Unified Modelling of Aerospace Systems: A Bond Graph Approach

Dominic John Diston

Thesis submitted for the degree of Doctor of Philosophy
in Electronics and Electrical Engineering
at the University of Glasgow

January 1999

Preface

The intention behind this dissertation is to offer something different from other doctoral research in aerospace systems engineering. It builds a "picture" of *Vehicle Systems Integration* and asks questions about the role of individual systems within a "system of systems" and about the role of mathematical models in helping engineers to understand how such a system is composed. The key issue to be addressed is the feasibility of a common notation for representing system behaviour and the insight into integrated design that can be gained as a result.

Systems can be given a unified purpose, functionality can be shared and needless duplication can be avoided .... but here the problem starts! In the absence of segregation, a system is *"open"* in the sense the energy and information can be freely exchanged between constituent systems. Correct control of one system is then dependent on correct control of other systems and failure effects can propagate across system boundaries.

Many years from now, this will probably warrant a footnote in the history of science and technology and people will wonder what all the fuss was about. Right now, this issue is unresolved and there is little practical experience of modelling integrated systems, potentially extending to the so-called virtual aircraft.

This dissertation sets out an approach to the modelling which expressly draws together many types of system into *one* process. A hypothetical air vehicle has been invented for this purpose, containing functionally diverse and interactive systems which are typical of many advanced vehicle concepts. The technical problems are interesting because a lot of systems are squeezed into a compact airframe. Traditional disciplines (*e.g.* mechanical, electrical, hydraulic) are becoming more closely related and generic skills (*e.g.* simulation, control) will play a more prominent role. 'Modelling' is a key integrating technology because it deals with the functionality, performance and operability of complete systems.

The motivation for this work has grown out of almost twenty years in the aerospace business, covering periods in Aerodynamics, Structural Dynamics, Flight Control, General Systems and Advanced Projects. Experience has been gained on Buccaneer and F-4 Phantom (in the good old days), Harrier, Nimrod MRA4, as well as research into Integrated Flight/Propulsion Control and Computer-Aided Control Engineering. As such, my perspectives may be different from those of other researchers -- not necessarily better or worse, just different! Hopefully, all who read this work will agree that it does justice to real engineering.

Happy Reading!

Summary of the Thesis

Systems Integration is widely accepted as the basis for improving the efficiency and performance of many engineering products. The aim is to build a unified optimised system *not* a collection of subsystems that are combined in some *ad hoc* manner. This moves traditional design boundaries and, in so doing, enables a structured evolution from an integrated system concept to an integrated system product.

It is recognised that the inherent complexity cannot be handled effectively without mathematical modelling. The problem is not so much the large number of components but rather the very large number of functional interfaces that result. The costs involved are high and, if the claims of improved efficiency and performance are to be affordable (or even achievable), predictive modelling and analysis will play a major role in reducing risk.

A modelling framework is required which can support integrated system development from concept through to certification. This means building a 'system' inside a computer and demonstrating the feasibility of an entire development cycle. The objective is to provide complete coverage of system functionality so as to gain confidence in the design before becoming locked into a full development programme with associated capital investment and contractual arrangements.

With these points in mind the purpose of this thesis is threefold. First, to demonstrate the application of bond graphs as a unified modelling framework for aerospace systems. Second, to review the main principles involved with the modelling of engineering systems and to justify the selection of the bond graph notation as a suitable means of representing the power flow (*i.e.* the dynamics) of physical systems. Third, to present an exposition of the bond graph method and to evolve it into a versatile notation for integrated systems.

The originality of the work is based on the recognition that systems integration is a relatively new field of interest without a mature body of academic literature or reported research. Apparently, there is no open literature on the modelling of complete air vehicles *plus* their embedded vehicle systems which deals with issues of integrated dynamics and control. To this end, bond graph concepts need to be developed and extended in new direction in order to facilitate an intuitive approach to the modelling of integrated systems. It is believed that the thesis represents the first attempt to use bond graphs to model a complete integrated suite of aircraft systems.

Given the challenges that are recognised in connection with complex systems integration problems on future aircraft, this thesis sets out to challenge orthodox approaches to modelling. Dependency on mathematical modelling is certain to increase rapidly and, while there are no new theoretical issues being raised here, there are major issues of usability and reusability which directly impact on the ability of engineers to express complicated design ideas in a clear, concise manner. Bond graphs are a convenient and highly appropriate means of demonstrating this principle. Although this thesis happens to deal with aerospace systems and bond graph models of them, the underlying principles are wholly generic and could offer a great deal of insight in other types of systems integration.

Acknowledgements

British Aerospace for supporting and funding this work.

John O'Reilly of Glasgow University, as a patient, ever-helpful and non-intrusive supervisor.

Peter Gawthrop of Glasgow University, as an enthusiatic exponent of bond graph modelling.

Brian Weller of British Aerospace, as a walking encyclopedia on flight control and related systems (not to mention the Concorde air intake control system) and as an unselfish source of technical advice.

Colin Hooper of British Aerospace, for taking a chance on allowing bond graphs into a live project as a method for modelling systems.

Norman Maccallum of Glasgow University, for his generous help in explaining how gas turbine engines really work.

My many colleagues over the years, for dragging me up quite a few learning curves.

My wife Deirdre, for putting up with this seemingly interminable PhD.

My young children Ciarán, Siobhán and Caítlin, for their many generous offers of help.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Systems Integration is widely accepted as the basis for improving the efficiency and performance of many engineering products. The aim is to build an unified system which optimises the use of its subsystem components: it is *not* to build subsystems which satisfy local objectives and then attempt to combine them in some *ad hoc* manner. This moves the philosophy of engineering away from traditional design boundaries and, in so doing, enables a structured evolution from an integrated system concept to an integrated system product.

What is becoming abundantly clear in the aerospace industry is that the complexity inherent in an integrated system solution cannot be handled effectively without a comprehensive foundation of mathematical modelling. The problem is not so much the large number of components inside a big system but rather the very large number of functional interfaces which result. This is a combinatorial problem which impinges heavily on issues of performance prediction, failure analysis and safety assessment. The reason that the problem is taken so seriously is that, as system complexity increases, substantial design effort is needed in order to handle the effects of interaction within the system. The costs involved are high and, if the claims of improved efficiency and performance are to be affordable (or even achievable), predictive modelling and analysis will play a major role in reducing risk.

Typically, it is assumed that systems will be large-scale, highly connected and functionally diverse. They will be constrained by competing sets of requirements and the limitations of different technologies. They will possess several architecture definitions (*e.g.* functional, hardware/software, physical), will span a range of energy domains and will spawn challenging design problems in areas such as control law design, configuration design, human-machine interaction, installation and maintenance. This is a *general* concern for so-called 'critical' systems, where failures could have severe and unacceptable consequences: this is of *particular* concern for safety-critical systems, where failures could result in damage, illness or injury (perhaps even fatality).

Thus, a modelling framework is required which can support integrated system development from concept through to certification. Given the size and complexity of new system concepts, this means building a 'system' inside a computer and demonstrating the feasibility of an entire development cycle as part of the concept definition phase. The term '*Virtual Rig*' has been coined as a software alternative to traditional mock-ups and test specimens. The objective is to provide complete coverage of system functionality so as to gain confidence in the design before becoming locked into a full development programme with associated capital investment and contractual arrangements with the supplier chain.

This defines the need (or the metaphorical 'stick'). There is also an opportunity (a 'carrot') in that the use of modelling, within a properly constituted and accepted process, can replace activities that would otherwise have been performed on hardware rigs. Probably, a large number of tests can be reduced in scope or duration or both. In certain instances, a system rig might even be dispensed with altogether. It is clear that the judicious use of modelling could have a profound impact on the cost structure of a development project, by committing more resources to predictive assessment and less resources to expensive fixed-base facilities (such as an 'iron bird'). The validity of this approach is based on the correctness of models, the correct production and interpretation of analytical results and the correct decision-making process in order to focus the available development effort.

1.2 Purpose

The purpose which this thesis is intended to fulfill is threefold. First and foremost, it is intended to demonstrate the application of bond graphs as a unified modelling framework for aerospace systems. The focus of attention will be placed on vehicle systems integration (covering the main aspects of power provision/consumption and associated aspects of thermal energy management) and there will be some discussion of overall aircraft dynamics and the effect on fuel distribution.

Second, it is considered important to review the main principles involved with the modelling of engineering systems and the specific issues which arise in the aerospace context. This justifies the selection of bond graph notation as a suitable means of representing the power flow behaviour (*i.e.* the dynamics) of physical systems. It also raises many issues related to the purpose and content of models which, although peripheral to the treatment of bond graphs *per se*, offer an insight into the problems of complex systems which ordinarily would not be discussed in literature on mathematical modelling.

Third and final, there is a need to present an expository description of the bond graph method. This is not merely a regurgitation of existing text-book material but, rather, a working definition of the concepts and notations which will underpin the model definitions presented in this thesis. This includes major modifications which have been found to be necessary in order to represent complex systems in a compact manner without sacrificing the intuitive significance of bond graph models.

1.3 Originality

As for the contribution which this thesis could make to the wider study of systems engineering, it is intended that this will offer a framework within which to explore the manifold complexities and trade-offs associated with the next generation of aerospace systems. There are three claims which justify this view and these are summarised below.

Systems integration is a relatively new field of interest without a mature body of academic literature or reported research, especially in the sense defined in this thesis. The International Council on Systems Engineering (INCOSE) is currently sponsoring many initiatives, such as standards and conferences, but it is generally accepted that these are seminal initiatives.

Apparently, there is no open literature on the modelling of complete air vehicles *plus* their embedded vehicle systems which deals with issues of integrated dynamics and control. There is a lot of publication lavished on flight control problems, as a means of illustrating sophisticated control theories, and there is some interest in integrated flight/propulsion control. An extensive literature search failed to identify anything which covered the range of work contained in this thesis or even limited excursions away from established topics.

This thesis is believed to be the first attempt to establish bond graphs as a method for modelling complex aerospace systems. This suggests that the modelling method may have to evolve significantly in order to support human endeavours to model such systems .... and so it is concluded in Chapter 6. It also suggests that the conventional notions of modelling and simulation (based predominantly on signal flow block diagrams) may not be the most appropriate or effective approach to complex systems .... and so it is concluded as well.

These three claims combine to form an overall claim to originality and relevance to engineering research and practice alike. Although this thesis happens to deal with aerospace systems and bond graph models of them, the underlying principles are wholly generic and could offer a great deal of insight in other types of systems integration.

1.4 Layout of the Thesis

The thesis is organised in six chapters, tracing a progressive path from a basic motivation through to an actual trial implementation. The first two intermediate steps cover familiar ground insofar as they deal with the role of modelling and the basic principles of bond graphs. However, they offer some new perspectives on issues that are usually on the periphery of the subject (*e.g.* system architectures and development processes) and also introduce extensions to the bond graph notation. The remaining two steps cover the specifics of aerospace system modelling, firstly at subsystem level and then at the 'virtual aircraft' level. An overview of each chapter is given in the following paragraphs.

Chapter 1 (this chapter) lays down the motivation behind the thesis, drawing on practical experience of the day-to-day issues of integrated system development within the aerospace industry. It states the purpose of the work that is to follow and anticipates the key points to be discussed in detail in each chapter.

Chapter 2 summarises the role of modelling in aerospace systems. It introduces some general concepts and terminology associated with systems and analysis of systems. Specifically, it attempts to formalise the concept of system structure in a way which is most relevant to the modelling of system dynamics. There is a brief discussion of the main properties of a mathematical model, followed by a number of context-specific perspectives on how models can contribute to various aspects of system development. Finally, an overall perspective is presented which draws together the various strands of discussion into a single problem statement and then introduces the bond graph method as a suitable candidate solution.

Chapter 3 presents a summary of the bond graph method. Basic ideas and concepts are defined and illustrated with respect to a number of pertinent examples. A concise statement is given of the conventions to be applied to the models in the following two chapters, together with an indication of the sort of information model which would be required to support a neutral data exchange format for bond graphs. This also enriches the standard bond graph method by adding new features and notational refinements in order that it can be more readily applied to the modelling of complex systems.

Chapter 4 deals with the definition of five bond graph libraries for component modelling, namely **Generic**, **Hydraulic**, **Thermofluid**, **Electrical** and **Flight Dynamic**. The first of these considers a small number of generic functional mechanisms; the remainder cover basic physical principles and their mathematical expression, with more detail being furnished as appropriate for important component groups. The first aim is to provide an overview of all the main characteristics which apply to vehicle systems, across all relevant energy domains; such an overview gives a useful exposition although (surprisingly) no aeronautical text could be found in the literature search which collates this information. The second aim is to introduce the new bond graph features in order to standardise on a notation which optimises the use of graphical objects for model representations.

Chapter 5 presents a total system model in the form of a so-called *virtual aircraft*. A generic framework is defined for air vehicle integration within which to construct functional models of complete aircraft. A hypothetical aircraft is described and a model defined for vehicle systems integration (*i.e.* Electrics, Hydraulics, Actuation, Propulsion, Fuel and Environment). This achieves the primary purpose of this thesis, namely to demonstrate the application of bond graphs as a unified modelling framework for aerospace systems. Note that this is quite distinct from analysis and simulation, which are not considered in this work.

Chapter 6 presents the conclusions of the research and makes recommendations for future work in order to refine the bond graph method further, to develop its interfaces with other object-oriented methods and to provide a basis for parametric nonlinear analysis of complex systems (especially control systems).

1.5 Philosophy

The reason for pursuing the goal of a unified modelling framework is to produce a virtual rig, as mentioned already. This enables a full airborne system to be built and operated inside a computer and, progressively, to be replaced by real hardware and software. This implies that model objects must be analogous to physical objects in terms of their form, fit and function. Interfacing must be 'carefree' in the sense that data flow and energy flow paths are fully compatible and that measurements can be recorded at any point in the model. The rig must host a common set of models and provide facilities for data logging, fault injection, experimental testing, static and dynamic analysis, visual animation and concurrent simulation.

Given the composition of integrated systems, it is no longer possible for an engineer to be an expert in all relevant disciplines. Therefore, it is desirable that system models involve a minimum amount of mathematical manipulation and encourage the use of physical analogy, *i.e.* conveying the architecture of an engineering system in an intuitive and obvious way with reference to physical laws of cause and effect. It should be recognised that the priority is to render a conceptual model of the whole integrated system, not necessarily to model each and every component in fine detail. To be useful, a model must allow both progressive and selective substitution of component models with equivalent models of different resolution (either greater or lesser), as necessary to address specific aspects of system behaviour, while maintaining the correct representation of system structure.

From both of these observations, it is apparent that graphical methods for building models are highly valuable. The specific choice of the bond graph method is attractive because it is perhaps uniquely suited to handle the dynamic characteristics of physical networks at a conceptual level. A lot of practical experience has been gained during the course of this research project. What has been most surprising has been the ease with which engineers and modellers alike have been able to acquire a working knowledge of the bond graph philosophy and to focus quickly on the real issues of modelling physical systems. This has been especially true in connection with thermofluid modelling (a notoriously problematic area) and some effort has been devoted in chapter 4 to this, hopefully doing justice to the numerous issues that arise.

1.6 Overall Summary

Given the challenges that are recognised in connection with complex systems integration problems on future aircraft, this thesis sets out to challenge orthodox approaches to modelling. Dependency on mathematical modelling is certain to increase rapidly and, while there are no new theoretical issues being raised here, there are major issues of usability and reusability which directly impact on the ability of engineers to express complicated design ideas in a clear, concise manner. Bond graphs are a convenient and highly appropriate means of demonstrating the power of a graphical technique for model-building. In the context of the under-developed academic interest in systems integration to-date, the work is intended to advance an original contribution in this area .... one which will serve to stimulate and focus further work.

Chapter 2

The Rôle of Modelling in Aerospace Systems

SUMMARY

Systems integration is driven by the need for close cooperation between systems which otherwise would operate in an autonomous manner. As the overall level of complexity increases, the ability to realise a cost-effective engineering solution becomes critically dependent upon the ability to understand dynamic behaviour. Ultimately, this means building a 'full system' inside a computer and demonstrating its feasibility through mathematical modelling and numerical simulation. This chapter establishes the main ideas behind the construction of a modelling method and identifies the specific rôle which this might play in the development of aerospace systems.

2.1 Introduction

Mathematical modelling is becoming a major issue in the development and certification of new airborne systems. The main reason is that future air vehicle concepts incorporate a high level of functional integration in order to improve operational performance and efficiency. As a consequence, the control and management of key resources, such as electrical power, fuel and heat, requires an often complex interaction between physical systems and software systems which is difficult to optimise. The big risk is that development costs may spiral because of the large number of components, interfaces, switches and failure modes associated with a full system.

One way to address this problem is to undertake system-level modelling in order to help understand how a system behaves and how it can be controlled. In this vein, the idea of a "virtual aircraft" has become fashionable as a computer-based testbed for a range of experimental work. This builds a 'system of systems', containing diverse technologies and many subsystems and interfaces. The aim is to demonstrate that a complete air vehicle system is fit for purpose, to show how resources are shared and to analyse the interdependencies between subsystems (especially under failure conditions).

In simple terms, the main objectives of system modelling can be summarised as follows:

- **to confirm the concept of system operation**
- **to understand the inherent functional mechanisms**
- **to predict system behaviour**
- **to investigate parametric uncertainty and physical constraints**
- **to exercise control functions**
- **to propose and refine engineering design requirements**
- **to provide evidence in support of a safety case**

This purpose of this chapter is to indicate what is required in order to satisfy these objectives. It establishes the main ideas behind the construction of a modelling process and identifies the specific rôle that this might play in the development of aerospace systems.

2.2 What is a System?

The concept of a "system" is ubiquitous and, through over-use, its meaning is rather ambiguous. In engineering, it is not sufficient to adopt a minimalist description [*cf.* Bennett 1995, p.1] which talks about a system as a collection of objects (or some other named items) which "interact with each other, within some notional boundary, to produce a particular pattern of behaviour". Systems are designed to serve a purpose; because they contain many component parts, then they serve a *unified* purpose. Also, the embodiment of a system is always a compromise between cost, performance and risk. These issues determine what system is actually produced, how it is structured and how it behaves. No system exists in isolation from its requirements and constraints, a fact which is crucial in building effective models.

In order to avoid confusion, a number of general terms which are relevant to this discussion are defined in Table 2.1. Note that there is no universally accepted 'systems engineering dictionary' although there is a high degree of commonality between various internationally adopted standards. For illustration, a typical set of system concepts and their relationships are shown in Figure 2.1. These cover a wide range of operational considerations which lead to system failure and the occurrence of accidents. The underlying formal definitions are arranged in Figure 2.2, drawing on a UK Defence Standard [DEF STAN 00-56 (Part 2)/2] and two documents relating to safety and certification issued by professional committees of the Society of Automotive Engineers in the USA [namely ARP4754 and ARP4761]. Essentially there is little new in the formal definitions [*cf.* BSI Handbook 22:1983 on Quality assurance] but it is significant that they are being reinterpreted in the context of integrated systems or, generically, so-called complex systems.

System	A set of functions with a coherent operational purpose
Subsystem	Any part of a system which constitutes a system in its own right
Component	A building block for a system at the 'lowest' level of resolution
Environment	An external system or domain that interacts with a particular system of interest
Control System	A system which changes the dynamic behaviour of a system of interest
Critical System	A system for which failure would have unacceptable impact on system services
Integrated System	A system of systems with shared resources, with each constituent system having a delegated role and providing its own services
Real-time System	A system with timing requirements on delivery of its services
Attribute	Any formally recorded property of a system
Duty Cycle	The way in which a system is exercised over time, indicating the level of stress with respect to its maximum performance

Table 2.1 General Terminology

It is important to consider system safety because, in many respects, one learns more about a system by understanding the ways in which it can go wrong than by analysing its nominal or intended behaviours. As a most basic principle, a system is "safe" if it is free of any inherent characteristic or environmental factor which could result in an *accident*. An accident is an event which causes actual harm or damage; the link between an event and some potential harm or damage is identified as a hazard. Typically hazards include fires, high-energy uncontained explosion and escape of gases/liquids (especially if hot and/or pressurised). Another hazards might be inadequate protection against environmental threats and human exposure to high voltage or corrosive/toxic material. For practical purposes, hazards can be grouped as those that are known from experience or basic principles, those that are *identified* by analysis and those that are *"discovered"* during development or operational use.

Safety is distinct from other requirements as it is non-deterministic, it has no absolute measure and it is based on the non-occurrence of particular behaviour. This is one of the key points of DEF STAN 00-56 (Part 1)/2, defining safety management requirements for systems[1]. It relies on a human perception of tolerable risk in relation to an operational need or benefit. Lack of safety can only be assessed by recording accidents or incidents but, at this stage, system design is mature and a cost-effective remedy is usually not practicable. The problem with ensuring non-occurrence of a harmful event is that few systems exist in large enough quantity (without upgrades or corrections) for a quantitative "safety test" to have any statistical significance. The problem with human perception is that even one or two incidents can undermine confidence in system safety.

Another consideration which extends the basic concept of a "system" is the analysis that needs to be performed for a whole variety of purposes during the development cycle from requirements through design to qualification and thence entry into service. This is important because it determines how a system is going to be measured against the claims made for it at progressive levels of maturity.

[1] DEF STAN 00-56 deals specifically with defence systems containing programmable electronics but the basic philosophy is widely applicable.

Notwithstanding the variation in project-specific nomenclature and in engineering practice, there are three main categories of assessment, namely

- **System Functional and Performance Assessment**
- **System Safety Assessment**
- **Integrated Product Safety Assessment**

As the names suggest, these deal with distinct aspects of system design although they should not be considered a separate activities. The philosophy of integrated system design depends on the ability to perform analyses in parallel, from early in the design cycle, such that design teams can work concurrently and each team understands the context of other teams and the constraints under they must operate.

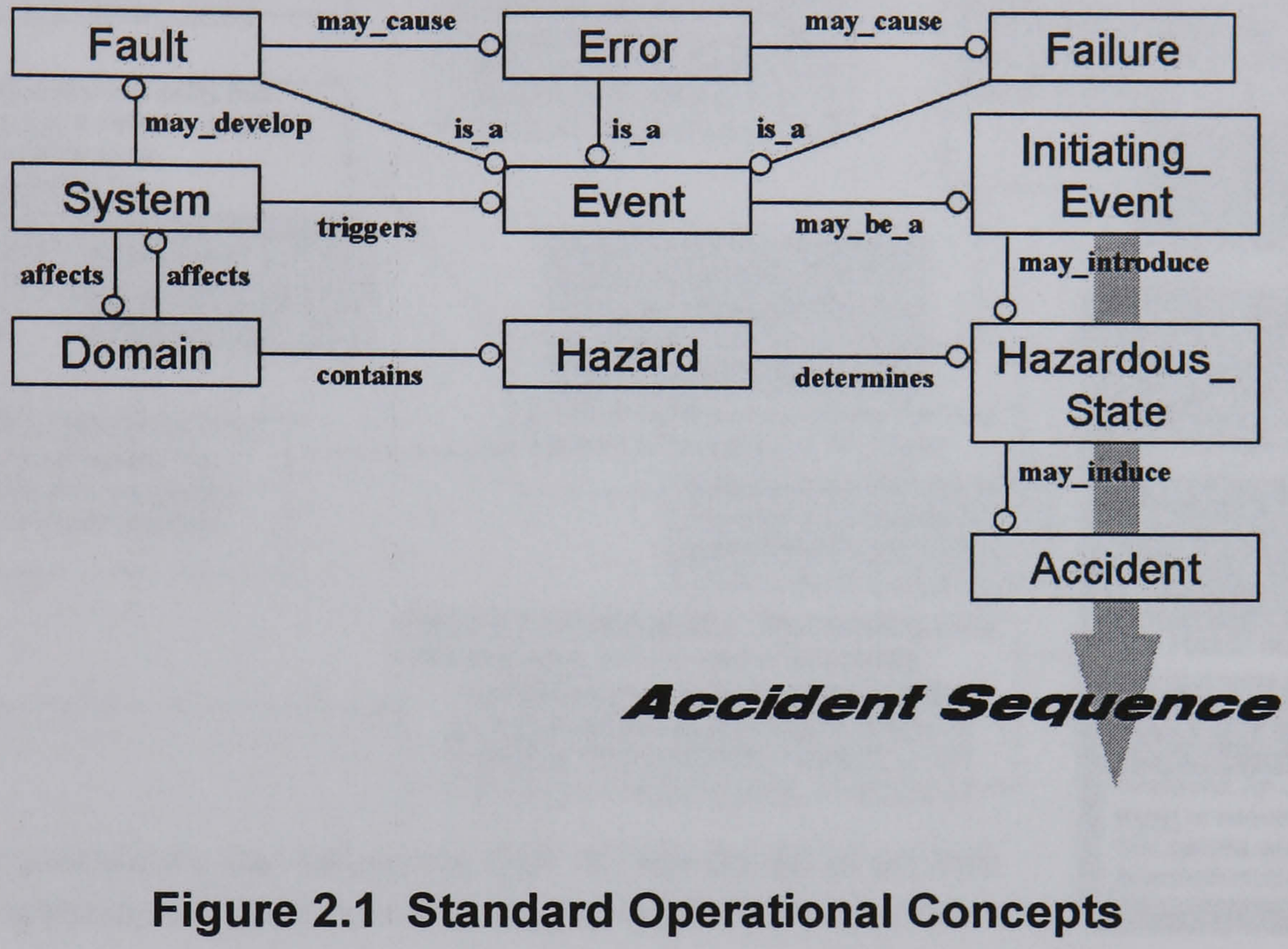


Figure 2.1 Standard Operational Concepts

2.2.1 System Functional and Performance Assessment

The main analytical procedures for system functionality and performance are aimed at confirming that the design is fit for purpose and that the characteristics and behaviour of a system are within its specified tolerances. Typically, they include the following activities:

- **Steady-State Performance Analysis (SSPA)**
  to determine the trade-off between system design parameters and to establish equilibrium conditions
- **Open-Loop Dynamic Analysis (ODA)**
  to determine the stability and performance characteristics of a system without feedback control
- **Closed-Loop Dynamic Analysis (CDA)**
  to determine the stability and performance characteristics of a system with feedback control
- **Moding and Logic Analysis (MLA)**
  to establish the effects of system configuration, sequencing of stimuli and handling of events
- **Failure Performance Analysis (FPA)**
  to identify the effect of failures on the stability and performance of a system
- **Human Engineering Assessment (HEA)**
  to evaluate the acceptability and efficiency of the interface between the system and its human users

The first of these activities addresses much of the traditional analytical approach to engineering design, from components through to entire systems, such as aircraft and engines. The style of presentation is typically performance tables/graphs which trade off key parameters. These contain large amounts of empirical data, summarised in a convenient form so that optimum design points can be identified and that off-design performance can be estimated.

Open-loop and closed-loop dynamic analysis are mainly the province of control engineers and deals with the transient performance of systems and the interaction with their environment. Such analyses are crucial if system behaviours are to be regulated in a satisfactory manner. This involves the determination of stability and performance characteristics which are inherent in the underlying physics and which are emergent as a result of feedback control action. Note that, under feedback control, these characteristics are artificially generated by the addition and modulation of external power.

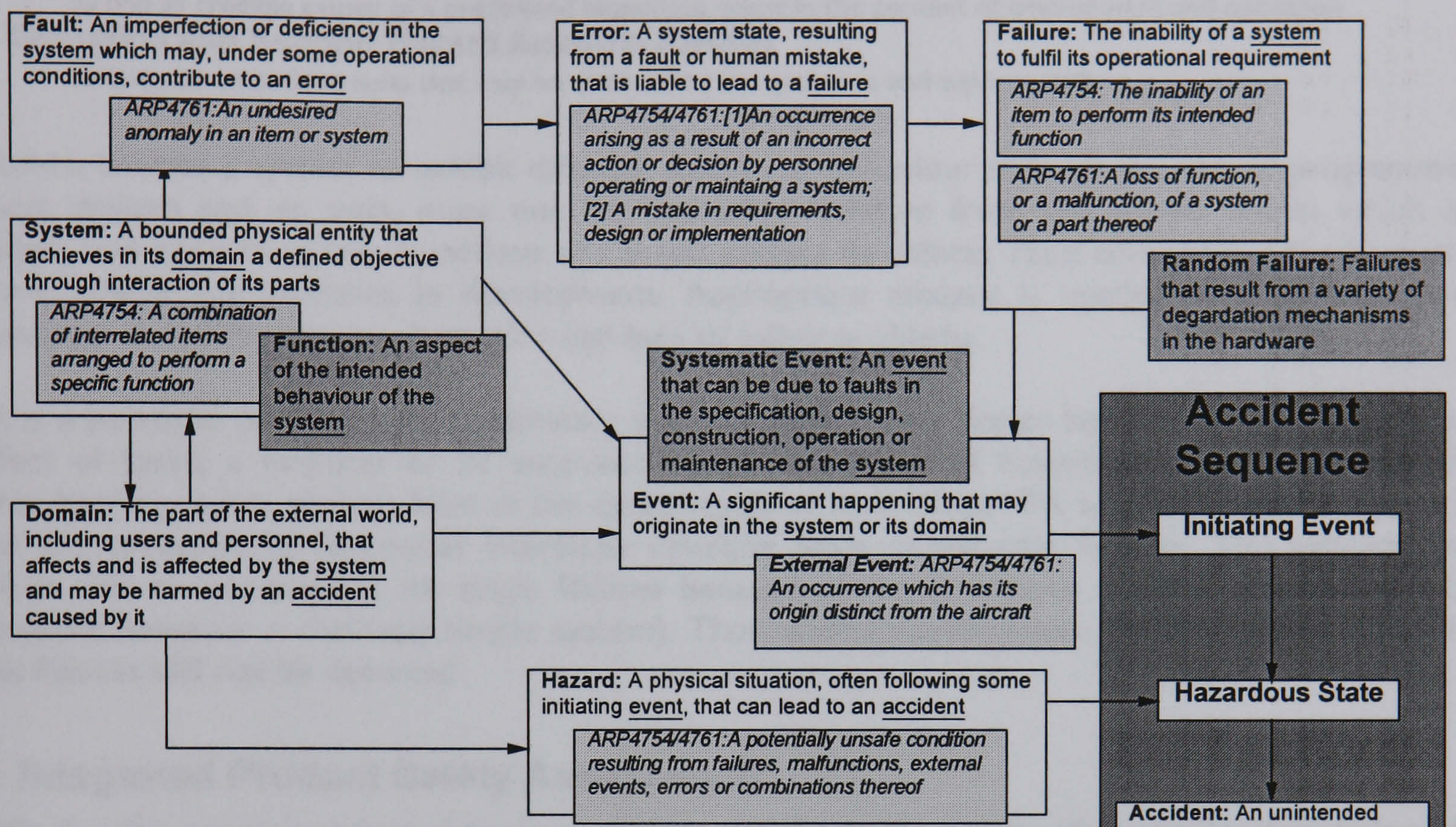


Figure 2.2 Definition of Terms

Operational moding and logic, together with the assessment of failure modes, are major activities associated with the realisation of a system architecture for the eventual system product. Moding deals with changing the manner in which the system is to be used whereas logic design is intended to handle combinations of measurable conditions which have to be true in order for something to happen or to be enabled. Failure modes exist in all systems and the performance implications must be analysed so that critical failure effects can be identified.

Human engineering is a key aspect of system design, both for operability and maintainability. The efficiency of the human-system interface will have a major impact on the effectiveness of the system product. It has long been recognised that, for vehicle design specifically, handling qualities are an essential part of the engineering specification. Poor handling leads to high workload, over-sensitivity to disturbances and even perhaps loss of control (as evidenced in pilot-induced oscillation, or PIO, in aircraft).

2.2.2 System Safety Assessment

For system safety assessment, the focus of activity is understanding the mechanisms whereby a system can fail or misbehave and potentially cause (unwanted) loss, damage or injury. Analysis usually includes:

- **Functional Failure Analysis (FFA)**
  to assess the effect of loss of function, provision of incorrect function or provision of correct when not required
- **Failure Modes, Effects and Criticality Analysis (FMECA)**[2]
  to assess the cause and effect of all possible failure modes, together with the probability and severity of occurrence
- **Fault Tree Analysis (FTA)**[3,4]
  to find all credible causes of a predefined hazardous event in the context of environment and operation
- **Operation and Support Hazard Analysis (OHSA)**
  to evaluate hazardous tasks that may be undertaken by operation and support staff

Software enables a system to exhibit different functional behaviour through the use of programmable electronic devices and, as such, must not be treated in isolation from the system within which it is embedded. It is not intrinsically hazardous and is not subject to defects. Fault conditions arise because of hardware defects and mistakes in development. Appropriate analysis is needed in order to identify circumstances in which software execution can lead to safety problems.

FFA is a powerful technique for preliminary assessment of a new design because it deals directly with the effect of losing a function or of encountering various types of functional misbehaviour. FMECA supports rigorous safety analysis later in the design cycle as it develops FFA to component level and can indicate the presence of functional interfaces resulting from component failures. The problem with FMECA is that it concentrates on single failures because of the enormous number of possible failure combinations (even for a relatively simple system). Thus, without automation, most hazards produced by multiple failures will not be detected.

2.2.3 Integrated Product Safety Assessment

Finally, for the system product, there is a need to coordinate the results of system-level analysis and to consider factors in the physical architecture and systems integration which could contribute to safety problems. This could include the following activities, depending on the maturity of design:

- **Zonal Hazard Analysis (ZHA)**
  to establish the effects of failures in adjacent locations, design installation and operation/maintenance
- **Energy Trace Analysis (ETA)**
  to find conditions in which unwanted energy is transferred to vulnerable targets in the absence of adequate barriers
- **Human Engineering Safety Assessment (HESA)**
  to integrate the operational aspects of system-level OHSA and assess overall operability of the integrated system
- **Maintenance Human Interface Assessment (MHIA)**
  to integrate the support aspects of system-level OHSA and assess overall supportability of the integrated system

The first two analyses (ZHA and ETA) are especially important to the intrinsic safety of the system because they cover physical interaction between system constituents. This class of analysis is usually not feasible until later stages of the development cycle, when the detailed aspects of physical assembly have been determined. However, for integrated systems, there is a case in favour of predictive modelling at the system concept level in order to identify problems earlier. This might have a big impact on the overall design given the potential vulnerability of shared functions and potential difficulties associated with re-configuring or re-routing elements of the physical architecture.

---

[2] MIL-STD-1629A

[3] Note that FTA is not a quantitative model, nor is it a model of all possible causes of system failure. The technique can be used to construct an event tree of all credible consequences of a predefined hazardous event.

[4] NUREG 0492

2.3 Motivation for System Modelling

Computer-based models are intended to be 'simple' representations of 'complicated' things. They are created for the purpose of assessing some aspect of a system of interest, as a substitute for a real object, *e.g.* an engineering system. Because it is always important to gain confidence that requirements can be achieved and that an emerging design is acceptable, models can play an invaluable role from early in the development cycle. Even a very crude concept model can enable design decisions to be made and risks to be identified.

Taking the forms of system assessment discussed so far, it is important to appreciate the modelling issues that arise. This will help focus on the key requirements for a unified approach to builds models of aerospace systems (concentrating on vehicle systems integration, as stated in Section 1.2). It is apparent that there are (at least) six issues, namely:

- **Function**, *i.e.* what a system does and how it behaves
- **Signal**, *i.e.* information passing around a system
- **Event**, *i.e.* changes in the operation of a system
- **Cause**, *i.e.* what produces those changes
- **Geometry**, *i.e.* the physical shape of a system
- **Human**, *i.e.* how human beings interact with a system

Figure 2.2 attempts to show the dependencies that these issues introduce into system assessment.

Assessment	Modelling Issue					
	Function	Signal	Event	Cause	Geometry	Human
SSPA	●					
ODA	●	●				
CDA	●	●				
MLA	●		●			
FPA	●					
HEA			●		●	●
FFA	●	●	●			
FMECA			●	●		
FTA				●		
OHSA			●	●	●	●
ZHA	●				●	
ETA	●			●	●	
HESA			●	●	●	●
MHIA			●		●	●

Table 2.2 Modelling Issues in System Assessment

Although one could debate the correctness of a dependency matrix of this type because it is not produced by exhaustive analysis: it serves purely as an illustration. However, it can be claimed with some justification that this particular matrix adequately summarises the main issues that each form of assessment must address. There may be others and there may be overlapping issues but that is of minor consequence here.

What is striking (but no surprising) is that virtually the complete set of assessments involves Functions, Signals and Events. These are different aspects of what a system does and show the observed effects or consequences of how it is being used. Causal factors constitute a separate issue and their analysis usually takes the form of 'what if' scenarios, starting with overall system events or local component events and then postulating all possible and credible consequential effects. A system ultimately will ultimately reside in the real world and therefore it will require a physical reference model, detailing its geometry, materials, assembly and so on. Finally, there are human factors, which take into account psychological issues (perception, cognition and so on) and physical processes (geometry, mechanics, environment and so on).

For integrated systems, because of their size and complexity, all of these issues demand a conceptual framework based on models. There will be a family of models, each considering a specific aspect of system design or operation, each cutting away irrelevant detail in order to reveal a particular technical principle in a more or less abstract way. For integrated vehicle systems, it is clear that the provision, distribution and consumption of power are the key priorities and therefore the key requirements of any unified modelling approach. This implies a comprehensive treatment of dynamics, based on a functional understanding of physical processes. To this end, system geometry can be treated simply as data.

Once functional models are constructed, they can be used to verify causal factors underlying observed behaviour. They can also be used to support human engineering investigation, generating results and metrics as appropriate. The core requirement is to model Function and how that is affected by Signal processing and Event triggers.

2.4 System Structure

In order to organise the properties of a system which are most relevant to modelling, it is convenient to define three characteristics, based on architectural composition, events/transitions and dynamic indicators. The overall concept of 'structure' represents the content of a system, the relationships between its constituent parts and the factors that determine its behaviour.

2.4.1 Composition

The composition of a system can be defined as a layout, consisting of *architecture* objects, which define the internal arrangement, and *component* objects, each of which defines a relationship between parameters. There are numerous techniques for object analysis and design which can be applied across many domains and it is not necessary to review those here. It is sufficient to note that dynamic systems, in their operation, must satisfy a top-level specification composed of discrete classes (abstract objects) and, in their physical realisation, are composed of hardware items (physical objects). This can be mapped to any level between these extremes in order to reflect engineering definition, functional allocation, technology selection and so on. This leads to partitioning of object structures into substructures, each fitting into an architectural hierarchy of the type shown in the Table 2.3.

The architectural hierarchy is a data repository for complete system representations. It is not intended to imply any particular lifecycle model or any particular method of system decomposition: it could be related to any one of many lifecycle models [*e.g.* EIA 632] if required. However, it is probably more useful to consider this type of hierarchy as a possible framework for bridging the gap between system functionality and design description which is recognised in the emerging area of Multi-disciplinary Design and Optimisation (MDO) [Laan *et al.* 1997]. This is "a methodology for the design of systems where the interaction between several disciplines must be considered, and where the designer is free to significantly affect the system performance in more than one discipline" [Sobieszczanski-Sobieski & Haftka 1994].

Level 0	**Operational Interface** Relationships between the system and its environment	**Human/Function Interface** Information exchange between The system and its users
Level 1	**Functional Architecture** Associative relationship between functions, which define what the system is intended to do and *not* how it is to be implemented	
Level 2	**Essential Architecture** Template for system implementation based on assumptions regarding hardware and software implementation	
Level 3	**Hardware Architecture** Arrangement of hardware for transporting matter, energy and information	**Software Architecture** Arrangement of software for acquiring, manipulating and distributing information
Level 4	**Physical Architecture** Embodiment in terms of hardware packaging, housing, installation, protection and compatibility	**Software Mapping** Assignment of software on to computing resources

Table 2.3 System Architectural Hierarchy

The basic dichotomy is that system functionality involves a mixture of *intuitive* and *mathematical* measures of optimality whereas the design description involves a mixture of *system concept* and *product assembly* information. The two halves of the picture are not the same. The refinement of a system results in an increase in the level of detailed information, which is usually depicted as a pyramid (with the growth of information indicated by the increase in area from top to bottom). The MDO approach is based on the iteration of four steps, namely Problem Definition, System Analysis, Sensitivity Analysis and Optimisation. These are arranged, as shown in figure 2.3, in such a way as to develop system functionality in conjunction with an impact assessment on product engineering.

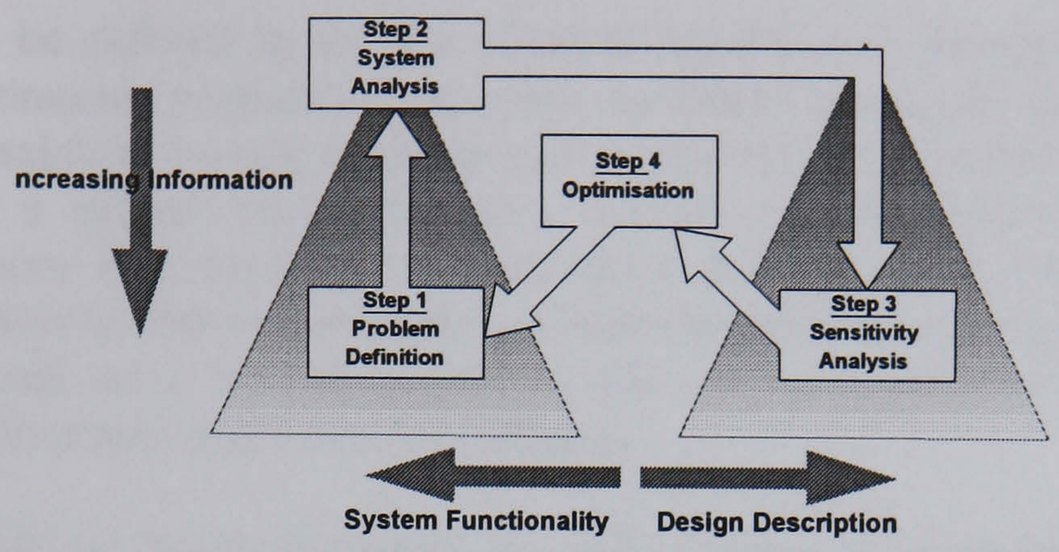


Figure 2.3 Steps in the MDO Approach

The problem recognised in MDO can be seen more generally between all levels in the system hierarchy defined in Table 2.2. In some respects, the problem can be even worse because, at any given time, the constituents of a system design might be distributed across all levels of the hierarchy. Operational requirements are applied at *Level 0* and propagate downwards; implementation constraints, ultimately, exist at *Level 4* and propagate upwards. Customer requirements and other external requirements (*e.g.* quality, safety, environmental, legislative) can be applied at any level.

At whatever level, engineering development will increase the amount of design information and, while the focus may be predominantly within the current level, the impact on subsequent design cannot be ignored. This will involve decomposing and refining requirements, deriving new requirements and introducing lower-level assumptions (*i.e.* predictions of what constraints will apply). The transition from one level to the next involves a mapping of the system structure into a form appropriate for the next stage of development; it will probably involve some re-structuring consistent with a different viewpoint [contrast functional design with electrical circuit design, for instance]. The aim is that most of the validation will be non-regressive such that there is little need to refer back to requirements or assumption, other than the immediate level above. In this way, the steps shown in figure 2.3 could be applied between any two levels in the hierarchy.

2.4.2 Events

Events can be defined as the collection of discrete transitions that can occur in a system or in the stimuli that drive the system, together with the conditions that cause those transitions to occur. This is expressed as a set of event triggers relating to mission segments, operational scenarios, threat scenarios and availability models.

Discontinuity is characterised by events that occur within a system and between a system and its environment. These can have structural and parametric effects that propagate throughout a system. Typical events include faults, failures, mode changes, signal logic, hardware limits and environmental limits. The system specification needs to reference an event table, containing a formal description of enabling conditions, state transitions, timing properties, system effects and so on. In the design of a control system, particular attention must be paid to the design of an event handler because of possible requirements on deadlines for identification and reaction, on prediction of critical events and on synchronisation or sequencing of system-wide responses.

There is a very large amount of literature which covers discrete event systems and a range of modelling techniques and simulation tools that can support engineering design. Although this is not relevant to this thesis it is interesting to note the growing interest in so-called *hybrid* systems modelling and *hybrid* languages such as Modelica [Elmqvist *et al.* 1997, van Broenink 1997] and Chi [van Beek & Rooda 1997]. Petri nets are also increasingly popular because of their capacity to visualise concurrent processes. A good survey paper is available in the literature [David & Alla 1994] as well as standard texts on theory [Peterson 1981, Reutenauer 1990] and applications [Genrich & Lautenbach 1983].

2.4.3 Dynamics

Dynamic Structure can be defined to be the physical mechanisms through which energy stores can interact. This drives a continuous, periodic or sporadic evolution of system parameters and gives rise to indicators of interaction, stability, inverse stability and performance. Dynamic behaviour is the result of transferring power within a system and between a system and its environment. The assessment is dominated by control theory and, as such, is primarily concerned with linearised representations of physical systems. It is possible to reference nonlinear representations in the generation of both time and frequency response data and, to a limited extent, in the use of predictive techniques like 'describing function' analysis, variable structure and Lyapunov theory.

Practical specification will lay heavy emphasis on time domain characteristics and on the frequency domain concepts of bandwidth, phase/gain margins and so on. Typically this might include rise time, settling time, overshoot, steady-state error, phase/gain margins, disturbance rejection and maximum cross-channel interaction. The list of data requirements could be extensive for an integrated system and will depend on the particular design process to be employed. Note that the acceptability of any *figure of merit* will be dictated ultimately by safety and clearance considerations (even to the extent that a particular figure is deemed to have no direct bearing on safety or clearance).

In general, any engineering system can be described by *nonlinear time-varying* equations of the form:

$$\dot{\mathbf{x}} = \mathcal{F}(\mathbf{x}(t), \mathbf{u}(t), t)$$
$$\mathbf{y} = \mathcal{G}(\mathbf{x}(t), \mathbf{u}(t), t) \qquad (2.1)$$

where x(t), u(t) and y(t) are the state, input and output vectors of the system, respectively. For an operating condition in which the system is in equilibrium, it is usually possible to approximate the behaviour by a *linear time-invariant* model, expressed in so-called descriptor form [Luenberger 1977]:

$$E\dot{\mathbf{x}} = A\mathbf{x}(t) + B\mathbf{u}(t)$$
$$\mathbf{y} = C\mathbf{x}(t) + D\mathbf{u}(t) \qquad (2.2)$$

The equivalent transfer function matrix is $G(s) = C(sE-A)^{-1}B + D$. This is valid provided the system exhibits only small perturbations about a steady-state operating condition. Note that there is an emerging body of work on velocity-based linearisation which renders models of this form valid at any operating condition [Leith & Leithead, 1998a, 1998b].

In a typical control scheme, output feedback is used to match a set-point and, *via* compensation, satisfy stability and control requirements. Inputs **u** are derived as a function of the error **e** between the reference inputs **r** and the outputs **y**. Outputs are often called *controlled variables* and, depending on whether they number one or many, the resulting controller is designated as single-variable or multivariable. The functional relationships are:

System Dynamics	$\mathbf{y} = G(s)\,\mathbf{u}$	(2.3)
Control Law	$\mathbf{u} = K(s)\,\mathbf{e}$	(2.4)
Output Feedback	$\mathbf{e} = \mathbf{r} - \mathbf{y}$	(2.5)

for a *system* G(s) and a *controller* K(s). The system dynamics are described by composite relationships:

Open-loop Dynamics	$\mathbf{y} = L(s)\,\mathbf{r}$	(2.6)
Closed-loop Dynamics	$\mathbf{y} = [I + L(s)]^{-1}L(s)\,\mathbf{r}$	(2.7)

where L(s)=G(s)K(s) is called the *loop transfer function* matrix and I+L(s) the *return difference* matrix.

Mathematical techniques, such as singular value analysis [Doyle & Stein 1981, Ridgely & Banda 1986, Maciejowski 1989], structured singular value analysis [Maciejowski 1989, Doyle 1982] and interacting subsystem analysis [Schierman & Schmidt 1991, 1992, Schierman *et al.* 1993], have yet to fully evaluated in practical engineering. Notwithstanding the voices of descent [Nesline & Zarchan 1983, Keel & Bhattacharyya 1997], there is still considerable momentum in this area.

A very large theoretical evaluation has been completed recently under the GARTEUR[5] Flight Mechanics project FM(AG08)[6], entitled "Robust Flight Control" [Magni *et al.* 1997]. In contrast, only a small amount of flight testing appears to have been done [Burken 1992] and the results are inconclusive. A similar AIAA challenge [Brumbaugh 1991] gave similarly inconclusive results. Also, exploratory papers have been published on integrated flight/propulsion control (IFPC) [Rock *et al.* 1994] and on partitioning centralised IFPC laws [Schmidt *et al.* 1991, Garg 1993]. These highlight the immaturity of approach and the need to *demonstrate* benefits from new techniques remains paramount [cf. Stewart *et al.* 1992].

For this reason, it is expected that the emphasis will continue to be placed on directly testable and measurable properties. To this end, Individual Channel Analysis and Design (ICAD) is a recent method for multivariable control which follows classical Nyquist-Bode concepts for single-loop control [Bode 1945, Nyquist 1932, O'Reilly & Leithead 1991]. The control problem is recast as a set of individual channels, which preserve the effect of loop interactions and which enable the systematic analysis and design of complex control laws. Thus, for an m-input m-output system G(s) and a diagonal controller K(s), structural equivalence exists between the tracking function matrix $T(s)=[I+L]^{-1}L$ and the closed-loop dynamics of the m individual channels, as shown in figure 2.2. The *open-loop* transmittance of the i^{th} individual channel is

$$c_i(s) = g_{ii}(s)[1 - \gamma_i(s)] \qquad (2.8)$$

where $g_{ii}(s)$ is the direct path and $\gamma_i(s)$ is defined as the multivariable structure function.

Comparisons have been performed [Leithead & O'Reilly 1995, Leithead *et al.* 1997] to show the relationship of ICAD to Quantitative Feedback Theory (QFT) [Horowitz 1979, 1982, Yaniv & Horowitz 1986], Sequential Return Difference [Mayne 1973, 1979], Inverse Nyquist Array [Rosenbrock 1969, 1974] and Relative Gain Array [Bristol 1966, 1967].

The interpretation of poles and zeros is a matter of considerable interest in control system analysis and is discussed at length elsewhere [*e.g.* MacFarlane & Karcanias 1976, Leithead & O'Reilly 1994, 1994, Maciejowski 1989]. The stability of a linear system is ensured by the absence of poles in the right-half *s*-plane (RHP). The absence of RHP zeros conveys 'minimum phase' behaviour, which is better described as controllability (intended as an intuitive concept rather than a strict mathematical definition indicating non-singularity of a particular matrix [Kwakernaak & Sivan 1972, Kailath 1980]).

2.5 System Development Context

In order to start to discuss the development of mathematical models in a more systematic manner, it is useful to review a number of basic concepts which have evolved into a framework for recording items of information and their inter-relationships. In order to reflect the development context, (at least) four complementary projections or views need to be defined and related. These are as follows:

- **Information Model** to identify the basic categories of information to be recorded during development and the relationships between them
- **Process Model** to describe different development activities and their relationships, leading to a definition of development artifacts
- **Documentation Model** which packages the development artifacts into pre-defined formats in order to record the results of development activities and their inter-relationships
- **Enterprise Model** which defines the structure of organisations that participate in development[7], identifying and delineating roles within/between organisations

---

[5] Group for Aeronautics Research and Technology in Europe
[6] http://www.nlr.nl/inc/garteur/rfc.html
[7] *cf.* Strens & Dobson 1994

These views are 'models' in their own right, which cover the human domain and provide a foundation for understanding what is done in order to develop a system. As an observation, traceability is a key concern in complex systems and is one of the main topics in recent requirements engineering research [IEE 1991, Gotel & Finkelstein 1994, Pohl & Jacobs 1994]. It is primarily concerned with the construction of an information model and can be characterised as a set of managed relationships between development artifacts, documentation artifacts and enterprise roles. A summary of a top-level information model is shown in figure 2.4 [*cf.* Ramesh *et al.* 1995] using the concept of entities and relationships [Chen 1977].

2.5.1 Traceability

The ideas involved in the interpretation and traceability of development activities are illustrated in figure 2.5. Note that this highlights the distinction between development activities and traceability activities. A fundamental issue is to find an effective methods of recording information that emerges during development in a set of traceable structures. This has been most extensively examined in schemes for capturing design rationale, *e.g.* Questions, Options and Criteria [MacLean *et al.* 1991], functional representation [Chandraeskaran 1993] and the Design Rationale Capture System (DRCS) [Klein 1993].

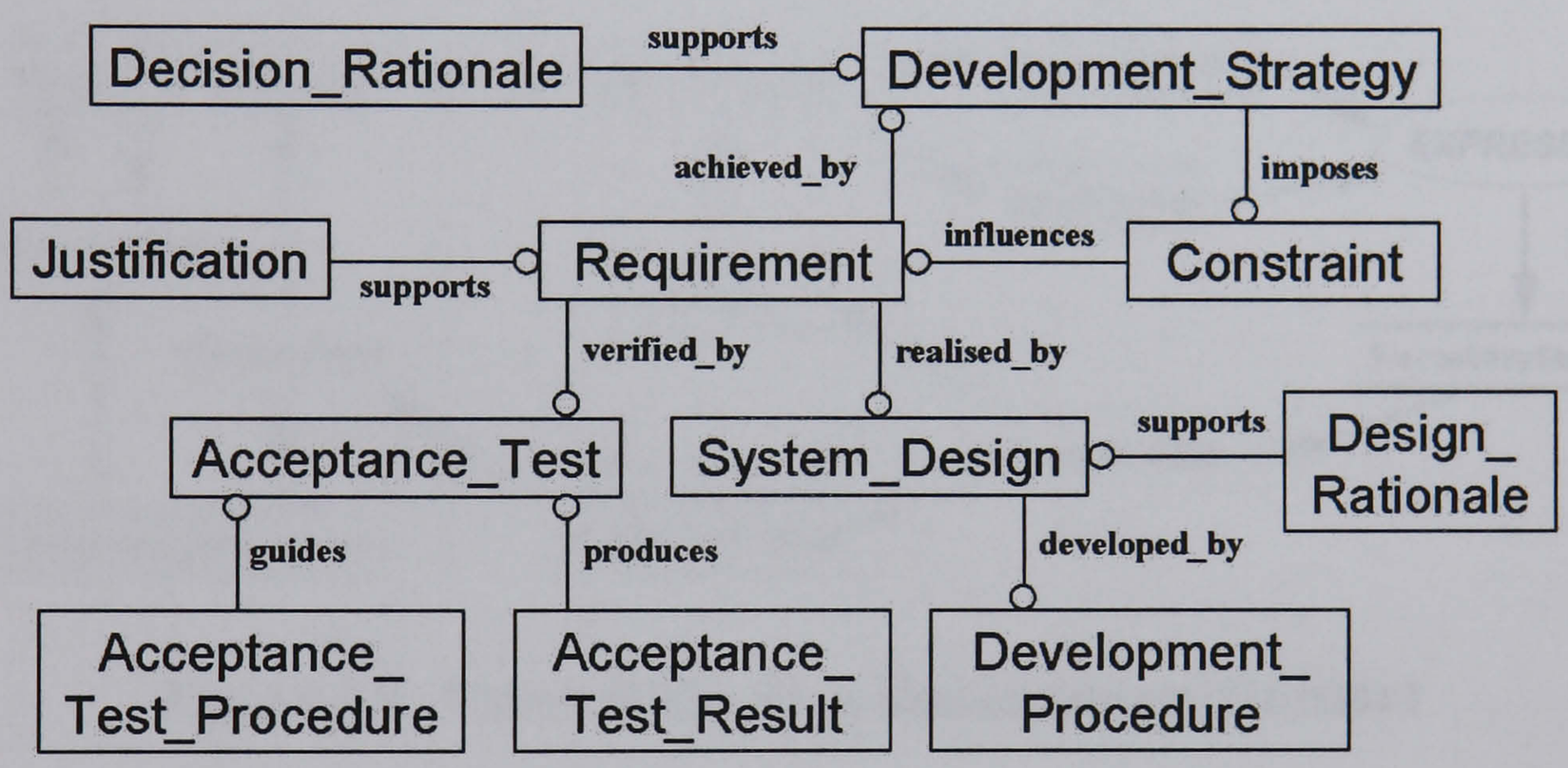


Figure 2.4 Extract of Information Model for System Development

The DRCS language is a useful illustration of basic principles. It has five components, each viewed as a self-contained information structure which focuses on a particular aspect of development. These can be summarised as follows:

- **Synthesis** to record actions to define artifacts and plans
- **Evaluation** to record the assessment of different versions against a specification
- **Intent** to record the association of a decision problem with a solution strategy
- **Versions** to record explorations of the design space for a decision problem
- **Argumentation** to record the reasons for and against accepting a solution

Traceability is the means whereby information of these types is structured in such a way as to record its significance in a design process. The activities through which this is achieved are delineated as *Elicitation*, *Expression* and *Analysis*.

Elicitation is intended to identify information categories that are relevant to development, to generate prompts/questions from elicited information and to incorporate responses. It follows steps that tabulate basic elements of information and their inter-relationships, as well as checking for gaps or inconsistencies in recorded information.

Expression is the act of information extraction in accordance with structuring principles (*i.e.* a projection). This maps information from the traceability tables into component structures (which provide views of a system) and makes the relationships explicit to a specific context. The end-result is a consolidation of these structures into a network of traceability structures that record the information form previous development steps.

Finally, Analysis is principally intended for exploring the design space of a system and recording changes and justifications. It confirms the completeness and consistency of the traceability structures and examines the information in order to produce things like impact assessments against change requests.

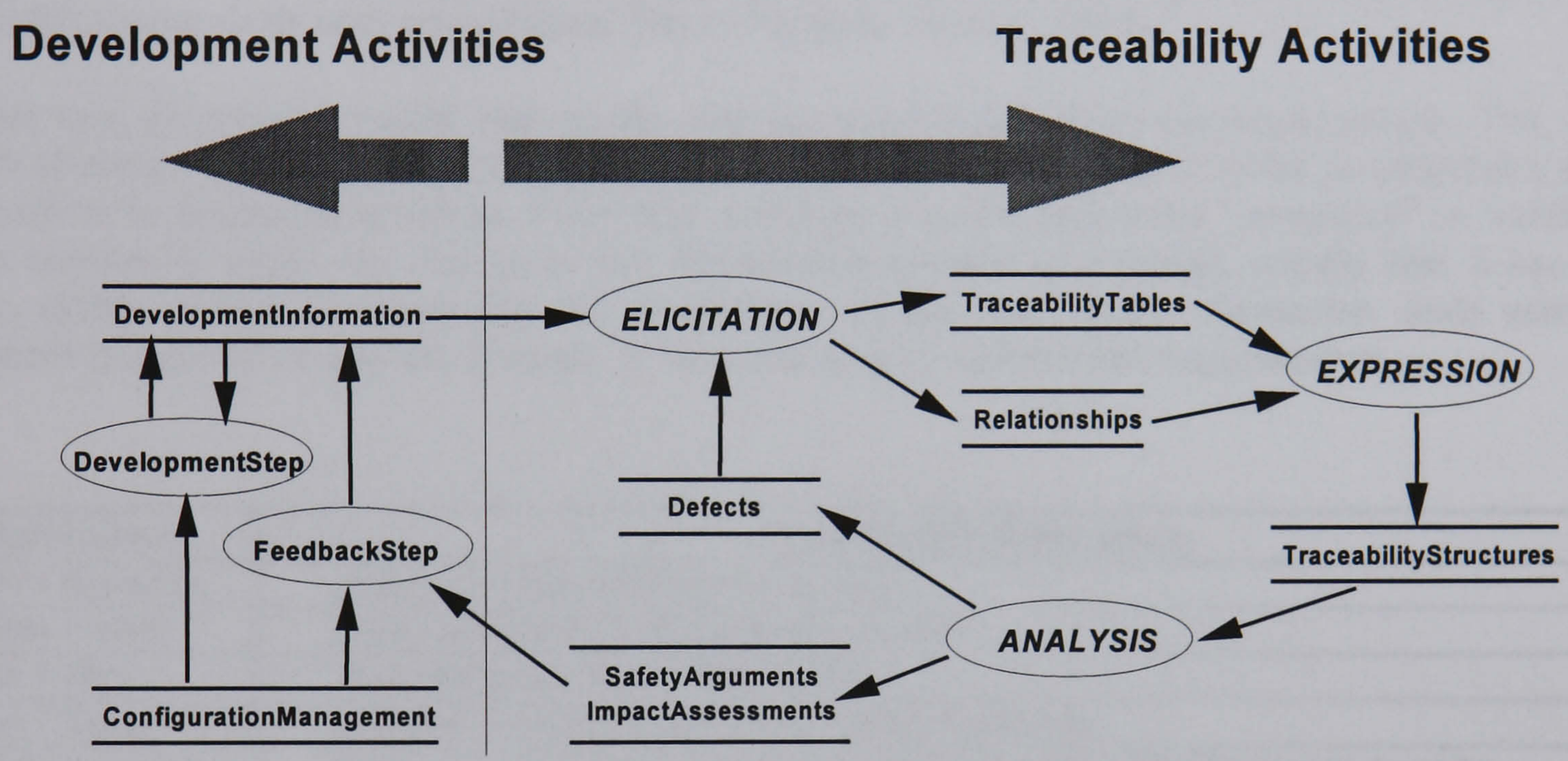


Figure 2.5 Traceability for a Development Context

With respect to the DRCS components, plus two useful additions to cover the design definition and the responsibility and rôles of stakeholders in the design, suitable traceability tables might be established with information fields as specified in Table 2.4. This is not necessarily complete in any given application. What it does illustrate is the set of information types which relate to the components (*i.e.* views of system development) and a few of the more obvious relationships (*i.e.* information which is referenced in more than one table). Note that *Argumentation* is intended to raise questions about any piece of information.

Component	Information Fields					
Synthesis	Module	Attribute	Constraint	Interface	Connection	System
Evaluation	Attribute	Specification	Version			
Intent	Assertion	Decision Problem				
Versions	Version	Status	Design Problem			
Argumentation	Question	Claim	Procedure			
Design	System	Element	Description			
Responsibility	Assertion	Stakeholder	Role			

Table 2.4 Possible Content of Traceability Tables

2.5.2 Design Data Management

The crux of the development problem is that, with the increasing complexity of designs and design processes, the use of computer-aided development facilities will lead to a proliferation of design information that has to be managed. *Design Data Management* (DDM) is regarded as a key enabling technology to achieve higher efficiency and productivity in design. Arguably, *Traceability* is the key enabler for DDM and so it is appropriate to consider the wider issues that would have to be addressed.

It is claimed that five orthogonal dimensions are required for DDM [van den Hamer & Lepoeter 1996], namely Version, View, Hierarchy, Status and Variant. Although not made clear in the published paper on this topic, it is reasonable to assume that the first three dimensions would be driven mainly by technical considerations. The *Version* dimension spans a succession of design modifications produced throughout development. It is noted that very few CAD tools and operating systems have been built with versioning in mind, which means that read/write operations have to be intercepted either by a wrapper (*i.e.* tool encapsulation) or by dedicated functions (*i.e.* tool integration). The *View* dimension provides many representations of a system that is too complicated to describe using any single representation. The *Hierarchy* dimension deals with conventional ideas of system decomposition.

The last two dimensions would deal mainly with issues of organisation and requirements. The *Status* dimension of design corresponds to the organisational procedures employed in order to establish a design and to confirm its fitness for purpose. Note that a change in status (*e.g.* from 'completed' to 'validated') does not necessarily imply any change in the information content of a design, merely that it has been judged to satisfy certain requirements for assessment. Finally, the *Variant* dimension deals with one system which is tailored to address diversity in technical and/or commercial requirements.

Dimensions	Typical Modelling Issue
Version + Hierarchy	Static vs. Dynamic hierarchy models
Hierarchy + View	Level-by-level vs. Non-isomorphic models
Version + View	Data-centric vs. "Roadmap" models
Version + Variant	Describe evolution process of a family of products
Hierarchy + Variant	Relate the required system diversity to the resulting product design diversity
View + Variant	Describe how generic product definitions relate to specific products
Version + Status	Relate the status of a version to the status of its predecessor and successor
View + Status	Introduce quality control elements into a data flow model
Hierarchy + Status	Relate the status of a design to the status of its sub-designs
Variant + Status	Relate the status of a product family to the status of individual product variants

Table 2.5 Modelling Issues for Two-Dimensional Information

Certainly, Computer-Aided Engineering (CAE) frameworks must support versions, views and hierarchies because these are standard projections of a system design. Support for other dimensions is emerging but it is not believed that a fully multi-dimensional data management model could be applied across a wide range of design disciplines. It is also recognised that, to gain user acceptance of such a model, it must closely match what designers think about their data. Finally, on this point, the combination of dimensions raises modelling issues and, as seen in Table 2.5, the implication of combining pairs of dimensions is quite complicated.

2.5.3 Control System Development Activities

Control system design is a major consumer of modelling effort and expertise [*cf.* Section 2.4.3] and it is relevant to consider some of the process implications. One view of Computer-Aided Control System Design (CACSD) has emerged from GARTEUR project FM(AG08) [Magni *et al.* 1997]. This introduces the concept of control design 'activity triangles', as shown in **Figure 2.6**. The supporting explanation [Grubel 1997a, 1997b] is regrettably brief but the basic purpose to identify the computer-aided services which are required to support control design engineering.

The outer triangle refers to the physical plant and its controller, which have to conform to a set of design goals. The inner triangle claims to describe the engineering design activities, based on mathematical representations. The underlying idea is sound, namely that design in the physical domain (*i.e.* the outer triangle only) has given way to a range of predictive modelling and associated activities which are supported by software tools and environments. Iterative development is accommodated at the bottom of the figure, where the results of control synthesis are realised as algorithms and applied either for closed-loop simulation or for hardware-in-the-loop testing.

