05ICES-120

Requirements for an Autonomous Control Architecture for Advanced Life Support Systems

G. Biswas[1], P. Bonasso[2], S. Abdelwahed[1], E.J. Manders[1], D. Kortenkamp[2], J. Wu[1], and S. Bell[2]
[1]Dept. of EECS & ISIS, Vanderbilt University, Nashville, TN; [2]NASA Johnson Space Center/ER2, Houston TX.

ABSTRACT

This paper describes a series of life support control experiments at NASA Johnson Space Center and at Vanderbilt University. These experiments involved two distinct, layered control architectures; one used a model-based approach and the other a procedural approach to control complex, distributed systems. Both sets of experiments produced good results, but it also brought out the strengths and weaknesses of the two underlying technologies. Our goal in this paper is to come up with the requirements for an integrated architecture for autonomous controller design that combines the best of the two approaches. This paper discusses a potential approach to this integration and the advantages it offers.

INTRODUCTION

NASA has been designing advanced life support (ALS) systems to support long-duration manned missions to the moon and Mars [1]. Successful execution of such missions without compromising the safety and scientific goals of the mission will require the design and development of integrated computational architectures that support, planning, health monitoring, and advanced control of such systems to maintain autonomous, efficient, and reliable operation.

ALS systems are complex. They involve interactions between biological systems, such as humans and plants on one hand, and a number of complex physical processes that cover the chemical, thermal, mechanical, electrical, and fluid domains [2]. The system is made up of multiple loosely coupled subsystems, such as (*i*) a Water Recovery System (WRS), (*ii*) an Air Revitalization System (ARS), (*iii*) a Power generation system, (*iv*) a Thermal control system, (*v*) a Biomass production system, (*vi*) a Food production subsystem, and (*vii*) a Solid waste collection and conditioning system. These subsystems comprise a number of interacting control loops, such as the fluid flow loop, the energy management loop, the thermal control loop, the bio-regeneration and gas transfer loop, and the chemical production loop. A number of these loops are also closed, in that materials (such as oxygen and potable water) have to be continually regenerated with minimum losses.

Our goal in this work is to provide a computational framework that supports autonomy and robust operations for long-duration manned missions while maintaining safety and efficient performance. Long-duration manned missions, present a number of unique requirements and challenges that NASA has not fully addressed before. The complexity of the systems combined with long communication delays and information blackout for periods of time requires that the spacecraft systems acquire a significant degree of autonomy. The operation in unknown environments and the increased likelihood for component degradation and failures makes it necessary that the systems be equipped with health monitoring, diagnosis, and adaptive control algorithms. Moreover, autonomous operation must also allow for human crew intervention at various levels from goal setting and refining to system-level control decisions, and the flexibility to adjust to changing mission goals and environmental conditions. This calls for tight integration between planning, control, diagnosis, and health monitoring. We believe the proposed multi-level integrated planning and control architecture is a first step toward addressing a number of these issues.

In this paper, we present a multi-level computational architecture that integrates planning and hierarchical control schemes to develop a dynamic planning and control system that is reactive and fault-adaptive, but at the same time, is designed to manage resources for the duration of a long mission. The computational architecture adopts a novel approach to integrating components of the 3T control architecture developed at NASA JSC and Metrica [3] with the hierarchical model-based multi-level control systems that have been developed at Vanderbilt University [4].

Section 2 discusses previous work at NASA JSC in advanced life support control using the 3T control system. Section 3 presents the challenge mission that was developed recently at NASA JSC, a 90 day stay by four astronauts in a lunar habitat. Section 4 describes the lunar habitat and ALS modeling, and the control scheme developed at Vanderbilt. Section 5 extracts integrated control system requirements from these previous activities. Section 6 describes our proposed integrated control architecture that meets these requirements. Section 7 presents the conclusions and directions for future work.

3T ALS EFFORTS

Since 1995, Bonasso, Kortenkamp and others have been building AI control systems in support of Johnson Space Center's investigations in advanced life support (ALS). In 1995, as an early human test, the ALS group experimented with a human in an airlock linked to a ten-foot diameter chamber of wheat [5]. For fifteen days, the human lived, worked and exercised in the chamber airlock while the wheat crop took in the carbon dioxide he generated, and provided oxygen. The AI control system (discussed below) monitored and provided caution and warnings for the climate and nutrient environment of the wheat crop.

In 1997, the ALS group experimented with four people in a thirty-foot chamber for ninety-one days [6]. A physical-chemical air revitalization system (ARS) recycled the air for three of the four people, while a wheat crop in the ten-foot chamber did the same for the fourth. The ALS team also experimented with a solid waste incinerator in the airlock of the ten-foot chamber. The AI-based control system managed the transfer of O_2 and CO_2 among the gas reservoirs for this test to ensure crew and crop health and to recycle gases produced by waste incineration. The reservoirs included a crew habitat, the plant chamber and airlock, and a number of pressurized tanks. Operating 24/7, the AI system also employed a generative planner that scheduled waste incinerations and crop planting and harvesting, coordinating those tasks with the day-to-day product gas transfer.

For both of these projects the AI control team used a three-layer architecture known as 3T [3] to design, organize and develop the control software (see Figure 1). 3T separates the general intelligent control problem into three interacting tiers:

• A set of hardware specific situated skills or behaviors that represent the architecture's connection with the world through the sensors and actuators. The term situated skills is intended to denote a capability that, if placed in the proper context, will achieve or maintain a particular state in the world. 3T's implementation includes primitive actions, queries, and monitoring events that can be combined to form autonomous behaviors. 3T's skill layer is a distributed set of skill groups coordinated by a skill manager for each life support subsystem, e.g., the ARS.

• A sequencing capability that can differentially activate the situated skills in order to direct changes in the state of the world and accomplish specific tasks. 3T uses the Reactive Action Packages (RAPs) system [7] for this portion of the architecture. The RAPs engine is an interpreter, indexing RAPs (essentially sets of linear plans) from a library based on the changing world situation. Thus, one can change a RAP or add new RAPs while the sequencer is executing. Other AI "executives" provide similar capabilities, e.g., [8]

• A deliberative planning capability that reasons in depth about goals, resources and timing constraints. 3T's hierarchical task net planner known as AP [9] uses the highest level RAPs as its primitive plan operators, and can replan both spatially and temporally. This planner can deal with multiple, concurrent goals and multiple agents. It has an integrated replanning capability to handle changes caused by unforeseen events. It also has a scheduling component for optimizing task execution. Other planners can provide a similar capability, e.g.,[10].

The planner is efficient, but it becomes even more potent when its level of detail is abstracted to the RAPs of

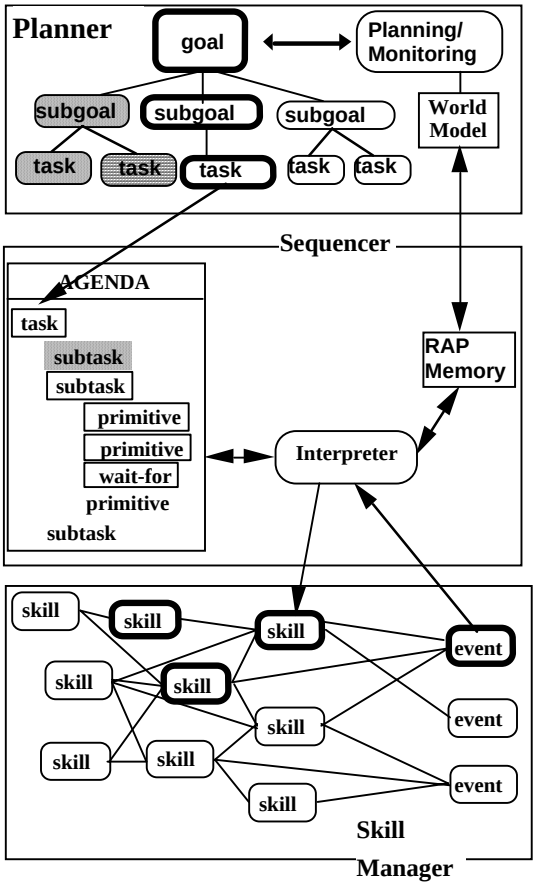


Figure 1: The 3T Intelligent Control Architecture

the sequencing layer below it. It is important to note that once the planner generates a plan, it executes the plan by placing primitive plan actions on the sequencer's agenda and monitoring the results of the sequencer's actions. Communication among the layers uses the IPC message passing protocol [11]. With this communications infrastructure, data from any part of the system can be monitored by any other part of the system.

