of the dataset based upon the data integrity requirements of the consuming system(s).

9.4 Defining a graphical representation for a data supply chain

Appendix C provides a graphical representation as a means of presenting the data supply chain as a directed graph. For ease of representation standard flowchart symbols are re-used, as these are commonly available in a number of commercial diagramming packages.

9.5 A design and assessment method for a data supply chain

This research proposes that the design of a well-formed data supply chain should take into account not only the technology aspects of the integrity requirements, but should also consider the organisation which will execute procedures (including manual processes) to support these data supply chains. The execution of these manual processes may be a source of data errors. The opportunity for data errors may arise from a range of support services such as the administration of backup and recovery to human error during data entry. Therefore the capability of the organisation to supply data of the required integrity must include those components of the supply chain within its responsibility and all the processes used to support it.

A data supply chain should be 'fit for purpose', and contain only those elements required to supply data of the required integrity. Short supply chains, involving fewer components, are to be preferred over long (and possibly complex) supply chains. Much of the opportunity for optimisation is dependent upon the re-use of verification and validation criteria, the length of the data supply chain and its complexity. Short supply chains may not contain many re-used items, while long or complex supply chains, which receive data from many sources, may present significant opportunities for optimisation. The criteria for optimisation are based upon the number of instances of use and the integrity required from the supply chain.

The design process for the data supply chain should:

i) Identify the data origins.

ii) Identify the (organisational, legal, and political) boundaries within the supply chain.

iii) Identify the process and adaptation phases required.

iv) Apportion the integrity requirements along the data supply chain.

v) Identify evidence requirements.

vi) Specify corrective action processes.

vii) Assess the design of the data supply chain.

Additional detail of this design method is provided in Appendix C

9.6 Common formations in the data supply chain

9.6.1 The example of data passed across the system boundary

The confidence in the integrity of the data within the data supply chain is based upon the available evidence. However where data is passed from an external organisation such evidence may not always accompany the data. To illustrate this point the situation of data passed across the boundary, with supporting evidence, is contrasted with uncorroborated data passed across the boundary is developed in the following subsections. Uncorroborated data will require verification. In such cases verification may be difficult as, using the above description, no evidence is presented to support the data integrity claim. One option is the use of default values to separate (or even replace) low integrity data elements within the dataset with high integrity default elements during the verification process. However, the use of default values may significantly influence the data integrity that may be attained as the dataset may become sterile.

9.6.2 Data passed across the boundary with supporting evidence

In order to produce the desired distribution media the required tailored data set is created (select), this tailored dataset is then formatted (format) and transferred to the distribution media (distribute). The distribution media is passed across the boundary, received (receive) and stored. This fragment of a data supply chain is shown in Figure 15.

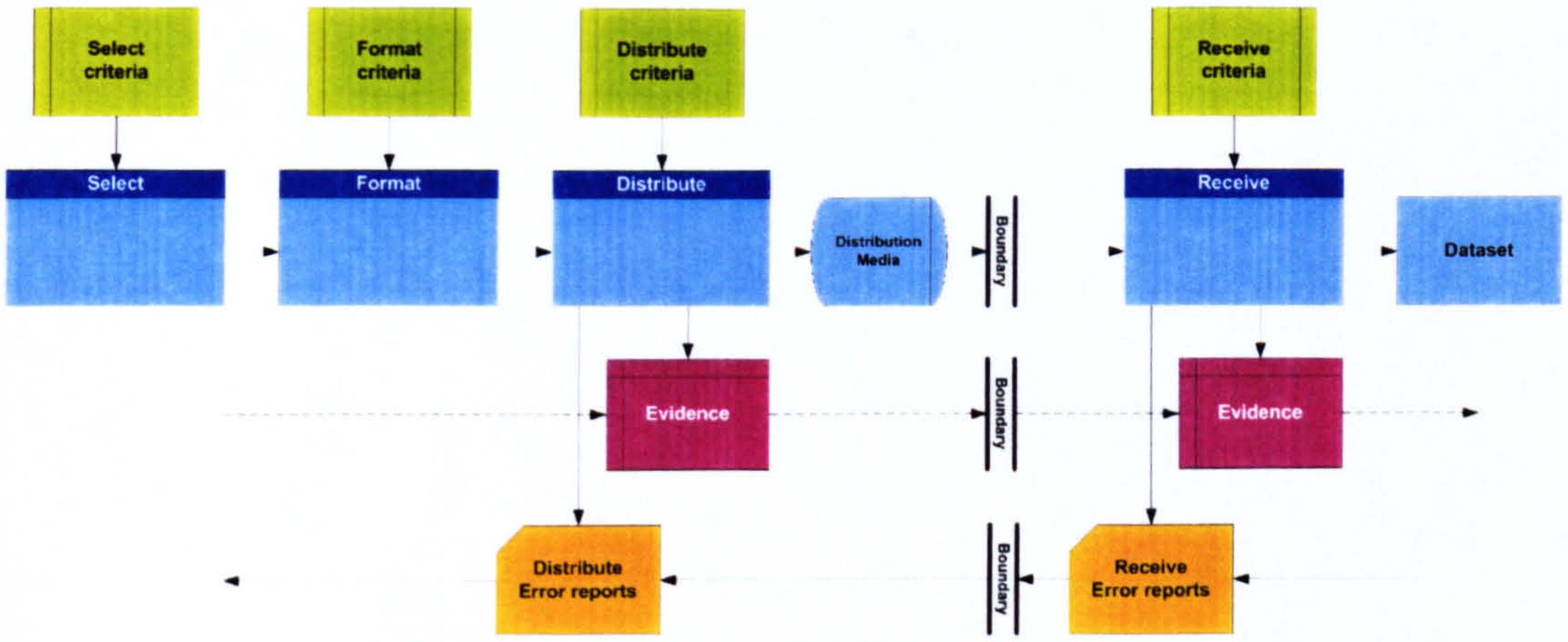


Figure 15: Data passed across a boundary with supporting evidence

At each phase the verification criteria are applied and any errors are logged into the corrective action system. The corrective action system may correct a localised failure such as a 'bad block' on the distribution media in which case the error is corrected by re-transmission. Where the corrective action is not localised, the corrective action is passed to an earlier stage for correction.

Evidence accompanies the data in its passage across the boundary either as references (to items such as to test certificates) or the actual evidence (such as analysis results).

9.6.3 Uncorroborated data passed across the boundary

When data is passed across a boundary without the evidence to support its claimed integrity, additional measures are required to establish the integrity of this data. The integrity of the data must be established before its use. On receipt the distribution media is verified for transmission errors (by the receive phase). The lack of evidence then gives rise to the requirement for a *transform* phase. This fragment of a data supply chain is shown in Figure 16.

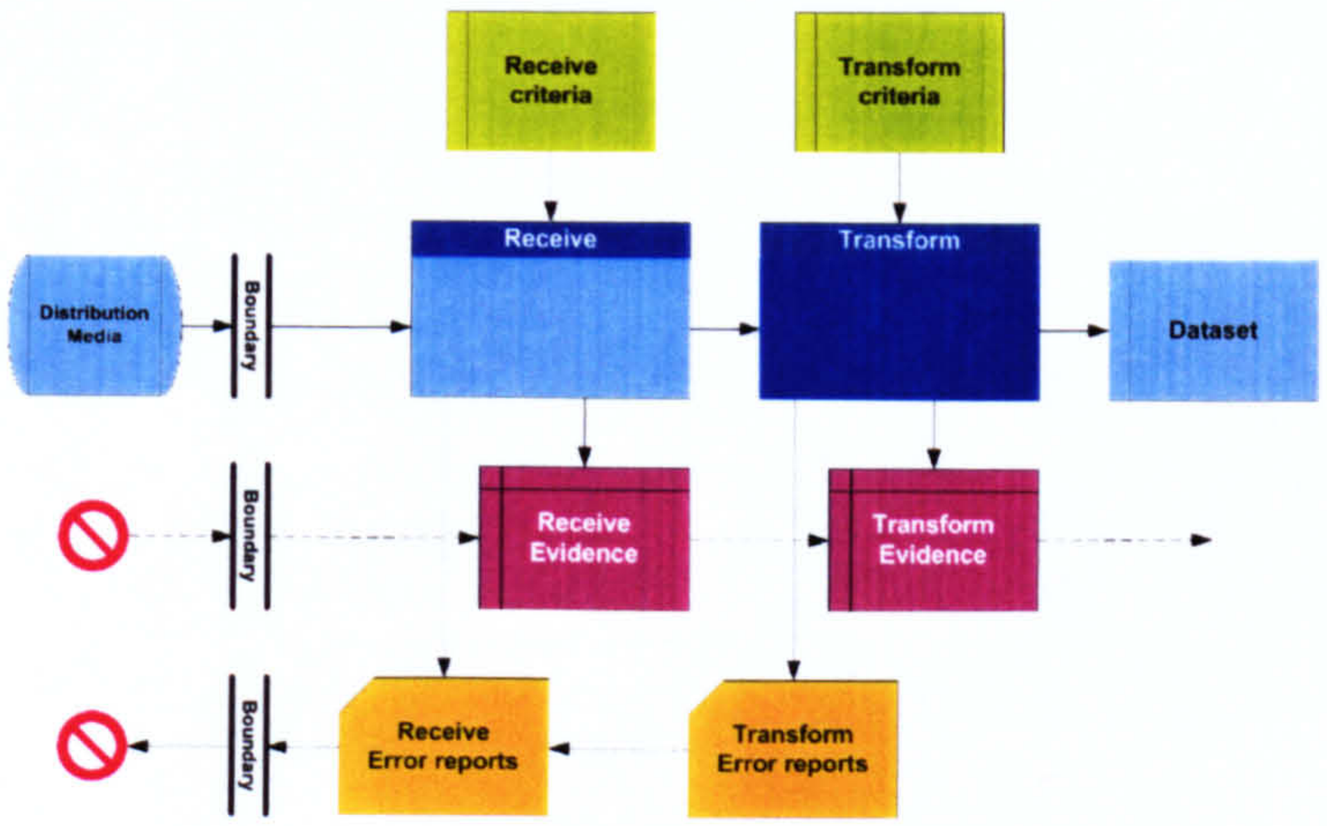


Figure 16: Data passed across a boundary without supporting evidence

The data is processed by the transform phase, based upon the verification criteria, to produce evidence to support the integrity claim. The only evidence available to support the claimed integrity is that provided by the receive and transform phases. In some cases corrective actions may not be passed across the boundary (to the data source), leaving open the question of how to correct this uncorroborated data.

9.6.4 Verification and the use of default values

The repeated application of default values and data elements to sanitise data may render the dataset sterile. Such a sterile dataset may contain little useful information representing the real world and hence may be considered useless. The basis of the use of default references and data elements therefore requires extensive justification and should be documented to a level commensurate with the risk of the introduction of error and its possible consequence. Any synthesis must assume the most restrictive form to be consistent with a conservative risk management policy. Any inclusion of default data elements in the transformed dataset should be justified by assessment and where necessary by direct comparison with the real world.

9.7 Abstraction in data provision

Sections 9.6.2 and 9.6.3 have described data passed across the boundary without any consideration of the abstract hierarchy that may be implied in the processing of data within the data supply chain. Data of higher abstraction may be produced through the selection and processing of data of lower abstraction. This abstract data is typically a single value or a reduced number of values representing a consolidation of the selected data. The ability to produce data of a higher abstraction gives rise to the requirement for a *consolidate* phase [32]. The data supply chain may then implement a data processing model [33] including several levels of abstraction, which may also consist of signal processing elements [32].

9.8 A data processing model for the data supply chain

An enterprise may employ a number of applications that are required to share data, based upon a common data supply chain. This requirement to share data may also require the design and definition of an Enterprise Data Processing Model. The Enterprise Data Processing Model provides visibility of the nature of data shared between these applications and also provides a means to identify and enforce the rules governing the exchange of data between systems of differing integrity.

Figure 17 is a fragment of Figure 10 (page 44) and illustrates the relationship between the Enterprise Data Architecture, the Enterprise Data Processing Model and the Application Data Architecture.

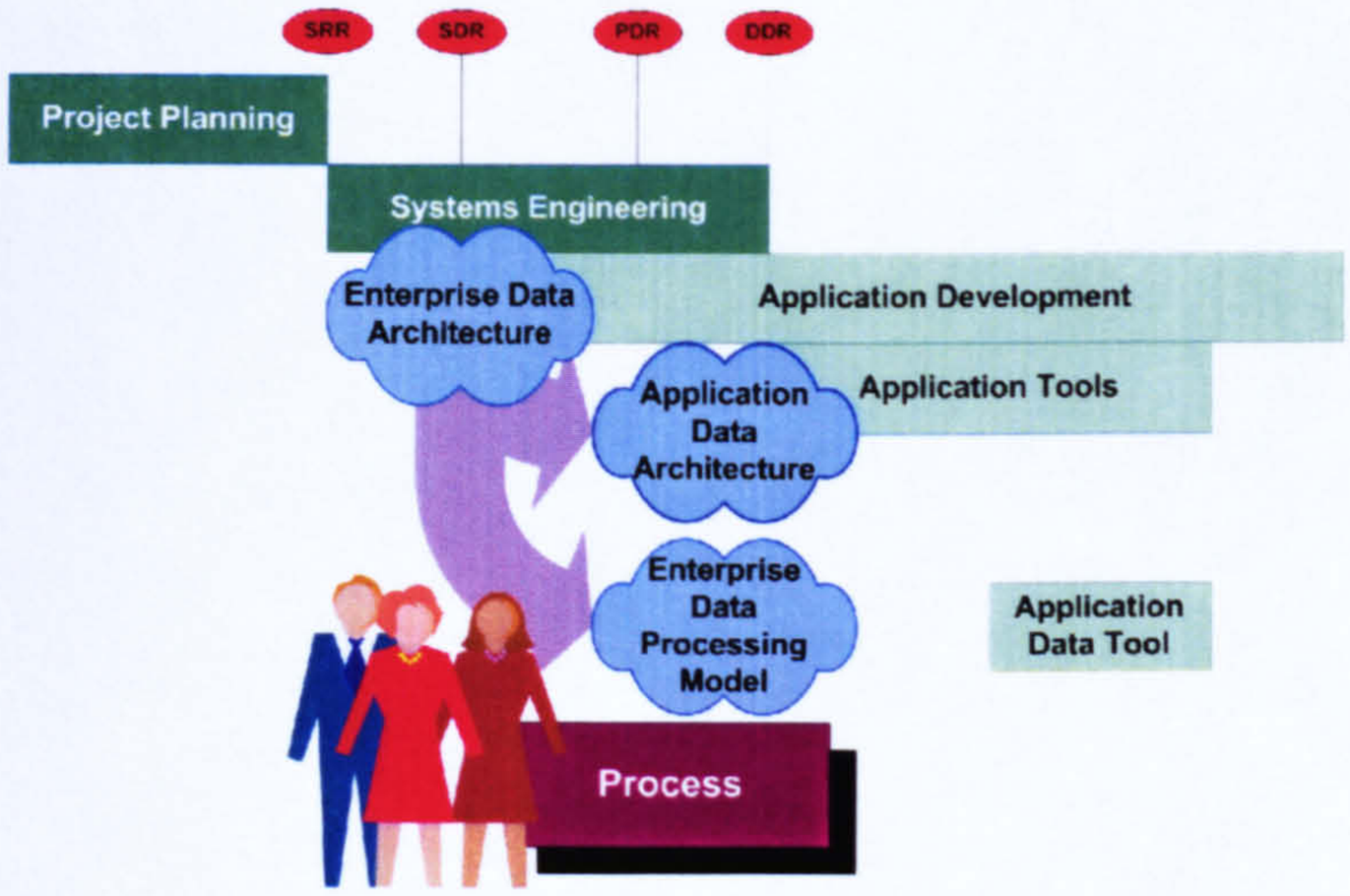


Figure 17: A fragment of the system development lifecycle.