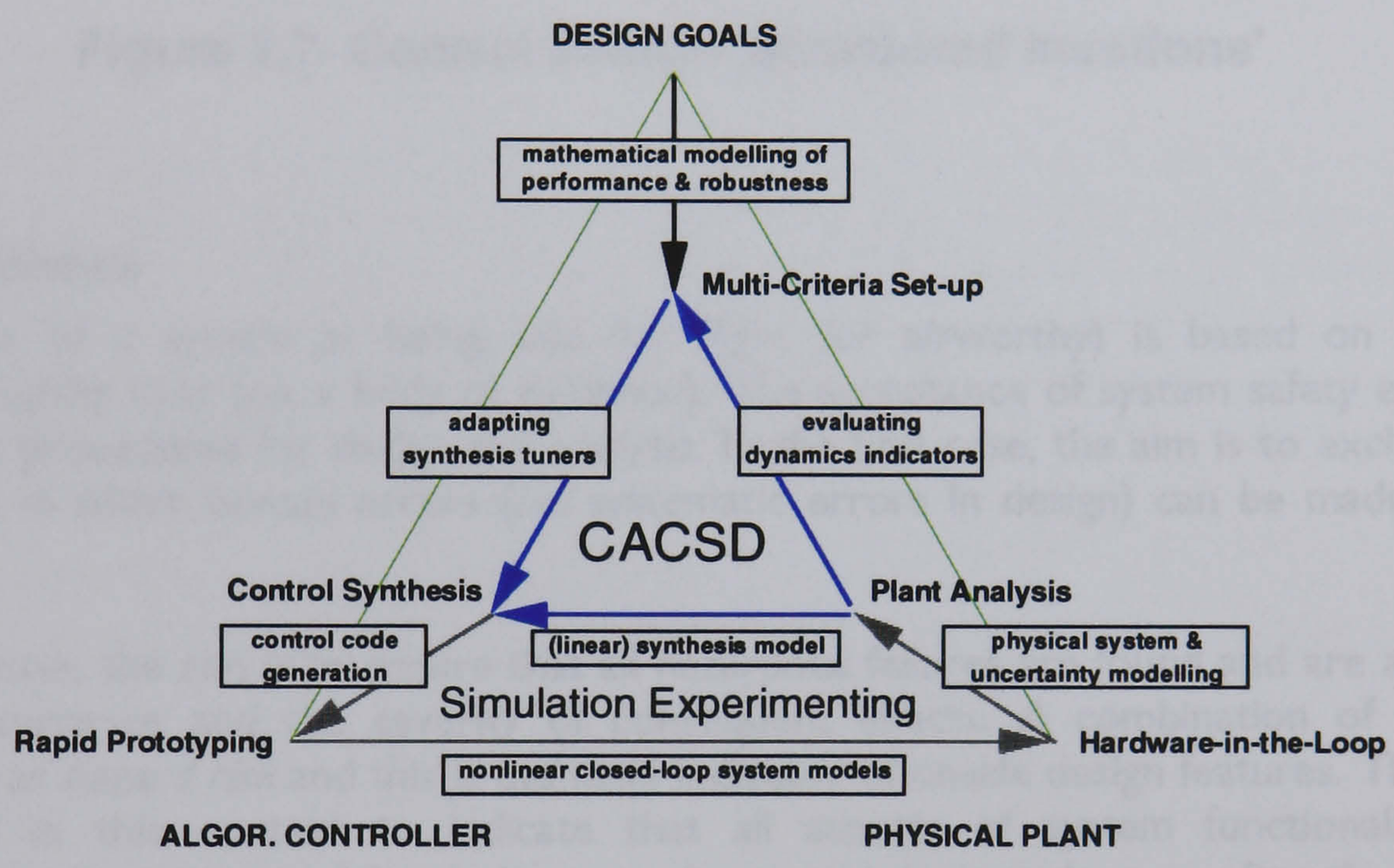


Figure 2.6 Control Design 'Activity Triangles'

In this perspective, it is not clear what process is implied by the activity triangles. Apparently, the vertices denote activities and the arrows denote the delivery of service support dependent on some other activity. For instance, 'Control Synthesis' requires the generation of a synthesis model or multi-model (arising from plant analysis) and the tuning of synthesis parameters (arising from defined multi-criteria). However, CACSD is known to be a highly iterative process but the nature of those iterations is hidden. In order to motivate a process definition for control design and to underpin that with a relevant process definition of mathematical modelling, the activity-based perspective needs some refinement.

Refining the concepts of activity triangles by introducing more explicit activity paths, it is possible to produce a more intuitive view of iterative system evolution. This is proposed in **figure 2.7** and, in order to distinguish it from a mere collection of activities, it describes a set of so-called *structured iterations*. The vertices denote objects which can contain development artifacts (*e.g.* information, models, software and physical hardware). In the inner triangle, the three edges provide exchange activities between objects. This enables an object of interest to be transformed and results to be fed back. As before, the outer triangle relates to the physical domain but, this time, closed-loop simulation is separated from hardware-in-the-loop testing. Simulation is handled by activities which augment the system model (by adding and/or updating control law components) and support system assessment. Testing is a process whereby the control law is realised as a processing specification and then implemented for a hardware target.

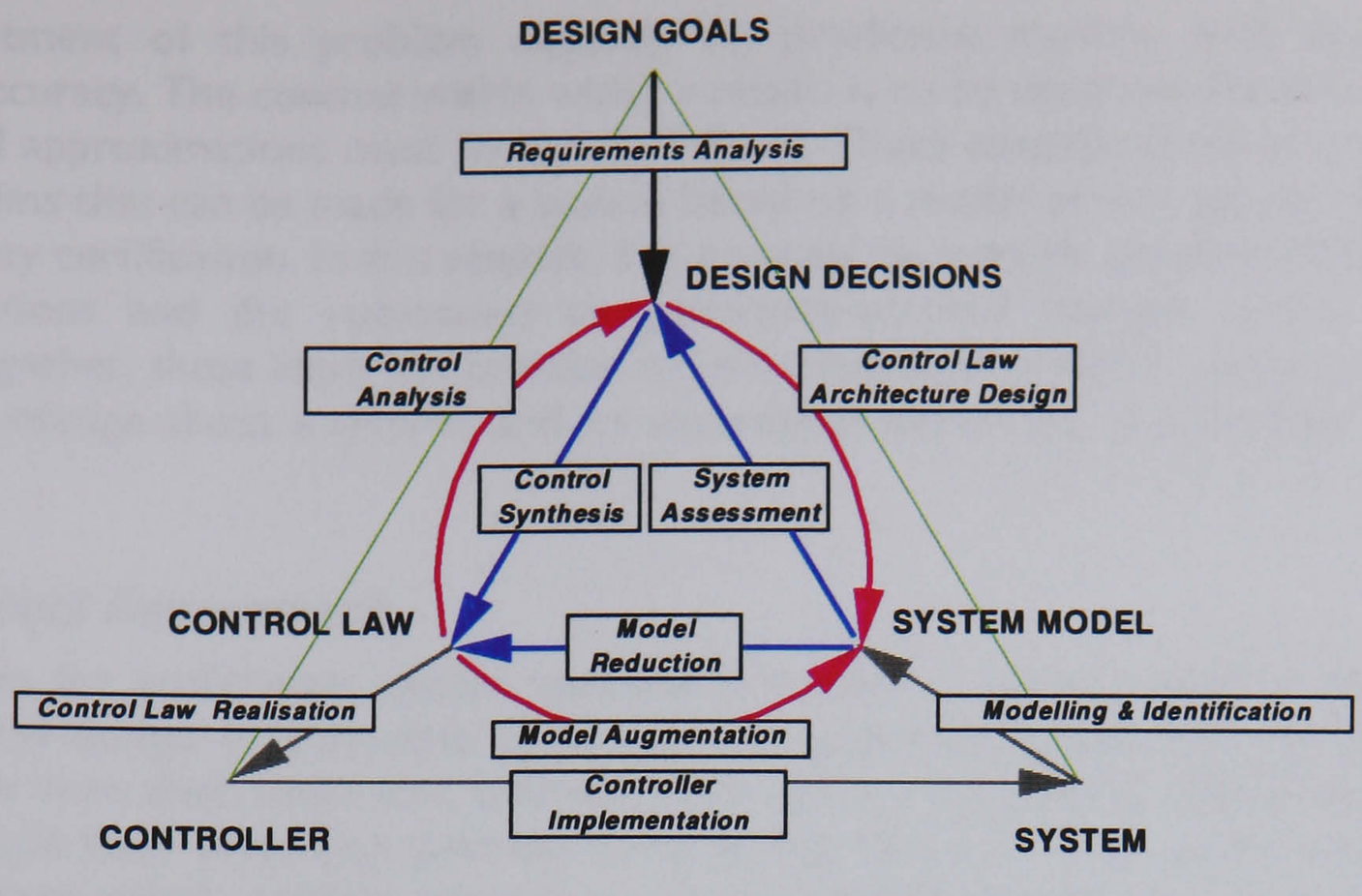


Figure 2.7 Control Design 'Structured Iterations'

2.5.4 Airworthiness

The acceptance of a system as being safe for flight (or airworthy) is based on the meticulous construction of a safety case (*i.e.* a body of evidence). The acceptance of system safety evidence is itself based on rigorous procedures for design and analysis. In the first case, the aim is to exclude or regulate the circumstances in which human errors (*i.e.* systematic errors in design) can be made without being detected.

In the second case, the aim is to ensure that all hazardous failures are found and are assessed for the probability of occurrence and the severity of consequent effects. A combination of probability and severity is defined as *hazard risk* and this is the main indicator of unsafe design features. The term 'Design Integrity' is used in this context to indicate that all aspects of system functionality, design and implementation have been covered by design ... and not merely tested as an after-thought! The term 'Operational Integrity' conveys the idea that a system must continue to work in a predictably correct manner over the full range of expected operational conditions. Thus, a system is designed to fail in particular ways that minimise the risk of losing critical functions or causing hazardous events.

Risk reduction will have an impact on system design *via* the addition of architectural features and functional checks and it may also have an impact on system operation *via* the need for safety devices, warning devices and/or operational restrictions. An additional consideration is that certain failures will only be hazardous under specific conditions, such as:

- **in combination with an independent event** *(i.e. functional independence)*
- **under a specific set of foreseeable circumstances** *(i.e. "time at risk")*
- **in the absence of adequate protection** *(i.e. vulnerability)*
- **in the absence of adequate warning** *(i.e. lack of predictability)*

These conditions mitigate the hazard risk but they do not remove the hazard. It would be unreasonable to argue a safety case solely on the basis of such conditions. A hazard exists regardless of the system implementation and operation; the objective is to avoid the circumstances which may trigger the associated hazardous event [*cf.* Figure 2.1].

As system complexity increases, the number of components and interfaces increase. The main consequences of this from a safety viewpoint are that there will be many more failure modes and specific failure modes can be very complicated. For an integrated system, there is increased potential for systematic errors (*e.g.* specification, design, build, maintenance) and these could have a system-wide effect. Also, unintended coupling might occur between subsystems and the subsystem boundaries might be difficult to visualise.

Effective treatment of this problem depends on predictive models, with appropriate structure, resolution and accuracy. The context within which a model is to be used and the nature of any omissions, uncertainties and approximations must be clearly defined. These considerations will determine the limits of validity for claims that can be made for a system based on a model of that system, *i.e.* the "claim limits" that apply to safety certification. In this respect, it is essential to quantify the *sensitivity* to small/structured parametric variations and the *robustness* to gross/unstructured changes in the system and/or its environment. Together, these imply the creation of two models of a system, namely a *nominal* model (*i.e.* the assumed knowledge about a system) and an *uncertainty* model (*i.e.* the assumed lack of precision in the knowledge).

2.5.4.1 Functional Assessment

FFA is effective for preliminary system assessment because it treats simplified failure mechanisms in advance of detailed design. It is possible to extend this analysis to generic failures that are specified by their effect rather than their cause and, thereby, shed light on the overall robustness to failures. Typical cases would include fixed input (*e.g.* jammed component), step (*e.g.* hard-over), ramp (*e.g.* drift error), impulse (*e.g.* voltage spike), random signal (*e.g.* noise), bandwidth reduction (*e.g.* partial power supply failure). This relies on prior knowledge of the system type and its concept of operation and can provide valuable information on acceptable performance targets.

Having performed failure analysis, there is a need to filter the results in order to obtain a final set of *significant* failure combinations that need to be subjected to hazard risk assessment. To this end, it is necessary to identify those combinations that are likely to cause loss of control or which exhibit severe performance degradation. Also, it is necessary to identify where redesign is feasible in order to remove problems prior to examining hazard conditions. The following steps represent a progressive approach to demonstrating overall functional integrity, with failure modes being filtered at each step on the basis of low probability, minor impact or compliance against requirements.

Step 1: Assess Probability of Loss of Control (P-LOC)

Isolate failures that are critical to the continuous control of a system, consistent with its operability. Assign a probability of occurrence to each failure[8].

Step 2: Assess Worst Case Probability of Failure

Assign a worst-case probability to each remaining failure.

Step 3: Assess System Operability

Identify the impact of human factors and system design on the operational capability of the system. Assess the potential for human-machine interaction to influence or interfere with the intended use of the system within its intended/expected operational environment.

Step 4: Assess Compliance with Performance Requirements

Under the action of each remaining failure, compare the performance of the system against the requirements for input/output performance.

Step 5: Assess Potential for Cost-effective Redesign

For each remaining failure, assess potential redesign options and establish the trade-off between cost and effectiveness of redesign.

Step 6: Assess Potential for Revised Requirements

For each remaining failure, assess the potential for changing the system requirements in order to allow more flexibility in design and/or operation, thereby enabling significant multiple failures to be avoided.

2.5.4.2 Hazard Assessment

Having performed a functional robustness assessment, there is a need to filter the results in order to obtain a final set of *hazardous* failure combinations that need to be incorporated in the hazard log. To this end, it is necessary to perform a full hazard risk assessment in order to understand the criticality of the remaining failure modes in terms of their potential to cause harm. The following steps represent a progressive approach to demonstrating acceptable risk, with failure modes being filtered at each step on the basis of low probability or minor impact.

---

[8] If not analysable, assign a probability of one (i.e. in the absence of other evidence, assume that failure will always occur).

Step 1: Identify System Hazards

Identify physical situations that could lead to an accident, given the occurrence of some initiating event. Define the conditions associated with the hazard and the nature of any injury, illness or damage that would be involved.

Step 2: Identify Accident Sequences

Identify the progression of events (especially failures) that could result in an accident, according to the sequence: Initiating Event → Hazardous State → Accident.

Step 3: Assess Hazard Risk

Map accident sequences to hazards. For individual accidents, assess the probability contributing to an overall probability and the worst credible outcome contributing to an overall severity of a given hazard. Combine probability and severity into an overall risk assessment for each hazard.

Step 4: Assess Potential for Cost-effective Redesign

For each hazard, assess the potential redesign options and establish the trade-off between cost and effectiveness of redesign.

Step 5: Assess Potential for Revised Requirements

For each remaining hazard, assess the potential for changing the system requirements in order to allow more flexibility in design and/or operation, thereby enabling significant multiple failures to be avoided.

2.6 Interim Summary

The discussion so far has covered three main themes. Firstly, it has presented various forms of system assessment. This highlighted 'function', 'event' and 'signal' issues as paramount for modelling vehicle systems integration. Secondly, the main concepts of system structure have been summarised. These were concerned with the composition of a system, especially its architecture, the events that affect dynamic behaviour and the mathematical basis for expressing dynamic behaviour. Thirdly, a perspective on the context of system development context has been developed. This is completely atypical of published work on mathematical modelling. Key elements of this context have been identified as Traceability, DDM, Process (specifically a control system process) and Airworthiness.

All of this demands a modelling capability because extensive analysis must be supported throughout system development. The discussion will now turn to consider what a model must contain and what the most appropriate form of model would be integrated vehicle systems.

2.7 What is a Model?

A model of a system is anything to which an experiment can be applied in order to answer questions about the system. As with the concept of a "system" [*cf.* Section 2.2], the concept of a "model" is ubiquitous and, through over-use, its meaning has become ambiguous, especially where simulation is the purpose for constructing a model. For purposes of the current discussion, a model is defined simply as a *representation* of a system, expressed in a *language*. The process of converting a model between representations will be called a 'transformation'; the process of converting between languages will be called a 'translation' [Gawthrop & Ballance, 1998].

The properties that characterise of a model can be summarised in the form shown in figure 2.8 [*cf.* Bennett 1995], showing various options for deciding on the content of a model and, thus, the methods which are appropriate to its construction. In this scheme, models are distinguished as either physical replicas or mathematical abstractions. The latter are relevant to this thesis and the properties of particular interest are highlighted. The modelling of physical processes will concentrate on models of *dynamic* behaviour using constitutive relationships that are *deterministic* (*i.e.* obeying known laws of physics). Models will be *equation-based* in the sense that they are non-causal (*i.e.* cause and effect are not pre-defined for components) and will focus on *time-driven* solutions, especially in the form of ordinary differential equations with respect to time. One further distinction is important here, namely that between lumped and distributed parameters; for compactness of functional models, the model development will deal almost exclusively with *lumped* parameter models.

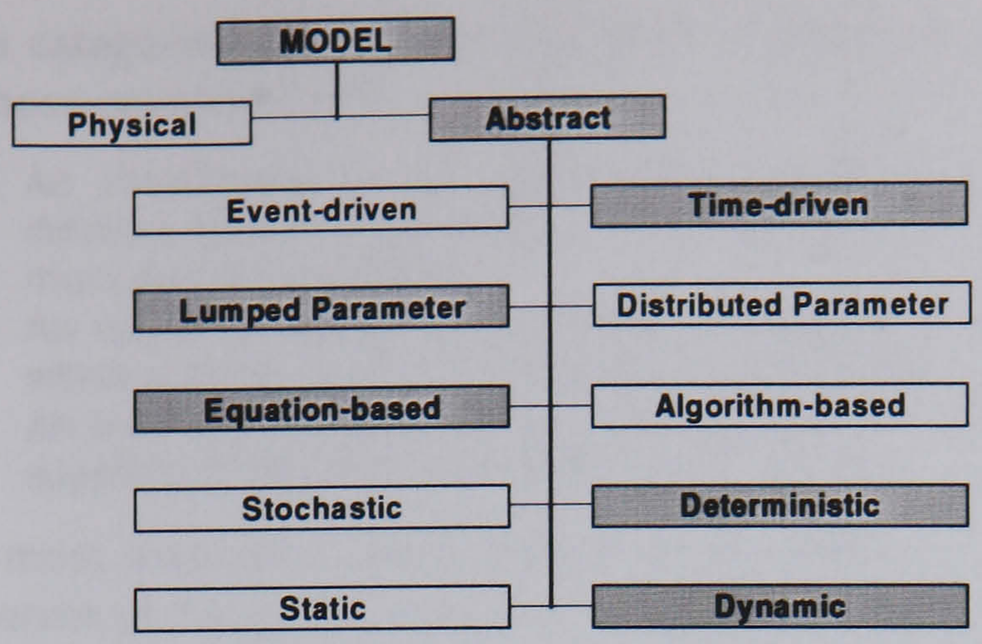


Figure 2.8 Designation of Model Properties

Having obtained a model of a system, a number of activities can be performed. For instance, it can be used to predict steady-state performance, to establish linearised models for control design and to develop simulation codes. The last of these warrants a specific comment because much of the ambiguity referred to above concerns the lack of rigorous distinction between modelling and simulation. For clarification, it is stated here that *modelling* is the process of mapping the structure of a physical system into a mathematical form suitable for analysis. By contrast, *simulation* is the process of experimentation whereby behaviour is calculated from a computational algorithm. This confusion has a well-recognised and justified historical basis, given that much of the effort devoted to modelling of aircraft dynamics (in common with many other industrial applications) was motivated by the need to undertake detailed performance simulation as an integral part of design and qualification.

One view of model development is shown in **figure 2.9** [MacFarlane 1970]. Although the use of language is such that the term 'model' actually means 'simulation', the basic principle is perfectly valid. It is worth mentioning that a 'simulation model' is merely a *representation* of a system in the sense described already, one which can be used to reproduce the actual, expected or approximate behaviour of that system. The process is one of iterative evolution; when there exists close agreement between 'system' and 'model' then the system can be replaced by the model in theoretical investigations. However, it must never be forgotten that a model is usually evolved for a quite specified purpose and its validity is contingent on how it is used.

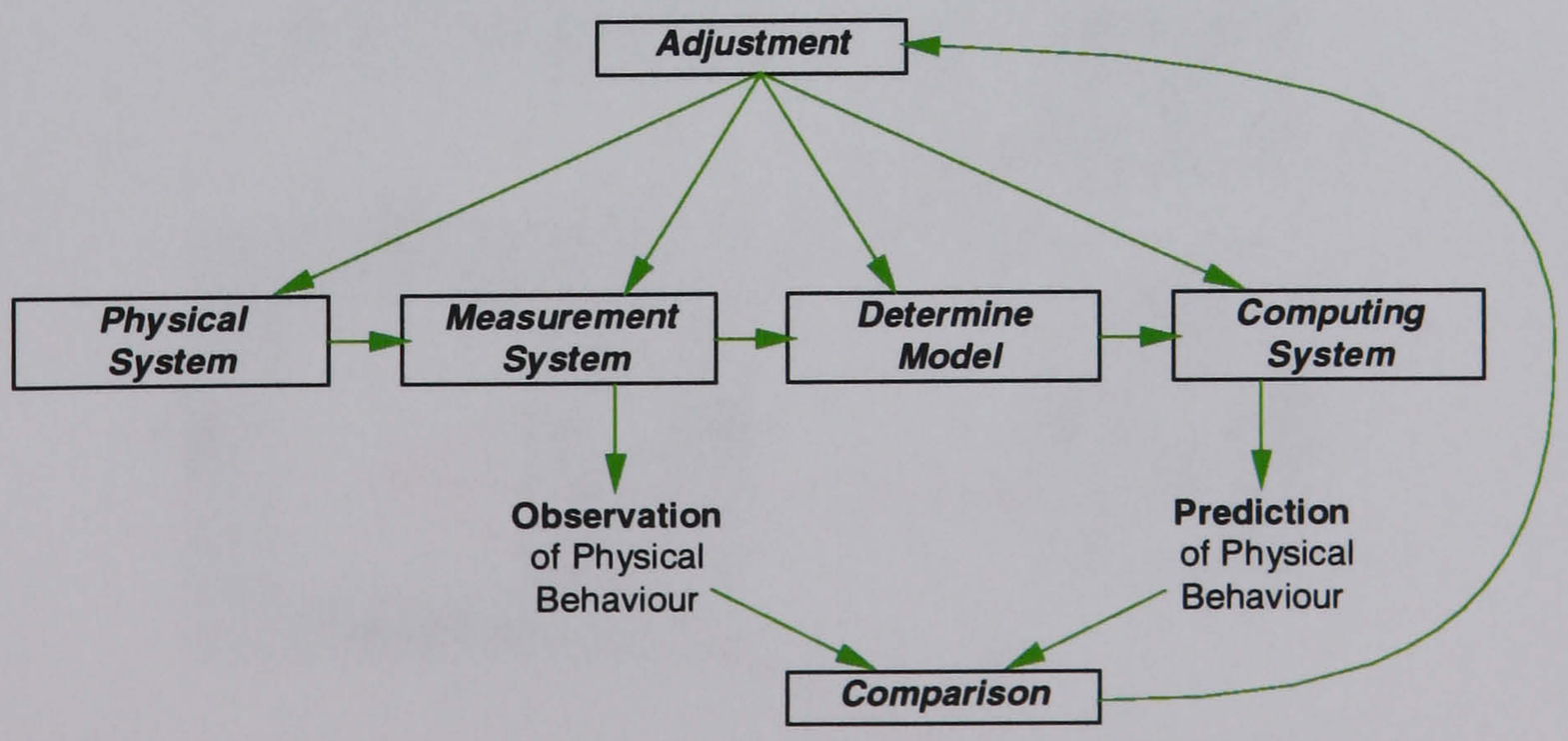


Figure 2.9 Traditional 'Model' Development

It is useful to define three categories of model on the basis of *purpose*, *i.e.* the reason why a model is required in the first place. These categories are

Heuristic Model	An *investigative* model which assists in eliciting fundamental principles, defining system requirements or establishing an appropriate structure for more detailed modelling
Functional Model	An *indicative* model which reflects the essential behaviour of a system and which supports system/subsystem-level analysis and design
Performance Model	An *intensive* model which accurately represents system characteristics and supports component-level analysis and design

Regardless of category, the most important characteristic of a model is 'fitness for purpose', expressed variously and selectively in terms of accuracy, resolution, coverage and so on. This dictates how a model can be used and, crucially, what claims and decisions can be made on the basis of data generated from a model.

It is highly appropriate and intuitive to think of modelling physical systems based on principles of object-oriented development [*cf.* Rumbaugh *et al.* 1991]. Physical systems are built using physical objects, linked using physical connectors of various types. Systems can contain subsystems (*i.e.* systems in their own right), giving the property of *Hierarchy*. Connections transfer all the energy necessary for systems to interact and thus provide the interface medium between objects. The content of an object can be defined independently of its interfaces, giving the property of *Encapsulation*. In fact, many different object types could potentially conform to the same interface definition, giving the property of *Polymorphism*. Note also that a hierarchy of subsystem objects would access the top-level system interface, giving the property of *Inheritance*.

The adoption of this software development paradigm confirms and formalises what modellers and simulationists have taken for granted over many decades. It is particularly useful as a basis for discussing modelling languages and notations, as in this thesis. The basic concept of object-oriented modelling is shown in **Figure 2.10**, namely an arrangement of system component objects within a defined architecture, with interfaces to an external environment.

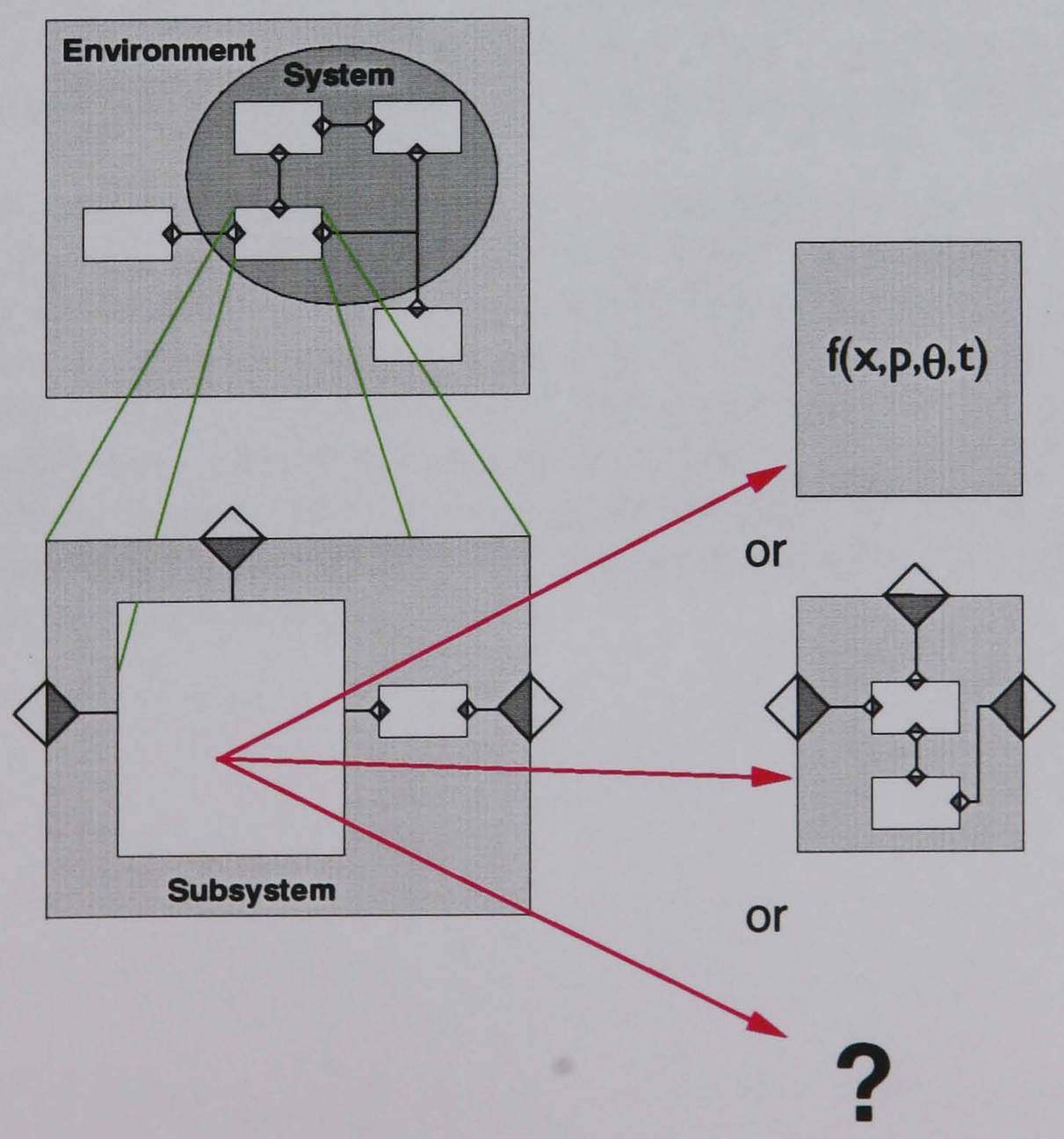


Figure 2.10 Polymorphic Modelling in terms of encapsulated components

The issues and motivations behind *polymorphic modelling* are discussed by de Vries [1994, Chapter 5]. In his definition, this is "the combined application of modularization and subtyping during model building". The first concept introduces an abstraction principle that focuses on the separation between essential and incidental properties of a subsystem. Essential properties are those that are necessary in order to classify the subsystem: incidental properties are not, and may differ depending on context. The second concept makes it possible to refine or specialise a generic type in various forms.

In Figure 2.10 associations between components are created by means of links which 'plug' into ports located on the edges of the component icons. Each port (depicted by a *diamond*) is partitioned into a shaded region that signifies the port declaration within a component definition and an unshaded region which signifies the same port reference in any instance of that component definition. The decomposition reveals that one subsystem happens to be a supertype of various component entities. One is a constitutive relationship (in this case, a mathematical expression involving states x, port variables p, internal variables θ and time t). Another is a composite component definition (which is a model in its own right). The question mark indicates that potentially any other entity could satisfy the same interface or, perhaps, the interface is left open (such that a model is only partially defined).

Language constructions for supertypes, subtypes and interfaces are well established but a few comments are worth making here. The EXPRESS[9] data modelling language contains the concept of *abstract* supertypes, which collect together common attributes but which cannot exist without specialisation; this has no direct value in a system model but may be helpful in organising categories of object that appear in a model. Also, *generic* types are typical in high-order machine languages; to this end, the Java[10] language has the concept of an interface that is most apt in the modelling context. An actual component would then be said to *implement* an interface, as indicated by the subtyping relationship. One important principle of this type of modelling is that, while there can be many component instances within a component definition, it does not make sense to allow *recursion*. This prohibits an instance of a component being contained either directly in its own definition or indirectly in any component hierarchy that stems from its own definition.

It is clear that a model which contains a generic subsystem type is only partially specified and, as such, cannot be simulated or analysed in order to produce numerical results [de Vries 1994, p.98]. A more general statement would be that a model that contains *any* generic component or *any* specific component without instance data is only partially specified. Depending on the purpose of the particular model, it may be appropriate to treat generic components as additional (or temporary) interfaces.

Briefly reviewing the relevance of this philosophy to the system development context [*cf.* section 2.5], it is most convenient to refer to the five dimensions of DDM. *Versions* and *Variants* are supported by polymorphic modelling (and thence bond graphs) because they both rely on rigorous modularisation and subtyping; in fact, from a pure modelling perspective, it is arguable whether these dimensions are orthogonal as originally claimed. *Hierarchies* are also supported by virtue of general principles of decomposition and classification although there is an issue with regard to system properties that are 'flat', such as electrical distribution layout. *Views* are not currently supported in any of the standard treatments of modelling and this is a major limitation to the treatment of complex integrated systems. Finally, *Status* is strictly a traceability issue, not a modelling issue.

---

[9] This forms Part 11 of ISO 10303 *Product Data Representation and Exchange*, otherwise known as the STandard for Exchange of Product model data (STEP).

[10] http://www.javasoft.com/doc/language_environment

2.8 A Model of a Model

In order to impose some formality on the discussion, it is appropriate to develop an outline information model for a 'system model', as shown in **Figure 2.11**. This will define the main objects that a model needs to contain and will remove the use of ambiguous terminology. With further development, this might form a extract from a much larger information model. The development up to this point has been inspired by ESPRIT Project **20496** entitled Systems Engineering Design Representation and Exchange Standardisation (SEDRES)[11] and a review of relevant work in the field of Computer-Aided Control Engineering [Varsamidis 1998].

The main principles within the information model are, firstly, to identify model components, model connectivity, CRs and data with separate groups of entities and, secondly, to rigorously distinguish between definitions and instances.

A Model is a **Component_Definition** is this particular context (although the term could be given several interpretations). In turn this can be specialised as a **Composite_Component** or a **Primitive_Component**, depending on whether the model is reducible or not, respectively. A composite component, by definition, is decomposed into a **Component_instance** network; A primitive component has its behaviour defined by a **Constitutive_Relationship**. Note that an component interface, as discussed in **Section 2.7**, is simply a component definition that has not specialised (*i.e.* its form has yet to be determined). Note also that, in the way that component definitions and instances are related, a hierarchical decomposition is non-recursive.

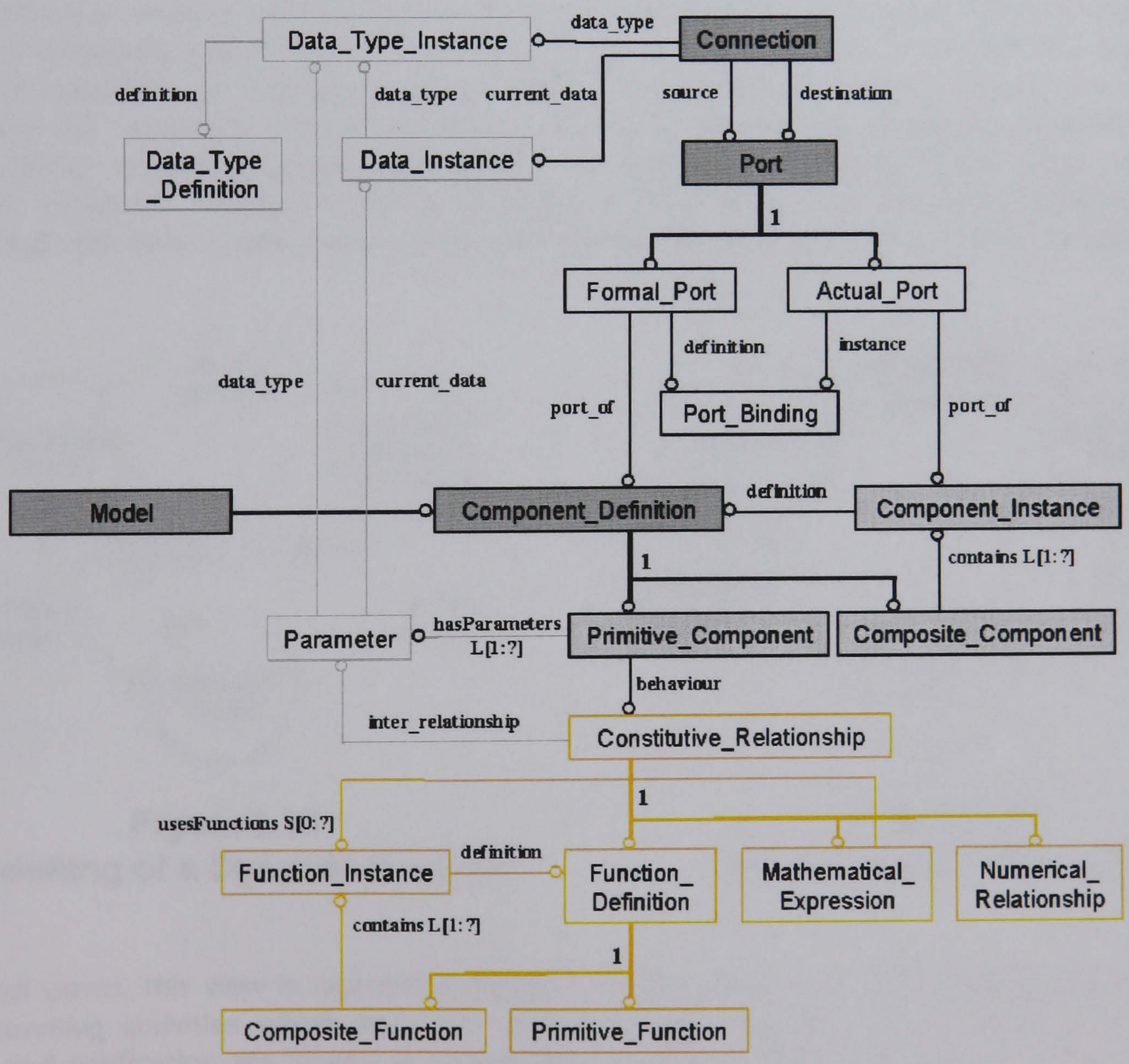


Figure 2.11 Extract of Information Model for System "Modelling"

[11] http://www.ida.liu.se/projects/sedres

Models are drawn using components, ports and connections. **A Connection** has a **Port** at each end, typically designated a source and a destination. The ports of a component definition are defined by the concept of a **Formal_Port** while those of a component instance are defined by the concept of an **Actual_Port**. When models are built, each connection to an actual port needs to be referenced to the equivalent formal port in the relevant component definition by means of a **Port_Binding** mechanism.

A constitutive relationship (or CR) can be specialised in various forms, as shown. For illustration, one scenario has been developed in which functions can be defined using a hierarchy. Also, mathematical expressions (in other words, equations) can optionally contain functions. This topic is a major area of research in its own right[12] and will not be discussed further in this thesis.

Finally, what about data? Primitive components and CRs contain **Parameter** sets and the **Connection** mechanism transfer information between components, via the port bindings. Both of these entities need to reference a **Data_Instance** in order to hold current values of data and a **Data_Type_Instance** in order to declare a parameter or variable as being of a particular data type. Each data type instance then references a **Data_Type_Definition**.

2.9 Model Verification and Validation

No discussion about modelling would be complete without **V**erification and **V**alidation (V&V). These have a strong distinction, as recommended by the Society for Computer Simulation (Technical Committee for Model Credibility) [SCS 1979]. With a number of specific points of clarification [*e.g.* Murray-Smith 1995], *Verification* confirms that the *internal* structure of a model is correct and that its constituent parts are mutually consistent and *Validation* confirms that the *external* behaviour is credible and that it satisfies its customer requirements. Typically, the latter implies the use of test scenarios in order to demonstrate that a model can reproduce known benchmarks and it is important to distinguish between *theoretical* validation (which considers general principles) and *functional* validation (which deals with specific mechanisms contained in actual systems). Arguably, a third category of *empirical* validation could be added, dealing with direct measurements from real systems (independent of any model constructions).

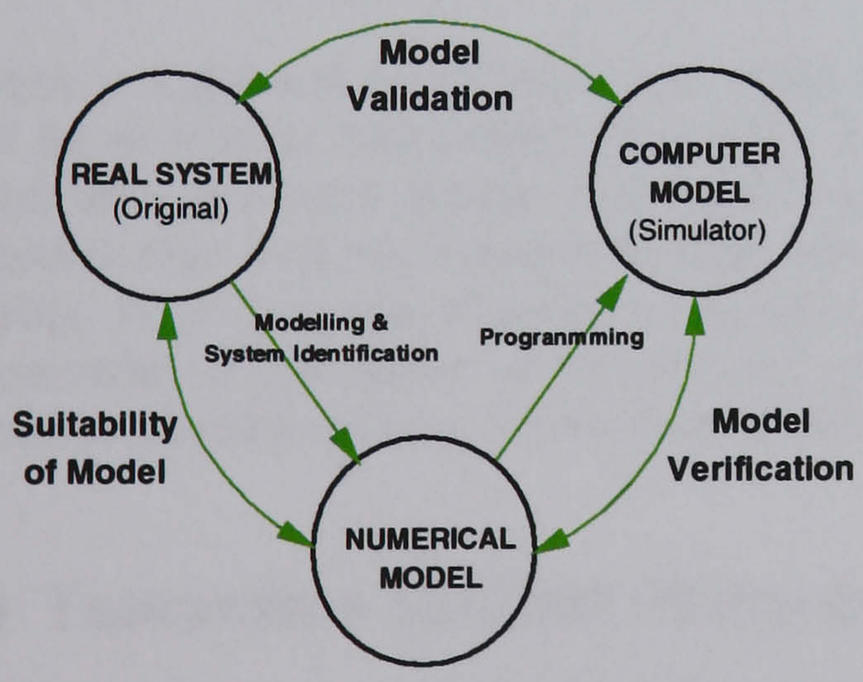


Figure 2.12
"Modelling of a Dynamic System"

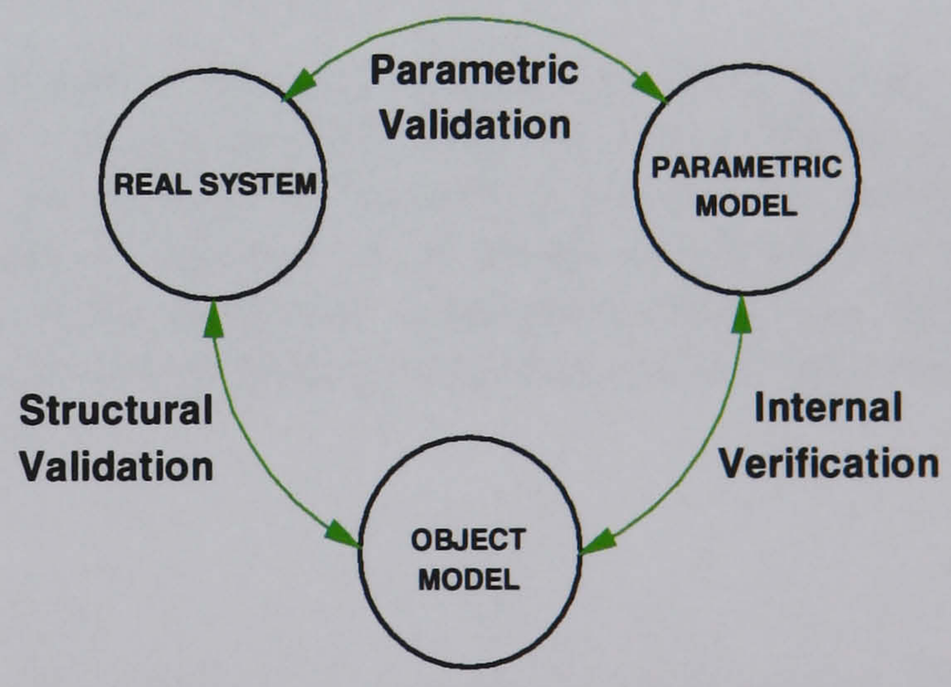


Figure 2.13
Validation and Verification

In overall terms, this view is depicted in **figure 2.12** [Buccholz *et al.* 1995] highlighting the modelling and programming activities which lead from a real system through to a computer-based simulation. Validation and verification are shown as comparative exercises, along with a vague reference to model 'suitability'.

---

[12] Refer to http://www.OpenMath.org/intro.html, http://www.inria.fr/OpenMath/ or http://pdg.cecm.sfu.ca/OpenMath/Lib.

Adopting a different perspective, as discussed in Section 2.7 it is useful to separate the structure of a model from its parametric instantiation. In this way, the fundamental task is to map the constituents of a real system into an object model. A *structural* validation can be performed in order to confirm that both the component resolution and interface definition correctly reflect the functional organisation of the system and that the model is capable of delivering the information required of it. Instance data transforms an object model into a parametric model that can support analysis and simulation. A *parametric* validation is then applied in order to compare model prediction with actual measurement. This revised scheme is shown in figure 2.13. In contrast, the numerical and computer models of figure 2.12 are merely two parametric representations. The prime distinction is the explicit declaration of model structure which offers an interpretation of 'suitability' which can support a strategy for model testing.

Test strategies for models fit into six broad categories although the terminology can vary considerably. These involve simulation and other analyses in order to confirm static and dynamic figures within known tolerances. The main categories can be described as follows:

- **Replication** where consistent results are produced by independent means
- **Substitution** where part of a model is replaced by an equivalent model to confirm its behaviour
- **Approximation** where a model is 'hacked' to its bare essentials to confirm dominant properties
- **Inversion** where a model is reconstructed in order to reproduce input stimuli from a known response
- **Identification** where Internal model parameters are reconstructed from measured inputs/outputs
- **Sensitivity** where parametric variation is quantified against predefined design margins

The rationale is to build confidence in the correct operation of a model. Choice of strategy and the level of testing will depend on the criticality of application and the perceived complexity of the system of interest. This will be reflected in the number of components and component interfaces; it will be influenced by carry-over experience from systems of similar types and previous application of relevant technologies.

With increasing scale and connectivity of systems, less reliance can be placed on testing and more has to be placed on knowledge that model development is dependable and that development activities are traceable. The basic principle is that, when a model is too big to test effectively, qualification and certification rests on adherence to a set of standard development practices. These form part of a disciplined process and auditors will look for evidence that the process has been followed.

There is a general recognition that risks associated with a complex system can never be zero but should be as low as reasonably practicable. In pragmatic terms, this says that risk reduction should be pursued until the *cost* grossly outweighs *benefit*. In the context of modelling, there is an additional recognition that it is not feasible to fully specify a model in advance of its design and implementation. Invariably, in all but the simplest of systems, there will be significant uncertainty about the detailed characteristics of the system of interest and, depending on the modelling requirements, the full extent of development problems may not be immediately apparent.

2.10 Towards a Unified Philosophy

Applying the ideas presented in Sections 2.7 through to 2.9, it is possible to formulate a unified philosophy for modelling complex systems. It should be recognised from the outset that, while a characterisation of the form $F(\underline{x}(t),\underline{u}(t),t)$ [as used in (2.1)] is applicable to any system, it is a representation based on mathematical equations alone. This does not explicitly incorporate the decomposition of a system into subsystems or the classification of those subsystems. Therefore, it does not offer insight into system structure [de Vries 1994].

In the interests of integrity, efficiency and V&V, a modelling method must be simple to use (for the specialist modellers through to intelligent observers). Its notation must be sufficiently expressive to be able to represent a wide range of system types (as appropriate to vehicle system integration). It must offer a guarantee of internal consistency and, to that end, an information model is essential. Above all, the method must be verifiability to a high level of coverage.

In the interests of long-term viability, a modelling method needs to provide features that support reuse of models and model components. Also, there needs to be a means of changing and extending the notation in order to accommodate new concepts and to respond to new types of modelling.

These ideas are formalised in the following set of principles:

Simplicity
: It will be easy to read and use, and its concepts must be clearly reflected in its semantics. It will be easily recognisable as a graphical representation and avoid complicated and/or ambiguous syntax.

Expressibility
: It will be able to express all component classes, properties and inter-connection mechanisms that are necessary or desirable for modelling aircraft vehicle systems.

Consistency
: It will provide a means of enforcing rules about the construction of models so that component classes, properties and inter-connections can only be manipulated within a correct context.

Verifiability
: It will provide features that enable analysis to be performed on whole models or fragments of models in order to demonstrate the notation correctly represent model structure and the corresponding dynamic behaviour.

Reusability
: It will enable models and model fragments to be embedded in larger model structures without re-design or wrapping (unless this violates the rules for model consistency).

Extensibility
: It will support the introduction of new graphical notations and amendments to existing notations.

These principles will be applied in order to select a modelling method and subsequently (in the next chapter) develop the features necessary to address the detailed requirements of large, complicated engineering systems.

2.11 Candidate Methods

2.11.1 Signal Flow Block Diagrams

Signal flow is the legacy of simulation practice from analogue computation [*cf.* Bennett 1995]. It is highly effective as a notation for mapping algorithms (*e.g.* for signal processing or control laws) and it is the standard approach to the teaching of modelling and simulation techniques. It is not effective for representing power transfer because it must separate the effort and flow variables (or, equivalently, the through and across variables) and assign signals to each. The notation is verbose in this application and straightforward power transfer around a network can become difficult to interpret. The focus is placed on direction of signal flow and not what the signals represent (*i.e.* effort and flow being freely interchanged or even abandoned in favour of some other convenient variable. Admittance is tracked 'forwards' through the model and impedance is accounted by numerous feedback loops.

2.11.2 Multiport Networks

Multi-port notations overcome much of this inconvenience. The usual format is that of a two-port network, handling data exchange between components. It can be to model effort and flow variables and, thus, power transfer but it is not necessarily the case. The main problem lies with causality [*cf.* Section 3.4] or, in other words, the directions in which the two variables are passed. Note that they are passed in opposite directions but there are two possibilities. This means that, if there is a need to accommodate components in a range of causal contexts (which is invariably the case), a range of component definitions are required, varying only in respect of their interface definition. For instance, a resistor has three possible contexts, depending on how it relates to voltage/current data on its connections. The equation $\Delta V = iR$ can be resolved as $V_2 := V_1 + iR$, $V_1 := V_2 - iR$ or $I := (V_2 - V_1)/R$. This becomes extremely tiresome when attempting to construct very large models.

2.11.3 Bond Graphs

Bond graphs [Paynter 1961; Thoma 1975, 1990; Wellstead 1979; Breedveld 1984b; Cellier 1991; Karnopp 1969; Karnopp, Margolis & Rosenberg 1990; Gawthrop & Smith 1996], on the other hand, offer excellent insight and are well suited to an engineering context. Decomposition is based on energetic behaviour and classification of the resulting elements is natural and rigorous. The notation lends itself to a powerful graphical representation that is able to map the topology of physical systems in a clear and unambiguous manner. The applicability of bond graphs is consistent with the properties highlighted in figure 2.8 and, as will be discussed in the following chapters, this proves to be highly appropriate for aircraft vehicle systems.

2.11.4 A Motivating Example

In order to illustrate the relative merits of the three notations, an example is given in Figure 2.14. It is a very simple arrangement of two coupled tanks. The bond graph representation is a one-to-one mapping of the system topology. It uses primitive components throughout [*cf.* Chapter 3] with causal augmentation to indicate the propagation of pressure information across the model. In the manner which the model has been established, the tanks determine pressure and also pressure is imposed at each external port.

The two-port network is very similar is its function although, now, the transfer of variables is explicit. Colour has been used in order to distinguish effort and flow variables but there is nothing in the notation that requires this (or anything similar) to be done and therefore the variables are anonymous. The non-junction components are transfer function blocks. For each block, the transfer function will depend on which variable is supplied as the input. In the equivalent bond graph, this does not matter because the block is replaced by a primitive component with a CR; the CR (*i.e.* an equation) is resolved automatically depending on the causal augmentation.

The signal flow block diagram is already showing its weakness. Topologically it is not the same as the system that it represents. In effect, it has disregarded the natural coupling between complementary power variables and provided, in stead, a flattened out version with a clear forward path and numerous feedback paths. Origami has been played in reverse. As with the two-port network, colour has been used to distinguish between variables but there is nothing the signal flow notation that requires this. In comparison with bond graphs, signal flow is unstructured and models of physical systems are completely anonymous. This reiterates the message delivered in the first paragraph of Section 2.10; it is essential that models of big systems offer insight into system structure.

The real test of versatility in a modelling notation is to establish how it responds to a change in causality. This is a major issue in network models because the insertion of new components can have a profound impact on the underlying signal traffic. The bond graph solution is trivial; the causal augmentation is flipped and this is a graphically operation. The two-port network solution is to swap its signal connections and make any adjustments that might be necessary in the transfer function blocks. This means some re-programming or the substitution of a different variant. The signal flow solution is totally inelegant. It involves re-wiring its connection network such that, not only is the signal flow block diagram not necessarily an obvious match for a system topology, it is not even stable in the face of minor changes in system content. Inserting a single resistor into an electrical circuit model might involve major changes in the signal flow model.

Even with the brief assessment, the bond graph notation is obviously the appropriate choice for building integrated system models. The other two methods are well suited to other classes of problem.

Figure 2.14 Comparison of Network Model Notations

2.12 Conclusion: Feasibility of a Unified Modelling Method

With the growth of mathematical modelling as an integrating technology for *development* of complex airborne systems, it is clear that a highly structured and intuitive approach will be required to the *modelling* of complex airborne systems. This must ensure the rapid development of families of models for a range of different purposes, such as performance prediction, safety assessment, functional test, concept demonstration and so on.

It is known that the bond graph method is effective in representing physical processes and in exposing the underlying structure of energy transfer that may be hidden by other methods. The philosophy is based on a close mapping of the topology of aircraft systems, based on architectural schematics and, where appropriate, physical layouts. The ultimate justification of this method (as opposed to any other method) for object-oriented modelling of aircraft vehicle systems is based on its ability to satisfy the aims of Simplicity, Expressibility, Consistency, Verifiability, Reusability and Extensibility (as discussed in section 2.10).

A range of method developments has been undertaken as part of the work discussed in this thesis. These are directed towards the improvement of existing notations in order to provide a modelling method that can be readily applied to aircraft vehicle systems and assist engineers in design and evaluation tasks. Standard bond graph concepts have been modified in various ways by the addition of novel features that were found to be necessary and/or desirable in order to handle practical engineering situations. These will be describes the next chapter and illustrated in the subsequent chapters.

Chapter 3

Principles of Bond Graph Modelling

SUMMARY
Bond graphs represent an established and effective approach to polymorphic modelling, one which offers a system decomposition based on energetic behaviour and a natural and rigorous classification of resulting elements. This has the advantage of using a simple notation which is common across all energy domains and which provides a one-to-one mapping between a system schematic and its underlying functional mechanisms. This chapter establishes a framework for bond graph modelling concepts and introduces a range of new concepts which can be applied to the modelling of air vehicle systems.

3.1 Basic Concepts

Bond graphs provide an object-oriented method for representing power transfer around a network of system components. Each bond provides an interconnection along which energy can pass (similar in concept to a chemical bond). As a philosophy, bond graph modelling allows a direct mapping of engineering schematics and a clear physical interpretation of how systems behave and interact. As a notation, it provides a functional system schematic and it is very quick to apply. It handles causality in an open and explicit manner and, at an elementary level, it ensures that model components are internally consistent.

Bond graphs can be applied across all energy domains because they represent systems in terms of their energy flow (*i.e.* power) characteristics. Power is related to the modulation of two complementary variables (or *covariables*), called *effort* and *flow*, the product of which is power. There are two state variables, called *momentum* and *displacement*, which relate to the covariables in the following way:

$$\frac{d}{dt}[\textbf{Momentum}] = \textbf{Effort}$$

$$\frac{d}{dt}[\textbf{Displacement}] = \textbf{Flow}$$

Energy domain variables are summarised in Table 3.1. Their inter-relationship is shown in Figure 3.1, introducing the familiar concepts of inertia, capacitance and resistance. Functionally, the first two concepts relate to energy storage and the third relates to energy dissipation. It is worth noting straight away that magnetic and thermal processes involve a permeation of energy through a medium rather than a bulk movement of matter from one place to another. At an atomic level, there is mutual interaction between neighbours in order to redistribute the stored energy. Thus, mechanical, electrical and hydraulic processes possess momentum: thermal and magnetic processes do not.

Domain	Covariables		State Variables	
	Effort	Flow	Momentum	Displacement
Linear Mechanics	Force	Velocity	Momentum	Displacement
Rotational Mechanics	Torque	Angular Velocity	Angular Momentum	Angle
Hydraulics	Pressure	Volumetric Flow	Pressure Momentum	Volume
Electrics	Voltage *or* EMF	Current	Flux	Charge
Magnetics	MMF	Flux Rate	-	Flux
Thermodynamics	Temperature	Entropy Flow	-	Entropy

Table 3.1 Standard Bond Graph Variables

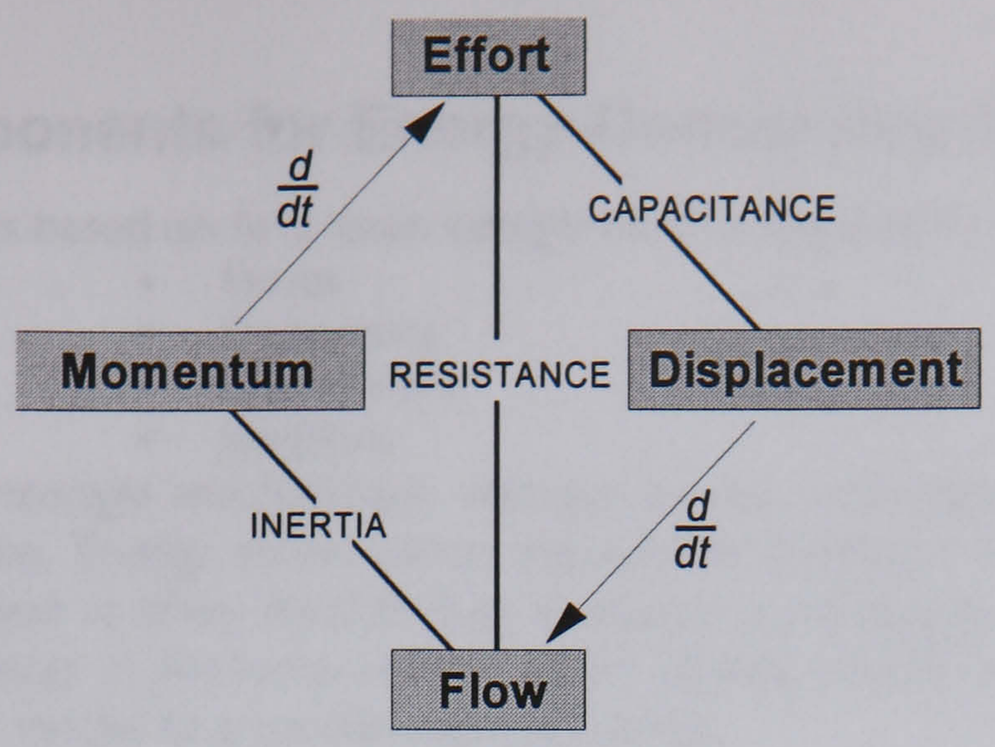


Figure 3.1 Inter-relationship between Bond Graph Variables

In its simplest form, a bond is an association between two objects and is drawn as a line. *Orientation* is indicted by a harpoon, denoting the assumed direction of positive power transfer. For annotation (by arbitrary convention), 'flow' is on the side with the harpoon tip and 'effort' is on the other side. These variables are individually passed along the bond, normally in opposite direction although there are special circumstances in which the directions are the same.

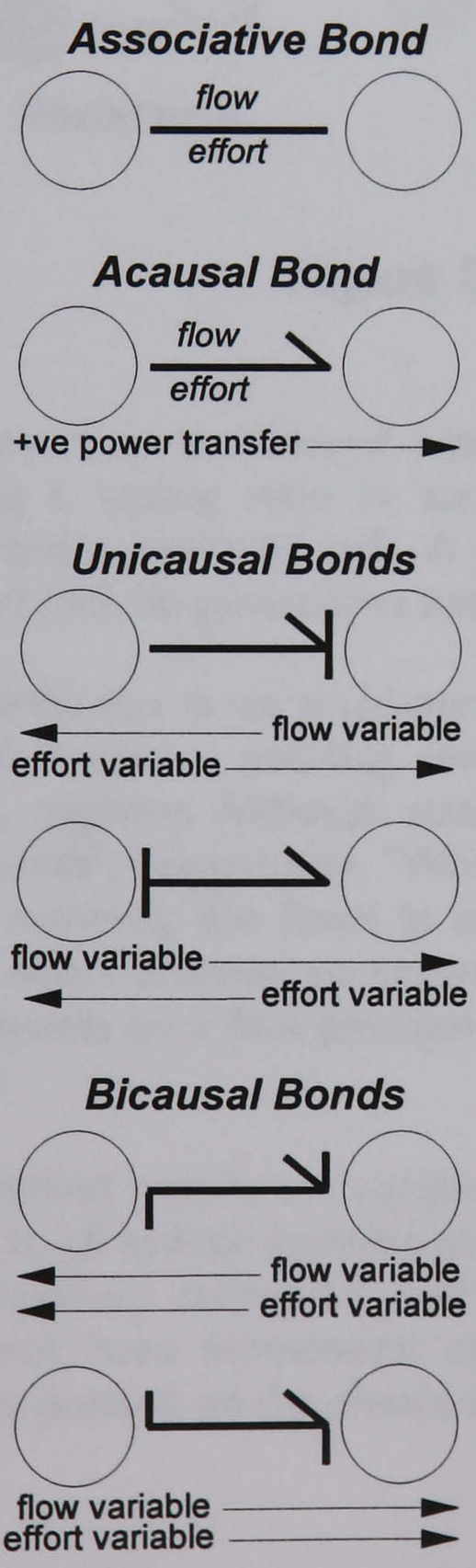


Figure 3.2 Basic Notation

The exchange of variables between objects determines the context of each object with respect to its neighbours. This property is called *Causality* and it introduces the crucial distinction between cause and effect for any object. In terms of signal flow, this means 'input' and 'output', respectively.

A bond is unicausal if its variables pass in opposite directions: it is bicausal is they pass in the same direction. Graphically, on the effort side of the bond, a short stroke is drawn orthogonal to the bond at the end at which an *effort* signal arrives and, on the flow side, a short stroke is drawn at the end from which a *flow* signal originates.

Bicausal bond graphs arise in cases where particular constraints are imposed on the way in which information passes around a model. System inversion is one such case, in which a model is used in order to determine what stimuli would be required in order to generate a given response. Bicausal bonds will arise if a given stimulus and its associated response are not collocated in the same bond. Parameter identification is another case, in which a model is set up with stimuli and response data in order to calculate a value of some internal parameter (*e.g.* resistance). This has application in system identification and in fault diagnostics.

A summary of notation is given in Figure 3.2.

3.2 Primitive Components for Energy-Conserving Systems

Bond graph functionality is based on four main categories of component, as shown in Figure 3.3:

- Stores
- Dissipators
- Transducers
- Junctions

There are two energy *storage* mechanisms, namely inertia and capacitance, and one *dissipation* mechanism, namely resistance. Energy stores drive the system dynamics and dissipators determine the operating efficiency. Dissipation is often modelled as if energy were simply being lost. In fact, all except thermal systems dissipate energy in the form of heat, which strictly should be represented as power being transferred from one system model to a parallel thermal model.

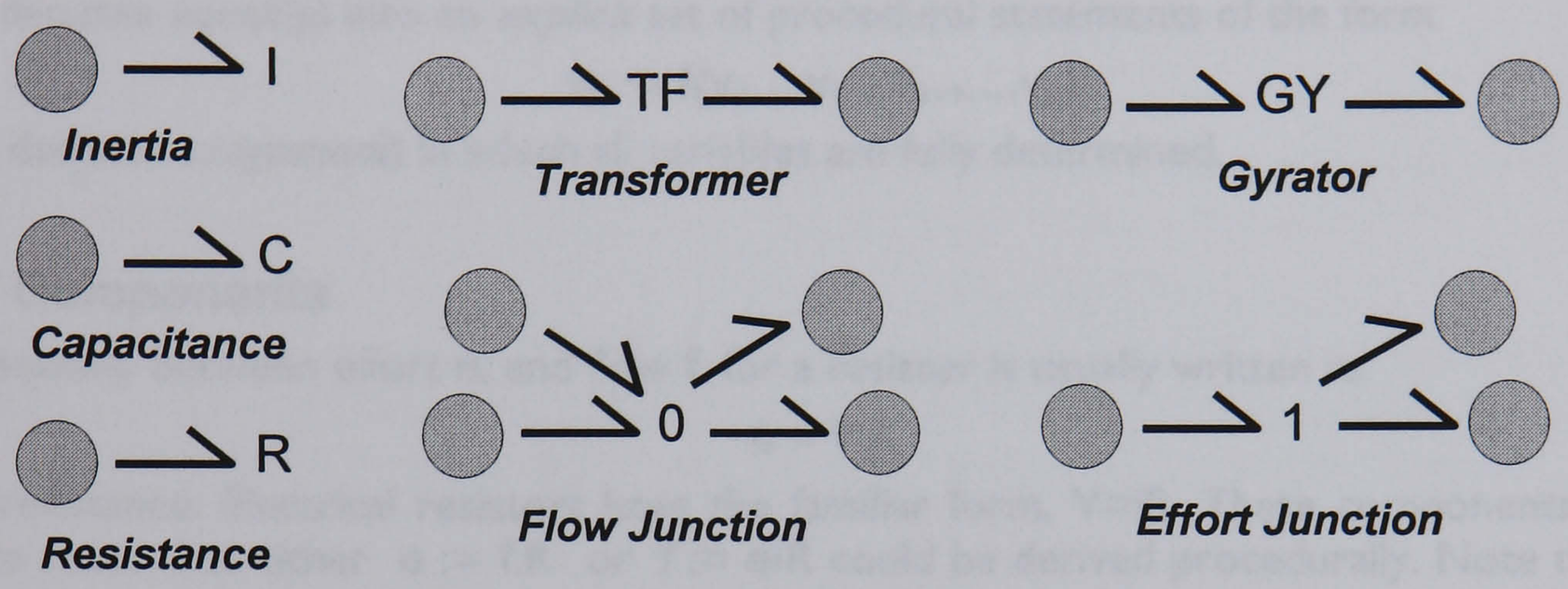


Figure 3.3 Primitive Bond Graph Objects

Power conversion is achieved with two-port *transducers*. A transformer transmits effort and flow variables using a scaling ratio in such a way as to conserve energy. Examples include an electrical transformer[1] and a hydraulic jack. A gyrator exchanges effort and flow variables using a similar scaling ratio. Examples include gyroscopes and electric motors[2].

Power distribution is via multi-port *junctions*. The bond orientations define the sign convention for summation. A harpoon pointing inwards is interpreted as *positive and* one pointing outwards is interpreted as *negative*. Although strictly incorrect, it is intuitively obvious to speak about 'input bonds' and 'output bonds', respectively. "What goes in must come out!" So, a flow junction generates a flow in one bond by summing the flows in all other bonds. "A lot of little pushes can be replaced by one big push!" So, an effort junction generates an effort in one bond by summing the efforts in all other bonds. Note that all bonds on a flow junction experience the same effort and all bonds on a n effort junction see the same flow.

These primitives enable the construction of models of physical processes that conserve energy. They are applicable to all energy domains and, as such, the bond graph method is generic. The only exception is that the I component cannot be used as an effort store in thermal or magnetic processes because these domains do not have momentum and, thus, have no concept of 'inertia'. The guarantee of energy conservation is posited on the choice of effort and flow variables which are power covariables, of the type in Table 3.1.

---

[1] The ratio of primary to secondary windings determines the change in both voltage and current; if the voltage steps up by that ratio then the current steps down by the same ratio and vice versa.

[2] The voltage across the terminals is proportional to the rotational speed; the torque on the motor shaft is proportional to the current flowing through the motor.

3.3 Primitive Class Definitions

Each component object signifies a physical mechanism that can be characterised by a mathematical relationship between effort and flow, a so-called *constitutive relationship* (CR). This default CRs are linear although this does not prevent the modeller from introducing nonlinear versions of arbitraty complexity. Also, the modeller is free to define new primitive objects as required.

Without question, *causality* is the most important issue in modelling [*cf.* van Dijk 1994, Gawthrop & Smith 1992]. It determines how a CR is rearranged in order to establish the link between *cause* and *effort* for any object given the context in which it finds itself. In practice, this means deciding for each bond connected to an object, which covariable (*effort* or *flow*) is the 'input' and which is the 'output'. The aim is to transfer an *implicit* set of mathematical equations of the form

$$\mathbf{f}(v_1, v_2, v_3, \ldots v_n) = 0$$

(where "=" denotes *equality*) into an *explicit* set of procedural statements of the form

$$v_k := \mathbf{f}(v_1, \ldots v_{k-1}, v_{k+1}, \ldots v_n)$$

(where ":=" denotes *assignment*) in which all variables are fully determined.

3.3.1 'R' Components

The relationship between effort **e**, and flow **f**, for a resistor is usually written as

$$\mathbf{e} = \mathbf{f}.R$$

where R is resistance. Electrical resistors have the familiar form, V=iR. These components are causally neutral in the sense that either $\mathbf{e} := \mathbf{f}.R$ *or* $\mathbf{f} := \mathbf{e}/R$ could be derived procedurally. Note that, although the resistor CR is known, it has no context until it is placed in a circuit and causality cannot be assigned.