A key aspect of 3T is that it gives developers the ability to integrate the continuous, near-real time control algorithms in the bottom layer with advanced AI algorithms in the top layer, i.e., automated planners and schedulers that are event driven but more computationally expensive. 3T does this through the integrating action of the middle layer. Essentially, the middle layer translates the goal states computed by a planning/scheduling system

into a sequence of continuous activities carried out by the skills layer, and interprets sensor information from the skills layer as events of interest to the upper layers.

3T applications run autonomously due in large part to the principle of "cognizant failure" [12] embodied in each level of the architecture. The skills level notifies the sequencer when it fails to achieve or maintain required states; the sequencer uses alternative sequences when the primary methods fail, ultimately putting the control system in a safe state; and the planner can synthesize alternative plans in light of the failures of the lower two tiers.

From 1998-1999, the AI team used the bottom two layers of 3T to provide autonomous control for a second-generation biological water processor during a 450 day 24/7 test. From 2000 to 2002 the AI team used 3T to help the ALS group develop and control an advanced water recovery system, AWRS [13]. The AWRS consisted of four next generation WRS subsystems, which generated potable water using fewer consumables (filters, resins, etc.) and much less power than the components then in use: (i) a biological water processor (BWP) to remove organic compounds and ammonia; (ii) a reverse osmosis (RO) subsystem to remove inorganic compounds from the effluent of the BWP; (iii) an air evaporation system (AES) to recover additional water from the brine produced by the RO; and (iv) a post processing system (PPS) to bring the water to within potable limits. The combined total number of sensors and actuators used for control numbered over 200.

In both the air regeneration and water recovery operations, 3T's autonomous execution reduced human workload significantly. Human participation was still necessary, but as intended, the human's role in control changed from vigilant monitoring with frequent command intervention to supervisory monitoring with infrequent intervention. Some tasks, such as calibrating sensors, and changing out filters are not easy to automate and still require human intervention.

In addition, despite 3T's capability for autonomy, it was important that the user not only have insight into system operations but also have the provision to take control of the system, when necessary. We have developed a number of user interfaces to the planner, sequencer and to primitive sequences to facilitate user interaction. Moreover, in both the 91 day human tests and the two year AWRS tests, infrequent anomalous situations, such as computer hardware failures, network performance problems, and data acquisition anomalies occurred, which required human intervention. We also discovered that biological regenerative systems are more sensitive to their environment than physical–chemical systems and were thus more likely to exhibit surprising or unexpected behaviors requiring intervention. For example, microbial clogs in the BWP required a manual procedure to slough a layer of the microbe colony; loss of wheat crops due to nutrient leaching required planting extra crops and manually adjusting nutrient balance. Our approach to design automation for these anomalous situations was to provide capabilities for adjustable control autonomy [14], so that users could (i) temporarily suspend portions of the automation while carrying out manual activities, such as maintenance, (ii) rely on the control system to continue monitoring flows and rates even if the human had temporarily taken over control, (iii) adjust set points and warning thresholds without taking the control system offline.

90-DAY SCENARIO

The 3T approach presents an excellent framework where humans and partially automated control is integrated to provide effective control of complex systems. To extend this approach to even more complex situations, such as a set of interacting distributed systems that have to operate reliably for long periods with minimal intervention from humans (these are astronauts whose primary task involve exploration and science tasks in hostile environments), we refer to an Advanced Life Support design problem for a 90 day reference mission with four astronauts involved in a one time use of a lunar habitat [15]. The specifications state that the habitat, located in the lunar South Pole, is to be initiated and operated nominally upon crew arrival. The challenge mission document specifies a 28 day cycle, with 14 days of light with the sun above the horizon followed by 14 days of night. Our system design used a large array of solar cells for power generation and Ni-Cd cells for storage (this is the technology used on the ISS) to accommodate the larger period of darkness. The day-night cycle repeated uniformly over the 90 day period.

The surface temperatures on the lunar surface vary between 210^{o}K and 230^{o}K during the day, and drops to 50^{o}K during the night. The energy consumed to keep the habitat at 298^{o}K was taken into account in our energy computation requirements. The habitat atmosphere specified includes air composed of 29% oxygen at an overall pressure of 65.5 kPa (the rest was nitrogen) and a leakage rate of 0.00224 kg/day. Most of the food required was shipped, but the specifications did allow the possibility of growing small salad crops, such as lettuce and tomatoes. Food consumption was set at 0.257 kg/crewmember/day (cmd) of moist food and 0.665 kg/cmd of dry food. Air, water, and waste recovery systems are part of the habitat. The specified EVA activities include all four crew members involved in EVA activities for 2 days (i.e., 16 hours) during the first week of the mission, then 46 days of EVA activity over the remaining period involving teams of two crew members at a time, and a final set of EVA activities for shutdown, where all four crew members are involved in one full day of EVA activity. The value of losses at the airlock for EVA activity is specified at 10%.

It is clear that the lunar habitat for such a mission would require air and water regeneration systems, power generation and biomass production systems, as well as a number of other systems for thermal control and waste disposal. The complexity of multiple, interacting systems

operating in different regimes and at different time scales makes it hard to develop skill managers that take into account these interactions, while ensuring close to "optimal" performance and effective resource management for the duration of the mission. Inefficient control leads to large increases in the Equivalent Systems Mass index that could render the mission infeasible or prohibitively expensive. To overcome this problem, we adopted a model-based approach to system monitoring and controller design to allow for dynamic online analysis. As a result, the system could then adapt online to a variety of different scenarios. Our solution to the challenge problem was to develop a multi-level hierarchical control scheme described in the next section. To achieve this, the first step was to develop dynamic hierarchical models of the ALS systems, the power generation system, and the crew chamber. We discuss the models and controllers in the next section.

MULTI-LEVEL MODEL-BASED CONTROL

We develop the hierarchical control scheme for the lunar habitat as a three-tier architecture with a supervisory controller at the top-level controlling a set of system-level controllers for the crew chamber, the Water Recovery System (WRS), the Air Revitalization System (ARS), and the Power generation system (see Fig. 2). These systems cover multiple physical domains, and operate at multiple time scales. An effective way to describe the behavior of the controlled subsystems is to model them as hybrid dynamic systems [16, 17]. The rest of this section describes our approach to modeling the lunar habitat and then discusses the hierarchical control scheme implemented for the 90 day lunar mission scenario.

