N93-25970

An Autonomous Satellite Architecture Integrating Deliberative Reasoning and Behavioural Intelligence.

Craig A. Lindley

Department of Computer Science and Engineering
University of New South Wales, Kensington, NSW, Australia

ABSTRACT

This paper describes a method for the design of autonomous spacecraft, based upon behavioural approaches to intelligent robotics. First, a number of previous spacecraft automation projects are reviewed. A methodology for the design of autonomous spacecraft is then presented, drawing upon both the European Space Agency technological centre (ESTEC) automation and robotics methodology and the subsumption architecture for autonomous robots. A layered competency model for autonomous orbital spacecraft is proposed. A simple example of low level competencies and their interaction is presented in order to illustrate the methodology. Finally, the general principles adopted for the control hardware design of the AUSTRALIS-1 spacecraft are described. This system will provide an orbital experimental platform for spacecraft autonomy studies, supporting the exploration of different logical control models, different computational metaphors within the behavioural control framework, and different mappings from the logical control model to its physical implementation.

Keywords: Spacecraft Control, Space Robotics, Artificial Intelligence, Subsumption.

Introduction

AI applications in space systems are becoming more readily accepted, and constitute a key enabling technology for ambitious projects such as the Space Station Freedom and Space Exploration Initiative. Current or proposed constellations of unmanned spacecraft, particularly in low earth orbit, and multiple deep space missions with long telecommunication propagation delays, can also gain substantial benefits from the use of more autonomous spacecraft operation. Teleoperation of industrial space facilities and orbital experimental platforms using highly autonomous onboard systems may provide crucial competitive advantages in the commercial and industrial exploitation of space.

This paper presents an architecture for autonomous spacecraft control that supports the integration of behaviour-based approaches to emergent intelligence with numerical and computational simulation models, and symbolic reasoning systems such as expert and knowledge based systems. Firstly, a methodology is proposed for developing autonomous space systems. Using this methodology, system operational functions are hierarchically decomposed, but functional levels are not mapped directly onto computational models. The operational decomposition is used to refine specifications of layered competencies, based upon a generic layered competency model. Each competency level defines a virtual machine interface from the point of view of superordinate levels. Hence, the hierarchical decomposition of system functionality during operational analysis does not imply a strict corresponding hierarchical synthesis for design and implementation, but provides a framework for specifying system behaviours and resources, and for understanding their interactions.

The realisation or implementation of the functionality of successive virtual machines can be carried out using the most appropriate computational paradigm, or a rich combination of paradigms. From this point of view, a knowledge base or expert system can be regarded either as a convenient abstraction adopted during the design process to define the input/output behaviour of behavioural modules, or as a resource for use by behavioural modules much as human

operators would use expert systems for particular tasks.

A simplified example application of this approach is presented. The resulting satellite control architecture is significantly different from previous satellite designs, having improved robustness, decreased operating overheads, and more autonomous fault tolerance. Current plans are to validate and refine this approach in a rich simulation environment, and eventually to build and operate a satellite for ongoing in-orbit trials and experiments.

Precedents

Onboard autonomy is a matter of degree. Pidgeon et al (1992) describe how a number of current spacecraft have mechanisms for fault detection, reporting, and subsequent switching to component, subsystem, or system fail-safe modes. Human operators must then diagnose faults and initiate appropriate contingency recovery procedures. Increasing abstraction levels in spacecraft command languages have also been adopted. For example, in normal operations, the Hipparcos spacecraft is controlled by processed commands which are sent to the onboard computer for distribution to other systems, and for possible time tagging. Direct commands are also available, which bypass the onboard computer as a backup in the event of computer failure, and for more direct access to the controlled systems. Priority real time commands can also be issued, which allow direct switching of systems. The ERS-1 spacecraft, which is in a low polar orbit with limited ground access, has a similar command macro system, with four command types providing different functions and levels of authority. The lowest levels of commands bypass the onboard computer and data handling system, again ensuring control if those systems fail. The EURECA system, comprising fifteen separate payloads, uses an onboard Master Schedule which contains a list of time tagged command macros for execution by the onboard data handling system. Those commands include rudimentary failure routines, backed up by safe modes to deal with command loop failure. Again, direct telecommands can also be used, to bypass the data handling system and onboard computer in the event of their failure.

The degree of onboard autonomy is continuously increasing, and can be expected to incorporate a wide variety of techniques from AI and intelligent robotics research. A number of prototype systems have been built to investigate onboard expert system applications, including DIPOLE, SAGES, APS, SACV, and SICON (see below). Operational systems may include the Cassini Titan Probe, and many Space Station Freedom applications. Most of these systems involve onboard architectures comprising a number of distinct modular functions. Autonomous systems of increasing size and complexity tend to have distributed functionality, with the various functions running on separate physical processors. Another recurrent theme is the devolution of autonomous functions to the lowest possible abstraction levels.

Tello (1986) describes DIPOLE, a system for satellite control which is intended to integrate "shallow" heuristic or rule based reasoning with "deep" model-based reasoning. The shallow system uses fault-tolerant mathematical and algorithmic subroutines, and has the form of real-time expert systems with data-driven switches for controlling their performance. The aim is for the deep reasoning system to take over when the shallow system gets into difficulty, and to allow the shallow system to resume when the deep reasoning system has resolved the problem. The DIPOLE architecture addresses two particular problems for real time deliberative controllers. *Circumspection* is the problem of enumerating all implicit conditions and assumptions associated with given knowledge, and the ability to handle situations when these are no longer valid. *Inference thrashing* is the situation when inferencing cannot produce solutions quickly enough to keep up with changing circumstances. DIPOLE seeks to address these problems by using shallow rule based expert systems as reflex processors in real time, with longer deliberative processes performed by the deep reasoning system.

Ciarlo et al (1987) describe a spacecraft expert system prototype study conducted for the

European Space Agency. Some of the conclusions of the initial study include:

- highly simplified interfaces typical of spacecraft modular units reduce integration and control problems. However, this severely limits the information available for monitoring each unit, and the choice of actions available to correct failure, to the point of making the advantages of expert systems questionable when compared to standard algorithmic or table-driven software.
- it is difficult, and not necessarily advantageous, to use an expert system in a satellite designed without this in mind.