3.3.2 'I' Components

Inertial (or mass[3]) components are defined by a CR between effort **e**, flow **f** and (momentum) state **p**:

$$\mathbf{f} = \mathbf{p} / I \qquad \textit{and} \qquad \frac{d}{dt}\mathbf{p} = \mathbf{e}$$

where the value of inertia is denoted by I. The most familiar example is Newton's Second Law of Motion. It is preferable to combine these relationships in order to integrate effort in order to evaluate flow. This specifies *integral causality* and the corresponding assignment is:

$$\mathbf{f} := \frac{1}{I}\int \mathbf{e}.dt$$

This means that the component can be realised as an ordinary differential equation (ODE)[4].

3.3.3 'C' Components

Capacitive components are defined by a CR between effort **e**, flow **f** and (displacement) state **x**:

$$\mathbf{e} = \mathbf{x} / C \qquad \textit{and} \qquad \frac{d}{dt}\mathbf{x} = \mathbf{f}$$

where the value of capacitance is denoted by C. A mechanical example is Hooke's Law. By combining these relationships in such a way as to integrate flow in order to evaluate effort, *integral causality* is realised by the following assignment:

$$\mathbf{e} := \frac{1}{C}\int \mathbf{f}.dt$$

---

[3] The bond graph method uses 'effort' and 'flow' variables, which leads to a mass-inductance analogy. Alternatively, the use of 'through' and 'across' variables would lead to a mass-capacitance analogy.

[4] The opposite assignment produces *differential* causality which means that the state variable is no longer independent, *i.e.* it is determined by other components and thus a set of differential/algebraic equations (DAEs) will result, which is difficult both to initialise and to solve numerically [*cf.* Mattson 1989].

3.3.4 'TF' Components

The transmission properties through a transformer are defined by the following CR:

$$\mathbf{e}_2 = n.\mathbf{e}_1 \qquad \textit{and} \qquad \mathbf{f}_1 = n.\mathbf{f}_2$$

where the subscripts '1' and '2' distinguish between the two bond connections and n is the transformer ratio. The consequence of the structure of this CR is that causality is directly propagated.

3.3.5 'GY' Components

The transmission properties through a gyrator are defined by the following CR:

$$\mathbf{e}_2 = n.\mathbf{f}_1 \qquad \textit{and} \qquad \mathbf{e}_1 = n.\mathbf{f}_2$$

where the subscripts '1' and '2' distinguish between the two bond connections and n is the gyrator ratio. The consequence of the structure of this CR that the propagation of causality is reversed.

3.3.6 '0' Components

Flow junctions are multi-port objects that combine the *flow* associated with arbitrarily many bonds and establish a common *effort*. The CR associated with a flow junction is defined by:

$$\mathbf{f}_1 + \mathbf{f}_2 + \mathbf{f}_3 + \ldots\ldots \mathbf{f}_N = 0 \qquad \textit{and} \qquad \mathbf{e}_1 = \mathbf{e}_2 = \mathbf{e}_3 = \ldots\ldots \mathbf{e}_N$$

where the subscripts 1, 2, ...N enumerate the bonds attached to a junction. For a total of N bonds, one bond will carry the *resultant* flow determined by the other N-1 bonds; also, one bond (not necessarily the same bond) will impose a *common* effort, which will be transmitted to all other bonds.

3.3.7 '1' Components

Effort junctions are multi-port objects that combine the *effort* associated with arbitrarily many bonds and establish a common *flow*. The CR associated with an effort junctions is defined by:

$$\mathbf{e}_1 + \mathbf{e}_2 + \mathbf{e}_3 + \ldots\ldots \mathbf{e}_N = 0 \qquad \textit{and} \qquad \mathbf{f}_1 = \mathbf{f}_2 = \mathbf{f}_3 = \ldots\ldots \mathbf{f}_N$$

where the subscripts 1, 2, ...N enumerate the bonds attached to a junction. For a total of N bonds, one bond will carry the *resultant* effort determined by the other N-1 bonds; also, one bond (not necessarily the same bond) will impose a *common* flow, which will be transmitted to all other bonds.

3.3.8 Modulated Components

It is possible to alter component behaviour by linking internal parameters with externally-defined values. In this way, the size (or *modulus*) of a parameter is variable and the component is said to be *modulated* [5]. Typically, this facility is provided for dissipators and transducers although, in principle, any component other than a junction could be modulated.

The relationship between effort **e**, and flow **f**, for a modulated resistor (MR) component is

$$\mathbf{e} = \mathbf{f}.R.v$$

where R is a constant resistance and v is the modulating variable. Similarly, a modulated transformer (MTF) component has the following CR:

$$\mathbf{e}_2 = v.n.\mathbf{e}_1 \qquad \textit{and} \qquad \mathbf{f}_1 = v.n.\mathbf{f}_2$$

where the subscripts '1' and '2' distinguish between the two bond connections. A modulated gyrator (MGY) component has the following CR:

$$\mathbf{e}_2 = v.n.\mathbf{f}_1 \qquad \textit{and} \qquad \mathbf{e}_1 = v.n.\mathbf{f}_2$$

where the subscripts '1' and '2' distinguish between the two bond connections.

---

[5] Note that 'modulation' is the bond graph equivalent of multiplication!

3.4 Principles of Causal Augmentation

The basic rules of causal augmentation of bond graphs is summarised diagrammatically in figure 3.4. Effort and flow causality are assigned separately except where unicausal assignment is required. The rationale behind causal augmentation has been discussed already. One pertinent comment is that junctions have arbitrarily many bonds and, although the diagram attempts to show a general situation, there are many possible permutations. For instance, as shown in the figure, effort is imposed on a flow junction by an input bond whereas it could be imposed by any bond depending on the context.

It is the responsibility of the modeller to decide on the causality associated with port components and to resolve any conflicts that might arise because of causal propagation from elsewhere in the model. It will be the role of the bond graph method to impose integral causality on energy stores, unless the modeller decides otherwise. Dissipators have no causal preference and, thus, they are assigned in any way which is convenient. The modeller will always have the privilege to coerce the causal assignment of an object but it is strongly advised that this freedom is not exercised unless absolutely imperative.

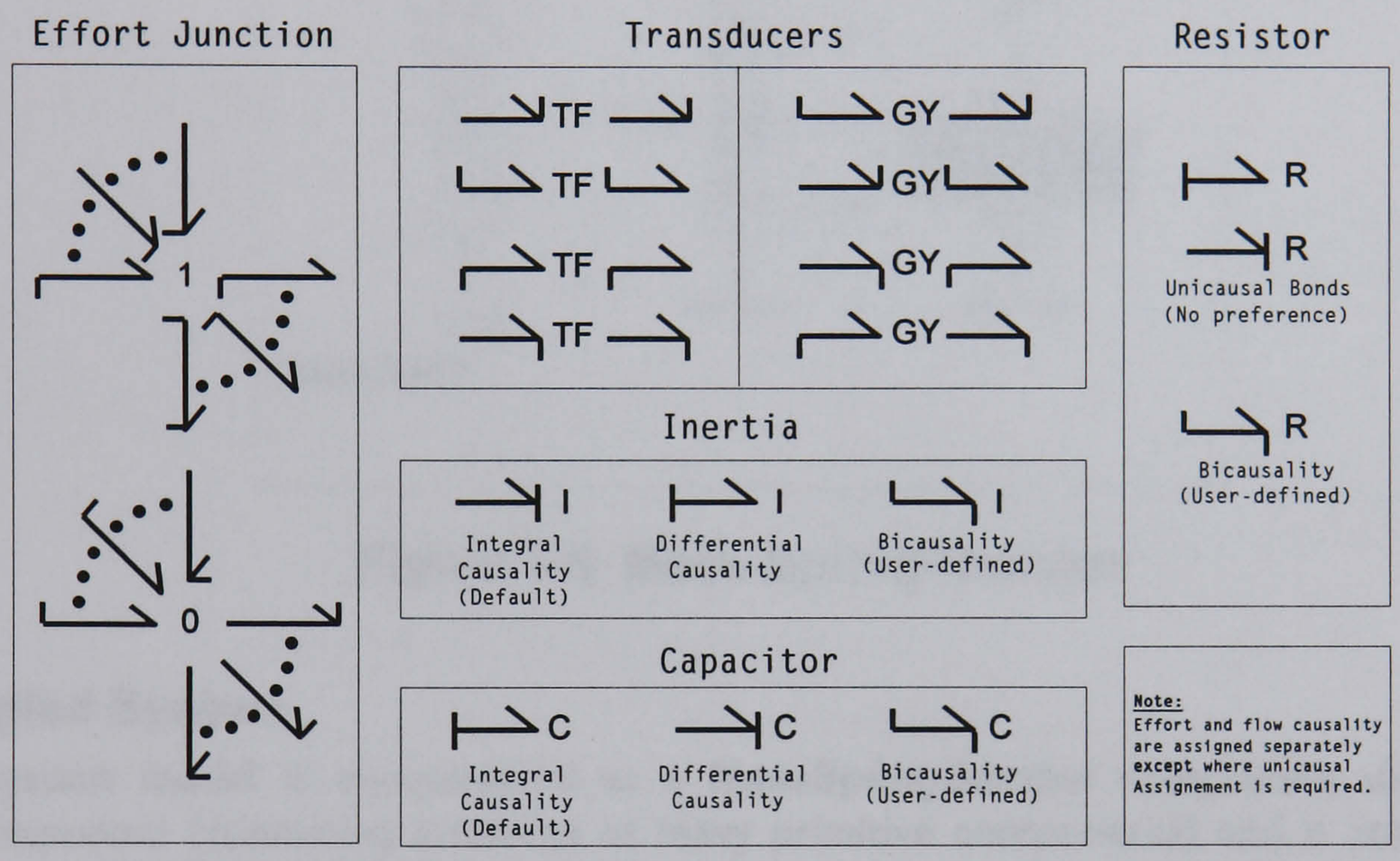


Figure 3.4 Causal Augmentation of Bond Graphs

3.5 Example

A standard textbook example is shown in figure 3.5, with a mass suspended from a rigid beam by a spring-damper mechanism and acted upon by a force (with is measured *positive* in the downwards direction). The position x of the mass is measured from its datum level when at rest.

3.5.1 Simple System

The bond graph model is drawn adjacent to the system schematic. It has an outline of the schematic as a backdrop and the bond graph is overlaid so as to show the link between form and function. The mass has velocity **v** which is defined at the **1:v** junction; the equation of motion is contained in the **I:m** object, where m is the mass. An external force is exerted on the mass from a port, labelled as **[Force]**; the other forces are due to the spring and damper, with relevant equations referenced by the **C:k** and **R:d** objects, respectively. The suspended mass reacts against the beam, represented by a port labelled as **[Reaction]**, which imposes velocity. Assigning causality is then straightforward. In the case of a fixed beam, that velocity would be zero and, thus, the junction **1:v0** would be redundant (as would be the two **'0'** junctions).

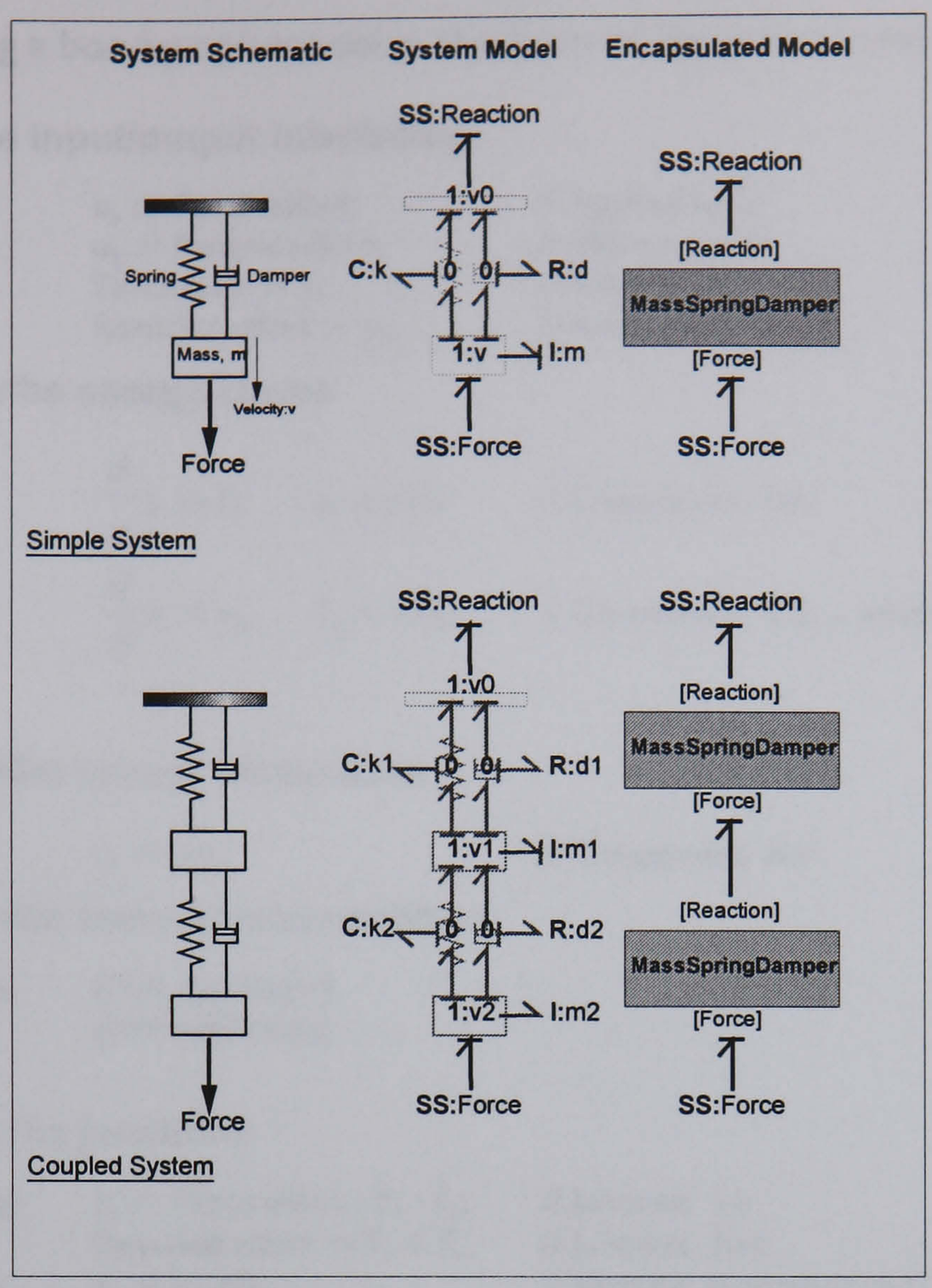


Figure 3.5 Mass-Spring-Damper

3.5.2 Coupled System

The simple system model is encapsulated as a **MassSpringDamper** component definition; this is a composite component (containing instances of many primitive components) and is accessed via **[Force]** and **[Reaction]** ports. Using this form, the creation of a coupled system is trivial, as shown in the bottom half of figure 8. The internal detail of the system model hidden; a 'flat' model becomes a hierarchical model. Two instances of **MassSpringDamper** are linked together; the function of the **1:v1** junction in the equivalent 'flat' model is performed by the **'1'** junctions adjacent to each port of the encapsulated component definition.

3.5.3 Transforming a Bond Graph into Simulation Code

The procedure for generating an algorithm from a causal bond graph is straightforward and will be illustrated here for the simple mass-spring-damper model. There are six basic steps, as follows:

1. Define input/output interfacing
2. Describe energy stores
3. Describe energy dissipators
4. Describe transducers/amplifiers
5. Describe junctions
6. Perform symbolic reduction

The first step sorts out the model interface definition. The next four steps establish 'equations' for the model components and the last step combines them into a coherent algorithm, with interfaces defined by the ports. The end result is effectively a set of state equations but, by this procedure, these can be generated automatically from the bond graph. Having demonstrated that an algorithm can be derived, this document will not offer any further explicit derivations.

The procedure for realising a bond graph model in the form of simulation code is shown as follows:

Step 1: Define input/output interfacing

Inputs	u_1 := Force.effort;	// Applied force
	u_2 := Reaction.flow;	// Beam velocity
Outputs	Force.flow := y_1;	// Velocity
	Reaction.effort := y_2;	// Reaction on Beam

Step 2: Describe energy stores

Inertia	$\frac{d}{dt}$ p := F_i;	v := p/m;	// Component I:m
Capacitance	$\frac{d}{dt}$ x := v_c;	F_c := x/K;	// Component C:k ... where K=1/k

Step 3: Describe energy dissipators

Resistance	F_r := d.v_r;	// Component R:d

Step 4: Describe transducers/amplifiers

Transformers	[Not applicable]
Gyrators	[Not applicable]

Step 5: Describe junctions

Effort junction	F_i := Force.effort - F_r - F_c;	// Junction 1:v
	Reaction.effort := F_r + F_c;	// Junction 1:v0
Flow junction	v_c := v - v0;	// Junction 0 adjacent to C component
	v_r := v - v0;	// Junction 0 adjacent to R component

Step 6: Perform symbolic reduction

States	x_1 := x;		// Spring extension
	x_2 := p;		// Momentum
Rates	$\frac{d}{dt}$ x_1 := x_2/m;	$\frac{d}{dt}$ x_2 := u_1 - x_1/K - d.v_1;	
Outputs	y_1 := x_2/m;		// Velocity
	y_2 := x_1/K + d.v_1;		// Reaction on Beam

Note that a bond carries two variables, effort e and flow f. For present purposes, a port denoted by SS:*Port*, is assumed to carry two corresponding signals, namely effort *Port*.e and flow *Port*.f. When executing this code, the interface definition for a causal bond graph is part of a function call:

```
[outputs] := BondGraphName(inputs);
```

Thus, the current example might be specified by a statement of the form:

```
[Force.flow,Reaction.effort] := MassSpringDamper(Force.effort,Reaction.flow);
```

which is compatible with any structured high-level language.

3.6 Pseudo-bond Graphs

It is possible, and sometimes desirable, to define an effort/flow domain that is not one of the options listed in **Table 3.1**. In such cases, a model is called a *pseudo*-bond graph. In general, the components become merely a shorthand for the underlying CRs and, thus, suitable constraints must be defined in order to govern their use. Examples of pseudo-bond graphs include control law definitions[6] [*cf.* Gawthrop 1995] and thermodynamic systems [to be discussed later in this chapter].

Energy might or might not be conserved in a pseudo-bond graph and so there is a need for strong typing rules in order to ensure consistency across those parts of a model where power is being explicitly transferred. The principle to be explored is to differentiate between energy domains and information domains and, in the first case, to define sub-domains in such a manner that energy is correctly accounted. Effort and flow 'co-variables' (**e**,**f**) now become effort and flow variables (ε,ϕ). Concentrating first energy conservation (with reference to **Table 3.1**), the usual issues are:

- Conservation of massflow of a compressibole hydraulic fluid, *i.e.* $\varepsilon = \rho \times \mathbf{e}$ (where ρ is density)
- Replacing Entropy Flow by Enthalpy Flow for a thermal system, *i.e.* $\phi = \mathbf{e} \times \mathbf{f}$.

This requires a fundamental change in the understanding of how bond graph interfaces operate.

The relationship between standard bond graph domains is shown in **Figure 3.6** (in black), with specialised domains for different units and scaling (in red) and for different physical mechanisms (in blue). For instance, it might be appropriate to distinguish between orders of magnitude of electrical power consumption (e.g. electronic [mW], low_power [KW] and high power [MWx100]). In general thermofluid modelling, it will certainly be necessary to distinguish between compressible and incompressible flow and to distinguish between heat transport mechanisms. These specialised domains will be user-defined and it is useful to introduce the concept of composite domains (shown in green), which allow the modeller to identify the energy components (in different forms) associated with a physical process. In **Figure 3.6**, two examples are illustrated, namely a thermodynamic domain (comprising compressible flow with convected heat flow) and a composite flow domain (comprising a mixture of chemical constituents).

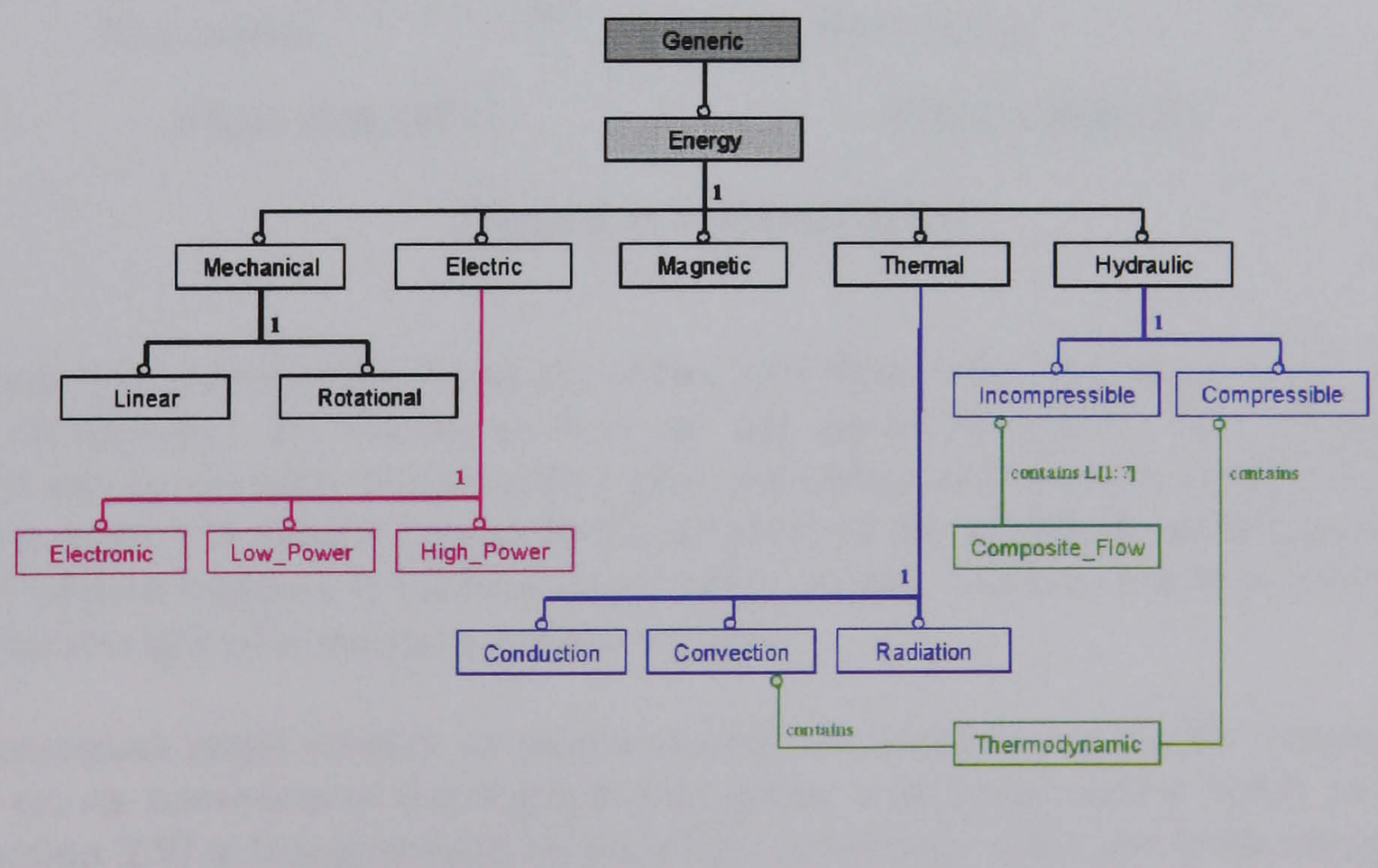


Figure 3.6 Bond Graph Domains (with possible specialisations)

[6] This is a signal processing application which traditionally and perhaps more appropriately would be handled using a signal flow graphs (or block diagrams).

Removing energy conservation allows free-form modelling, in the sense that a bond graph structure could retro-fit any set of equations, functions or empirically-defined relationships. Note also that transducers (*i.e.* **TF** and **GY** components) cease to have any real meaning. In engineering, this is unavoidable to some extent and so the aim is to ensure that pseudo-bond graphs are contributing to the production of an intuitive system model (as opposed to merely re-packaging a piece of complicated mathematics).

3.7 Amplifiers and Signals

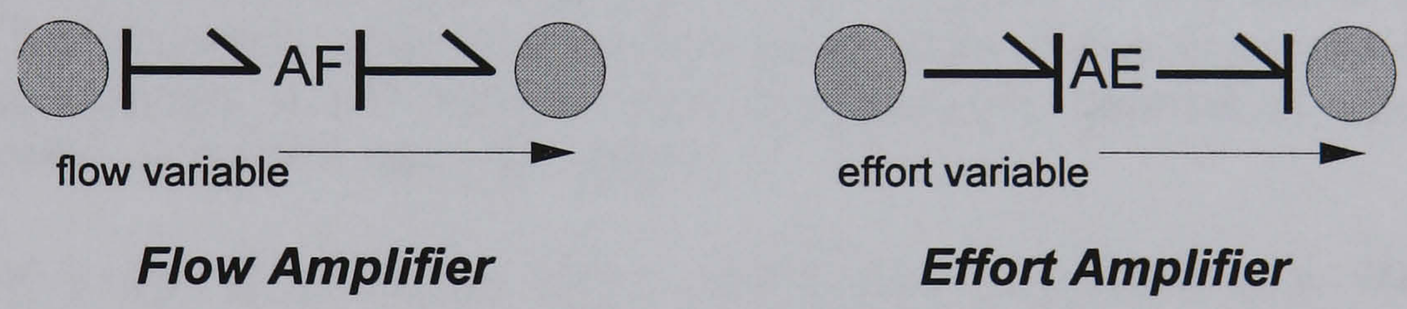
3.7.1 Definitions

If energy is not being conserved then direct amplification of effort or flow can be effected without considering the other variable. This is achieved by the introduction of two additional primitive components, as shown in Figure 3.7. A flow amplifier transmits a flow signal in the direction indicated by the orientation of the bonds: it does not pass an effort signal. An effort amplifier transmits effort in an analogous manner.

The transmission properties through effort and flow amplifiers are defined by the following CRs:

Effort Amplifier	$\mathbf{e}_2 = n.\mathbf{e}_1$	*and*	$\mathbf{f}_1 := 0$
Flow Amplifier	$\mathbf{f}_2 = n.\mathbf{f}_1$	*and*	$\mathbf{e}_1 := 0$

where the subscripts '1' and '2' distinguish between the two bond connections. Note that the causality associated with the amplified variable is propagated directly. In principle, the non-amplified variable is set to zero thereby leading to a bicausal assignment. However, in most cases, this presents an unnecessary complication and it is sufficient to constrain the 'input' side of the amplifier and to ignore the 'output' side, which allows amplifiers to be used in a standard unicausal bond graph.



Flow Amplifier ***Effort Amplifier***

Figure 3.7 Amplifiers

Amplifiers break the relationship between effort and flow and, therefore, break the interaction between model components. By definition, they do not conserve power! Accordingly to their CRs, amplifiers transmit one co-variable multiplied by a gain and constrain the other variable to be zero so that the power input is zero. The power output is the product of the amplified variable and whatever value the non-amplified variable happens to have at the amplifier output; that value is ignored and effectively the output bond can be thought of as carrying a single variable.

The need to introduce single-variable connections between components is often interpreted as a signal flow mechanism or, its conventional bond graph equivalent, a so-called active bond. In the interests of 'simplicity' [*cf.* Section 2.9] it is appropriate to introduce semantics which are easily recognisable. To this end, the *harpoon* has been replaced with an *arrow*, as shown in Figure 3.8. The principle followed here is to define a *signal bond* that is distinct from a power bond and to treat it, in effect, as a "half-bond". It only carries one variable and, therefore, causality is applied on just one half of the bond. The variable can be treated as effort or flow at the interface with power bonds but otherwise the distinction has no physical significance. Graphically, the half-causal stroke on the harpoon would be replaced by a full causal stroke on the arrow.

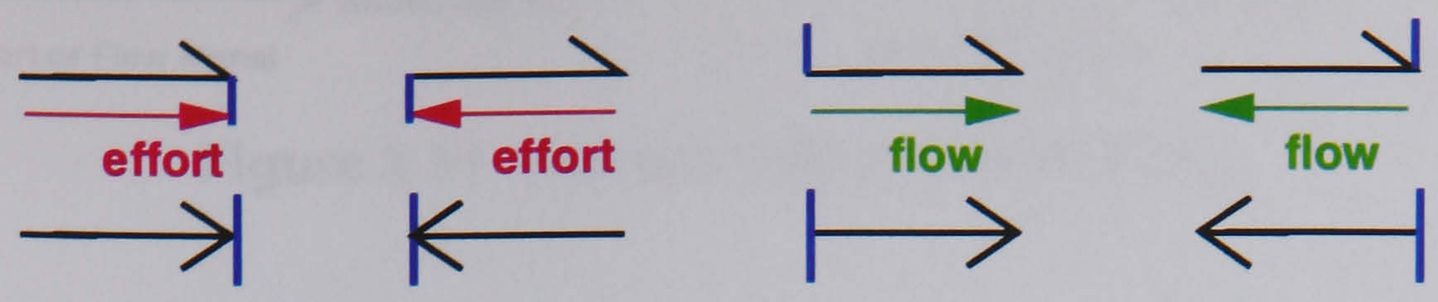


Figure 3.8 Signal Bond Notation

With this notation, it will be obvious when bonds are being used as a carrier for signal information. Also, in contrast with active bonds (at least, in the way they are often defined), it will enable causality to be reversed as part of an inversion process, by flipping the whole symbol (*i.e.* arrow plus causal stroke). What is lost is the independence between flow direction (*i.e.* causality) and bond direction (*i.e.* orientation). Under conventional notions of signal flow [cf. **Section 2.10.1**] this would be solved by introducing a sign convention at each junction. However, signal bonds can interface with power bonds in a variety of ways and thus the solution here is not so straightforward (as will be discussed in the following sub-sections).

3.7.2 Input/Output Signal Buffering

The important use of amplifiers in combination with proper (*i.e.* energy-conserving) bond graphs is to represent electrical buffers, both for actuator drive signals and for sensor measurements. Effectively, this provides an idealised interface between a 'power' domain and an 'information' domain, based on the reasonable presumption that power levels in the two domains will be orders of magnitude apart.

This is shown in **Figure 3.9** (for output buffering) and **Figure 3.10** (for input buffering), each case showing a bond drawn from a junction and 'amplified' in order to preserve only one of the bond variables (*i.e.* effort or flow). The equivalent signal notation [*cf.* **Section 3.6**] is shown in parallel. For signal output, note that causal augmentation is not required explicitly because it determined automatically by the amplifier type (for bonds) or junction type (for signals).

For signal input, it is appropriate to consider a control signal modulation to an MR component [*cf.* **Section 3.3.8**], for example, as might be used to represent a flow control valve in a hydraulic circuit. Here the bonds (and their signal equivalents) link to a component port [*cf.* **Section 3.8.2**] which is construed as a 'parameter port' because it is not a power interface. In this case, causal augmentation is optional for signals; while it may provide useful annotation, a more convenient form might be as shown in **Figure 3.11**.

Figure 3.9 Output Signal Buffering

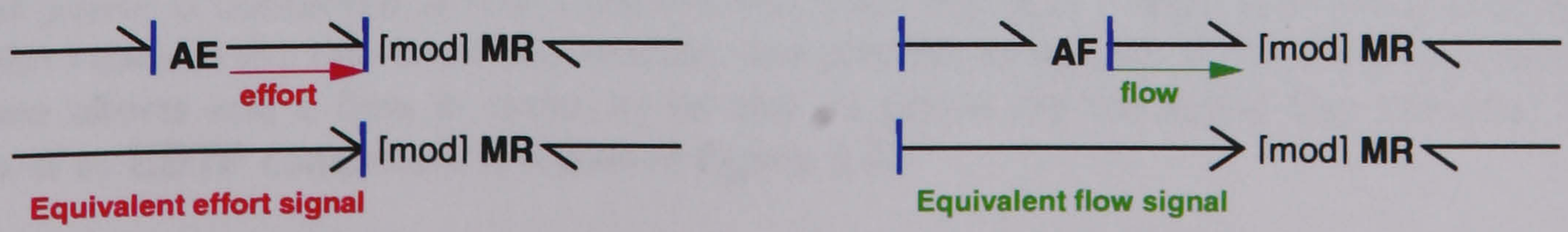


Figure 3.10 Input Signal Buffering

Figure 3.11 Generic Signal Modulation

3.7.3 Imposing Constraints

Signal buffering associated with junctions is fine provided that measurement is being *taken from* the system: it certainly is not fine if an effort constraint is being *imposed on* a system. This situation is shown in **Figure 3.12**. While the amplifier ignores the flow associated with its output the downstream components will not ignore it. In the example shown, this has the serious consequence of introducing leakage flow. Because bond graphs (as opposed to pseudo-bond graphs) are energy-conserving, the occurrence of a leakage path on the 'true bond' side of an amplifier is counter-intuitive.

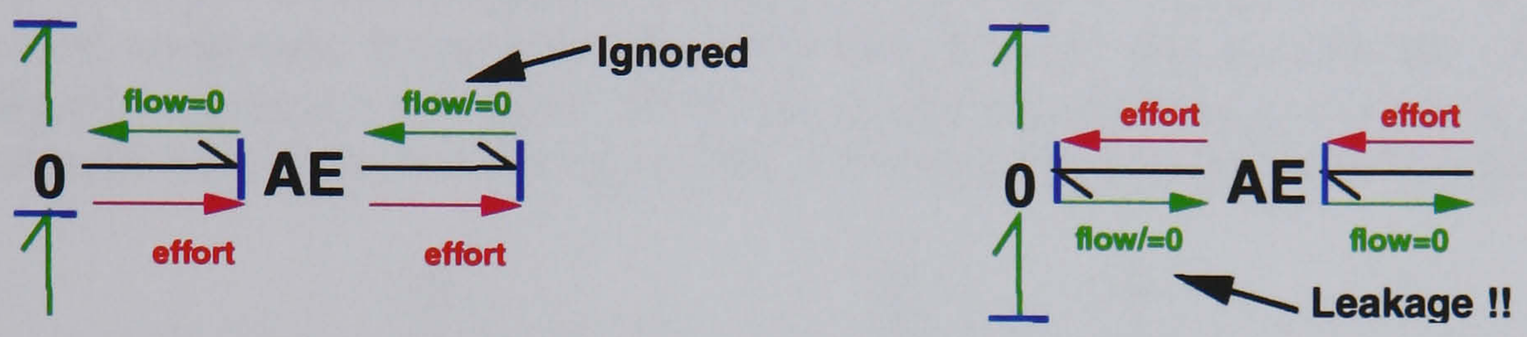


Figure 3.12 Flow Effects due to Effort Amplification

This problem can be overcome by offering alternative 'power amplifier' primitives in order to allow flow to be transmitted. These are specified as **PAE** and **PAF** components, with CRs defined as follows:

Effort Amplifier	$\mathbf{e}_2 = n.\mathbf{e}_1$	*and*	$\mathbf{f}_1 := \mathbf{f}_2$
Flow Amplifier	$\mathbf{f}_2 = n.\mathbf{f}_1$	*and*	$\mathbf{e}_1 := \mathbf{e}_2$

where the subscripts '1' and '2' distinguish between the two bond connections. Thus, amplification can be treated as any other component. Considering effort amplification, a suitably large resistance could be placed next to the amplifier, as in **Figure 3.13**, if flow were not required. This allows an imposed effort constraint with virtually no leakage. The alternative approach would be to use a standard **AE** component and to impose zero flow (without adding any extra effort!) at the amplifier output, as in **Figure 3.14**, but this introduces a bicausal bond.

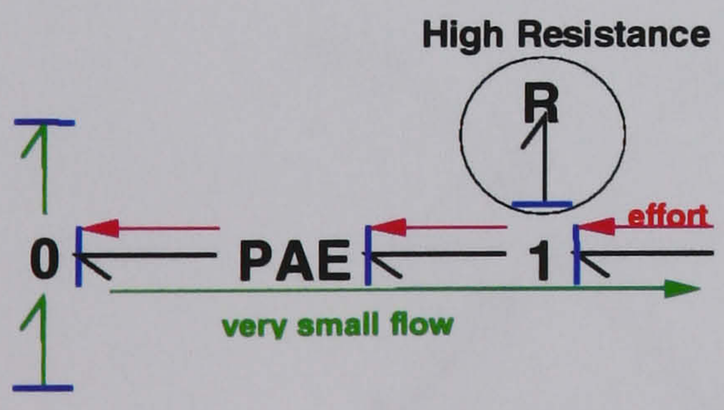


Figure 3.13 Imposed Effort Constraint

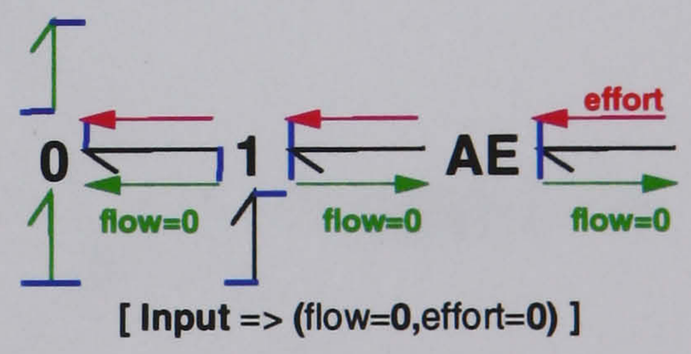


Figure 3.14 Flow-balanced Imposed Effort Constraint

As an observation, the bicausality can be removed by means of an Effort Bicausal Transformer (**EBTF**)[7]. Noting that power is conserved across a transformer, such that $\mathbf{e}_1\mathbf{f}_1 = \mathbf{e}_2\mathbf{f}_2$, where the subscripts '1' and '2' distinguish between the two bond connections, one possible projection of the CR is $\mathbf{f}_2 := (\mathbf{e}_1/\mathbf{e}_2)\,\mathbf{f}_1$. This resolves two efforts and a flow in order to be able to derive the remaining flow variable. The causal assignment of an **EBTF** component is shown in **Figure 3.23**.

[7] This idea was developed by Professor PJ Gawthrop of Glasgow University as a way of accommodating the causal constraints associated with polytropic compression and expansion in turbomachines.

Figure 3.23 Effort-Bicausal Transformer

3.7.4 Signal/Signal and Signal/Power Interfaces

In order to build (pseudo-)bond graphs[8] that incorporate signals, it is necessary to define a rigorous set of interface rules. There are two situations to be considered, namely 'signal/signal' and 'signal/power' interfacing. Remember all the time that a signal bond is not the same as a signal flow although, by slack use of terminology, each might be referred to a 'signal' in different context.

The interfaces for signal bonds and power bonds are illustrated in **Figure 3.16** for '0' and '1' junctions. The rules for causal assignment are exactly what they should be for any combination of bonds (including the bicausal outcome equivalent to **Figure 3.14**). No particular significance is attached to the harpoon directions but, clearly, the arrow direction is crucial. A few examples are shown in **Figure 3.17**.

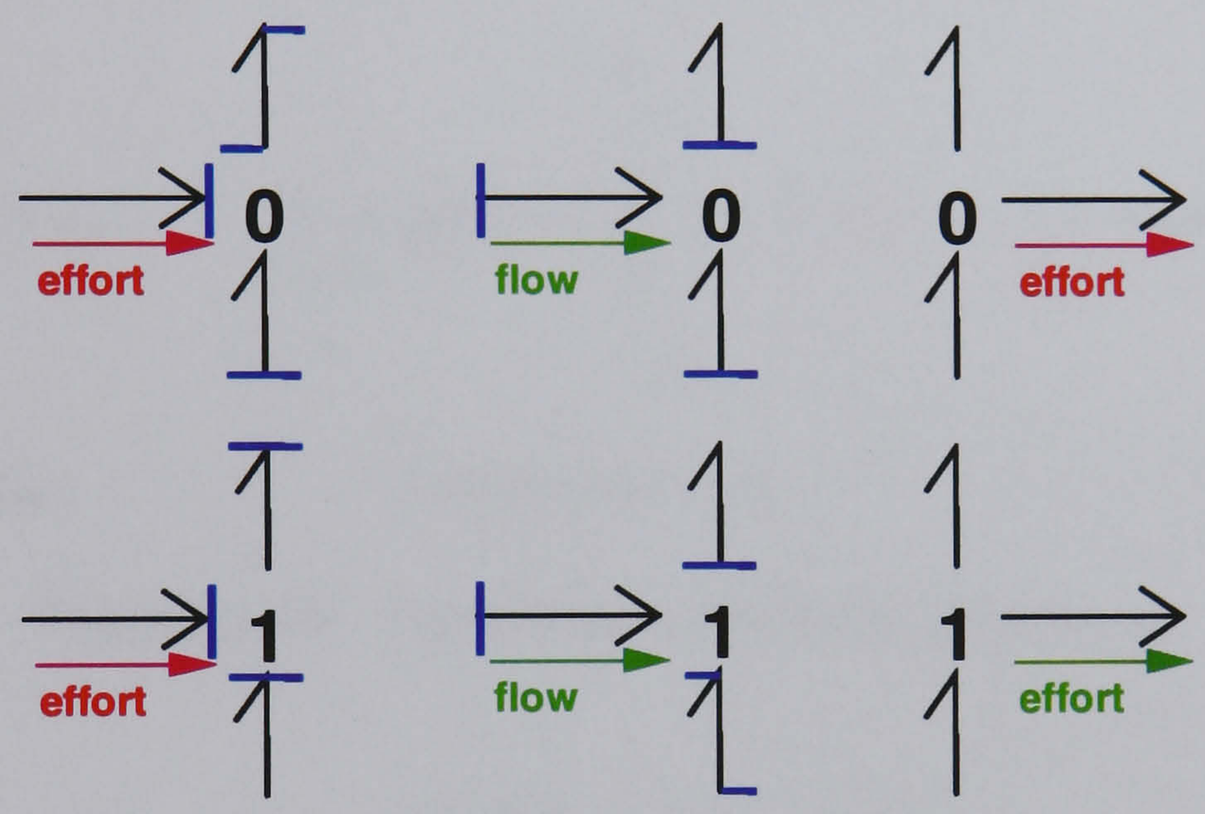


Figure 3.16 Signal/Power Interfacing

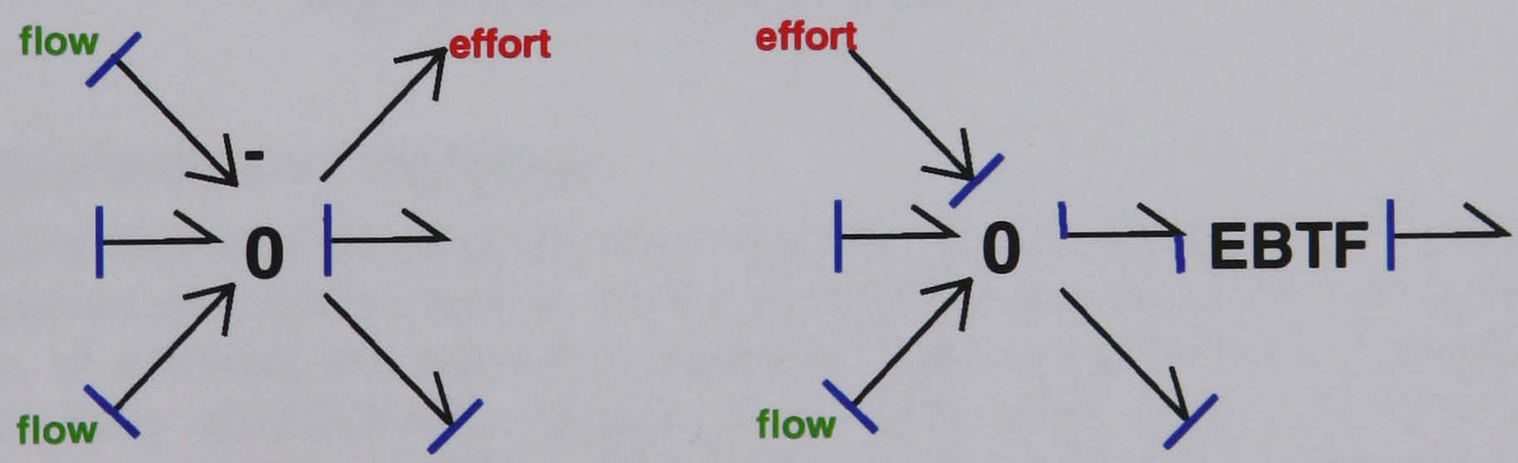


Figure 3.17 Signal/Power Interface Examples

As a useful refinement, the optional use of symbols '+' and '-' is introduced here as an intuitive methods of applying a sign convention. Note that the causal assignment need not be shown for an output signal because it is determined by context. However, causal assignment is essential for input signals! If a signal is used in order to bridge between two junction then causality must be defined for the signal input (to the destination) and not the signal output (from the source).

---

[8] For convenience, it is sufficient to discuss 'bond graphs' and 'power' in general terms. The same interface rules will apply to pseudo-bond graphs and pseudo-power.

The interfaces for signal bonds on their own are very simple and, in all key respects, follow the same rules as for signal flow diagrams (as can be seen in **Figure 3.18**). It is convenient to allow either '**0**' or '**1**' junctions to be used as summing junctions, at the discretion of the modeller. Causality is not shown because, in pure signal terms, there is no distinction between effort and flow.

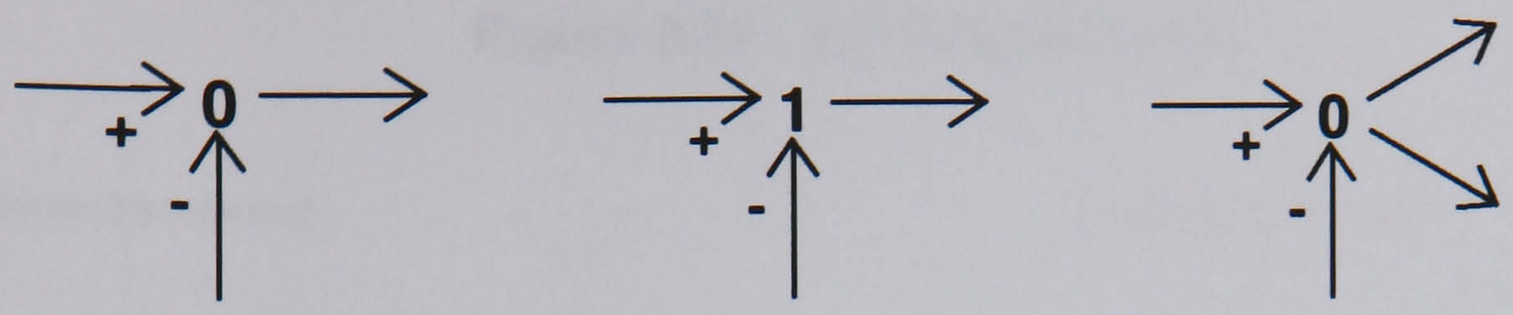
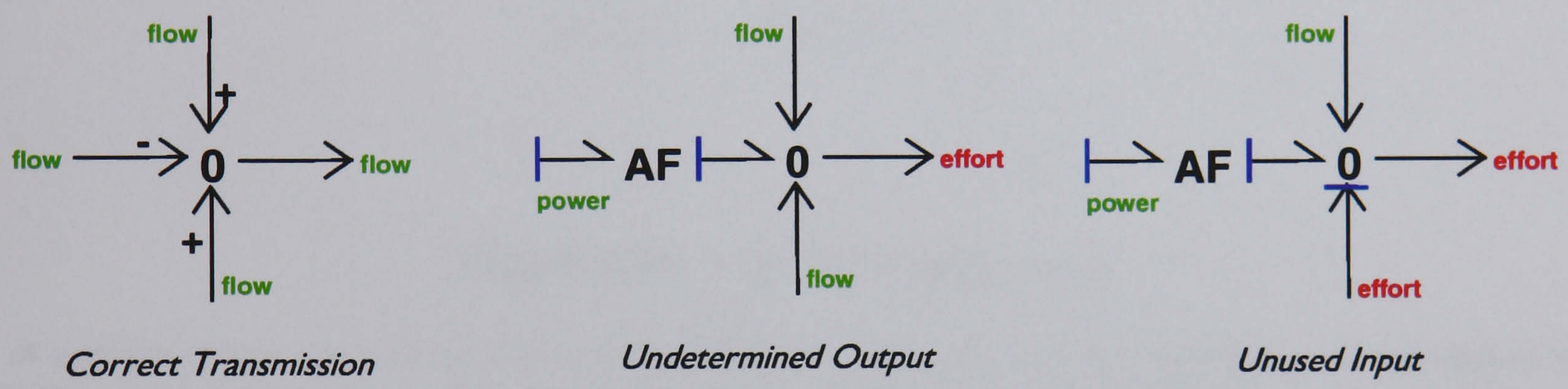


Figure 3.18 Signal/Signal Interfacing

Beware the innocent mixing of signal/power bonds in situations where amplifiers are used, such as those shown in **Figure 3.19**. This can lead to counter-intuitive situations in which a bond graph might be intuitive correct (i.e. "it looks OK!") but is functionally incorrect because of the causal constraints. The two types of exception here are called 'undetermined output' or 'unused input'. They are avoided by the defensive mechanisms defined in **Figure 3.20**, which could be encapsulated as 'Signal Buffer' components.



Correct Transmission *Undetermined Output* *Unused Input*

Figure 3.19 Incoherent Interface Examples

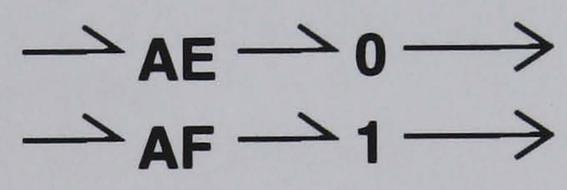


Figure 3.20 Signal Buffers

3.7.5 Logic Components and Switches

Although bond graphs deal with the physical world, they do not provide any means of applying hard constraints on the dynamics of a system, such as a vehicle colliding with a mountain! Pseudo-bond graphs do provide the means of achieving this but not in standard components. What is required is a set of components: True_or_False (BOOLEAN), Greater_Than (GT), Less_Than (LT) and On_or_Off (SWITCH). These are shown in **Figures 3.21** to **3.24** inclusive. The basic facility is provided by a Logic CR (labelled as **R::Logic**) which returns an output of one if the input is greater than zero: otherwise it returns zero.

Figure 3.21 BOOLEAN Components

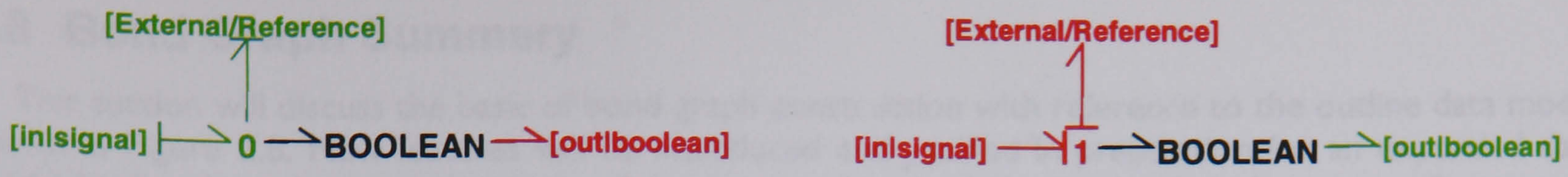


Figure 3.22 GT Components

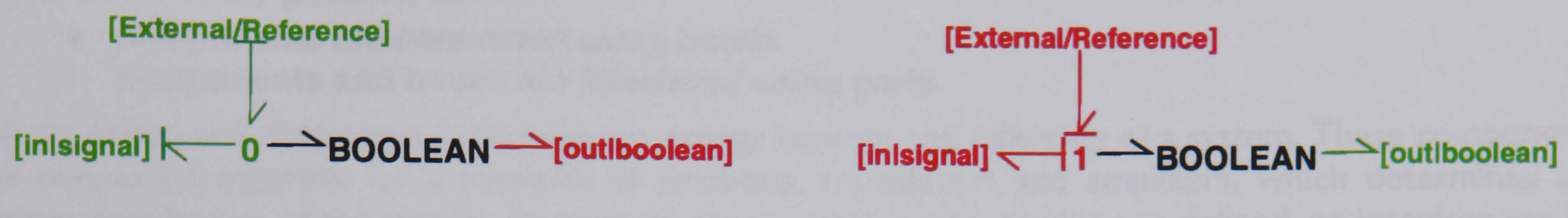


Figure 3.23 LT Components

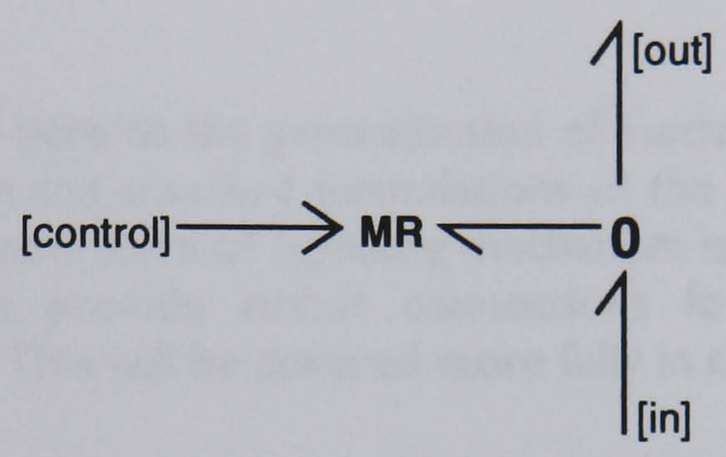


Figure 3.24 SWITCH Component

A number of features and comments are significant in these component definitions, as a motivation for changing and extending standard bond graph concepts. These are listed as follows:

1. There may be more than one way to implement a particular function [cf. **BOOLEAN**] depending on the available interfaces.
2. It may be necessary to allow component connections that are either signal bonds or power bonds.
3. A modulating signal [cf. **SWITCH**] can be supplied by a signal bond but, equally, it could be supplied by one of the variables on a power bond.
4. Optional names might be appropriate for interface ports, as shown here by a multiple-naming convention of the form [in | signal] and so on.
5. Reserved names for interface ports, like [in] and [out], can simplify the construction of composite models by allowing bond orientation to distinguish ports (rather than necessarily having to use names).
6. It is often necessary and/or convenient to specify interface ports as being external, as in the example **[External/Reference]**, rather than to assume a connection made explicitly one level higher in the model hierarchy. In fact this principle can be generalised in such a manner as to define many externally-defined distribution paths for power/signal bonds.

With these ideas in mind, a simple example is provided in **Figure 3.25**. It is a fluid storage tank with an over-fill protection function implemented by a **SWITCH** modulated by a **LT** comparison between the tank content (i.e. its state) and an externally-defined upper limit. Under the conventions being introduced here, it is permissible to prefix **LT** with **'?'** in order to signify a boolean operator.

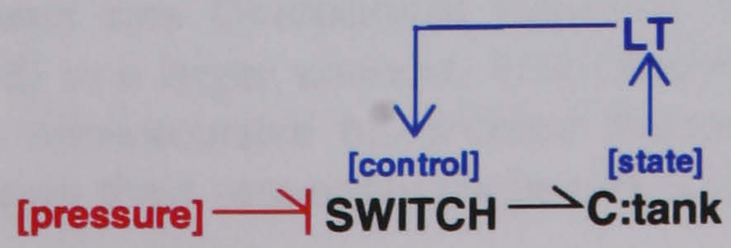


Figure 3.25 Example: Tank Over-fill Protection

3.8 Bond Graph Summary

This section will discuss the basic of bond graph construction with reference to the outline data model shown in Figure 2.8. New features will be introduced and justified in preparation for an expanded data model for bond graph concepts, which will be presented in the following section. As an overall summary, the generic structure of a bond graph model is shown in figure 3.26 (assuming integral causality). Bond graph models are constructed using three types of object, *i.e.* components, ports and bonds. In the construction of any given model:

- **components are *connected* using bonds**
- **components and bonds are *interfaced* using ports**

Energy stores and dissipators determine the energy content and efficiency of a system. These components are connected together by a network of junctions, transducers and amplifiers, which determines the energy distribution of the system. In cases where an actual component is not defined, an interface can be defined which is a place-holder for a set of possible components all with compatible port definitions, compatible with the principle of polymorphic modelling. Once a component is in place, it is said to *implement* the interface.

Particular attention has been paid here to the generalisation of model 'ports', with the aim to provide a wider range of options than exists in the standard formulations of the bond graph method. Power ports are the baseline concept and often some form of signalling mechanism is provided also. What is new is the concept of 'through' ports, which provide circuit connections for power components and loop connections for signal components. This will be covered more fully in the following discussion.

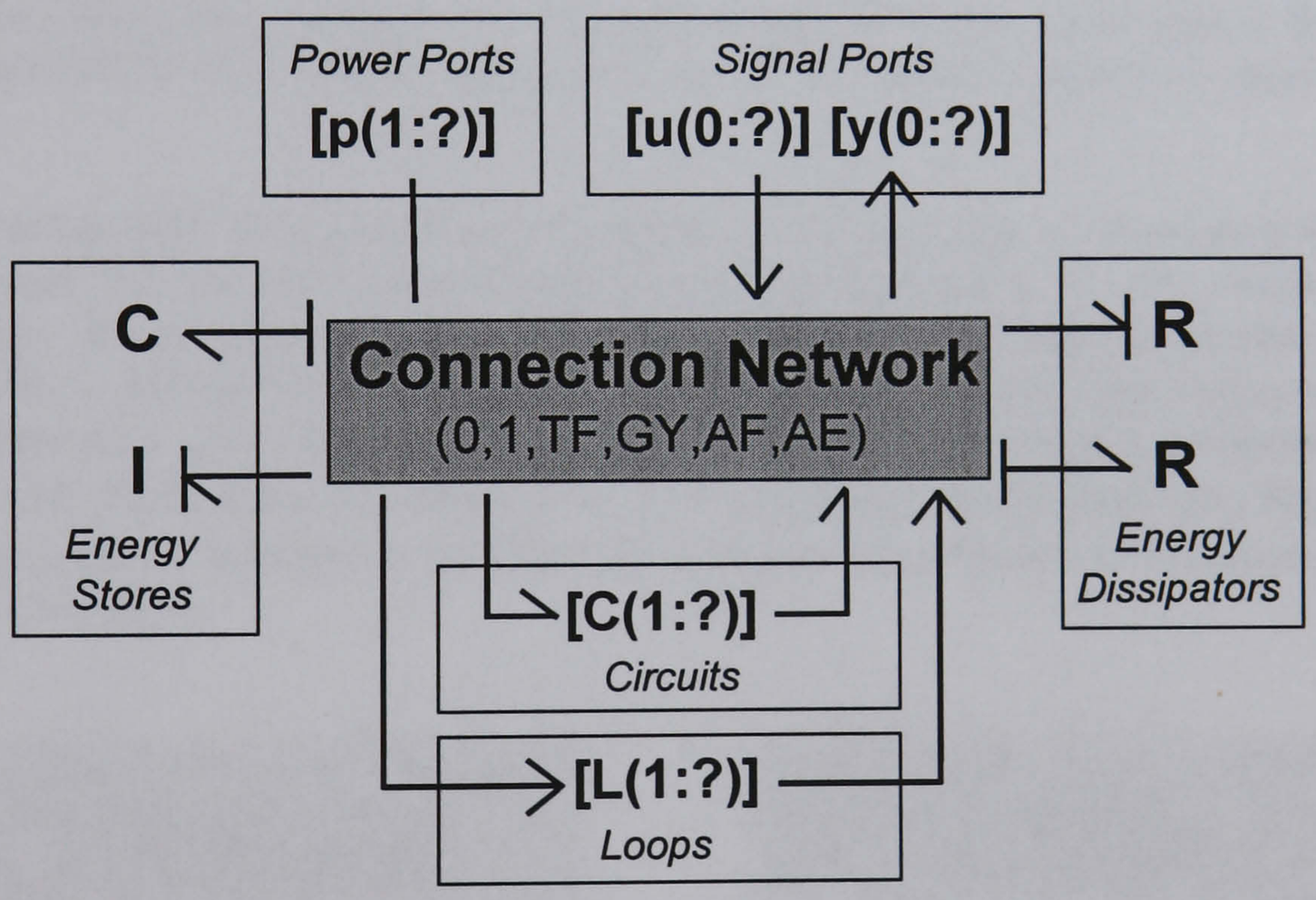


Figure 3.26 Generic Model Structure

With specific reference to Figure 2.8, recall that a **Model** (a system model!) is considered to be a **Component_Definition**. This, in turn, is either primitive or composite. Primitive component definitions contain a **Constitutive_Relationship** that describes the behaviour of the component; composite components contains contain at least one **Component_Instance**, which means that the component definition been used (or instantiated) in a larger context. This circular dependency between component definitions and instances implies a non-recursive hierarchical decomposition. Ports are designated as either *formal* or *actual*, depending upon their respective usage with a component *definition* or *instance*.

3.8.1 Components and Bonds

A component will normally be designated by a *label* of the form `ClassName:ComponentName`. Component names are optional in a bond graph drawing but all components have a unique identity. A *composite* component is a complete bond graph model in its own right; a *primitive* component contains a CR which can be defined in one of a number of forms, namely mathematical, numerical or functional (*i.e.* algorithmic). Given that bond graphs support non-causal modelling, it is usual to think of a CR as a set of equations.

A CR inter-relates bond variables, signal variables and internal parameters. The default CR will be linear but any form could be created. Component labels can optionally be extended to specify a CR, as is `ClassName::CRName:ComponentName` or `ClassName:ComponentName::CRName`. As a convenient refinement, for primitive components that are characterised by a single parameters, the label may be used in order to assign a numerical value. In its simplest form, this would be written in the form `ClassName:=ParameterValue` although component and CR names could be included also.

Note that the use of a class name together with a specified CR means that the propagation rules (for orientation, causality and so on) will be inherited from the class definition. The new CR has to conform to the interface definition for that class. In cases where special relationships are required, the modeller has the option of defining either a new class or a one-off CR. In the first case, conventional labelling will apply: in the second, the class name will be replaced by `#CRName`.

The basic *connection* mechanism is a **bond**, which conventionally is used to form a power connection between labelled *interactive* ports. Where bonds are connected directly to a component, the interfacing will be achieved via one or two implicit ports, consistent with the rules stated above. Bonds also support single-variable connections with labelled *information* ports, as well as component parameters, states and state derivatives.

Encapsulation is the basis of polymorphic modelling [*cf.* section 2.8], as illustrated for components in **Figure 2.12**. Associations between components are created by means of links which 'plug' into ports located on the edges of the component icons. Each port is drawn as a diamond, partitioned into a shaded region which signifies a *formal* port of a component definition and an unshaded region which signifies the binding to an *actual* port of a component instance. The basic ideas behind component definition are shown in **Figure 3.27**. A *primitive* component definition is a constitutive relationship $\mathbf{f(x,p,\theta,t)}$ with states **x**, port variables **p**, internal variables θ and time **t**. A *composite* component definition is a model (*e.g.* a bond graph) in its own right.

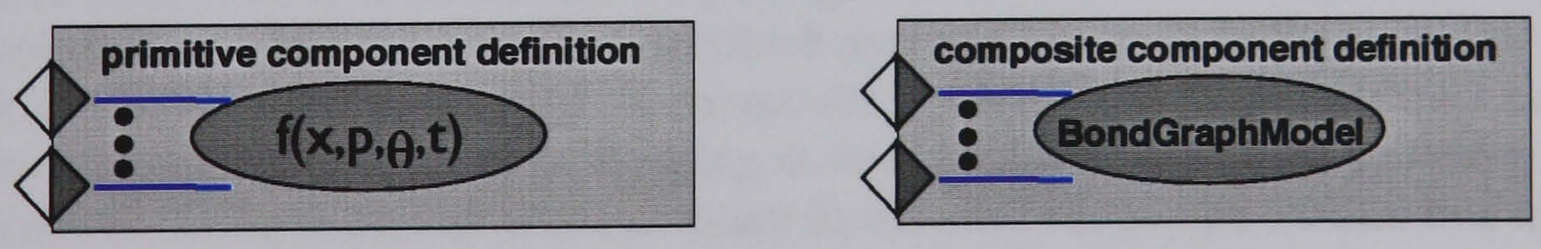


Figure 3.27 Standard Component Definitions

A hierarchical decomposition allows individual elements of a model to be defined by other models. There is clear distinction drawn between what a model contains (model *definition*) and how it is used (model *instance*). One model can contain instances of other models, expressed in component form (perhaps as an icon if the model is defined graphically or as a function call if it is defined procedurally). It is important to distinguish between simple, or *primitive*, components and *composite* components; primitives are irreducible elements of whatever notation is being applied whereas composites are themselves constructed from other components, either primitive or composite.

A simple example of component definitions is shown in **Figure 3.28**, related to basic fluid transfer and storage mechanisms. These contain **R** and **C** components together with appropriate junctions. The encapsulation is indicated by an icon boundary drawn around a bond graph fragment with ports located on the boundary. Adopting these component definitions, the corresponding component instances are shown in **Figure 3.29**. Bonds are attached to actual ports as indicated on the icon diagram.

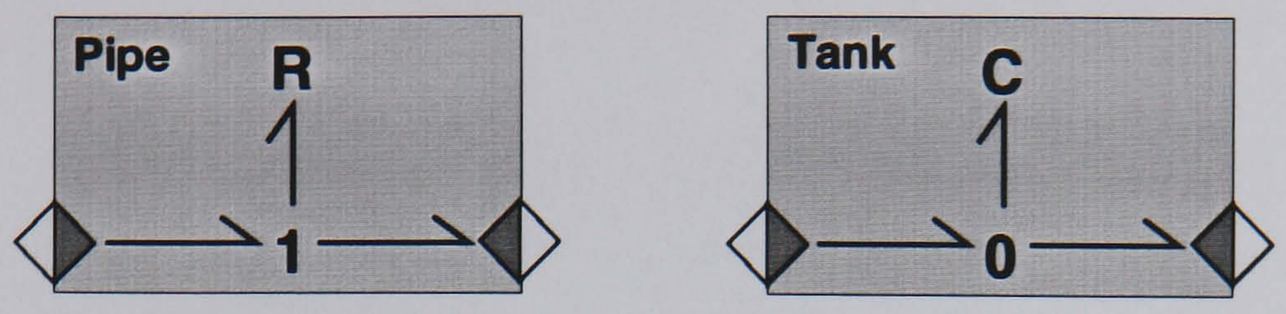


Figure 3.28 Example of Component Definitions

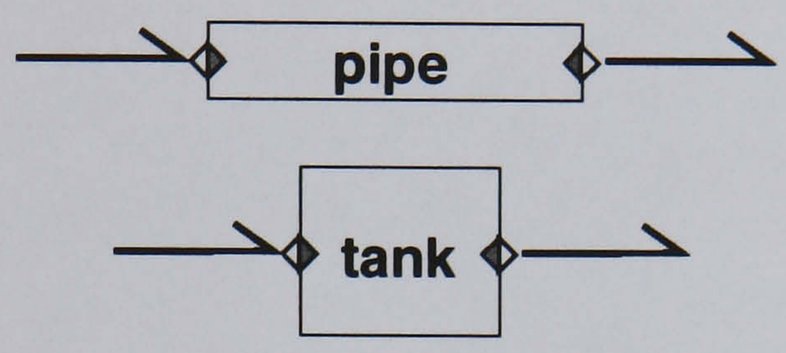


Figure 3.29 Example of Component Instances

With these elements, it is a simple matter to build a composite model of fluid transfer into and out of a tank, as in **Figure 3.30**. This is hierarchical in the sense that the top-level model decomposes into three elements, which are instances of two component definitions. There is a single instance of **tank** and two instances of **pipe** although the pipes are not distinguished by name.