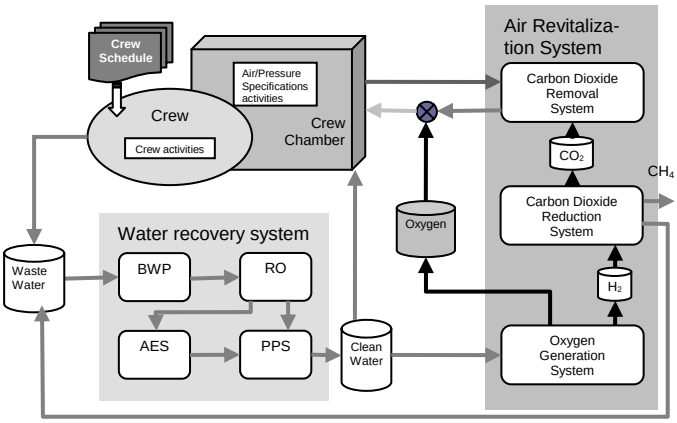


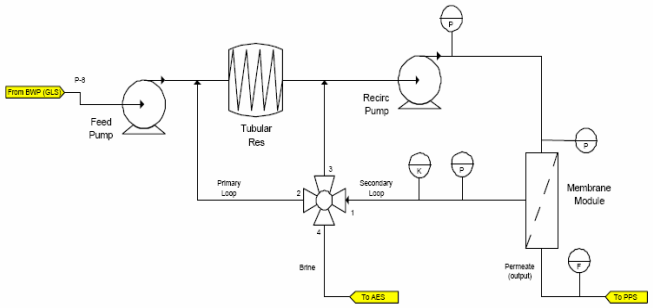
FIGURE 2: LUNAR HABITAT WITH ADVANCED LIFE SUPPORT SYSTEM

LUNAR HABITAT MODEL

The lunar habitat is designed for a crew of four astronauts. We briefly describe the four systems and our approach to modeling these systems in the MATLAB/SIMULINK environment.

THE WATER RECOVERY SYSTEM - This system recycles urine and wastewater into potable water. As discussed earlier, the WRS, itself has four subsystems: (i) the BWP, for removal of organic compounds including ammonia using a packed bed and nitrifier columns, (ii) the RO system for removal of particulate inorganic matter from the water using a membrane, (iii) the AES, for purifying the concentrated brine by an evaporation process, and (iv) the PPS, for removal of trace organic and inorganic compounds by ultra-violet treatment to bring the water to potable limits. The combination of the BWP and RO subsystems produce about 85% of the clean water. The remaining 15% is produced by an evaporation and condensation of concentrated brine that is passed to the AES from the RO subsystem.

The RO subsystem, shown in Fig. 3 is the linchpin subsystem in the WRS loop. It pulls water from the GLS (gas liquid separator) of the BWP and pushes it through a cylindrical membrane that acts like a molecular sieve at high speed. The clean water permeate is passed on to the PPS. The RO is designed to go through six modes of operation. The primary mode draws water into a coiled section of pipe that acts like a reservoir, while processing permeate in the outer loop. When the brine concentration increases above a preset level, the system is switched to a secondary mode, where the brine is circulated faster in the smaller inner loop with the recirculation pump, to push it harder against the membrane to keep the clean water production at a reasonable rate. The concentration of brine in the inner loop continues to increase, and when it becomes high enough to reduce the output from the RO system significantly, the brine is purged into the AES, a new batch of water is drawn in from the GLS, and the primary cycle starts again. Periodically, as particulate matter accumulates in the membrane, it needs to be cleaned by running the water back-



backwards in the inner loop.

Figure 3: RO system schematic

The SIMULINK model of the RO system was derived by decomposing the system into three principal domains of operation. The mechanical and fluid domains are the primary energy domains that define the flow behavior in the system. However, the effect of time-varying impurities in the water on the flow process is accounted for by explicitly modeling the fluid conductivity domain and its interactions with the flow process [18].

The AES subsystem contains a reservoir to collect the brine. The brine is absorbed onto a wick and evaporated using hot air. The evaporated water is condensed by passing it through a heat exchanger, and collected in a tank before it is sent to the PPS system. The SIMULINK model for the AES consists of three domains: hydraulic, pneumatic and thermal. The hydraulic domain models the amount of vapor being generated in the wick and the amount of vapor condensed in the heat exchanger. The pneumatic domain is modeled simply with a blower pushing air through a pipe modeled as a resistance. The thermal domain defines the primary behavior of the AES, and uses capacities to model the heat capacity in the AES loop [19].

THE AIR REVITALIZATION SYSTEM (ARS) – This system replenishes the oxygen after removing the excess carbon dioxide generated by the crew before the air is circulated back to the crew chamber. This CO_2 removal task is performed by the Carbon Dioxide Removal Assembly (CDRA). A second task is to recover the O_2 from the CO_2 exhaled by the crew. This is done in two steps. First, H_2 and O_2 are generated by electrolysis of water using an Oxygen Generation Assembly (OGA). The oxygen goes into a storage tank, and the hydrogen is fed into a reactor (CRS) to reduce the carbon dioxide to water and methane. In the current configuration, the water is sent back to the WRS for purification, and the methane is vented. Fig. 2 shows the interactions of ARS subsystems with other ALS subsystems.

The CDRA subsystem uses a "four-bed molecular sieve" to remove CO_2 exhaled by the crew. At any time two of the beds are adsorbing CO_2, while the other two are releasing adsorbed CO_2 as they are heated. When the adsorbing beds become saturated, they are switched to the desorption mode, and the two previously desorbing beds are used for adsorption. This cycle occurs several times a day. A compressor takes the desorbed CO_2 and stores it in a tank before it is passed on to the reduction system. The CDRA system involves complex spatial-temporal dynamics. In this work, we build a simplified lumped parameter model in Matlab/Simulink, with multiple lumps to capture the spatial dynamics. The input parameters for the CDRA include system pressure, cycle time, airflow rate, temperature, and the inlet CO_2 level. The key outputs for this unit are CO_2 concentration and flow rate.

The CRS uses a Sabatier reactor with a catalyst to react the CO_2 and H_2. Optimal reaction performance is maintained by controlling the temperature, pressure, and the molar ratio of the gases. The Sabatier model simulates the behavior of one primary reactor zone and two secondary reactor zones. For this subsystem, the input parameters are: system pressure, temperatures and inlet H_2/ CO_2 Molar Ratio. The key simulated steady state output parameters are at the end mass (percentages) of each component involved in the reaction which show the conversion situation of CO_2 and H_2.

THE CREW HABITAT – This is the crew living and working quarters. The goal of the controller is to maintain the air quality (29% oxygen with nitrogen as the dilutent gas) and temperature in the habitat. We assume the crew consumes O_2, H_2O, and food, and the habitat provides these resources, while removing waste water and solid wastes. A biological model for a typical crew member determines the amount of resources consumed by the crew while performing different activities.

During the mission the astronauts work in pairs on a schedule that involves 8 hours of work, 8 hours of sleep, and the remaining time is divided into eating, exercising, maintenance and leisure activity. In general, the crew can either be in the habitat or outside on an EVA mission. The difference between the main habitat and EVA environment is that the main crew habitat is in the ALS loop, whereas resources produced/consumed in EVA are considered losses from the system. The crew model encapsulates activity scheduling and resource utilization based on that activity level. Each crew activity has an intensity level associated with it, and this is mapped on a heart-rate value, which in turn is used to compute the O_2 consumption and CO_2 production. This approach mimics that taken in the BioSim simulation engine [20]. The habitat model has the amount of each gas as a state variable. The state variables are updated by the gas flow through the habitat that is imposed by the ARS, and the gases consumed/produced by the crew. In our model, it was assumed that the air circulated at a constant speed through the chamber and the ARS. Similarly, both the potable water consumed and waste-water produced are coupled to the WRS system.