In this research the Enterprise Data Architecture (EDA) is considered as the 'master' data definition and that both the Application Data Architecture and Enterprise Data Processing Model are derived from the EDA.

9.9 Concerns over the stability of a data supply chain

This research observes that the data supply chain may be considered to be a form of signal processing as information flows from the data source to the consuming systems. This abstraction of the data supply chain is shown in Figure 18, which shows how detected errors may be fed back for correction. In some cases the corrective action may be fed forward (positive feedback) or fed backwards (negative feedback). This correction may occur locally within a limited group of phases, within the organisation or along the data supply chain.

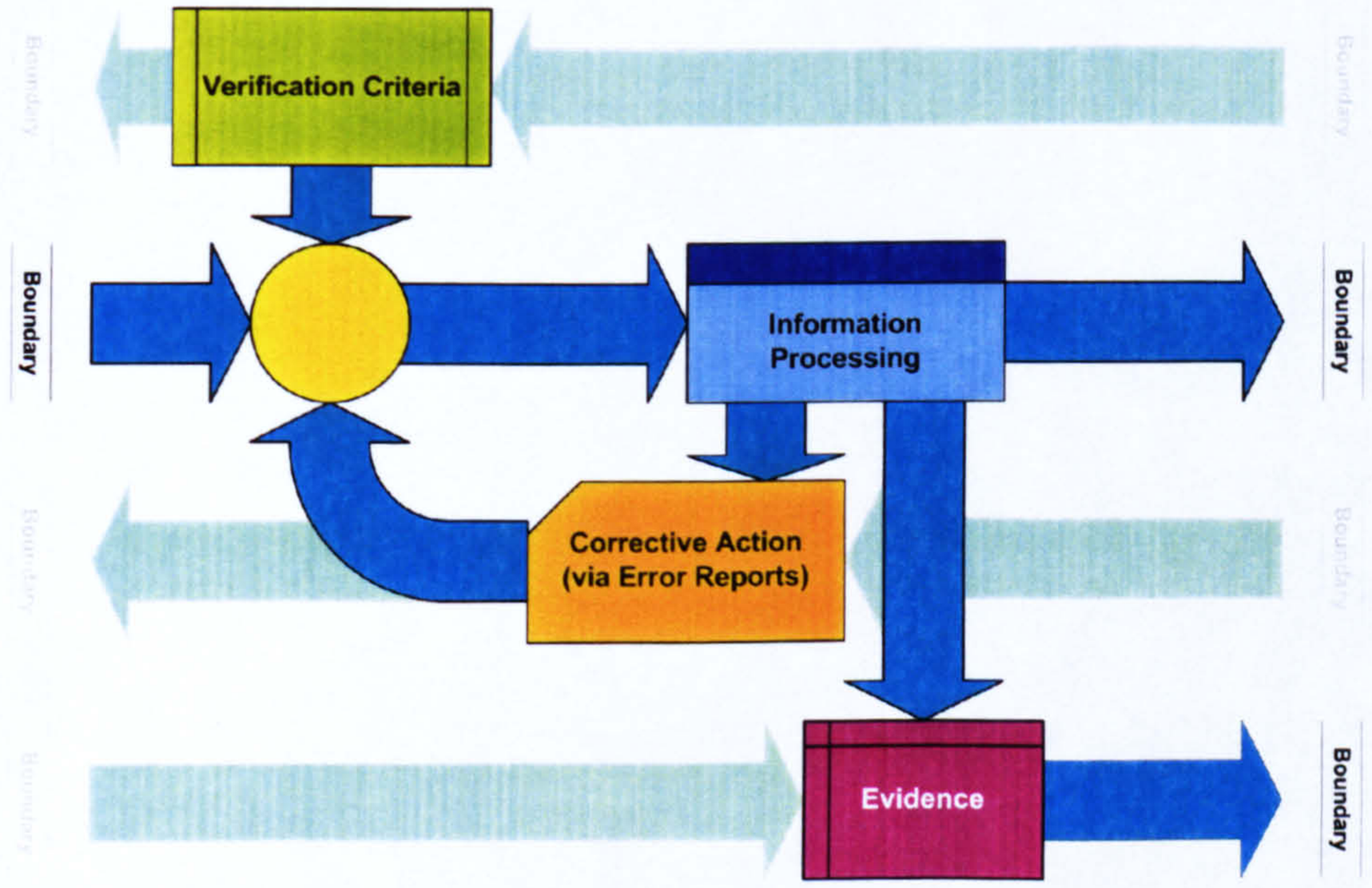


Figure 18: A fragment of the data supply chain to illustrate feedback.

Treating the data supply chain as a feedback control system facilitates design and analysis to determine its properties (see Atkinson [38], "*Feedback control theory for Engineers*"). It should be borne in mind that the time constants for such a feedback system might be in the order of days, weeks or even months, rather than the hours or seconds commonly associated with control systems which control process plant.

Two of these properties are:

i) Stability (based upon known arrangements of positive or negative feedback); and,

ii) Response (usually expressed as the damping factor, to describe the response to a step input).

The use of the feedback control model allows the recursive use of this model, as the control system itself may comprise one or more sub-systems.

9.10 Data ownership, liability for data errors and risk management

A data supply chain may extend across many organisational and political boundaries. An example of such a data supply chain is shown in Figure 19. This Data Supply Chain transports data across two organisations (shown as the pale yellow boxes). In each organisation data is received, processed, and formatted for delivery to subsequent elements of the data supply chain. Data may be owned by the each organisation, particularly if that organisation 'adds value' to the data through processes applied to the data.

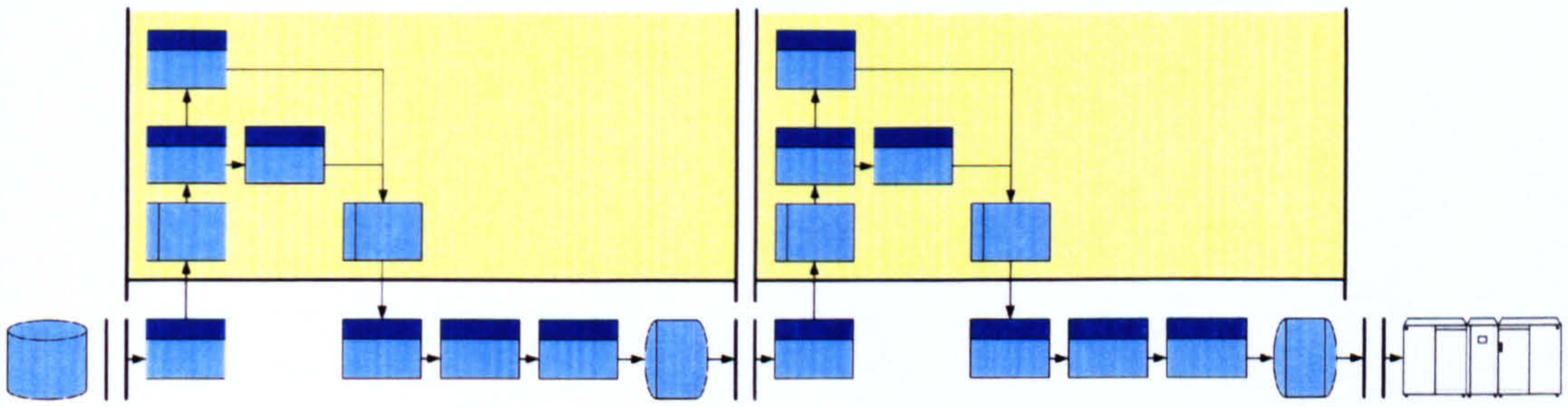


Figure 19: A data supply chain crossing organisational boundaries

Using the general description of a Data Supply Chain shown in Figure 19, this research observes that ownership is a complex issue as data may originate from a number of organisational and political bodies. In addition changes in ownership may result from data processing such as consolidation to produce a higher data abstraction. For example, operational considerations may allow ownership to change based upon pre-defined criteria such as handover into maintenance and hand back into operation upon completion of maintenance.

Data provision requires that organisational and political boundaries be documented, including responsibilities, ownership, intellectual property rights, legal constraints, liabilities and restrictions on use. The misuse of a dataset may attract liability to the data provider, which is outside the scope of supply. The data supply chain may include a data processing model in which data is processed including consolidation into higher abstractions. The ownership of these consolidated data items should also be established as this consolidated data also attracts the attributes of ownership, liability and risk and may subsequently be passed across organisational boundaries.

In addition, evidence (or references to evidence) should also accompany the data to support any integrity claim. Corrective actions may also be passed back along the data supply chain. For example, the ownership of corrective actions for errors found in data may be local or passed down the data supply chain to the data origin. Any corrective action placed upon the original source may still not yield data of suitable integrity. If the dataset is corrected locally then the connection between the input dataset and the delivered dataset may become untenable as the number of corrections rise. In these circumstances the corrected dataset may become a data origin it its own right and separate from the original dataset, attracting the attributes of ownership, liability and risk. Therefore the design options for corrective actions may require justification based upon the integrity requirements of the data and its intended use.

10 Discussion

10.1 The treatment of data used by safety systems

This research has undertaken a wide-ranging overview of the use (and abuse) of data used by safety systems. Examination of the treatment of data in existing literature [5] and a survey of industrial practice [20] provides a stark insight into the conventional view of data as simply an aspect of software.

Treating data as an aspect of software or simply failing to recognise the influence of data (and data errors) and their potential effect on the overall system often exposes those systems to risk. The risk associated with data (and data errors) is often untreated in the safety justification for such systems and may be regarded as an 'unattended' risk. In some cases the risk associated with a lack of data integrity has been a significant factor that has led to harm (people have been killed [13, 21 and 22] and many others remain at risk [23 to 25]).

The survey of industrial practice [20] has shown that there is little consistency in the methods used for data development. In addition the treatment of data has often been confined to data development, without recognising that data is often provided from outside the development arena [21]. The proposition of this research (see section 1.2) is to treat data as a separate and distinct component. This represents a novel paradigm or conceptual framework. One of the properties of this data paradigm is that it simplifies the treatment of data and also facilitates examination of the structure and complexity of the safety arguments for data and the systems that depend upon data. This data paradigm requires the support of strategies and techniques for the treatment of data. One of these strategies is the use of generic data and application data lifecycle models. These strategies and techniques are discussed later in this section.

10.2 A conventional view of data

A conventional view of data fosters a number of assumptions associated with the cost and availability of the data. An example of this approach is a large infrastructure project that in 1998 proposed to use generic elements and systems as a means to reduce the acquisition cost of the overall system. However, the decision to use (or re-use) these generic elements was based on the assumption that the required data was readily available at little cost and that data changes could be made without the need for re-validation of the data or the system as a whole. These assumptions have subsequently proved to be unfounded.

Many engineers seem reluctant to recognise the importance of data integrity. In many cases data is assumed to be correct, with any errors being seen as the responsibility of the data supplier [20]. In order to control data integrity it is necessary for an enterprise to manage all aspects of data integrity. This research proposes that this control also requires that the enterprise develops and manages one or more data architectures. Data production also becomes a significant aspect of the management and control of data integrity.

Perhaps one of the reasons why data is given so little prominence is that many engineers make the tacit assumption that data is simply an aspect of software and is therefore covered by the general guidance given [1, 5, 6, 19 and 20]. Certainly, many definitions of software include data (and documentation) within its remit [1, 6 and 19]. However, it is quite clear that in many cases the data component is not developed alongside more conventional software, and is not developed with the same degree of care [20, 21, 23 to 25, 30 and 31].

10.3 A data paradigm - the treatment of data as a separate systems component

Modern software development practices contain conceptual frameworks that are often taken for granted. Industry specific [27 and 34] and generic standards [19] are based upon a broad consensus that the software development conceptual framework comprises a development lifecycle model that is combined with techniques and measures recommended based upon the proposed system (and the technological solutions required to implement it) and the required system integrity. This consensus is also represented at the system level in best practice guidance [6 and 35].

The data paradigm, proposed in this research, draws on the software consensus. The conceptual framework for data requires that:

i) Data is adequately defined and its requirements are specified;

ii) A data lifecycle model is identified;

iii) Strategies for the implementation of data are identified; and

iv) Data techniques and measures are identified for use in each of the stages of the lifecycle model.

Data may also be exchanged between applications. This represents a departure from the parallel drawn with software. This places a requirement on the data paradigm to address the exchange of data between systems.

10.4 Defining data for use by safety systems

Good practice requires that the data component of a system be organised into collections of related data items and structures; this is usually known as a data dictionary. A well organised and well structured data dictionary facilitates navigation and allows examination of related items. A complex system may require an extensive data dictionary. This research has proposed a series of requirements that extend the types of information to be recorded alongside the definition of data in a data dictionary (see section 4.2). Recording a data classification with each data dictionary entry provides an indication of how that data item will be used (see section 4.3). The data classification takes account of static and dynamic data elements as well as that data which is used to describe the future use of the system (schedule data).

Where these data items or data structures are passed between systems or between layers of the layered model any errors in this data may be propagated across the overall system. The layered model provides a means to visualise the use made of this data and facilitates consideration of how the propagation of data errors will be detected and controlled. One aspect of this control is to propose a set of rules for the exchange of data between systems of differing integrity (see section 6.5 and [30]). These rules form the basis of design criteria so that the appropriate detection and controls may be embodied within the application (or system) design.