Ciarlo and Schilling (1988) report upon work following on from this initial study to consider an expert system embedded within the Cassini Titan probe, for autonomously managing the descent of the probe into Titan's atmosphere. The authors note that to keep the complexity and susceptibility of the system to faults as low as possible, the autonomous system should be implemented at the lowest possible level, with capabilities such as component and sensor self testing and redundancy switching. Scientific management, which involves adaptation to the situation according to complex rules, is regarded as an appropriate function for implementation as a knowledge base. Engineering management, involving FDIR and subsystem control, is regarded as an appropriate function for conventional technology.

The Satellite Autonomy Generic Expert System (SAGES) architecture, developed by Rockwell, is based upon the definition of four intelligent agents, corresponding to phases of the mission operation cycle, including planning, scheduling, execution, and analysis (providing feedback into the planning phase; Raslavicius et al, 1989). A SAGES prototype has been developed for a "typical" surveillance satellite.

The Boeing Aerospace Autonomous Power System (APS) testbed has been assembled for use in developing improved control techniques for aerospace electrical power systems (Spier and Liffring, 1989). The main emphasis of APS is the development of a programming environment to properly control the concurrent execution of multiple autonomous algorithms coupled with continuous input and output data flow. Expert system functions include fault diagnosis and recovery, and battery charge control. The expert systems use event-driven processing within a blackboard environment.

The European Space Agency (ESA) Standard Generic Approach to Spacecraft Autonomy and Automation (SGASAA), is a hierarchical model which aims to devolve decision-making to the lowest possible level (Pidgeon et al, 1992). To this end, it is a distributed onboard architecture, with each payload and subsystem having a certain degree of "intelligence", in addition to an Onboard Mission Manager (OBMM) responsible for the control of the spacecraft as a whole. Separate subsystem managers are intended to handle their own failures and report the results of their diagnoses via LAN to the OBMM, along with a proposed recovery action. The OBMM can authorise the proposed recovery, or block it if the failure is caused by a failure elsewhere. SACV (*ibid*) is an investigation of the SGASAA concept involving the implementation of a fully autonomous spacecraft based upon EURECA.

SICON, built by LISP Machine Inc, is a simplified prototype system for satellite intelligent control, concentrating upon the electrical power system (Leinweber, 1987). The SICON prototype deals with load distribution and switching, solar array orientation, power system verification and checkout, fault diagnosis, trend analysis, contingency management, battery charge and reconditioning-cycle optimisation, and fuel cell monitoring and control. Leinweber notes that there are a number of SSF processes and subsystems that are amenable to real-time process control, including the electrical power system, attitude and orbital control system, environmental and life support system, propulsion system, monitoring of docked vehicles, manufacturing process control, and ground communications and network control. Such real-time onboard applications require particular expert system features, including high-speed context-sensitive rule activation, efficient memory recycling, acceptance of

interactive commands without suspending execution, and communication between multiple expert systems in order to provide redundancy.

The Space Station Freedom (SSF), will require an extensive data processing support environment. The communications and information processing backbone of the SSF is the Data Management System (DMS). The DMS has the dual role of providing hardware resources and software services which support data processing and communications needs of the system, its elements, and payloads (Erickson, 1987). It also functions as an integrating entity, providing a common operating environment and human-machine interface for the operation and control of orbiting SSF systems and payloads by both the crew and ground operators. The DMS provides signal conditioning, and timing synchronisation of data required for interpreting time-critical information and results between expert systems, knowledge-based systems, and robotics elements distributed throughout the SSF environment. Woods (1992) notes that the DMS may use artificial intelligence techniques for fault detection, isolation, and recovery (FDIR) on DMS components. An Integrated Systems Executive (ISE) will provide overall software scheduling and control for all other systems, experiments, and elements. Low level software modules will take care of time critical control loops and fault recognition. Development projects are currently underway in a number of SSF expert system applications.

A Methodology for Autonomous Spacecraft Development

In the ongoing development of autonomous spacecraft, devolution of decision-making to the lowest possible level, and the modularisation and distribution of functionality, are prominent trends. These trends, driven by the particular requirements of real time autonomous agency, have been most fully developed in the context of mobile robotics research, and are captured most strongly by behavioural approaches to autonomous systems design. Behavioural approaches to mobile robotics have also demonstrated more fundamental benefits in addressing the problems of circumspection and inference thrashing that have plagued deliberative robot control systems. There is therefore considerable potential for behavioural approaches to contribute to the increased automation of space systems. Toward this end, this paper proposes a methodology for autonomous spacecraft development which draws from both behavioural approaches to mobile robotics and an ESTEC (European Space Agency's technological centre) methodology for space automation and robotics. While behaviour-based robots have been suggested for planetary surface exploration (Brooks and Flynn, 1989), the behavioural paradigm has not previously been used to design orbital spacecraft. Hence, the proposed methodology is not fully articulated, but groundwork for a more complete approach is presented. Indeed, it is probably dangerous to suggest any comprehensive standardised approach until behaviour based spacecraft have been well demonstrated, and the paradigm and its benefits in this domain are well understood.

The main technical objectives of a design methodology include the achievement of full bidirectional traceability between user requirements and system solutions selected, the breakdown of complex problems into successively simpler ones, unity of system architecture, rigorous interfaces between subsystems, improved communications with end users, precise communication within a development team, efficient parallel development of subsystems, reduced control complexity, greater simplicity of design, and sound data analysis and administration (Elfving and Kirchoff, 1991). The resulting benefits include a high-quality product which is easy to maintain and upgrade, better project control during the design process, reduced time to completion, and lower cost for system development.