THE POWER SYSTEM – An elementary generation and storage model patterned after International Space Station (ISS) technology is employed. An array of solar cells generates the required energy to sustain all of the ALS systems and provide the thermal energy to keep the crew chamber at 298^{o}K, while also generating excess

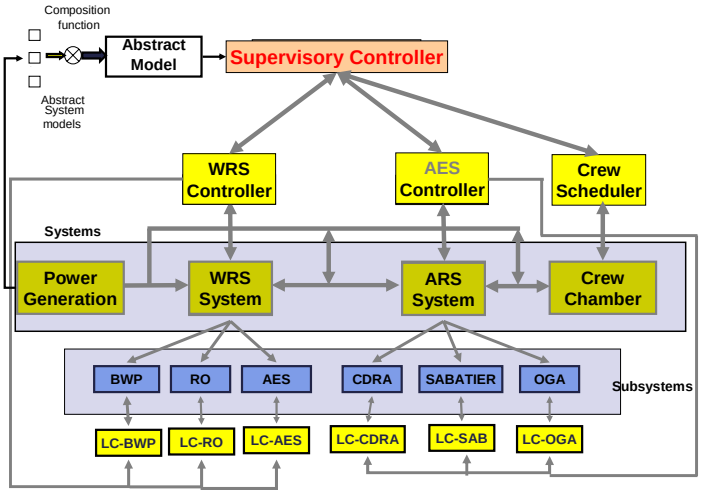


Figure 4: A Multi-level Control Structure

energy that is stored in Ni-Cd batteries for use during the night (dark) periods when no power can be generated by

the solar array. The power generation model is linear, i.e., it uses a constant production rate. Power consumption is also defined by linear functions specific defined over time intervals.

MULTILEVEL ONLINE CONTROL

This approach applies to distributed systems, each having its own control specification [21]. Typically, these systems must interact to achieve a desired global objective. This suggests a multi-level distributed control structure, where systems have independent controllers, and system interaction is managed by a supervisory controller that addresses overall requirement specifications. Fig. 4 shows the multi-level control structure.

LOCAL CONTROLLERS – At the lowest level of the control structure a set of local controllers manage the individual subsystems of the lunar habitat. Each subsystem has an individual optimizing controller, which does not directly interact with other subsystems. Interactions are handled by the system controller and by placing physical buffers between subsystems. The local controllers receive commands in the form of input-output requirements and resource constraints from the system-level controllers at periodic intervals. These requirements and constraints are specified as modes of operation, which have accompanying control objectives and system parameters, such as control input restrictions.

The local controllers continuously monitor the current state of their subsystem, and select the input that best satisfies the given specification. In addition, the controller is required to keep the system stable within the domain that satisfies the specification. In this setting, the controller is simply an agent that generates a sequence of events to achieve a given objective. This objective is typically expressed as a multi-attribute utility function that takes the form $\sum_i V_i(P_i)$, where each V_i corresponds to a value function associated with performance parameter, P_i. The parameters, p_i, can be continuous or discrete-valued, and they are derived from the system state variables, i.e., $P_i(t) = p_i(x(t))$. The value functions employed have been simple weighted functions of the form $V_i(P_i) = w_i P_i$, where the weights take on values in the interval [−1 1], and represent the importance of the parameter in the overall operation of the system. The controller explores only a limited forward horizon in the system state space and selects the next event based on the input that maximizes a given utility function (see Fig. 5). This function assigns to each state of the system a cost associated with reaching and maintaining that state [22].

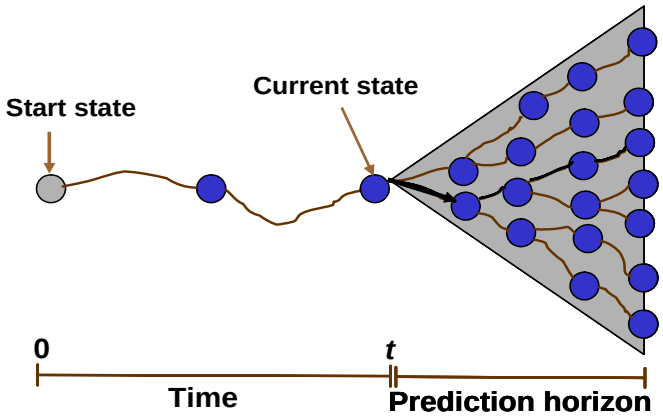


Figure 5: Limited Lookahead Search

SYSTEM CONTROLLERS – These controllers are responsible for managing the interactions between controlled subsystems by managing their interactions through the intervening buffers, and by distributing resources, such as power, in a manner that all of the subsystems can produce their desired output in an efficient way. The controllers at this level use an abstract model defining the average behavior of the subsystems and how they affect the level at the connected buffers. Note that system controllers target only the buffer quantities and not the dynamics of subsystem operation. The main objective is to maintain buffer levels based on mass flow predictions considering the crew schedules. Like local controllers, the function of the system-level controllers is influenced by the global controller. The global controller chooses from among a finite number of options to adjust the behavior of these controllers. The habitat controller features the WRS, ALS, and crew chamber controllers.

SUPERVISORY CONTROLLER – This controller ensures the mission success by balancing resource consumption with the available level of resources [23]. The daily crew schedule is computed based on the constraints and performance goals determined by the global controller. Specifically, the global controller allocates resources for maintenance, exercise, and EVA activities.

Since a detailed behavioral model of the underlying distributed system may be very complex, the Supervisory controller uses an abstract (simplified) model to describe the composite behavior of the system components that is relevant to the overall requirements and operational constraints. The abstract model uses a set of global variables that are related by the input-output interactions between the individual systems. Moreover, since the global controller's decisions are based on aggregate behaviors, which are determined over longer time frames compared to the individual systems. The global model is represented by $y(k + 1) = g(y(k), v(k), \mu(k))$, where $y(k)$ is the global state vector, $v(k) \in V$ and V is the set of global control inputs which represent a set of local control settings for the local modules, and $\mu(k)$ are the global environmental inputs. The map g defines how the global state variables respond to relevant changes in environment inputs with respect to the global control inputs. The objective of the system controller is to minimize a given cost function over the operation span of the system. We assume here also that the cost function takes the form of the set point specification. The global specifications are communicated down to the system and subsystem controllers. The local controllers try to optimize the performance as described earlier, while ensuring that conditions imposed by the system controller are not violated. To summarize the hierarchical con-

trol scheme, the supervisory controller performs the following functions:

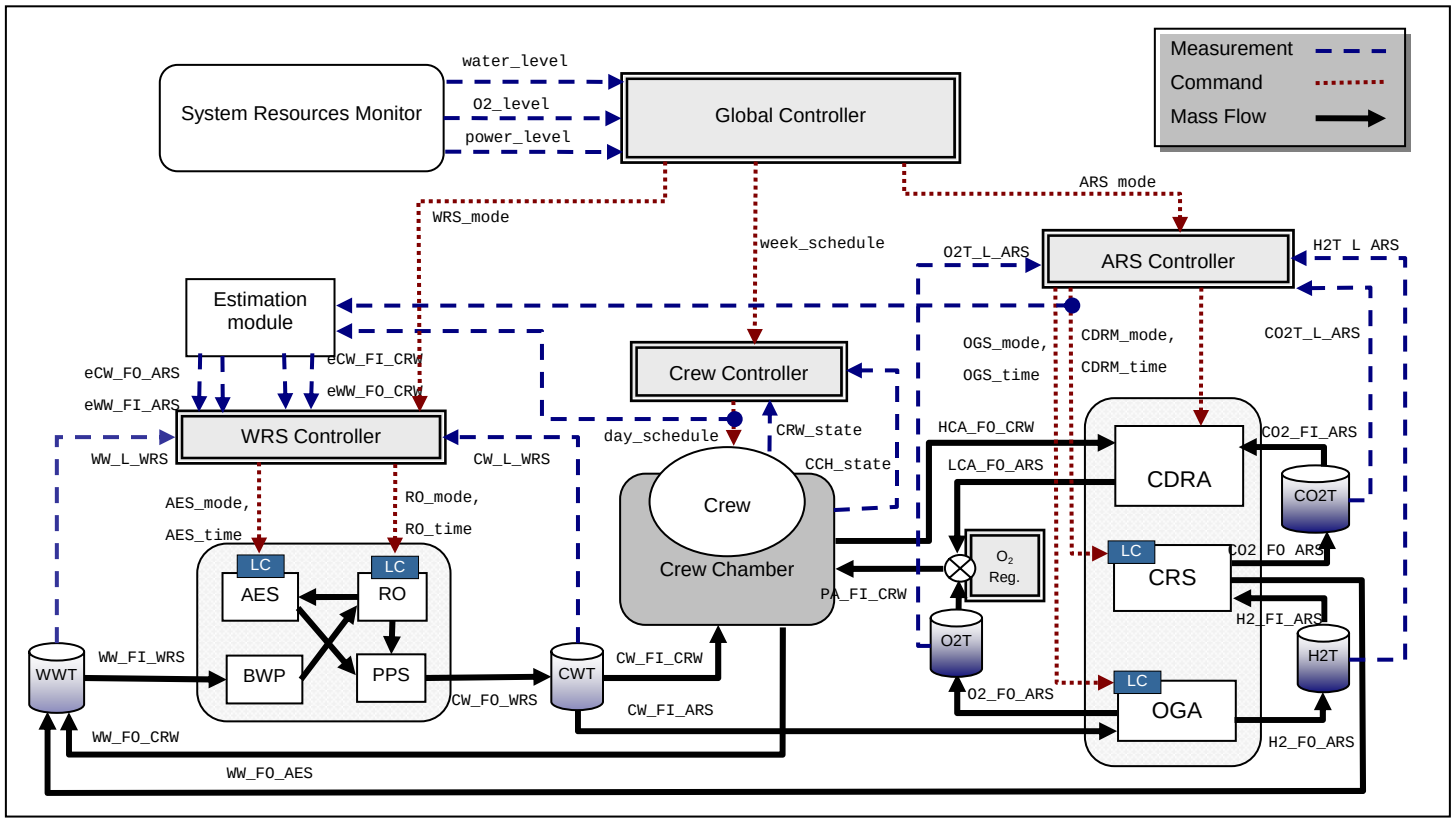


Figure 6: Habitat Structure with Hierarchical Control

- Forecast long-term trends of the environment and based on the abstract system model examines the effect on the overall performance of the system.
- Optimize the system performance by changing the operational settings of local module, or the distribution of loads and resources among these modules
- Obtain performance feedback from local modules, which then used to identify the current global state.

The computational structure and information flow for the three tiered control structure is illustrated in Fig. 6. As discussed, information flow is strictly top-down for control commands with the top-level having the highest priority, and bottom up for information on state of the subsystems and the systems. Each level of control addresses a different aspect of overall system behavior.

EXPERIMENTS

We present simulation experiments run on the 90 day challenge scenario to illustrate multi-level online control of the system. The scenario assumes four crew members on the habitat. On the average, they consume 9 liters of water and 1 kg of O_2 per day, while producing 1 kg of CO_2 per day. Controllers at the subsystem-level use the following information to compute their control actions: (i) estimated crew activities; e.g., for the WRS, according to the daily schedule, the waste water created per hour and potable water spent per hour can be estimated, and for ARS, the CO_2 level can be estimated; (ii) average production and treatment rates and corresponding average power consumed for each preset modes; and (iii) values of the relevant buffers.

The effect of the multi-level control schedule is shown in Fig. 7. This figure shows the evolution of levels of waste water tank and potable water tank for the 90 days. In our 90 day simulation run, the initial store of potable water was 650 liters, and it reduced to 150 liters at the end of the mission. The maximum value of waste tank is never more than 25 liters.

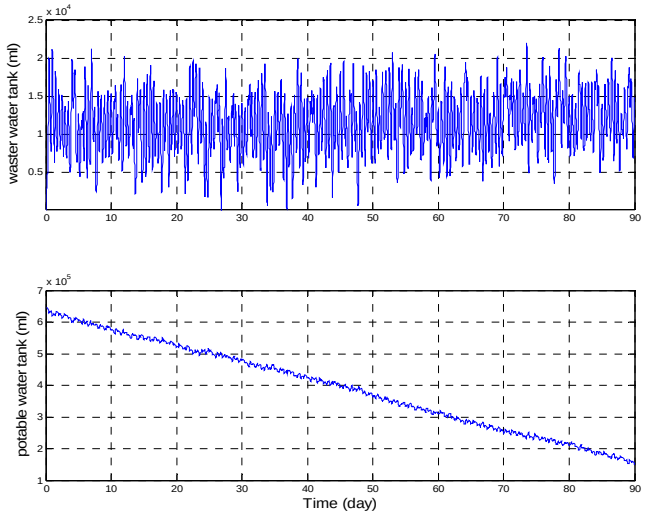


Figure 7: The level of waste and potable water tanks during the 90 day mission

Fig. 8 shows the O_2 and CO_2 buffer levels. The O_2 tank level varied between 9.8 kg and 10 kg, and the CO_2 buffer never exceeded 1.4 kg. We believe the controller design was the key to achieving high levels of regeneration, and keeping the material values at predetermined ranges. Furthermore, the supervisory and system controllers kept resource usage within bounds, mainly to ensure that the resources would last for the duration of the mission. This further supports the assertion that model-based multi-level control not only allows for increased autonomy and efficiency during operation, but translates to much smaller buffer sizes in the design and implementation of the system.

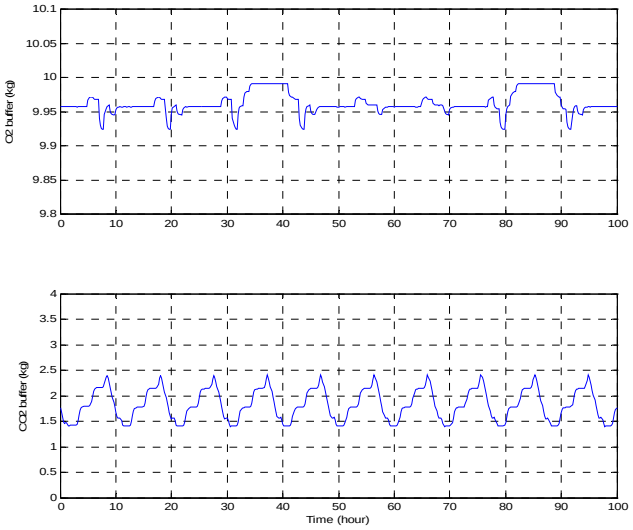


Figure 8: O_2 and CO_2 levels in the ALS buffers.

INTEGRATED CONTROL SYSTEM REQUIREMENTS

Our experiences in building integrated control systems for the life support scenarios described in the first part of this paper have led us to compile a list of requirements for integrated monitoring and control systems. These requirements can be used to guide designers in building control systems and can inform research and development in this area.