10.5 The determination of data integrity requirements

Over the years the various safety-related industries have gained considerable experience in assessing the safety of systems. The general characteristics of hardware and software are well understood and many designers are very experienced at partitioning between these two resources. Standards such as IEC 61508 [1] and best practice guidance such as ESM [6] give detailed guidance on techniques for the determination of integrity requirements. Using these techniques, it is possible to have reasonable confidence that all the hazards associated with the safety functions have been identified. However these standards and best practice guides do not address the determination of data integrity requirements.

This research has described a method for determining the data integrity requirements (see section 5). This method uses an adaptation of the commonly used Functional Failure Analysis technique [2] (which is itself a variation on Failure Modes and Effects Analysis). This research has also contributed a series

of functional failure guidewords for data to be used in conducting the Functional Failure Analysis (see section 5.4, [47] and [52]).

10.6 Identifying and managing the desirable properties of data

A system possessing low coupling and high cohesion will generally possess the desirable design properties of resilience and stability [37]. In addition, low coupling and high cohesion will generally minimise the effects that changes to one component may have on other components of the system. In the context of software this definition reflects the functionality implemented by software. Additional care should be exercised when considering data as the same data element may be used in several contexts and possibly by several applications or systems. In each context this data element will be ascribed subtly different meanings. Therefore changes to data may give rise to unexpected and possibly undesirable behaviour based upon the context and use made of the data.

The properties of low coupling and high cohesion have been considered self evident for both good software and good systems design, appearing in many texts. However anecdotal evidence suggests that the same general consensus cannot be said to exist for data used by data-intensive systems [20]. Data, in common with hardware and software, is a system component and broad parallels can be drawn as to the desirable properties for good design of data.

The role and influence of data (and data error) may be reviewed in the context of the layered systems model [29] to determine the influence of the application and the wider system (including the data it produces and consumes) [36]. A system visualised using the layered model might reveal that one or more data attributes are used in more than one layer of the model, and that any change to these data attributes may influence one or more real-time safety functions. In such circumstances this research would propose that additional design elements are required to implement mechanisms to manage change of the data attribute without adversely affecting the safety of the individual or overall system.

10.7 The selection of a data lifecycle model

A reasonable starting point in the consideration of such a lifecycle model is the software development lifecycle given within IEC 61508 [19]. IEC 61508 uses the example of the 'V' lifecycle model, with which most engineers will be familiar [41].

The use of other lifecycle models is a matter of choice of the Engineer, the organisation or where data is to be supplied through the use of a data supply chain by agreement with those involved in data provision. Perhaps one of the

most significant aspects of this research is the recognition that data design and development may be separate from the data provision [44].

As described in section 8.3, the design and development of data may require extensive co-ordination of activities, processes and personnel far beyond that which can be shown in a simple 'V' lifecycle model. This research proposes that it is not sufficient to assume that the development of data forms an integral part of the overall system development process. The separation of data development and data provision requires that the data integrity requirements must be communicated from those responsible for the development of a safety system, to those responsible for the provision of the associated data.

10.8 Lifecycle models for generic and application data

As well as the generic data development lifecycle model, this research has developed an application data lifecycle model (see section 8.3.2). This application data lifecycle model recognises that the left hand side of the 'V' model is likely to be executed in the development arena and that the application data will use the data design elements as the basis for data provision.

In common with the hardware and software components of the system, the techniques and measures employed within the generic data development lifecycle model will contribute to the demonstration that systematic development processes have been applied to the design of the data component. These techniques and measures will have been chosen to match the rigour required by the data integrity requirements.

10.9 The design and structure of the data component

10.9.1 The interrelated nature of the application and enterprise data architectures

The design, development, implementation and maintenance of a large scale data-driven systems is a complex undertakings. These complexities come about in part through the shared use of common data definitions and datasets. In such a system an individual application will be required to co-operate with, and be an integral part of the overall system or enterprise.

A data model for an application, and the resulting dataset may contain hundreds of thousands of data items, often with complex relationships. The overall system (or enterprise) may contain many such applications and ensuring the integrity of all this data is then far from trivial [4].

DO 200A [18] describes an Enterprise Data Processing Model (EDPM). This research draws from the description of the EDPM a requirement for the management of data models at the application and enterprise levels. This

research proposes that the Enterprise Data Architecture should describe the data model for the enterprise as a whole and that an Application Data Architecture should correspondingly describe the data model for each application.
The description of the data lifecycle model (Figure 10 and Figure 17) depicts the relationship between the data structures in the Enterprise Data Architecture and a number of Application Data Architectures as well as the Enterprise Data Processing Model.

10.9.2 The design and control of the Enterprise Data Architecture

The design of the Enterprise Data Architecture should address the requirements of the application data as well as the requirements of data update during normal system operation. These requirements may involve the ability to change the Enterprise Data Architecture to reflect changes in the revenue patterns of the enterprise. The Enterprise Data Architecture should therefore contain the inherent design properties of modularity and independence. Therefore all systems, which use all, or part of the Enterprise Data Architecture or a dataset derived from it, must register their interest in the data dictionary that describes the Enterprise Data Architecture. The register of interest should identify each data element, structure, data reference or attribute together with its respective integrity requirements.

This should be undertaken as part of data definition and this research has described the requirements of these dependencies in the definition of a data dictionary (see section 4.1).

10.9.3 The design and control of the Application Data Architecture

The architectural design of an application should clearly identify the data elements within the system, distinct from the hardware and software components [5]. Therefore, data should also be described by an appropriate data model that is self-sufficient, clear, analysable and unambiguous [5]. In addition data that is derived from external sources, or data that is developed separately from the safety-related system itself, should be subject to the same requirements of verification and documentation, as data produced as an integral part of the project [5]. The design and data should be clearly described in the lifecycle of the system and in terms of data as a separate systems component.

10.9.4 The development of the Enterprise Data Processing Model

DO 200A [18] describes an Enterprise Data Processing Model and an implementation of an aeronautical data supply chain using *phases* that may be combined to provide the required application dataset. The DO 200A aeronautical

data chain has been extended into a generalised data supply chain in the submission *Data Provision* [32].

Each phase of the data supply chain may be implemented as software and as such will be subject to the software development lifecycle. An Enterprise Data Processing Model may be extensive as its definition may contain the requirements and data definitions of many application, systems and organisations.

10.10 Consideration of the safety arguments for generic and application data

10.10.1 Safety arguments for data

The separation of data development and application data provision leads directly to the consideration of the nature and structure of the safety arguments required to support the data component. As the development and provision of data are likely to be undertaken in two separate arenas, this research proposes that the safety argument will comprise of at least two components, one for data development and one for data provision.

10.10.2 The management and control of data updates

A desirable feature of an existing systems environment would be an existing Enterprise Data Architecture. Where an Enterprise Data Architecture does not exist, the application developer should consider the structure and nature of any safety justification, as without an Enterprise Data Architecture the construction of the safety case may be difficult. For example, a source of error may occur when more than one system or application updates shared data. Good design practice requires that only one system be permitted to change the data; therefore requests to change data are passed to the responsible system, which validates the request and updates the data as required. Hence, the authority and responsibility for changes to data are set out in the Enterprise Data Architecture. An alternative would be to allow more than one system to change data. In the alternative arrangement, these data updates may induce data errors. Under this alternative arrangement establishing the liability for any data errors may also prove problematic.

Therefore the management and control of data updates and changes to data models may be a significant influence on the nature and complexity of the data safety arguments.

10.10.3 Safety arguments for data development

The design of a data-driven system may be constrained by issues associated with compatibility with existing systems. There may be no recorded integrity requirements for existing systems. Therefore an existing systems environment may require extensive effort to establish the integrity requirements of the existing systems as a precursor to establishing the integrity requirements for the new systems. Once the integrity requirements for the existing systems are established assessment of the existing datasets might well identify significant shortfalls in the data integrity of the existing data. Indeed many of the pre-existing systems may benefit from improvements in data quality (as a pre-requisite for data integrity).

The safety arguments for data development are likely to parallel those for software. The data development safety argument will show that a valid set of requirements has been documented and that the data component has demonstrated compliance with those requirements through testing. Testing may be demonstrated at unit (dataset) as well as integration and system levels. The data development safety argument should also demonstrate that an appropriate set of processes have been executed by suitably trained, experienced and competent staff [6 and 27].

10.10.4 Safety arguments for data provision

In contrast with data development, the safety argument for data provision may be extensive. This data provision safety argument must address data origination, the data supply chain, the data validation policy as well as the demonstration that data may be updated during the normal system operation. This research notes that the data provision safety argument may also be required to address the organisational and political influences on data integrity. In addition, the data provision safety argument may be influenced by the design and structure of the data component.

10.10.5 A safety argument for data update during normal operation

An additional safety argument may also be required to address the process of performing data updates during normal system operation.

10.10.6 The influence of data validation on the structure and nature of the safety argument

The strategy for data validation may be a significant influence on the structure and complexity of the safety argument for data (see section 7). DO 200A [18] proposes that the responsibility to ensure that the data integrity requirements

have been met rests with the user of the data. An adaptation of the DO 200A 'responsibility' policy for data validation encourages those responsible for the system to identify and define a boundary. In defining the boundary, this allows evaluation of the complexity of any proposed safety argument for the data component.

10.11 Data provision using a data supply chain

A data supply chain may be used to transport the data from its origin to the consuming system. Such data supply chains may be explicit in the case of systems that conform to DO 200A [18]. These data supply chains may also be unrecognised and implicit where data entry is separate from the consuming system [31].

Data may be provided either on a batch basis or as a continuous process (or as a mixture of the two). Any errors detected in this data should then be passed through the corrective action process, where errors may be corrected In the received dataset or in some future dataset. Where these data errors affect the operation of the system, additional operating procedures may be required upon discovery of any data errors. These additional procedures may represent an additional workload for the system operators.

This research observed that if data from an existing system is to be re-used in a safety-related system, additional measures are required to establish its integrity. This may require extensive off-line processing to produce sufficient evidence to support a claim that the required integrity has been attained. After the system enters service these off-line data processes (and metrics) may be required in order to ensure that the required system integrity is maintained. Therefore the process for data provision should be designed to support the capability of the organisation to supply, meet, and maintain the data integrity requirements.

10.12 Rules for the exchange of data between systems of differing integrity

This research asserts that it is undesirable for a low integrity system to pass data to a high integrity system without that data being subject to verification [30]. Clearly this low integrity data, by definition, is more likely to exhibit a data error rate greater than that required by the high integrity system. Once data is treated as a separate component these requirements become self-evident based upon the data definition and data integrity requirements of each system.

10.13 The design and construction of the data supply chain

The design and construction of the information processing within the data supply chain may give rise to additional complexity. The use of the *consolidate* phase allows the production of additional datasets which contain abstract data derived from existing data within the data supply chain. The design of the data supply chain will be influenced by changes to functional interfaces, which may induce dependant changes in both the data requirements specification and the integrity requirements.

This research notes that although expressed as a linear construction, the data supply chain may include many loops reflecting organisational boundaries. Such organisational boundaries may produce changes of ownership of the data and hence responsibility and liability for data errors requires careful management. Error correction will be concerned with feeding back the error reports to earlier components of the supply chain. The recognition of corrective action as a feedback mechanism facilitates the analysis of stability and response of the supply chain to change in the input data. Corrective actions may also produce additional corrections. Such further changes may exacerbate the original perturbation leading to multiple changes and possible instability.

Many organisations are likely to participate in the provision of data. In order to support a data supply chain an organisation will execute processes and procedures and therefore any data supply chain design should be supported by a suitable human factors assessment. Additional data errors may be introduced through process failures due to human errors such as misinterpretations of instructions or identification labelling.

10.14 Use of software tools and applications to support the data component

This research notes that one or more configuration tools may be used to configure a system. These configuration tools should be created using a software development process with sufficient rigour to reduce the probability of configuration tool induced data errors to a minimum.

These configuration tools may translate data into a form required by the application. These software tools may also be used to tailor the application to the operational requirements [39 and 40]. However, these are not the only configuration data requirements. The increasing use of generic components leads inevitably to greater use of configuration data.

Configuration data may also describe the physical arrangement of the computational infrastructure (that is the computers required to support the

application); the logical arrangement of the services provided by operating systems and other software infrastructure as well as the description of the application environment.

In the same way as for the definition of data, the use of the data classification and layered model are required to expose and make explicit the nature of data used by these safety systems. In addition these classifications and models allow standard safety engineering techniques to be adapted to data as a means to demonstrate the development of data safety requirements. These data safety requirements lead directly to the statement of data integrity requirements.

However, note that Data design is only one element of data integrity as data provision is also required.

11 Innovation, review and dissemination

11.1 The treatment of data prior to this research

Welborne and Bester [17] provide perhaps the most succinct summary of the state of literature at the commencement of this research. In 1996 they commented that:

> *"It appeared, after considerable searching, that almost no theoretical work or research study had been made of application data. Work had concentrated on methods of dynamic data correction in messages, database management, aspects such as software languages, and methods for definition of complete, correct and non-ambiguous software, without reference to the problem that incorrect application data can create. Incorrect application data can cause correct software, in an approved language, to have major failures in its performance"* [17].

Their comments are borne out in this research by the literature search [5] and survey of industrial practice [20]. Perhaps part of this failure to address data comes about because data is treated simply as an aspect of software.

11.2 Demonstration of innovation

A significant innovation of this research has been to propose the data paradigm, to consider data as a separate systems component. However it is not the only innovation. Perhaps the most concerning aspect of the conventional treatment of data is the recognition that the safety arguments used to reason about the use and operation of such systems are unlikely to address the specific requirements of data. This research identifies a number of possible safety arguments that concern data development, data provision and data update during normal system operation. This research also notes that the data validation policy may be a significant influence on the nature and complexity of these data safety arguments.

Once data is treated as a separate system component this research asserts that it follows that not only should this data component attract an apportionment of the system integrity requirements but also that the design, development and provision of data should be treated with appropriate rigour. The survey of industrial practice [20] has demonstrated that the treatment of data is largely confined to data development without recognising that data is often provided outside the development arena [21]. DO 200A [18] provides some guidance in this area but tends to focus upon data preparation.

This research addresses the requirements of data definition by proposing an extension to a data dictionary. In addition this research has examined the classification of data proposed by Welbourne and Bester [17]. This classification has been adapted and extended to address data that describes the future use of the system (Schedule data). One of the aspects of defining data is the identification of data integrity requirements. This research has proposed a method for the determination of data integrity requirements based upon Functional Failure Analysis (FFA). To support the use of FFA this research has also proposed a series of guidewords based upon the proposed classification of data.