Elfving and Kirchoff (1991) present logical reference models which postulate the essential functions of an automation and robotics system, to be used for structuring user requirements and transforming them into distinct design solutions. This ESTEC methodology is derived from structuring

principles of hierarchical decomposition and hierarchical structuring, fundamental properties from disciplines such as control theory and mechanical engineering, and generic principles of structured analysis and structured design. It is characterised by a clear separation between operational analysis and system synthesis, and the application of the principle of abstraction in the form of reference model techniques. While the ESTEC methodology has many desirable features, it does *not* immediately lend itself to the synthesis of behaviour-based autonomous systems. Behavioural approaches are, however, in need of methodological guidelines to support a systematic composition of behaviours into competencies, to assist in ensuring that the resulting system design will meet its overall requirements for a given application (Brooks, 1990, 1991). It is therefore valuable to draw from both the behavioural approach and the ESTEC methodology, in order to achieve a methodology combining the benefits provided by both.

Essential Characteristics of the Subsumption Methodology

The subsumption architecture was specifically developed to address requirements for autonomous mobile robots (see Brooks, 1986, 1990, and 1991) including multiple, often conflicting, goals, multiple sensors, robustness, and extensibility. The approach is based upon a number of principles which can be drawn upon and adapted here as principles which bear upon the spacecraft autonomy problem:

1. complex or "intelligent" behaviour can be an emergent phenomenon, arising from the interactions of a spacecraft with its environment and users.

2. component interfaces should be simpler than the components that they interconnect.

3. if a module solves an unstable or ill-conditioned problem, then it is probably not a robust solution.

4. autonomous model-making is important, since idealised models may be inaccurate.

5. the spacecraft must operate in a three-dimensional world.

6. relational models can avoid the cumulative errors that characterise absolute coordinate systems.

7. there is no global internal model, or global planning activity with a hierarchical task structure.

8. for robustness, the spacecraft must be able to perform when one or more of its sensors fails or malfunctions. Recovery should be rapid, so built-in self-calibration is required at all times.

9. the spacecraft control problem is decomposed in terms of layers or levels of competency. Those levels are task-achieving behaviours, and as such are external manifestations of the control system. "Higher" levels correspond with more specific classes of behaviour.

10. each successive level subsumes as a subset each earlier level, and provides additional constraints on the class of valid behaviours defined by the earlier levels.

11. successive levels *extend* competency, but do not alter the structure of the implementation of lower levels. A level can receive data from lower levels in order to monitor their behaviour, and can output data to lower levels in order to modify behaviours by inhibiting or exciting them.

12. competencies are parallel processes. Hence, lower level competencies can ensure that spacecraft behaviour is sensible, while higher level competencies take time to produce more optimal control solutions.

13. there is no central locus of control, either for the system as a whole, or within any particular competency layer. This is essential for robustness.

14. each layer can run on its own processor, and individual layers can also be run over many loosely coupled processors.

15. within each particular layer, a traditional decomposition of functionality may be used "to some extent".

These principles conform with the paradigm for autonomous systems design which Maes (1990a) refers to as the *behavioural* approach. Inspired by biological models of autonomy, the behavioural approach has abandoned the older *sense-model-plan-act* (SMPA) paradigm (Brooks, 1990), and in so doing has achieved many successful demonstrations of greatly improved robot performance and robustness. Those demonstrations represent explorations within the new paradigm, but can by no means be regarded as definitive or fully matured.

It is important to distinguish the logical design of an autonomous control system from its implementation. Some explorations of the behavioural paradigm have concentrated upon software design, with the software being compiled to run on a single embedded processor (eg. the MIT Squirt robot, Brooks 1990). However, while such systems can demonstrate the effectiveness of a control architecture based upon situatedness and behavioural interaction, rather than model-based deliberative reasoning, the use of a single physical processing element creates a single physical locus of control, and therefore a single point failure mode in the resulting robot. The behavioural paradigm is a *systems* paradigm. As such it should lead to a distributed hardware functionality of the kind that typifies the more sophisticated behavioural robots. Research within the new paradigm continues at a vigorous pace, and the question of how competencies at a conceptual level can be achieved as emergent properties of increasingly parallel and distributed computational processes at the implementation level has only just begun to be investigated. Continuing advances in parallel and distributed hardware architectures reinforce the viability of the behavioural approach, and demand a radical rethinking of how autonomous systems can be structured.

A number of researchers have adopted the behavioural approach as a method for designing the lower level control functions within an autonomous system, and have then provided one or more "layers" above the behavioural layers which perform higher level deliberative and symbolic computations. For example, Steels (1991) proposes a frame based system for "high level" cognitive tasks (such as language use), in which frames are grounded by sensory inputs and influence the behaviour of the system in proportion to their "fit" to the current situation. Arkin (1990) describes the Autonomous Robot Architecture (AuRA), a framework for experimenting with the integration of behavioural approaches with model-based reasoning. AuRA allows the advantages of modularity, incremental design, adaptability, and robustness of the behavioural approach to be supplemented by the use of model-based knowledge to configure behavioural strategies in an efficient form. The AuRA architecture comprises five basic subsystems: Perception, Cartographic, Planning (both a hierarchical planner and a distributed reactive plan execution subsystem), Motor (the actuator set interface), and Homeostatic control (monitors internal conditions of the robot for both the higher level planning mechanisms and the motor schemas). Flexibility is incorporated into the AuRA system by drawing modularised behavioural patterns and sensory strategies from a library and configuring them to meet the needs of a particular mission and any known environmental constraints. World models play an important role in this configuration process. Bonasso (1991) also describes an architecture in which a declarative model is used for reasoning about plans and controlling the activation of behaviours implemented within an underlying subsumption layer. Malcolm and Smithers (1990) describe SOMASS, a robot assembly system that combines a PROLOG assembly planning subsystem with a plan execution subsystem that handles uncertainty by means of behavioural modules which accomplish useful motions of assembly parts. Malcolm and Smithers note that, although the "cognitive" planning function was implemented in a high-level symbolic language, and the "subcognitive" execution agent in a low level language, these decisions were motivated by convenience. Also, the cognitive/subcognitive distinction was itself found to be a useful construct for the SOMASS system, but is not a necessary or convenient construct for artificial mentality in

general. The interface between the cognitive and subcognitive systems presents a virtual machine model to the cognitive system, which heavily determines and permeates the design of the cognitive system. The need to present a virtual machine interface to the cognitive system results in a modularity of behaviours in the subcognitive layer, which is not needed in approaches (eg. subsumption) which do not include a cognitive component. Gat (1991) describes ATLANTIS, a system which integrates behaviours and deliberative processes in a three layered structure. The first layer comprises a behaviour-based system for robust motion control and low-level competencies. A deliberative layer provides high level reasoning, such as plan generation. These layers are joined together by a sequencing layer based upon Reactive Action Packages.