The first set of requirements concern control of the habitat and its associated life support systems. This control can be executed simultaneously at multiple levels to accommodate different time scales, while optimizing performance at different levels of granularity. Specific requirements include:

1. Controlling life support subsystems in real time, taking into account the dynamic behaviors to meet set point objectives or to maximize multi-attribute utility functions.
2. Modeling the dynamics of systems and system interactions to ensure optimal control of a non-stationary, non-linear dynamic system.
3. Using system level controllers to handle the interaction constraints between subsystems using a coarse model of dynamic interactions based on regions of operation of the subsystems to avoid computational intractability in controller design.
4. Having a top-level supervisory controller that uses an abstract constraint model of resource usage and predictive techniques to ensure resources will not be exhausted before the completion of the mission.
5. Planning and scheduling habitat activities and procedures that include crew activities, EVAs, possible robotic activities and life support actions such as planting and harvesting crops.
6. Automatic integrating between the multi-level control and planning and scheduling of habitat activities to accomplish mission objectives.
7. Integrating model-based approaches for dynamic online control with procedure-based approaches that cover activities, such as maintenance and other human-related activities that cannot be easily incorporated into plant and process models.
8. Monitoring activities, procedures and processes and determining when the current control approach is failing and adjusting accordingly.

Second, the integrated control system must maintain system health in the face of anticipated and unanticipated failures. Again, this will happen at many levels and timescales. Specific requirements include:

9. Detecting and diagnosing multiple, simultaneous system faults. Improving the effectiveness of automated detection and isolation, so faults do not cascade, and cause catastrophic effects [24].
10. Achieving desired operating conditions despite multiple, non-critical system faults.
11. Replanning and rescheduling when necessary, but with minimum possible disruption to the existing plan/schedule.
12. Bringing down one or more subsystems while the rest of the control system continues to operate with no interruption.
13. Decision making to coordinate nominal control, fault-adaptive control, and maintenance operations so to provide minimum disruptions to crew activities that are related to exploration and science tasks.

Third, the integrated control system must inform the users (both ground control experts and crew members) and allow the users to interact with it naturally and effectively to achieve mission goals. Specific requirements include:

14. Allowing for control to be overridden at any level by local or remote operators while still providing for robust control of those systems not being affected by the override.

15. Achieving multiple mission goals and resolving conflicting goals via crew or ground interaction and control system advice.
16. Providing aggregated, abstracted and summarized monitoring data and appropriate context to external programs to provide situation awareness to crew and controllers.

Finally, the integrated control system must optimize life support such that buffer and component sizes can be minimized. Effective, optimal control can reduce oscillations in resource utilization over time thus reducing the need for buffers or oversized components to deal with the oscillations. A specific requirement is:

17. Optimizing resource and buffer utilization such that total Equivalent System Mass (ESM) is reduced.

We illustrate these requirements through an example scenario. We begin with the assumption of a ninety-day mission plan that is scheduled in 28-day segments. Within the first 28-day period, the mission goal for the habitat might be "to conduct habitat operations while supporting an EVA on day eighteen". An automated planning capability (requirement 5) produces a plan of operation that includes tasks to maintain and operate the habitat, operate the WRS, ARS and crew quarters climate control, support the required EVA, sustain crop growth, and ensure safe disposal of solid waste. Using resource models of the dynamics of the habitat subsystems (requirement 4) the plan will make efficient use of power, air and water stores and habitat inventories.

Next, a reactive planning capability selects routine procedures for carrying out the first step of each part of the plan for each subsystem (requirement 3). For example, for the ARS:

1) Seven days of nominal operations.
2) Four days in high CO_2 consumption state to clear CO_2 reservoirs in preparation for incineration operations,
3) Four days in an extreme high CO_2 state to scrub the CO_2 resulting from incineration,
4) One day providing O_2 to tanks to be used for the upcoming 24 hour EVA on day eighteen, and
5) Resume nominal operations on day ten.

This sequence is then passed to a dynamic control execution capability that examines the existing resources for the ARS and suggests an extra day to ensure the O_2 tank level increases above a pre-determined value (say 10 kg). Since the extra day will still support the EVA on day eleven, the reactive planner makes no further changes to the ARS execution plan (requirement 11). The dynamic control executive issues time-ordered control specifications for all the habitat systems (WRS, ARS, Power generation, Biomass, etc.) and their corresponding subsystems commensurate with the procedures (i.e., partial plan sequences) from the reactive planner. The subsystem controllers execute the directives "optimally" taking into account the continuous dynamics of the respective subsystem (requirements 1 & 13) for the first nine days. For example, a change detection algorithm might notice an increase in power usage in the CO_2 removal system (CDRA), but its subsystem controller is able to compensate the increase by decreasing the heater temperature a little, and also adjusting blower and pump speeds (requirement 10).

On day ten, however, the dynamic control executive determines that the CDRA behavior has continued to drift away from the nominal, and the system is operating suboptimally. By now, the fault detection module has reliably established that there is a restriction in the CO_2 output line and also a leak is detected in the desiccant bed (requirement 9). The system controller has adjusted for this by reducing OGA and CRS (Sabatier) operating times, but if this trend continues, air quality in the crew chamber will start dropping below acceptable levels, or lot more energy will have to be directed toward the CDRA. With the night period approaching, this is not considered a good option (by the supervisory control predictor). This situation is reported by the supervisory controller to the RAPS (reactive planner) unit. This unit (the Sequencer) is told that it will now take five days to clear the CO_2 reservoirs. The reactive planner can make no adjustment that will compensate for the extra day and informs the planner. The planner sees the situation and determines there are options at this time such as (i) perform a CDRA repair and, (ii) drop the scheduled EVA activity.

The habitat planner considers the situation, and through its own analysis using its world model determines that a new plan that includes a two-day crew task for repair of the CDRA, which will create an O_2-restricted situation for a few days. As a result, the EVA activity is pushed back to day twenty, since one of the crew repairing the CDRA is also needed for the EVA (requirement 11). Furthermore, the astronauts are required to be cautious while exercising, e.g., none of the crew should exercise at the same time.

At this stage, using an interface to the planner (requirement 15), the habitat commander informs the planner that the EVA task cannot be slipped because it involves a communications experiment that depends on the relative orbits of the moon and the earth about the sun, a constraint unknown to the habitat planner (requirements 14 & 15). The planner, in further conference with the model-based resource manager, determines that if the crew completely omits their exercise period until after the EVA, the ARS can meet the incinerator and EVA requirements. The resulting habitat plan omits crew exercise from the crew plan and schedules the CDRA repair after the EVA (requirement 11).

When the CDRA repair takes place, the reactive planner will select an appropriate repair procedure for the crew and a set of modes for ARS and other affected subsystems, and the dynamic controller will execute these changes efficiently. For example, oxygen generation may be suspended, thus reducing the water requirement

from the WRS during the repair period (requirement 10). As well, during the repair, the reactive planner will serve as the subsystem level interface to the dynamic controller (requirement 12).

When the repair is complete, the dynamic controller will verify the normal operation of the CDRA and inform the reactive planner, which in turn informs the habitat planner. The habitat planner will adjust the inventory of materials used in the repair and replan if necessary.

A key observation from this scenario is that once anomalous situations are detected, mechanisms kick in at different levels to attempt to contain and compensate for the fault, without having to sacrifice mission goals. For less critical faults of small magnitude, the subsystem controllers can compensate for the change in behavior. At the next level, the system controller may redistribute resources or, if possible reassign some tasks, to keep the system performance and output at different levels. Then the supervisory controller jumps in to determine if it can impose non-critical restrictions to avoid over draining of resources or reduction in effort without significant loss of capabilities. If the problems persist, the reactive planner or the replanner may be invoked to determine new plans. Last, mission control or the crew may want to change some of the mission goals to avoid potential problems. In all of these situations, decisions made at the top take precedence, which imply that the lower level units, especially the lower-level controllers have to change their strategy to satisfy the new requirements. The next section of this paper, presents an overview of a multi-level planning and control architecture that can address these requirements and a brief description of the component modules and their interactions.