An innovation of this research is to develop a generic data lifecycle model, and also an application data lifecycle model that recognises that data design and development may be separate from the provision of the application data.

The data paradigm presents by definition a conceptual framework for data. Using a parallel with software development has suggested a number of strategies and techniques for use with data. However as data may also be exchanged between applications and hence the same data may have different influences dependent upon its use. This aspect of data represents a divergence from the parallel drawn with software. This places additional requirement on the data paradigm to address the exchange of data between systems.

To facilitate the visualisation of the use (and reuse) of data this research has developed the layered system model [29 and 36]. The development of the layered systems model provides the safety assessor or approval body with a means of examining the scope of a safety-related system in an overall systems context.

An additional innovation within this research is the development of an enhanced data supply chain, from the limited, industry specific form proposed within DO 200A [18]. This research supports the generic data supply chain through the definition of a design description, notation and an assessment method. To aid presentation this research also proposes a graphical representation of the data supply chain.

The use of a graphical description of the data supply chain makes clear that data may be transported across one or more organisations before its delivery to the consuming systems. This research notes that liability for data errors may lie within the data supply chain, particularly if these organisations 'add value' through the provision of data processing services. This exposes concerns over ownership, responsibility and inevitably costs associated with data provision.

11.3 'Raising the profile' of data

One of the aims of this research (see section 1.3.1) was to 'raise the profile' of data. The author has presented at the following professional forums:

i) BCS (Northwest branch): "Data Integrity: The use of data by safety-related control systems", Manchester (January 2003);

ii) Safety Critical Systems Club (Joint Event): "Safety in the presence of data", London, (April 2003); and

iii) NCAF: "Data Integrity: The use (and re-use) of data by safety-related systems", Exeter, (May 2003).

The research has also produced fifteen papers (including one journal paper), all but the journal paper have also been presented at conferences. The paper "*Data Management in Data-Driven Safety-Related Systems*" [41] won the "Best Paper" award of the 20th International Safety System Conference 2002, Denver. Colorado USA. As a direct result of the best paper award the invited paper "*Data: An often-ignored component of safety-related systems*" was presented to the MoD *Equipment Safety Assurance Symposium* ESAS 2002 Bristol [42].

A growing recognition of the role of data lead to the one-day event hosted by the Safety Critical Systems Club "Safety in the presence of data" and Neil Storey and the author presented the first and last sessions respectively (see ii) above).

In addition the System Safety Society also created a separate conference session on data at the 21st International Safety System Conference 2003, Ottawa. Canada. The author contributed two papers to the conference [43 and 44].

11.4 Dissemination and peer review

Dissemination has not only been achieved through the fifteen published papers but also through the structured interviews. These structured interviews were conducted across a range of industries. It became abundantly clear that many engineers had not considered data and as a consequence that data integrity was being ignored across these industries.

Perhaps the most satisfying indication of the author's success in raising the profile of data through research, is the testimonial received from the IEC 61508 maintenance committee (MT12) [45] acknowledging the contribution of the published papers on the deliberations of the MT12 committee. The author notes that MT12 does not have a remit to create an additional section akin to part 3 (software) of IEC 61508 to address the requirements of the data component.

12 Conclusions

12.1 The stated aims of this research

The stated aims of this research project include the provision of guidance for professionals who assess and analyse safety systems, and to 'raise the profile' of data as a significant element within such systems. The material provided within this Executive Summary would suggest that these aims have been successfully achieved.

12.2 The treatment of data as a separate systems component

This research has shown that a conventional view of data would consider data as simply an aspect of software. In this conventional view of data the system developer may not recognise the influence of data (and data errors) on the integrity of the overall system. This research has shown that data is indeed poorly addressed in standards, literature and industrial practice. Without directly addressing data, in particular data integrity requirements, it is difficult to see how those responsible for the system safety case can argue that data-driven systems are tolerably safe. The conventional view of data becomes self reinforcing as without data integrity requirements data development and data provision may not attract the degree of rigour that would be required of other system components of a similar integrity.

In order to break the self reinforcing argument this research proposes a data paradigm, that data be treated as a separate system component. This treatment of data requires that data (and data errors) be considered as part of the system safety argument. This research provides much needed guidance on the safety aspects of the use of, and reliance upon, data by safety systems for their safe operation.

The author hopes that this work will be used as the basis of other work that will subsequently address issues identified within this document, and that this will be the foundation upon which to build a consensus as to the treatment of data used and relied upon by safety systems.

Irrespective of whether data is to be treated as a separate component, the importance of the early identification of the consequence of data errors on the overall system integrity cannot be overstated.

12.3 Guidance for the management of the data component of safety systems

This research proposes guidance that:

i) Describes and classifies data used by safety systems;

This research has proposed the requirements of a data dictionary to record the definition of data used by safety systems. This research has also proposed a taxonomy of data used by safety systems.

These data definitions of this data should not be considered to be static, as they will change and evolve based upon the changing requirements of the systems that produce data and of the systems which use the data. Therefore data definition requires management and control. These changes will require version control of the data definitions similar to that used for software.

ii) Describes a method to derive data integrity requirements;

In this research Functional Failure Analysis (FFA) has been used as one method to establish Data Integrity requirements. This research also proposes a series of guidewords to be used with FFA for each class of data within the data taxonomy.

iii) Describes a layered model to express the use (and re-use) of data within a systems context;

Large-scale systems are challenging and complex undertakings. The implementation and project management of large-scale systems becomes more difficult through increases in the coupling and complexity of the system itself and through its need to use and interface with many different data models.

This research has proposed a layered model to express the potential influence of an application or system on other elements within the layered model. This model is then adapted to express data used within the model and derive the desirable properties of the data component as low coupling and high cohesion.

iv) Describes the requirements of generic data and application data lifecycle models;

A finding of this research is that it is common for the data component to be provided from outside the development environment that produced the hardware and software components of the system. Away from the rigour of the development process, and coupled with the flexibility offered

by malleable data descriptions, the potential for error is great. This potential is not limited to data used to configure the application or system but also applies to data passed (in both directions) across the system interfaces.

Perhaps the major proposals of this research are those of the generic data and application data lifecycle models. The design and development of data may require extensive co-ordination of activities, processes and personnel far beyond that which can be shown in a simple 'V' lifecycle model.

v) Describes the process of data provision

This research has provided a description of the process of data provision through the development of a generic data supply chain. The description includes a description, notation and an assessment method for a data supply chain.

vi) Describes the nature of safety arguments for data driven safety systems

This research concludes that any system safety argument must address data development, data provision and data update during normal system operation. The data development safety argument should comprise a product argument addressing the validity of the data requirements and a demonstration that the data requirements have been met and a process based argument. The data development process safety argument will demonstrate that the data development has been undertaken using adequate process executed by trained, experience and competent personnel.

The data provision safety argument must address data origination and the data supply chain and once again address both product and process components of the argument. An additional process based safety argument may be required to demonstrate that data can be safely updated during normal system operation.

In addition this research observes that the data validation policy may have a significant influence on the nature and complexity of the safety arguments.

vii) Describes the relationship between different data models within the enterprise

This research concludes that where an Enterprise is reliant upon the quality of the data (and the data integrity) careful consideration is required not only of the construction and maintenance of the Enterprise Data Processing Model, but also the capabilities of the enterprise and those organisations that supply the enterprise with data. This research has demonstrated that data and data errors may adversely affect the behaviour of the system and that they have a potential effect on the overall system integrity, and hence influence the Enterprise. For this reason, the Enterprise Data Processing Model requires a foundation of building blocks through which the Enterprise Data Processing Model may be implemented.

The Enterprise Data Processing Model should also recognise that transportation of data through the data supply chain will give rise to changes in ownership. This research observes that such changes are also likely to involve changes in the liability for the data errors that these datasets may contain. The Enterprise Data Processing Model may be distributed across many systems, and hence there may also be organisational, political and national issues associated with the ownership of the data.

Where a number of systems share a common data model, as is the case in the use of the Enterprise Data Architecture and Application Data Architecture, each of these data architectures should be the subject of joint review by the System Design Authority and the Application Design Authority in order to ensure the integrity of each application and the data required (and maintained) by the enterprise. Shared data definitions may also be an indication that the required data is derived from shared datasets. These shared datasets may require data processing in the form of a data supply chain.

12.4 The dissemination of the research through published papers and presentation to professional forums

This research has contributed 15 papers to the literature on the subject. The author has also presented at professional forums. Recognition of this dissemination has come through an award of best paper, an invited paper and a one day event focused upon data hosted by the Safety Critical Systems Club.

In addition the author has received a testimonial from the maintenance committee of IEC61508 acknowledging the contribution of the published papers from this research on their deliberations.

13 Recommendations

13.1 A definition of data integrity

This research has used the term data integrity, albeit derived from a number of sources, to describe a requirement for the correctness or goodness of data used by safety systems. The author recognises that a definition of data integrity will be based upon consensus within the wider safety community. It is perhaps fitting that the first recommendation of this research be a definition of data integrity to begin the process of building such a consensus.

In many cases data integrity will be a profile of properties of a dataset that contains many data items, elements, structures and data references. Therefore data integrity should be focused on the properties of those data items that are used by an identified safety function. Data integrity requirements are a testament to the role played by data errors in the behaviour of the system and their potential affect on the overall system integrity.

R1 The first recommendation of this research is a definition of data integrity as *"a measure of data quality"* (see Definitions, page 12).

13.2 Data as a separate systems component

This research has identified that the conventional view of data has lead to many aspects of data integrity being ignored. Where such systems are used in safety-related applications, the safety of the resulting system will often be dependent on the correctness of this data.

R2 This research recommends that data be treated as a separate system component.

This recommendation (R2) would help to ensure that data is treated appropriately in such systems and would simplify the task of providing detailed guidance that is specific to data.

In data driven-systems data often represents a major part of the system and its generation and maintenance often represent a substantial part of the cost of the system. One such strategy would be to develop data integrity requirements that allow the targeting of resources by a classification of risk, based upon the role played by data errors in the behaviour of the system and their potential affect on the overall system integrity failures due to data errors.

R3 This research recommends that the data component be considered as part of the overall system hazard or risk analysis, leading to an apportionment of the system safety requirements to data. In this way the data component would be assigned specific integrity requirements.

Once data has been apportioned integrity requirements, these data integrity requirements may then be used to direct the treatment data with appropriate rigour.

13.3 Techniques and measures to be used with the data component

If recommendations R2 and R3 are accepted than it follows that the treatment of the data component should parallel the treatment of the hardware and software components of the system. These hardware and software components are subject to an extensive array of techniques and measures recommended based upon the targeting of resources using a classification of risk.

R4 This research recommends that further research is required to establish a range of techniques and measures to be used with the data component.

These techniques and measures should be selected based upon the degree of risk that data errors might compromise the integrity of the overall system.

R5 This research also recommends that appropriate suites of metrics be developed to facilitate the selection of these techniques and measures.

This research notes that the parallel between the data component and the hardware and software components may only be limited to the development arena.

13.4 The definition of data used by safety systems

This research has presented requirements for the definition of data for use by safety systems (see section 4).

R6 This research recommends the creation of an extended data dictionary based upon the requirements discussed in section 4.2.

The extent and nature of the requirements of data definition is dependant upon the system (or systems) under consideration. Where this data and its associated data models are extensive, considerable resources may be required for the creation and management of data.

R7 This research recommends that further research is required into the definition of data used by safety systems. The author draws parallels between the hardware and software components and the recommendations for using differing arrangements or structures of these hardware and software components based upon the integrity required.

For example, high integrity systems commonly employ diversity and redundancy as a means of providing additional confidence in the integrity of the system. This additional research should examine how data diversity and redundancy may be incorporated into the data representation. These data representations may also aid both verification and validation.

These data representations may allow some relief of the burden of evidence (and the associated evidence management) that otherwise accompanies a claim for high integrity data.

R8 This research recommends that the definition of data recognise the use of data by separate systems in the form of separate data architectures for each system and that these data architectures are appropriately managed.

The author suggests that an Enterprise Data Architecture may be used as a single data definition for the overall system (or enterprise). One means to appropriately manage the collection of data architectures would be to use the enterprise data management plan to document the processes and procedures used to manage the data and the data architectures. In addition a data management plan could be used to describe both the Application Data Architecture and the definition of the application data interfaces for each application. The data descriptions in each data management plan will be derived from the data definitions within the Enterprise Data Architecture.

This author notes that part of the definition of data is the determination of its integrity requirements. A method of the determination of data integrity requirements is presented in section 5 and is supported by recommendation R3 above.

13.5 The use of the layered model when considering the use of data within a systems hierarchy

This research has presented a layered system model (see section 6).

R9 This research recommends that any proposed system be considered in the context of the layered system model.

R10 This research also recommends that the use (and re-use) of data be considered in the context of the systems that produce and consume data within the systems hierarchy.

R11 This research recommends that the layered systems model be considered as subject of future research in the context of the formulation and evaluation of safety arguments.

The layered systems model is generic and provides a means to express the relative position of one or more systems within the systems model. This model allows whole safety functions to be expressed and when linked to rules governing the extent of a safety function provides powerful criteria to assess a part or all of a proposed system solution.

R12 This research recommends that additional research is required to identify architectural patterns (and possibly anti-patterns) in existing and proposed systems to provide guidance as to suitable general arrangements for large-scale safety systems.

13.6 Lifecycle models for generic and application data

This research has identified that data design and development may be separate from the data provision. Perhaps a necessary first step in any proposed system development or system implementation is to identify the requirements of the lifecycle models for both generic and application data. These lifecycle models will require extensive co-ordination of activities, processes and personnel far beyond that which can be shown in a simple 'V' lifecycle model.

R13 This research recommends that a lifecycle model is created as part of the design and development of the generic data component;

R14 This research recognises the separation of data development and data provision and recommends that the data integrity requirements be communicated from those responsible for the safety argument of the system, to those responsible for the provision of the associated data.

R15 This research also recommends that a lifecycle model be created as part of the application data component. This application data lifecycle model will also encompass all the activities of data provision using one or more instantiations of the application data lifecycle model.

The design and implementation of a single data-driven system should take into account the data provision as well as the relationship of this individual system within the enterprise and the implied systems hierarchy.

13.7 Roles and responsibilities for those involved in the design, development and provision of data

This research has discussed the requirement for the extensive co-ordination of activities, processes and personnel in the development and implementation of data driven systems.

R16 This research recommends in any data driven system development or implementation of a data driven system that the responsibilities of a Data Design Authority be defined.