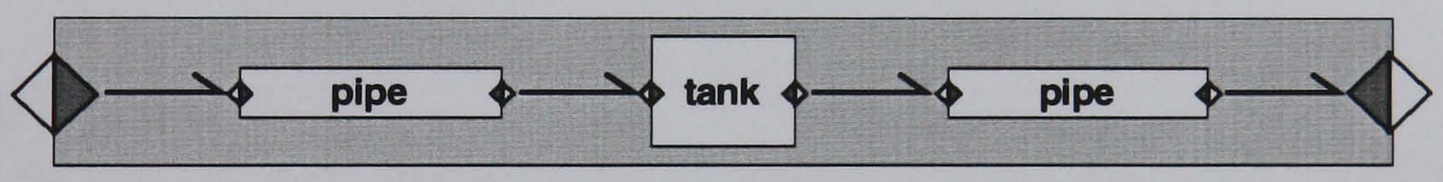


Figure 3.30 Simple Hierarchical Model

For purposes of definition, it is convenient to package bonds in a similar manner to components. This is shown in **Figure 3.31** in two forms, namely primitive and composite. In most cases, for simple models, primitives bonds would be used as standard. A composite bond is introduced as a new concept in order to facilitate more sophisticated links, by embedding in-line components (either primitive or composite). This can be effective in hiding incidental point-to-point functionality (*e.g.* pipe friction in a hydraulic circuit) and in using more detailed models without having to rebuild the top-level model with additional components. The obvious restrictions which must be placed on this facility are that bonds can only ever connect two components and, to remain intuitive, their external connections must belong to the same domain. Composite links must not be used to hide arbitrarily complex functionality.

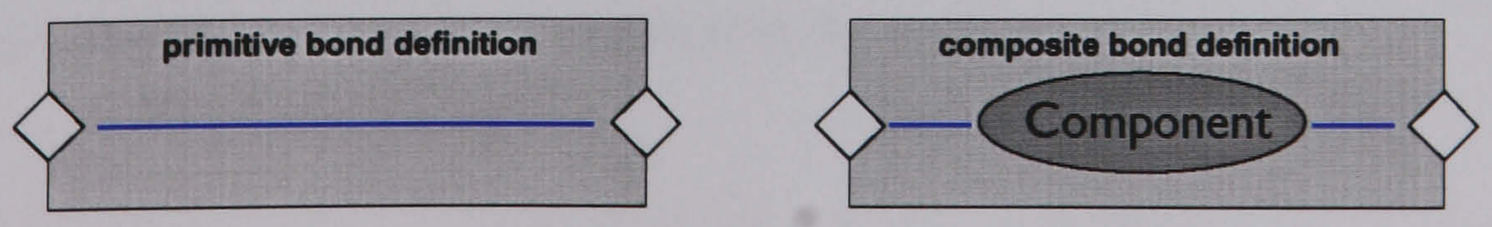


Figure 3.31 Standard Bond Definitions

The hierarchical model shown in Figure 3.30 can be re-built using composite bonds in order to hide the pipe components. Each bond now carries a solid rectangle, which indicates that there is a component embedded inside. It is envisaged that icons and names would be introduced so as to indicate readily the role being played by the composite bond. This is shown in Figure 3.32 and, while it might appear trivial to bury one pipe component in this way, it should be noted that pipes may be 'long' compound objects (comprising numerous straights and bends) and the facility to bury detail of that type might be highly valuable.

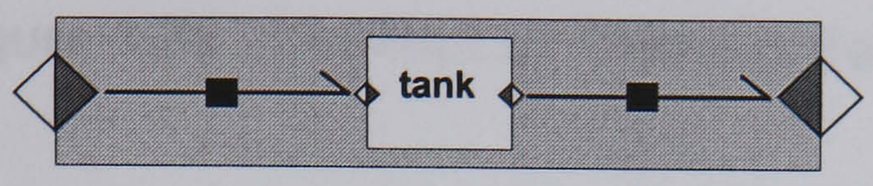


Figure 3.32 Simple Hierarchical Model using Composite Bonds

As a more general argument in favour of composite bonds, it is considered important to be able to define links with built-in behaviour that would be taken for granted in particular bond graph domains (especially composite domains) [*cf.* Figure 3.6]. While it is is always true that the same effect could be achieved explicitly with in-line components, attempting to achieve this on a wide scale would be tedious and would obscure the content of a model.

3.8.2 Ports

Ports provide the means of transferring power across a system boundary. In the particular notation being defined here, a port is a named object enclosed by brackets, *i.e.* a 'label'. This generalises the concepts of 'effort source' and 'flow source' denoted by Se and Sf respectively, which appear in the standard bond graph formulation. It replaces the concept of a 'source/sensor', denoted by SS, which appears in the bond graph implementation contained in MTT[9]. These concepts are, in effect, port *components* but this conflicts with the definition of a model as summarised by Figure 2.8. In that context, a port 'component' is strictly a **Formal_Port**, *i.e.* a port declared in the definition of a **Model**.

3.8.2.1 'Through' Ports

The concept of a port is motivated by the need to encapsulate component definitions so that they can be used (as component instances) within a hierarchical model. It acts as a 'stub', *i.e.* somewhere for power to come from or go to (by magic!). Modelling notations treat stubs as single-sided. This is not necessarily always the best approach, especially where circuit layouts are being modelled. Here, in-line components have two ports, which means that a double-sided stub would be a natural concept to adopt. For a component, this would provide a placeholder for components (supporting inter-changeability) as well as an opportunity to separate a network model from its embedded component models.

For these reasons, the concept of a *through*-port is introduced in this thesis and this is believed to be an entirely new feature for a modelling notation of this type. This is illustrated in Figure 3.33 for a simple hydraulic power system. An external source provides shaft power to a pump and oil flows around a circuit; the basic circuit is open but is closed by the insertion of a component, in this case an actuator. The diagram is drawn in such as manner as to suggest that components can be readily interchanged and, thus, it is useful to be able to define the two in-line ports by a single through-port with two bond connections (one oriented to point inwards and the other to point outwards). This idea is illustrated in Figure 3.34 using a double port icon (two overlapping diamonds).

[9] Model Transformation Tools [http://www.mech.gla.ac.uk/~peterg/software/MTT]

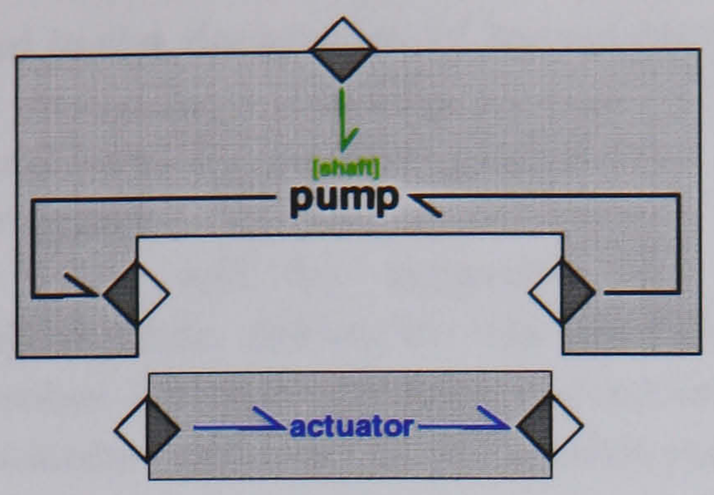


Figure 3.33 Concept of Through Ports

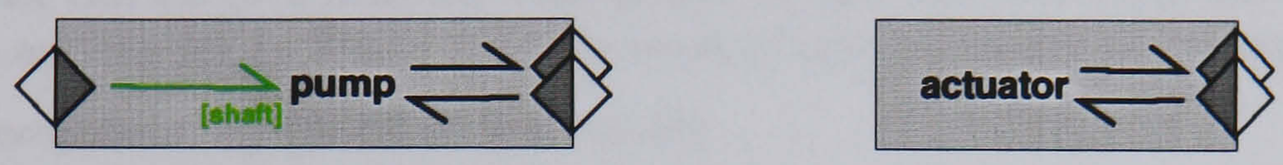


Figure 3.34 Component Definition based on Through Ports

3.8.2.2 Port Orientation

The orientation of a bond (*i.e.* the direction of the harpoon) shows the assumed direction of positive power transfer. It is, as much as anything else, a visual aid to interpret how a system functions but it does serve an essential purpose of establishing the sign convention for effort and flow junctions. Orientation of a port is defined by the orientation of the bond(s) attached to it.

Orientation does not have to be fully defined in order for a model definition to be valid. In fact, in many situations (especially where reusability is an issue), it is desirable that this should be case so that the component definition does not unnecessarily constrain orientation associated with ports; orientation can then be established locally in the context of each component instance.

Any port will be distinguished as a *null-port*, *in-port*, *out-port* or *through-port*, determined by the orientation of the bonds attached to a component instance. A through-port with a given identifier is equivalent to an in-port and an out-port with the same identifier. Note that power flowing into a component will be indicated by a harpoon pointing in towards a port of the component instance whereas the matching harpoon[10] points away from the equivalent port within the component definition.

3.8.2.3 Labelling

Implicit referencing of ports means an external connection without a label. Formally, this is established by applying reserved words for an in-port (**in**), an out-port (**out**) and a through-port (**thru**) in the component definition. It is convenient to call these *implicit* ports. A model can contain one implicit in-port and/or one implicit out-port; alternatively, it can contain one implicit through-port. The use of labels can be dispensed with here because the bond connections can be identified unambiguously by orientation.

This principle can be extended in order to remove labels in situations where bond connectivity should be obvious, by using bond properties such as energy domain. An example of this is shown in Section 4.6.7.1, where formal ports are referenced separately in the *Electric* and *Magnetic* domains; the port labels take the form **[in@Magnetic]**, **[out@Electric]** and so on.

---

[10] A bond attached to a component instance will be matched with a bond in the component definition via a 'Port_Binding' mechanism, as shown in **Figure 2.8.**

Optional names will be permitted in the declaration of formal ports, so that one name can be selected from many in order to suite the particular modelling context. The set of names is specified within brackets and are separated by vertical bars, *e.g.* **[in|fluid|gas]**. Actual ports will then be labelled using only one of the names, *i.e.* **[in]**, **[fluid]** or **[gas]**.

Component parameters and states will be accessible via ports, with labels of the form **[ParameterName]** or **[StateName]**. A state derivative will be referenced in the form **[StateName']**, **[StateName'']** where the *prime* symbol denotes the time derivative, or **[StateName^Index]** where the index shows the number of differentiations required of the system state.

3.8.2.4 Vectored Ports

Port declarations can be vectorised[11] by augmenting the port names with an index enclosed within parentheses. The index can be of a discrete expression or any discrete class but typically it would be a positive integer or an enumeration. There are four ways of using a vector port, namely:

- one vector component, *e.g.* **[port]** or **[port(1..6)]**
- subvector components, *e.g.* **[port(1,3,5)]**
- individual components, *e.g.* **[port(1)]**, **[port(2)]**
- mixed individual/subvector components, *e.g.* **[port(1,3..6)]**

The labelling of formal ports and actual ports does not need to have the same partitions. For example, formal ports **[port(1..2)]** and **[port(3..4)]** could be bound to actual ports **[port(1,3)]** and **[port(2,4)]**. What is important is that all ports are correctly accounted, even if some are to be unconnected. Also, if a port is to carry more than one bond then the labelling (in the definition, instance or both) must clearly identify this; if the number of bonds in the vector is unknown then a question mark should be used in order to denote the unknown index, *e.g.* **[port(1..?)]**. Note that, rising up through a hierarchical model, vectored ports could be attached to bonds that are interfaced structured ports [*cf.* Section 3.8.2.5].

3.8.2.5 Structured Ports

Port declarations can be structured[12] by augmenting the port names with *field* names, separated using a point '.' (in common with the concept of a data structure in any high-order language). The purpose of a so-called structured bond is to create a hierarchy rather than just an indexed list [*cf.* Section 3.8.2.4], thereby indicating in a more clear way the role being played by particular ports and bonds. There are three ways of declaring a structured port, namely:

- single label, *e.g.* [Control]
- individual labels, *e.g.* [Control.Fuel]
- multiple labels, *e.g.* [Control.<NozzleArea,IGV>]
- vectored labels, *e.g.* [Offtake.AirBleed(1..3)]

Note that, rising up through a hierarchical model, structured ports could be attached to bonds that are interfaced via vectored or structured ports.

3.8.3 Port Binding

'Port Binding' is the mechanism whereby the interface attributes of a component *definition* are transferred to any *instance* of that component when it is used in a model. The whole issue of component instantiation means that the contents of a component definition are copied into the higher-level model structure and then component attributes are allowed to propagate along the inter-component bonds. To this end, the component *ports* play a critical role in the classification of modular components and the achievement of model consistency.

---

[11] Implicit referencing is valid for vectored ports using reserved words in, out and thru, in conjunction with indices.

[12] Implicit referencing is valid for structured ports using reserved words in, out and thru, in conjunction with structure fields.

From the preceding discussion in this chapter, there are six categories of attribute to be 'bound' between *formal* ports (of a component definiton) and *actual* ports (of a component instance), namely:

- Link, *i.e.* power-power, signal-power, signal-signal
- Orientation, *i.e.* direction of information flow or (positive) power flow
- Causality, *i.e.* directions of effort/flow propagation
- Multiplicity, *i.e.* vectored bonds, structured bonds
- Domain, *i.e.* definitions of effort/flow variables
- Units and Scaling, *i.e.* conventions for quantification and measurement

In all cases, the port binding involves a set of propagation rules and a set of consistency checks. Propagation means that an attribute that is defined on one 'side' of the port is copied across to the other. Consistency means that any attribute that has been defined on both 'sides' of the port has an identical value (*e.g.* length of a vectored bond), an equivalent value (*e.g.* subject to units and scaling) or a compatible value (*e.g.* domain or sub-domain), depending on what is being considered.

The first three cases involve a range of inter-mixed exceptions that must be caught during model compilation whereas the last three cases amount to little more than a straightforward matching of individual attributes. As a general principle, if an attribute is undefined then it will acquired from instance data. Constraints on Orientation[13], Causality and Link attributes are inter-related as shown in **Figure 3.35** and the exceptions that can arise (based on the bond graph notation developed in this thesis) are defined in **Figure 3.36**.

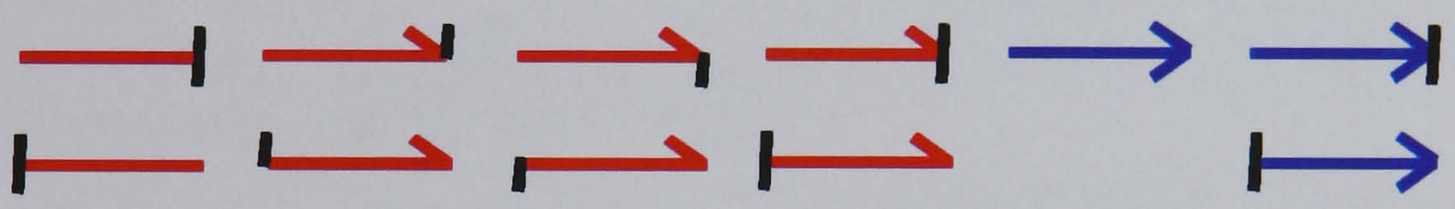


Figure 3.35 Bond Attributes for Port Binding

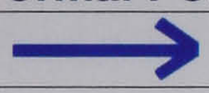
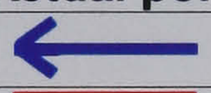
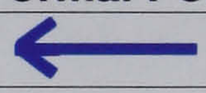
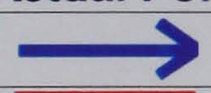
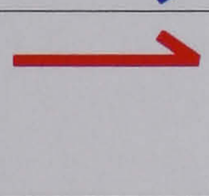
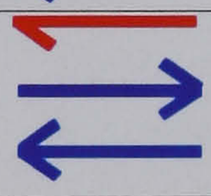
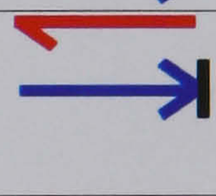
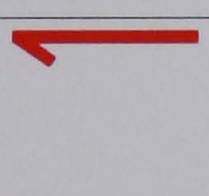
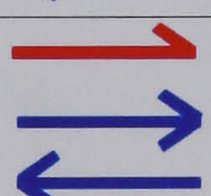
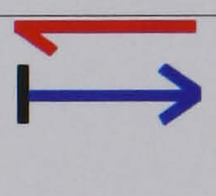
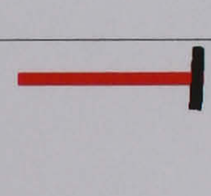
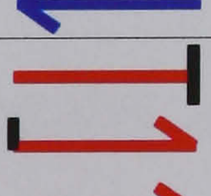
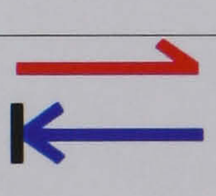
Formal Port	Actual port	Formal Port	Actual Port
			
			
			
			

Figure 3.36 Conflicts in Port Binding of Bond Attributes

[13] Propagation rules apply to orientation similar to those for causality [*cf.* Top 1994].

A specific comment is necessary for multiple bonds. Under normal circumstances, the port binding implies a direct comparison of structures or vectors. In circumstances where a multiple bond is attached to an individual component, the component itself is considered to have multiple instances that correspond to the structure/vector arrangement of the bond. Clearly, all bonds that are attached to such a component must have the same composition. Note that this conception of multiple bonds is completely different from the standard interpretation [*cf.* Breedveld 1985].

3.8.4 Algebraic Loops!

One classic problem that can arise in modelling is the creation of algebraic loops in resistive networks (*i.e.* parts of a bond graph that do not contain energy stores) [*cf.* Lorenz & Wolper 1985] . The symptom is one or more closed causal paths between **R** components, with effort and flow signals passing in opposite directions along chains of bonds, as seen in **Figure 3.37** (with 'flow' shown in green and 'effort' in red). This situation might occur when modelling the junction between two pipes. The trivial solution is to change the causality imposed from outside but this may not be possible; the adjoining components themselves may be resistive or they may be subject to causal constraints for some special reason.

One approach would be to insert an energy store (in this case, a 'stray' capacitance), as seen in **Figure 3.38**, showing a fragment of some hypothetical bond graph model. For simulation, this might generate a stiff numerical problem, especially if high fidelity is stipulated; this would prevent the use of artificially large volumes just to sort out causality because the flow characteristics would be corrupted. Signal transmission is resolved because, by inspection, it can be seen that there is no longer a closed causal path around the bond graph. Note that '!' appears as a prefix to component class name to indicate that the component is present in order to overcome an algebraic loop problem.

Another approach would be to introduce an intermediate variable and then to solve an extra equation. This would be implemented as a special interface (introduced here as an internal port, labelled as **[Internal/Loop]**), which replaces the energy store. It causal assignment must be user-defined but then the equation would ensure that its output (in this case, effort) exactly corresponds to zero input (in this case, flow); thus, the power characteristics are not disrupted by breaking the algebraic loop. However, this requires an implicit solver, which carries a computational overhead. Compromise is unavoidable in such cases.

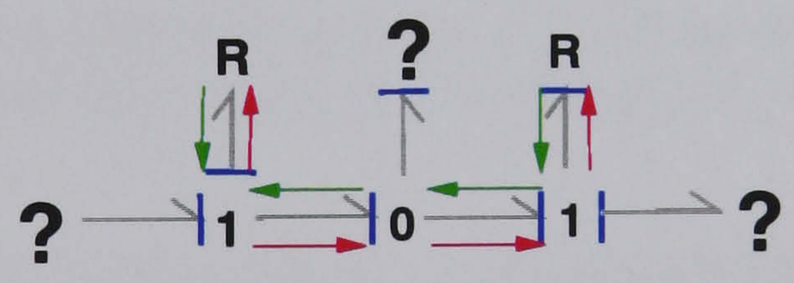


Figure 3.37 Algebraic Loop Mechanism

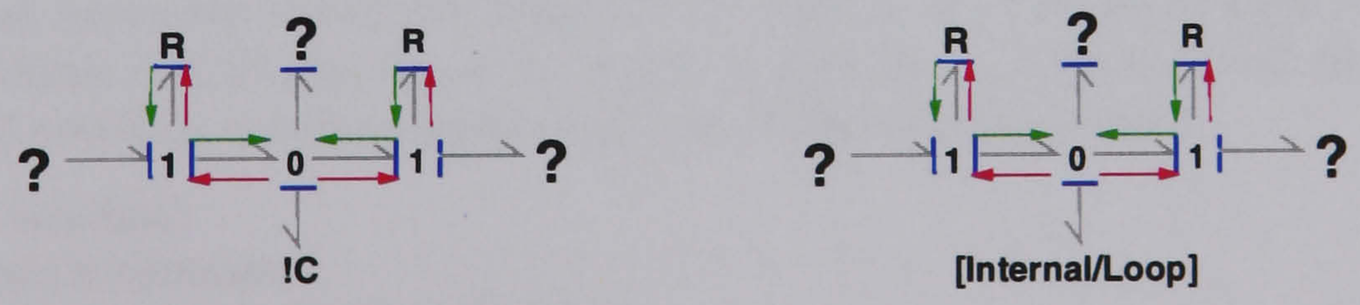


Figure 3.38 Breaking an Algebraic Loop

Notwithstanding the need to compromise, algebraic loops become a real nuisance whan they occur between composite component instances. This means that the causal factors are buried in the model hierarchy, hidden from view by virtue of component encapsulation. As a matter of good design practice, algebraic loops should be resolved within the lowest-level component definition in which they arise. Otherwise, component re-use would be compromised by algebraic 'booby traps'.

3.9 Bond Graph Superstructures

Having defined a basic structure for bond graph modelling and introduced a wide range of new features, it is possible to build models of compex, large-scale and functionally diverse systems. However, so that the issues of scale, complexity and diversity might be handled in an intuitive and efficient manner, this section will define four types of superstructure, namely Libraries, Distributions, Views and Partitions, that should help the modeller to organise and visualise large amounts of information.

3.9.1 Libraries

Support for libraries (sometimes called packages, blocksets and so on) is a standard feature in most, if not all, commercial products although, curiously, this never appears to have written into any method definition. For current purposes, the concept of a 'Library' is formally incorporated into the bond graph method. The impact on model is slight, amounting to a prefix to each component class drawn from the library. Thus, an instance of a standard component definition would be labelled as

`ClassName:ComponentName`

and an instance of library component definition would be labelled as

`LibraryName/ClassName:ComponentName`

If necessary, components can be held in a hierarchical library `Library/subLibrary/...` and so on.

3.9.2 Distributions

A specific mechanism is required for handling connections and dependencies within models which do not conform to any hierarchical decomposition. Two keys issues are:

- **connectivity from one model fragment to all other fragments of a large model**
- **dependency of one model on components defined in another model or in a library**

This is a problem when modelling the provision of a generic service like electrical power, and usually results in very large number of intermediate interfaces between providers and consumers. The solution is to use a *distributed* port, which is newly defined here as a type of global interface handled through a centralised transaction table. The use of libraries enables access to items that are (by definition) *distributed* items.

The naming convention for the a distribution point is `DistributionName/PortName`. If necessary, this can be decomposed in the form `Level1/Level2/Level3/` in order to give a multi-level distribution.

Within a fuel tank called 'Tank1', a port might be declared as **[Elec/AcPump]**; this could be referenced explicitly by a label **[AcPump]** (or **[Elec/AcPump]**) or implicitly by a label **[Elec/]** which includes all electrical ports. Within the electrical system, power could be provided from a power port or from a busbar **0**:Elec/AcBus. In the latter case, the junction itself belongs the electrical distribution and bonds can be connected to it remotely. What the **'Elec/'** prefix does is to establish an external or high-level view of the electrical interfaces distributed across an entire model. For a system of fuel tanks, numbered from 1 to 10, the electrical interface could be labelled in one of the following forms:

- [Elec/] to access all interfaces
- [Elec/Tank1] to access a submodel
- [Elec/Tank5.AcPump] to access a particular interface

If the electrical distribution were to be configured through several local nodes, say LeftWing, RightWing and Fuselage, then the labelling could also take the form [Elec/LeftWing/Tank1], in which case the **Elec/** distribution table would contain LeftWing/Tank1 as a lower-level distribution.

3.9.3 Views

The structure of a model provides the mechanism for holding components and for establishing links between them. The component definitions are concerned with functionality (in whatever form happens to be appropriate, *e.g.* equations, algorithms, bond graph, etc.) and interfacing *via* ports. The main problem that becomes apparent in practice is that, for multi-domain models, the port labelling is a real clutter.

The solution is to provide a 'View' mechanism in the construction of bond graphs. For any model, this would allow the modeller to take selective views, hiding components and bonds as appropriate. This could certainly be based on multiple objects (if any have been defined) but, in general, this should be available for any selection of bonds in the bond graph. In a gas turbine model (*cf.* Figures 5.8, 5.9, 5.10), this could be organised as follows:

Engine->Turbomachinery
Engine->FuelSystem
Engine->Cooling

where the separator '**->**' means **'from the viewpoint of'**. A combined view could be established, in which case it would be specified in the form Engine->Turbomachinery+Cooling. Views can be defined in one of five ways, namely:

[i] **hierarchical**, *e.g.* For example, Engine->GasPaths could be decomposed into

Engine->[GasPaths->[Core,Bypass,Bleed,Cooling], FuelSystem, ...]

[ii] **equivalenced**, *e.g*

Engine->Cooling = Engine->GasPaths->Cooling or
Engine->Flow = Engine->GasPaths->BypassFlow+CoreFlow

[iii] **referenced by interface**, *e.g.*

Engine->Utilities = Engine.Utilities where 'Utilities' would be a structured bond or
Engine->Elec = Engine.Elec/ where 'Elec/' would be a distributed bond

[iv] **referenced by class**, *e.g.*

Engine->Nozzles = Engine.<>'class==Nozzle
where the thing on the right-hand side is looking for any component whose class is defined as 'Nozzle'.

[v] **referenced by instance**, *e.g.*

Engine->HPSystem = HPcompressor + Combustor + HPturbine + HPspool

Views can be defined functionally in any way appropriate to the purpose of the model, *e.g.* common membership of energy domains. Essentially, what is being established is a set of representations of interesting parts of the system. This will enable the modeller to browse through a complicated model, concentrating on particular aspects rather than being confronted with everything at once.

3.9.4 Partitions

Partitioning is an essential adjunct to integration. It defines boundaries within a system which, thus, mark the interfaces between subsystem segments. Model partitioning is a useful facility for organising component groupings and an important facility for implementation and test. To these ends, partitions can be defined in one of three ways, namely:

[i] **individual grouping**, *i.e.* creating a submodel that can be compiled and exercised separately from the full model of which it is part.

[ii] **component aggregation**, *i.e.* defining tightly coupled component/bond clusters in order to encapsulate new composite components or to abstract a high-level overview of the model architecture.

[iii] **functional differentiation**, *i.e.* allocating components and bonds to functional sets that are exhaustive and exclusive, in order to be able to deal with specific attributes (*e.g.* energy domain) and specific implementation issues (*e.g.* numerical integration).

Note that a partition is a specialised form of 'view'. It may or may not have complete coverage but it must not contain intersections between partitioned sets.

3.10 Information Model for Bond Graphs

The preceding discussion has overhauled the basic ideas of bond graph modelling and has generalised and extended these in various ways. The overall aim throughout has been to prepare a comprehensive framework within which to model big engineering systems. The new concepts that have been formulated have arisen and evolved during the construction of very many system/subsystem/component models (as will be illustrated in Chapters 4 and 5). What is needed is an unified view of these concepts before moving on. The means of achieving this is an information model. This section will build up an overview of a unified information model for bond graphs. This is believed to be the first such model of its type.

3.10.1 Basic Structure

The basic information content of a bond graph (as implemented in most bond graph tools) is shown in **Figure 3.39**. A **Bond_Graph** is a **Composite_Component_Definition**; composite and primitive component definitions are subtypes of **Component_Definition**. Composite component definitions contain many component instances, each of which has its own definition. This creates a non-recursive hierarchy Because a component cannot be used before it has been defined.

The distinction between component 'definition' and 'instance' is explicit, although each can have behaviour specified by a **Constitutive_Relationship_Instance** (previously referred to as a CR, as shown in **Figure 2.8**), which has its own **Constitutive_Relationship_Definition**.

The terms 'formal' and 'actual' are used to connote ports of a component 'definition' and a component 'instance', respectively.

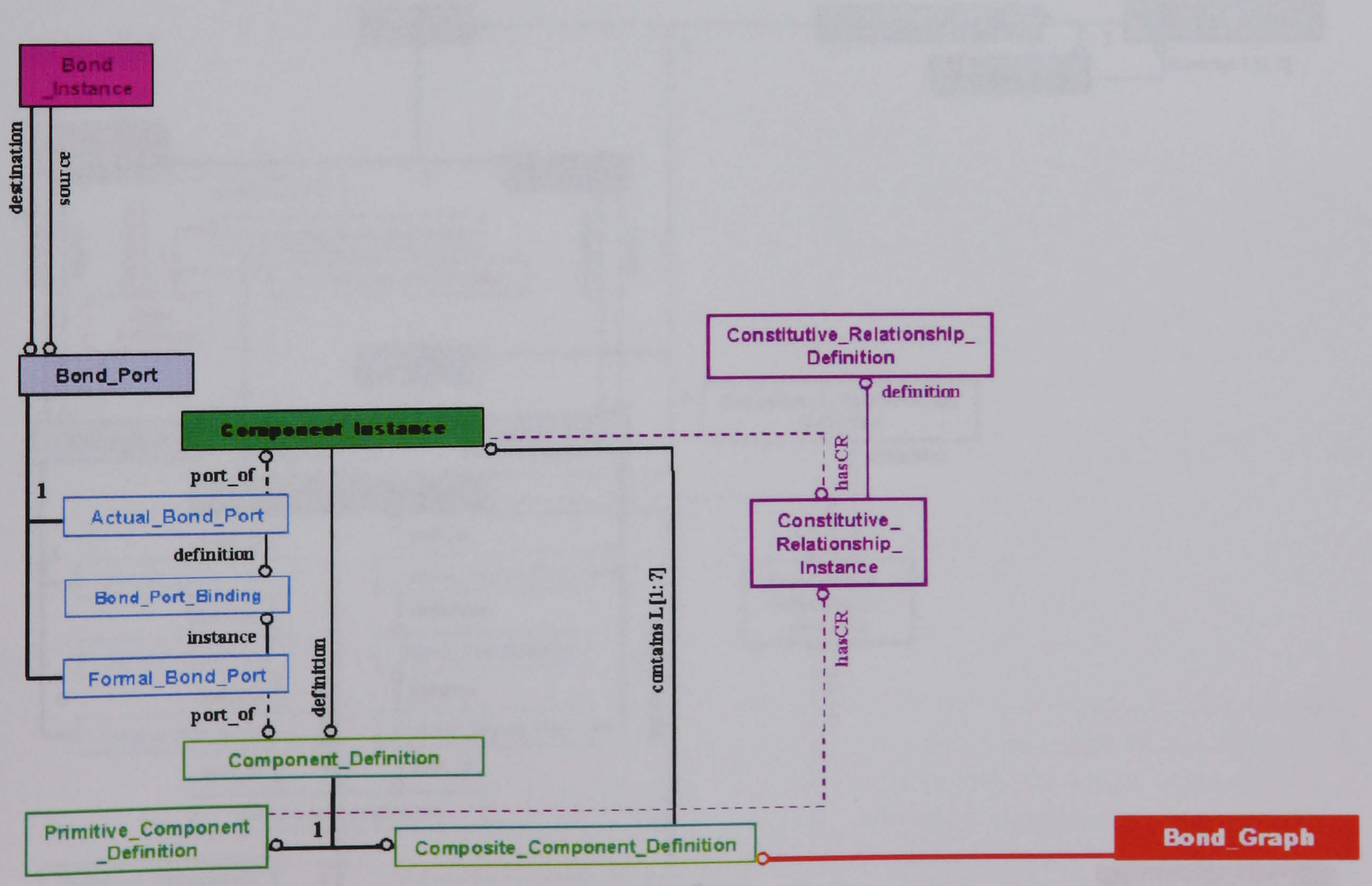


Figure 3.39 Standard Bond Graph Concepts

A **Formal_Bond_Port** is a port of a component definition while an **Actual_Bond_Port** is a port of a component instance. Note that, like CRs, these have the same content but, unlike CRs, they do not have the same significance and hence they are separate entities. A formal port occurs once (as part of a component definition) but an equivalent actual port occurs as many times as there are component instances. The **Bond_Port_Binding** provides the matching of parameters and transference of attributes, as discussed in **Section 3.8.3**.

The main model items are bonds, ports and components. More specifically, as they appear in a bond graphs, these should be designated as **Bond_Instance**, **Bond_Port** (both formal and actual) and **Component_Instance**, respectively.

3.10.2 Adding Signals, Composite Bonds and Multiple Items

The significant role of signal bonds (or signals) [*cf.* **Section 3.7**] justifies their explicit inclusion in the information model. This adds a set of entities that mirror interactive bonds. The presence of **Signal** and **Signal_Port**, in addition to **Bond_Instance** and **Bond_Port**, leads to the definition of new supertypes **Link** and **Port**. This is shown in **Figure 3.40**.

The explicit reference to a bond instance (but not signal instance) is because bonds can be defined as composite objects [*cf.* **Section 3.8.1**]. The **Bond_Definition** is a supertype of primitive and composite bond definitions. As a comparative comment, a bond is taken to be a connection between two ports whereas a signal is a directed transfer, with a source and a destination. With this additional information, the main model items are now **Link**, **Port** and **Component_Instance**, respectively. The top-level supertype is **Model_Item** and this can be decomposed progressively by **Multiple_Item**. The significance of this has been explored already for structured and vectored ports [*cf.* **Sections 3.8.2.5 and 3.8.2.4**] and for bonds and components [*cf.* **Section 3.8.3**].

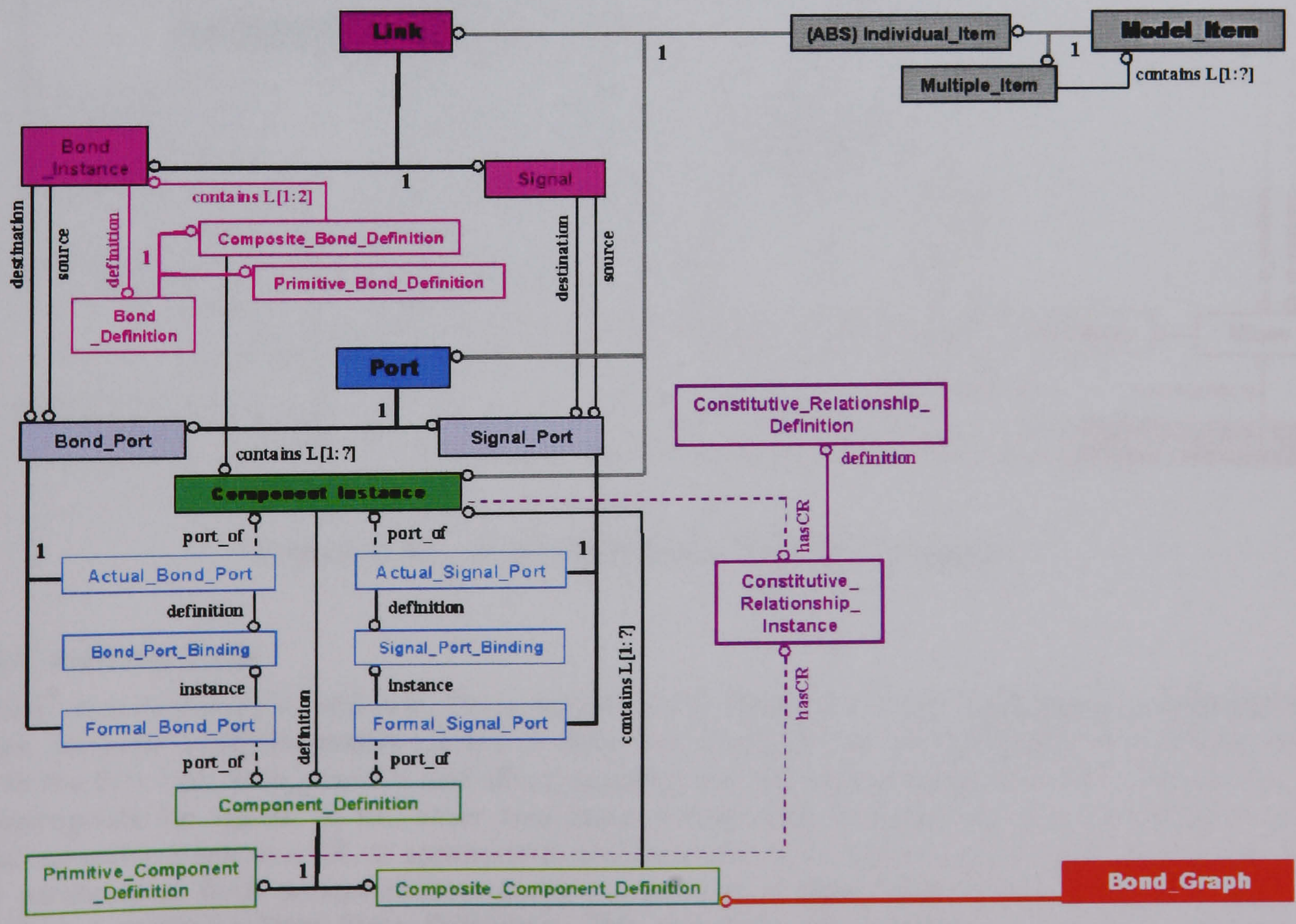


Figure 3.40 Generalised Bond Graph Concepts

3.10.3 Adding Superstructures

At this point the bond graph information has been generalised and the main extensions that need to be introduced are the superstructures, *i.e.* **Library** [*cf.* **Section 3.9.1**], Distribution [*cf.* **Section 3.9.2**], View [*cf.* **Section 3.9.3**] and Partition[*cf.* **Section 3.9.4**]. As a brief summary, it is sufficient to note that

- a Bond_Graph can use library components
- a Library can contains bond definitions, component definitions and CRs
- ports and junctions can be members of a Distribution
- model items and distributions can be members of a View
- a Partition is a specialised view

The extended bond graph concepts are shown in **Figure 3.41**.

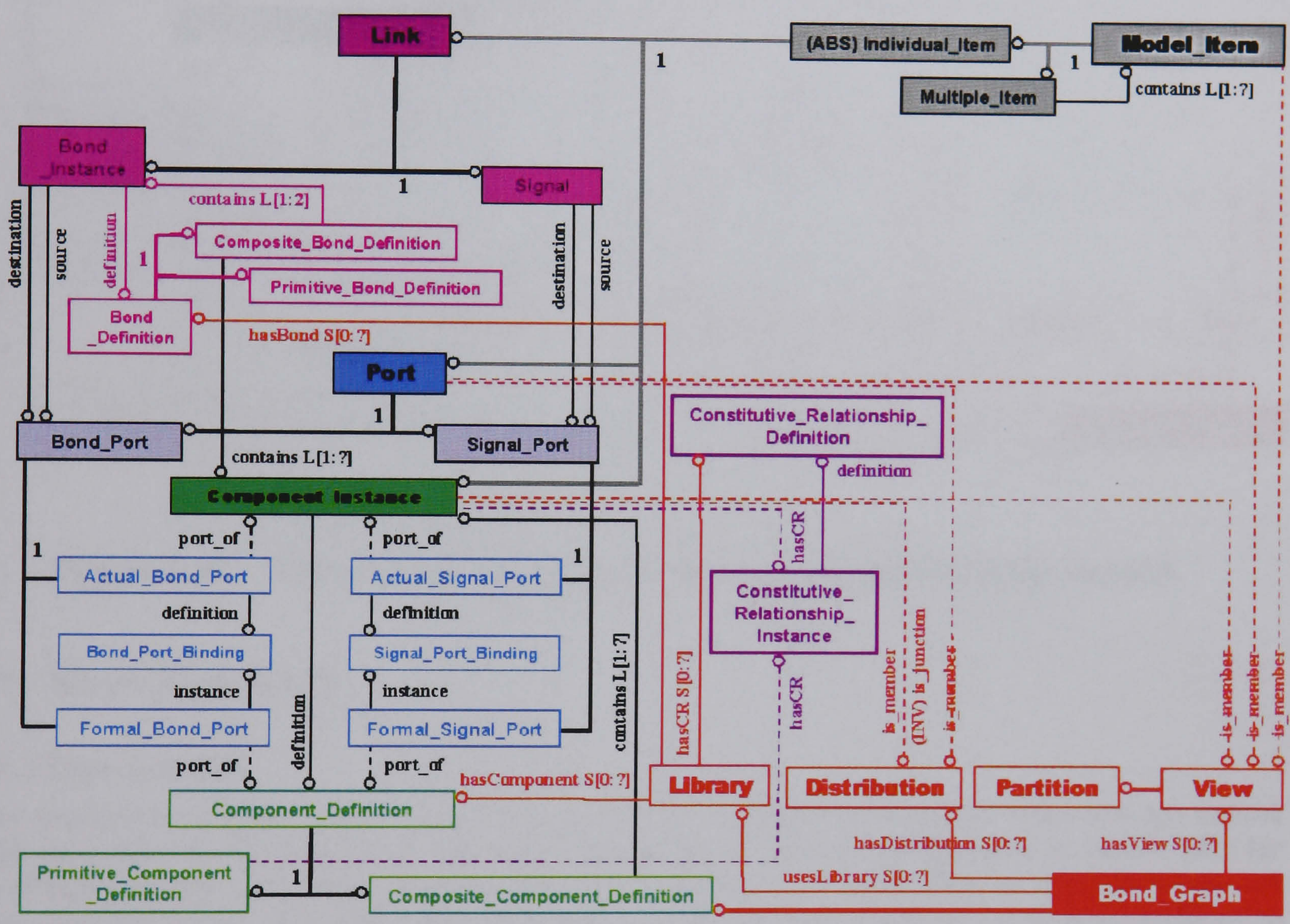


Figure 3.41 Extended Bond Graph Concepts

3.10.4 Adding Data

Finally, it is necessary to add data. These fall into three categories, firstly, **Orientation** and **Causality** of links, secondly, **Link_Parameter** carried on links and, thirdly, **State** and **Parameter** data defined with CRs. In the first case, flow_causality and effort_causality are defined separately for bonds whereas this is not appropriate for signals. In the other two cases, a distinction is drawn between parameters/states (contained within a link or a CR, as appropriate) and parameter/state values (carrying the current data for those parameters). Each parameter/state entity is related to a **Data_Type_Instance**, which in turn is defined by a particular **Data_Type_Definition**. This completes the overview of a unified information model for bond graph, as shown in **Figure 3.42**.

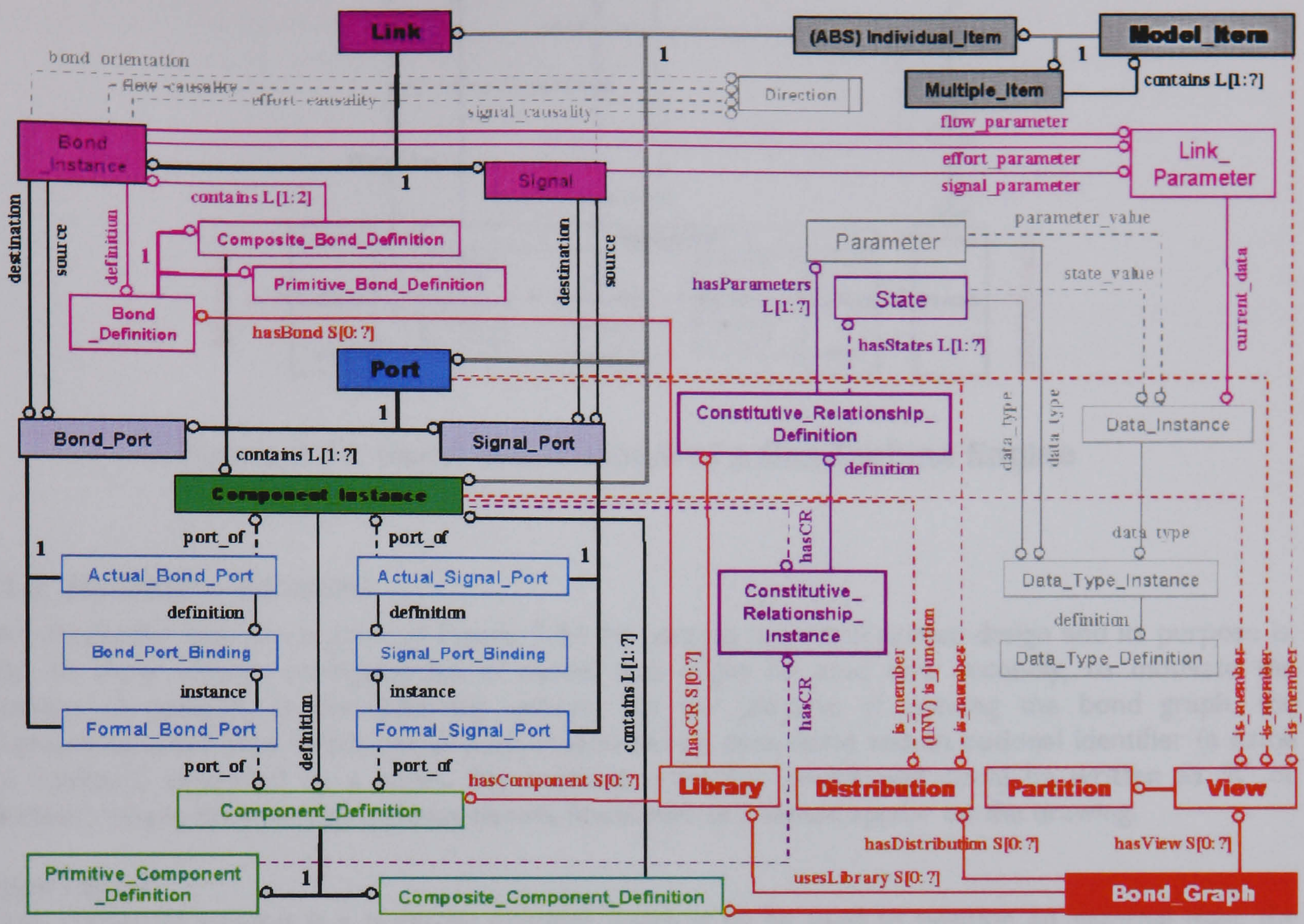


Figure 3.42 Overview of a Unified Information Model for Bond Graphs

3.11 More Examples

3.11.1 Gas Turbine

An example bond graph model is given in **Figure 3.43**. It shows a conventional twin-spool gas turbine engine, decomposed into its major components. The interaction between components is colour-coded for ease of interpretation. Airflow (in *red*) passes from intake to thrust, with separate paths through core and bypass. Bleed is drawn off for ECS supply. Mechanical links (in *grey*) transmit power from turbines to compressors; in addition, shaft power from the HP system is used to drive the fuel pump and the secondary power system (SPS). Fuel (in *blue*) is supplied from outside and is pressurised for injection into the combustor; it is also used to supply fuel-hydraulic (or "fueldraulic") power for the nozzle area actuator. Control (in *green*) is applied to valve/actuator positions.

From this example, it is seen that a bond graph shows a unambiguous picture of power flow around a system. The structure and purpose of the model should be recognisable to any technical onlooker not merely a modelling specialist. Arguably this is the greatest strength of the method and justifies its use on advanced aircraft development projects. Bond graph tools are available which automatically process models into many forms for analysis and generate code for simulation. For all its benefits, modelling and simulation still present challenges and bond graphs do not offer magic solutions. What they do is to increase productivity of working models and to allow engineers to focus on system design and integration, rather than having to spend time debugging simulation code.

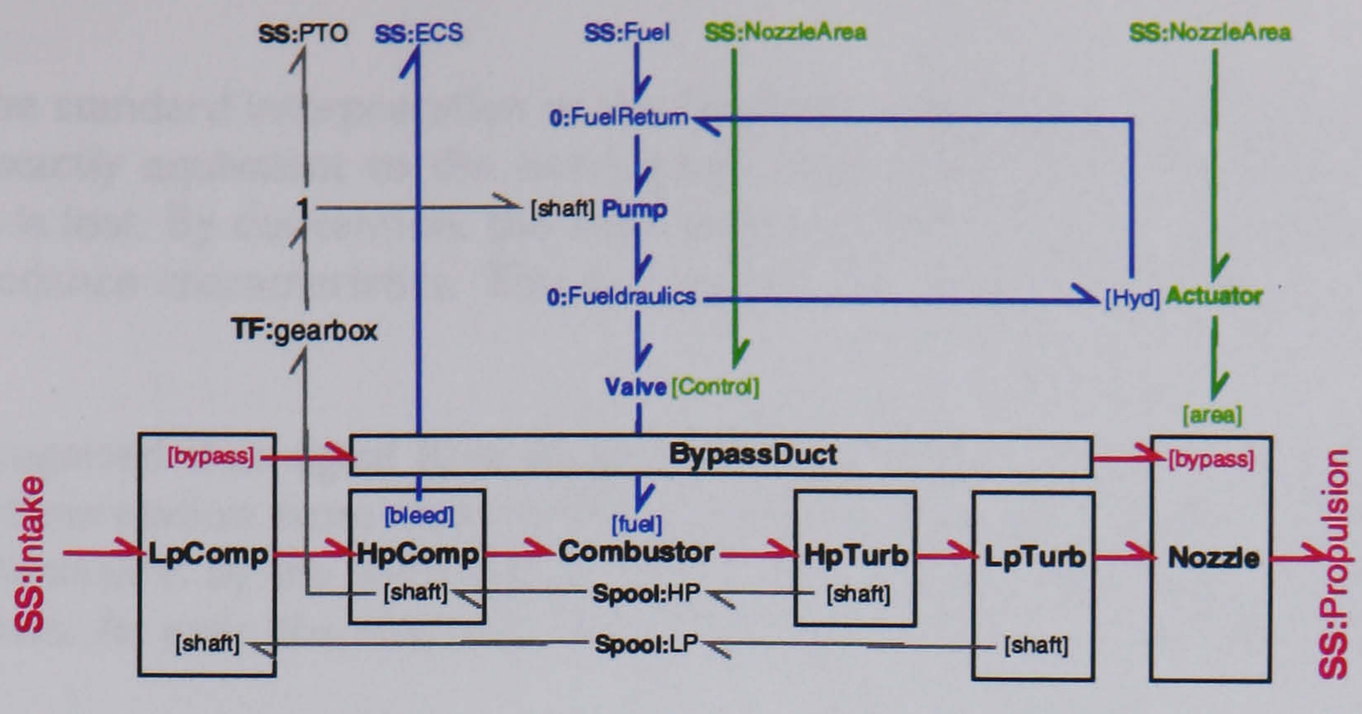


Figure 3.43 Bond Graph Model of a Gas Turbine Engine

3.11.2 Hydraulic Actuator

An illustrative example is given in **Figure 3.44**. Its context is control system design and its purpose is, firstly, to show various configurations of model that might be used and, secondly, to motivate the discussion of causality in the following section. For the purpose of drawing the bond graph, the convention adopted here is that component objects have a class name and an optional identifier (a name or a number), separated by a colon. For instance, a friction component could be written as '**R**' or '**R**:friction', where the inverted commas denote literal text as it would appear on the drawing.

System Model

The system of interest is a hydraulic actuator which is to be used to position an external load, the nature of which is not important here. The basic system model receives a flow from **[Hyd]** and an effort for **[Mech]**. These interfaces imply a distinction between hydraulic and mechanical domains. The hydraulic flow charges a fluid capacitance, **C**:fluid, and the mechanical effort drives the mass of the ram, **I**:mass, against friction, **R**:friction, and against the hydraulic effort. The inter-domain boundary lies at the piston face, represented by a **TF** component; across the boundary, in this model, effort information passes from the hydraulic side to the mechanical side and flow passes in the opposite direction. The **TF** component scales the effort and flow, consistent with the piston area.

The hydraulic effort (pressure) is determined from the difference between the imposed flow *into* the swept volume and the *increase* in that volume, calculated at a flow junction, **0**. The ram velocity is the result of nett mechanical force, calculated at an effort junction **I**, acting on the inertia. These 'facts' or (more correctly) modelling decisions are summarised by the causal strokes shown in *blue*. Causality propagates through the remainder of the model, as shown in *green*. Once engineers become familiar with this notation, it provides a very rapid and unambiguous overview of component interaction.

System plus Control Law

The role of pseudo-bond graphs is illustrated in the context of closed-loop control. The basic system is augmented by an actuator, characterised in this case as a pump, and by a position sensor. For convenience, the measurement signal is a 'flow' and the actuation signal is an 'effort' although this does not have any particular significance except that the two signals need to be segregated if a pseudo-bond graph is to be used for the control law model. The input and output signals are functionally equivalent to amplifiers with infinite impedance. Thus the control law model does not affect the total energy content of the physical system model; it merely observes the system via the sensor and modulates the power being added to the system via the actuator. The external load is treated as a disturbance. Note that causal strokes indicate the nature of physical component interaction and explicitly show the signal flow within the control law.

Feedback Control

For reference, the standard interpretation of the feedback control system is given in a signal flow block diagram, which is exactly equivalent to the bond graph representation except that the internal detail of the physical system is lost. By convention, the load disturbance is included as a bias on actuator position, determined by impedance characteristics. The control law is a proportional-plus-integral filter acting on the position error.

It should be recognised that signal flow diagrams show transmission properties and leads to a very different style of interpretation from that can be gained from a bond graph. They are not intended to represent physical structure: by the same token, it could be argued that bond graphs are not intended to represent control laws. As ever, the choice of notation is ultimately a matter for the modeller.

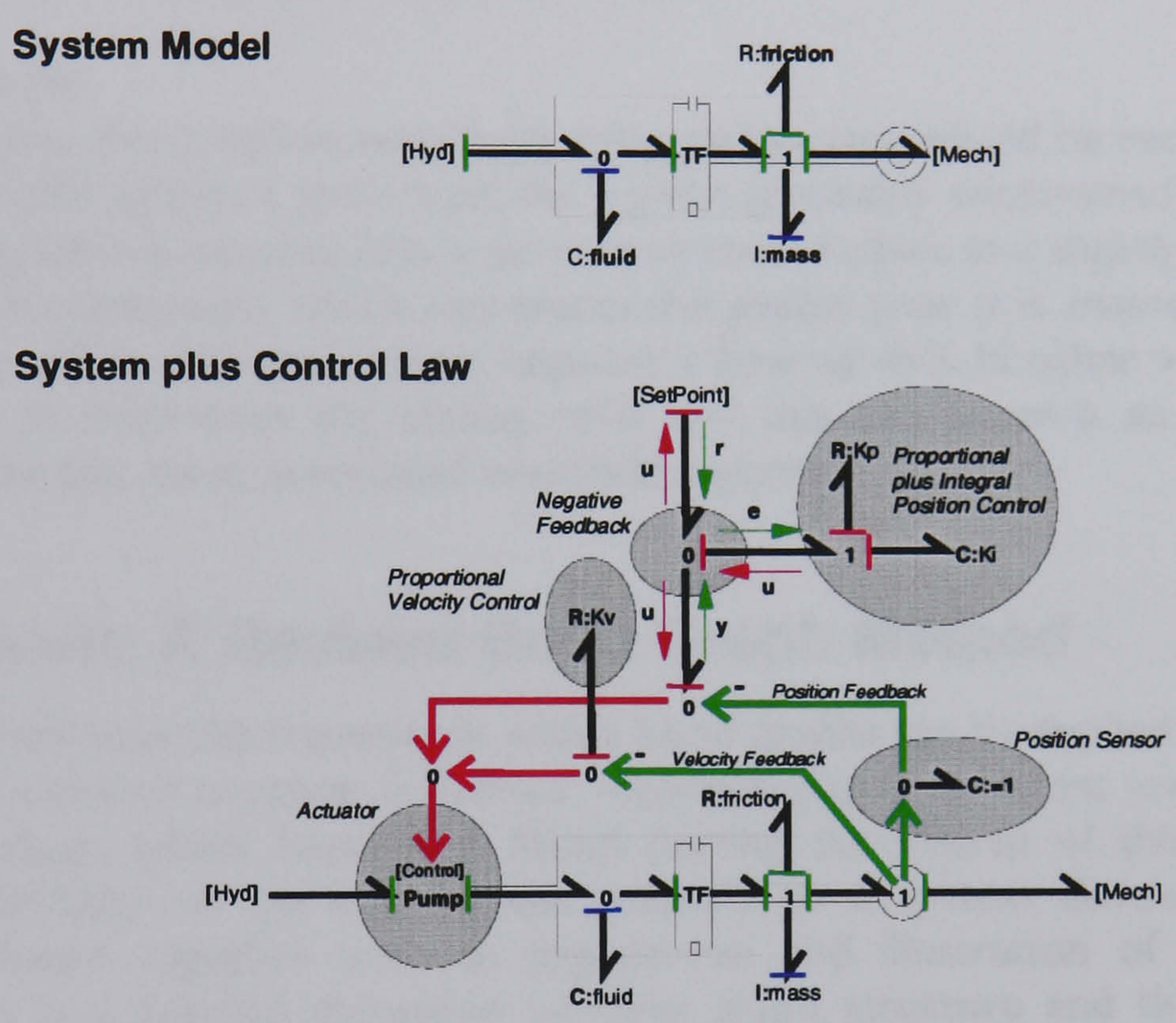


Figure 3.44 Hydraulic Example

Inverse System Model

A powerful feature of bond graph models is that they can be configured for different purposes simply by changing causality. An inverse system is sometimes a useful thing to have because it, from a control perspective, it allows questions to be posed about controllability and system sizing. In the first case, the question would ask how the set-point would have to change over time in order to produce a desired position profile against a given load. This means that at the mechanical interface, **[Mech]**, both effort and flow would be imposed and at the hydraulic interface, **[Hyd]**, both effort and flow would be determined.

By imposing these constraints, the model becomes *bicausal*. The model still represents a physical system but it would be impossible to drive the system in this way. This is seen in the split between effort and flow causality along the path linking the two interfaces. Also note that the energy stores operate differently from before. They do not cause the system states (fluid volume and linear momentum) to change but simply respond to external stimuli. Thus, the inverse model does not have any states because the state variables cannot be assigned independently.

Actuator Sizing Model

In the second case, the question would ask what piston area would be required in order to produce a desired position profile against a given load, for a given (probably constrained) set-point profile. Much the same issues arise as before because this is an inverse model albeit in a slightly different configuration. The impact is on the TF component which represents the piston area; it is *over*-causal in the sense that both domains impose an effort and one domain imposes a flow as well. In other words, the effort information alone is sufficient to determine the scaling ratio and this can become an output of the model. This particular model has one state, associated with fluid capacitance.

3.12 Conclusion: A Revised Bond Graph Method

This chapter establishes the framework within bond graphs can be applied to the modelling of air vehicle systems. A baseline notation is defined, incorporating a significant number of novel features and notational changes which have been found (during the course of this research project) to improve the applicability of the bond graph method to a system development context. Basic notation is introduced together with an explanation and illustration of the mathematics that underlies it. There is a detailed discussion of bond graph structure and the rationale behind the significant changes that have been made. Finally, the concepts of bond graph modelling have been placed into the formal context of an information model. It is believed that the work presented in this chapter revises the standard bond graph method in a comprehensive way not attempted hitherto. It is intended to make bond graphs much more attractive for complex integrated systems.

Chapter 4

Bond Graph Model Libraries

SUMMARY
Bond graph modelling can be applied across all energy domains in order to represent the dynamics of physical processes. As a method, it allows supports the construction of complex models, usually based on a small number of standard primitive components. In the interests of reusability, a library facility is important because it obviates the need to rebuild common configurations and it provides a useful opportunity to standardise the interpretation of component parameters in the appropriate domain context. This chapter defines a set of bond graph libraries which will support the development of air vehicle system models in the following chapter. The aim is to summarise the main physical principles that are of interest in each domain (so as to declare the limits of applicability of models) and to build bond graphs of the main equipment components that are relevant to this project.

4.1 Introduction

Bond graph modelling offers a powerful method for representing power transfer within a system. It is especially useful for air vehicles because it covers all utility functions (*i.e.* those which generate and distribute power in order to allow the aircraft to fly, to regulate the aircraft environment and to support the avionics functions). A bond graph provides a very convenient shorthand notation which, in many respects, this simplifies the process of building mathematical models. The purpose of this chapter is threefold. Firstly, it will give concrete examples of how the notation is to be used in practice. Secondly, it will provide a foundation of component-level modelling which can be applied to the construction of full system models [Chapter 5]. Thirdly and finally, it will summarise the physical principles involved in the major subsystem technologies.

The main result is a consolidation of the main elements required for aerospace system modelling, presented in a way which (surprisingly) is hardly ever considered in published material. Most standard texts focus on particular technologies and tend towards mathematically-based expositions, supported by simulation studies. Notable exceptions exist [Cellier 1990, Karnopp et al. 1990] which unify a number of different viewpoints. This philosophy will be followed here because of the need to make predictions about the behaviour of integrated systems.

This chapter is organised as a set of library definitions for various domain systems. These are established to a level appropriate for building conceptual models of aircraft systems and may not necessarily be reusable in other engineering contexts. For current purposes, it is sufficient to cover *five* major component categories, as follows:

Generic
Hydraulic
Thermofluid
Electrical
Flight Dynamic

The aim is to demonstrate the practicalities of the bond graph method, in general, and the novel developments discussed in Chapter 3, in particular. Also, if full system models are going to constructed then sufficient variety has to be introduced in these libraries in order to model power transfer within and between on-board systems.

4.2 General Principles

There are a number of general principles that apply when making decisions about the amount of detail to be contained in a model. It is useful to review these here because they have had a profound (if implicit) influence on the design of models presented in this chapter. Five principles are relevant and these are summarised below:

Interpretation

A trade-off is needed between the level of detail that goes into a model and the ease with which the model content can be interpreted. For integrated systems, the crux of the issue is what level of detail can be introduced without obscuring the functional mechanisms (reflected in dominant nonlinear characteristics and low-order dynamics).

Encapsulation

Encapsulation means that components have rigorously defined interfaces, arranged in such a way as to support re-use. Assembly of models should be trivial, thereby allowing the modeller to concentrate effort on model design.

Parsimony

Models should proceed from a conceptual level and evolve to a detailed level, observing a principle of parsimony. In other words, detail ought to be added sparingly so that the models are fit for purpose and not over-embellished. Clearly, this means that purpose must be explicit, so must the acceptance criteria and the method for assessing models.

Topology

The layout of an integrated system model should be intuitively obvious to an intelligent observer (not just a modelling specialist). If necessary, a family of models can be built in order to show a logical progression from a functional system concept through to a detailed performance model. Not all members of the family need necessarily be compilable or executable but they should all use the same notation.

Resolution

The limit of component resolution should be low enough to represent all moving parts (fluids, inertias, etc.) in a system, including actuators and sensors, but high enough to ignore phenomena outside a pre-defined bandwidth (*i.e.* above some frequency) and outside a pre-defined tolerance band for parametric data.

What all this amounts to is a philosophy that says that a model of a system should match the system topology, its assembly should be easy and its design should emerge and evolve in a way that is appropriate and well-understood.

4.3 Domain-Specific Issues

Particular issues arise in the construction of models of complicated engineering systems that span several domains. The basic problem was discussed in Section 3.6 in order to understand the role of pseudo-bond graphs. There is considerable freedom available to the modeller in specifying variables that are not power co-variables although power may be present in some combination of the variables, perhaps as a result of scaling.

Individual domains do not present a problem provided that the interface between domains is correctly and rigorously defined. However, there is a significant problem when domains are defined in such a way as to dependent upon other domains. Typical examples include:

- Fluid flows which carry a concentration of material in solution
- Flows of mixed gases
- Convection of heat with a fluid or gas (generically known as a thermofluid)

The first two examples define a transport problem and the third defines a convection problem. Both represent one-way (or uni-directional) processes in the sense that the dependent bonds must propagate information in the same direction as the flow of the fluid or gas. A combined problem occurs in gas turbine models, for instance, where the working medium (air) carries water vapour and, downstream of the combustor, fuel/air combustion products.

The immediate consequence for bond graphs is that, when **C**, **I** and **R** components are applied to dependent bonds, they become modulated components. The modulation is provided by the independent flow variable, usually *massflow*. Also, since the dependent portion of the model is invariably a pseudo-bond graph, these components need to be used with care because they will not represent physical mechanisms in the true sense in which a 'true' bond graph would. Uni-directional 'flow' means that amplifiers, or equivalently signals, will be employed. A requirement for bi-directional flow must be addressed through a dedicated composite bond that enables the dependent variables to follow the independent flow variahle.

The reason for moving away from true power bonds in these applications is to be able to build models in which flows of one type or another are conserved at flow junctions. When dealing with gas flow, for instance, volumetric flow is not a conserved quantity and this introduces many problems when connecting components. One possible solution is to build pseudo-bond graphs as composite component definitions that appear externally like true bond graph components. This is not adopted here because individual components would be buffered by redundant conversions. It is much preferable to encapsulate the pseudo-bond graph at the level of major subsystems, *e.g.* an engine.

This might be thought to compromise the simplicity of bond graphs but, provided that domain properties are properly constructed, the modelling approach is still highly intuitive. This section will briefly cover some of the key issues involved for transport and thermofluid processes.

4.3.1 Transport Processes

The variables applied to a transport process are summarised in **Table 4.1**. The independent 'Massflow' domain replaces the standard hydraulic domain, using massflow as opposed to volumetric flow. The dependent 'Transport' domain contains the dissolved material or the minor gas component (whichever is relevant). It carries massflow as its flow variable and introduces concentration (*i.e.* Kg s^{-1} of transported massflow per Kg s^{-1} of total massflow) as its effort variable.

Domain	Variables		State Variables	
	Effort	Flow	Momentum	Displacement
Massflow	Pressure	Massflow	Pressure Momentum	Mass
Transport	Concentration	Massflow Ratio	-	Mass Ratio

Table 4.1 *Pseudo*-Bond Graph Variables for a Transport Process

Compatibility between Massflow and Transport domains[1] is achieved as shown in **Figure 4.1**. The independent variables (**P**,**F**) [denoting pressure and massflow, respectively] are bound to the transport massflow $\mathbf{f_i}$ via an **R** component and to the transport concentration $\mathbf{c_i}$ via a pair of **C** components, with the total stored mass **M** being transferred as a constraint variable.