A PROPOSED INTEGRATED PLANNING & CONTROL ARCHITECTURE

Given the above requirements list, it should be clear that neither 3T nor the multi-level model-based control architecture alone present the complete solution for long-duration autonomous operations. The former lacks the dynamic models necessary to make efficient use of scarce resources and to maintain smooth transitions among controller states, while the latter lacks domain procedural knowledge to allow the execution of specialized activities, such as fault-recovery as well as a meaningful interface for the user. The information flow and decision sequences in the integrated architecture are shown in Fig. 9.

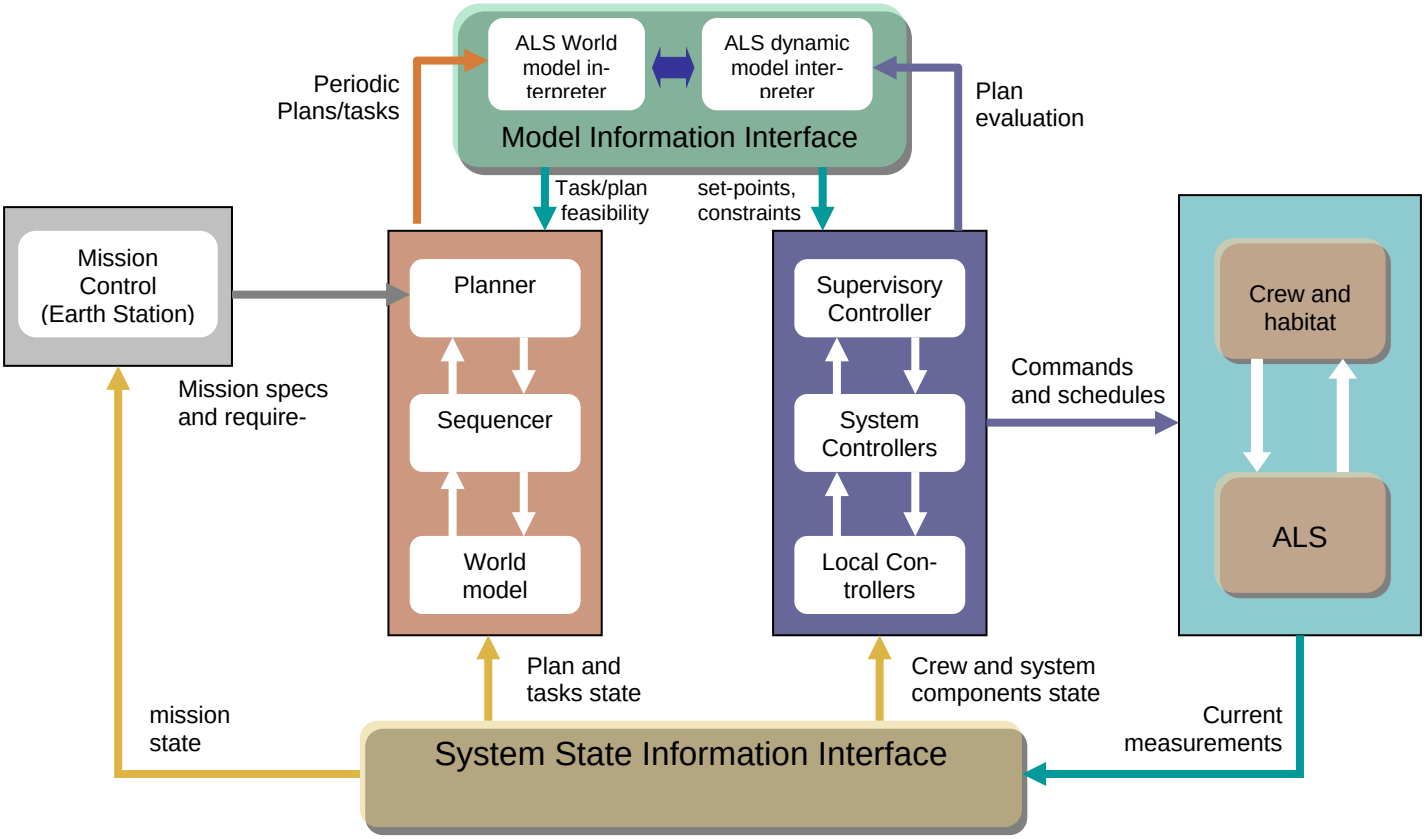


Figure 9: Integrated Planning and Control Architecture

This integrated computational architecture combines the best of these two approaches. Our general design uses the dynamic models of model-based control architecture to inform the state-based procedural schemas during plan development and execution, as well as to carry out the dynamic control of the habitat subsystems. The 3T planner will provide overarching mission plans, while the 3T sequencer can instantiate procedures that cannot be represented by the system-based models of the model-based control architecture.

The 3T planning module drives the supervisory control scheme. Given a top-level goal such as "conduct habitat operations while supporting EVA", the planner automatically generates a habitat plan for a given duration, e.g., the 28-day cycle of the previous section. The planner reasons in depth about goals, resources and timing constraints. It integrates mission goals with *a priori* knowledge such as the crew schedule, EVA schedule, crop plantings and harvesting, resource limitations, etc. This knowledge is stored in the world model. During plan generation, the 3T planner draws form the task-resource consumption model of the Resource Manager (middle level), to take into account the dynamic effects of planning decisions. The resulting plan steps and ordering will be tailored to make the best use of scarce resources. Using the user interface capabilities of the planner, the plan would be reviewed by both mission control and the habitat crew before going into effect.

The middle level of our combined architecture consists of the 3T sequencer working in concert with the model-based supervisory controller. To execute the plan, the planner passes the next step in the plan for each area of the habitat to the 3T sequencer, which decomposes the plan step into RAPs that are further decomposed until the final sequences are at the level of the system controllers in the third level of the architecture, e.g., the WRS or the crop chambers. An example sequence for the ARS was given in the previous section. An example sequence to sustain crop growth might be 1) harvest a wheat crop, 2) harvest a soybean crop, 3) plant a soybean crop, 4) and harvest a salad crop. The selection of RAPs from the RAP library will be guided by dynamic constraints provided by the models in the model based supervisor also in the middle layer. The resulting sequences are then passed to the supervisory controller through the model information interface, which uses them as ordering constraints; e.g., the supervisor may force the ordering of a set of parallel tasks if it will make better use of power at the system level of control, or may adjust the duration of one of the steps as in the previous scenario. Using resource constraints, the supervisory controller transforms the sequence into a schedule of control specifications for the system level controllers, which then carry out the execution of the sequence. Mission controllers and the crew have access to the state of the executing procedures via the system state information access module. This is especially needed when the crew carries out maintenance or ad hoc procedures that do not follow nominal operating schemes.

The system level controllers are designed to control individual habitat systems, such as the Water Recovery System (WRS) and Air Revitalization System (ARS). The system level controller sees each system as an input-output module, where material and energy are input to the system with the goal of producing desired states within the system and output that can be expressed in terms of material, energy, and performance quality parameters. The input-output mappings created by these controllers define utility-based multi-criterion objective functions that the lowest-level subsystem controllers employ to optimize dynamic behavior of subsystems in a way that they minimize the use of resources, while producing the necessary output. For example, given the levels of gases and the amount of energy available to the ARS during the above example sequence period, the system controller for the ARS will regulate the CO_2 and O_2 stores to maximize the CO_2 consumption to support the incineration operations.