The author suggests that the Data Design Authority shall be responsible for the identification of any systems and tools required to establish the data management process. The Data Design Authority will also be responsible for the development a data schedule (or programme) to be used for review of the following:

i) Integration of data resource requirements into the master schedule;

ii) Establishment of lead time requirements for data provision;

iii) Identification of risk areas caused by peak periods of effort or event-driven data items; and,

iv) Provision for the assignment of data item originators, by name, who will be tasked with preparing individual data items.

To provide the above information, the Data Design Authority should ensure that all data sources are identified, and that the data processing requirements are also identified. To assist in the management of each application, a data management plan may also be required.

13.8 The provision of data for safety systems

This research has described data provision in the form of a generic data supply chain (see section 9). Data provision is strongly influenced by the integrity of the source data and the processes required to transport it to the systems, which will consume it. This is a multi-facetted problem; on one hand are small-scale systems whose data can be adequately managed through data entry and delivery of a validated dataset. At the other extreme are large-scale systems drawing data from a number of sources. This data is processed, transformed, consolidated, transported and finally delivered to one element of the overall system.

R17 This research recommends that the design and development of the generic data supply chain be the subject of further research. One aspect of this further research should examine patterns and re-use within the data supply chain.

R18 This research recommends that further research is required to identify architectural constraints, formations and reusable components within the Enterprise Data Processing Model.

13.9 Data ownership, the cost of data, liability for data errors and risk

This research has described the use of data provision after the system development has been completed. The author notes that data provision, in particular, may be subject to a number of financial and political influences.

R19 It is a recommendation from this research that a strategy for the data component should recognise that data may be created by a number of organisations, groups, or political bodies.

This strategy may also include one or more data supply chains and hence should recognise changes in ownership, liability and risk as data may also cross the boundaries of a number of organisations, groups, or political bodies.

R20 This research also recommends that additional research is required to establish costs associated with data and data provision.

If the true cost of a data-driven system is to be established, the costs associated with data must also be taken into account. When estimating software a variety of metrics are used to describe these attributes such as lines of code or productivity metrics. Research is required to identify metrics describing data. For example, additional research might propose metrics associated with data production such as data volume or

productivity. This research should also examine how a range of different integrity requirements might influence these metrics.

13.10 Strategies to be used in the construction of a safety argument for a data-driven system

The system safety case should consider all aspects of the system, its behaviour and the processes used in its creation, operation and maintenance. In the course of its working life, a data-driven system may be used by one or more organisations, which change and evolve over time. Organisational responsibilities and boundaries also change.

R21 This research recommends a data validation policy, based upon DO 200A, where the responsibility to ensure that the data integrity requirements have been met rests with the user of the data. This policy encourages those responsible for the system to identify and define a boundary and to use this boundary in the evaluation of the complexity of the safety argument.

It is common practice to construct a generic safety argument for a system. When this system is implemented in an application context it is also common to create an application safety argument.

The use of a data driven system may require an additional safety argument to address data provision. Where data is updated during the normal operation of the system and an additional 'process' based safety argument may also be required.

References

1. International Electrotechnical Commission; IEC 61508-1 Functional Safety of electrical / electronic / programmable electronic safety-related systems – Part 1:2000 General Requirements. Geneva, 2000.

2. D. Pumfrey; DPhil Thesis "The principled design of computer system safety analysis", Department of Computer Science, University of York. September 1999.

3. International Electrotechnical Commission; IEC 61508-4 Functional Safety of electrical / electronic / programmable electronic safety-related systems – Part 4:1998 Definitions and abbreviations. Geneva 2000.

4. A. Faulkner; *"The nature of data used by safety-related systems"*; Engineering Doctorate portfolio of Alastair Faulkner, October 2003 (Unpublished).

5. A. Faulkner; *"Data Integrity: Literature Review"*; Engineering Doctorate portfolio of Alastair Faulkner, October 2003 (Unpublished).

6. Engineering Safety Management (Yellow Book 3); Issue 3, Volumes 1 and 2 Fundamentals and guidance. Published by RAILTRACK on behalf of the UK Rail Industry; London 2000 ISBN 0-9537595-0-4

7. Ravi S Sandhu, *"On five definitions of data integrity"*, Proceedings of the IFIP WG11.3 Workshop on database security, Lake Guntersville, Alabama, Sept 12-15 1993

8. R. Courtney, *"Some informal comments about integrity and the integrity workshop", Proceedings of the invitational workshop on data integrity"*, (Ruthberg, Z. G. and Polk W. T., editors), National Institute of Standards and Technology, Special Publication 500-168, September 1989, Section A.1, pp 1-18

9. Contract Research Report 336/2001, Justifying the use of software of uncertain pedigree (SOUP) in safety-related applications, prepared by Adelard for the Health and Safety Executive, HSE Books, ISBN 0-7176-2010-7

10. Contract Research Report 337/2001, Methods for assessing the safety integrity of safety-related software of uncertain pedigree *(SOUP),* prepared by Adelard for the Health and Safety Executive, HSE Books, ISBN 0-7176-2011-5

11. J. A. McDermid; "*The cost of COTS*", IEE Colloquium - COTS and Safety critical systems. Digest Number: 97:013 London January 1997

12. I. J. Sinclair; "Use of Commercial-Off-The-Shelf (COTS) Software in Safety-related applications"; HSE books. CRR80 1995 ISBN 0-7176-0984-7

13. Aeronautica Civil of the Republic of Columbia, 'AA965 Cali Accident Report' December 20, 1995

14. Captain Rob Roland, UAL-retired, 'Last flight to Cali', http://members.aol.com/safeflt/cali.htm (27-December-2000)

15. Flight Safety Foundation, Flight Safety Digest May-June 1998, Vol. 17 No. 5/6

16. Captain Bertrand de Courville, Second AIS Symposium, 'The safety objectives', Toulouse, March 2002.

17. D. Welbourne and N. P. Bester. '*Data for Software Systems important to safety*'. GEC Journal of Research, Vol. 12, No. 1, pp50-57, 1995.

18. RTCA; DO 200A Standards for processing aeronautical data. Radio Technical Commission for Aeronautics, 28 September 1998

19. International Electrotechnical Commission; IEC 61508-3 Functional Safety of electrical / electronic / programmable electronic safety-related systems – Part 3:2000 Software Requirements. Geneva, 2000

20. A. Faulkner; "*Data Integrity: Industrial Practice*"; Engineering Doctorate portfolio of Alastair Faulkner, October 2003 (Unpublished).

21. Shawn Coyle, "*Aircraft On-Board Navigation Data Integrity A Serious Problem*", Transport Canada Database Working Group Paper Aircraft Certification, Flight Test. Transport Canada, Ottawa, Ontario 1997

22. W. S Greenwell, J C Knight, '*What should aviation safety incidents teach us?*' http://www.cs.virginia.edu/~jck/publications/safecomp.2003.lessons.pdf

23. Eurocontrol: TRS105-03 "*Data integrity assurance – pilot study*", 29 August 2003

24. Eurocontrol: TRS150-03 "*Support the development of an implementation for data integrity regulation*", 23 October 2003

25. Eurocontrol: "*Aeronautical Information Management Strategy for the years 2000+*", Volume 2, 18 December 2002, Draft V0.1

26. IEC 61511-1 *Functional safety: Safety Instrumented Systems for the process industry sector* – Committee DRAFT for Vote: 2000, Geneva: International Electrotechnical Commission, 1998.

27. UK Civil Aviation Authority, Safety Regulation Group, "*Air Traffic Services Safety Requirement*", Document CAP 670 section SW01 "Regulatory Objective for Software in Safety-related Air Traffic Services", Issue 6. London: October 2002

28. A. Harrison and R. H. Pierce. *Data Management Safety Requirements Derivation.* RAILTRACK: West Coast Route Modernisation Internal report. June 2000. RAILTRACK PLC, London 2000 (Unpublished)

29. R. Allan: Internal RAILTRACK Memorandum *"An 'Architectural Context' to assist in framing the WCRM Safety Case Argument."* Ref: R00122A.doc 22 December 2000 (Unpublished)

30. DISC Consortium, *'Research into waterborne transport area, Demonstration of ISC – DISC: Final report'*, Erik Styhr Petersen, SCL DISC Project Manager, Ref D101.00.01.047.003C pp 33-35

31. J. Tillotson; *"System Safety and Management Information Systems"*; Aspects of Safety Management: Proceedings of the ninth Safety Critical Systems Symposium Bristol UK 2001. Ed F. Redmill and T. Anderson; pp 13-34 ISBN 1-85233-411-8

32. A. Faulkner; *"The provision of data using a data supply chain"*; Engineering Doctorate portfolio of Alastair Faulkner, September 2003 (Unpublished)

33. A. Faulkner; *"The provision of data using a lifecycle model"*; Engineering Doctorate portfolio of Alastair Faulkner, December 2003 (Unpublished).

34. CENELEC European Committee for Electro-technical Standardisation. CENELEC EN 50128: Railway Applications – Communications, signalling and processing systems Software for railway control and protection systems. March 2001.

35. Hazards Forum; Safety-related systems – Guidance for engineers; ISBN 0-9525-103-0-8; Issue No. 2, London August 2002

36. A. Faulkner; *"The use (and re-use) of data within the organisation"*; Engineering Doctorate portfolio of Alastair Faulkner, Februrary 2003 (Unpublished)

37. M Page-Jones; "The practical guide to structured systems design"; ISBN: 0-917072-17-0; Yourdon Press, New York, 1980.

38. P. Atkinson, *"Feedback control theory for Engineers"*, ISBN 0-435-71813-4, Heinemann Educational Books, 2nd Ed. 1978

39. A. Faulkner and R. H. Pierce, *"Is it Software or Is it data"*; Proceedings of the 19th International Safety System Conference 2001, Huntsville. Alabama, USA. pp 323-329

40. A. Faulkner; *"Data Integrity: Published papers"*; Engineering Doctorate portfolio of Alastair Faulkner, October 2003 (Unpublished).

41. N. Storey and A. Faulkner: "*Data Management in Data-Driven Safety-Related Systems*"; Proceedings of the 20th International Safety System Conference 2002, Denver. Colorado USA. pp 466-475 ISBN 0-9721385-1-X

42. A. Faulkner and N. Storey: "*Data: An often-ignored component of safety-related systems*". Proceedings of the MoD *Equipment Safety Assurance Symposium ESAS02*, Bristol, UK. Oct. 2002

43. A. Faulkner and N. Storey, "*Strategies for the Management of Data-Intensive Safety-Related Systems*", Proceedings of the 21st International Safety System Conference 2003, Ottawa. Canada. pp 855-864, ISBN 0-9721385-2-8

44. N. Storey and A. Faulkner: "*Data Requirements for Data-Intensive Safety-Related Systems*", Proceedings of the 21st International Safety System Conference 2003, Ottawa. Canada. pp 865-874, ISBN 0-9721385-2-8

45. Letter from Ron Pierce, secretary of IEC 65A MT12, dated 17 December 2003.

46. A. Faulkner; "*An introduction to data-driven safety-related systems*"; Engineering Doctorate portfolio of Alastair Faulkner, May 2002 (Unpublished).

47. A. Faulkner; "*Data integrity requirements*"; Engineering Doctorate portfolio of Alastair Faulkner, December 2002 (Unpublished).

48. A. Faulkner, P. A. Bennett, R. H. Pierce, I. H. A. Johnston and N. Storey: "*The safety management of data-driven safety-related Systems*", Lecture notes in computer science - Proceedings of the 19th International Conference SafeComp 2000, pp 86-95, ISBN 3-540-41186-0

49. A. Faulkner, "*The use of safety-related data within a railway command and control system*"; Proceedings of the Engineering Doctorate Conference 2001, IMC Univ. of Warwick

50. N. Storey and A. Faulkner, "*The role of data in safety-related systems*"; Proceedings of the 19th International Safety System Conference 2001, Huntsville. Alabama, USA. pp 26-35

51. A. Faulkner and N. Storey, "*The role of data in safety-related railway control systems*"; Proceedings of the 19th International Safety System Conference 2001, Huntsville. Alabama, USA. pp 793-800

52. A. Faulkner: "*Safer Data: The use of data in the context of a railway control system*", Proceedings of the tenth Safety-critical Systems Symposium, pp 217-230 ISBN: 1-85233-561-0, Southampton, UK 2002.

53. A. Faulkner, "*Data integrity: Assessing data used by safety-related control systems*"; Proceedings of the Engineering Doctorate Conference 2002, IMC Univ. of Warwick

54. A. Faulkner and A Harrison, "Is asset information '*good enough*' to be used by safety-related systems", Proceedings of Engineering Asset Management, London, 2002 ERA Report 2002-0409, ISBN 0 7008 0765 9

55. R. H. Pierce, M. Nicholson and A. Faulkner, "*Assessing Operating Systems for Use in Safety-related Systems*", Proceedings of the 21st International Safety System Conference 2003, Ottawa. Canada. pp 581-590, ISBN 0-9721385-2-8

56. A. Faulkner, "*Data-intensive systems: Sourcing data of the required integrity, the problem of origination*"; Proceedings of the Engineering Doctorate Conference 2003, IMC Univ. of Warwick

57. N. Storey and A. Faulkner: "*The Characteristics of Data in Data-Intensive Safety-Related Systems*", Lecture notes in computer science - Proceedings of the 22nd International Conference SafeComp 2003, pp 396-409, ISBN 3-540-20126-2

58. A. Faulkner; "*Data Integrity: Journal Paper*"; Engineering Doctorate portfolio of Alastair Faulkner, October 2003 (Unpublished).

59. N. Storey and A. Faulkner: "*Data - The Forgotten System Component?*"; "*Journal of System Safety*"; A publication of the System Safety Society, Volume 39, No 4 Fourth Quarter 2003, pp10-14

60. ISO Model is a collection of documents describing the services provided by the model; The book "*Communications Network Protocols* 3rd Edition, Appendix B (Selected ISO Standards), by Brian Marsden provides a comprehensive list of these documents; ISBN 91-44-23043-5; Chartwell-Bratt Ltd 1992.

61. P. Checkland; "*Systems thinking, systems practice*", ISBN 0-471-27911-0, John Wiley 1993.

62. D. E. Avison and A. T. Wood-Harper; "*Multiview: An exploration in information systems development*"; ISBN: 0-632-03026-7, Alfred Waller Ltd Reprinted 1990.