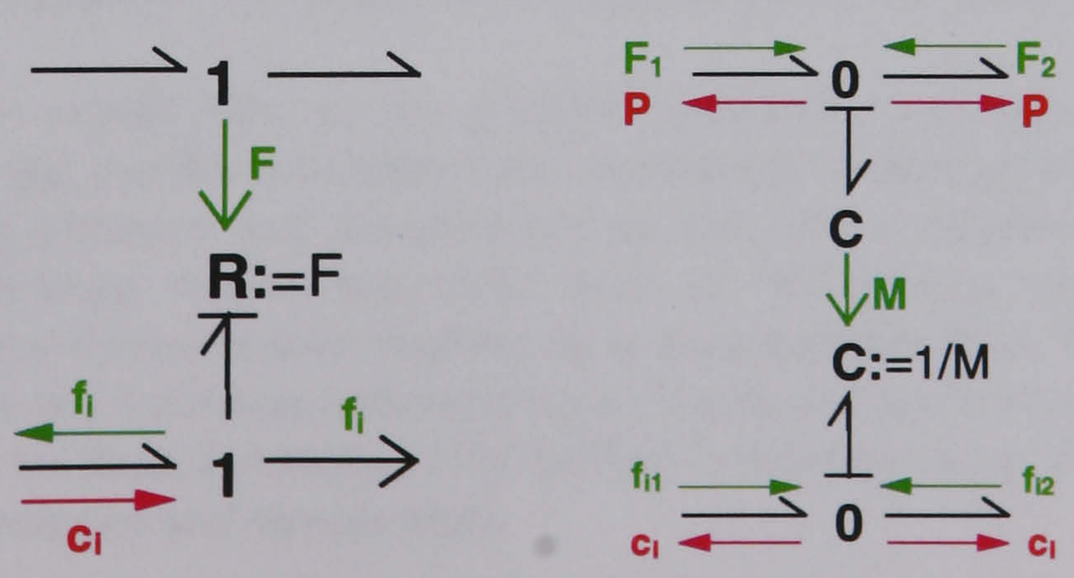


Figure 4.1 Domain Compatibility for a Transport Process

[1] For more details, refer to Gawthrop & Smith (1996), Section 8.2.3.

4.3.2 Thermofluid Processes

The variables applied to a thermofluid process are summarised in **Figure 4.2**, replacing the standard hydraulic and thermal domains with equivalent 'Massflow' and 'Enthalpy' domains, respectively. This covers the general case of *compressible* flow; for incompressible flow, it is a simple matter, either re-introduce volumetric flow or to constrain the relationship between mass and volume to be constant (*i.e.* constant density).

Domain	Covariables		State Variables	
	Effort	Flow	Momentum	Displacement
Massflow	Pressure	Massflow	Pressure Momentum	Mass
Enthalpy	Temperature	Enthalpy Flow	-	Enthalpy

Table 4.2 *Pseudo*-Bond Graph Variables for a Thermofluid Process

Compatibility between Massflow and Enthalpy domains[2] is achieved as shown in **Figure 4.2**. The independent variables (**P**,**F**) [denoting pressure and massflow, respectively] are bound to the heatflow $\mathbf{Q_i}$ via an **R** component and to the temperature **T** via a pair of **C** components, with the total stored mass **M** being transferred as a constraint variable.

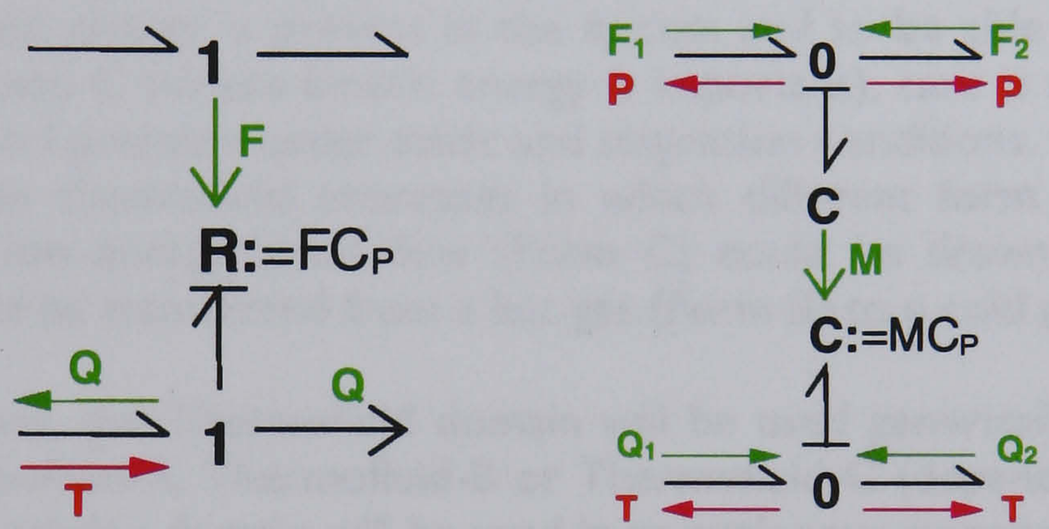


Figure 4.2 Domain Compatibility for a Thermofluid Process

The main problem lies in ensuring that enthalpy flow remains consistent with massflow. Essentially, temperature is a property of the fluid; it is carried along with the fluid. In the preparatory work for the library definitions, ideas were explored to introduce a 'Temperature' attribute which could be assigned to fluid flow but the severe drawback of this approach was that hydraulic aspects of systems could be given a neat graphical representation but the thermal characteristics could not. All thermal effects would have to be modelled using specialised components, even if the thermal processes would otherwise have had an obvious bond graph representation. Because the bulk of problems arise locally because of empirical relationships into models, a complete conceptual re-orientation could not be justified.

In a similar vein, a major modification to the graphical notation, as proposed elsewhere [Krikelis & Papadakis 1988] to handle gas turbine modelling was considered inappropriate. This introduced a so-called *double* bond carrying pressure and temperature as two effort variables and a single (common) massflow variable. This addresses exactly the same issue of redundancy, recognising that only three variables are necessary to specify the power transfer by a compressible fluid. The underlying philosophy to that approach is very different from that advanced here. There, the aim is to standardise all elements of a thermodynamic system to be Basic Functional Unit (BFUs), consisting of an actuator disc and a volume with uniformly distributed pressure and temperature.

[2] For more details, refer to Gawthrop & Smith (1996), Section 8.2.4.

While clearly effective as an approach to model-building and as a way of providing useful insight into equation-based modelling, it does not offer (or indeed set out to offer) a conceptual framework for modelling physical systems. This is a moot point; depending on the purpose behind the modelling process, both approaches could be valid but the approach being presented here has opted for a closer association with system topology.

An important question that comes out of this discussion is that of how best to partition the total energy content of a fluid into its constituent parts. The well-known equation for Steady-Flow Energy is

$$E = mC_V t + pV + \frac{1}{2}\rho U^2 \qquad \text{(Form A)}$$

(neglecting the effects of gravity). By collecting together the terms in this equation, two other forms are

$$E = mC_P t + \frac{1}{2}\rho U^2 \qquad \text{(Form B)}$$

$$E = mC_P T \qquad \text{(Form C)}$$

such that the hydraulic effects are embedded inside C_P and, thence, kinetic energy is incorporated in the value of temperature (*i.e.* 'static' temperature becomes 'stagnation' temperature). This gives three forms of energy accounting. The choice is one of preference.

In Form A, a model would require three separate bonds for *internal*, *hydraulic* and *kinetic* energy. In Form B (neglecting kinetic energy), hydraulic power (pV) would have to be accounted separately in order to know how much *non*-heat energy is present in the system and to be able to calculate massflow and pressure distributions. In Form C (where kinetic energy is important), care is required in reconciling the definitions of temperature and pressure under static and stagnation conditions. Care is also needed where flow is transferred between thermofluid processes in which different form of energy accounting are appropriate. For example, low-energy bleed flow (Form C) could be drawn from a high-energy main stream (Form B). Heat could be transferred from a hot gas (Form B) to a cold pipe (Form A).

In subsequent descriptions, the Thermofluid domain will be used generically and, where appropriate, will be specialised as Thermofluid-A, Thermofluid-B or Thermofluid-C (depending on the form of energy accounting required). The Enthalpy domain will be used in an analogous manner.

4.4 Presentation Style for Libraries

A small amount of annotation will be defined in order to enable bond graph libraries to be organised in a 'readable' layout. Each bond graph definition has a rectangular enclosure with a legend to identify what the definition is. This is the unit of declaration and can be applied in a hierarchical structure. A unit can be a **library**, a **model**, an **interface** or a **bond interface**.

A unit can contain other units. The only restrictions are applied to interfaces or bond interfaces, such that they can only be implemented by models or bonds respectively. They can be named or un-named (provided that some form of context dependency exists). If the name of the one of the implementations is the same as the name of the interface, then it will be treated as the default implementation. For models, a particular implementation will be labelled as `InterfaceName->ComponentName` or, if there is no ambiguity, simply as `ComponentName`. For bonds and bond interfaces, the definition must include a picture of the new bond style.

Models can be specialised or redefined with particular properties by means of a statement of the form:

model NewModel **is** OldModel **with** Property = PropertyValue

This can be used, for instance, to define a mandatory CR for a primitive component or to substsitute a component class with another, perhaps one from another library.

As a final comment, the declarative legend can contain supplementary information, such as a domain (*e.g.* **domain** Electrical) or a library dependency (*e.g.* **use** Hydraulic).

4.5 'Generic' Library

Useful component classes exist which are independent of the context in which they are applied. These mainly concern switching operations. For example, a line contactor in an electric circuit serves the same basic function as a hydraulic shut-off valve; a diode serves the same function as a non-return valve. A set of logic operators is handling discrete events and operational states. In addition, there are a few minor but important components for integrating signals (*e.g.* deriving position from velocity) and for three-way power branches.

Ignoring the details of how these devices work in reality, simple switch mechanisms can be applied across a wide range of modelling applications. It is convenient to define these components as generic because they are simple and they can be readily used in other libraries. The library definition is shown in Figure 4.3.

4.5.1 'Logic' Sub-Library

Boolean

This derives a boolean signal (0 or 1) according to the CR denoted by **#Logic**, which inter-relates the port input and output variables as folows: `p.out:=1-p.in>0` .

GT and LT

A comparison test is performed on an input signal in order to establish whether it is greater than or less than an external reference signal.

Positive and Negative

A comparison test is performed on an input signal in order to establish whether it is greater than or less than zerol.

Not

This derives the complement of a boolean signal (0 or 1).

4.5.2 'Switch' Sub-Library

Valve

A generic valve is a variable flow restriction. The modulating signal has two elements, injected at **[mod(1..2)]**. The first element is filtered by the combined effect of **I** and **R** components, which give a first-order lag. The second element is driven directly. If the modulation is driven by a boolean component then the valve will be *either* "on" (*i.e.* non-zero conductance) *or* "off" (*i.e.* zero conductance).

NonReturn

Non-return characteristics are provided by a generic valve with the flow measurement passing through a boolean component which establishes whether is positive or not. The result then drives the valve modulation at **[mod(2)]**. For convenience, the external modulation through **[mod(1)]** is retained so that flow control can be exercised via a single component, rather than via a Valve and NonReturn in series (with the introduction of an algebraic loop).

4.5.3 'Branch' Sub-Library

Branch

Flow branching is established as two controlled flow paths (using generic valves) and one resultant. The modulation is now injected as a structure **[mod.<L,R>]**, giving independent control of left and right paths. By default, modulation will be applied to the **[mod(1)]** port on each Valve.

Selector

Flow selection is established by a Branch modified to accept a boolean input signal. This is passed directly to one of the valves and its complement is passed its complement to the other valve, thereby ensuring that one flow path is active at any given time (notwithstanding transient periods when valve switching is taking place).

Star

A star coupling is defined as a three-way flow junction.

Delta

A delta coupling is defined as a three-node network with through ports separating the flow junctions in order to provide place-holders for other components.

Differential

A differential coupling is defined as a three-node network similar to a delta coupling, except that two of the nodes are flow junctions and the remaining node is an effort junction.

4.5.4 'Integrator' Interface

Signal integration is often helpful in setting up component modulation. This facility is provided here as an interface to a FlowIntegrator and an EffortIntegrator. These are pseudo-bond graphs that apply energy storage components (with unit capacity) purely in order to derive an internal state which is exported.

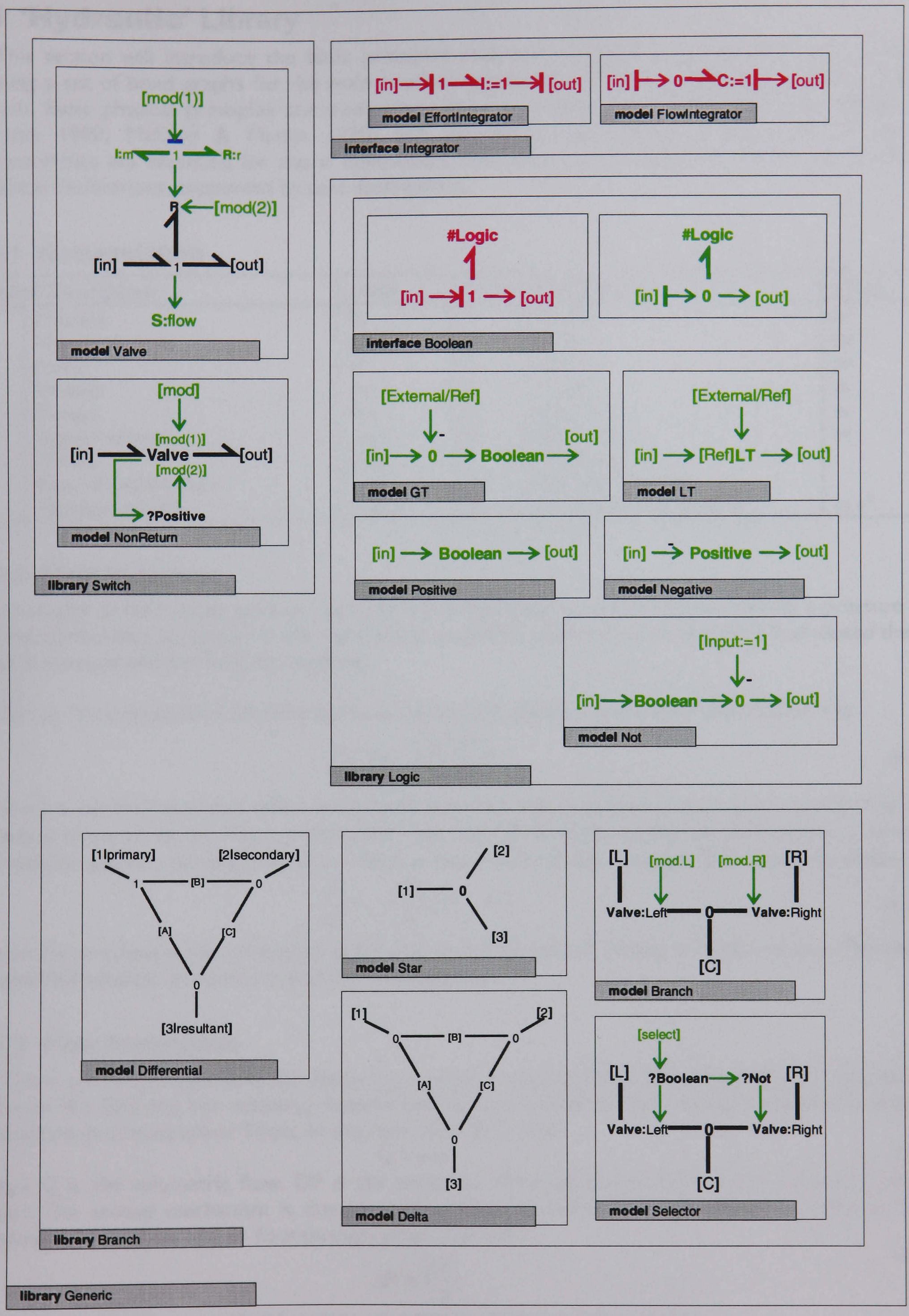


Figure 4.3 'Generic' Library Components

4.6 'Hydraulic' Library

This section will introduce the basic principles underlying hydraulics and actuation with a view to defining a set of bond graphs for the main product components which go together to form a hydraulic system. Basic physical principles are summarised from a number of standard texts [*e.g.* Green 1985, Viersma 1980, McCloy & Martin 1980] and elsewhere [*e.g.* Harpur 1953], and the dominant characteristics are identified for use in component CRs. Component classes are defined in the form of graphical declarations, supported by text descriptions.

4.6.1 Nomenclature

Symbol/Description		*Units*	*Symbol/Description*		*Units*
P	Pressure	Pa	L	Length	m
Q	Volumetric flow	$m^3\ s^{-1}$	A	Area	m^2
F	Force	N	V	Volume	m^3
v	Velocity	$m\ s^{-1}$	h	Height	m
τ	Torque	Nm	D	Diameter	m
ω	Angular velocity	$rad\ s^{-1}$	R	Bend radius	m
ρ	Density	$Kg\ m^{-3}$	f	Friction factor	
γ	Ratio of specific heats		Re	Reynolds Number	
μ	Absolute viscosity	$N\ s\ m^{-2}$	g	Gravitational acceleration	$m\ s^{-2}$

4.6.2 Fluid Volumes

A volume of fluid under pressure behaves like a capacitor, storing fluid and exerting a pressure force on the surrounding container. If the container is *open* then fluid mass can change; if it is *closed* then the mass is constant and the fluid acts a spring.

For an "*incompressible*" fluid like hydraulic oil, the CR between pressure P and volume V is:

$$Q - Av = \left(\frac{V}{\beta}\right)\frac{d}{dt}P \tag{4.1}$$

where β is the bulk modulus, which varies with pressure and temperature and varies considerably in the presence of entrained air. For a *compressible* gas, the CR is ideally written as $PV^{\gamma}=constant$, where γ is the ratio of specific heat capacities at constant pressure and constant volume. This can be re-written as:

$$\frac{d}{dt}P = \gamma\left(\frac{P}{V}\right).(Q - Av) \tag{4.2}$$

where the nett flow is the volumetric inflow, Q, minus the rate of change in swept volume. This assumes a reversible adiabatic process of compression/expansion.

4.6.3 Flow Resistance

There are two mechanisms for dissipating energy in moving fluids. The *first* is due to viscous forces between the fluid and the retaining material and applies to laminar flow through capillaries and to flow through porous materials or filters. In this case, the CR is linear:

$$Q = k.\Delta P \tag{4.3}$$

where Q is the volumetric flow, DP is the pressure difference across the restriction and k is the flow factor. The second mechanism is due to viscous forces between fluid particles and applies to sudden restrictions or orifices and to flow through pipes. The CR is

$$\Delta P \propto \frac{\rho V^2}{2} \tag{4.4}$$

i.e. pressure drop is proportional to dynamic pressure. This is usually written as:

$$Q = k.sgn[\Delta P]\sqrt{|\Delta P|} \tag{4.5}$$

For a variable orifice, the flow factor has the form

$$k=A.C_D.\sqrt{(2/\rho)} \tag{4.6}$$

where A=orifice area, ρ=density and C_D=*discharge coefficient*. For pipes, the pressure loss depends on Reynolds Number although most engineers work in terms of friction factor, f, for a given pipe geometry and surface roughness. For a straight pipe of length L and diameter D, the discharge coefficient is

$$C_D=A.\sqrt{(D/Lf)} \qquad (4.7)$$

A common estimate for friction factor is:

f	=	64/Re	for Laminar flow, Re<2000
	=	Re3/4/9430	for Unstable flow, 2000<Re<3000
	=	0.32Re-1/4	for Turbulent flow, 3000<Re<105

For bends in pipes, the flow factor becomes more complicated and is described by an empirical relationship:

$$k=A.f(R/D).\sqrt{(2/\rho)} \qquad (4.8)$$

for some function f of pipe diameter D and bend radius R.

4.6.4 Pumps and Motors

Aircraft hydraulic pumps contain a reciprocating piston arrangement with a variable swash plate, which determines the sweep of each piston and, in combination with the shaft speed and outlet pressure, the overall flow rate through the pump. At any given speed, delivery will drop off as outlet pressure increases and ultimately the pump will stall; by scheduling the swash plate angle against outlet pressure, this drop off can be minimised over a wide range, up to the so-called system pressure. Delivery will drop off rapidly thereafter. Typical characteristics of such a pressure-compensated pump are shown in figure 4.4. These convey the usual constitutive relationships for a pump, assuming hydro-mechanical control of the swash plate. For a variable pressure system, these can be changed under software control using a servo-valve to drive the swash plate angle.

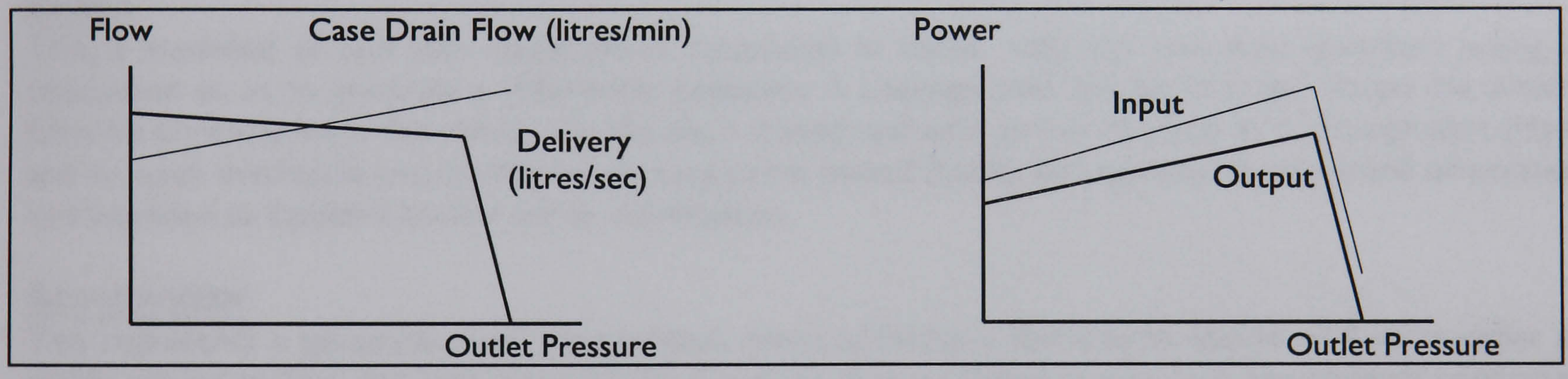


Figure 4.4 Typical Pump Characteristics

An ideal pump (with 100% efficiency) acts as a transformer, with a standard CR:

$$Q = D.\omega, \quad \tau = D.\Delta P \qquad (4.9)$$

where

$$D=AN.r.\tan\alpha/2\pi \qquad (4.10)$$

is the fluid volume delivered *per* unit rotation, for a given swash-plate angle α. The pump configuration is defined by the radial position of the pistons, r, the piston cross-sectional area, A, and the number of pistons, N.

Losses occur in practice from two sources. Fluid losses arise from *internal leakage* between fluid lines and from *external leakage* from each chamber into the casing. Because clearances between mating surfaces are very small, leakage flow is assumed to be laminar. Mechanical losses arise from *friction* between pump components, fluid *shear* inside clearances and friction due to piston *seals* (which is usually neglected).

Pump efficiency is quoted in three ways. Firstly, the *volumetric efficiency* is the ratio of actual flow to that predicted from pump geometry. Assuming that suction and case pressures are small, pump delivery is roughly

$$Q = D\omega\left(1 - \frac{c_s P}{\mu\omega}\right) \tag{4.11}$$

where P is outlet pressure, μ is absolute viscosity and c_s is the so-called machine slip coefficient. This enables the leakage resistances to be found provided their relative size is known. Secondly, the mechanical efficiency is the ratio of input torque for an ideal pump to the actual input torque required to drive a real pump. Under the same assumptions, the input torque can be approximated as

$$\tau = D.\Delta P\left(1 + c_f + \frac{c_0\mu\omega}{P}\right) \tag{4.12}$$

where c_0 is a fluid shear coefficient and c_f is a mechanical friction coefficient. When outlet pressure is large, fluid effects are negligible. Finally, the *overall efficiency* $\eta = (PQ)/(\tau\omega)$ is the ratio of fluid power output to mechanical power input.

4.6.5 Component Class Definitions [*cf.* Figure 4.5]

Jack

This represents a simple hydraulic jack or a single-sided piston. It comprises a capacitive volume which stores fluid and, in so doing, produces an outward pressure. The inlet flow experiences a back pressure and the piston is subjected to a force F=A.ΔP, as described by a TF object, with a gearing ratio equal to the area. Component connections are provided as *open* junctions, a **0** junction on the hydraulic side and a **1** junction on the mechanical side. This will allow, resepctively, for several hydraulic lines (if required) and for the use of double-ended shafts and the addition of inertia and so on.

Piston

This is modelled as two Jack components connected in series, with the two fluid chambers acting in opposition so as to generate a differential pressure. A Leakage path has be included across the piston. External connections in this model are *circuits*, a closed hydraulic circuit denoted by a through port **[Hyd]** and an open mechnical circuit denoted by two ports named **[Link]**, distinguished by the bond orientation and intended to facilitate further series connections.

Accumulator

This represents a gas-pressurised accumulator, which provides a short-term source of fluid in order to satisfy maximum flow demands close to the actuators. It is modelled as two Jack components connected in tandem, with an open chamber storing hydraulic fluid (with the CR defined in (4.3)) and a closed chamber charged with nitrogen (with the CR defined in (4.4)).

Reservoir

This describes a typical reservoir with two chambers, the smaller 'bootstrap' chamber being subjected to full system pressure (nominally 4000psi in most military applications) and stepping it down (to between 50 and 100psi) through a substantial area ratio in order to maintain a base pressure for pump supply. Note that a gas-pressurised reservoir would serve the same purpose but is functionally equivalent to an accumulator.

Spool

Two classes of spool valve object are included here, having two and three lands respectively. For primary actuation (*i.e.* flight control) Spool2 has the more usual configuration of two lands covering the two ports controlling flow to/from the consumer; accordingly, it is designated as the *default* model. Spool3 has three lands covering the three ports controlling flow. Both of these model a combination of hydraulic and mechnical power flows based on a Land component.

Land

A land operates in conjunction with a physical port to expose an area through which fluid can pass. The Land component encapsulates a piston and a generic flow branch consistent with the principle of spool valve operation.

Actuator

An actuator (in this case, a linear actuator) is the basic component for generating mechanical force. It comprises a piston and a spool valve. In order to allow analysis of impedance characteristics, power transfer is modelled between the moving parts of the actuator and the casing via friction; this enables actuators to be attached to aircraft structure.

TandemActuator

Tandem actuators are standard for flight control, provided dual redundancy in hydraulic power and actuator drive (through the spool valves). For purposes of modelling, a tandem device is simply two unitary devices with a common shaft and a common casing.

Pump

This represents the functional mechanism for a pump. A TF object represents the effect of swash plate angle on the positive displacement of fluid, with a transformer ratio equal to $D=AN.r.\tan\alpha/2\pi$, as discussed in section 4.6.4. A modulating signal is injected at S:Control which determine the value of $r.\tan\alpha$. The differential pressure is calculated across a 1 junction. Leakage paths are provided across the valve plate and across the piston heads.

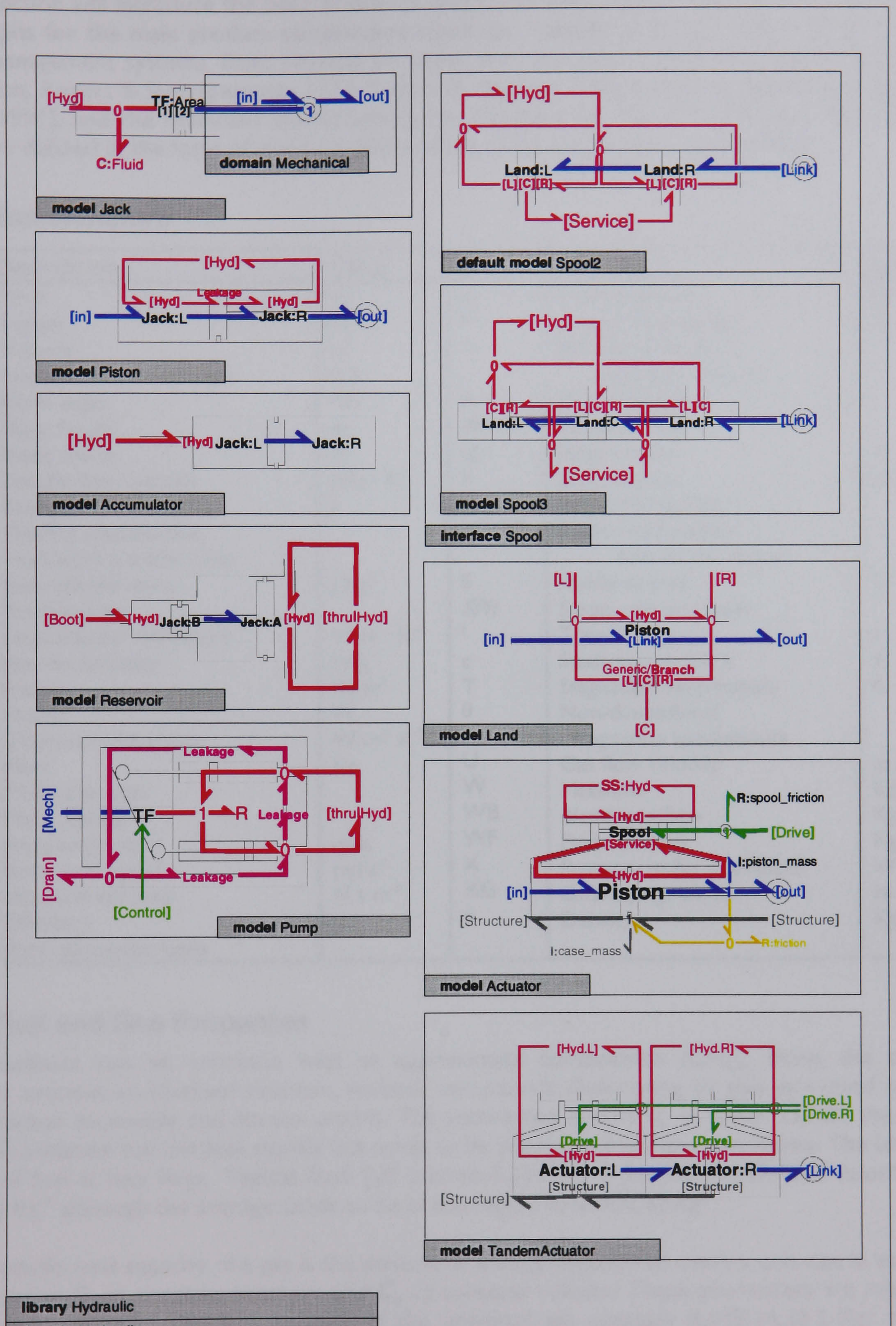


Figure 4.5 'Hydraulics' Library Components

4.7 'Thermofluid' Library

This section will introduce the basic principles underlying thermofluids with a view to defining a set of bond graphs for the main product components which go together to form turbomachines and thermal energy management systems. Basic physical principles are summarised from a number of standard texts [*e.g.* Cohen, Rogers & Savaranamutto 1972, Rogers & Mayhew 1980] and other publications [*e.g.* Ismail & Bhinder 1991], and the dominant characteristics are identified for use in component CRs. Component classes are defined in the form of graphical declarations, supported by text descriptions.

4.7.1 Nomenclature

Symbol	*Description*	*Units*	*Symbol*	*Description*	*Units*
A	Area	m^2	p	Static pressure	Pa
L	Length	m	P	Stagnation pressure	Pa
V	Volume	m^3	Δ	Non-dimensional stagnation pressure	
ANG	Nozzle contraction angle	rad			
β	Blade angle	rad	Pr	Prandtl Number	
b	Blade height	m	*PR*	Pressure ratio	
c	Blade chord	m	Q	Heat transfer	W
C	Specific heat capacity	J Kg^{-1} K^{-1}	R	Gas constant	J Kg^{-1} K^{-1}
E	Energy	J	Re	Reynolds Number	
EC	Cooling effectiveness		*RTDF*	Radial temperature distribution factor	
F	Heat/work transfer ratio				
FCV	Fuel calorific value	J Kg^{-1}	S	Planform area	m^2
FAR	Fuel/air ratio		*SPR*	Surge pressure ratio	
h	Heat transfer coefficient	W m^{-2} K^{-1}	*t*	Time	s
H	Specific enthalpy	J Kg^{-1}	t	Static temperature	K
I	Inertia	Kg m^2	T	Stagnation temperature	K
J	Power	W	θ	Non-dimensional stagnation temperature	
k	Thermal conductivity	W m^{-1} K^{-1}			
m	Mass	Kg	U	Gas flow velocity	m s^{-1}
m	Molecular mass		W	Massflow	Kg s^{-1}
M	Mach number		WB	Bleed massflow	Kg s^{-1}
N	Rotational speed	rpm	WF	Fuel massflow	Kg s^{-1}
Ω	Rotational speed	rad s^{-1}	X	Power transfer coefficient	W K^{-1}
μ	Absolute viscosity	N s m^{-2}	XG	Gross gauge thrust	N
η	Efficiency		ρ	Density	Kg m^{-3}
γ	Ratio of specific heats				

4.7.2 Fuel and Gas Properties

Gas turbines run on kerosene with an approximate composition $(CH_2)_n$. Here, the combustion chemistry assumes an idealised reaction, without compounds dissociating to give unwanted by-products such as carbon monoxide and nitrous oxides. The assumption is valid up to about 1500K; thereafter, the level of dissociation can increase rapidly but tends to be suppressed at higher pressures. The latent energy content of fuel is very large. Typical fuels [*cf.* Section 5.5] have a minimum permitted calorific value of 42800 KJ Kg^{-1} although the average tends to be in the region of 43420 KJ Kg^{-1}.

The specific heat capacity of a gas is the amount of energy required to cause a unit rise in temperature. It is defined as C_P at constant pressure and C_V at constant volume. These parameters are related to the gas constant $R=C_P-C_V$. This is a function of the universal gas constant $R_0=8314.33$ J Kg^{-1} K^{-1} and the relative molecular mass of the particular gas mixture *m*. A working definition can be written as

$$R = R_0 \frac{1 + FAR}{m_1 + m_2.FAR} \tag{4.13}$$

where FAR is the fuel/air ratio, using values $m_1=28.965$ for dry air and $m_2=14.020$ for aviation fuel. The absolute values of specific heat at constant pressure C_P and the ratio of specific heats $\gamma=C_P/C_V$ are shown in figure 4.4.

4.7.3 Specific Enthalpy

The specific enthalpy of a gas at a given temperature is the total energy stored per unit mass. It is defined as an integral of specific heat capacity, as follows:

$$H = \int_0^T C_P dT \tag{4.14}$$

The difficulties in estimating C_P at low temperature can be ignored because gas turbine predictions are based on changes in specific enthalpy, not on absolute values. In fact, it is often sufficient to use the approximation $\Delta H = C_P.\Delta T$ for the transition across a component, treating C_P as if it were a constant defined at inlet conditions.

4.7.4 Thermodynamic Relationships

The basic relationships for thermodynamic systems are summarised, as follows:

Energy

$$T = t + \frac{U^2}{2C_P} \tag{4.15}$$

Mach Number

$$M = \frac{U}{\sqrt{\gamma R t}} \tag{4.16}$$

Gas Laws

$$R = C_P - C_V \qquad \gamma = \frac{C_P}{C_V} \qquad p = \rho R t \tag{4.17}$$

Duct Massflow

$$W = \rho A U \tag{4.18}$$

Isentropic Flow

$$\frac{T}{t} = \left(\frac{P}{p}\right)^{\frac{\gamma-1}{\gamma}} \tag{4.19}$$

Using these relationships, a number of useful derivations can be found, as follows:

$$\frac{T}{t} = 1 + \frac{\gamma-1}{2}M^2 \tag{4.20}$$

$$\frac{P}{p} = \left(1 + \frac{\gamma-1}{2}M^2\right)^{\frac{\gamma}{\gamma-1}} \tag{4.21}$$

$$\frac{U}{\sqrt{T}} = M\sqrt{\gamma R}\left(1 + \frac{\gamma-1}{2}M^2\right)^{-\frac{1}{2}} \tag{4.22}$$

$$\frac{W\sqrt{T}}{AP} = M\sqrt{\frac{\gamma}{R}}\left(1 + \frac{\gamma-1}{2}M^2\right)^{-\frac{\gamma+1}{2(\gamma-1)}} \tag{4.23}$$

4.7.5 Adiabatic and Non-adiabatic Processes

The main approximation in gas turbine models is based on the assumption of *adiabatic* processes. If an amount of heat ΔQ is supplied to a gas, it raises the internal energy of the gas by ΔE (thereby producing a temperature increase) and it enables the gas to do external work ΔW. The first law of thermodynamics expresses the conservation of energy for this situation:

$$\Delta Q = \Delta E + \Delta W \tag{4.24}$$

An adiabatic process is defined by a constant amount of heat, *i.e.* $\Delta Q=0$. Thus, if a gas is made to do work then its temperature decreases and there is no heat flow to or from the external environment. Examples of such a process include those which occur very rapidly or which take place inside a 'perfectly' insulated container.

Turbomachines are essentially adiabatic and the processes of compression and expansion can be idealised by *isentropic* relationships (*i.e.* constant entropy or heat content). In reality, this is not quite true and it is necessary to consider the efficiency of compressors and turbines as well as other thermal effects in the prediction of gas turbine behaviour.

4.7.6 Inter-Component Volume (ICV) Equations

4.7.6.1 Momentum

The acceleration and deceleration of gas is described in terms of the nett pressure force across an ICV. The constitutive relationship derives from Newton's Second Law and is written as

$$\frac{d}{dt}W = \frac{A}{L}(P_{in} - P_{out}) \tag{4.25}$$

where W is massflow, P is stagnation pressure; L is the length and A is the cross-sectional area.

4.7.6.2 Continuity

The capacitance of an ICV defines the rate of change of pressure in response to a mismatch between input and output massflow. At constant temperature the constitutive relationship is

$$\frac{d}{dt}P = \frac{RT}{V}(W_{in} - W_{out}) \tag{4.26}$$

where P is stagnation pressure, T is stagnation temperature, W is massflow, V is volume and R is the gas constant.

4.7.6.3 Energy

Adding heat to a gas passing through an ICV will cause a change in both pressure and temperature. The constitutive relationship is essentially capacitive but with the change in the effort variable of one domain being dependent upon the flow variable from the other. It is described as follows:

$$\frac{d}{dt}P = \frac{\gamma - 1}{V}(E_{in} - E_{out}) \tag{4.27}$$

$$\frac{d}{dt}T = \frac{T_{mean}}{P}\left(\frac{d}{dt}P - \frac{RT_{mean}}{V}(W_{in} - W_{out})\right) \tag{4.28}$$

where P is stagnation pressure, T is stagnation temperature, W is massflow, E is enthalpy flow, V is volume, R is the gas constant and γ is the ratio of specific heat capacities at constant pressure and constant volume. The enthalpy flow is composed as follows:

$$E_{in} = W_{in}H_{in} + FCV.\eta.WF \tag{4.29}$$

$$E_{out} = W_{out}H_{out} \tag{4.30}$$

where H is specific enthalpy (*i.e.* enthalpy per unit mass), WF is fuel flow, *FCV* is the fuel calorific value and η is the combustion efficiency.

4.7.7 Ducts

The pressure drop associated with flow through a duct is approximated by the following expression:

$$\frac{\Delta P}{P} = PLF\left(\frac{W_{in}\sqrt{T_{in}}}{P_{in}}\right)^2 \tag{4.31}$$

where *PLF* is called the 'pressure loss factor'.

4.7.8 Compressors

The constitutive relationship for compression is usually depicted as lines of constant aerodynamic speed on graphs of [i] inlet flow function against pressure ratio and [ii] steady-state efficiency against pressure ratio. Rotational speed will be written as Ω if expressed in $rad.s^{-1}$ and as N if expressed in rpm; the aerodynamic speed is usually defined as $N/\sqrt{T_{in}}$ and the flow function as $W_{in}\sqrt{T_{in}}/P_{in}$.

Compression efficiency is revealed in the relationship between pressure ratio (*PR*) and temperature ratio (*TR*) across the compressor. This can be conveniently expressed either in the form:

$$TR = 1 + \frac{1}{\eta_{isen}}\left(PR^{(\gamma-1)/\gamma} - 1\right) \tag{4.32}$$

where η_{isen} is the isentropic efficiency and γ is the ratio of specific heat capacities at constant pressure, or in the form:

$$TR = PR^{(n-1)/n} \tag{4.33}$$

where η_{poly} is the *polytropic* (or small-stage) efficiency and n is the adiabatic 'compression index':

$$\frac{n-1}{n} = \frac{1}{\eta_{poly}}\left(\frac{\gamma-1}{\gamma}\right) \tag{4.34}$$

Knowing the temperature ratio across the compressor and the inlet temperature T_{in}, it is then a simple matter to calculate the outlet temperature T_{out}. From (4.14), the corresponding values of specific enthalpy, H_{in} and H_{out} can be found. From this, and the massflow W, the so-called 'compressor work' (which is in fact *power*) is calculated as:

$$J = W\,(\,H_{out} - H_{in}\,) \tag{4.35}$$

Compressor surge is a limiting condition which is associated with sudden drops in delivery pressure and violent aerodynamic flow pulsations which are transmitted throughout the gas turbine. It occurs because, for a given rotational speed, the blade setting angles and blade geometries are optimised for a particular flow $W\sqrt{T}/P$. Theoretically, at that condition, the pressure ratio attains its maximum value; at any other condition, it must be less than maximum. When the machine is operating at or near maximum, a sudden reduction in flow will result in a sudden reduction in pressure ratio and, if the downstream components do not respond fast enough, the airflow can transiently reverse direction. This reduces the inlet pressure and so increases the pressure ratio. The ensuing process is unstable.

The Surge Pressure Ratio (*SPR*) defines the point at which surge will occur for a given flow condition. The relative separation of the pressure ratio during normal operation (*PR*) and that associated with compressor surge is estimated by a so-called Surge Margin (*SM*), which is defined by:

$$SM = 1 - \frac{PR}{SPR} \tag{4.36}$$

4.7.8.1 LP Compressor

In turbofan engines the flow through the Low-Pressure (LP) compressor (otherwise known as the 'fan') is split into two flow paths, designated as *core* and *bypass*, drawn from the inner and outer sections of the fan. It is noted compressor blade speed is significantly higher at the tip than the hub. The use of so-called 'transonic fans' for LP compression means the tip Mach number is supersonic relative to the flow and, accordingly, the outer part of the fan will suffer losses associated with aerodynamic shocks. Thus, if the fan is modelled as a split compressor, there will be a discrete difference between the pressure ratios and compression efficiencies of the inner and outer parts (rather than a smooth distribution). A suitable estimate is given by:

$$PR_{core} - PR_{bypass} = 0.060\,(\,PR_{core} - 1.9\,) \tag{4.37}$$

$$\eta_{core} - \eta_{bypass} = 0.060\,(\,PR_{core} - 1.9\,) \tag{4.38}$$

where *PR* denotes pressure ratio and η denotes *isentropic* efficiency.

As a consequence of split flow, compressor work must now be calculated as a combination of core and bypass compression. By analogy with (4.30), the calculation becomes:

$$J = W_{core}\,(\,H_{core} - H_{in}\,) + W_{bypass}\,(\,H_{bypass} - H_{in}\,) \tag{4.39}$$

where the total massflow is $W = W_{core} + W_{bypass}$. The bypass ratio is defined by:

$$BR = \frac{W_{bypass}}{W_{core}} \tag{4.40}$$

The effective surge margin is determined as a combination of pressure ratios and flow conditions. It is convenient to form an aggregate surge margin in this case, as follows:

$$SM = \left(\frac{W_{core}}{W}\right)\left(SM_{core} + BR.SM_{bypass}\right) \tag{4.41}$$

4.7.8.2 HP Compressor

Typically, the High-Pressure (HP) compressor provides the bulk of airbleed offtake for external services and for cooling and sealing within the engine. For current purposes, it is sufficient to assume that bleed will be drawn either compressor delivery (*i.e.* at the highest pressure) or from an intermediate stage. This means that the HP compressor could be modelled as two compressors in series. Alternatively, a single compressor model could be used with a pressure ratio imposed for intermediate bleed offtake; a reasonable approximation would be to assume that each stage of compression has the same pressure ratio. In this case, the calculation of compressor work must account for the change in massflow resulting from the bleed offtake.

4.7.9 Combustion

Combustion gives a constant heat release per unit mass of fuel burnt. It adds a large amount of heat to the gas, perhaps resulting in a 1000K temperature rise or more at high power settings. At steady state, the energy balance is expressed as follows:

$$(W_{in} + WF) H_{out} = W_{in}H_{in} + WF.\eta.FCV \tag{4.42}$$

where combustion efficiency η is assumed constant for a given combustion chamber geometry. The pressure drop can be modelled in the same way as for a duct [*cf.* section 4.7.7], except that *PLF* now has two components, namely 'cold' (or 'aerodynamic') loss which is constant and 'hot' (or 'fundamental') loss which varies with temperature. Typically, the overall loss is very close to 5 *per cent.*

The model of a reheat system is essentially the same as for a combustion chamber. The peak combustion efficiency is defined as a function of inlet temperature and pressure; the actual efficiency is determined a s a function of fuel/air ratio. The pressure loss caused by the intrusion into the gas flow by the reheat burner and its attachments is assumed to be about 1 *per cent.*

4.7.10 Turbines

The constitutive relationship for expansion through turbines has the same basic form as for compression. Turbine efficiency is revealed in the relationship between pressure ratio (*PR*) and temperature ratio (*TR*), which can be conveniently expressed either in the form:

$$TR = 1 + \eta_{isen}\left(PR^{(\gamma-1)/\gamma} - 1\right) \tag{4.43}$$

where η_{isen} is the isentropic efficiency and is the ratio of specific heat capacities at constant pressure, or in the form:

$$TR = PR^{(n-1)/n} \tag{4.44}$$

where η_{poly} is the *polytropic* (or small-stage) efficiency and n is the adiabatic 'expression index':

$$\frac{n-1}{n} = \eta_{poly}\left(\frac{\gamma-1}{\gamma}\right) \tag{4.45}$$

Knowing the temperature ratio across the compressor and the inlet temperature T_{in}, it is then a simple matter to calculate the outlet temperature T_{out}. From (4.14), the corresponding values of specific enthalpy, H_{in} and H_{out} can be found. From this, and the massflow W, the so-called 'turbine work' is calculated as:

$$J = W (H_{in} - H_{out}) \tag{4.46}$$

4.7.11 Nozzles

Nozzle conditions determine the overall power output of an engine, in the form of gross thrust and shaft power. For a convergent nozzle (assuming ideal expansion), the Mach number of the fully expanded jet is

$$M = \sqrt{\frac{2}{\gamma - 1}\left[\left(\frac{P}{p_0}\right)^{(\gamma-1)/\gamma} - 1\right]} \tag{4.47}$$

where *NPR* is the nozzle pressure ratio, P/p_∞, *i.e.* the ratio of *stagnation* pressure at the nozzle entry and the ambient *static* pressure. The flow in the nozzle throat will be sonic unless the fully expanded Mach number is less than one. The 'critical' pressure ratio required for the nozzle to be choked is given by

$$CPR = \left(\frac{\gamma - 1}{2}\right)^{\gamma/(\gamma-1)} \tag{4.48}$$

If the nozzle is choked than the static pressure inside the nozzle is $p=P/CPR$, otherwise it is $p=p_\infty$ (*i.e.* ambient pressure). Therefore, the value of static temperature in the nozzle throat determines the local speed of sound and thus the local flow velocity U.

The gross thrust delivered at nozzle exit is composed of pressure force and momentum force components:

$$XG = A.(p - p_\infty) + W.U \tag{4.49}$$

where is A is nozzle area. Note that, when the nozzle is unchoked, there is no pressure force. Also note:

$$\frac{XG}{A} = \left(1 - \gamma M^2\right)p - p_\infty \tag{4.50}$$

referring back to basic thermodynamic relationships (4.16), (4.17) and (4.18).

Nozzle performance is represented by a thrust coefficient C_X and a discharge coefficient C_D, expressed as functions of nozzle contraction angle and nozzle pressure ratio P/p_∞. The discharge coefficient defines the effective reduction in nozzle area due to flow restriction: the thrust coefficient define the effective reductive in thrust due to non-axial flow components. The modified calculation of gross thrust is:

$$XG = C_x\left[C_D A.(p - p_\infty) + W.U\right] \tag{4.51}$$

The flow through the nozzle is constrained as follows:

$$\frac{W\sqrt{T}}{P} = C_D A.M\sqrt{\frac{\gamma}{R}}\left(1 + \frac{\gamma - 1}{2}M^2\right)^{-\frac{\gamma+1}{2(\gamma-1)}} \tag{4.52}$$

referring back to (4.23).

4.7.12 Thermal Effects

4.7.12.1 Overview

There are a number of thermal effects which are relevant to transient performance prediction in gas turbines [Maccallum 1973, 1976, 1978; Maccallum & Pilidis 1985]. These can be summarised as:

- **Non-adiabatic flows in compressors and turbines**
- **Changes in boundary layer development on blade aerofoils**
- **Changes in boundary layer development on blade end walls**
- **Changes in tip/seal clearances**

These have a significant effect on component responses and therefore on the overall response of the engine. This section will deal with the first three effects because they can be modelled in a relatively generic way; the fourth will be specific to particular engines and will require detailed definitions of engine configuration, geometry and structure.

4.7.12.2 Temperature Estimates

Gas temperature is approximated to an 'average' value in heat transfer calculations for compressors and turbines. This corresponds to a hypothetical intermediate rotor stage which can be taken to represent the complete assemblage of rotors. For compressors, it is sufficient to adopt the arithmetic mean:

$$T_g = \frac{T_{in} + T_{out}}{2} \tag{4.53}$$

For stators and turbines, it is found that a better estimate is given by:

$$T_{mean} = \left[1 + RTDF\left(1 - \frac{1}{TR_{energy}}\right)\right] T_{in} \tag{4.54}$$

$$T_{mean} = \left[1 + RTDF\left(1 - \frac{1}{TR_{energy}}\right)\right] \frac{T_{in} + T_{out}}{2} \tag{4.55}$$

respectively, where *RTDF* denotes the radial temperature distribution factor and TR_{energy} denotes the temperature ratio associated with energy input across the combustors. This is critical because, the absence of any cooling, this is the temperature which the material will reach; it is especially critical for the HP stator and turbine because their thermal properties ultimately limit the performance of the entire gas turbine.

Blade cooling is applied to stators and turbines, and the resulting gas temperature is determined by heat exchange between blade material and coolant. The highest temperature is T_{mean} and the coldest is $T_{coolant}$, and so the maximum possible cooling effect is T_{mean}-$T_{coolant}$; in practice, this is never achieved and the resultant gas temperature is approximated by:

$$T_g = T_{mean} - EC(T_{mean} - T_{coolant}) \tag{4.56}$$

where *EC* is the *cooling effectiveness* and the coolant is provided by airbleed from the HP compressor [*cf.* section 4.7.8.2]. Because the cooling fraction in modern can be in excess of 20 *per cent* of the core massflow, significant design effort is devoted to maximising the cooling effectiveness and minimising the blade cooling requirements. This is done by using the coolest air available, at a pressure that just exceeds that of the air flowing over the turbine.

4.7.12.3 Boundary Layer Characteristics

Heat transfer characteristics are determined by the boundary layer mechanisms that apply over the compressor and turbine blades. It is reasonable to use a flat plate correlation for developing laminar and turbulent boundary layers. The average heat transfer coefficient for a laminar layer of length L is

$$h_{lam} = 0.664 \frac{k}{L} \sqrt[3]{Pr} \sqrt{Re} \tag{4.57}$$

and for a turbulent layer of length L is

$$h_{turb} = 0.037 C_P \rho U (Re)^{-1/5} (Pr)^{-2/3} \tag{4.58}$$

where Re is the Reynolds number and Pr is the Prandtl number, as defined by

$$Re = \rho U \frac{L}{\mu} \tag{4.59}$$

$$Pr = \frac{\mu}{k} C_P \tag{4.60}$$

respectively. The parameter μ is the absolute viscosity of the gas (*i.e.* air with or without combustion products) and k is its thermal conductivity. Note that μ, k and C_P are all dependent on temperature and, certainly, C_P is also dependent on fuel-air ratio. Because there can be a significant difference between the aerofoil blade temperature T_a and the freestream gas temperature T_g, these parameters are evaluated at the mean value, the so-called *film temperature*.

For a compressor, it is proposed that the average heat transfer coefficient is given by

$$h_C = 0.25\, h_{lam} + 0.75\, h_{turb} \tag{4.61}$$

assuming that the boundary layer on the pressure surface is turbulent along its length and initially laminar, becoming turbulent, along its suction surface. In general, experimental evidence suggests that this method of estimation must be augmented by 60 *per cent* for compressors and 80 *per cent* for turbines in order to bring them in line with typical observations. Using direct experimental results, a turbine might be better represented by the following empirical expression:

$$h_T = 0.235\frac{k}{L}(Re)^{0.64} \tag{4.62}$$

For convenience throughout, assume that L=c, the blade chord length.

The earlier use of ρU can be replaced by W/A (*i.e.* massflow per unit area) where A denotes the cross-sectional flow area between blades. For an average blade setting angle β, this can be approximated as $A=2\pi r.b.\cos\beta$, where b is the blade height and r is the mean radius of the blade aerofoils.

4.7.12.4 Power Transfer Estimates

The overall heat transfer from a compressor or turbine to the gas flowing through it is calculated in three parts, namely *aerofoil*, *shroud* and *platform*. This represents the blade construction, the retaining structure at the blade tip and the mounting structure at the blade root, respectively. A set of power transfer coefficients can be derived as follows:

Aerofoil
$$X_a = h.S_a \tag{4.63}$$

Shroud
$$X_s = h.S_s \tag{4.64}$$

Platform
$$X_p = h.S_p + \sqrt{2L.k.A_{xa}.h}\,\frac{T_p - T_a}{T_p - T_g} \tag{4.65}$$

where the same heat transfer coefficient (h), with units of $W.m^{-2}.K^{-1}$, has been assumed for all three components. Note that the power transfer coefficients have with units of $W.K^{-1}$, *i.e.* entropy. The additional term in the platform calculation assumes that the aerofoil blade acts as a cooling fin for the platform. It is estimated that X_p should be increased by a factor of 2 in order to account for the thermal link between the platform and rotor disc. The resulting heat transfer from the rotor to the gas is given by

$$\Delta Q = (T_a\text{-}T_g)X_a + (T_s\text{-}T_g)X_s + (T_p\text{-}T_g)X_p \tag{4.66}$$

From this and from the calculations of compressor/turbine work in (4.35), (4.39) and (4.46), the heat/work transfer ratio can be derived as follows:

$$F = -\Delta Q/J \tag{4.67}$$

where ΔQ is the heat transfer to the gas and J is the work transfer to the gas. Note that J is positive for turbines and negative for compressors; note further that the value of F varies dynamically.

4.7.12.5 Compressor/Turbine Performance

As a direct consequence of non-adiabatic flows, there is a change in the efficiencies of compressors and turbines. This can be introduced most conveniently as a re-definition of compression index (4.34) and expansion index (4.45), as follows:

$$\frac{n-1}{n} = \frac{1-F}{\eta_{poly}}\left(\frac{\gamma-1}{\gamma}\right) \tag{4.68}$$

$$\frac{n-1}{n} = \eta_{poly}(1-F)\left(\frac{\gamma-1}{\gamma}\right) \tag{4.69}$$

respectively, where F is the heat/work transfer ratio defined in (4.64).

The effects of heat transfer have important implications for predicting the transient performance of compressors. There is an effective change in speed given by:

$$\frac{\Delta N}{N} = c_1 \frac{T_a - T_g}{T_g} + c_2 \frac{\Delta Q}{E_{in}} \tag{4.70}$$

and an effective change in the surge pressure ratio (*SPR*) given by:

$$\frac{\Delta SPR}{SPR - 1} = c_3 F \tag{4.71}$$

As an illustration, the coefficients can be assigned nominal values c_1=-0.07, c_2=-0.07 and c_3=0.35. It is worth noting that an equivalent procedure is not required for turbines because, firstly, speed has little appreciable effect on turbine performance and, secondly, turbines do not surge!

4.7.12.6 Transient Response of Heat Transfer Mechanisms

The dynamics associated with heat transfer produce a first-order temperature response in the aerofoil, shroud and platform. The respective time constants are determined by the power transfer coefficients X_a, X_s and X_p, together with the mass (m) and specific heat (C_p) of each component. The relevant equations for the *aerofoil* are as follows:

$$\frac{d}{dt} T_a = \frac{1}{\tau_a} \left(T_g - T_a \right) \tag{4.72}$$

where T_a is the aerofoil temperature, T_g is the gas temperature (the *set point*) and τ_a is the time constant which is given by:

$$\tau_a = \left[m.C_P \right]_a / X_a \tag{4.73}$$

The thermal behaviour of the other components is defined by analogous expressions and it should be recognised that, under steady-state conditions, the temperature of the whole blade assembly is equal to the calculated gas temperature (incorporating the effects of cooling, where appropriate).

4.7.13 Component Class Definitions [*cf.* Figure 4.6]

4.7.13.1 Basic Elements

Bond

The standard bond definition for thermofluids is defined by a structure of two bonds, belonging to Massflow and Enthalpy domains, respectively. This is a minimum provision which will suffice to illustrate the basic principles. Note that composite flow will probably exist under normal circumstances, *e.g.* turbomachines work on air which contains water vapour and fuel/air combustion products.

'0' Interface

Flow junctions need some modification to be used for convected flow because the enthalpy associated with different flow paths is not independent of the massflow along those paths. This requires different causal constraints, such that enthalpy flow is *merged* and temperature is *branched*. Consistency between massflow and enthalpy flow is maintained by a **Convector** bond [Section 4.7.13.4]. Note that the structured bonds are exploded in order to show the underlying causal constraints.

4.7.13.2 'Turbomachine' Sub-Library

Rotor

The **Rotor** component is the standard building block for a turbomachine. It consists of a through shaft and a multi-port component which holds the **#Rotor** CR [described in this section]. The outlet feeds a **HeatSink** which allows heat conduction to the rotor blades via a **HeatTransfer** interface [Section 4.7.13.3].

If non-adiabatic effects are important then heat conduction must be incorporated. Also, since heat storage is happening, the **HeatTransfer** must be implemented as a **ThickWall**. Heat transfer coefficients assigned to the **R** components and a thermal capacity assigned to the heat store **C**. Although not considered in detail here, the choice of heat transfer coefficient and thermal capacity must replicate the dynamics given in (4.72) for blade aerofoils and (analogously) for blade platforms and shrouds. Note that aerofoil temperature is required in **#Rotor**.

#Rotor Constitutive Relationship

This is a CR which contains the common equations for compressors [Section 4.7.8] and turbines [Sections 4.7.10]. Temperature change is given by (4.32) and (4.43), respectively, and mechanical power is given by (4.35) and (4.46), respectively. There is an empirical relationship (a 'map') between pressure ratio (not pressure difference), normalised shaft speed $N/\sqrt{T}$ and normalised massflow $W\sqrt{T}/P$. Also, there is a map between pressure ratio and efficiency η. Note that, by convention, the values of pressure ratio and efficiency must be inverted between compressors and turbines if a common formulation is to be applicable.

If non-adiabatic effects are important then the heat/work transfer ratio of (4.64) must be incorporated as shown in (4.68) and (4.69) and, in turn, the resulting change in effeciency must be mapped onto whatever form is used in the compressor/turbines maps. Recall, also, that there is an effective change in shaft speed as defined by (4.70) which depends on aerofoil temperature T_a, gas temperature T_g and heat transfer ΔQ.

Stator

In comparison with rotors, the standard lumped-parameter **Stator** is trivial. For purposes of high-level modelling it is sufficient to treat it as a **HeatSink** [Section 0] which allows heat conduction to the stator blades.

SplitCompressor

A split compressor represents a fan, with separate exit paths for inner and outer portions, *i.e.* near the hub and tip respectively.

TandemCompressor

A tandem compressor corresponds broadly to the configuration used for compression incorporating bleed offtakes. This would be typical of an HP compressor in a twin-spool engine.

Turbine

A special turbine component is also useful for gas turbine engines. This comprises a Stator and a Rotor with ducted flow for blade cooling. Note that the cooling effect as approximated in (4.56) requires a value of mean temperature and the temperature ratio across the combustor!

Combustor

The steady-state energy balance of a combustor is given in (4.42). For dynamic modelling, this must be used in conjunction with an Energy ICV component [Section 4.7.13.5] positioned immediately afterwards, which will account for the total energy input versus the swallowing capacity of the next component downstream, in this case a Turbine [discussed in this section].

Duct

A Duct component accounts for pressure loss as approximated in (4.31). Note that, since the thermal side of the model contains the total enthalpy flow, the heat generated by hydraulic friction (assuming no transfer to the external environment) is already accounted for and, thus, the two sides of this component model are separate.

Bond Duct

For the sake of compact models, it is convenient to bury the **Duct** component inside a composite bond. It is likely to be a frequently used feature and one that might reasonably be thought of as a link.

Nozzle

The Nozzle component is simply an encapsulation of the #Nozzle CR, together with a controlling signal injected through the **[control]** port which determines the nozzle area. The resulting thrust appears at **[thrust]**.

#Nozzle Constitutive Relationship

Standards equations for nozzles are given in (4.51) and (4.52). The first determines thrust and the second determines massflow.

4.7.13.3 'Conduction' Sub-Library

HeatSink

A heat sink provides a means of adding and subtracting heat from a flow path without affecting the massflow or pressure. It is an idealised component that separates pressure and temperature effects and is applicable to heat transfer associated with high velocity flow or with low compressibility.

HeatStore

This functions as a heat transfer mechanism to/from a thermal capacity. Note that this component belongs to the Enthalpy domain.

HeatTransfer Interface

Heat transfer is achieved via conduction across a 'wall'. A thin wall model has no thermal capacity and is driven by a simple heat transfer coefficient and differential temperature across an **R** component. A thick wall has thermal capacity, as defined by a **C** component, and transfer of heat on each side of the **C** is determined by independent **R** components. Note that the **HeatTransfer** is purely thermal, *i.e.* it does not involve fluid transfer.

HeatExchanger

The **HeatExchanger** component is a packaged version of **HeatTransfer** with connections to hot and cold fluid flow paths, accessed via **HeatSink** components. Although not considered here, the usual arrangement for a lumped-parameter model of a heat exchanger is to drive the heat transfer from 'hot' to 'cold' by the difference in mean temperature on the two sides.

4.7.13.4 'Convector' Bond Interface

Heat convection implies redundancy in the information content of a thermofluid model, as discussed in Section 4.3.2. In order to ensure consistency between massflow, temperature and enthalpy (for given specific heat capacity), a constraint is applied between hydraulic and thermal bonds via a modulated resistance. Massflow is measured directly from the hydraulic side and multiplies the nominal value of resistance contained in the **R** component. Two different versions of this are provided to cover a change in causality, plus a trivial version to cover direct propagation. Note that this functionality addresses straightforward point-to-point compatibility issues and it is convenient to package this as a composite bond, rather than a composite component.

4.7.13.5 'ICV' Interface

Inter-Component Volumes (ICVs) encapsulate the dominant dynamic mechanisms within a thermofluid system model. It is convenient to specify an interface and allow the particular implementation to be determined by context. The Momentum ICV [Section 4.7.6.1] is represented by an I component on the hydraulic side, with an equivalent 'mass' of L/A as in (4.25). The Continuity ICV [Section 4.7.6.1] is represented by a C component on the hydraulic side, with an equivalent 'capacitance' of V/RT as in (4.26). The Energy ICV [Section 4.7.6.1] is represented by two C components, exchanging state information between the massflow and enthalpy sides of the model, as shown in (4.27) and (4.28).

4.7.13.6 'Valve' Interface

A valve is established as a **Generic/Switch** applied to the hydraulic flow path, with a convected heat component determined by a Convector bond. Note that the valve acts as a modulated resistor and that a signal drives the modulation. Flow modulation is achieved by a **Generic/Valve** component while the behaviour of a non-return valve is represented by a **Generic/NonReturn** component. As an extension, a control valve (**CV**) component is defined which specialises a **Generic/Branch** within the thermofluid domain.

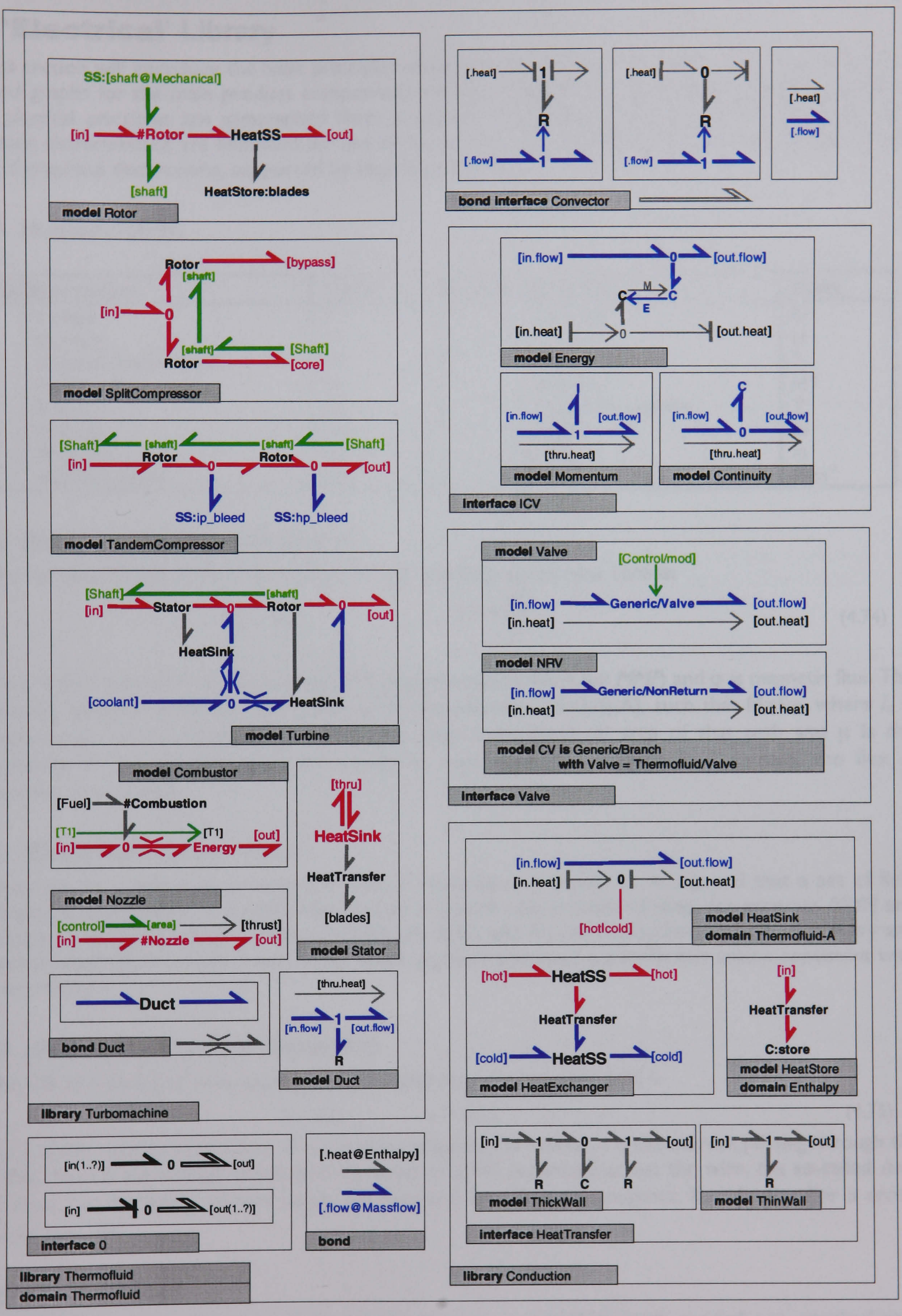


Figure 4.6 'Thermofluid' Library Components

4.8 'Electrical' Library

This section will introduce the basic principles underlying electrical power with a view to defining a set of bond graphs for the main product components which go together to form electrical power systems. Basic physical principles are summarised from a number of standard texts [*e.g.* Pallett 1981] and the dominant characteristics are identified for use in component CRs. Component classes are defined in the form of graphical declarations, supported by text descriptions.