Results of the execution from the system controllers are aggregated from the system controllers in the bottom later and provided to the supervisor. The supervisor will update its dynamic models as well as pass the execution results to the sequencer as a set of execution states. The RAPS interpreter has the capability to determine new task sequences when faults occur in the system or in the face of unsuccessful execution of task steps. As RAPs sequence complete, the interpreter informs the planner which will update the plan and pass down the next plan step to be executed. Such an update may simply change start and stop times of steps while maintaining the original ordering. If the RAPs interpreter reports a failure of a plan step, as in the case of the faulty CDRA above, the planner may replan the mission steps, adding or omitting steps depending on the effect of the failed step on the overall mission objectives. As in plan generation, the task resource models of the supervisory controller will inform the replanning. As well, users will be able to modify the plan at their discretion as the crew did in the above scenario by requiring that the EVA take place as originally scheduled.

The principle of "cognizant failure" is still embodied in each level of the architecture. The system controllers provide robust regulation of the habitat subsystems, notifying the middle layer of any failing processes. The supervisory controller dynamically adjusts control schedules as the situation changes, informing the sequencer as to the state of tasks. The sequencer in turn serves as the mechanism to invoke alternate procedures as well as fault recovery procedures. Equally important, in light of severe failure, the sequencer will invoke "safing" procedures for the habitat subsystems, informing the planner which in turn will carry out replanning.

Additionally, the user has access to the levels of control where the aggregate of information and control stratagems is meaningful, and yet the complex details of such things as multi-criterion objectives functions remain hidden.

The architecture just described should be able to operate as in the scenario given in the previous section and thus meet all the major requirements of a next generation integrated monitoring and control system.

CONCLUSIONS

Integrated, autonomous control architectures are an enabling technology for long-duration space missions. An autonomous control architecture can 1) provide for crew autonomy from ground controllers; 2) relieve the

crew from constant, vigilant monitoring of life support equipment; 3) allow for fault diagnosis and recovery so that mission goals can be accomplished even in the face of equipment malfunctions and degradation; and 4) reduce the system mass required to achieve mission goals by optimizing resource and buffer utilization. Achieving these four objectives imposes many requirements on any integrated, autonomous control architecture and we presented a proposed architecture that meets these requirements. We are currently working to implement the proposed architecture and evaluate it with respect to a 90-day simulated lunar mission and will report evaluation results in upcoming publications.

ACKNOWLEDGMENTS

This work was supported in part through grants from the NASA-ALS program (Contract number: NCC 9-159). We acknowledge the help provided by the members of the ALS group at NASA Johnson Space Center (JSC), primarily, Molly Anderson and Wen-Chin Lee, and other members of the Lockheed group at NASA JSC.

REFERENCES

1. NASA Exploration Systems Interim Strategy, NASA HQ Document: NP-2004-07-362-HQ.
2. Duffield, Hanford, and Lange, "Advanced life support requirements document," Tech. Rep. JSC-38571, Rev. B, NASA-Lyndon B. Johnson Space Center Houston, TX, Sept. 2002.
3. Bonasso, R.P., R. J. Firby, E. Gat, D. Kortenkamp, D. Miller and M. Slack, "Experiences with an Architecture for Intelligent, Reactive Agents," *Journal of Experimental and Theoretical Artificial Intelligence*, Vol. 9, No. 1, 1997.
4. Abdelwahed, S., J. Wu, G. Biswas, J. Ramirez and E. J. Manders, "Online Fault Adaptive Control for Efficient Resource Management in Advanced Life Support Systems, Habitation: International Journal for Human Support Research, Vol. 10, No. 2, pp. 105-116, 2005.
5. Lai-fook, K. M. and Ambrose, R. O. 1997. Automation of Bioregenerative Habitats for Space Environments. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2471-2476. Albuquerque, NM: IEEE.
6. Schreckenghost, D., Edeen, M., Bonasso, R. P., and Erickson, J. 1998b. Integrated Control of Product Gas Transfer for Air Revitalization. In *Proceedings of the 28th International Conference on Environmental Systems*. Danvers, MA.
7. Firby, J. R. 1999. *The RAPS Language Manual*, Neodesic, Inc., Chicago.
8. Freed, M., R. Harris, and M. Shafto. Human Interaction Challenges is UAV-based autonomous surveillance. in the AAAI Spring Symposium. 2004. Stanford, CA.
9. Elsaesser, C. and J. Sanborn, "An Architecture for Adversarial Planning," IEEE Transactions on Systems, Man and Cybernetics, Vol. 20, No. 1, pp, 186-194, 1990.
10. Wilkins, D. and K. Myers. A Multiagent Planning Architecture. in Artificial Intelligence Planning Systems. 1998. Pittsburgh, PA.
11. Simmons, R. and Dale, J. 1997. Inter-Process Communication: A Reference Manual. IPC Version 6.0: CMU Robotics Institute.
12. Gat, E. 1998. Three-Layer Architectures. In *Mobile Robots and Artificial Intelligence*, Kortenkamp, D., Bonasso, R. P., and Murphy, R., Eds.: AAAI Press.
13. P. Bonasso, D. Kortenkamp, and C. Thronesbery, "Intelligent control of a water recovery system: Three years in the trenches," AI Magazine, pp. 19–43, 2003.
14. Kortenkamp, D., D. Keirn-Schreckenghost, and R.P. Bonasso. Adjustable Control Autonomy for Manned Space Flight, in IEEE Aerospace Conference. 2000. Big Sky, MT.
15. Hanford, A.J. Transient Modeling Challenge: A Lunar Reference Mission for a 90-day Habitat. NASA JSC Technical Report. Sept. 2004.
16. P. J. Mosterman and G. Biswas, "A theory of discontinuities in dynamic physical systems," Journal of the Franklin Institute, vol. 334B, no. 6, pp. 401–439, 1997.
17. P. Antsaklis, editor. Special Issue on Hybrid Systems, volume 88 of Proceedings of the IEEE, July 2000.
18. G. Biswas, E.J. Manders, J.W. Ramirez, N. Mahadevan, and S. Abdelwahed, "Online Model-Based Diagnosis to Support Autonomous Operation of an Advanced Life Support System," *Habitation: International Journal of Human Support Research,* vol. 10, no. 1, pp. 21-38, 2004.
19. S. Abdelwahed, J. Wu, G. Biswas, J.W. Ramirez, and E.J. Manders, "Online Hierarchical Fault-Adaptive Control for Advanced Life-Support Systems," *Proc. 34th Annual Meeting of Intl. Conf. on Environmental Systems (ICES),* paper number 2004-01-2441, Colorado Springs, CO, July 2004.
20. D. Kortenkamp and S. Bell, "BioSim: An integrated simulation of an advanced life support system for intelligent control research," in Proc. of the 7th Intl. Symposium on Artificial Intelligence, Robotics and Automation in Space, 2003.
21. S. Abdelwahed, J. Wu, G. Biswas, J. Ramirez, and E.J. Manders, "Online Fault-Adaptive Control for Efficient Resource Management in Advanced Life Support Systems," *Habitation: International Journal of Human Support Research,* vol. 10, no.2, pp. 105-115, 2005.
22. M. Morari and J. Lee. Model predictive control: Past, present and future. Computers and Chemical Engineering, 23:667-682, 1999.
23. J. Cury, B. Krogh, and T. Niinomi. Synthesis of supervisory controllers for hybrid systems based on approximating automata. IEEE Trans. Automatic Control, 43(4):564-568, 1998.
24. G. Biswas, G. Simon, N. Mahadevan, S. Narasimhan, J. Ramirez, G. Karsai, "A robust method for hybrid diagnosis of complex systems," 5th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes (SAFEPROCESS), Washington, D.C., pp. 1125-1130, June 2003.

CONTACT

Gautam Biswas, Dept. of EECS and ISIS, Box 1824 Station B, Vanderbilt University, Nashville, TN 37235.Tel: (615)-343-6204; e-mail: gautam.biswas@vanderbilt.edu.