These different explorations of the relationship between deliberative reasoning and behavioural autonomy are not driven by any well-proven limitations of the behavioural approach. As Brooks (1990, 1991) notes, the question of how far a behavioural approach without the use of deliberative computations can go in achieving higher levels of competence in autonomous agents is one which must be addressed by ongoing empirical investigations. However, an issue arises within the behavioural paradigm regarding the range of computational metaphors that can usefully be adopted for designing behavioural units. In the subsumption architecture, Brooks uses augmented finite state machine (AFSM) models to define primitive behavioural elements, and AFSMs are further grouped into more complete behaviours. However, any number of computational metaphors can potentially be used to describe a system with a given transfer function. The effectiveness of AFSMs has been demonstrated for some behaviours, but metaphors of multiple interacting agents (see Grant, 1992), objects, processes, production systems, etc. may equally provide convenient metaphors. Similarly, knowledge-base systems, rule bases (AFSMs are defined by rule sets in Brooks' behaviour language), and expert systems metaphors may be convenient. The metaphor adopted should be that which most "naturally" describes the behaviour of a module from the perspective of the system designer. Adopting the behavioural paradigm for the overall control system architecture does not rule out the adoption of other metaphors for structuring and designing the computational processes within a given behaviour or module in order to achieve a desired set of input/output mappings. The danger in adopting heterogeneous metaphors within a behavioural control system is if any metaphor distorts the behavioural framework by encouraging the centralisation of behavioural coordination and control, or the centralisation of data flow. It can be argued that this should not occur in the case of spacecraft control if the control system is modelled upon manual spacecraft operation, since ground based, manual spacecraft control involves highly distributed, cooperative decision-making by numerous "agents" of mixed expertise, generally in a way that is highly redundant, robust, and adaptive. The behavioural paradigm is not violated by experimentation with alternate metaphors for developing the *internal* structure of behavioural modules *within* a behavioural control framework.

Towards Reference Models for the Behavioural Methodology

The reference models described by Elfving and Kirchoff are intended to provide clear traceability from user requirements to design solutions, which justifies technical decisions made and avoids excessively flexible, expensive, and complex systems with high technical risk. The reference models are intended to cover robots, mobile vehicles, and process control, and reflect the operational use of the system. Elfving and Kirchoff divide A&R capabilities in space into three major fields:

- external servicing of payloads
- servicing of scientific experiments within pressurised orbiting laboratories
- surface mobility and sample acquisition for planetary exploration

To this can be added the field of current concern:

- autonomous orbital spacecraft control

The starting point for this ESTEC automation and robotics (A&R) control design methodology is an established conceptual layout for the A&R system, typically available as a result of mission and system definition studies. The logical reference models capture the essential principles of this methodology.

The objective of *Operational Analysis* (or *Task Analysis*) is to derive the essential abilities of an A&R system such that mission objectives are sure to be fulfilled. This is done by a step-by-step decomposition into levels of equal importance, from mission objectives down to a level of elementary actions. This defines "what has to be done", ideally without anticipating any solution, but by using initial knowledge about system functional layout based upon initial mission and system definition studies. This analysis phase requires system operational expertise. *System Synthesis* involves the definition of solutions that satisfy the different operational features required at the various levels of decomposition, thereby addressing "how is it to be done?". System Synthesis involves an aggregation process, from elementary abilities to system capabilities. Synthesis requires A&R technology expertise.

Given these processes of analysis and synthesis, the key methodological question is that of how to achieve traceability between the two areas, assuming that the decomposition logic of analysis must be in agreement with the synthesis logic, so the inherent structure of the analysis process is a virtual system solution. This nexus is achieved by the use of the logical reference models. Three logical reference models are distinguished:

1. *Functional Reference Model (FRM)*: represents a decomposition of all functions and information flow and structures, and is valid for all A&R applications.

2. *Application Reference Model (ARM)*: represents an FRM derivative which focuses on individual classes of A&R applications.

3. *Operations Reference Model (ORM)* : represents an FRM derivative which focuses on models of operation and systematics for man/machine and preparation/utilisation allocation.

The objectives of applying reference model techniques are:

- guarantee a common 'thinking model' which is valid for all A&R applications, to enable and ease communication within and between development teams
- provides a generic system information structure and identifies essential functions for which application-specific design solutions should be found
- establishes "rules-of-thumb" and heuristic design strategies that allow the designer to systematically derive good and relevant solutions to common types of problems
- uses formal documentation techniques and graphic support tools that emphasise the hierarchical functions and information flow and structures of complex systems

A *logical model* represents the essentials of a system, assuming ideal internal technology of the system and excluding application-specific topics. A physical model represents the implementation of a specific application, and therefore needs to represent the actual constraints imposed by the chosen internal technology. Hence, only logical models can be reference models that meet the requirement of being valid for a multitude of applications and/or implementation technologies. However, a major goal of a design methodology is to maintain a strong and traceable link between the logical and physical models of a system in order to achieve design commonalities and open implementation architectures.

A Behaviour-Based Functional Reference Model

The FRM proposed by Elfving and Kirchoff involves a hierarchical ordering of functions and information with increasing precision and decreasing planning horizon from top to bottom, where the hierarchical information interface layers are:

1. A&R mission: the objectives of end users

2. task: clustered activities performed on a single subsystem defined as the element submitted to motion or actuation by the A&R system

3. action: elementary activity for a functional subsystem

This decomposition leads to hierarchical functional and information layers, with the controlled devices and processes at the bottom, and successive layers for action, task, and mission execution, planning and control. State information flows from each layer to the next high layer, while commands and activity attributes flow down through the structure.