4.8.1 Nomenclature

Symbol/Description		*Units*	*Symbol/Description*		*Units*
V	Voltage	V	R	Resistance	Ω
i	Current	A	L	Inductance	H
M	Magneto-motive force	AT	C	Capacitance	F
φ	Magnetic flux	W	S	Reluctance	H^{-1}
F	Force	N	B	Magnetic flux density	T
v	Velocity	$m\ s^{-1}$	*l*	Length	m
τ	Torque	Nm	r	Radius	m
ω	Angular velocity	$rad\ s^{-1}$	μ	Permeability	$H\ m^{-1}$

4.8.2 Electro-magnetic Interaction

The constitutive relationship for a set of N field windings around the core is:

$$M = N.i_F \qquad , \qquad \mathbf{V} = N\frac{d\varphi}{dt} \tag{4.74}$$

where i_F is field current, V is field voltage, M is magnetomotive force (or *MMF*) and φ is magnetic flux. The relationship between *MMF* and flux is defined by the reluctance $S=l_F/(\mu A)$, such that $M=S\varphi$, where l_F is the path length of the magnetic field, A is the mean cross-sectional area of that path and μ is the permeability of the magnetic medium, usually an iron core. For a given field current, the flux is determined as $\varphi=(N/S)i_F$.

4.8.3 Electrical Transformers

From the basic principles of electro-magnetic interaction, it should be recognised that a set of field windings acts as a gyrator. A transformer consists of a core with primary windings (to generate *MMF*) and secondary windings (in which a current is induced). If N_1 and N_2 are respective numbers of primary and secondary windings, the mutual inductance of the gyrators produces a transformer action consistent with the ratio N_2/N_1.

4.8.4 Electro-mechanical Interaction

The CR of a electrical wire passing at right angles across a magnetic field is

$$F = B.l.i \quad , \qquad e = B.l.u \tag{4.75}$$

where u is the velocity of the wire, F is the force opposing its motion, i is the current passing through the wire and the e is the voltage (electromotive force or *EMF*) generated across the wire, the so-called *back EMF*. Also, *l* is the length of wire inside the field and $B=\varphi/A$ is the magnetic flux density (for a cross-sectional area, A).

4.8.5 DC Machines

Extending the basic electro-mechanical interaction to an armature (with n windings) attached to a drive shaft, the CR becomes:

$$\tau = B.2nl.r.i \quad , \qquad e = B.2nl.r.\omega \tag{4.76}$$

where τ is the motor torque, ω is the shaft speed and r is the armature radius. This shows that the motor acts as a gyrator modulated by a field current Ki_F, where $K=(N/SA).2n\mathit{l}.r$.

4.8.6 AC Machines

From the same principles, a single-phase AC machine is modulated by field and shaft speed, $K.i_F.\sin\omega t$, where the electrical frequency is equal to the rotational frequency of the shaft. This model assumes a two-pole rotor. With a four-pole rotor, the modulation becomes $K.(i_{F1}.\sin\omega t + i_{F2}.\cos\omega t)$. Note that the electrical frequency will be doubled if the field current is modulated such that $i_{F1}=i_F.\cos\omega t$ and $i_{F2}=i_F.\sin\omega t$. Three-phase machines have a single mechanical connection and three separate electrical connections with overlapping waveforms with a phase separation of $2\pi/3$ (or 120°). Note that the doubling in electrical frequency resulting from generator excitation maintains the same relative phase.

4.8.7 Component Class Definitions [*cf.* Figure 4.5]

Core

The Core component is the interface between electrical and magnetic domains resulting from winding an iron core (as in a transformer). Note that the two GY components in series are equivalent to a single TF component in the electrical path and that the combination of a GY with magnetic permeance (a C component which is not shown here) is equivalent to an electrical inductance.

Filter

Filtering is provided as part of the power supply conditioning. Here, it is a simple first-order lag determined by the combined effect of a resistance (R component) and an inductance (I component).

StarCoupling

A particular form of star coupling is applied to the electrical connections to three-phase AC machines. As shown, the arrangement has each phase of supply coupled by a simple Star coupling as defined in the **Generic** library [Section 4.5].

Gyrator Interface

A simplistic model of an AC machine splits the mechanical power into three paths and gyrates each in order to produce/consume single-phase power (shown by a vector GY component); the three-phase power passes through a StarCoupling in order to give the correct electrical arrangement.

Rectifier

Rectification from AC to DC power is achieved by the use of a Filter component in series with a NonReturn component as defined in the **Generic** library [Section 4.5]. Note that, here, the principle of domain specialisation is illustrated by the introduction of a **DcElectric** domain

TRU

Using the basic library components above, a composite model of a transformer-rectifier unit can be constructed with relative ease, as shown. It is worth noting, however, that the difficulties of drawing pictorial representations increase rapidly with the number of components because of the resulting number of connections. The TRU model does not make use of different *views* because its function should be sufficiently familiar in order to be understandable in 'flat' form. Realistically, models of any greater complexity would benefit from being partitioned in some manner.

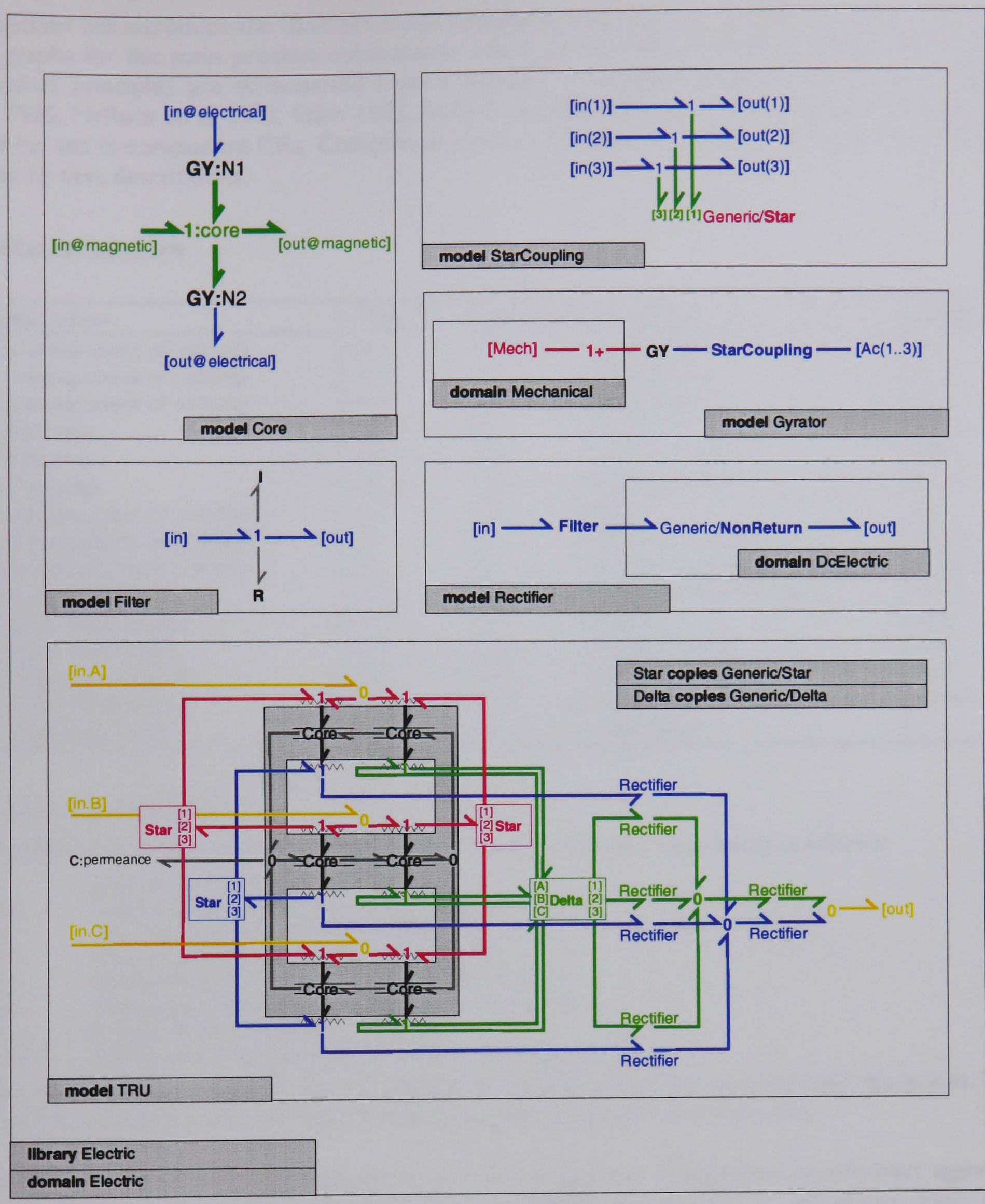


Figure 4.7 'Electrical' Library Components

4.9 'Flight Dynamic' Library

This section will introduce the basic principles underlying electrical power with a view to defining a set of bond graphs for the main product components which go together to form electrical power systems. Basic physical principles are summarised from a number of standard text on atmospheric flight [*e.g.* Babister 1980, McRuer *et al.* 1973, Etkin 1972, Rolfe & Staples 1986] and the dominant characteristics are identified for use in component CRs. Component classes are defined in the form of graphical declarations, supported by text descriptions.

4.9.1 Nomenclature

Symbol/Description		*Units*	*Symbol/Description*		*Units*
u	x-component of *velocity*	m s^{-1}	X	Axial force	N
v	y-component of *velocity*	m s^{-1}	Y	Lateral force	N
w	z-component of *velocity*	m s^{-1}	Z	Vertical force	N
p	Roll rate	rad s^{-1}	L	Rolling moment	Nm
q	Pitch rate	rad s^{-1}	M	Pitching moment	Nm
r	Yaw rate	rad s^{-1}	N	Yawing moment	Nm
x	x-component of *position*	m	m	Mass	Kg
y	y-component of *position*	m	I	Moment of inertia	Kg m^2
z	z-component of *position*	m	μ	Angular momentum	Kg m s^{-1}
ψ	Euler angle (azimuth)	rad	g	Gravitational constant	m s^{-2}
θ	Euler angle (pitch)	rad	V	Velocity	m s^{-1}
φ	Euler angle (roll)	rad	α	Angle of attack	rad
χ	Flight path heading	rad	β	Angle of sideslip	rad
γ	Climb/dive angle	rad	Ω	Turn rate	rad s^{-1}
λ	Flight path bank	rad	n	Load factor	

4.9.2 Axis Definitions

Many different axis systems are used in aircraft models, the main ones being as follows:

- **earth** Local positional reference
- **airframe** Geometric reference for aircraft components
- **engine** Geometric reference for engine components
- **body** Measurement reference for aircraft dynamics
- **inertia** Principal axes of inertia referred to centre of gravity
- **airflow** Instantaneous airflow given by angles of attack and sideslip
- **thrust** Axis set based on propulsive thrust line
- **flightpath** Instantaneous orientation relative to flight path trajectory

The names given to these systems are the subject of convention and can vary between disciplines. For this reason, and to ensure a consistent interpretation, a brief description is given below.

Earth axes provide the absolute reference for aircraft position. Navigation is performed against a flat earth fixed in space and motion through space is calculated with respect to orthogonal axes, pointing North, East and Down (parallel with the gravity vector).

Airframe axes define a coordinate frame for aircraft geometry. This provides the location of structural components, engine, equipment, sensors, effectors and the pilot (typically the pilot eye position).

Engine axes define a coordinate frame for engine geometry, aligned with the spool axes, and for thrust calculation (which is especially convenient for aircraft equipped with vectored thrust devices).

Body axes provide the baseline for the calculation of aircraft flight dynamics. They are centred at the aircraft CG and their orientation is fixed within the airframe. The spatial orientation of the aircraft is defined with respect to the earth through a sequence of rotations in azimuth ψ, pitch θ and bank φ. The parameters (ψ,θ,φ) are called Euler angles.

Inertia axes define the principal axes of inertia of the aircraft which essentially decouple the inertial effects on aircraft motion. These resolve the components of applied forces and determine how the aircraft rotates under the action of applied moments. The equations of motion for the aircraft are defined with respect to these axes and, if another a different axis system were required, the inertia couplings would have to be calculated explicitly.

Airflow axes (otherwise known as Stability axes) define the reference frame for calculating aerodynamic forces and moments and their rates of change with respect to airflow (so-called stability derivatives) and with respect to control surface deflection (so-called control derivatives). The predominant influences are airspeed V, incidence (or angle of attack) α and sideslip β. These define the relative velocity vector.

Flight Path axes define the instantaneous trajectory of the aircraft along a curve in space. Velocity is tangential to the curve and centripetal acceleration is normal to it. The tangent is specified by heading κ and climb/dive angle γ (which is sometimes called flight path angle). Note that this requires the absolute velocity vector of the aircraft: relative velocity would define the flight path relative to the airmass.

Axis System	Origin	X-Axis	Y-Axis	Z-Axis
earth	Point of Reference	North	East	Gravity
airframe	Aircraft RP	Rearward	Starboard	Upward
engine	Engine RP	Rearward	Starboard	Upward
body	Aircraft CG	Forward	Starboard	Downward
inertia	Aircraft CG	Forward PAI	Starboard PAI	Downward PAI
airflow	Aircraft CG	Rel. Velocity Vector	Sideforce	*Negative* Lift
flightpath	Aircraft CG	Tangent Vector	Binormal Vector	Normal Vector

4.9.3 Axis Transformations

Transformations are needed in order to align the applied forces and moments with different frames of reference. These can be constructed as a sequence of elementary rotations of the following types:

$$\mathbf{R}_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos\theta & sin\theta \\ 0 & -sin\theta & cos\theta \end{pmatrix}, \quad \mathbf{R}_y = \begin{pmatrix} cos\theta & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & cos\theta \end{pmatrix}, \quad \mathbf{R}_z = \begin{pmatrix} cos\theta & sin\theta & 0 \\ -sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{4.77}$$

where $R_x(\theta)$ represents a right-handed rotation of θ radians about the x-axis, and so on. These can be combined in various ways in order to build any compound rotation but (to state the obvious) note that the order in which elementary rotations are applied is crucial. A number of key transformations are shown in the following table:

Axis Transformation	Parameters	Specification
earth2body	Euler Angles (ψ,θ,φ)	$R_x(\varphi)R_y(\theta)R_z(\psi)$
earth2flight path	Heading κ, Climb/Dive Angle γ	$R_y(\gamma)R_z(\kappa)$
body2airflow	Incidence α, Sideslip β	$R_z(\beta)R_y(\alpha)$

Distinct from many textbook descriptions, the policy here is not to evolve explicit expansions in order to transform from one axis system to another. The bond graph method is underpinned by symbolic algebra and therefore this can be handled in software.

4.9.4 Coordinate Transformations

Forces and moments are invariably calculated at many reference points and then brought together at a common point, usually the centre of gravity. Also, velocities and angular rates are calculated at the centre of gravity and are translated into equivalent velocities elsewhere. Typical reference points include:

- cg Aircraft centre of gravity
- wing Wing moment reference centre
- engine Engine moment reference centre

The convention here will be use word associations, such as cg->wing, in order to denote the mathematical operation of translating coordinates from one point to another.

For a given moment reference centre (mrc) the displacement matrix has the form:

$$\text{cg->mrc} = \begin{pmatrix} 0 & -\Delta z & -\Delta y \\ \Delta z & 0 & -\Delta x \\ \Delta y & \Delta x & 0 \end{pmatrix} \tag{4.78}$$

where $\Delta x = x_{mrc} - x_{cg}$ and so on, noting the convention that aircraft geometry is referred to an axis set in which 'x' points rearwards and 'z' points upwards, whereas aircraft motion is referred to an axis set in which 'x' points forwards and 'z' points downwards. The inverse transformation is

$$\text{mrc->cg} = -(\text{cg->mrc}) \tag{4.79}$$

which simply means that the relevant displacement vector has been reversed.

4.9.5 Rigid Body Motion

The dynamic motion of rigid bodies is governed by Newton's Second Law and can be expressed using vector algebra. Particular attention is required when equations of motion are formulated for a rotating axis system. The time derivative of any vector z is given by

$$\dot{\mathbf{z}} + (\omega \times \mathbf{z}) \tag{4.80}$$

where the first term defines the rate of change observed within the rotating frame and the second defines the relative motion as frame rotates with respect to another. Aircraft motion must be described in this way because the aircraft body axes are free to rotate.

The velocity of a fixed point inside the aircraft, defined by a position r relative to the CG, is given by

$$\mathbf{v}_i = \mathbf{v} + (\omega \times \mathbf{r}_i) \tag{4.81}$$

where v gives the overall linear velocity of the aircraft measured at the CG. In a similar way, acceleration is calculated in the form:

$$\begin{aligned} \mathbf{a}_i &= \dot{\mathbf{v}} + (\omega \times \mathbf{v}_i) \\ &= \dot{\mathbf{v}} + (\omega \times \mathbf{v}) + (\dot{\omega} \times \mathbf{v}) - w^2 \mathbf{r}_i + (\omega \bullet \mathbf{r}_i)\omega \end{aligned} \tag{4.82}$$

The equations of motion for the CG can be developed readily, as follows:

$$\begin{aligned} \mathbf{F} &= \sum_i \partial m_i \mathbf{a}_i \\ &= \sum \partial m_i (\mathbf{v} + \omega \times \mathbf{v}_i) \end{aligned} \tag{4.83}$$

$$\begin{aligned} \mathbf{M} &= \sum_i \partial m_i \mathbf{r}_i \times \mathbf{a}_i \\ &= \sum_i \partial m_i (r^2 \dot{\omega} - (\mathbf{r}_i \bullet \dot{\omega})\mathbf{r}_i - (\omega \bullet \mathbf{r}_i)\mathbf{r}_i \times \omega) \end{aligned} \tag{4.84}$$

assuming a mass distribution ∂m_i located at points r_i relative to the CG. The intermediate steps in the derivation are not shown but it is noted that $\Sigma \partial m_i r_i = 0$ by definition, *i.e.* the mass distribution is equivalent to a point mass located at the CG.

The representation of body inertia can be deduced by considering pure rotation under the action of a moment **M** as follows:

$$\begin{aligned}\mathbf{M} &= \sum_i \partial m_i \mathbf{r}_i \times (\dot{\omega} \times \mathbf{r}_i) \\ &= \sum_i \partial m_i (r^2 \dot{\omega} - (\mathbf{r}_i \bullet \dot{\omega})\mathbf{r}_i) \\ &= \sum_i \partial m_i (r^2 \mathbf{1} - \mathbf{r}_i \mathbf{r}_i) \bullet \dot{\omega} \\ &= \mathbf{I} \bullet \dot{\omega}\end{aligned} \tag{4.85}$$

where $\mathbf{r}_i = x_i\,\mathbf{i} + y_i\,\mathbf{j} + z_i\,\mathbf{k}$ and $\mathbf{1} = \mathbf{ii} + \mathbf{jj} + \mathbf{kk}$ (with base vectors i, j, k for the aircraft body). The quantity **I** is the inertia tensor and **1** is the unit tensor. These are geometrical objects but can only be evaluated in the context of a vector dot product, *e.g.* $\mathbf{1} \bullet \mathbf{j} = \mathbf{j}$. By definition, the inertia tensor is expanded in the form

$$\mathbf{I} = I_{xx}\mathbf{ii} + I_{xy}\mathbf{ij} + I_{xz}\mathbf{ik} + I_{yx}\mathbf{ji} + I_{yy}\mathbf{jj} + I_{yz}\mathbf{jk} + I_{zx}\mathbf{ki} + I_{zy}\mathbf{kj} + I_{zz}\mathbf{kk} \tag{4.86}$$

It should be recognised that individual components of the inertia tensor can be isolated by applying two dot products, *e.g.* $I_{zx} = \mathbf{k} \bullet \mathbf{I} \bullet \mathbf{i}$, and that symmetry exists of the form $I_{xy} = I_{yx}$. This gives rise to a matrix

$$\mathbf{I} = \begin{pmatrix} I_{xx} & I_{xy} & I_{zx} \\ I_{xy} & I_{yy} & I_{yz} \\ I_{zx} & I_{yz} & I_{zz} \end{pmatrix} \tag{4.87}$$

although it is stressed that this is not the same type of object as a tensor. The matrix components are expressed relative to a particular axis system; rotate the axes and the matrix will change but the tensor will not.

Working in tensor notation, the moments of inertia (I_{xx},I_{yy},I_{zz}) and the products of inertia (I_{xy},I_{yz},I_{zx}) can be deduced explicitly:

$$\begin{aligned}\mathbf{I} &= \sum_i \partial m_i (r^2 \mathbf{1} - \mathbf{r}_i \mathbf{r}_i) \\ &= \sum_i \partial m_i (y_i^2 + z_i^2)\mathbf{ii} + \sum_i \partial m_i x_i\, y_i\, \mathbf{ij} + \ldots.\end{aligned} \tag{4.88}$$

and so on. Working in matrix notation, the following equivalence can be established:

$$\mathbf{I}\dot{\omega} = \sum_i \partial m_i (r^2 \dot{\omega} - (\mathbf{r}_i \bullet \dot{\omega})\mathbf{r}_i) \tag{4.89}$$

which can be used to simplify the equations of motion.

Collating all of this information, the equation of rigid-body motion can be written as

$$\begin{aligned}m\dot{\mathbf{v}} &= \mathbf{F} - m\omega \times \mathbf{v} \\ \mathbf{I}\dot{\omega} &= \mathbf{M} + \sum_i \partial m_i (\omega \bullet \mathbf{r}_i)\mathbf{r}_i \times \omega\end{aligned} \tag{4.90}$$

where the total mass is the summation of mass increments, $m = \Sigma \partial m_i$.

4.9.6 Equivalent Forces and Moments

Rigid body motion is expressed by velocities and angular rates defined at the CG. However, in general, the applied forces and moments which cause that motion will <u>not</u> be generated at the CG. This necessitates a coordinate transformation consistent with a position vector $\mathbf{r}_i$ relative to the CG, such that

$$\mathbf{v}_i = \mathbf{v} + \omega \times \mathbf{r}_i \quad , \qquad \omega_i = \omega \tag{4.91}$$

These relationships can be expressed in matrix form, as follows:

$$\begin{pmatrix} \mathbf{v}_i \\ \omega_i \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \Omega_i \\ 0 & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{v} \\ \omega \end{pmatrix} \tag{4.92}$$

where I is a 3x3 identity matrix and the submatrix Ω_i acts as an operator $\Omega_i[] = []\times r_i$. For illustration, it is convenient to reduce the transformation to a generic form $x = Tz$, where x and z each represent vectors containing linear and rotational positions. Kinetic energy is expressed in the form:

$$E = \frac{1}{2}\ddot{\mathbf{x}}^t M\ddot{\mathbf{x}} = \frac{1}{2}\ddot{\mathbf{z}}^t T^t M T\ddot{\mathbf{z}} \tag{4.93}$$

where M is the generalised mass matrix, incorporating the inertia matrix associated with the rotational degrees of freedom and a diagonal matrix (with the body mass M in each element) associated with the linear degrees of freedom. Introducing generalised force vectors X and Z, consistent with the position vectors x and z, the kinetic energy can be re-expressed in the form

$$E = \frac{1}{2}\ddot{\mathbf{x}}^t \mathbf{X} = \frac{1}{2}\ddot{\mathbf{z}}^t T^t \mathbf{X} = \frac{1}{2}\ddot{\mathbf{z}}^t \mathbf{Z} \tag{4.94}$$

which shows that the appropriate force transformation is $Z = T^tX$. This relies on the fact that kinetic energy must remain constant when coordinate transformations are applied, *i.e.* energy cannot be gained or lost merely because the mathematical definition changes.

Reverting back to the explicit formulation which is required in this case, the transformation of forces and moments is given by:

$$\begin{pmatrix}\mathbf{F}\\ \mathbf{M}\end{pmatrix} = \begin{pmatrix} I & 0\\ \Omega_i^t & I\end{pmatrix}\begin{pmatrix}\mathbf{F}_i\\ \mathbf{M}_i\end{pmatrix} \tag{4.95}$$

where I is a 3x3 identity matrix and the transpose of submatrix Ω_i acts as an operator $\Omega_i^t[] = -[]\times r_i$.

4.9.7 Equations of Motion

The translational motion of the centre of gravity of an inertial body is modelled in three degrees of freedom by simply replicating the use of inertia components (all with the same value of mass). The effect of applied forces is given by Newton's Second Law:

$$\frac{d}{dt}(m.u) = X \qquad \frac{d}{dt}(m.v) = Y \qquad \frac{d}{dt}(m.w) = Z \tag{4.96}$$

The change in attitude of an inertial body is modelled as a set of rotations, each measured about the centre of gravity and a principal axis of inertia (PAI). The effect of applied moments is given by Euler's equations which, for constant inertia, have the form:

$$\begin{aligned} I_x\frac{d}{dt}p &= L + (I_y - I_z)qr\\ I_y\frac{d}{dt}q &= M + (I_z - I_x)rp\\ I_z\frac{d}{dt}r &= N + (I_x - I_y)pq\end{aligned} \tag{4.97}$$

where (L,M,N) are the applied moments and (p,q,r) are the angular velocities. In a bond graph, the states associated with these inertia components are angular momenta, not angular velocities. Thus, recasting the equations in appropriate terms, it is seen that:

$$\frac{d}{dt}\mu_x = L + \frac{I_y - I_z}{I_y I_z}\mu_y\mu_z \tag{4.98}$$

and so on, where (μ_x,μ_y,μ_z) are the momentum states corresponding to (p,q,r), *e.g.* $p = \mu_x/I_x$.

The bond graph model of Euler's equations can be arranged in a ring pattern [Karnopp 1969]. The main features are the gyrators which provide cross-coupling between axes of rotation; the gearing ratio of each gyrator is the angular momentum (*i.e.* the state) associated with the inertia diametrically opposite from it.

4.9.8 Position

The rate of change in position is determined identically by the velocity components in earth axes:

$$\frac{d}{dt}\begin{pmatrix} x_E \\ y_E \\ z_E \end{pmatrix} = \begin{pmatrix} u_E \\ v_E \\ w_E \end{pmatrix} \tag{4.99}$$

4.9.9 Orientation

The rate of change in orientation is evaluated in a much less straightforward manner, as follows:

$$\frac{d}{dt}\begin{pmatrix} \chi \\ \theta \\ \varphi \end{pmatrix} = \begin{pmatrix} q.\sin\varphi.\sec\theta + r.\cos\varphi.\sec\theta \\ q.\cos\varphi - r.\sin\varphi \\ p + q.\sin\varphi.\tan\theta + r.\cos\varphi.\tan\theta \end{pmatrix} \tag{4.100}$$

When $|\theta| = \pi/2$, the rate of change in both χ and φ is indeterminate. Numerical problems can also arise simulating very high angular rates, *i.e.* an aircraft performing rapid manouevres.

A well-known and widely-used alternative representation of attitude is given by a set of *quaternions* (or Cayley-Klein parameters). These are defined by the following expressions:

$$\begin{aligned} e_0 &= \cos\tfrac{\chi}{2}\cos\tfrac{\theta}{2}\cos\tfrac{\varphi}{2} + \sin\tfrac{\chi}{2}\sin\tfrac{\theta}{2}\sin\tfrac{\varphi}{2} \\ e_1 &= \cos\tfrac{\chi}{2}\cos\tfrac{\theta}{2}\sin\tfrac{\varphi}{2} - \sin\tfrac{\chi}{2}\sin\tfrac{\theta}{2}\cos\tfrac{\varphi}{2} \\ e_2 &= \cos\tfrac{\chi}{2}\sin\tfrac{\theta}{2}\cos\tfrac{\varphi}{2} + \sin\tfrac{\chi}{2}\cos\tfrac{\theta}{2}\sin\tfrac{\varphi}{2} \\ e_3 &= \cos\tfrac{\chi}{2}\sin\tfrac{\theta}{2}\sin\tfrac{\varphi}{2} - \sin\tfrac{\chi}{2}\cos\tfrac{\theta}{2}\cos\tfrac{\varphi}{2} \end{aligned} \tag{4.101}$$

where the following algebraic constraint applies:

$$e_0^2 + e_1^2 + e_2^2 + e_3^2 = 1 \tag{4.102}$$

The derivation is achieved by reconstructing the transformation from earth axes to body axes as a single rotation about a line in space. The details are rather tedious but if the line makes direction angles α, β and γ with the earth axes and the rotation is Δ then it can be shown that

$$e_0 = \cos\frac{\Delta}{2}\ , \quad e_1 = \cos\alpha.\sin\frac{\Delta}{2}\ , \quad e_2 = \cos\beta.\sin\frac{\Delta}{2}\ , \quad e_3 = \cos\gamma.\sin\frac{\Delta}{2} \tag{4.103}$$

Using this formulation, the rate of change in orientation is now evaluated as follows:

$$\frac{d}{dt}\begin{pmatrix} e_0 \\ e_1 \\ e_2 \\ e_3 \end{pmatrix} = \begin{pmatrix} 0 & -p & -q & -r \\ p & 0 & r & -q \\ q & -r & 0 & p \\ r & q & -p & 0 \end{pmatrix}\begin{pmatrix} e_0 \\ e_1 \\ e_2 \\ e_3 \end{pmatrix} \tag{4.104}$$

This overcomes the singularity when $|\theta| = \pi/2$, and the numerical issues of simulating rapid manouevres. However, it does introduce a numerical problem of ensuring that (4.99) is satisfied.

Quaternions can be translated into a set of direction cosines which define the instantaneous transformation from earth axes to body axes. These populate a transformation matrix as follows:

$$\begin{pmatrix} l_1 & l_2 & l_3 \\ m_1 & m_2 & m_3 \\ n_1 & n_2 & n_3 \end{pmatrix} = \begin{pmatrix} e_0^2 + e_1^2 - e_2^2 - e_3^2 & 2(e_1e_2 + e_0e_3) & 2(e_1e_3 - e_0e_2) \\ 2(e_1e_2 - e_0e_3) & e_0^2 - e_1^2 + e_2^2 - e_3^2 & 2(e_2e_3 + e_0e_1) \\ 2(e_1e_3 + e_0e_2) & 2(e_2e_3 - e_0e_1) & e_0^2 - e_1^2 - e_2^2 + e_3^2 \end{pmatrix} \tag{4.105}$$

and, using the specification of the earth2body transformation [*cf.* section 4.9.3] in conjunction with the elementary rotations defined in (4.74), the following expressions can be derived for the Euler angles:

$$\theta = \sin^{-1}(-l_3) \quad , \quad \chi = \cos^{-1}(l_1/\cos\theta).\,\mathrm{sign}[l_2] \quad , \quad \varphi = \cos^{-1}(n_3/\cos\theta).\,\mathrm{sign}[m_3] \tag{4.106}$$

4.9.10 Velocity

For aircraft applications, there are numerous definitions of velocity which are required for various purposes. These fall into categories of inertial velocity, airspeed and incidence and, finally, air data. The last of these is related to atmospheric properties and the in-flight measurement of pressure and temperature and is covered thoroughly in British Standard 2G199 [BSI 1984].

4.9.10.1 Inertial Velocity

The aircraft trajectory through space is described by velocity parameters, flight path heading and climb/dive angle, according to the following relationships:

$$\begin{aligned} V_E &= \sqrt{u_E^2 + v_E^2 + w_E^2} \\ GS &= \sqrt{u_E^2 + v_E^2} \\ VS &= -w_E \\ \gamma &= \tan^{-1}(VS/GS) \\ \chi &= \tan^{-1}(v_E/u_E) \end{aligned} \tag{4.107}$$

4.9.10.2 Airspeed and Incidence

Aircraft velocity through the airmass is derived from a combination of inertial velocity and wind disturbance:

$$\begin{pmatrix} \Delta u \\ \Delta v \\ \Delta w \end{pmatrix} = \begin{pmatrix} l_1 & l_2 & l_3 \\ m_1 & m_2 & m_3 \\ n_1 & n_2 & n_3 \end{pmatrix} \begin{pmatrix} u_E - U_E \\ v_E - V_E \\ w_E - W_E \end{pmatrix} \tag{4.108}$$

where (U_E,V_E,W_E) is the wind shear disturbance vector, with time-varying properties which are parameterised with respect to a time history, displacement along the flight path or position in space. Characterisations of wind shear are available in open literature [*e.g.* Woodfield & Woods 1983, Schanzer 1983]. From these velocity components, the calculation of total airspeed (V) and incidence angles (*i.e.* angle of attack α and angle of sideslip β) proceeds as follows:

$$\begin{aligned} V &= \sqrt{\Delta u^2 + \Delta v^2 + \Delta w^2} \\ \alpha &= \tan^{-1}(\Delta w/\Delta u) \\ \beta &= \sin^{-1}(\Delta v/V) \end{aligned} \tag{4.109}$$

4.9.11 Non-inertial Frames

If equations of motion are developed for a rotating axis system then acceleration is no longer fully determined by the applied force. The reference frame is non-inertial and fictitious forces are added in order to compensate for the effects of rotation. The effective forces are:

$$\begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + \begin{pmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{pmatrix} \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{4.110}$$

where (X,Y,Z) is the total force, (u,v,w) is velocity and (p,q,r) is the angular velocity. The *load factors* are equivalent to the effective forces normalised with respect to gravitational acceleration, *i.e.*

$$n_x = X'/g \quad , \quad n_y = Y'/g \quad , \quad n_z = Z'/g \tag{4.111}$$

Thus, in common aircraft parlance, a "Nine-G Turn" corresponds to a manoeuvre in which $n_z=9$. Note that, by convention, gravitational force (*i.e.* weight) is not included in the load factors although, obviously, it must be accounted in the total force acting on the aircraft.

4.9.12 Component Class Definitions [*cf.* Figure 4.6]

An important purpose in presenting a standard formulation of vehicle dynamics is to be able to highlight the extremely compact form of bond graph models. The components defined below are relatively trivial, yet they adequately represent most of the preceding subsections.

Euler

The Euler component gives a bond graph equivalent of Euler's equations in the form expressed in (4.98).

Newton

The Newton component gives a bond graph equivalent of Newton's Second Law in the form expressed in (4.96).

Transformation

The **Transformation** component encapsulates a **TF** component with a 3x3 matrix CR (*i.e.* a transformer which acts across vector bonds) for three degrees of freedom.

AT and AT2

Axis transformations follow a specification of the type described in Section 4.9.3, based on elementary rotations defined in (4.77). These are implemented in one of two ways. The AT component performs a transformation for *either* linear *or* rotational degrees of freedom; the AT2 component performs a transformation for both, in parallel, based on a matrix class called **AxisTransformation**, which conforms to (4.78).

CP

Transformations between reference points are described in Section 4.9.4. When multiplied with a vector, this matrix effectively performs a vector cross-product with $(\Delta x,\Delta y,\Delta z)$, the elements being the values contained within the matrix. For this reason, the **CP** component is merely a specialised form of **Transformation** based on a matrix class called VectorCrossProduct [*cf.* (4.92)].

LeverArm and OS

Lever arm compensation follows immediately from the **CP** component, enabling forces and velocities to be transformed between reference points. The **OS** component is a straight copy of **LeverArm**. It is intended to represent an *offset* between reference points, a term which perhaps has better intuitive value, but a good practical consideration is that its name is shorter and, thus, a bond graph which uses it can be rendered a little more tidily that one using long component names.

Motion

The **Motion** component describes the complete rigid body motion of an aircraft, combining **Euler** and **Newton** with the aerodynamic and propulsive forces. The OS components bring all applied forces together at the aircraft CG and the AT2 components ensure alignment with aircraft body axes. Separate **AT** components re-align the force/velocity information with aircraft PAIs prior to applying the equations of motion. An additional feature is a **R** component which implements the compensation required for effects of frame rotation as defined by the matrix CR in (4.110). Also, note that the effects of gravity and wind shear have been included.

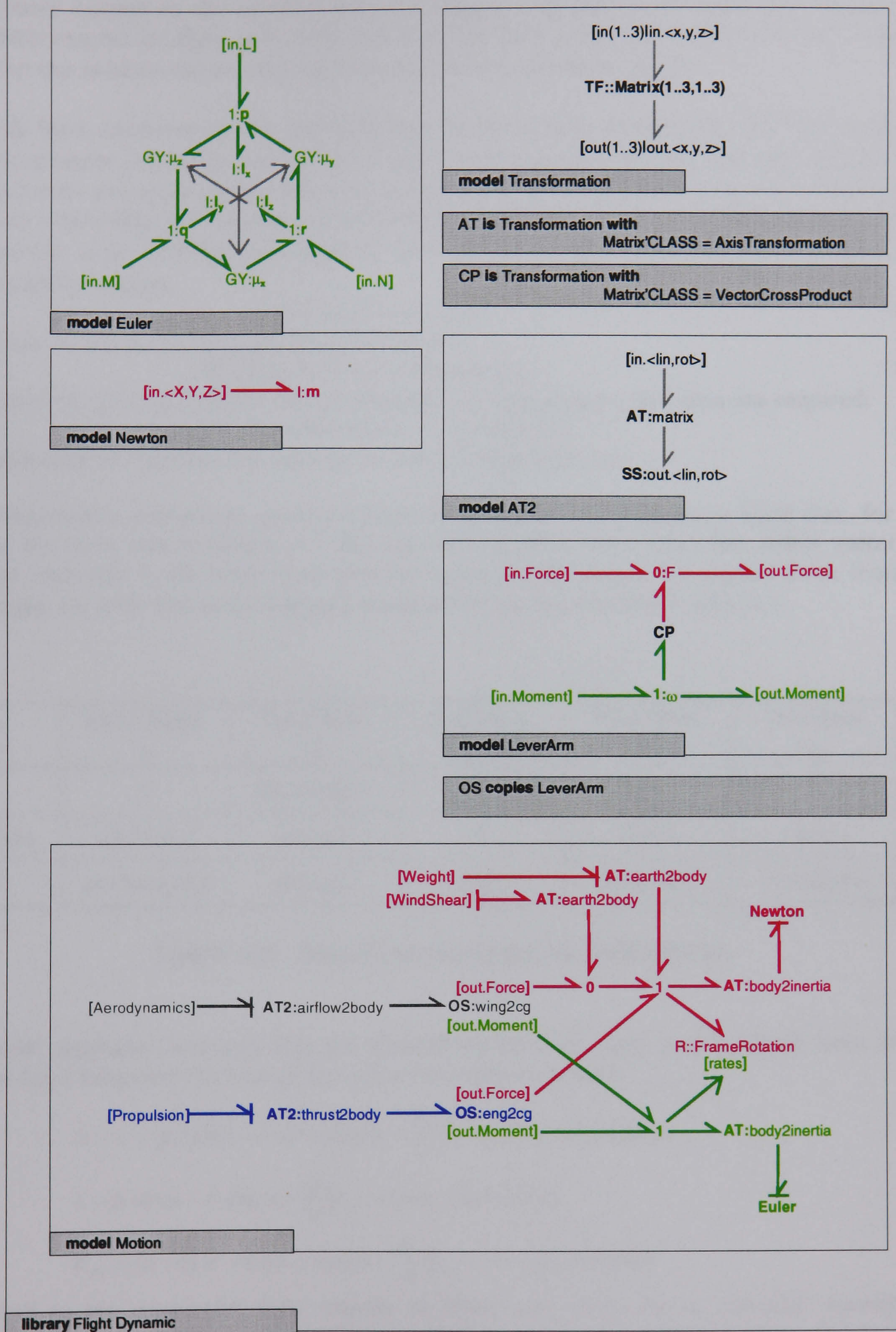


Figure 4.8 'Flight Dynamic' Library Components

4.9.13 Steady-state Conditions

It is important to be able to establish balanced flight conditions, in which forces and moments combine to create an equilibrium. To this end, it is most appropriate to work in flight path axes and to assume that the applied force normal to the velocity vector is aligned with the z-axis. Note that angular rates are expressed with respect to flight path axes and that the bank angle (μ) is measured as a right-handed rotation about the velocity vector, starting from the vertical reference plane.

The steady flight condition will be specified here by the aircraft velocity (V), climb/dive angle (γ) and the applied load factor (n_z) along the negative z-axis. In the most general case, a steady manoeuvre can be described as a turn executed about a point in space, with a turn radius R measured normal to the flight path. The turn rate is then Ω=V/R, measured normal to both the velocity and radius vectors. If the radius vector is inclined at an effective bank angle λ_0 then the bank angle associated with the applied normal force is obtained by solving:

$$n_z \sin(\lambda_0-\lambda) = \sin\lambda_0 \cos\gamma \qquad (4.112)$$

From this solution, the turn rate is calculated as follows:

$$W = [\, n_z \cos(\lambda-\varphi) - \cos\lambda \cos\gamma \,]\, g/V \qquad (4.113)$$

In order to provide correct manoeuvre coordination, the following angular rates are required:

$$p_{vv}=0, \quad q_{vv}=\Omega\cos(\lambda_0-\lambda), \quad r_{vv}=\Omega \sin(\lambda_0-\lambda) \qquad (4.114)$$

where the subscript 'vv' denotes the velocity vector, *i.e.* flight path axes.

This development is completely general and can be applied to any manouevre. Note that , for flight in a straight line, the turn rate is infinite and the turn rate is zero. Also, when the radius vector lies in a vertical plane such that λ_0=0, there is an obvious intuitive requirement the applied force should have a zero bank angle, *i.e.* λ=0. The most frequent manoeuvres are summarised in table 4.1.

Manouevre	Bank Angle λ	Turn Rate Ω	Roll Rate p_{vv}	Pitch Rate q_{vv}	Yaw Rate r_{vv}
Pull-up	0	$(n_z-\cos\gamma)g/V$	0	Ω	0
Horizontal Turn	$\cos^{-1}(1/n_z)$	$\tan\lambda.g/V$	0	$\Omega\sin\lambda$	$\Omega\cos\lambda$
Helical Turn[3]	$\cos^{-1}(\cos\gamma/n_z)$	$\tan\lambda.g/V$	$-\Omega\sin\gamma$	$\Omega\cos\gamma\sin\lambda$	$\Omega\cos\gamma\cos\varphi$

Table 4.3 Standard Manoeuvre Definitions

By a similar argument, and including the transformation from flight path to body axes, the general relationships for a sustained velocity-vector roll are written as follows:

$$\dot{\alpha} = q - (p.\cos\alpha - r.\sin\alpha)\tan\beta + \frac{g}{V}(n_z - \cos\gamma.\cos\lambda)\sec\beta \qquad (4.115)$$

$$\dot{\beta} = p.\sin\alpha - r.\cos\alpha + \frac{g}{V}(n_y + \cos\gamma.\cos\lambda)\tan\beta$$

$$p_{vv} = (p.\cos\alpha - r.\sin\alpha)\sec\beta - \frac{g}{V}(n_z - \cos\gamma.\cos\lambda)\tan\beta$$

where n_y and n_z are expressed with respect to flight path axes. For a correctly coordinated roll manoeuvre, the roll and yaw rates should be controlled so as to ensure that the rates of change of angles of attack and angle of sideslip are both zero.

[3] Turn rate is defined about a vertical axis and not an axis normal to the flight path

4.10 Conclusion: Building Blocks!

The application of bond graph modelling has been demonstrated across five of the major energy domains relevant to physical systems. This has resulted in library definitions established to a level appropriate for building conceptual models of aircraft systems and these provide the basis for the development of air vehicle system models in the following chapter. The aim is to summarise the main physical principles that are of interest in each domain (so as to declare the limits of applicability of models) and to build bond graphs of the main equipment components that are relevant to this project. On route to component modelling, much work has had to be done in order to prove the feasibility and usability of new notational features introduce in Chapter 2 and, overall, it is seen that the bond graph method offers good support for complex system models. Based on the work presented in this chapter it can be argued with confidence that the method, apart from offering a very convenient shorthand notation, simplifies the process of building mathematical models.

Chapter 5

The Virtual Aircraft

SUMMARY

The motivation for developing a unified approach to modelling based on bond graphs has been to build a so-called *virtual aircraft* and to demonstrate how it might offer insight into the behaviour of integrated aerospace systems. Adopting a common notation provides compatibility between models; adopting a functional notation (as defined in chapter 3) ensures consistency in model construction; adopting a bond graph notation gives a direct association with physical processes. The issue remaining is whether or not this really assists engineers in understanding the structure of large-scale system integration.

This chapter introduces a hypothetical but representative aircraft and defines for it a unified model, of a type which might be developed to support control system design. As mentioned earlier, a control perspective is helpful because it places a genuinely integrated functional requirement on a collection of systems, usually expressed in the concept of a *vehicle management system*. The aircraft happens to be a high-performance combat aircraft although the benefits of increasing systems integration are generic. A range of appropriate systems are included and, in order to illustrate wide-envelope behaviour, the example aircraft has been given a Short Take-Off and Vertical Landing (STOVL) capability.

5.1 Introduction

The purpose of a so-called *virtual* aircraft is to be able to integrate the design of an aircraft without having to rely on physical mock-ups and test facilities. This allows specific options to be evaluated in the context of the whole aircraft and supports a proof of concept early in the design cycle. With confidence in the correctness of design, risks are reduced and so too are the requirements for rig testing. It should be noted that some testing will always be required in order to *demonstrate* the operability of the aircraft and its systems, to *verify* that design assumptions hold in practice, to *validate* what was modelled and to *measure* significant phenomena which were not or could not be modelled, for whatever reason.

In order to stimulate productive research into aerospace systems integration, a virtual aircraft model has been developed which describes the energy transfer characteristics and functional behaviour of a typical high performance aircraft. It covers both vehicle dynamics and vehicle systems in order that the implications of design in either area can be recognised at aircraft level, especially in the context of integrated control. A major objective is to counter the preponderance of flight control problems in learned publications and to break new ground in defining benchmark control problems for overall vehicle management. The new model provides a framework for energy management and control law design concepts. The overall model structure is shown in Figure 5.1 and is partitioned as follows:

Motion	Six degree-of-freedom equations of motion
Airframe	Aircraft structure and aerodynamics
Aerodynamics	Aircraft loading due to airflow
Actuation	Mechanical force generators
Engine	Gas turbine powerplant
Electrics	Electrical power generation and distribution
Hydraulics	Hydraulic power generation and distribution
Cooling	Environmental control
Fuel	Fuel management

It is depicted as a heuristic bond graph, the aim being to highlight the structure of dynamic interaction between the components. As a bond graph, it indicates the paths along which energy is passed: as a heuristic model, it does not show the detail of how that is actually achieved and it does not necessarily show all paths. In order to place this model in a number of standard contexts, various subsystem interactions have been highlighted. Usually, these are discussed separately as specialised disciplines in their own right. Here the aim to show the widespread interaction between different subsystems which must be taken into account when integrated control is a primary consideration in aircraft design.

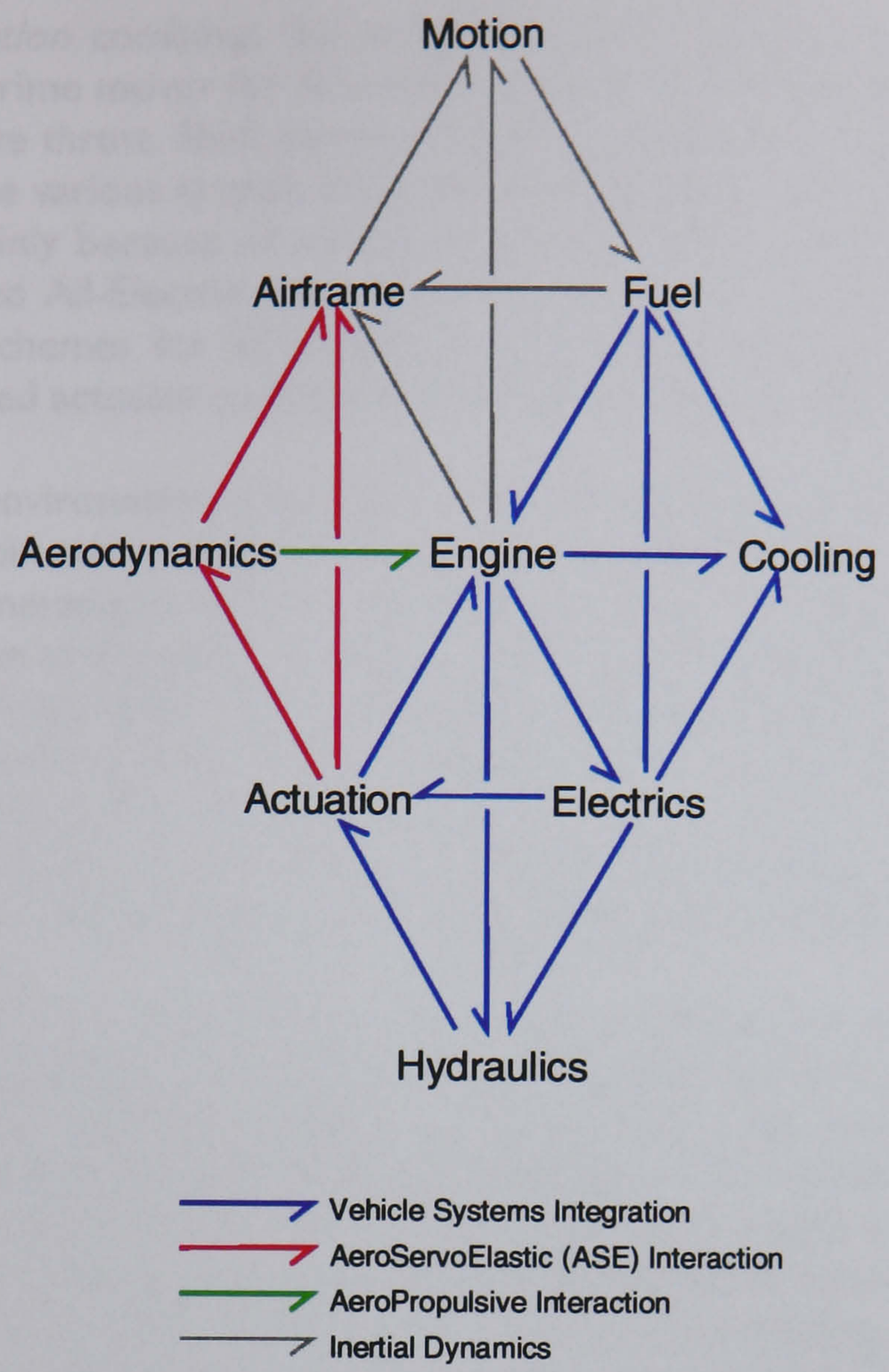


Figure 5.1 Air Vehicle Integration

Inertial Dynamics covers conventional six degree-of-freedom flight dynamics. In the model adopted here, forces and moments are accounted by separate contributions for the engine, airframe and fuel. The engine develops propulsive force through one or more nozzles which, in military powerplants, might be "vectorable" (*i.e.* rotated in order to steer the thrust vector). The airframe develops aerodynamic force as functions of external geometry and incidence (*i.e.* orientation with respect to the airflow). It reacts aerodynamic and gravitational forces, as well as internal forces transmitted through the engine and actuator mounting points, all of which produce second order effects of structural distortion and consequent small changes in the aerodynamics. The airframe contains fuel and the dynamics of fuel movement change the centre of gravity (CG) of the aircraft, thereby changing the moment of applied forces. The calculation of aircraft flight dynamics is then merely an application of Newton's Second Law.

Aeropropulsive Interaction covers a number of phenomena associated with airframe-engine integration which produce changes to aerodynamic performance as a direct result of thrust generation. This takes into account the effects of engine installation. Most important in conventional up-and-away flight is intake aerodynamics and the various types of drag force which result, such as momentum drag, spillage drag and inverse cowl suction. In powered-lift, there are suckdown and fountain effects as discussed later. In transitional phases (and occasionally elsewhere), the use of vectored thrust has a direct impact on flowfields around the inboard wing and wing-body carry-over structure which affects the flight dynamics.

Aeroservoelasticity is a highly specialised discipline dealing with interactions between aerodynamics, structural dynamics and actuation. It combines aeroelasticity, which considers the energy transfer between airflow and structure, and structural coupling, which considers how actuators dissipate power into airframe flexible modes as a consequence of control surface movement. Relevant phenomena include the reduction in control effectiveness (known as *Reversal*), a range of unsteady aerodynamic effects (such as *Buffet* and *Buzz*) and a range of structural instabilities (collectively known as *Flutter*) caused by interaction with airflow.

Vehicle Systems Integration combines the areas of engine operation, control actuation and aircraft utilities. The engine is the prime mover for all aircraft systems, delivering shaft power and airbleed offtake capacity as well as propulsive thrust. Shaft power is used to generate electrical and hydraulic power, both of which are used to actuate various aircraft equipment. Note that primary actuation (for flight control) is powered by hydraulics, mainly because of a high power-to-volume ratio, although there has been long-standing interest in so-called All-Electric and More-Electric Aircraft. These involve all shades of opinion from electrical actuation schemes for all aircraft requirements through to electro-hydrostatic devices (formally known as integrated actuator packages) which use local electrically-powered hydraulic circuits.

Control of the aircraft environment (especially thermal management) is required for aircrew, avionics, general equipment, fuel, cabin and a range of miscellaneous purposes (such as rain dispersal, canopy seal, demist/antimist, oxygen generation). This is provided by means of engine airbleed at sufficiently high pressure to service all zones of the aircraft which require venting and at sufficiently low temperature to avoid wasteful heating/cooling cycles. Essentially the airbleed is split into two paths, one which is expanded to become cold and the other which remains hot; the two paths are mixed in order to control temperature. Many elaborate schemes exist which vary the basic principle in order to provide independent control over many variables and to improve the efficiency of heat transfer by means of closed heat cycles and liquid loops (certainly including fuel) for intermediate cooling.

Fuel management represents a major design issue in high-performance aircraft. Fuel is essential as an energy source but the disadvantage is that, because aircraft require quite a lot of it under normal operation, it implies a weight penalty and it has to be maintained in the correct locations within the airframe in order for the aircraft CG to be within specified limits. For larger aircraft, the wing root bending moment (combining distributions of structural weight, fuel weight and aerodynamic loading) also presents non-trivial problems. In many respects, a reasonably accurate fuel model is a key component of a virtual aircraft; it has a profound effect on flight dynamics, structural loading, thermal management (especially on hot days), electrical power consumption (by fuel pumps) and hydraulics (because fuel is usually the primary heat sink).

5.2 Air Vehicle Description

The baseline air vehicle configuration is shown in Figure 5.2. It is a Short Take-Off and Vertical Landing (STOVL) aircraft, with conventional geometry, similar in concept to the Harrier family. The engine is mounted in the centre fuselage in order to provide a steerable thrust vector from a compact unit without flow switching or remote lift devices. This enables the aircraft to perform transitions between powered-lift and conventional flight *via* a redistribution of control power, shown schematically in Figure 5.3.

The engine is an Augmented Vectored Thrust (AVT) gas turbine. It is a straight-flow, twin-spool turbofan with high thrust-to-weight ratio for powered lift and large bleed offtake capacity for attitude control in hover. Mechanically independent Low Pressure (LP) and High Pressure (HP) compressor systems are co-axial and contra-rotate in order to minimise gyroscopic coupling. There are two pairs of convergent nozzles, mounted on rotating collars, positioned symmetrically with respect to the engine centre line. Bypass air from the LP compressor is ducted through a plenum chamber to the front nozzles, which incorporate variable area actuation and a reheat system. The engine core acts as a turbojet, with flow being discharged through the two rear nozzles, which have fixed area and operate 'dry'.

Flight dynamics are conventional except for the interference effects of jet entrainment and ground proximity, which are characteristic of powered-lift in fixed-wing applications. Substantial modification to the wing aerodynamics (and the propagation downstream) result from large thrust vector angles at high thrust levels. Effectively, at low speed, the airframe has a high drag configuration. Ground effects in the hover are significant because jet efflux from the powerplant is energetic. The aircraft experiences a 'fountain' effect due to jet recirculation underneath the fuselage and a 'suckdown' effect due to wider recirculation around the aircraft. The flowfield results in hot gas reingestion (HGR) at the intake and a consequent reduction in thrust.

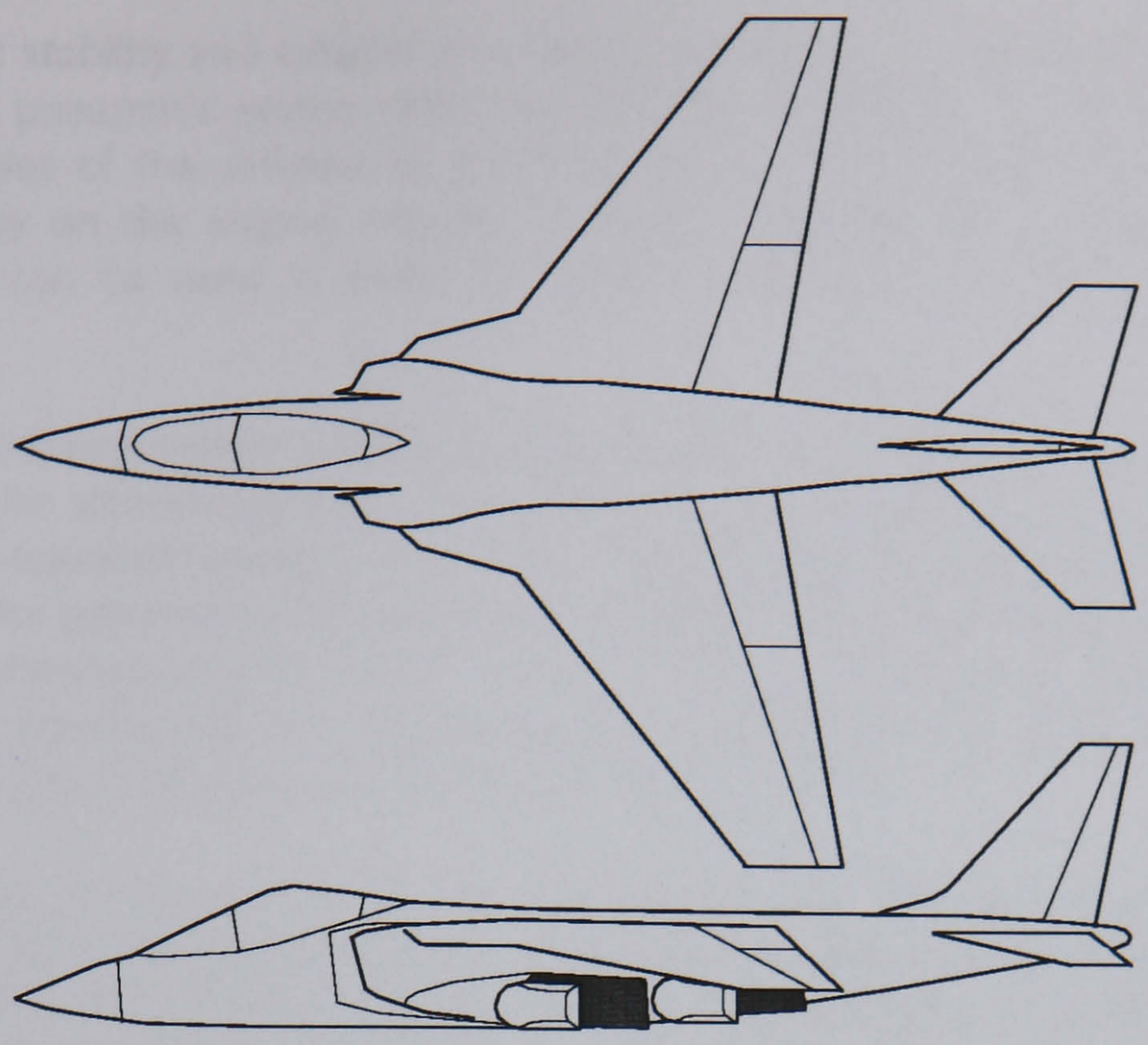
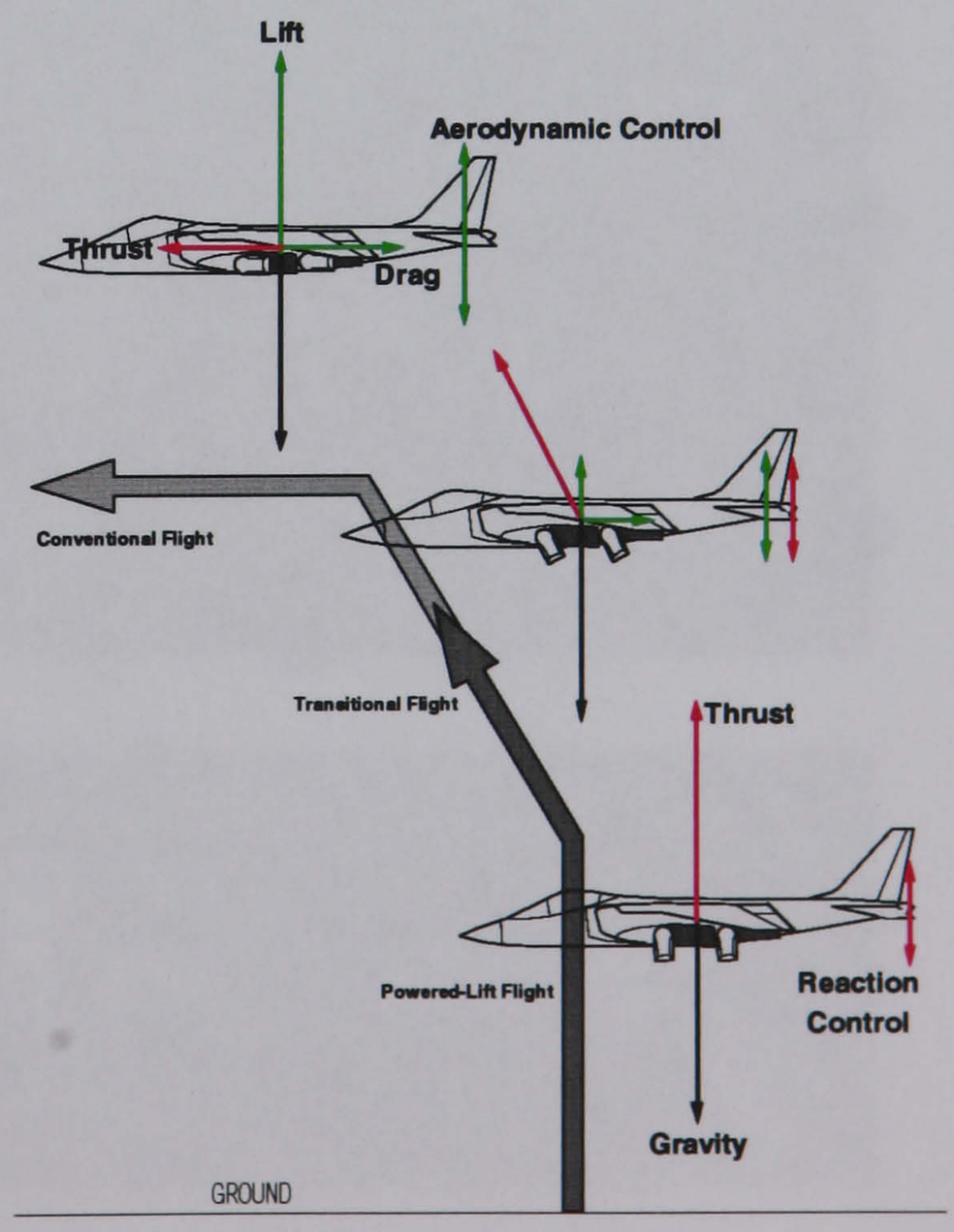


Figure 5.2 Aircraft General Arrangement

There are three types of *primary* effector, namely flying controls (flaperons, tailerons and rudder), engine nozzles and reaction control valves (or RCVs). *Secondary* effectors include the wing leading-edge devices and the air intake variable geometry. In the AVT concept, the main fuel flow and the reheat fuel flow can be modulated independently in order to vary gross thrust and thrust distribution fore and aft. In addition, the independent control of front and rear nozzles can reduce the offset between thrust line and aircraft CG. This is important during transitions because only limited aerodynamic control is available to counteract potentially large thrust-induced pitching moments.

Figure 5.3
STOVL Flight Regimes



Lack of aerodynamic stability and control at very low speeds is compensated by the Reaction Control System (RCS). This is a pneumatic system which draws high-pressure (HP) bleed air from the engine and vents it at the extremities of the airframe in order to produce a three-axis torque reaction. Because of the performance penalty on the engine, reaction control is intended for transient control. Variation in thrust centre position can be used in order to satisfy steady-state requirements, thereby minimising airbleed offtake.

The engine is the primary power system on the aircraft; its purpose is to generate thrust and to provide offtake power for aircraft utilities (*e.g.* hydraulics and electrics). In this study, the utilities will be combined into a fully integrated system incorporating design features for improved operating efficiency. For current purposes, the subsystems of interest are variable-pressure hydraulics, Ac/Dc electrical power, fuel management and environmental control. The last of these combines the functionality of a traditional Environmental Control System (ECS), a flight-operable Auxiliary Power Unit (APU) and a secondary electrical generator (in fact, an integrated starter/generator or ISG).

The configurations of the hypothetical thermofluid systems for this aircraft are shown in **Figure 5.4**. The engine in this case has a standard intake and unmixed flow paths for bypass and core, giving separate thrust vectors. There are two offtakes, one for shaft power (labelled conventionally as **PTO** or Power Take-Off) and one for airbleed (labelled as **Bleed**). The enhanced ECS, under normal flight operation is driven by engine bleed air (drawn in *red*); this is cooled by expansion across a turbine (labelled as **PT** or Power Turbine) and the cool air (drawn in *blue*) is mixed with hot air via a Temperature Control Valve (TCV) in order to provide conditioned air to manage the cabin heat load. The avionics heat load is managed by very cold air, perhaps at sub-zero temperature, within a closed-loop cycle (drawn separately in *red* and *blue*) based on a compressor/turbine combination. This is coupled mechanically to the power turbine and an ISG via a common shaft. These systems will be described in more detail later.