63. Open University; "*Human aspects of man-made systems*"; ISBN 0-335-00047-9; Open University Press 1977.

Appendix A Structure of the Engineering Doctorate Portfolio

A.1 Overview of the submissions

The Engineering Doctorate consists of a portfolio of work, which is summarised by this *Executive Summary*. The submissions included within the portfolio are:

i) Introduction to data-driven safety-related systems

This submission [46] provides an introduction to data-driven safety-related systems and sets the context for the research.

ii) Data integrity requirements

This submission [47] discusses the safety concepts of hazard, opportunity, and accident, together with errors, faults and failures [6]. This discussion then develops a classification for data together with a set of guidewords for each data classification.

iii) The use (and re-use) of data within the organisation

This submission [36] recognises that *data-driven* and data intensive systems often form part of a systems hierarchy. These systems exchange data through their respective interfaces and may also use other shared data such as configuration data.

The submission presents discussion of a systems model in which the relative position of each system may be represented. This systems model may also be used to describe coupling and cohesion between these systems elements.

This is a general-purpose systems model and may be of use in the justification of other forms of safety-related systems.

iv) The provision of data using a data supply chain

This submission [32] discusses the issues associated with the use of a data supply chain. The description of the data supply chain is based upon DO 200A [18]. The DO 200A data supply chain description is extended into a generalised form to facilitate its wider application. The submission also presents a design method for data supply chains, together with a diagrammatic notation.

This submission identifies the issues associated with ownership and responsibility, particularity the ownership of data errors and the implicit liability often associated with these errors, faults and failure.

v) Data integrity: published papers

This submission [40] contains fourteen published papers [41 to 44, 48 to 57] together with a brief review of each paper identifying the salient points and the contribution of the authors.

The paper "*Data Management in Data-Driven Safety-Related Systems*" [41] won the "Best Paper" award of the 20th International Safety System Conference 2002, Denver. Colorado USA.

vi) Literature review

This submission [5] contains the literature review.

vii) The nature of data used by safety-related systems

This submission [4] combines a number of elements from existing submissions to provide a focused discussion on the 'nature' of data. The purpose of this submission is to provide a summary of the descriptions of the 'nature' of data that have been proposed, albeit informally, during the course of this research.

viii) Industrial practice

This submission [20] records industrial practice relating to *data-driven* systems through twelve structured interviews and examination of supporting documents.

ix) Data integrity: Journal paper

This submission [58] contains a journal paper [59], together with a brief review.

x) The provision of data using a lifecycle model

This submission [33] recognises that once data is treated as a separate systems component, the apportionment of the system safety requirements demand that data is treated with the same rigour as the hardware and software components of the system component, including the use of an appropriate lifecycle model. The data lifecycle model is complicated by data production (or data extraction), as these activities are typically external to the development environment. The data lifecycle must also take account of data provision to ensure that data of the required integrity is available when the system enters service. While the system is in service data production and data provision may also be required to provide in-service data updates.

A.2 A suggested order of reading the submissions

The following is a suggested order of reading:

i) The nature of data used by safety-related systems.

ii) Literature review.

iii) Industrial practice.

iv) Data integrity requirements.

v) The use (and re-use) of data within the organisation.

vi) The provision of data using a lifecycle model.

vii) The provision of data using a data supply chain.

viii) Data integrity: Published papers.

ix) Data integrity: Journal paper.

The submission "*Introduction to data-driven safety-related systems*" has been superseded by "*the nature of data used by safety-related systems*".

Appendix B A layered model for a hierarchy of systems

B.1 Introduction

It is common for large distributed systems to be composed of many interrelated subsystems. One of the challenges in managing large-scale systems is demonstrating that any change made to the system or subsystem is independent and does not also induce undesired changes. In such interrelated (and interdependent) systems even small changes may require extensive justification.

This research has developed a layered model to explore and express the relationships within these large distributed systems. As already discussed in section 6, the form of these distributed systems varies tremendously; one can gain an insight into the interaction between system elements by considering a typical distributed control system. For this purpose, section 6 introduced the very specific example of a large-scale railway control system, though it should be noted that not all systems would follow this scheme.

The application of this layered model should not be restricted to the discussion of data and data integrity. Subsystems may only exchange limited data volumes. This layered model may also be used to express the influence of functional and non functional behaviour in the context of the overall system.

B.2 Describing the layered model

B.2.1 The use (and re-use) of data within an organisation

It is common for a hierarchy of control systems to use shared definitions of the physical environment as configuration data. Each of these systems requires data to satisfy the desired functional behaviour using a range of data at differing levels of abstraction and detail. Where low-level protection devices are used, these devices are generally subordinate to the control system. These protection devices combine low-level data describing a small number of physical devices to create a high integrity protection system. Several low-level protection devices may be used within one control area under the supervision of the control system and its operator.

However such distributed systems need not be data-intensive. The layered model may also be used to express the influence of the subsystem in terms of the functionality it provides.

B.2.2 A generalised description of the layered model

The generalised description of the layered model is based upon consideration of a distributed system. In particular the layered model seeks to establish the role of each layer and the relationship between layers. The following is very specific example of a large-scale railway control system. It should be noted that not all systems would follow this scheme. This model is developed from an incomplete and unpublished RAILTRACK internal memorandum by Richard Allan [29]. In addition, elements of the layered model are also drawn from concepts contained within the '*Basic Reference Model for Open Systems Interconnection*' (ISO OSI) model [60]. This latter model partitions communications services between 7 layers with defined interfaces and peer protocols that permit the separation of application development from the underlying communication system.

Two important elements of the OSI model are that each layer [29]:

i) Communicates with its peer layer in a different communication unit; and

ii) Provides a service to the layer above and expects a particular kind of service from the layer below.

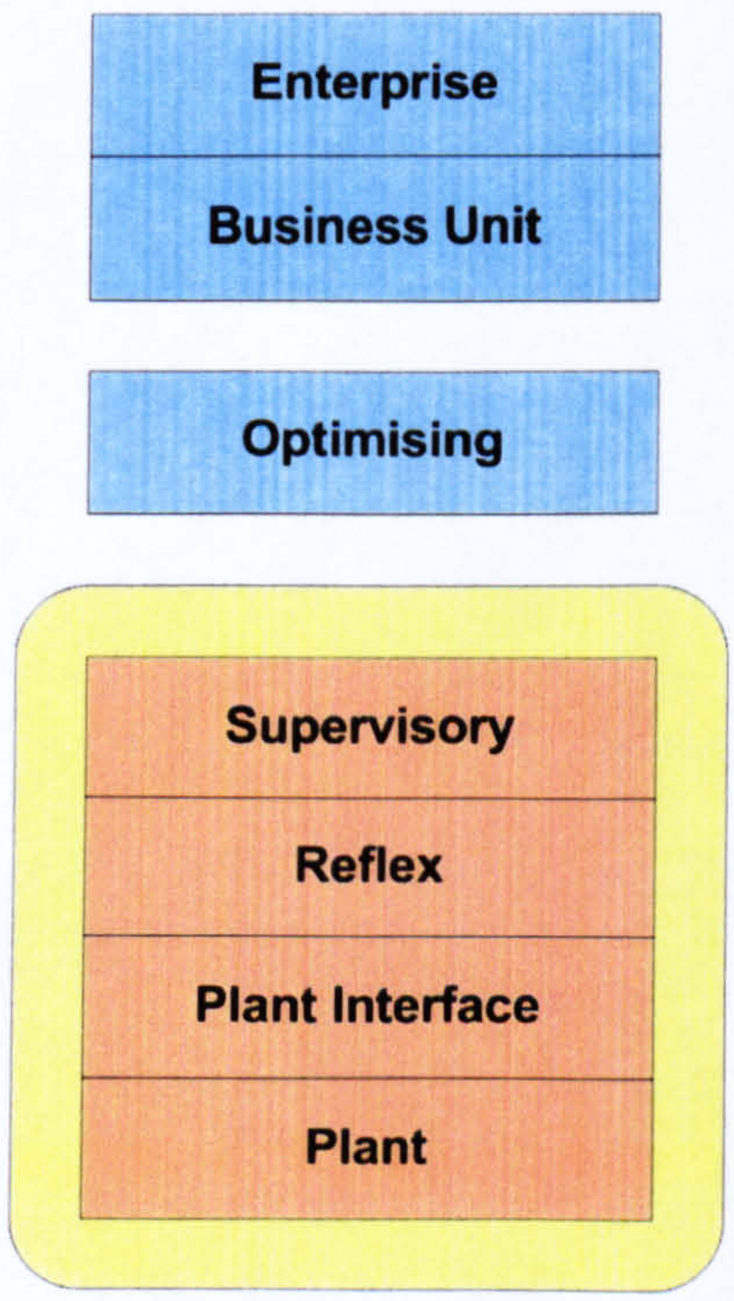


Figure 20: A layered model for a hierarchy of systems

The model, in Figure 20, describes an abstraction into layers and allows the development, replacement of the underlying layers based upon respect for the services provide by the layer and preservation of the interfaces. These layers are:

i) The '*plant*' layer represents single instances of elements of the plant infrastructure, the physical equipment;

ii) The '*plant interface*' layer represents the interface to plant infrastructure elements. In essence this layer converts physical phenomenon from sensors (including feedback from actuators) into abstract representations such as electrical signals or data. This layer also provides the control interfaces to actuators;

iii) The '*reflex*' layer is the lowest layer at which the measured status is interpreted and control (or protection) actions are carried out. These actions may be based upon information (which may include stored information), any demands upon the system and some set of rules. In this reflex layer the rules and information completely determine the control action. These reflex actions are in essence rule based. In principle all activities in the reflex layer can be automated. Where a protection system does not require the intervention of an operator these protection systems are described as reflex actions. Safety-critical functions commonly require a fast response and therefore often make use of reflex actions;

iv) The '*supervisory*' layer represents a more complex level of control. This complexity may be a result of large-scale operation, integrating a number of dissimilar functions, or interpreting complex (or ambiguous) data (or some combination of these). The distinction between the reflex and supervisory layers is the judgement or knowledge that must be applied, particularly in degraded or emergency situations. Supervisory systems are characterised by the need to support the judgement of the operator doing the supervision. Predominantly the supervisory layer is downward looking viewing the performance of the lower levels;

v) The '*optimisation'* layer represents the most sophisticated control layer. At its most developed the optimisation layer should maximise the use of resources from the delivery of the service. The optimisation layer should respect the performance and safety constraints of the underlying (transportation) system. The information demands on the optimisation layer are high, requiring a full understanding of the underlying system, the planned service and contingency plans. The full understanding of the underlying system includes the performance capabilities and constraints of intervening layers of the system;

vi) The '*business unit'* layer represents the divisional responsibility of the delivery of the planned service. This layer normally plays little part in the real-time operation of the operational parts of the system being more concerned with the medium term maintenance (including competencies) and development of the infrastructure, and the subsequent future delivery of the planned service. The business unit will become involved in the short-term operation of the system in response to a serious incident that cause substantial impact on the delivery of the service.

vii) The '*enterprise'* layer represents the corporate entity; responsible for the planning and execution of large-scale changes to the infrastructure; responding to changes in legislation; setting and maintaining standards, procedures and competency requirements.

The point is well made by Richard Allan [29], that implementation of large-scale control systems requires a framework in which to express the role played by respective system components and provide a mechanism by which a large-scale system safety may be argued. . The author concurs with Richard Allan [29] in considering that the supervisory layer should be the highest layer at which a safety function should be implemented. This boundary depicted in Figure 20 by the box surrounding the plant, plant interface, reflex and supervisory layers.

The optimisation layer should take into account knowledge of current and possible future operational conditions. These operational conditions may be restrictions on the use of the transportation infrastructure due to planned or unplanned maintenance. Optimisation is therefore required to respect the performance and safety constraints of the system. Clearly, optimisation should only employ safe functions; that is, the optimisation of the execution of the planned service should not be capable of compromising the safety of the system. Hence the upper limit of the safety function should be the supervisory layer.

B.3 Coupling and independence

B.3.1 Introduction

Coupling and cohesion are ways of describing the functional decomposition or partitioning of the design. Coupling is concerned with the interrelationships between elements of the design (modules), whilst cohesion is concerned with how well the elements within a module are related to one another.

B.3.2 Coupling

Structured methodologies are by definition concerned with the structure of the solution by addressing the partition of the solution into modules. The division of the problem into '*manageable*' components is used to deal with complexity.

The classification of coupling has been proposed in a number of design methodologies and this submission uses the example of a structured methodology. Although primarily concerned with the structure of software, structured methodologies combine a number of techniques and measures to describe the solution. Structured methodologies represent the hard systems view and hence are focused upon the engineered solution. This is in contrast to the soft systems methodologies, which attempt to take a more human centred approach, particularly but not exclusively to requirements elicitation [61 to 63].

In a systems context coupling need not be restricted to a single design. The system will interact with its environment and quite possibly other instantiations of the same or similar systems as peers or with subordinate and supervisory systems. Therefore the concept of coupling should be extended to consider vertical coupling between the system and subordinate and supervisory systems, and horizontal coupling between other instantiation of the same or similar systems as peers.

In the context of this research, coupling is primarily based upon the exchange of data either through a shared (static) description of the infrastructure or dynamic data passed across the system boundary [36].

B.3.2.1 Vertical coupling

Vertical coupling refers to coupling between different adjacent vertical layers of the layered model. An example of vertical coupling is instantaneous status data passed from the reflex layer to the supervisory layer.

Data volumes are characterised by the control strategy that also may include some stored data. The degree of coupling whether tightly or loosely coupled will influence the resources required to either maintain or upgrade elements within the control system.

B.3.2.2 Horizontal coupling

Horizontal coupling refers to coupling between the same adjacent layers of the layered model. An example of horizontal coupling is instantaneous status data passed from one instance of the reflex layer to one or more instances of the reflex layer within a control strategy.

From the description above data is passed horizontally between layers. Layers within the layered model are therefore horizontally coupled. The degree of coupling whether tightly or loosely coupled will influence the resources required to either maintain or upgrade elements within the control system.

B.3.3 A single control system

A practical control system will use both vertical and horizontal coupling, especially where the control strategy employs a number of similar components. These control elements may possess subtle or gross behavioural differences as in many cases they may be supplied by a range of different manufactures.

Figure 21 depicts the physical components of an operator workstation, excluding the human operator (and by inference the optimisation an operator may use in the course of their normal duties). The control of two protection systems is combined within this workstation. In this example, each protection system consists of two plant interface units.