This hierarchical decomposition, with "vertical" divisions between distinct objectives, tasks, and actions at each level, is not compatible with a behavioural methodology. To be compatible with a behavioural approach, the following modifications are required:

- instead of a decomposition of system functions in terms of a tripartite structure of mission, tasks, and actions, the system can be decomposed in terms of layered competencies. This overcomes the somewhat arbitrary parsing of activity into three levels, providing a more flexible layering and abstraction mechanism which subsumes and extends the tripartite structure. It also has the advantage of retaining the visibility of what the system is required to do, rather than decomposing system functions according to how those functions are implemented. System requirements can be mapped directly onto competencies, and competencies then onto their implementation.
- instead of placing all sensor and actuator interfaces at the lowest level of a hierarchical structure, each competency level is associated directly with a subset of the total set of sensors and actuators required, in addition to having interfaces to the competency levels above and below itself.

A&R Mission planning and control can be retained as the highest general level of system competency. The lowest level is also general, and comprises basic survival competencies. Intermediate levels are variable in number and form across different A&R application classes. The resulting vertical FRM structure is then as shown in figure 1.

Elfving and Kirchoff describe forward control, nominal feedback, and non-nominal feedback functions across all hierarchical levels. Forward control involves activity decomposition and execution planning, based on a priori knowledge, and plan execution. Nominal feedback deals with forseen refinements and updates to ensure that the system achieves the forward control execution goals. Non-nominal feedback covers the case of system performance diverging from the allowable region around the nominal execution plan and is realised by means of monitoring discrepancies between actual and allowable states, diagnosis of reasons for possible discrepancies, and generation of recovery strategies and constraints.

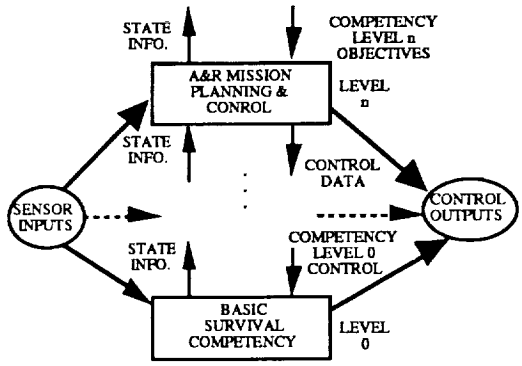

Figure 1. Behavioural FRM.

This model may be used to address individual competencies in the vertical decomposition of system functionality. However, nominal feedback, non-nominal feedback, and feedforward control functions should not be regarded with restricted computational paradigms in mind. It must be regarded as an open question how the respective functions can best be modelled for any particular application or level of problem abstraction. It is yet another question how the modelled functionality should then be implemented. For example, a logical model of planning may or may not be the best way of specifying a type of planning activity that may be required.

Moreover, if the planning activity is defined logically, the implementation of the planner may take quite a different form, such as that of a highly distributed and emergent function (eg. gradient field approaches, as described by Payton 1990, analogical planning, as described by Steels 1990, or emergent goal arbitration, as described by Maes 1990b). In a behavioural system, inputs and outputs for these functions include behaviour monitoring and control signals, in addition to sensor inputs and actuator outputs, depending upon their role within the competency in which they occur. In other words, nominal feedback, forward planning, and non-nominal feedback become *behaviours* within a competency layer.

These redefinitions result in a very different FRM, less detailed than that presented by Elfving and Kirchoff, and as such placing more importance upon Application Reference Models for the different classes of A&R applications.

Behavioural Application Reference Model

The Application Reference Model (ARM) tailors the functional blocks and the information structure of the FRM to the characteristics of each class of A&R application, in order to ease acceptance and understanding. Elfving and Kirchoff identify these classes as:

- motion systems with fixed linkages to the environment (eg. robot manipulators)
- mobile systems (eg. moving vehicles)
- continuous processes (eg. climate control)
- event/sequence control (eg. PLC equipment)

To this can be added the class of current concern (a subtype of mobile systems):

- autonomous orbital spacecraft

In physical realisations, a combination of these classes may be integrated to form the overall A&R control function.

The ARM leads to a more detailed decomposition than the FRM. From the behavioural viewpoint this amounts to identifying for each class of application:

- the competency layers required
- generic requirements for nominal feedback, non-nominal feedback, and forward control within each layer
- individual behaviours within each competency

Ongoing research is needed to identify alternate implementation strategies for various competencies, and to systematically analyse the tradeoffs between different strategies as a basis for rational design decisions.

Proposed ARM for Autonomous Orbital Spacecraft

Due to very significant domain differences, the competency layers proposed here for autonomous orbital spacecraft bear little resemblence (other than at the highest level) to those proposed by Brooks (1986) for biologically-inspired mobile robots operating in earth-gravity environments. Some of these differences arise due to the different operational environment and sensor and actuator sets of orbital spacecraft in comparison with mobile surface robots. Other differences arise due to a major aspect of spacecraft functionality which concerns the collection, distribution, and reception of data.

The definition of layered competencies is made in terms of decreasing criticality and increasing autonomy. In this sense, the approach represents an extension of automated safe mode transitions into a more complex set of behaviours, and a set of autonomous operations which function during normal spacecraft operation in addition to emergencies. Consideration of the layered structure shows the non-hierarchical nature of the control system. Mission critical survival decisions, often made at the lowest levels, must have priority over higher-level decision making during emergency situations. However, the higher levels must be able to modify resource allocations, timing relationships, and data flow within the lower levels in order to establish priorities between activities within a wide range of variation of nominal operating modes in order to achieve higher levels of competence in autonomously meeting and optimising mission objectives.

For example, critical docking operations cannot have their power supply interrupted. Ensuring that this does not occur can involve power conditioning schedules that come into operation well in advance of the docking activity, and therefore must be planned by higher competency levels, and those plans then conformed to or used by the behaviours of the lower levels.