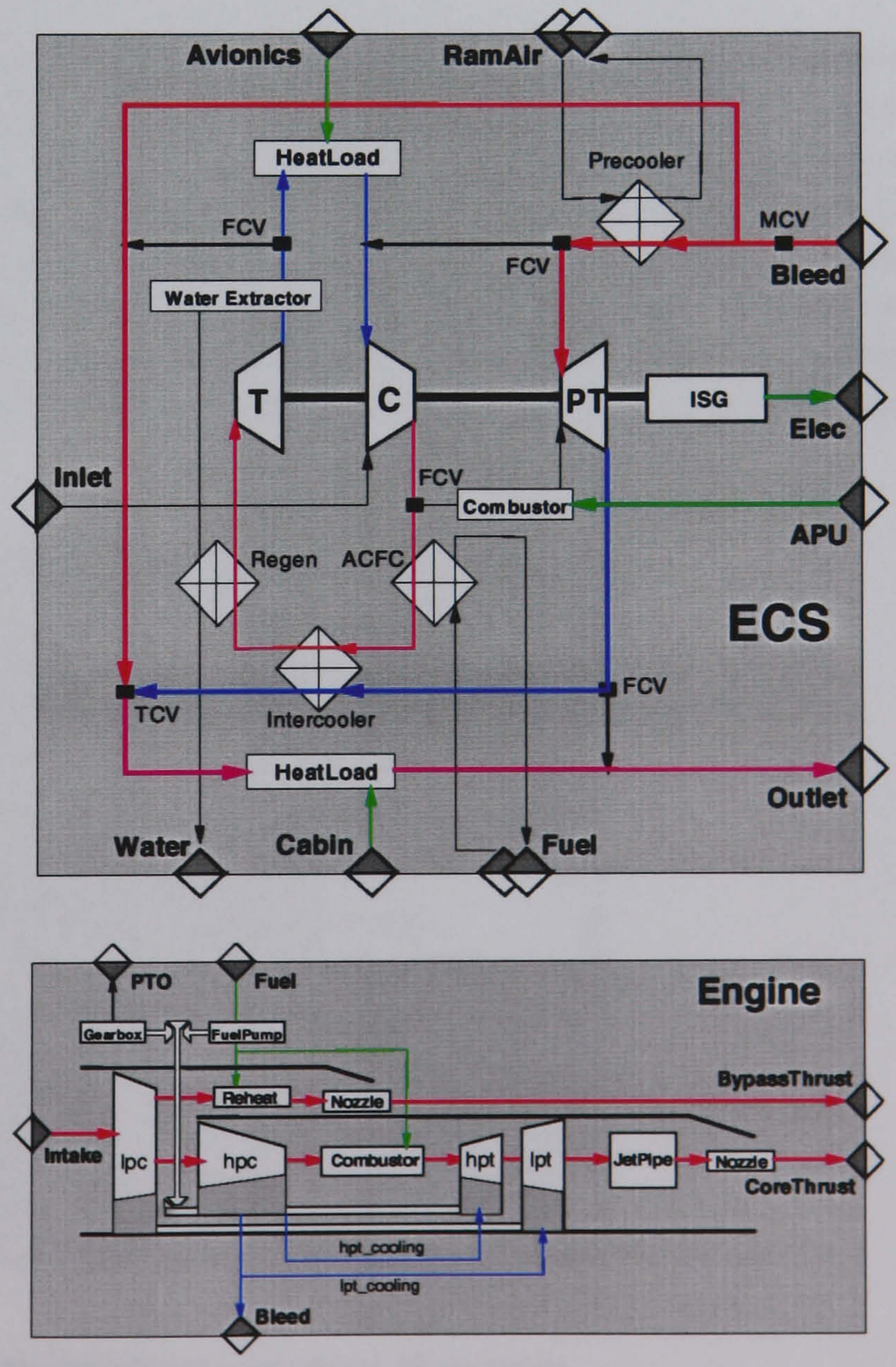


Figure 5.4
Main Thermofluid Systems

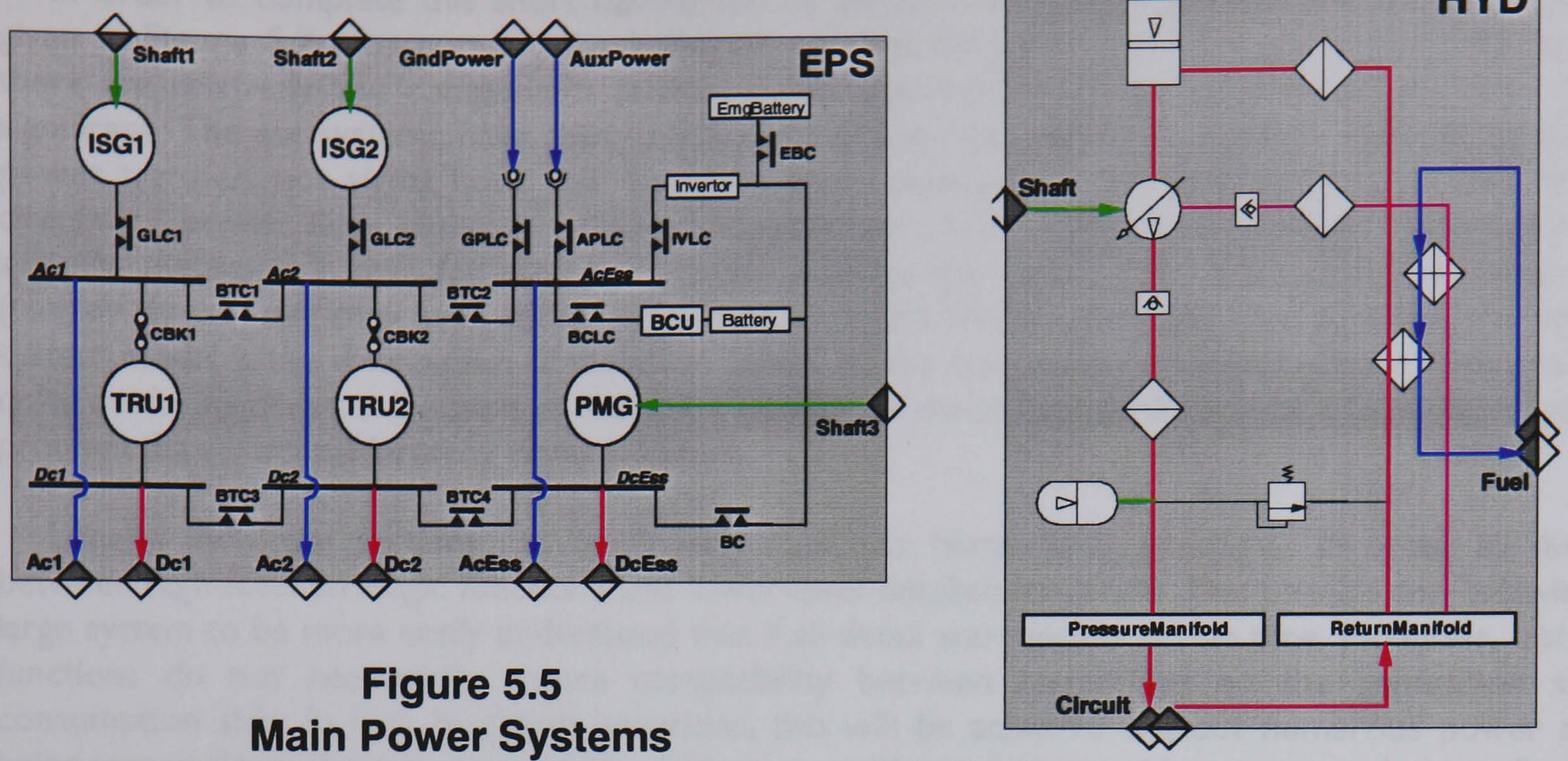


Figure 5.5
Main Power Systems

The main power generation and distribution systems are shown in **Figure 5.5**. In many respects these are very straightforward and quite ordinary. The Electrical Power System (EPS) has two lanes of AC generation and AC/DC conversion. There is also a DC generator (a Permanent Magnet Generator or PMG) and battery back-up for DC Essentials and for single-phase AC Essential via an invertor. In order to handle engine outage in-flight, there is provision for an one-shot emergency battery (possibly fueled by cordite). The hydraulic system (HYD) is a standard configuration containing a pump, a reservoir, an accumulator plus various valves, filters and heat exchangers. As is commonplace in military aviation, two hydraulic systems will be incorporated in the aircraft.

The Fuel Management System (FUM) is shown in **Figure 5.6**, illustrating a number of potential design features. There are standard refuel and jettison valves, one of each sufficing to enable the system to be filled and emptied (although typically there would be other fueling and defueling points as well). Fuel transfer is possible both longitudinally and laterally in order to control the position of aircraft CG. Under normal flight operation, the feed system and transfer system would be isolated but, under failure conditions, additional valves provided alternative paths from the tanks into the collector box, which supplies the Engine/APU feed line. There are heat exchange functions for air-cooled fuel cooling (ACFC) and fuel-cooled oil cooling (FCOC), where the latter is set up in this case to act by re-circulating fuel from the feed line back into tank system.

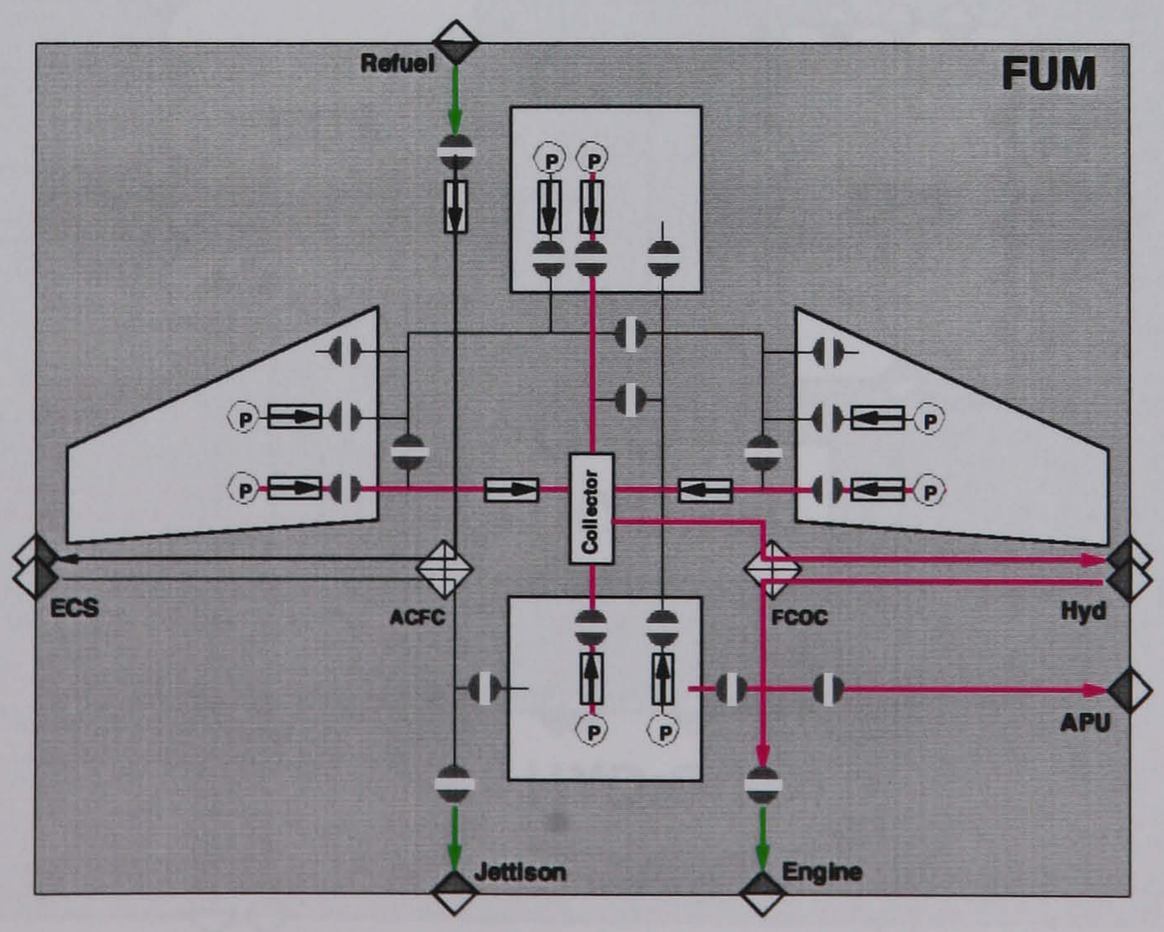


Figure 5.6 Fuel Management System

In order to complete this short description of the air vehicle, an integrated system configuration is given in **Figure 5.7** incorporating the individual systems introduced so far. It is noteworthy that, while there are relatively few components in this 'model' the number of functional interfaces (flow paths) is significant. The **six** systems have **thirty-six** interfaces associated with them. Note that the model shows power transfer, not signal flow, and therefore each interface represents an interface. Also, there are circulating power flow paths for thermal management and for hydraulic power supply/return lines; actually, the same is true for electrical power although the return path is through a common ground (namely the aircraft structure) and is not shown explicitly. What is definitely not shown in the integrated system model is the distribution of electrical power to the numerous consumers. Apart from producing a quite confusing diagram (because of the sheer number of connections), there would be a major interfacing problem if a rigorous hierarchy were followed.

Locally, individual systems can be decomposed into hierarchical structures in order to distinguish between high-level strategic functions and lower-level detailed functions. This enables the behaviour of a large system to be more easily understood that if all detail were visible at one time. However, distribution functions do not necessarily ensure compatibility between hierarchies on the generation side and consumption side. In fact, in almost situations, this will be achieved without numerous power conduits being passed down through many levels of hierarchy without doing anything on route. In aircraft electrical distribution, this might involve hundreds of connections which, while not redundant, could certainly be hidden from view. A relevant example here is the electrical supply to fuel pumps; power provision is referenced in the top-level model but power consumption is two-level down at least (encapsulated within components of the Fuel Management System model, shown in **Figure 5.6**). This will be discussed further.

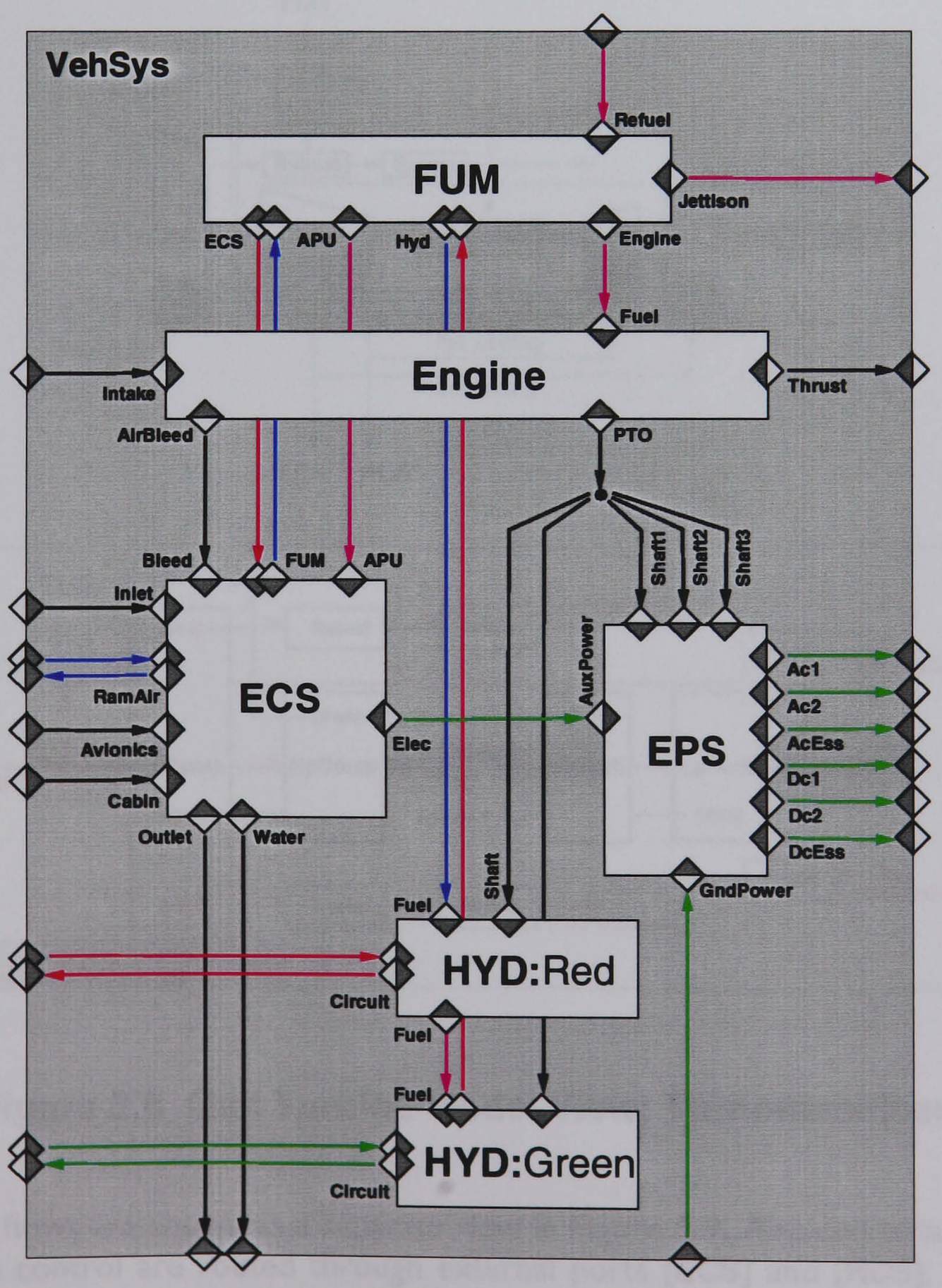


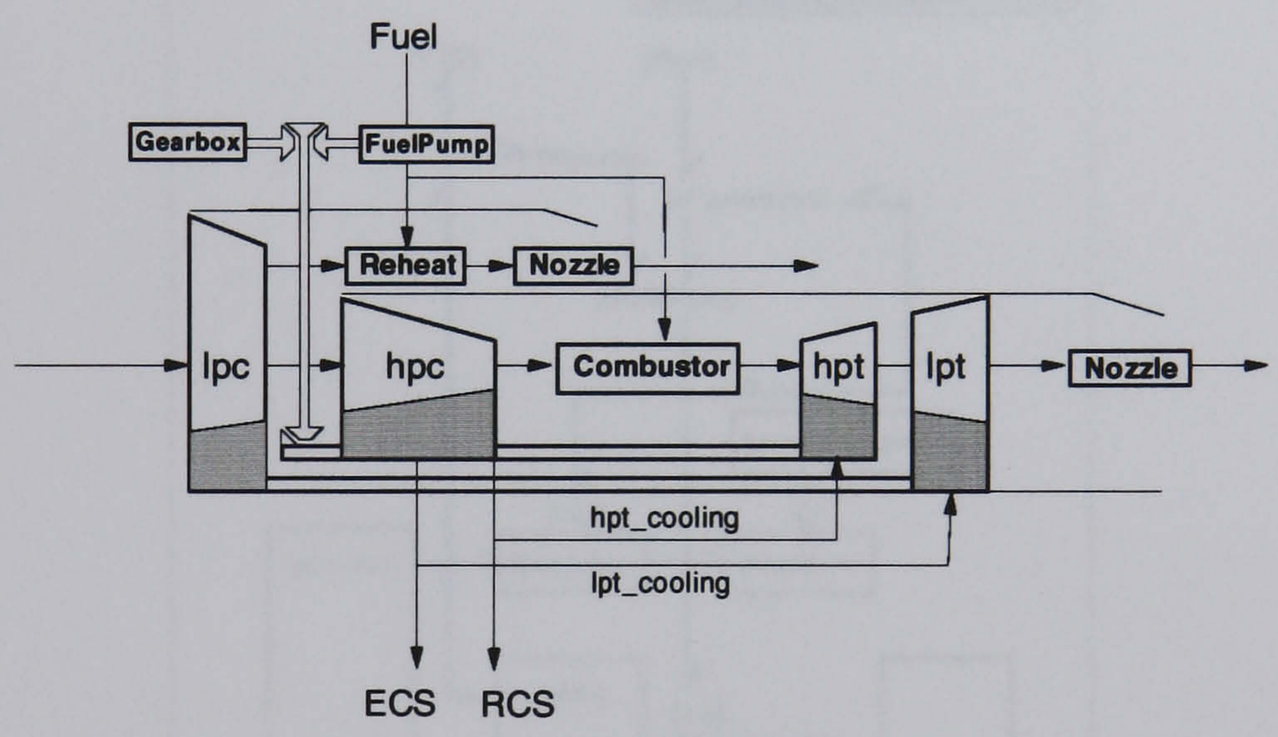
Figure 5.7 Integrated Vehicle System

5.3 Propulsion

5.3.1 Engine

The basic engine model is given in **Figure 5.8**, showing a standard schematic for a twin-spool configuration and its bond graph equivalent. The LP compressor (**LpComp**) is modelled as a split compressor, with separate exit paths for core flow and bypass flow and is driven by the LP turbine (**LpTurb**). The HP compressor (**HpComp**) is supercharged by the LP compressor and is driven by the HP turbine (**HpTurb**). The compressors and turbines are idealised as 'actuator discs' such that flow conditions change across them but there is no attempt to model the internal physics or dynamics of that change. Mechanical connection of a compressor and turbine is via a **Spool** component, which models the turbomachine inertia, bearing friction and gyroscopic coupling.

Both the core and bypass flows are split into two paths, designated as **Port** and **Stbd**, and the resulting four gas streams exit through separate **Nozzle** components. Combustion takes place in the main **Combustor** component and the Port/Stbd bypass **Reheat** components. Utilities are powered by a combination of shaft/airbleed offtake. A power take-off (PTO) shaft is driven from the HP spool to an accessory gearbox; drive pads are provided for the engine fuel pump, electrical generators, hydraulic pumps and so on. For convenience, this engine model incorporates an auxiliary flow path through **SS:Aux**, which provides an optional source of ram air (obviating the need for a separate intake) for other systems if required and an optional means of ejecting air from those systems.



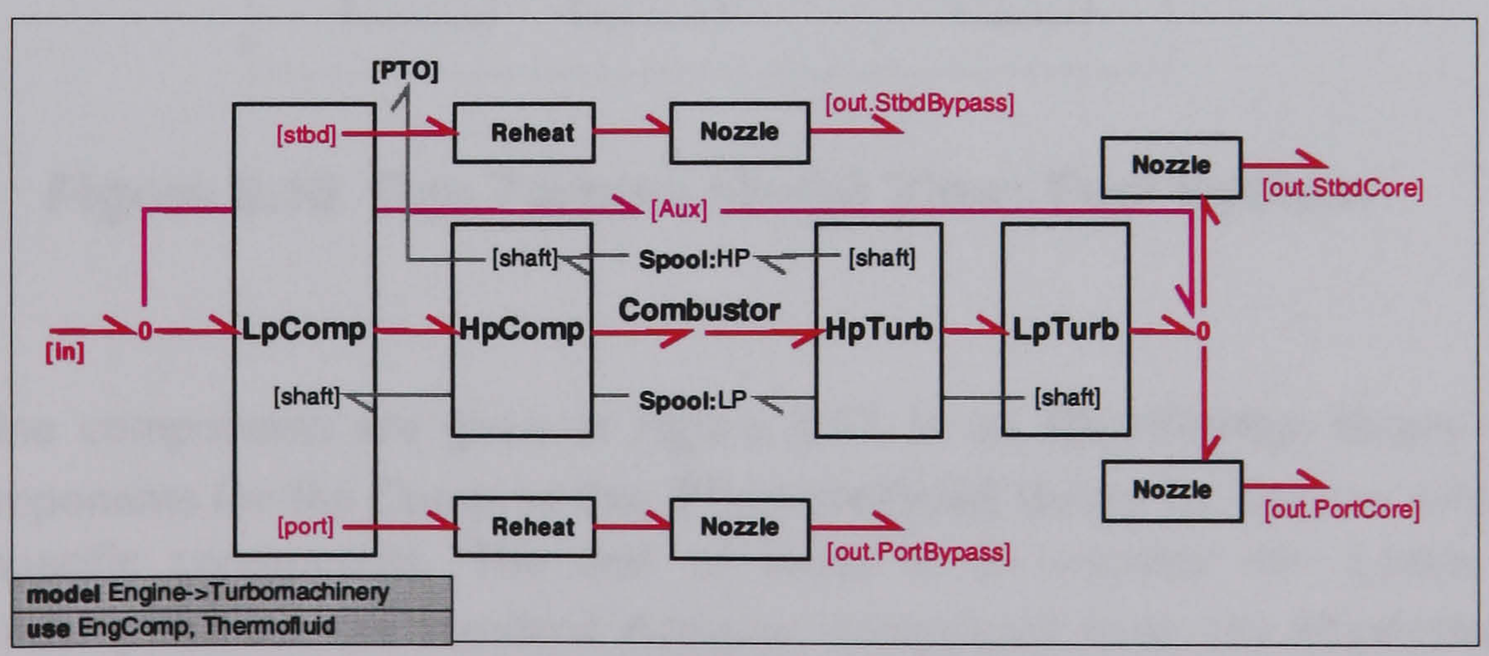


Figure 5.8 Gas Turbine Model View: Turbomachinery

The engine bleed flows are shown as a separate view in **Figure 5.9**. Airbleed offtake for environmental control and reaction control are routed through external ports **[ECS]** and **[RCS]**, respectively. Internal flows for engine cooling/sealing are drawn from different stages of HP compression and ducted to the HP and LP turbines.

In this particular engine configuration, bypass nozzle area is variable while core nozzle area is fixed. In the bond graph model, area variation is achieved by associating an actuator model with each bypass nozzle; this is powered through a hydraulic port, labelled **[Hyd]**. Note that the actuators are fuel-hydraulic (or so-called 'fueldraulic') devices and are powered by high pressure fuel. This is shown in **Figure 5.10** together with other fuel system aspects of the model.

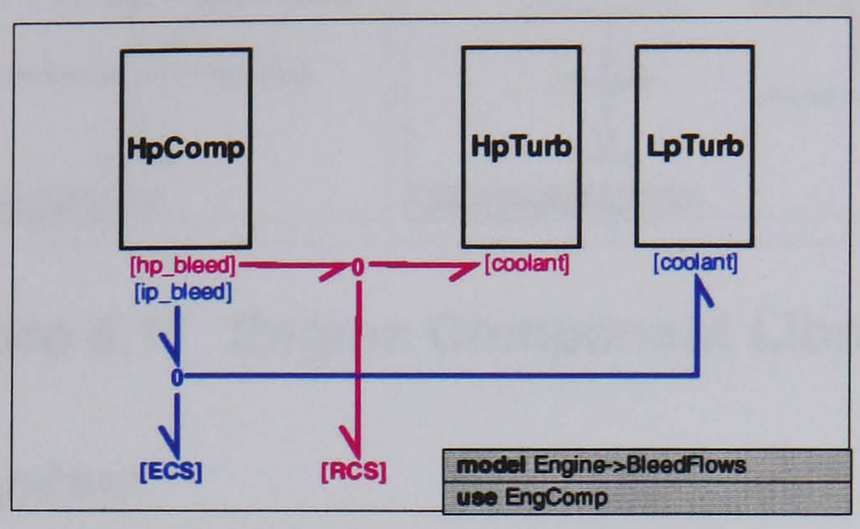


Figure 5.9 Gas Turbine Model View: Bleed Flows

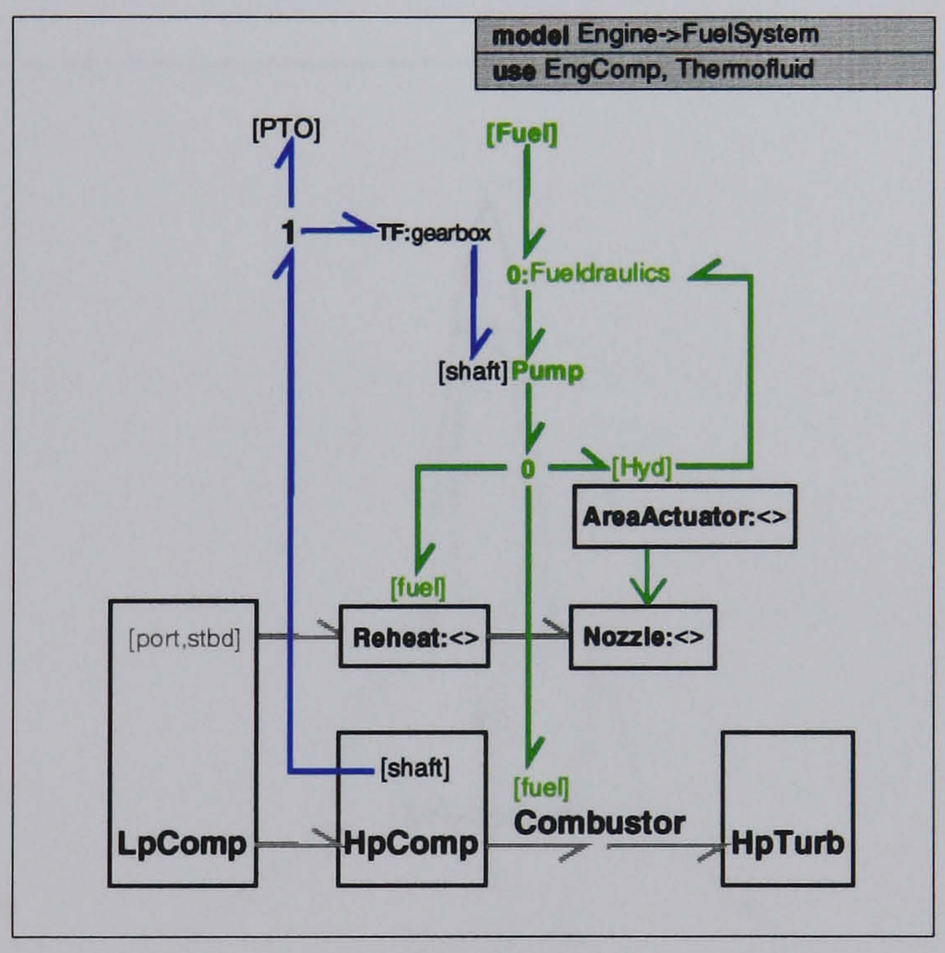


Figure 5.10 Gas Turbine Model View: Fuel System

The major engine components are given in Figure 5.11 in an **EngComp** library which renames various existing components for the Compressible **Thermofluid** library [*cf.* Section 4.4] and introduces two new engine-specific components. The first of these is an actuator for nozzle area variation (AreaActuator) which specialises the standard Actuator component from the **Hydraulic** library [*cf.* Section 4.3.5.7] (adopted for incompressible thermofluids, as discussed briefly in Section 4.5). Note that the actuator **[Drive]** port has been attached to an Actuator Drive Unit (ADU) as part of the **Control/** distribution. The second new component is a **Spool**, which represents the aggregate inertia of compressor, turbine and shaft, as well as introducing a CR for gyroscopic coupling (named **#GyroCoupling**) which, for brevity, is not defined here but would follow any standard text [*e.g.* Symon 1971, p.165].

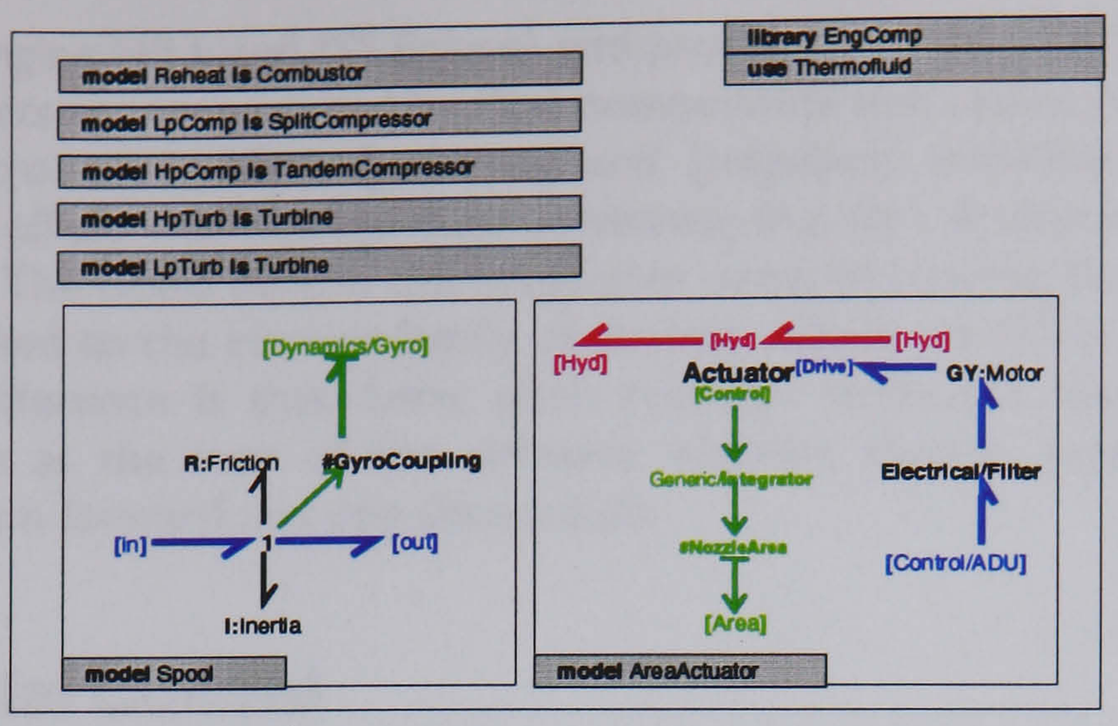


Figure 5.11 Engine Component Library

5.3.2 Reaction Control System

Figure 5.12 shows a schematic layout of the Reaction Control System (RCS) model. Recall that its purpose in STOVL aircraft is to provide attitude control power in jet-borne flight. The system concept is very simple, namely a network of pipes extending to the wing tips and to the empennage with pairs of RCVs at the extremities to generate torque on the airframe.

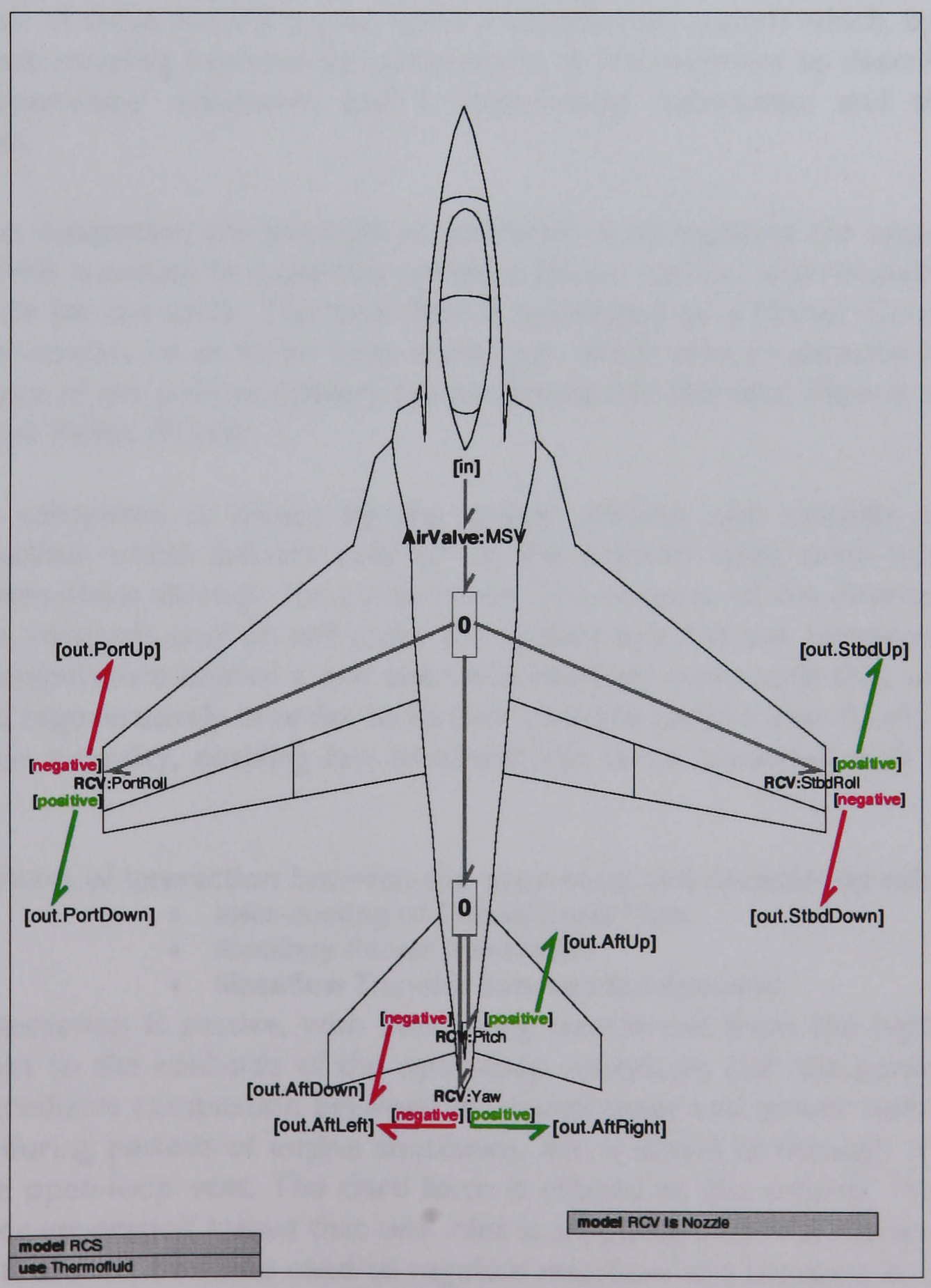


Figure 5.12 Reaction Control System Model

The system is fed by engine HP bleed (SS:Engine) and pressurisation is controlled by a Master Shut-Off Valve (MSV). The wing ducts are represented by Pipe components and each terminates in a two-way RCV component for roll control, with labels [positive] and [negative] denoting the sense of the rolling moment produced by the efflux. The fuselage duct terminates in a pair of two-way RCV components, one for pitch and one of yaw. The labels denote the respective sense of pitching and yawing moments. This is the same principle as applied to the Harrier family of aircraft, which use RCVs (or so-called 'puffer jets') to the same end. The difference is that, here, pitch reaction control is applied through upward and downward blowing RCVs at the rear of the airframe whereas Harrier incorporates two downward blowing RCVs, one situation forward and one situated aft.

5.4 Environmental Control

Environmental control is modelled in Figure 5.13. This configuration combines the functionality of a traditional ECS, together with an APU and an ISG, in a single package. Under normal airborne operation, the package is assumed to provide four main functions:

- **To satisfy pressure, temperature and massflow requirements for cabin conditioning**
- **To satisfy low temperature requirements for avionics cooling**
- **To provide a heat sink for the fuel system**
- **To provide supplementary electrical power generation**

Under other conditions, the package provides an auxiliary (or emergency) power function:

- **To provide limited cooling and electrical power in the absence of engine operation**

The combination of all of these functions gives quite a complicated system which, by its nature, exhibits a high degree of dynamic coupling between its components. It is convenient to describe this system in two parts, namely an 'open-loop' subsystem and a 'closed-loop' subsystem, and then to describe the interaction of the two.

For the open-loop subsystem, the principle of operation is to separate the engine bleed air into two flow paths, one of which is cooled by expansion across a power turbine, and recombine them into a single flow of conditioned air for the cabin. The total flow is modulated by a Master Control Valve (MCV) and passes through a pre-cooler, *i.e.* an air/air heat exchanger, which uses an external (variable flow) ram air source. If required, part of the turbine delivery can be diverted to the vent. Flow is split and merged using two-way Flow Control Valves (FCVs).

The closed-loop subsystem is driven by the power turbine and consists of a compressor, an intercooler and a turbine which delivers cold air to the avionics bays; once heated by contact with equipment the air recirculates through the compressor. The purpose of the intercooler is to reject heat so that high pressure, relatively cool air will enter the turbine and that low temperatures can be achieved by expansion. Low temperature implies a low saturated humidity point such that water can be extracted (and possibly be used regeneratively in order to further cool the turbine inlet flow). The water extraction process maintains low humidity, enabling low temperatures to be sustained with reduced potential for turbine icing.

There are three forms of interaction between the open-loop and closed-loop subsystems, namely

- **Inter-cooling of Closed-Loop Flow**
- **Auxiliary Power Generation**
- **Massflow Transfer between SubSystems**

The first form of interaction is passive, with heat being transferred from the high-pressure side of the closed-loop subsystem to the cold-side of the open-loop subsystem (for the purpose described above). The second form introduces combustion between the compressor and power turbine, thereby providing independent power during periods of engine shutdown. Air is drawn in through an auxiliary inlet and is expelled through the open-loop vent. The third form is related to the second. The mixing of flowpaths during auxiliary power generation means that one inlet is supplying both the compressor and the closed-loop subsystem and there will be some need to regulate massflow and pressure in that subsystem. In this mode, temperature control in the cabin can only be achieved by mixing the flows delivered from the two turbines in some proportion; regulation of the closed-loop subsystem will be a secondary requirement.

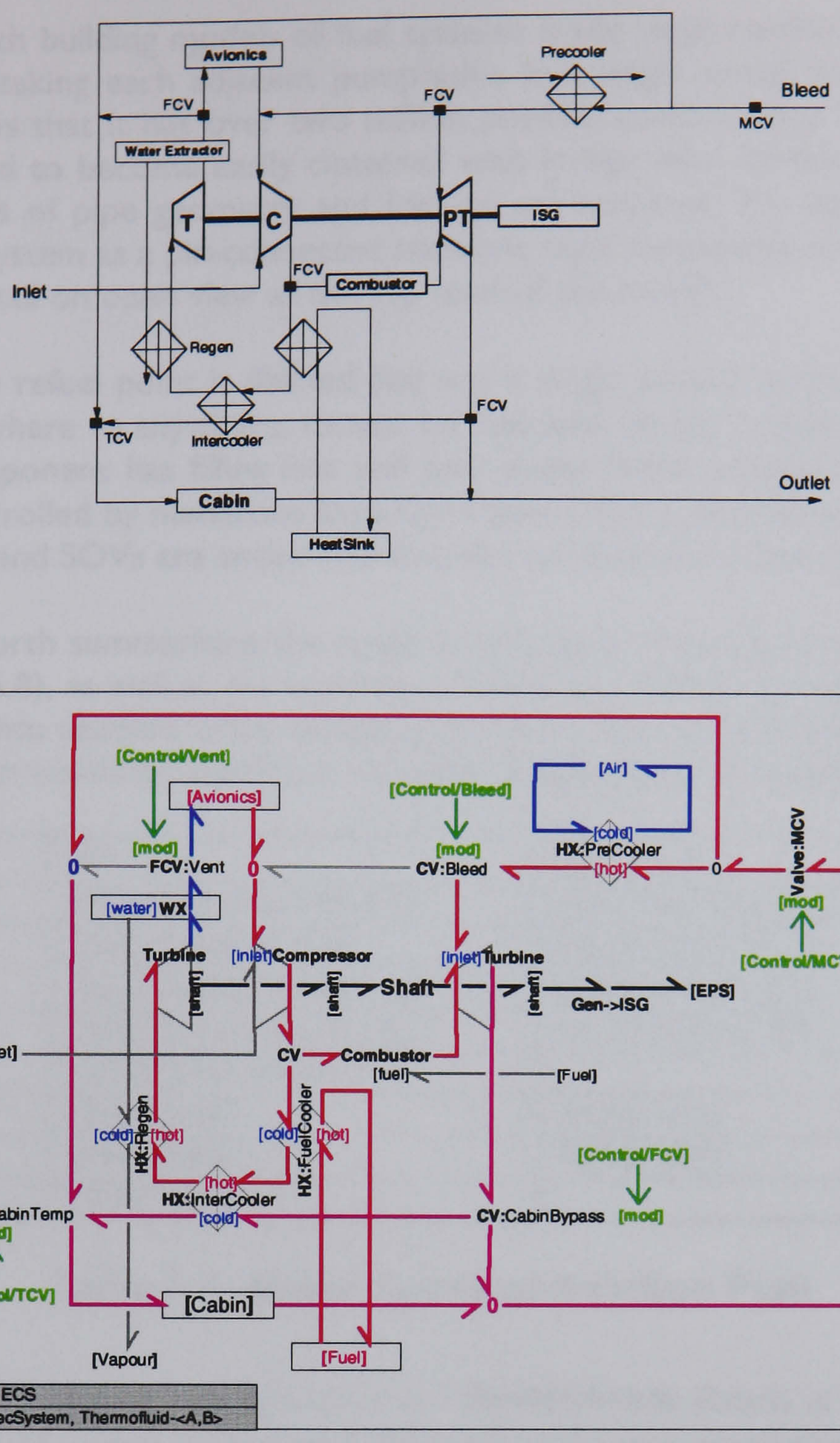


Figure 5.13 Environmental Control System Model

5.5 Fuel Management

Fuel Management (FUM) is modelled in **Figure 5.14** as an idealised four-tank fuel storage, transfer and feed system. There are numerous permutations and complexities in the design of actual aircraft fuel systems but the intention here is to address the overall system structure and the issues which concern the control of fuel CG, the modelling of gravity feed/transfer and the use of fuel as a thermal medium. For present purposes it is assumed that all tanks are the same, except for geometry, and that such effects as weiring, sloshing and low-fuel states (which in certain orientations might cause pumps to run dry) can be deferred for future study.

There are innumerable possibilities for controlling fuel movement, even for a system of this apparent simplicity; the objectives here have been to enable any point-to-point transfer between tanks, to provide some level of redundancy in fuel feed, to be able to isolate port and starboard sides and to allow a closed-loop recirculation between **Tank:Fwd** and **Tank:Aft** for heat exchangers, the air-cooled fuel cooling (ACFC) loop, **[Fuel]** , and the fuel-cooled oil cooling (FCOC) loop, **[Hyd]**. It is assumed that fuel will be used from wing tanks before fuselage tanks, thereby maximising authority over longitudinal CG position and support for heat transfer (given the chosen recirculation path).

The main problem with building models of fuel systems is the large number of switched components, valves and pumps. Even taking each adjacent pump/valve as a single entity, this system has twenty-one 'switches' .... which means that it has over two million possible combinations of switch states! Also, the bond graph diagrams tend to become easily cluttered with in-line valve components, a problem which is exacerbated if the details of pipe geometry and friction are required. For this reason, it is considered appropriate to map the system as a pin-connected network, with compound bonds being used in order to reduce the amount of detail on open view at the top level of the model.

For simplicity, a single refuel point is defined **[in]** and a single jettison point **[out]**. Effectively, fuel can be transferred from anywhere to anywhere, except for fuel feed via the collector box which is a one-way process. Each **Tank** component has filling line and two pump-driven outlets, one for feed and one for transfer. Fuel flow is controlled by numerous Shut-Off Valves (SOVs) and Non-return Valves (NRVs). The system is active full-time and SOVs are sequenced in order to open and close various paths.

For reference, it is worth summarising the range of fuel types in common usage (Table 5.1) and their main properties (Figure 5.8), as well as the varieties of additives (Table 5.2) and contaminants (Table 5.3) which have to be taken into account when designing fuel for airborne applications. In performance-based modelling, this information would be significant especially in relation to propulsion [*cf.* Section 5.3].

NATO	UK	MIL Type
F34	DERD 2453/ AVTUR FSII	MIL-T-83133D/ JP8
F35	DERD 2494/ AVTUR	MIL-T-83133D/ JP4
F40	DERD 2454/ AVTAG FSII	MIL-T-5624R/ JP4
F44	DERD 2452/ AVCAT FSII	MIL-T-5624R/ JP5
F43	DERD 2498/ AVCAT	
JET B	DERD 2486	ASTM 1655
JET A	DERD 2482	ASTM 1655
JET A1	DERD 2494	

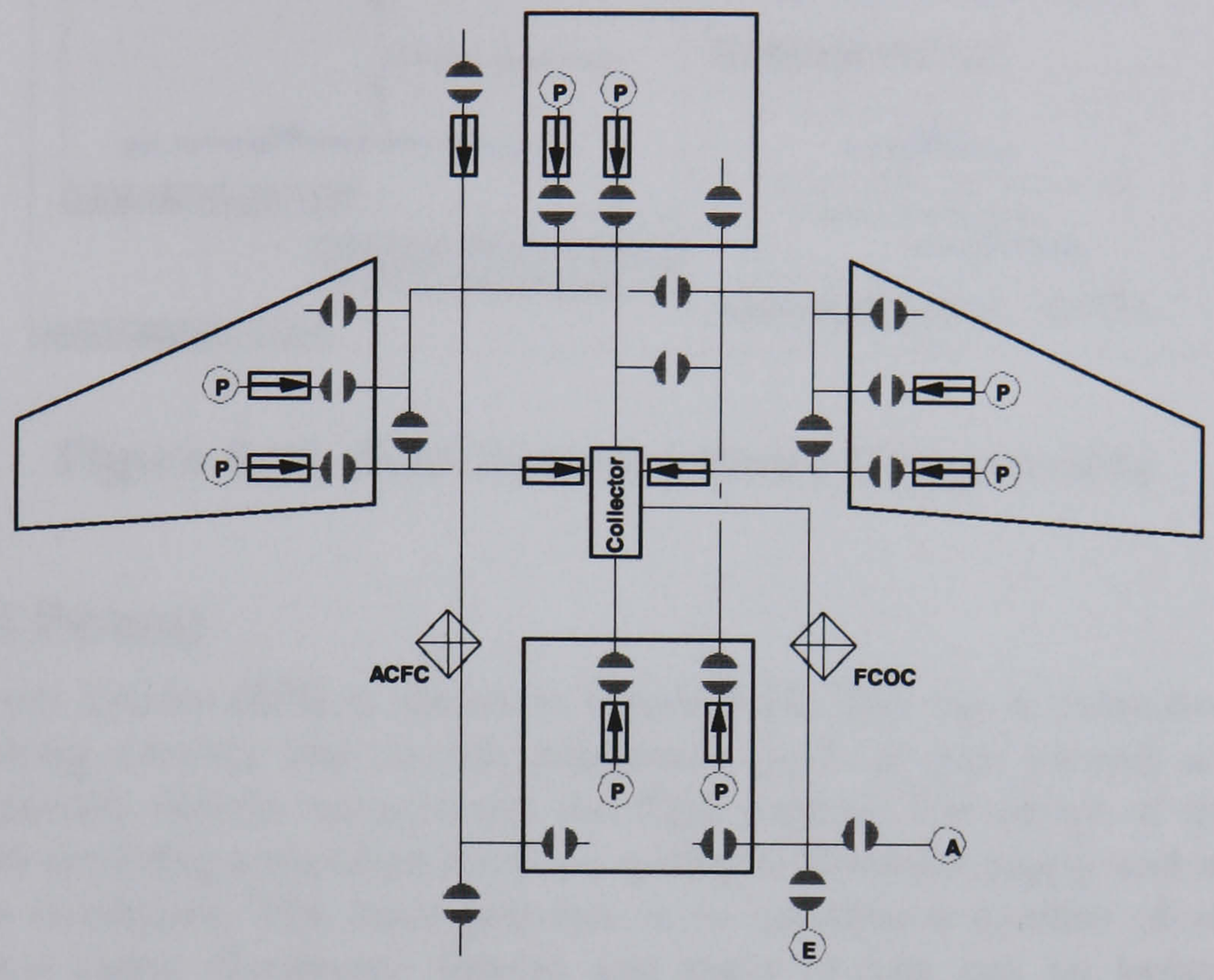
Table 5.1 Major Types of Aviation Fuel

The major fuel system components are introduced as a **FuelSystem** library, as shown in Figure 5.15. These are based on **Hydraulic** components [*cf.* Section 4.3.5] but could be re-interpreted as incompressible thermofluid components (as discussed in Section 4.5). The **Tank** and **Collector** components represent fuel storage, with a range of connections involving pumps and valves. Recall that a standard bond graph C component ordinarily imposes a pressure, consistent with integral causality. The provision of a port **[Dynamic/Head]** enables the effect of gravity to be accounted. A **FuelPump** is defined as a simple component, incorporating electrical drive components from the **ElecSystem** library [*cf.* Figure 5.17]. An elaborate model is not considered necessary because the primary aim is to represent gross fuel flow characteristics and, also, because issues of reverse flow are avoided by virtue of a non-return valve (NRV).

Fuel System Icing Inhibitor (FSII)	**DERD 2451**
Standard Type EGME	0.10%-0.15% by volume
High Flashpoint DI-EGME	0.12%-0.20% by volume
Corrosion Inhibitors	**DERD 2641**
Hitec E515	11-21 mg/l
Apollo PRI 19	9-23 mg/l
Tollad 245	21-34 mg/l
Emery 9855	13-34 mg/l
Du Pont DCI-4A	9-23 mg/l
Static Dissipators	
Shell ASA 3	1 mg/l
Du Pont Stadis 450	3 mg/l
Anti-Microbiological	
Methyl Cellusolve	0.15% by volume
Biobar JF	270 ppm

Hydraulic Fluid (DTD 585)	DEF-STAN 91-48 Grade Superclean
Solvents and Cleaning Fluids	Propan-2-ol (isopropyl alcohol), ref. BS 1595 Borothene (Amyity UK) Detergent No. 5 (TS 10281)
Lubricating Oil (OX-38)	DERD 2487

Table 5.3 Typical Fuel Contaminants



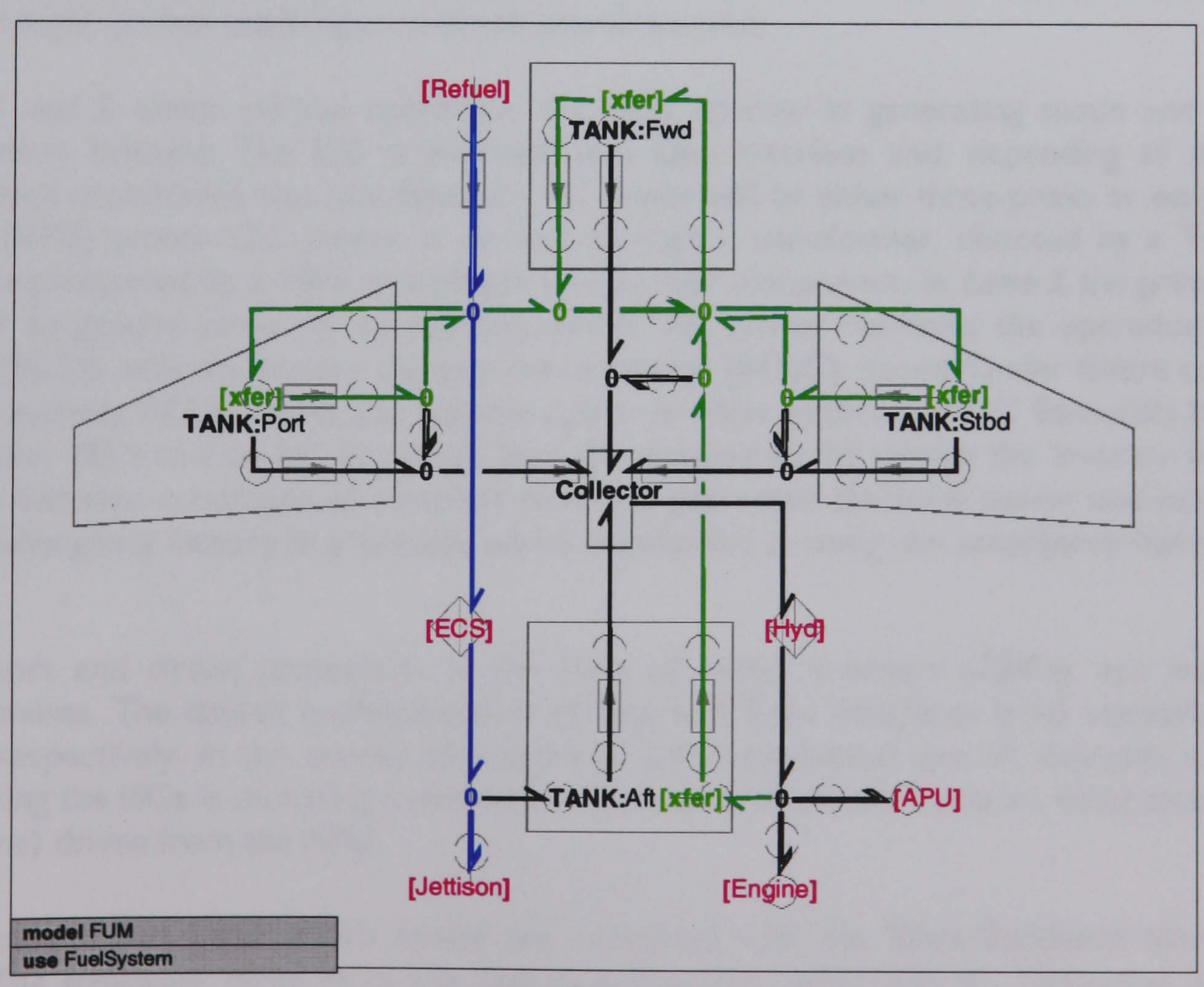


Figure 5.14 Fuel Management System Model

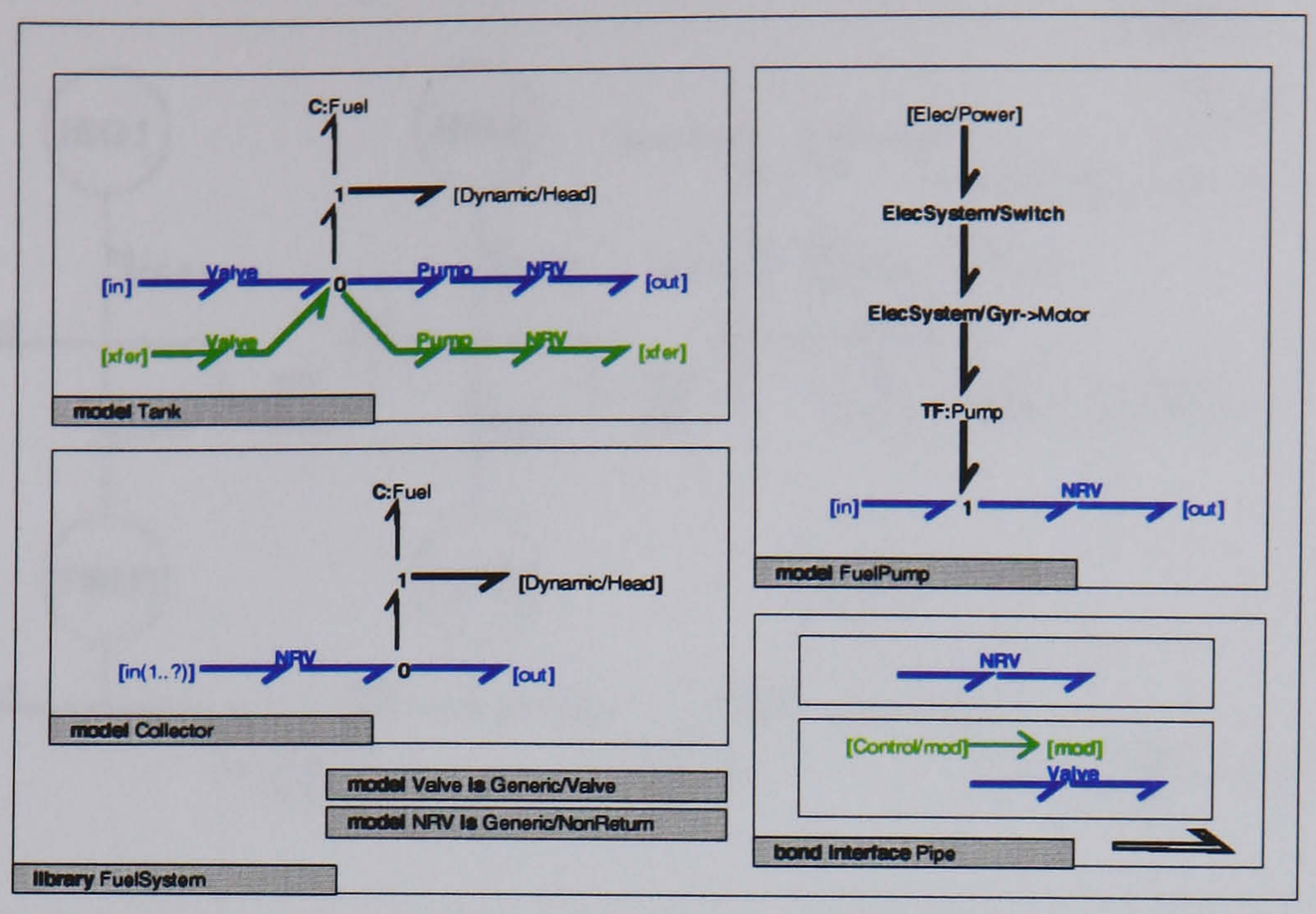


Figure 5.15 Fuel System Library Components

5.6 Electrical Power

The Electrical Power System (EPS) is shown in **Figure 5.16**. This has a three-lane configuration with Lanes 1 and 2 supplying avionics and aircraft equipment (such as fuel pumps) and Lane 3 supplying essential systems, especially vehicle management and flight control. The design of this class of system is mainly concerned with providing a specified level and quality of electrical supply and maintaining it under a wide range of failure conditions. The basic principle is to combine a number of voltage sources using redundant transmission paths. Generator failures and short circuits can be isolated while alternative sources are brought on-line ensuring a no-break power transfer.

In *Lanes 1 and 2*, under normal operation, the ISGs operate in generating mode and supplies AC power to primary busbars. The ISG is inserted as a **Gen** interface and, depending of the choice of component which implements that interface, the AC power will be either three-phase or equivalent Root-Mean-Square (RMS) power. DC power is derived through a transformer, denoted by a **Tran** interface which can be implemented by a TRU or a simple transformer component. In *Lane 3*, the primary busbar is supplied either by ground power or by auxiliary power. AC power maintains the operation of a Battery Charger Unit (BCU), with the battery charger line contactor (BCLC) closed. Under failure conditions (*i.e.* loss of all AC power), BCLC opens and battery power is made available to DC Essentials by closing the battery contactor (BC) and to AC Essentials through an **Invertor** by closing the invertor line contactor (IVLC). Under extreme conditions of complete failure of generated electrical power and battery back-up, a single-shot emergency battery is provided, which is switched in using the emergency battery contactor (EBC).

All contactors and circuit protection, in the form of circuit breakers (CBKs), are represented by **Switch** components. The default implementation of **Gen** and **Tran** interfaces is via standard **GY** and **TF** components, respectively. In the overall philosophy of this hypothetical aircraft example, engine start is achieved by using the ISGs is motoring mode rather than, as in traditional systems, using an air motor (*i.e.* a power turbine) driven from the APU.

Electrical components used in this model are contained with the **ElecSystem** library shown in **Figure 5.17**. The significant detail lie in the **Switch** component, which has its modulation driven from a **Control/** distribution, and in the **Bus** component, which incorporates an embedded port carrying the instance name of the bus. The **Invertor** shows two gyrators, representing a DC motor and a single-phase AC generator; note that the AC connection would be associated with the first element of a three-phase system if connected to such a system.

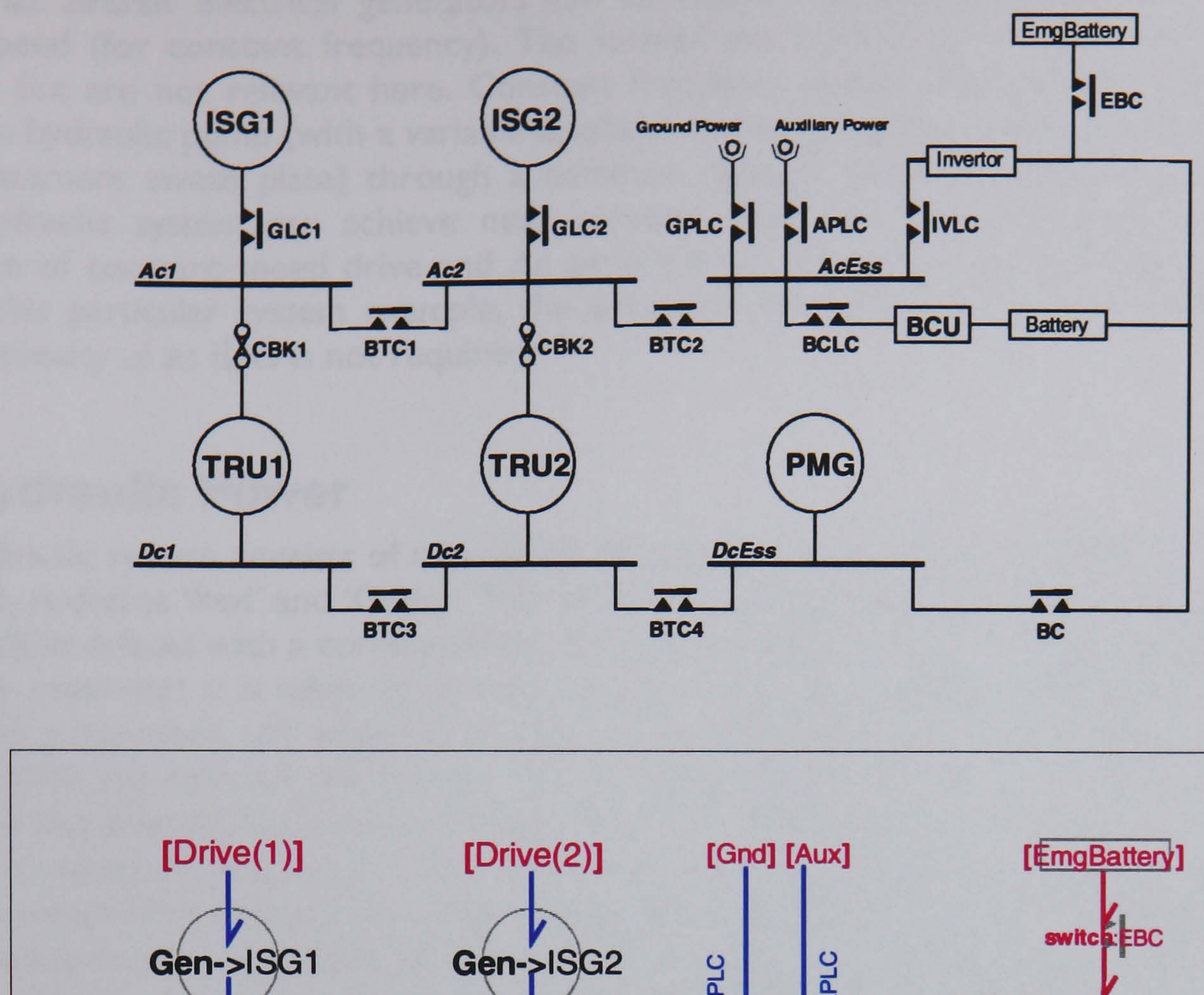


Figure 5.16 Electrical Power System Model

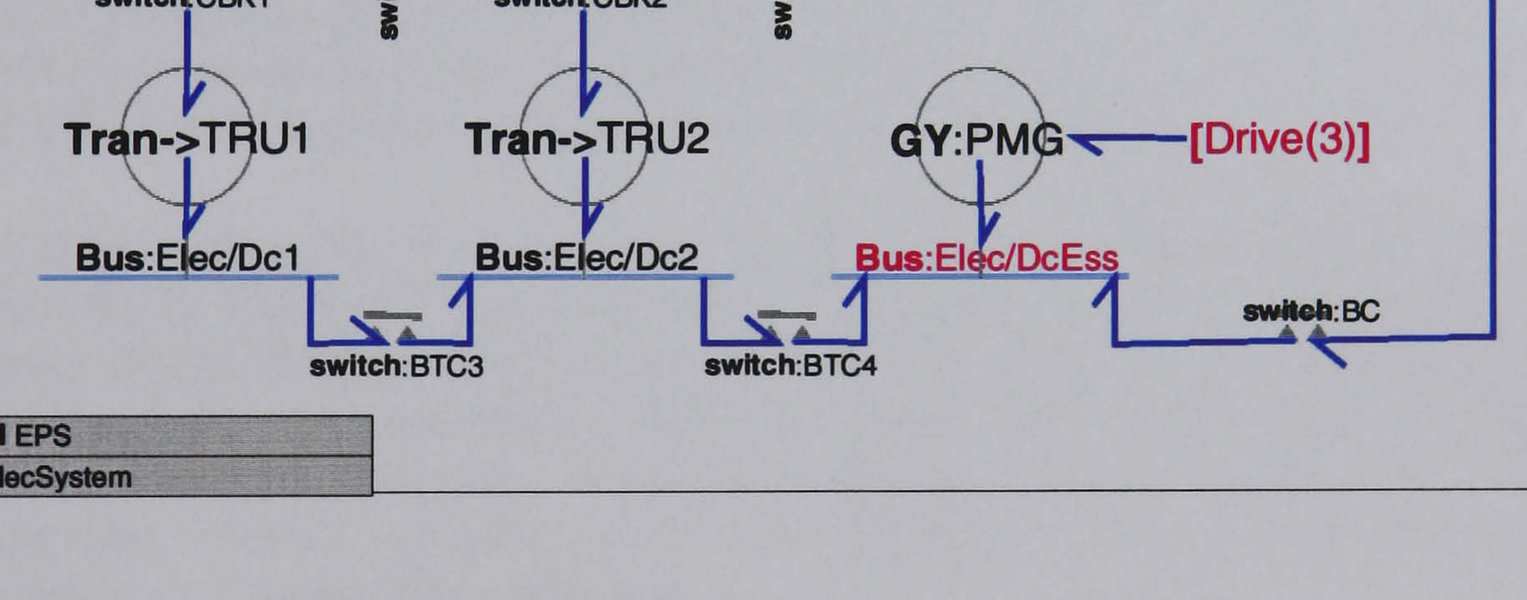


Figure 5.17 Electrical Component Library

Note that aircraft electrical generators can be variable speed (giving a frequency-wild source) or constant speed (for constant frequency). The former are applied to resistive loads and to Dc power conversion but are not relevant here. Constant frequency supply requires a variable ratio drive which comprises a hydraulic pump (with a variable displacement swash plate) linked to a hydraulic motor (with a fixed displacement swash plate) through a common cylinder block. By regulating the swash angle the internal hydraulic system can achieve near constant shaft speed at the output of the motor. The combination of constant-speed drive and Ac generator is referred to as an Integrated Drive/Generator (IDG). In this particular system example, the frequency of supply will not be modelled and, thus, the added complexity of an IDG is not required.