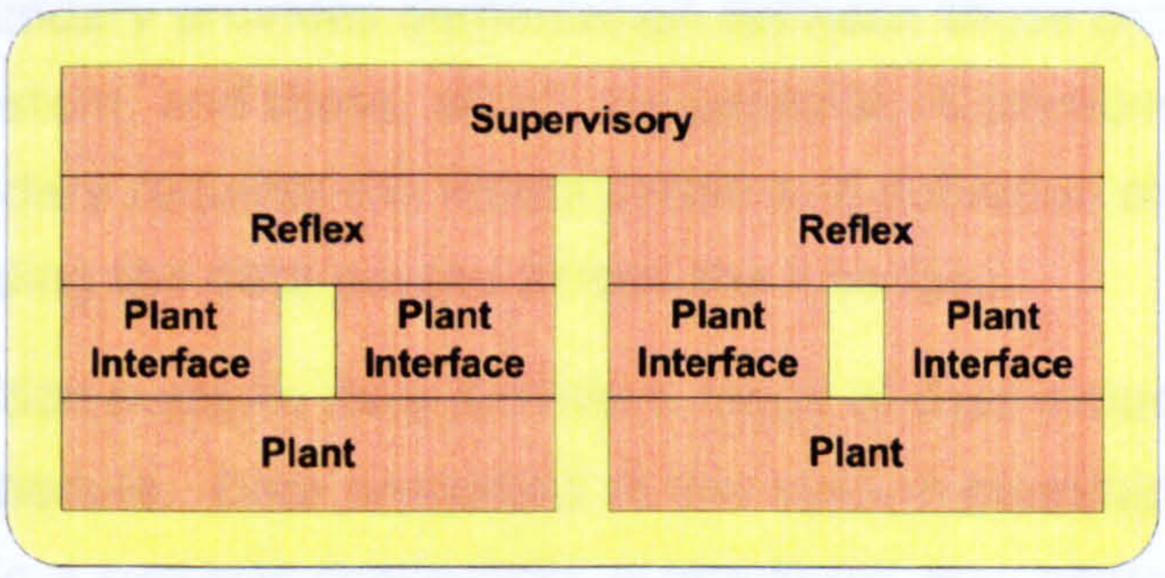


Figure 21: Coupling within an operator workstation

Although this is a simplistic representation, both the vertical and horizontal coupling of each component may be identified. Tightly coupled components are difficult to modify without modification of adjacent components. On the other hand loosely coupled component are more readily modified or upgraded (even replaced) without necessitating modification of adjacent components.

Modification or upgrade (even replacement) should also consider the properties of the data component, as well as the fit-form-function properties normally considered for the mechanical, electrical or software components of the system.

B.3.4 Independence

In this research data independence is taken to be the ability to maintain or upgrade (even replace) data components, where these data components possess the good design properties of low coupling and high cohesion.

Earlier discussions within this section have described coupling as a measure of the modularity and association of the system elements. The primary purpose of this modularisation is the management of complexity through the division of the problem at hand into successively more manageable components. This division should be considered not only at the system, sub-system, application and process, but also particularly for open systems, which interact with their environment and the data, used within the systems of systems context. A system possessing low coupling and high cohesion will generally possess the desirable design properties of resilience and stability but, in addition, low coupling and high cohesion will also minimise the effects that changes to one component have on other components of the system. A good data design will also possess these desirable properties of low coupling and high cohesion, and will therefore also possess the design properties of resilience and stability.

B.4 System boundary issues

B.4.1 Data exchanged across the systems boundary

The definition of the system boundary is an essential step in the definition of the system. The boundary provides demarcation between those components, which are within the system, and those, which are external. Communication across the system boundary requires the identification and definition of an interface description including the data passed across the interface.

External information systems may provide a range of data including status data as well as the schedule. Data presented at this system boundary may not be error free. Within a systems context these interfaces may contain implicit transformations between the external and internal use of data. In some cases the internal and external meanings of this data may also be different. Section 7.3, Figure 7 illustrates a generic system boundary and its associated issues.

Data presented at the system boundary by an external system will be transformed or adapted from the external representation to the internal

representation of the control system. This transformation or adaptation may occur in some automated function or may require manual intervention. This data will also require verification. Analysis of the control system design is required to establish the consequence of faults in this data as this data passes across the boundary of the control system. Further analysis should also establish the sensitivity to changes in this data. Such analysis will facilitate the definition of 'properties' or rules by which faults in the data may be detected at the boundary of the system.

B.4.2 Data verification at the system boundary

Data presented at the system boundary may contain errors. Error discovery is most effective when it is achieved at the system boundary. The management of data, which contains errors, will require some form of quarantine to separate out erroneous data.

The ultimate responsibility for ensuring that data meets its data integrity requirements for its intended use rests with the end-user of the data. This position is supported by DO 200A [18]. The end-user provides design criteria and hence may influence the mitigation of hazards through data error detection at the system boundary.

Whilst gross errors may be easily recognised as these will often breach some known pre-condition. Plausible errors are more difficult to detect, requiring additional pre-conditions and possibly additional redundant or duplicate data to facilitate error detection.

Data presented at the interface should be of an integrity required by the using system. Hence the system supplying the data should be of similar or higher integrity than the using system. The rules for passing data between systems of differing integrity are discussed in sections 6.5.

B.4.3 Automation and manual intervention

Error detection will generally be undertaken on both sides of the system boundary by each system. Any detected data errors will be reported to the DRACAS and where a transmission error is detected the data re-transmitted. Many external information systems may not facilitate the automated processing of these detected data errors, and therefore some form of manual intervention may be required.

It may be desirable that error detection may be undertaken by a software application. In many cases it is desirable that error detection is automated rather than automatic. Many existing system may present data at the system

boundary that does not conform to the required definition or data quality requirements. In such cases manual intervention may be required where addition interpretation and context is required from the human operator. However error detection may not always be undertaken on a single message, or data item. The use of a schedule provides an insight into this problem. A schedule is unlikely to be downloaded as a single transmission. Indeed this schedule may have a number of updates, over an extended period of time, taking account of persistent operation conditions or simply to modify (add or remove) a planned service. Updates may be delivered as single service descriptions. The update mechanism may require that each update be applied in sequence. Each data update may also add an additional service or modify an existing service or even delete the service entirely. In addition update messages may be used to add, delete, and modify the static data, which describes the service to be provided. Therefore error detection may take a number of forms, including manual intervention.

Manual intervention is not only required where the automated error detection detects badly formed messages but also where messages are missing (based upon update message sequence number) or where the described service uses attributes of the service to describe implausible or improbable services.

Anecdotal evidence suggests that the submission of flight plans within European airspace requires a significant manual intervention (of greater than 30% of all flight plans) before these flight plans may be submitted to the flight planning system. Anecdotal evidence also suggests that errors in flight plans, which enter the flight planning system, have the potential to cause significant system failures.

Appendix C The design and construction of a data supply chain

C.1 The design and construction of a data supply chain

C.1.1 The portfolio submission: "*The provision of data using data supply chain*"

This appendix is based upon the EngD submission "*The provision of data using data supply chain*" [32] and draws heavily from it. The concept of the data supply chain is based upon the aeronautical data chain set out in DO 200A [18]. The author contributes two additional *phases* to this description. A further contribution is the description of a method for the design of a data supply chain, including a graphical representation. When considering the design of a data supply chain it may become evident that combinations of phases are used several times. These phases may be collected together in formations to provide combinational re-use.

Well-defined data supply chains are not in common use. One exception is the provision of aeronautical navigation data for use in Air Traffic Management (ATM). The design and definition of the data supply chain should consider the capability of the organisations to supply, meet, and maintain the data integrity requirements. Where these criteria cannot be met alternative systems designs should be considered which apportion lower integrity requirements to the data component or a design, which employs alternatives to data-driven (or data intensive) systems.

The assertion that supplied data is of the required integrity can only be substantiated if the data is accompanied by evidence supporting this claim. When data passes along a data supply chain that crosses organisational or political boundaries additional considerations include the ownership of the data, and liability for any data errors.

Data provision demands a systematic, ordered design based upon the requirements of the system and the ability of the organisation to support data provision. It is possible to design robust data supply chains, but if the organisation does not or cannot support the provision of data of suitable integrity, the integrity of the operational system will be compromised. In extreme cases this may lead to system failures, harm or significant loss.

Good system design supports and enhances the capabilities of the organisation and may also reflect aspects of culture and education. Therefore the process for data provision should be designed to support the capability of the organisation to supply, meet, and maintain the data integrity requirements. Where these criteria cannot be met alternatives to data-driven (or data-intensive) systems should be employed.

C.1.2 A graphical representation of a data supply chain

The following graphical representation provides a means of presenting the data supply chain as a directed graph. For ease of representation standard flowchart symbols are re-used, as these are commonly available in a number of commercial diagramming packages. These reused flowchart symbols are shown in Table 2.

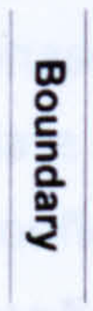
The key elements of the representation are the datasets, the phases, which act upon them and the criteria and evidence that support them. Corrective actions are represented within the error reporting blocks and should be taken to represent part of a DRACAS process.

To simplify the description of the data supply chain the following symbols are used to represent each phase.

Table 2: Data supply chain symbols

	This symbol is used to represent the data supply chain phase. These phases will be one of *assemble*, *receive*, *translate*, *distribute*, *select*, *transform* and *consolidate* phases.
	This symbol is used to represent a data store or a *dataset*
	This symbol is used to represent the *available evidence*. This evidence may take the form of physical evidence or a reference to the physical evidence.
	This symbol is used to represent the *error reports* as part of the DRACAS
	This symbol is used to represent the verification or validation criteria

This symbol is used to represent the distribution media, either physical media such as magnetic media (disk or tape), or logical media used for electronic transmission.



This symbol is used to represent an organisational or political boundary across which data is passed.

C.2 The design of a data supply chain

C.2.1 A design method for a data supply chain

This design method consists of seven steps to provide systematic, ordered design based upon the data integrity requirements of the system. In using this method the designer must bear in mind the ability of the organisation to support data provision.

Any component, phase or data chain, which attracts high integrity requirements, should be reviewed and where possible alternative arrangements of the data chain should be considered. This may also require a review of the system design and the apportionment of the system integrity requirements amongst the system components (hardware, software, data, operational process and *procedure). In* extreme cases the system design may be rejected in favour of one, which contains a re-distribution of *the system integrity requ*irements. If the high integrity requirement remains after this review, additional confidence may be required from diverse tools, processes and where possible diverse data sources.

Figure 22 to Figure 27 are used to describe each step the data chain design method, using an example of a data origin, two organisations and a consuming system. The reader's attention is drawn to the use of the cloud shape to highlight the area of the design being considered in that methodological step. These diagrams are presented in a small scale to illustrate the design method. The intention is to draw the reader's attention to the different layers within the data supply chain. A more detailed description is presented in the submission *the provision of data using data supply chain* [32].

The open box shapes represent the organisations within the data supply chain and the vertical sides of these boxes are intended to indicate the limits of each organisation. In some cases this will also represent the limitations of responsibility and liability.

The basic design method is described below:

i) Identify the data origins

This will be the start of the data supply chain. As a general rule the first phase will be a *receive* phase. Where data has a known integrity this data may be stored for use by subsequent phases. If additional confidence in the integrity of the data is required then this additional confidence may be attained through test, analysis and where necessary simulation.

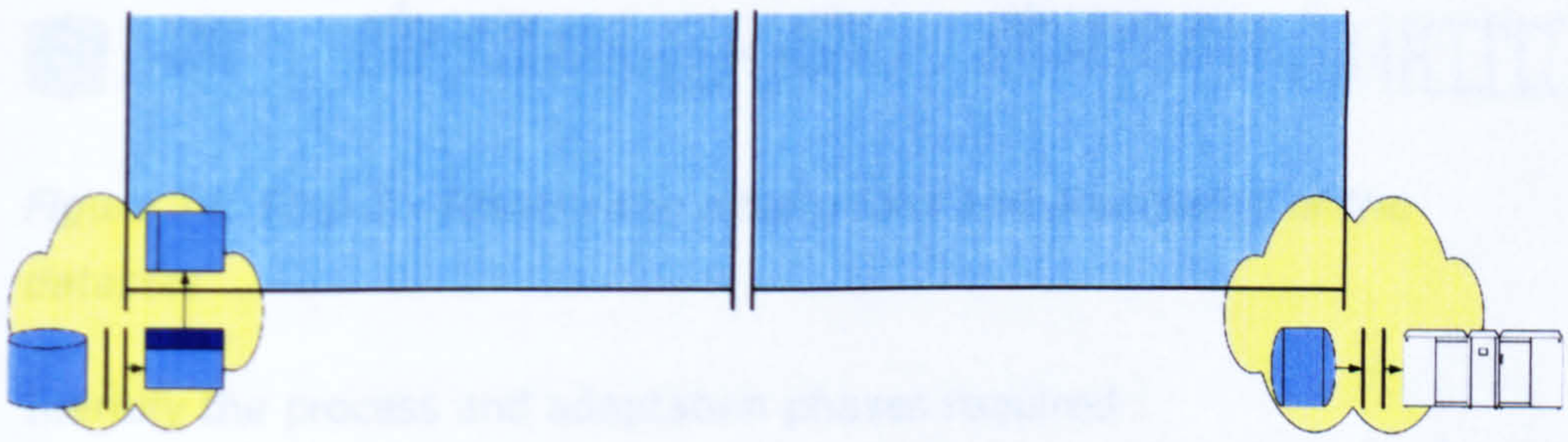


Figure 22: Step 1 - Identify the data origins

ii) Identify the (organisational, legal, and political) boundaries within the supply chain

Changes of responsibility will require that the data supply chain recognises the boundary and creates a suitable distribution media. As a general rule the *select*, *format* and *distribute* phases are used to create this media. The exchange need not be a physical media but may be an automated exchange. The important requirement is to recognise the change of responsibility, and possibly ownership. On the other side of the boundary will be a *receive* phase.

