NASA/WP—2013-Unpublished

The Metabolic Rate Advisor:

Using Agents to Integrate Sensors and Legacy Software

William J. Clancey
Ames Research Center, California
and Florida Institute for Human and Machine Cognition, Pensacola

Ron van Hoof
Ames Research Center

January 2013

Acknowledgments

This study was supported in part by the Human Research Project in FY 2007 by JSC-Ames Internal Task Agreement #7948, “EVA Biomedical Advisory Algorithm” and by associated funding from ETDP 12B Human-System Interaction for the Mobile Agents Project (2002-08). ARC Project participants in addition to the authors of this report included Maarten Sierhuis and John Dowding. JSC Project participants included Mike Gernhardt, Larry Kuznetz, Jeff Jones, Andrew Abercromby, Jennifer Jadwick, and Kathy Johnson. This unpublished working report was updated in October 2014 to explain the architecture in more detail in accord with other published NASA reports and to include additional references. The views expressed herein are those of the authors and do not represent the views of other past participants on the project or of NASA or the U.S. Government.

Table of Contents

Introduction

This document describes the technical design of a hardware-software system that could enable crewmembers during surface Extra-Vehicular Activities (EVAs), such as while exploring Mars, to interactively track, understand, and control their physiological performance based on real-time biosensor data. The objective is to increase astronaut self sufficiency given the time-delayed communications that prevent real-time monitoring and advising from Earth to manage workload within resource and safety requirements.

At the start of the project, NASA Johnson Space Center provided the following task description to the Ames Research Center (ARC):

The EVA Physiology, Systems and Performance Project (EPSP) is funding the Intelligent Systems Group at ARC to develop a Laptop Demonstration System of an EVA Biomedical Advisory Algorithm with Speech Recognition Capability. Specific tasks associated with this effort include the following:

1. *ARC shall develop/code a data input module that allows manual data entry of EPSP-specified parameters via keyboard. The module shall also be capable of using EPSP-provided data to create simulated data for each variable and generate a simulated, real-time data stream varying with respect to time.*
2. *ARC shall develop/code a metabolic algorithm module built from the equations embedded in the EPSP Excel metabolic algorithm (LEGACI). Outputs from the equations in this module shall exactly reflect those in the EPSP Excel algorithm.*
3. *ARC shall integrate their iMAS [individual Mobile Agents System] speech recognition system into the EVA Advisory Algorithm Demonstrator. The demonstrator shall allow the EVA crewmember (subject) to query the database for specific data to be defined by the EPSP. The system shall recognize the request, query the database/met algorithm module and provide the information to the crewmember (subject) both audibly and visually on the laptop screen.*

iMAS, described subsequently, is an EVA support system that allows individual explorers who are not connected to a wireless network to log science data and receive navigation and plan advice, as well as biosensor interpretation and alerting—where throughout the explorer and speech system communicate with each other in natural language. iMAS is constructed using an interoperability framework called the Mobile Agents Architecture.

The requirements stated by JSC highlight the use of a speech recognition system, but this is actually just one module that is part of the Metabolic Rate Advisor. Other modules include email, biosensors, GPS, and a camera.

More specifically, the technology of importance here is not the speech system but the multi-agent architecture and "dual-API" method that enables interoperability among legacy software and hardware components. This architecture made it possible to directly integrate the previously developed iMAS system with the Excel implementation of the metabolic algorithm. This approach was faster, more reliable, and more general than developing a new module (item #2). The interoperability architecture allowed as well integrating a path planner for emergency “walk back” advising by relating the terrain to metabolic rate information, which is also discussed in this report.

The Metabolic Rate Advisor was tested in the JSC Partial Gravity Simulator, nicknamed POGO (Figure 1). In this apparatus, an astronaut wears an experimental space suit that is pressurized and constrains his mobility, and especially affects using his hands because of the pressurized gloves. The suited astronaut is suspended by a counterweight is adjusted to simulate the gravity of the moon, Mars, or an asteroid. Motion is highly constricted to a small area, allowing testing of specific actions, such as picking up materials, operating tools, and walking up an inclined plane.