5.7 Hydraulic Power

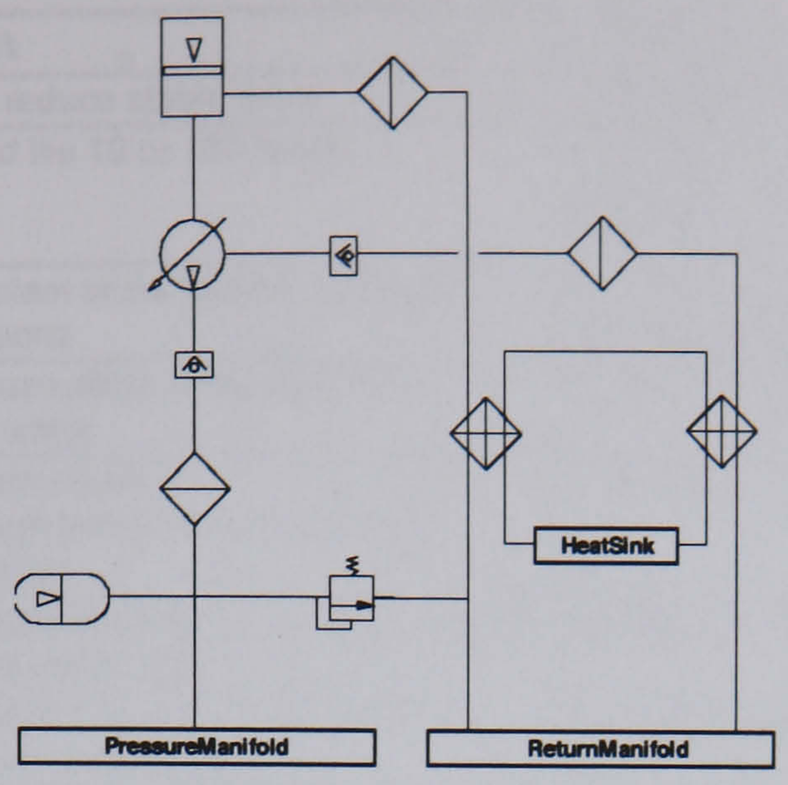
The hydraulic system consists of two identical hydraulic power modules (HPMs), of the type shown in Figure 5.18, coded as 'Red' and 'Green'. The modules operate independently and provide a self-contained circuit which interfaces with a corresponding distribution circuit. Fluid is routed *via* the return manifold to a bootstrap reservoir; it is taken by suction into a variable-pressure pump (represented by a VariPump component), pressurised and supplied through the pressure manifold. The nominal operating pressure is 27.5 MPa (4000 psi) although the actual pressure will be subject to variation under software control. An accumulator has been incorporated to satisfy peak flow demands from consumers. Flow from the pump case drain is routed directly to the return manifold. The heat content of that flow and of the main return flow to the reservoir is managed by a fuel-cooling loop, passing through heat exchanger (**HX**) components and terminating in a through-port, **[FUM]**. A pressure relief valve (**PRV**) is provided between supply and return in order to cope with transient over-pressure. In practical systems, thermal relief is also an issue although, for simplicity, it is not included here. Other minor components are non-return valves (**NRV**) and oil filters (Filter).

The **HydSystem** component library is shown in Figure 5.19. The **VariPump** component is based on a standard hydraulic pump but adds a yoke actuator to change the swash-plate angle. A **Filter** is assumed to a laminar restriction as defined by (4.3), combined with a capacitive volume. The **PRV** operates a **#PressureRelief** CR which basically would define some form of hysteresis mechanism.

The hydraulic fluid incorporated in the model is DTD 585 (of MIL-H-5606C). For reference, other fluids are summarised in Table 5.4 and their main properties are shown in Figure 5.12. Performance characteristics are a direct function of the base fluid and various additives. The classes of additive, their function and chemical types are described in Table 5.5. Note that additives enhance the performance of the base fluid and provide characteristics which assist the operation of the particular hydraulic machine. DTD 585 requires that additive materials in order to improve the viscosity-temperature characteristics, resistance to oxidation and anti-wear properties.

DTD 585	Most common fluid in use military aircraft and consists of a mineral oil with additives, including a polymeric Viscous Index improver.
	Also known as MIL-H-5606C, NATO H-515 and Joint Services Designation OM-15.
	Quoted temperature range: -54°C to 135°C
Chevron M2-V	Silicate ester developed from fluids originally designed to meet MIL-H-8446.
	Used in Concorde and Rockwell B1.
	Quoted temperature range: -54°C to 260°C
Skydrol 500B	High density type II phosphate ester.
	Suitable for civil aircraft, except Concorde.
	Quoted temperature range: -54°C to 102°C
Versilube F50	Chlorinated phenyl silicone fluid.
	Used in small quantities in constant-speed drives.
	Also known as MIL-S-81087B, NATO H-536 and Joint Services Designation OX-50
	Quoted temperature range: -54°C to 232°C

Table 5.4 Hydraulic Fluids



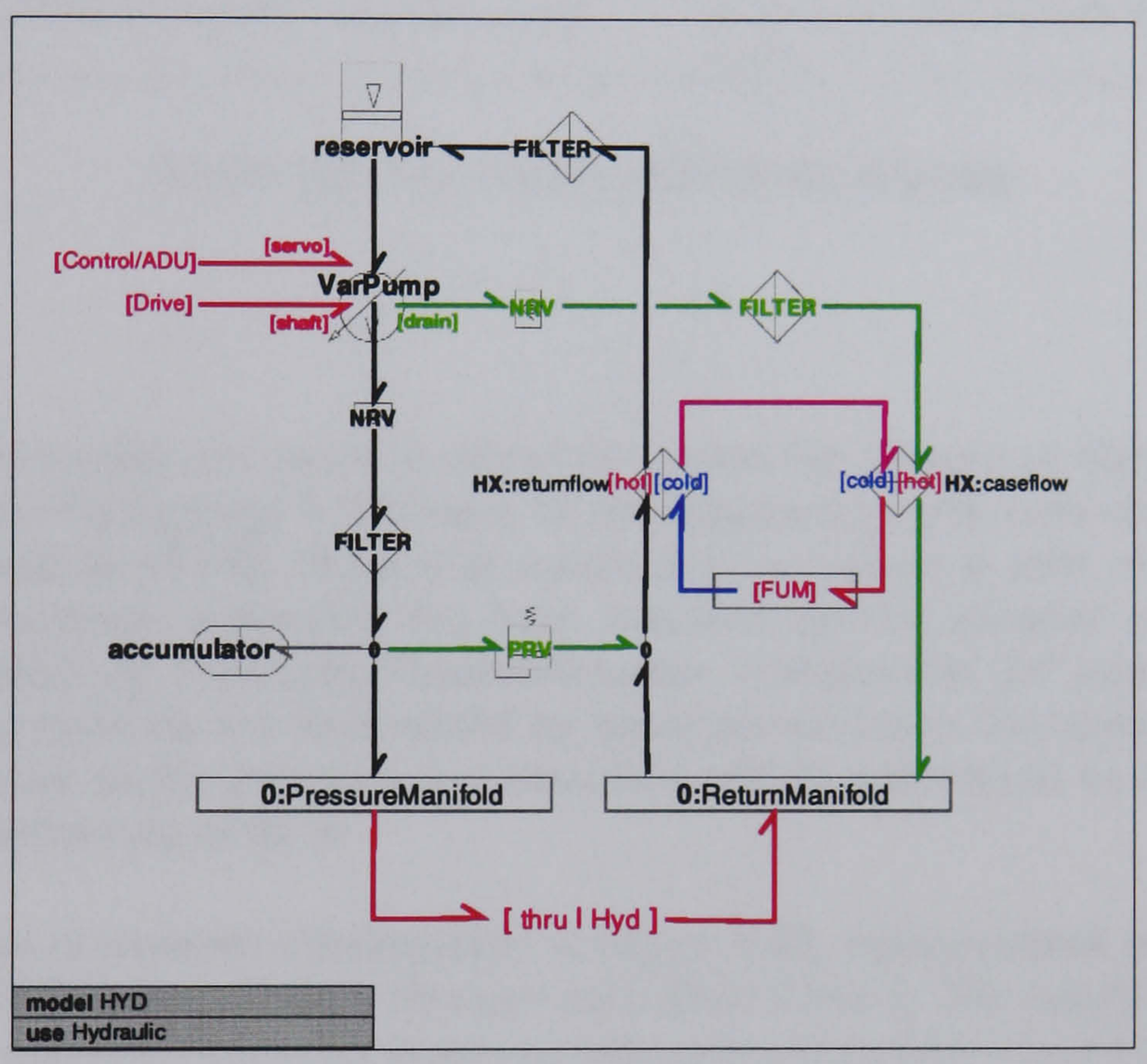


Figure 5.18 Hydraulic Power System Model

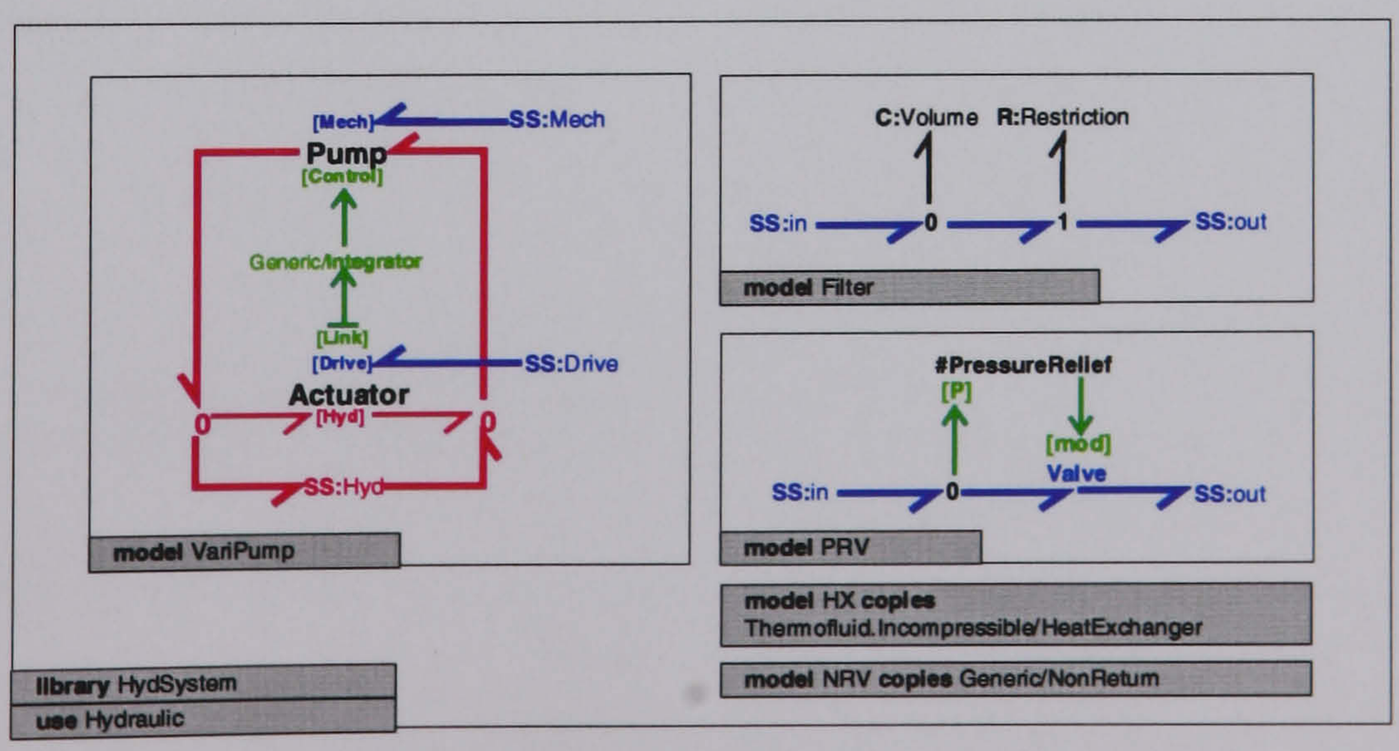


Figure 5.19 Hydraulic Component Library

Additive Class	Function	Chemical Types
Antifoam agents	Prevent of reduce stable foam	High molecular mass silicones and acrylate esters
Antioxidants	Extend fluid life 10 or 100 times	Zinc dithiophosphates, organic sulfides and thiophosphates, hindered phenols and aromatic amines
Antiwear agents	Protect system under startup and high-load conditions	Zinc dithiophosphates, organic sulfides and other thiophosphates
Demulsifiers	Enhance separation of suspended or emulsified water	Vary with application
Detergents/Dispersants	Keep system clean, enhance high temperature stability of hydraulic fluids	Sulfonates, phenates, succinate dispersants
Metal deactivators	Reduce catalytic activity of system metals and reacted metal ions	Organic triazoles, thiadiazoles and thiazoles
Pour depressants	Inhibit formation of wax and extend low temperature operating range	Acrylate polymers, alkylated naphthalenes
Rust inhibitors and other corrosion inhibitors	Prevent rust and chemical corrosion of system metals	Organic acids/esters/salts, metal sulfonates
Viscous Index (VI) improvers	Extend operating temperature range	Acrylates, methacrylates, isobutylene polymers, etc.

Table 5.5 Hydraulic Fluid Additives

5.8 Actuation

For purposes of this model, the primary actuation system for integrated flight/propulsion control is assumed to be hydraulic. Fluid power is provided to the flaperons (4-off), tailerons (2-off), rudder (1-off) and engine vectored nozzles (4-off). Note that nozzle area actuation is part of the engine model [*cf.* Figure 5.10]. For convenience, uniformity has been imposed on the actuator configuration, all eleven devices being represented by Hydraulic/TandemActuator components [*cf* section 4.3.5.8]. The load conditions on the flying controls are determined by aerodynamic hinge moments; the aerodynamic load on the nozzles is assumed to be unknown and therefore will be substituted by friction consistent with achieved a nominal rotation rate of 90°/s.

Hydraulic distribution is depicted schematically in Figure 5.20, superimposed on the aircraft planform. The interface with the HPMs is a multiple through-port, [Red,Green]. The supply/return lines are shown as bi-directional bonds and distribution to flaperons, tailerons and rudder is achieved using 0 components as bi-directional flow junctions.

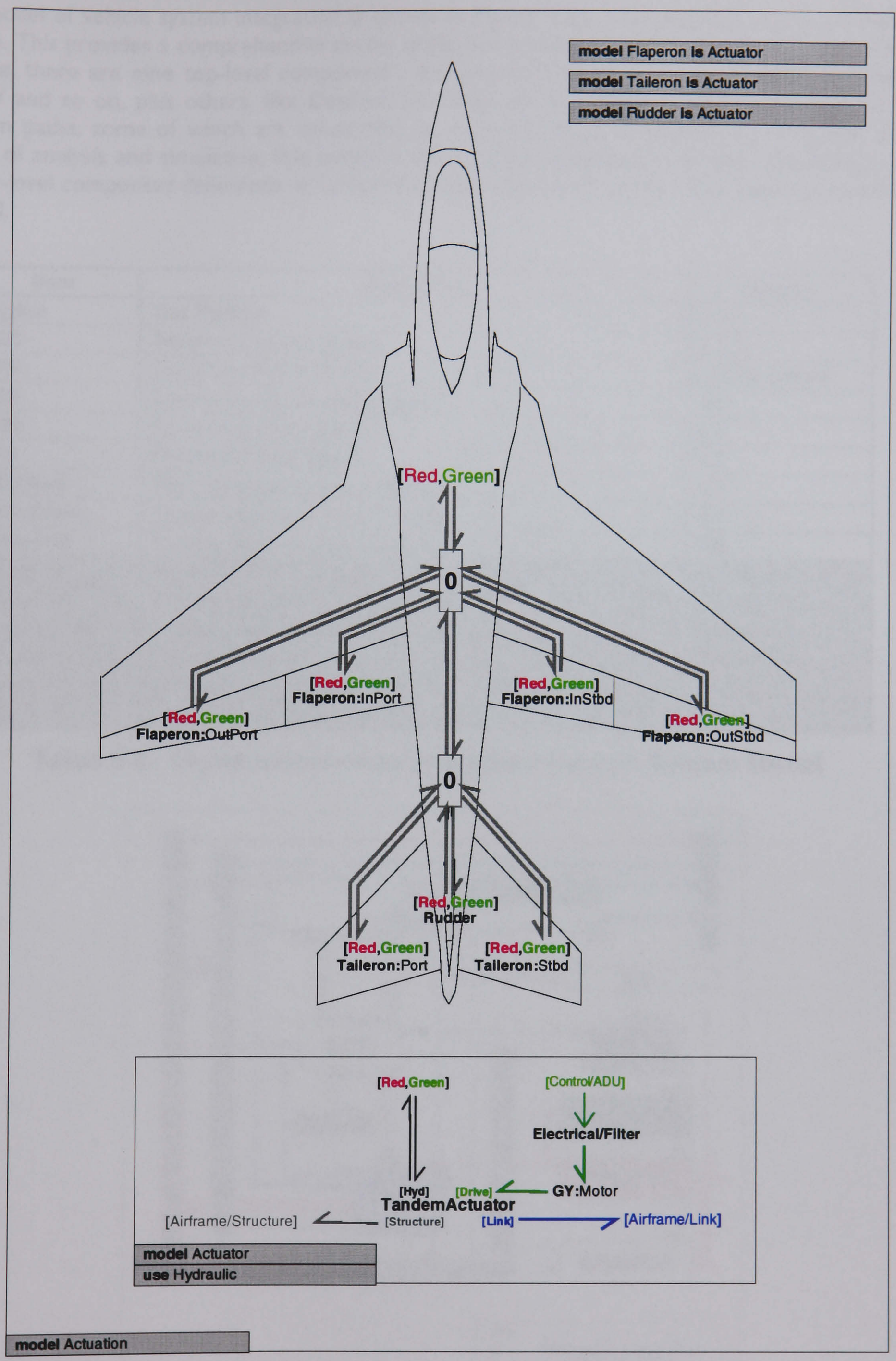


Figure 5.20 Flight Control Actuation Model

5.9 Vehicle System Integration

The model of vehicle system integration is shown in **Figure 5.21**. It incorporates the items defined in **Table 5.6**. This provides a comprehensive model of physical interaction between constituent systems. To summarise, there are **nine** top-level components, **six** external connections, **six** distributions (**Airframe/**, **Dynamic/** and so on, plus others, like **Control/** that may remain hidden at this level) and **thirty-three** interaction paths, some of which are structured or vectored bonds. It should be recognised that, for purposes of analysis and simulation, this model is already very complicated, without attempting to refine the lower-level component definitions or further decomposing them in order to increase the resolution of the model.

Item	Description	Section
Engine	Gas Turbine	5.3.1
RCS	Reaction Control System	5.3.2
SPS	Secondary Power System	Locally defined
ECS	Environmental Control System	5.4
FUM	Fuel Management System	5.5
EPS	Electrical Power System	5.6
HYD:Red	'Red' Hydraulic Power Module	5.7
HYD:Green	'Green' Hydraulic Power Module	5.7
Actuation	Primary Flight Control Actuators	5.8
Airframe/	Airframe reference parameters	Locally defined
Dynamic/	Dynamic motion parameters	Locally defined
Aerodynamic/	Aerodynamic forces/moments and atmospheric data	Locally defined
Propulsion/	Propulsive forces/moments	Locally defined
Elec/	Electrical power distribution	Locally defined
Thermal/	Thermal loads	Locally defined

Table 5.6 Components of an Integrated Vehicle System Model

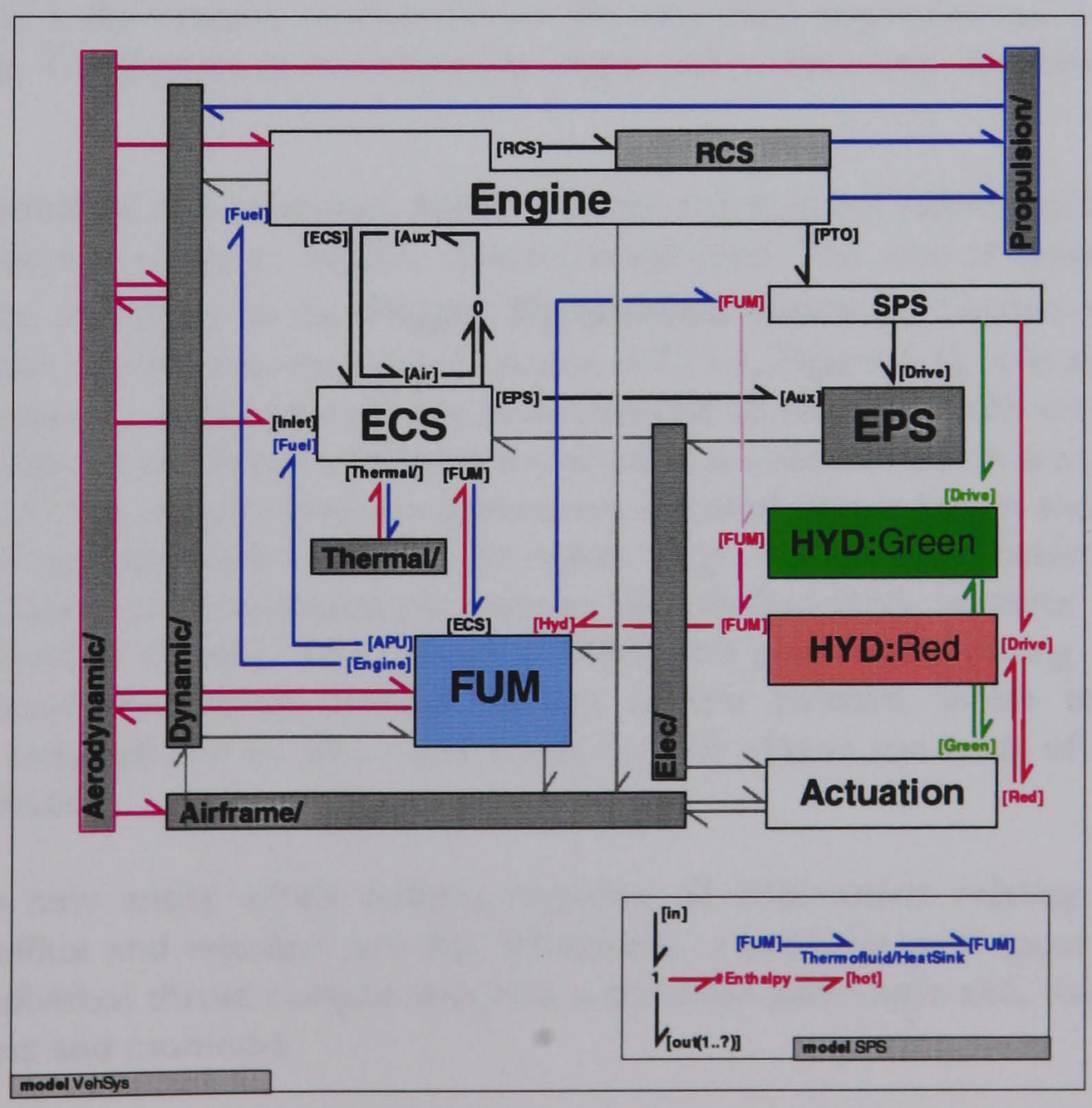


Figure 5.21 Integrated Vehicle System Model

This model is equivalent to the conceptual model of power flow in Figure 5.7. The use of two different notations suggests that, for providing a high-level overview, there is nothing special about bond graphs. However, what a bond graph does provide is a rigorous means of laying out the architecture of a mathematical model and ensuring that the components are compatible. The assembly of the actual mathematics (in the form of differential and algebraic equations) is hidden from view, enabling the modeller to concentrate on modelling and the physics of his/her system of interest. What is also hidden in the integrated model is the compatibility of energy domains but, as a result of constraints imposed on the various model libraries, there is sufficient information in order to establish the type of energy being conveyed by each bond.

Bond orientation is imposed at top level and is propagated down the model hierarchy. Causality has not be assigned here but the external ports are open to the atmosphere (because we are dealing with an aircraft!) and, as such, are set up to impose effort, *i.e.* either static pressure or stagnation pressure. Some of these ports are allocated to the **Aerodynamic/** distribution and the others have their information routed via the **Propulsion/** distribution (which also carries information regarding engine thrust). The **Elec/** distribution will impose effort, *i.e.* voltage; the **Dynamic/** and **Airframe/** distributions will impose flow, *i.e.* velocity and angular velocity. The heat 'flows' are convective thermofluid phenomena and, in this project, have been represented by structured bonds carrying hydraulic power (with *compressible* flow being massflow and *incompressible* flow being volumetric flow) and information on temperature and enthalpy flow (usually in the form of an information bond). Thus, the **Thermal/** distribution will impose hydraulic pressure and will propagate either temperature or enthalpy flow in the direction of hydraulic flow. What is interesting is that, where true power bond are being considered, it appears to be most intuitive to assign an *across* variable and allow the associated *through* variable to float.

The distributions (which optionally show the presence of *global* variables) give a strong focus of attention to functional interfaces that have yet to be defined. The main internal issue to be resolved is the allocation of electrical consumers (*e.g.* fuel pumps, avionic equipment) to particular generators, both under normal operation and under failure conditions. In practice, because this involves a substantial amount of load analysis and decisions on load-switching and load-shedding, this aspect of systems engineering design demands a high degree of flexibility. Hence, in this context, it would always be appropriate to employ a distribution mechanism so that electrical engineers can work off-line from the mainstream modelling. This feature is not currently supported in the many implementations of the bond graph method.

The **Dynamic/**, **Airframe/** and proposed **Aerodynamic/** distributions reference the three components of Figure 5.1 which do not relate to vehicle systems integration. The *first* of these has been dealt with quite extensively in the definition of the **Flight Dynamics** library [*cf.* Section 4.7] and, according to the causal assignment in the Motion model [*cf.* Section 4.7.11.7, Figure 4.6], it is apparent that the main interest is to collect together applied forces and moments and to return 6-DOF velocity information. The *second* deals with all aspects of aircraft structure and physical installation which are peripheral to the main purpose of this project. The *third* covers the generation of aerodynamic forces and moments, expressed typically in terms of stability and control derivatives [*e.g.* Etkin 1972, Babister 1980]. It covers, additionally, the calculation of atmospheric parameters [British Standards Institute 1984], the generation of windshear [Woodfield & Woods 1983, Schanzer 1983] and gust loads [Stirling 1987]. These detailed features are determined by aircraft motion relative to the airmass, which involves calculation of orientation, velocity, airspeed and so on .... and it was for this reason the body of equations from (4.99) to (4.111) was introduced.

Propulsion/ is an new entity which collects together all information relating to the generation of thrust, covering jet efflux and reaction control. Effectively, it provides a wrapper for Engine and RCS which resolves the individual thrust components into a common axis frame and, using airframe geometry, determines total forces and moments.

5.10 Model Initialisation

In the interests of being able to apply an integrated model for engineering analysis, it is necessary to clarify the procedure for initialising the many constituent parts. It is not sufficient, for models are large and complex of these, to rely on elaborate and all-embracing optimisation techniques for finding steady states.

Section 4.7.12 was included explicitly for this purpose. It is strongly recommended that the operating condition is determined from a balanced flight condition, as defined in (4.112) through to (4.115). The balance can then be established between aerodynamic and propulsive forces, both from fixed airframe geometry and (in this case) from thrust vectoring [Appleyard 1987]. With this information, engine and aerodynamic control settings can be calculated, taking into account reaction control and jet effects (in jet-borne flight), known mass state of the aircraft and a prescribed set of power/cooling requirements for the mission segment corresponding to the current flight condition. With these requirements, the vehicle systems can be initialised almost in isolation of from the vehicle itself. The issue is to resolve the power balance between EPS, HYD, FUM and ECS, especially to formulate the correct budget for thermal management. To this end, segments of the bond graph model of the integrated vehicle system can be configured in order to find the equilibrium state [Breedveld 1984a].

From experience, this is a relatively straightforward matter provided that not too much detail is contained within the model: as more detail is incorporated, the amount of numerical computation increases considerably. It is believed that, in general, the initialisation of complex system models should be achieved using two standards of model. The first has to be sufficiently accurate in order to home in on the approximate steady state using the two-stage optimisation procedure described above. The second, if required, would be fully compliant with the accuracy requirement for a specific purpose and would be driven by simulation in order to converge to a balanced steady state. Clearly, there are many context dependencies involved in nonlinear modelling and it is inconceivable that no problems will be encountered.

5.11 Conclusion: Towards the Virtual Aircraft

This chapter has developed an idealised pattern of power flow, as shown in Figure 5.1, into a complete top-level specification for a virtual aircraft model, with particular emphasis on vehicle system integration. It is believed that the use of traditional signal flow notations would produce a equivalent visual representation which would be so complicated as to be unintelligible. The combination of the bond graph method (which closely associates power covariables with physical system concepts), some of the DDM principles [*cf.* Section 2.5] (which facilitate model hierarchies and views) and the notational refinements have enabled the construction of a very compact and information-rich model.

All the external connections are allocated to distributions, which serve the function of global data, in order that the connections are categorised and accessible. Establishing these as global data, rather than encapsulated ports, allows the use of entirely separate methods for interfacing with other models. Also, it allows flexibility in implementing memory management strategies, which is crucially important in hardware/software integration for real-time simulation.

Overall, the creation of a virtual aircraft is considered feasible using bond graph modelling and, to this extent, it is concluded that bond graphs are well suited (if not uniquely suited) to building high-level concept models. From the viewpoint of power transmission, the integrated system model and the subsystem interactions that it contains are handled in a multi-disciplinary framework. Usually, these relate to specialised disciplines in their own right but here it has been demonstrated that these can be harmonised. This is especially important for integrated control system design, which is a major concern for future aircraft and, thus, a virtual aircraft is believed to be a worthwhile facility.

Chapter 6

Conclusions and Recommendations

SUMMARY
The stated purpose of this thesis is shown to have been satisfied and, given the relative immaturity of academic interest in systems integration or whole-aircraft modelling, it is believed that this is the first attempt to apply the bond graph method to complex aerospace systems in any meaningful sense. Although the work presented here deals explicitly with an aerospace context, the underlying principles are wholly generic and could offer a great opportunity in other areas of systems integration. During the course of this project, a number of different challenges have become apparent, any or all of which stand as valid and valuable topics for future research and development. These are outlined under the headings of Modelling, Control Design, Design Data Management.

6.1 Overview

The purpose of this thesis is threefold. Firstly, to demonstrate the application of bond graphs as a unified modelling framework for aerospace systems. Secondly, to review the main principles involved with the modelling of engineering systems and the specific issues that arise in the aerospace context. Finally, to present an exposition of the bond graph method (including major modifications which were found necessary in order to provide compact representations of big systems). With this in mind, the main conclusions from each chapter are summarised below.

6.1.1 Chapter 2: Aims of a Unified Modelling Method

Mathematical modelling is an integrating technology for *development* of complex airborne systems and it is clear that a highly structured and intuitive approach will be required to the *modelling* of complex airborne systems. The bond graph method is effective in representing physical processes and offers a close topological mapping of systems. The ultimate justification of this method is based on its ability to satisfy the following principles:

- **Simplicity**
- **Expressibility**
- **Consistency**
- **Verifiability**
- **Reusability**
- **Extensibility**

A range of method developments inderpin this thesis, directed towards the improvement of existing notations in order to provide a modelling method that can be readily applied to aircraft vehicle systems and assist engineers in design and evaluation tasks. Standard bond graph concepts have been modified and extended in order to handle practical engineering situations.

6.1.2 Chapter 3: A Revised Bond Graph Method

The framework is established within which bond graphs can be applied to the modelling of air vehicle systems. A baseline notation is defined, incorporating a significant number of novel features and notational changes that have been found to improve the applicability of the bond graph method to a system development context. The concepts of bond graph structure are discussed and placed into the formal context of an information model.

6.1.3 *Chapter 4:* Building Blocks!

The application of bond graph modelling is demonstrated across five relevant energy domains and libraries have been established to a level appropriate for building conceptual models of aircraft systems. The aim is to summarise the main physical principles that are of interest in each domain and to build bond graphs of major equipment components. On route to component modelling, much work has had to be done in order to prove the feasibility and usability of new notational features introduced earlier. Based on the work presented in this chapter it can be argued with confidence that the method, apart from offering a very convenient shorthand notation, simplifies the process of building mathematical models.

6.1.4 *Chapter 5:* Towards the Virtual Aircraft

A complete top-level specification for a virtual aircraft model is developed with particular emphasis on vehicle system integration. The combination of the bond graph method (which closely associates power covariables with physical system concepts), the principles of Design Data Management [*cf.* Section 2.5] and the notational refinements have enabled the construction of a very compact and information-rich model.

6.2 Conclusions

Overall, the creation of a virtual aircraft is considered feasible using bond graph modelling and, to this extent, it is concluded that bond graphs are well suited (if not uniquely suited) to building high-level concept models and to conducting consistency checks on models. From the viewpoint of power transmission, the integrated system model and the subsystem interactions that it contains are handled in a harmonised framework spanning many specialised disciplines. This is especially important for integrated control system design and, to this end, a virtual aircraft is believed to be a worthwhile facility.

It should be noted that, in spite of the apparent ease with which integrated models can be drawn [*cf.* Figures 5.7 and 5.21], they hide enormous complexity. In engineering terms, there may be many levels of decomposition that are significant to the integrated dynamics of the system. These may involve hundreds or thousands of components and a vastly greater number of component interactions. As suggested by the 'motivating' example [Section 2.11.4], signal flow would not provide an adequate or readable representation. Two-port networks are rather more effective although pre-defined causality can lead to a large amount of work when refinements are made to a model. In contrast, bond graphs are very effective straightforward and intuitive.

Reviewing the qualities of bond graph modelling, it is necessary to justify the approach against the principles identified in Chapter 2 [Section 2.10]. The key arguments are summarised below:

Simplicity

With a small amount of practice, engineers can become proficient with the notation and use it to represent complicated system topologies. More importantly, from experience, they use it to question and debate how systems function and what the performance determinants are. The close mapping of 'system' and 'model' is extremely useful as a semantic mechanism, as opposed to anonymous mathematical expressions.

Expressibility

As evidenced in this work, it is striking that bond graphs have the ability to draw compact functional models of complex systems and to cover entire domains with small numbers of component definitions.

Consistency

Much has been made of consistency rules in Chapter 3. Clearly, once models involve pseudo-bonds, there is no longer any guarantee of 'plug and play'. Also, as discussed in Chapter 4, composite domains for thermal convection and material transport can exacerbate these problems. This removes bond graphs from the 'back of an envelope' as far as detailed model assembly is concerned and, thus, necessitates proper computer-aided support. For this reason, an outline information has been defined in Chapter 3.

Verifiability

Three main issues arise here, namely connectivity, propagation and partitioning. All are explicitly built into the revised bond graph method and, with computer-aided support, functional checking and testing can be automated to a large extent.

Reusability

Component re-use is supported by definition. The bond graph method is equation-based and therefore free from any arbitrary constraints on causality or orientation. The discussion of port binding [Section 3.8.3] indicated a great deal of the re-use problem in the sense that it relies on rigorously specified interfaces and properties.

Extensibility

Changes to notation are essential to the continual evolution of any method, to meet new technical challenges and customer requirements. The adaptation of the bond graph notation undertaken in this thesis, together with the introduction of new concepts, adequately demonstrates this principle.

A number of general (modelling) principles were summarised [*cf.* Section 4.2] in order to make explicit the main factors in the decisions that modellers must take. They re-state the 'method' principles in such a way as to lay down guidelines for the development of models and model libraries. These have been effective in reducing very complicated (and sometimes long-winded) dynamic problems to very compact (and almost trivial) bond graphs. In itself, this is an original contribution to the field of systems engineering that should be neither under-estimated nor under-valued.

An exposition of the bond graph method has been provided, including major modifications and novel features which, it is believed, improve its usability as a practical engineering technique. Under the stated principles, the bond graph method has been successfully applied to aerospace systems and demonstrated as a unified modelling framework across many disciplines. It is clear that this framework will provide a powerful means of exploring the complexities and trade-offs associated with the next generation of integrated systems.

The significant new contributions to the bond graph method are the provision of an information model, the explicit definition of superstructures (especially distributions) and the thorough overhaul of the basic concepts of ports and bonds. Component ports present interesting challenges and opportunities, the main ones being identified in port binding, through ports and vectored/structured ports. The original concept of a primitive bond is now extended to cover domain-specific primitive bonds and user-definable composite bonds (with embedded components). Composite bonds, in particular, offer many new ways in which to simplify big models by managing the level of detail visible in the bond graph at any given time. Also, it will enable the introduction of fault conditions into big system models by embedding fault switches that ordinarily would be hidden from view.

It is recognised that systems integration is a relatively new field of interest without a mature body of academic literature or reported research, especially in the sense defined in this thesis. Also, there appears to be no open literature on the modelling of complete air vehicles *plus* their embedded vehicle systems which deals with issues of integrated dynamics and control. For these reasons, and taking stock of the work presented in this thesis, it is believed that this is the first attempt to apply the bond graph method to complex aerospace systems in any meaningful sense.

Given that the practical application to integrated systems is new (rather than merely a 'bond graph approach' per se), the modelling method has had to evolve significantly in order to address the human aspects of large-scale, diverse and complex models. It has also been recognised that bond graphs offer a very compact and mathematically pre-packaged approach to building models, so much so that it can be conjectured that they provide much clearer high-level representations of system power flow than any other notation. In fact, it is concluded that conventional approaches (such as signal flow block diagrams) may be extremely time-consuming and may obscure the physical significance of system interaction.

A major consideration in this work has been to discuss modelling not simulation. This has been deliberately and rigorously applied in order to make a stand against the 'modelling is just programming' lobby that exists across a large part of the engineering industry and the engineering software tool market. One significant and perhaps startling observation is that a lot of current bond graph tools attempt to bury causality, arguably the singlemost important modelling issue, hiding it from the modeller because allegedly it is of no great interest! Note that other simulation tools are genuinely programming tools. The work presented in this thesis has the benefit of real application on major aircraft projects and, as such, these conclusions carry authority even though, for commercial reasons, the rationale cannot be revealed here.

What has been attempted here is the construction of an original piece of applied engineering, one which can be of long-standing value to researchers and practitioners alike. It draws on a rich vein of experience in many areas of aerospace engineering and mathematical modelling and, as with all bond graph things, it owes a debt of gratitude to the insight of Hank Paynter. Although this thesis deals explicitly with an aerospace context, the underlying principles are wholly generic and could offer a great opportunity in other areas of systems integration.

6.3 Recommendations

Stemming from the activities and findings associated with this project, a number of different challenges have become apparent, any or all of which stand as valid and valuable topics for future research and development. These have been organised under five headings, namely Modelling, Analysis, Control Design, Design Data Management and Information Modelling.

6.3.1 Modelling

With regard to the scope of modelling within this thesis, there is an obvious need to extend what been achieved in order to cover Air Vehicle Integration, as shown in Figure 5.1. This means creating a systematic approach to aerodynamic and airframe (or structural dynamic) modelling, along the lines follows in Chapter 4. In principle, this should not present any new theoretical challenges but the modelling of this type for aeroelasticity and aeroservoelasticity is a specialised topic and is known to difficult. It is believed that bond graphs may provide a useful method for intermediate-level modelling (between rigid-body and finite-element representations) in order to study structural networks.

As a general topic of interest, there is a role for bond graph models in the investigation of nonlinear dynamics because they work with assemblages of discrete components which, collectively, reproduce a network of physical behaviours and interactions. Much of what is nonlinear has to be linearised about steady-state operation points (*i.e.* equilibrium conditions) before it is amenable to analysis and, moreover, the linearisation is only valid for small excursions or perturbations away from equilibrium. New opportunities have arisen in velocity-based linearisation [Leith & Leithead 1998] which effectively differentiate the state equations in order to produce an affine set of linear models which span the entire operational envelope of a system, *i.e.* both equilibrium and *non*-equilibrium points. It would be an attractive and useful facility of modelling tools to be able to derive this form of representation directly from a bond graph.

There is now considerable computational power available at low cost which can support symbolic algebra and its application to modelling. Indeed, bond graph tools are built on the paradigm of equation-based modelling and this requires symbolic manipulation to build model representations for analysis and simulations. **There is a need to establish the extent to which it is practicable to identify parametric trends symbolically from a bond graph model.**

For a large model, the information content would be explode: for a small model, it would be trivial. The challenge is to determine the level at which necessary and sufficient information can be supplied without outstripping the cognitive capacity of human observers. A uniquely important trend is *linearity* and, as such, there is a strong case for developing tools for 'searching' a nonlinear model in order to determine what density of operating points is actually required for system linearisation.

6.3.2 Control Design

Controllability is a major challenge for large integrated systems, mainly because the nature of dynamic interaction can be obscured by the convoluted nature of power flow. There is certainly a case for seeking a physical interpretation of inverse dynamics [Gawthrop 1998] but this can only be indicative. It appears to be most effective in cases where there is a parametric dependency between components which, in the inverse model, reveals itself as a positive positive feedback signal.

The generalisation of classical control concepts for multi-input multi-output systems, as exemplified by the Individual Channel Analysis and Design (ICAD) framework [O'Reilly & Leithead 1991], renders a thorough analytical assessment of dynamic interaction and accommodates inverse dynamics (especially right-half plane zeros) in the determination of multivariable structure. However, this is the result of a numerical recipe and, as such, the origin of dynamic peculiarities may not be immediately obvious.

It is recommended that a concerted effort be employed to establish the inter-relationship between bond graph modelling and individual channel analysis [cf. Gawthrop et al. 1989]. The interpretation and linearisation of nonlinear *input-output responses* is well-understood but the interpretation and linearisation of a nonlinear multivariable structure function, embedded within (2.8), is certainly not obvious. Alongside this idea is the need to adopt a systematic approach to uncertainty modelling, as represented by templates of relative uncertainty within ICAD [Leithead et al. 1997]. This requires an interrogation of the bond graph model in order to determine ranges of phase and gain variation across a given bandwidth. It also raises questions regarding the validity of augmenting linear models with structured uncertainty in order to synthesis so-called 'robust' controllers [*e.g.* Doyle & Stein 1981]. Accounting for uncertainty directly in a nonlinear model means that the linearisation itself contribute significant uncertainty in its own right and should be handled as if it matters, rather than just being ignored as at present.

Other areas of potential development are the validation of performance specifications and the systematic separation of fast and slow dynamics in complex models. The first of these has been illustrated [Ngwompo & Gawthrop 1984] using a simple electrical circuit and a two-link robot manipulation. A much greater and "engineeringly" relevant challenge would be the control of a closed-loop environmental control system of the type shown discussed in Section 5.4. **It is strongly recommended that this approach to inverse system simulation be thoroughly investigated for interactive thermofluid systems.** This class of system proved to be, far and away, the most awkward to model during this project and it become obvious that a control design specification would be very difficult given the potential number of thermal interactions ..... and because thermal effects and fluid flow effects are inter-dependent.

The remaining topic is relatively minor but of interest for partitioning models into their dynamically significant components. The issue is model-order reduction of multi-time scale systems [Dauphin-Tanguy & Borne 1985] and the separation of dynamic effects within given bandwidths. This may be helpful in extending the insights offered by ICAD to determine the most appropriate input-output pairings and to establish the natural hierarchy of feedback loops from low to high frequency in very complicated dynamic systems. **It is recommended that a method for deriving models of reciprocal systems be investigated for its applicability to aerospace systems.**

6.3.3 Design Data Management

Very little work has been developed in this project to underpin mathematical modelling of aerospace systems by a formal framework of design data management. This appears to be a critical requirement if large-scale tasks, such as building a virtual aircraft, is to become a reality in the system development context of major companies. Section 2.5 raises the issue together with the associated issues regarding traceability of design activities; clearly, this must incorporate modelling of systems as well as development of systems, if basic design integrity is to be assured. **Therefore, it is strongly recommended that work be conducted to formalise a modelling process and to represent both the process and the products of that process within an information model.**

Furthermore, a means is required for supporting independent assessments and computer-based model verification. **To this end, it is recommended that a text-based metamodelling language should be created for bond graphs and other graphical methods (e.g. Petri Nets, Signal Flow).** This should be allow all graphical constructs to be expressible in an equivalent scripted form and for indendepent syntactic and semantic analysis to be performed at 'code' level. With this facility, standard practices of software quality assurance, such as code walk-throughs, could usefully be applied to the subject of mathematical modelling ..... which currently has no internationally-recognised formal standards.

As a final observation, the scope of aerospace systems integration is likely to all-embracing as the drive towards performance optimisation continues. The scope of modelling must likewise become widespread and, while this thesis lays claim to a unified modelling framework based on bond graphs, the other aforementioned graphical notations need to be given due consideration. **It is recommended that the harmonisation of diverse notations is investigated with the intent of eventual providing a unified set of notations that can support hybrid functional models and their interfaces with non-functional models (e.g. reliability, geometry).**

6.4 And Finally .....

To recall the first paragraph of this thesis, "*Systems Integration* is widely accepted as the basis for improving the efficiency and performance of many engineering products. The aim is to build an unified system which optimises the use of its subsystem components: it is *not* to build subsystems which satisfy local objectives and then attempt to combine them in some *ad hoc* manner. This moves the philosophy of engineering away from traditional design boundaries and, in so doing, enables a structured evolution from an integrated system concept to an integrated system product." Bond graphs offer one approach to this problem, one that handles power flow and functional behaviour in a very effective way. There are many other equally valid approaches that have much to contribute. Thus, although the bond graph story has advanced as a result of this project, there are other modelling problems in other areas and the search goes on!

Thank you for your interest.

References

Appleyard GM, *1987*, Propulsion/Aerodynamic Integration in ASTOVL Combat Aircraft, Paper AIAA-87-2333, International Powered Lift Conference, Santa Clara, CA.

Babister AW, *1980*, Aircraft Dynamic Stability and Response, Pergamon Press: Oxford.

Bennett BS, *1995*, Simulation Fundamentals, Prentice Hall: London.

Bode HW, *1945*, Network Analysis and Feedback Amplifier Design, Van Norstrand: Princeton, NJ.

Breedveld PC, *1984a*, A Bond Graph Algorithm to Determine the Equilibrium State of a System, J. Franklin Institute.

Breedveld PC, *1984b*, Physical Systems Theory in Terms of Bond Graphs, PhD Thesis, Universitiet Enschede, Netherlands.

Bristol EH, *1966*, On a new measure of interaction for multivariable process control, IEEE Trans. Automatic Control, AC-11(1), 133-134.

Bristol EH, *1967*, A philosophy for single-loop controllers in a multi-loop world, ISA Chemical and Petroleum Division, CHEMPID Symposium, St. Louis, Missouri.

British Standards Institution (BSI), *1983*, Handbook 22: 1983 'Quality Assurance'.

British Standards Institution (BSI), *1984*, Tables relating to altitudes, airspeed and Mach numbers for use in aeronautical instrument design and calibration, British Standard 2G199.

Brumbaugh RW, *1991*, An Aircraft Model for the AIAA Controls Design Challenge, Paper AIAA-91-2631, Guidance and Control Conference, New Orleans, LA.

Buccholz JJ, Bauschat J-M, Hahn K-U, Pausder HJ, *1995*, ATTAS & ATTHeS In-Flight Simulators: Recent Application Experiences and Future Programs, AGARD CP-577, Flight Vehicle Integration Panel Symposium, "Flight Simulation - Where are the Challenges?", Braunschweig, Germany.

Burken JJ, *1992*, Flight Determined Stability Analysis of Multi-Input Multi-Output Control Systems, NASA Technical Memorandum 4416.

Cellier FE, *1991*, Continuous System Modelling, Springer Verlag: Berlin.

Chandraeskaran B, Goel AK, Iwasaki Y, *1993*, Functional Representation as Design Rationale, IEEE Computer, January 1993, 48-56.

Chen PP, *1976*, The Entity-Relationship Model - Toward a unified view of data, ACM Trans. Database Systems, 1(1), 9-36.

Cohen H, Rogers GFC, Saravanamuttoo HIH, *1972*, Gas Turbine Theory, Longman: London, 2nd Edition.

Dauphin-Tanguy G, Borne P, *1985*, Order Reduction of Multi-time Scale Systems Using Bond Graphs, the Reciprocal System and the Singular Perturbation Method, J. Franklin Institute, 319(1/2), 157-171.

de Vries TJA, *1994*, Conceptual design of controlled electro-mechanical systems: a modeling perspective, PhD Thesis, Universiteit Enschede, Netherlands.

DEF STAN 00-56, *1995*, Safety Management Requirements for Defence Systems Containing Programmable Electronics. Part1: Requirements; Part 2: General Application Guidance.

Doyle JC, Stein G, *1981*, Multivariable Feedback Design: Concepts for a Classical/Modern Synthesis, IEEE Trans. Automatic Control, AC-26(1), 4-16.

Electrical Industries Association, *1998*, CDIF Integrated Meta-model - Computer-Aided Control Systems Design Subject Area, Draft V7.

Electrical Industries Association, EIA Standard 632, *1997*, Processes for Engineering a System, Draft V0.9.

Elmqvist H et alia, *1997*, Modelica™ - A Unified Object-Oriented Language for Physical Systems Modeling.

Etkin B, *1972*, Dynamics of Atmospheric Flight, John Wiley:New York.

Garg S, *1993*, Partitioning of Centralised Integrated Flight/Propulsion Control Design for Decentralised Implementation, IEEE Trans. Control Systems Technology, 1(2), 93-100.

Garg S, Mattern DL, Bullard RE, *1991*, Integrated Flight/Propulsion Control System Design based on a Centralised Approach, J. Guidance, 14(1), 107-116.

Gawthrop PJ, *1991*, Bond Graphs: A Representation for Mechatronic Systems, Mechatronics, 1(2), 127-156.

Gawthrop PJ, *1998*, Physical Interpretation of Inverse Dynamics using Bond Graphs, The Bond Graph Digest, 2(1), http://www.ece.arizone.edu/~cellier/bg_digest.html

Gawthrop PJ, Ballance DJ, *1998*, Symbolic Computation for Manipulation of Hierarchical Bond Graphs, to appear in Munro N (ed), Symbolic Methods in Control System Analysis and Design, IEE Books.

Gawthrop PJ, Ballance DJ, Dauphin-Tanguy G, *1998*, Controllability Indicators from Bond Graphs, *submitted to the International Conference on Bond Graph Modelling, ICBGM'99.*

Gawthrop PJ, Smith L, *1992*, Causal Augmentation of Bond Graphs, J. Franklin Institute, 329(2), 291-303.

Gawthrop PJ, Smith L, *1996*, Metamodelling: Bond graphs and dynamic systems, Prentice Hall: London.

Genrich HJ, Lautenbach K, *1983*, Application and Theory of Petri Nets, Springer-Verlag: Berlin.

Gotel OCZ, Finkelstein ACW, *1994*, An Analysis of the Requirements Traceability Problem, Proceedings, 1st IEEE International Conference on Requirements Engineering, Colorado Springs, CO, 94-101.

Green WL, *1985*, Aircraft Hydraulic Systems, John Wiley: Chichester.

Grubel G, *1997a*, Another View of the Design Challenge Achievements, in Magni et al., *op.cit.*

Grubel G, *1997b*, CACSD-Frame for Robust Low-Complexity Control Law Design, Proceedings, 7th IFAC Symposium on Computer-Aided Control System Design, Gent, Belgium.

Harpur NF, *1953*, Some Considerations of Hydraulic Servos of the Jack Type, Proceedings, IMechE Conference on Hydraulic Servomechanisms, 41-50.

Horowitz I, *1979*, Quantitative synthesis of uncertain multiple-input multiple-output feedback systems, Int. J. Control, 30(1), 81-106.

Horowitz I, *1982*, Improved design technique for uncertain multiple-input multiple-output feedback systems, Int. J. Control, 36(6), 977-988.

Institution of Electrical Engineers (IEE), *1991*, Colloquium on Tools and Techniques for Maintaining Traceability during Design, Digest No. 1991/180.

Ismail IH, Bhinder FS, Simulation of Aircraft Gas Turbines, J. Engineering for Gas Turnines and Power, 113, 95-99.

Kailath T, *1980*, Linear Systems, Prentice-Hall.

Karnopp DC, *1969*, Power-conserving Transformations: Physical Interpretations and Applications using Bond Graphs, J. Franklin Institute, 288(3), 175-201.

Karnopp DC, *1978*, Pseudo Bond Graphs for Thermal Energy Transport, ASME J. Dynamic Systems, Measurement and Control, 100, 165-169.

Karnopp DC, *1979*, State Variables and Pseudo Bond Graphs for Compressible Thermo-Fluid Systems, ASME J. Dynamics, Systems and Control.

Karnopp DC, Margolis DL, Rosenberg RC, *1990*, System Dynamics: A Unified Approach, John Wiley: New York, 2nd Edition.

Keel LH, Bhattacharyya SP, *1997*, Robust, Fragile or Optimal?, IEEE Trans. Automatic Control, AC-42(8), 1098-1105.

Klein M, *1993*, Capturing Design Rationale in Concurrent Engineering Teams, IEEE Computer, January 1993, 39-47.

Kronlof K (ed.), *1993*, Method Integration: concepts and case studies, John Wiley: London.

Kwakernaak H, Sivan R, *1972*, Linear Optimal Control Systems, John Wiley.

Laan DJ, Shahroudi KE, Spee JBRM, *1997*, MDO: Closing the Gap between Systems Engineering and Product Engineering, Proceedings 1st Joint EAS/INCOSE Symposium on 'Systems Engineering -- The Future', European Space Agency, Netherlands.

Leith DJ, Leithead WE, *1998*, Towards a Theory of Local Model Networks and Blended Multiple Model Systems, Proceedings, UKACC International Conference on CONTROL '98, Swansea, UK.

Leithead WE, O'Reilly J, *1991*, Performance Issues in the individual channel design of 2-input 2-output systems: Part 1. Structural issues, Int. J. Control, 54(1), 47-82.

Leithead WE, O'Reilly J, *1992a*, Performance Issues in the individual channel design of 2-input 2-output systems: Part 2. Robustness issues, Int. J. Control, 55(1), 3-47.

Leithead WE, O'Reilly J, *1992b*, Performance Issues in the individual channel design of 2-input 2-output systems: Part 3. Non-diagonal control and related issues, Int. J. Control, 55(2), 265-312.

Leithead WE, O'Reilly J, *1992c*, M-input M-output feedback control by individual channel design: Part 1. Structural issues, Int. J. Control, 55(6), 1347-1397.

Leithead WE, O'Reilly J, *1994*, Investigation of the ICD structure of systems defined by state-space models, Int. J. Control, 60(1), 71-89.

Leithead WE, Robertson SS, O'Reilly J, *1997*, Relationship of QFT to ICAD, Proceedings, Symposium on Quantitative Feedback Theory and Other Frequency Domain Methods and Applications, Glasgow, UK.

Lorenz F, Wolper J, *1985*, Assigning Causality in the Case of Algebraic Loops, J. Franklin Institute, 319(1/2), 237-241.

Luenberger DG, *1977*, Dynamic Equations in Descriptor Form, IEEE Trans. Autom. Control, AC-22, 312.

McRuer D, Ashkenas I, Graham D, *1973*, Aircraft Dynamics and Automatic Control, Princeton University Press: Princeton, NJ.

McCloy D, Martin HR, *1980*, Control of Fluid Power, Ellis-Horwood: Chichester, 2nd Edition.

MacFarlane AGJ, *1970*, Dynamical System Models, GG Harrap: London.

MacFarlane AGJ, Karcanias N, *1976*, Poles and zeros of linear multivariable systems: a survey of algebraic, geometric and complex variable theory, Int. J. Control, 24, 33-74.

MacLean A, Young R, Bellotti VME, Moran T, *1991*, Questions, options and criteria: Elements of design space analysis, Human Computer Interaction, 201-250.

Maccallum NRL, *1973*, Effect of 'Bulk' Heat Transfer in Aircraft Gas Turbines on Compressor Surge Margins, IMechE Paper C36/73.

Maccallum NRL, *1976*, Models for the Representation of Turbomachinery Blades During Temperature Transients, ASME Gas Turbine and Fluids Engineering Conference, New Orleans, LA.

Maccallum NRL, *1978*, Thermal Influences in Gas Turbine Transients - Effects of Changes in Compressor Characteristics, ASME Gas Turbine Conference & Exhibition, San Diego, CA.

Maccallum NRL, Pilidis P, *1985*, The Prediction of Surge Margins During Gas Turbine Transients, ASME Gas Turbine Conference & Exhibition, Houston, TX.

Maciejowski JM, *1989*, Multivariable Feedback Design, Addison-Wesley: Reading, MA.

Magni JF, Bennani S, Terlouw J (eds.), *1997*, Robust Flight Control: A Design Challenge, Springer-Verlag: London.

Mattson SE, *1989*, On Modelling of Differential/Algebraic Systems, Simulation, 24-32.

Mayne DQ, *1973*, The design of linear multivariable systems, Automatica, 9(2), 201-207.

Mayne DQ, *1979*, Sequential design of linear multivariable systems, Proc. IEE, 126(6), 568-572.

MIL-H-5606C, *1971*, Miltary Specification: Hydarulic Fluid, Petroleum Base: Aircraft, Missiles and Ordnanace, US Department of Defense.

Murray-Smith DJ, *1995*, Advances in simulation model validation: theory, software and applications, Proceedings EUROSIM '95, 75-84, Elsevier.

Nesline FW, Zarchan P, *1984*, Why modern controllers can go unstable in practice, J. Guidance, 7(4), 495-500.

Ngwompo RF, Gawthrop PJ, *1998*, Bond Graph Based Simulation of Inverse Systems Using Physical Performance Specification, University of Glasgow Report CSC-98004.

Nyquist H, *1932*, Regeneration Theory, Bell Systems Technical Journal, 11, 126-147.

Office of Nuclear Regulatory Research, US Nuclear Regulatory Commission, *1981*, NUREG-0492: Fault Tree Handbook.

O'Reilly J, Leithead WE, *1991*, Multivariable control by 'individual channel design', Int. J. Control, 54(1), 1-46.

O'Reilly J, Leithead WE, *1995*, Frequency-Domain Approaches to Multivariable Feedback Control System Design: An Assessment by Individual Channel Design for 2-Input 2-Output Systems, Control Theory and Advanced Technology, 10(4), Part 5, 1913-1940.

Pallett EHJ, *1979*, Aircraft Electrical Systems, Longman: Harlow.

Paynter KM, *1961*, Analysis and Design of Engineering Systems, MIT Press:Boston, MA.

Peterson JL, *1981*, Petri Net Theory and the Modelling of Systems, Prentice-Hall: Englewood-Cliffs, NJ.

Pohl K, Jacobs S, *1994*, Traceability between Cross-Functional Teams, 1st International Conference on Concurrent Engineering Research and Application, Pittsburg, PA.

Ramesh B, Powers T, Stubbs C, Edwards M, *1995*, Implementing Requirements Traceability: A Case Study, Proceedings, 2nd International Symposium on Requirements Engineering, York, UK, 89-95.

Reutenauer C, *1990*, The Mathematics of Petri Nets, Prentice-Hall: London.

Ridgely D, Sanda S, *1986*, Introduction to Robust Multivariable Control, Report ADWAL-TR-85-3102, Flight Dynamics Laboratory, Airforce Wright Aeronautical Labs., Dayton, OH.

Rock SM, Emami-Naeini A, Neighbors K, *1994*, Integrated Flight/Propulsion Control: Subsystem Specifications, J. Guidance, Control and Dynamics, 17(1), 201-208.

Rogers GFC, Mayhew YR, *1980*, Engineering Thermodynamics and Heat Transfer, Longman: London, 3rd Edition.

Rolfe JM, Staples KJ (eds.), *1986*, Flight Simulation, Cambridge University Press: Cambridge, UK.

Rosenbrock HH, *1969*, Design of multivariable control systems using the inverse Nyquist array, Proc. IEE, 116(11), 1929-1936.

Rosenbrock HH, *1974*, Computer-Aided Control System Design, Academic Press: London.

Rumbaugh J, Blaha M, Premerlain W, Eddy F, Lorensen W, *1991*, Object-oriented modeling and design, Prentice-Hall: Englewood-Cliffs, NJ.

Schanzer G, *1983*, Influence of Windshear on Flight Safety, in AGARD-CP-347 "Flight Mechanics and System Design Lessons from Operational Experience", Athens, Greece.

Schierman JD, Lovell TA, Schmidt DK, *1993*, A Comparative Study of Multivariable Robustness Analysis Methods Applied to Integrated Flight and Propulsion Control, Paper AIAA 93-3809, Proceedings AIAA Guidance, Control and Navigation Conference, Mounteray, CA.

Schmidt P, Garg S, Lorenzo C, *1992*, A Method of Paritioning Centralised Controllers, NASA Technical Memorandum 4276.

Schmidt P, Schierman J, Garg S, *1992*, Analysis of Airframe/Engine Interactions - An Integrated Control Perspective, Paper AIAA-90-1918, 26th Joint Propulsion Conference, Orlando, FL.

Smith PR, *1991*, Application of Eigenstructure Assignment to the Control of Powered-Lift Combat Aircraft, Defence Research Agency: RAE Bedford, Technical Memorandum FS 1009.

Sobieszczanski-Sobieski J, Haftka RT, *1994*, Multidisciplinary Aerospace Design Optimisation, Survey of Recent Developments, AIAA.

Society of Automative Engineers (SAE), S-18 Committee, *1995*, Document ARP 4761: 'Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment', Draft #14a, *dated 8 December 1995*.

Society for Computer Simulation (SCS), Technical Committee on Model Credibility, *1979*, Terminology for Model Credibility, Simulation 32, 103-104.

Stewart JF, Burcham FW, Gatlin DH, *1992*, Flight-Determined Benefits of Integrated Flight-Propulsion Control Systems, NASA TM 4393.

Stirling R, *1987*, Performance of a Gust Alleviation System in an Advanced Combat Aircraft, University of Bristol Report RS/2/87.

Strens MR, Dobson JE, *1994*, Responsibility Modelling as a Technique for Organisational Requirements Definition, Intelligent Systems Engineering, 3(1).

Symon KR, *1971*, Mechanics, Addison-Wesley, Reading, MA.

Thoma J, *1975*, Introduction to Bond Graphs and their Applications, Pergamon: Oxford.

Thoma J, *1990*, Simulation by Bond Graphs, Springer-Verlag: Berlin.

van Beek DA, Rooda JE, *1997*, Design of Discrete Controllers for Continuous Systems using Hybrid Chi, Proceedings, 7th IFAC Symposium on Computer-Aided Control System Design, Gent, Belgium, 9-14.

van Broenink JF, *1997*, Bond graph Modeling in Modelica, European Simulation Symposium, Passau, Germany.

van Dijk J, *1994*, On the Role of Bond Graph Causality in Modelling Mechatronic Systems, PhD Thesis, Universitiet Twente, Netherlands.

van den Hamer P, Lepoeter K, *1996*, Managing Design Data: The Five Dimensions of CAD Framework, Configuration Management and Producy Data Management, Proceedings IEEE, 84(1), 42-56.

Varsamidis T, *1998*, Object-Oriented Modelling for Computer-Aided Control Engineering, PhD Thesis, University of Wales, Bangor, UK.

Viersma TJ, *1980*, Analysis, Synthesis and Design of Hydraulic Servosystems and Pipelines, Elsevier: Amsterdam.

Wellstead PE, *1979*, Introduction to Physical System Modelling, Academic Press: London.

Woodfield AA, Woods JF, *1983*, Worldwide Experience of Wind Shear During 1981-1982, in AGARD CP-347 "Flight Mechanics and System Design Lessons from Operational Experience", Athens, Greece.

Yaniv O, Horowitz I, *1986*, A quantitative design method for MIMO linear feedback systems having uncertain plants, Int. J. Control, 43(2), 401-421.