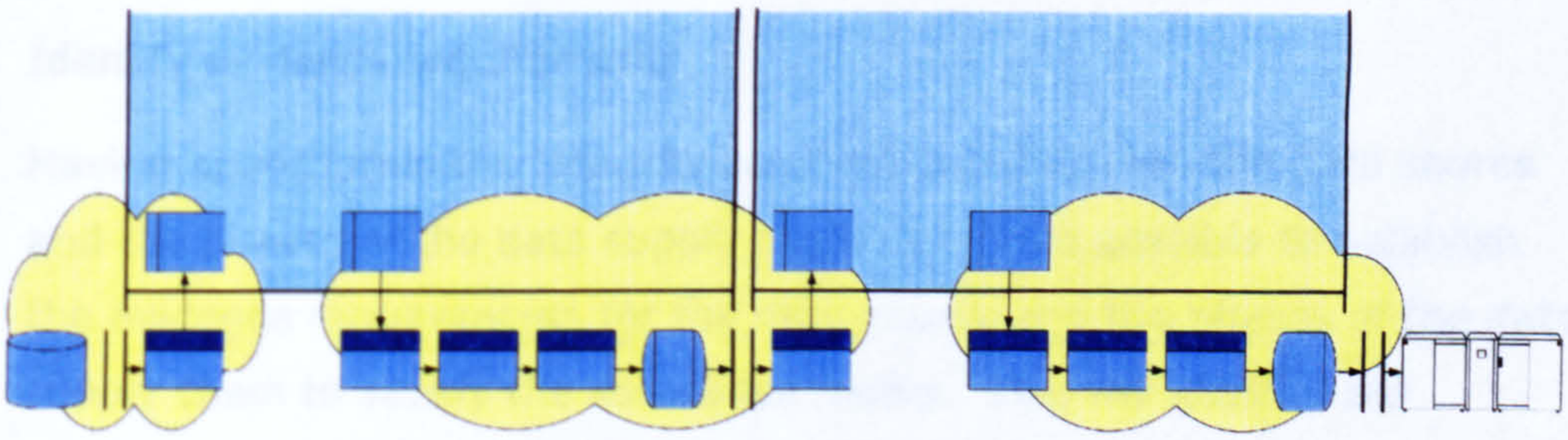


Figure 23: Step 2 - Identify the boundaries

iii) Identify the process and adaptation phases required

The major interface issues are established. The next step is to plan the required processing and adaptation of the datasets. This may be achieved with the *assemble*, *translate*, *select* and *transform* phases.

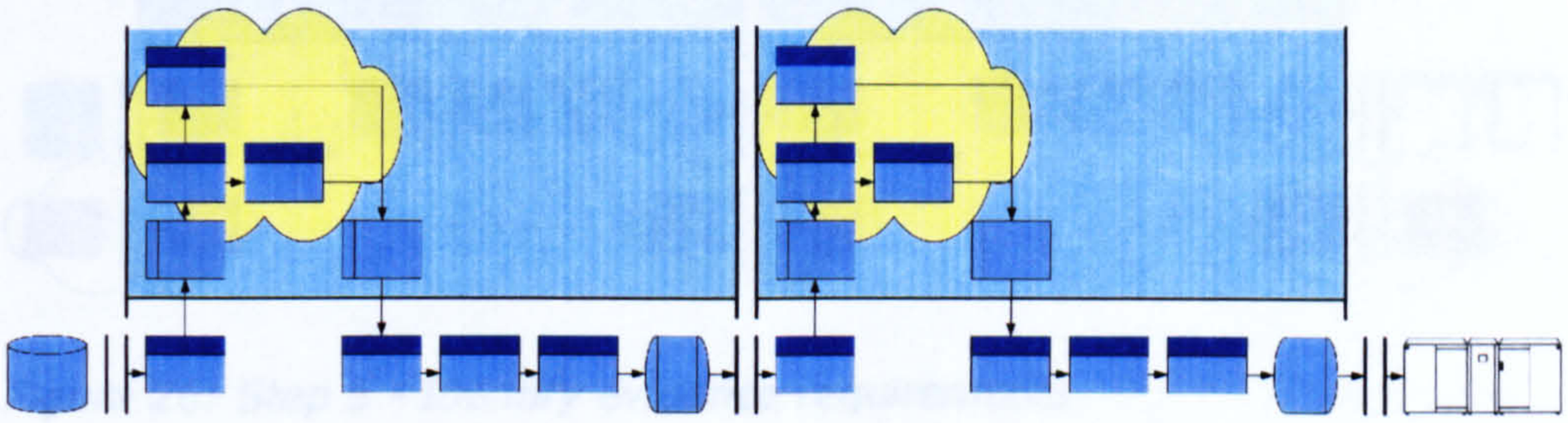


Figure 24: Step 3 - Identify the adaptations and processing of the datasets

iv) Identify the process and adaptation phases required

The integrity requirements for the data are specified. These integrity requirements are then apportioned between the data source and the phases of the data supply chain.

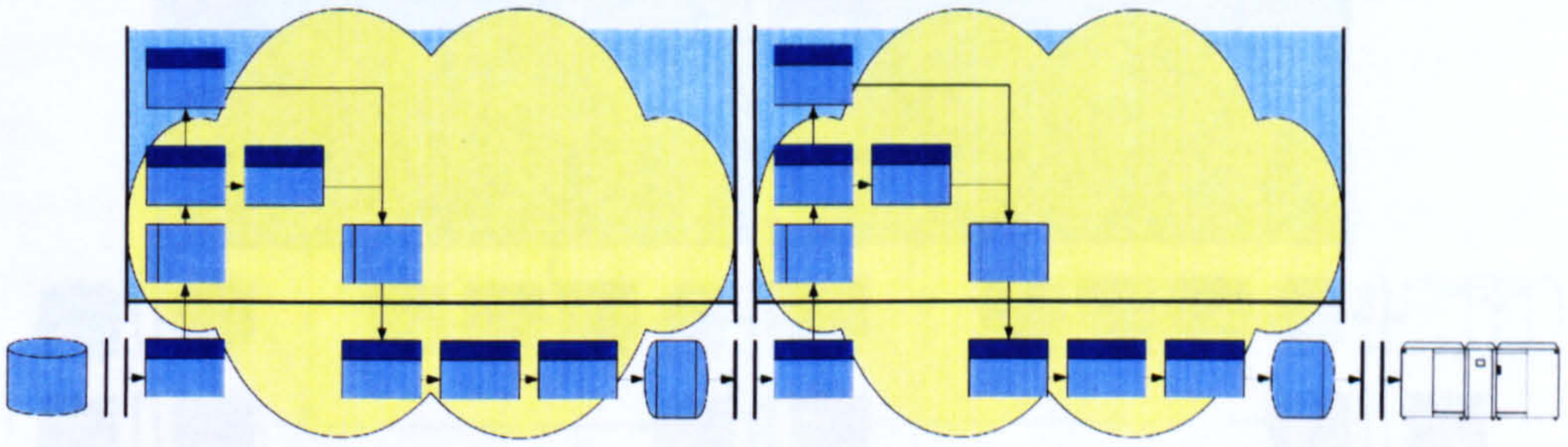


Figure 25: Step 4 - Apportion the integrity requirements

v) Identify evidence requirements

Having apportioned the integrity requirements between the data source and the phases of the data supply chain, it is then possible to establish the evidence requirements for the data source and the phases of the data supply chain to satisfy the assurance model. This will identify the verification criteria required by each phase.

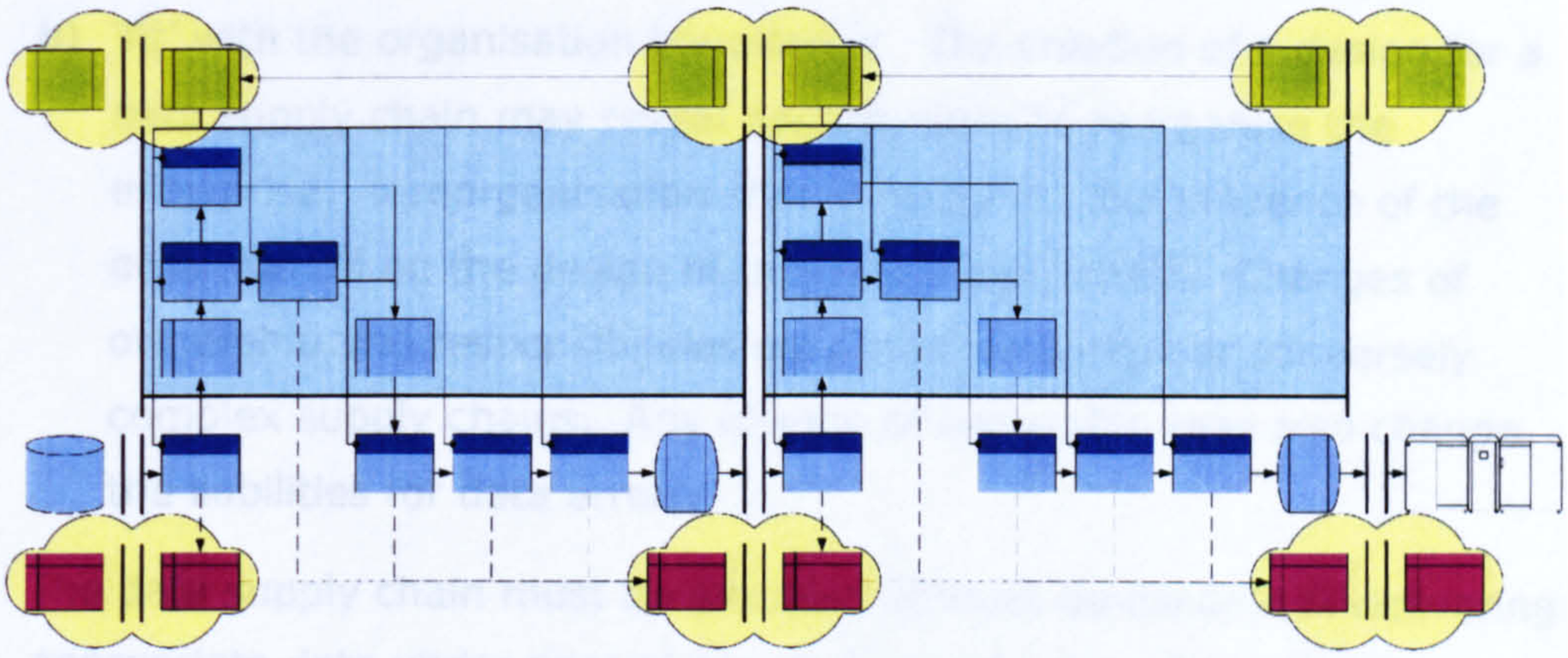


Figure 26: Step 5 - Identify evidence requirements

vi) Specify corrective action process

Failure to satisfy verification criteria of each phase will create error reports, which require corrective action. The final step is to identify the corrective action process for each phase; group of phases and the data supply chain as a whole.

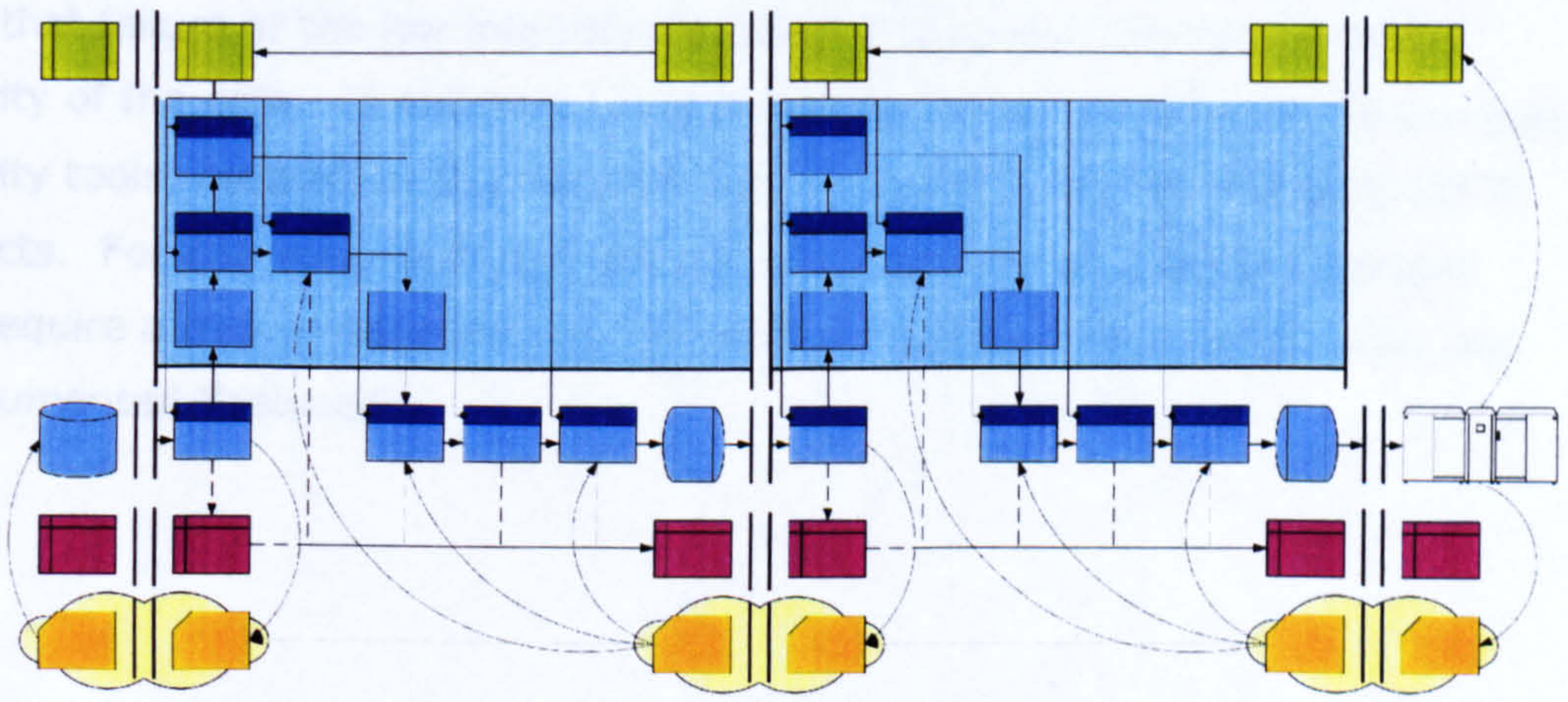


Figure 27: Step 6 - Specify the corrective action process

vii) Assess the design of the data supply chain;

Repeat steps 1 to 7 as necessary to attain required goals (integrity organisational responsibilities, liabilities, ownerships). These goals include:

a) the required integrity,

b) 'fit' with the organisation boundaries. The creation of a design for a data supply chain may reveal opportunities to reorganise the enterprise. A reorganisation should recognise the influence of the organisation on the design of the data supply chain. Changes of ownership and responsibilities may produce simple or conversely complex supply chains. Any change of ownership may also change the liabilities for data errors.

Note: The data supply chain must be practical; it must be capable of delivering appropriate data under normal operating conditions. Consideration should also be given to the failure modes of the data supply chain, its phases and their components.

The use of low integrity tools, such as databases, should only be considered where adequate error detection schemes are employed. When low integrity tools are used, these should be assessed to demonstrate that failure of these tools would not introduce data errors. However as these are low integrity tools, consideration ought to be given to how much assessment evidence is required to show that failure of the low integrity component does not interfere with the integrity of the data. In extreme cases it may be more cost effective to use high integrity tools, because of the frequency of update of these low integrity COTS products. For example, each new release of a commercial database product may require a new assessment, as the database product may contain new and undocumented 'features'.