A proposed competency model is shown in figure 2. The Maintain Thermal Balance competency is placed at level 0 as the most fundamental precondition of spacecraft survival. That is, any significant over- or under-heating can permanently damage or destroy the spacecraft. However, temperature variations tend to be gradual, and good thermal design can ensure that the normal range of temperature variations for the spacecraft is limited to a 20° range. The greatest danger is from more localised temperature fluctuations (especially accumulation of heat), possibly due to faults, which can be dealt with by temperature dependent inhibition or excitation of heat generating processes.

9	Plan and Execute Mission Based Upon Objectives
8	Rendezvous and Dock (| Land | Take Off)
7	Plan and Schedule Data Acquisition and Transmission
6	Plan and Execute AOCS Maneuvers
5	Acquire, Condition, Downlink Instrument Data
4	Maintain Attitude Stability and Orbit
3	Acquire, Condition, and Downlink Engineering Data
2	Maintain Telecommand Override
1	Maintain Battery Condition
0	Maintain Thermal Balance

Figure 2. Competency Layers for Orbital Spacecraft.

The Maintain Battery Condition competency maintains a battery charge/discharge cycle that will ensure long battery lifetime. This must be adaptive in the face of variations in battery and solar cell characteristics over the course of their lifetime, and must accommodate individual cell failures and variable solar irradiation cycles.

Levels 0 and 1 ensure spacecraft survivability, while successively higher layers are concerned with spacecraft accessibility, performance, and increasing independence from the ground. The Maintain Telecommand Override (level 2) competency is provided to ensure that, so long as the survival of the spacecraft is not threatened, the behavioural control system can be overidden by direct ground control. It is not, however, envisaged that this should sever the layered structure of the behavioural system. Rather, it provides an orthogonal access mechanism by which it is possible to inject data into the behavioural control system, and should have the capacity to override internal interconnections between spacecraft behaviours. Hence this level ensures command access to the spacecraft.

Acquistion, Conditioning and Downlink of Engineering Telemetry data (level 3) ensures access to data which is critical for diagnosing and understanding the status of the spacecraft from the ground. Maintainance of Attitude Stability and Orbit (level 4) are secondary priorities after maintaining the engineering telecommunications link, since the link is vital for understanding the state of the spacecraft, and initiating telecommand override if necessary.

The Acquisition, Conditioning, and Transmission of Instrument Data (level 5) can be ensured as a level above maintaining basic survival, two way telecommunications, and stability. Stability control (level 4) is an operational precedent for Planning and Execution of attitude and orbital control (AOCS) Maneuvers (level 6), and the acquisition and transmission of instrument data must also be accounted for in the AOCS maneuver planning process. AOCS Maneuvering competency is a precondition for autonomous Planning and Scheduling of Data Acquisition and Transmission (level 7).

Rendezvous and Dock (level 8) is a high level competency. Although this is already

automated at a low level in systems for docking Salyut and Progress vehicles with the MIR space station, in an integrated, behavioural control system it occupies a high level in order to automate all of the planning and resource allocation activities associated with the basic rendezvous and docking procedure. Landing and Take Off competencies are mentioned at this level as competencies of orbital spacecraft which also land on and take off from planetary bodies. Plan and Execute Mission Based Upon Objectives (level 9) allows the highest level of command abstraction in specifying behaviour desired of the spacecraft.

Operations Reference Model

The FRM and ARM represent general functions necessary to decompose a global task into process variables and the related information architecture. The Operations Reference Model (ORM) addresses a different question, that of how the overall A&R system is operated. The aim is to help to structure the definition of operating modes and achieve a clear and common understanding of what is meant. The ORM can also be used as a support tool for a systematic and consistent draft specification of man/machine task allocations, and allocations of which tasks are to be performed on the ground and which on the spacecraft.

Virtual machine models are useful as a basis for increasing command abstraction. Hence, on the basis of the ORM, separate virtual machine models can be used for telecommand generation and subsequent injection of data into each competency level. This provides a basis for telecommand language specification at several abstraction levels.

From Analysis to Design Synthesis

The primary sources for the operational analysis are the global spacecraft mission objectives, the characteristics of the process to be handled, and the layout of the spacecraft devices to handle this process. The application of the reference models means that the decomposed activity hierarchy of the operational analysis can be used to define functional and performance specifications for system competencies. Hence the hierarchical structures defined during operational analysis are used to develop specifications for the non-hierarchical layered competency model, and this competency model provides the basis for system design synthesis, as shown on Figure 3.

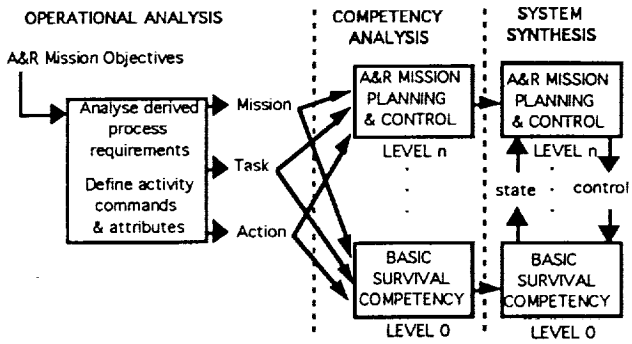

Figure 3. Sequence from analysis to synthesis.

System synthesis involves firstly finding the intermediate detailed specification of system competencies based upon a general Application Reference Model and the operational analysis, and then finding the appropriate design solution for the competency layers of the controller, as shown in the right hand side of Figure 3. This results in system solutions that are valid for classes of mission scenarios, in that activities to be performed become more generic towards the lower decomposition levels, supporting a 'standard set' of competencies which can accommodate evolving or changing mission task requirements without a critical impact upon the design.

Control System Synthesis

Elfving and Kirchoff (*ibid*) suggest that clear traceability of solution decisions made during design synthesis can be achieved by an inverse of the design process: use is made of a set of possible technical solutions which originates from existing fundamental engineering theories and which can be systematically ordered into solution trees by applying hierarchical structuring principles. While this is a desirable goal, it depends upon a well understood synthesis process. For a behavioural approach to orbital spacecraft design, this is premature. Since the spacecraft

control architecture presented here is novel, it represents at least one traversal of a more comprehensive solution tree, yet to be fully articulated. There are also major ongoing research questions regarding the relationship between the logical definition of system competencies and the implementation of the logical design (as discussed above).

Control System Example

The version of the subsumption architecture adopted for current purposes allows a module input to be suppressed by an output from another module, and the input signal is replaced by the suppressing signal (following Brooks, 1986). Limited aspects of control will be considered here to illustrate the principles involved. Behavioural interactions to achieve level 0 and level 2 competencies within a very simple model will be considered. Sensor inputs for this example comprise a set of temperature sensors. Actuators will comprise a telemetry transmitter and a receiver, each with controllable power levels (telemetry data flows will not be considered).

The level 0 competency maintains the thermal balance of the spacecraft. Temperature sensors distributed throughout the spacecraft and associated with each controllable component and subsystem are the primary inputs to this competency. The hypothetical spacecraft is a very simple design, and the only active temperature control available is by increasing or decreasing the power consumption, and hence heat dissipation, of controllable units. To achieve adaptivity and maximum robustness, the sensor inputs are mapped into an analogical representation of the temperature of the spacecraft (Steels, 1990, describes analogical maps). The analogical representation stores a model of each heat generating component of the system, conductive paths, uncontrollable heat sources and sinks (eg., the sun and "dark" space, respectively), and the spatial interrelationships between these elements. The resulting analogical representation resembles a low-resolution finite-element model. For each controllable temperature source in the model, there is a separate behaviour which computes a power level signal as a function of the current and previous temperatures of the particular element, its immediate spatial neighbours, the heat transfer characteristics of their interconnections, and an approximate measure of the specific heat of the component. This method deals very robustly with the failure of any given component to respond to direct temperature control, by calculating power (and therefore temperature) levels as a function of the average temperature of the component and its neighbours, thereby allowing indirect control of components which are not directly accessible by the use of temperature flow relationships. Each control behaviour could be implemented by a separate processor, and the method of obtaining temperature control for inaccessible elements also deals with the control of elements whose associated control processors have

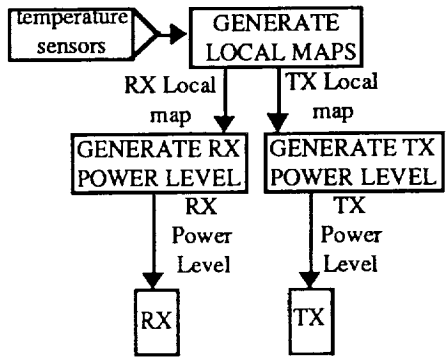

Figure 4. Example level 0 competency.

malfunctioned or failed. The resulting behavioural model is shown in Figure 4.
The analogical representation is an internal state model used by the Generate Local Maps behaviour. This behaviour may actually comprise a separate local map generator associated with each controllable unit, so the analogical model is a virtual construct. Each local map is a model of the immediate (ie. directly connected) thermal environment of the associated component, which is used by the power level generation behaviour of that component to calculate a power level which is then sent to the component.

The next level of competency considered is the level 2 Maintain Telecommand Override competency. For the current highly simplified

example, this is easy to create using a single behavioural module. The Maintain Telecommand Override module monitors the receiver power level, and suppresses the level 0 control signal if it falls below the minimum level required to operate the receiver for telecommand override. The minimum power level is then injected into the receiver input in the form of the suppressing signal from the Maintain Telecommand Override module. The structure giving this competency is illustrated in Figure 5.

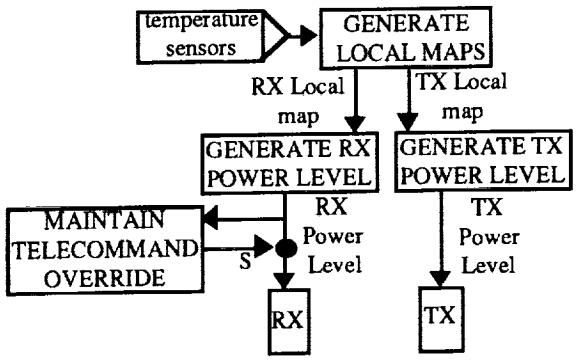

Figure 5. Example level 2 competency.

The level 0 competency can still deal with overtemperatures in the region of the receiver, not by reducing its power level below the minimum required for telecommand override, but if necessary by powering down neighbouring units. The structure and functionality of the level 0 competency ensures that this will happen automatically, without explicit control by the level 2 competency or by ground operators

In a more complete example, this minimum power level required to Maintain Telecommand Override is a function of spacecraft orientation, orbital height, position in relation to ground control stations, the states of a number of alternate transmitter system modules, etc., so the Maintain Telecommand Override competency emerges from a much more complex interaction between a number of modules.

Conclusion: The AUSTRALIS-1 Spacecraft Project and Ongoing Research

AUSTRALIS-1 is a microsatellite project currently under development by a number of Australian universities. Ongoing research with the spacecraft control architecture described here will be conducted in relation to the AUSTRALIS-1 project. A comprehensive control simulator for the satellite based upon the described architecture is currently under development. The outcome of simulation studies will be a detailed control architecture design, with well understood tradeoff studies of different control system configurations. Another major ongoing area of research will consider the relationship between the logical control structure and the physical computational architecture upon which it is implemented. It is particularly desirable to define a hardware architecture compatible with many possible mappings from the logical control model to its implementational structure. To this end, the hardware design approach will follow the following sequence:

1. a basic set of modules will be defined representing subsystems and/or components required to achieve desired spacecraft competencies in relation to the specific requirements of the AUSTRALIS-1 mission. Module definitions will include the definition of outputs from the modules (which are equivalent to sensor values or status signals), inputs to the modules (equivalent to actuator commands), the different operational modes of each module, the association of different sensor outputs and actuator inputs with different operational modes, and the definition of the conditions under which transitions between the different operational modes will occur (both in response to command inputs, to sensor values, and to internal states). This set of modules will be referred to as *substrate* modules, and constitute the lowest level description of the machine to be controlled.

2. a generic module interface unit will be defined at the physical and protocol levels. The connection between any two modules should be asynchronous, bidirectional, and

configurable between 1- and n-bit binary numbers.

3. each substrate module defined in step 1 will have a standard microprocessor-based interface unit associated with it.

4. in addition to the substrate modules, based upon an estimate of overall bandwidth requirements to achieve all levels of competency (or a specified subset thereof), a number of additional processing nodes will be specified, each with a standard interface, and a generic processor defined in order to implement each additional node. (The number and capacity of additional nodes required will be explored by simulation of the architecture.)

5. all behaviours will be integrated into a fully connected control signal network via their standard interfaces.

6. each standard processor will in addition have a dual-redundant connection to a behavioural override processor (BOP). The BOP will be able to download transfer functions to each processor in the control network, to allow downloading of software implementing different logical control architectures. The BOP will also be able to drive each behaviour directly, or bypass the behavioural control network altogether.

A significant difference between spacecraft and the mobile robots typically used to explore the behavioural paradigm is the explicit and substantial role of spacecraft as data acquisition, conditioning, and transmission systems. Options for accommodating this function include distributed routing of data via the behavioural control network, or use of a more conventional data communications topology (eg. a bus, ring, or star network). The approach adopted will be to provide for any of these mechanisms:

7. a high speed asynchronous communications link will be part of the standard processor interface, with multiplexed connections both to other processors in the network and to the BOP processor.

This control architecture will provide a hardware platform that conforms with the behavioural paradigm, but allows for considerable experimentation with different methods of mapping the logical control structure onto the hardware structure. In particular, the ability to download module transfer functions from the BOP will support experimentation with:

- dynamic control system reconfiguration
- adaptivity and learning of module transfer functions, network topology, and behaviours; ie. all of the control structure above the substrate level
- the distribution and form of multiple computational paradigms within the behavioural framework

In-orbit experiments will comprise a series of empirical evaluations aimed at refining and evaluating different detailed logical control models, the development of a software architecture and mappings onto the hardware architecture for each logical model, and methods of achieving adaptivity and robustness within these models, at increasing levels of behavioural competence. The results of the project will be valuable in defining and understanding techniques for increasing the autonomy of space systems in many diverse applications.

References

Arkin R. C., 1990, "Integrating Behavioural, Perceptual, and World Knowledge in Reactive Navigation", in Maes, P. *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*, MIT/Elsevier, 105-122.

Bonasso R. P., 1991, "Integrating Reaction Plans and Layered Competencies Through Synchronous Control", *12th International Joint Conference on Artificial Intelligence*, Morgan Kaufman, 1225 - 1231.

Brooks R. A., March 1986, "A Robust Layered Control System For A Mobile

Robot", *IEEE Journal of Robotics and Automation*, 2(1), 14-23.

Brooks R. A., 1990, "Elephants Don't Play Chess", in Maes, P. (ed) *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*, MIT/Elsevier, 3-15.

Brooks R. A., 1991, "Intelligence Without Reason", *12th International Joint Conference on Artificial Intelligence*, Morgan Kaufman, 569 - 595.

Brooks R. A. and Flynn A. M., Oct 1989, "Fast, Cheap, and Out of Control: A Robot Invasion of the Solar System", *JBIS*, 42(10), 478-485.

Ciarlo A., Donzelli P., Katzenbelsser R. and Møller B. A., 1987, "Satellite On-Board Applications of Expert Systems", *ESA Journal*, 11, 31-44.

Ciarlo A., and Schilling K., 1988, "Application of Expert System Techniques to the Cassini Titan Probe", *ESA Journal*, 12, 337-351.

Elfving A. and Kirchoff U., 1991, "Design Methodology for Space Automation and Robotics Systems", *ESA Journal*, 15, 149-164.

Erickson J. D., 1987, "Intelligent Systems and Robotics for an Evolutionary Space Station", *JBIS*, 40, 471-482.

Gat E., 1991, *Reliable Goal-Directed Reactive Control of Autonomous Mobile Robots*, PhD Thesis, Virginia Polytechnic Institute and State University, revised version.

Grant T. J., 1991/1992, "A review of Multi-Agent Systems techniques, with application to Columbus User Support Organisation", *FGCS7*, 413-437.

Leinweber D., Spring 1987, "Expert Systems in Space", *IEEE Expert*, 26-36.

Maes P., 1990a, "Guest Editorial", in Maes, P. *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*, MIT/Elsevier, 1-2.

Maes P., 1990b, "Situated Agents Can Have Goals", in Maes, P. *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*, MIT/Elsevier, 49-70.

Malcolm C. and Smithers T., 1990, "Symbol Grounding via a Hybrid Architecture in an Autonomous Assembly System", in Maes, P. *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*, MIT/Elsevier, 123-144.

Payton D. W., 1990, "Internalised Plans: A Representation for Action Resources", in Maes, P. *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*, MIT/Elsevier, 89-104.

Pidgeon A., Howard G. and Seaton B., 1992, "Operational Aspects of Spacecraft Autonomy", *JBIS*, 45, 87-92.

Raslavicius L., Gathmann T. P., and Barry J. M., 1989, "An Artificial Intelligence Framework for Satellite Autonomy", *Second International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, 2, 536-543.

Spier R. J. and Liffring M. E., Nov, 1989, "Real-Time Expert Systems for Advanced Power Control", *IEEE AES Magazine*, 33-38.

Steels L., "Exploiting Analogical Representations", in Maes, P. *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*, MIT/Elsevier, 71-88, 1990.

Steels L., 1991, "Emergent Frame Recognition and its Use in Artificial Creatures", *12th International Joint Conference on Artificial Intelligence*, Morgan Kaufman, 1219 - 1224.

Tello E. R., 1986, "Dipole: An Integrated AI Architecture Suitable for Space PRogram Applications", *Expert Systems in Government Symposium*, IEEE, 168-180.

Woods D., April 1992, "Space Station Freedom: Embedding AI", *AI Expert*, 33-39.