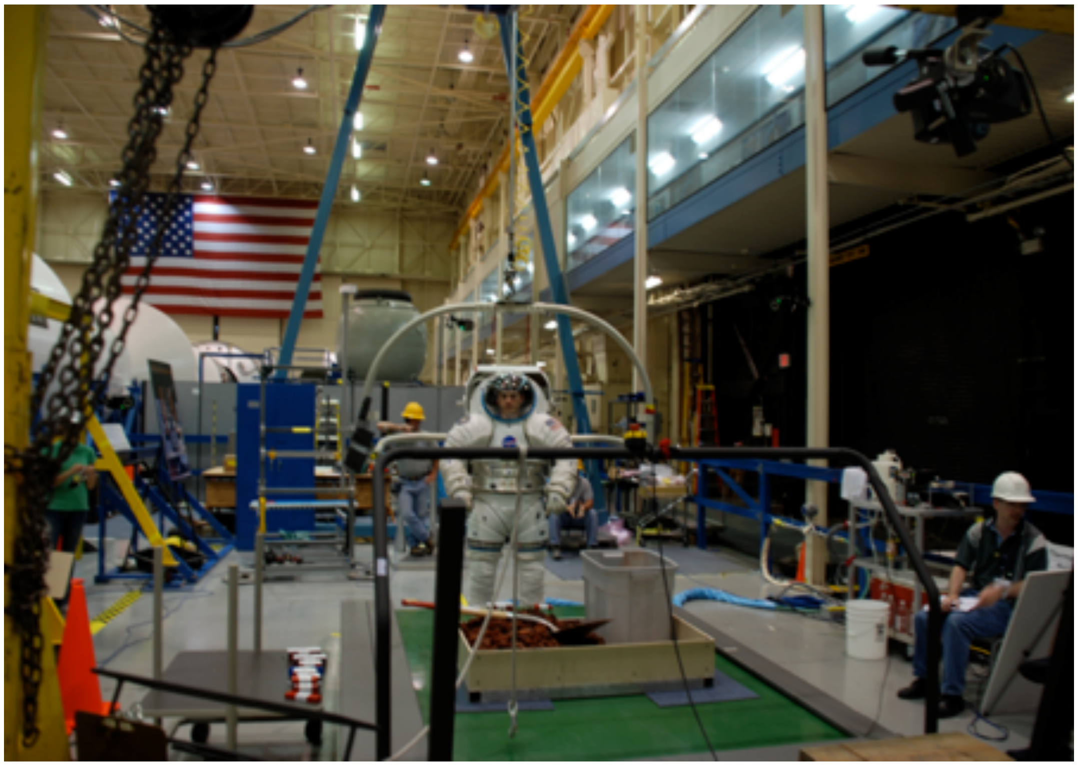

Figure 1. Astronaut in pressurized suit suspended by Partial Gravity Simulator (POGO) at Johnson Space Center (2007)

Note that POGO is a test apparatus primarily designed for experimenting with suit, glove, and tool design. As such it did not permit understanding or experimenting with the Metabolic Rate Advisor (MRA) in an authentic exploration environment as we had demonstrated throughout the development of Mobile Agents (Clancey et al. 2011). POGO enabled testing that the speech recognition system worked in the noisy environment of the pressurized suit, but it did not enable evaluating or refining the monitoring and alerting functionality or natural language interaction in the context of time-extended EVA activities for which the MRA was intended. Further remarks about the limitations of a component-based “test and package” approach to R&D appear at the end of this report.

Technical Architecture and Process Flow

The Metabolic Rate Advisor (MRA) is implemented as reconfigurable system in which “agents” proactively integrate data and command flow among software and hardware components, including interfaces by which people interact with the Advisor. Agents are dedicated software components that operate simultaneously and independently, often on different computers, communicating via messages; they may receive data from and control devices, displays, and other software. The MRA is implemented using the Mobile Agents Architecture (MAA); specifically, the MRA is an adaptation of the existing standalone system called the “individual Mobile Agents System” (iMAS).

Mobile Agents (Clancey et al. 2007) is based on the Brahms (Clancey et al. 1998; Sierhuis 2001) work practice simulation system. In Brahms people, facilities, geography, tools, procedures, communications, etc. are modeled explicitly so circumstantial, sometimes unanticipated interactions can be understood in creating and evaluating work system designs. Brahms has been integrated with other simulation systems and can be used to drive a virtual environment interface (BrahmsVE; Clancey et al. 2005). In Mobile Agents, Brahms agents may run on different computer platforms and communicate wirelessly. These computer systems may be integrated with any variety of software and hardware systems (e.g., robots, cameras, biosensors); the agents are “mobile” because they move with their host computers.

iMAS includes simulated versions of subsystems (e.g., the Biovest, GPS devices, camera) that may be used for developing and testing the agents. The iMAS system is generally run on a laptop computer carried in a backpack and operates without having to be connected to a wireless network or the Internet. In this configuration all science data recorded by the user (e.g., photographs and voice notes) and other EVA logs (e.g., time-stamped locations and biosensor data if any) are stored only in a local database called Compendium. In the networked Mobile Agents configuration, such data are transmitted to the web-interface ScienceOrganizer database and selectively communicated to specialists monitoring the EVA as alerts via e-mail. In a proper Mars EVA simulation such transmissions with Earth would be time-delayed.

Figure 2 shows the MRA’s components and process flow. Refer to the key below the figure for explanations of the figure’s icons such as the meaning of CA. The following is a summary of the process flow:

1. The suit provides raw physiology telemetry to the LEGACI Communications Agent (CA). (It is possible to run the MRA in simulation mode such that these data are provided from a database.)
2. The LEGACI CA stores the telemetry data in the LEGACI Excel spreadsheet and triggers the spreadsheet calculations. The results of the calculations are stored in

the LEGACI spreadsheet. The LEGACI CA reads the data results from the spreadsheet and sends them to the Medical Assistant Agent.[1]

3. The LEGACI CA reads thresholds from a configuration file and sends them via a message to the Medical Assistant Agent.
4. The Medical Assistant Agent processes the data received from the LEGACI CA, determines whether any thresholds are exceeded, and if so generates alerts. These alerts are sent to the Dialog Agent CA as well as the Science Data Collector CA. The SDC CA sends the data to the Science Data Manager CA, which distributes them to the Compendium CA, Console CA, and E-mail CA for storage and distribution.
5. Astronauts can query the advisor for metabolic rate information. These queries are sent to the Dialog System via its CA, which in turn sends corresponding request messages to the LEGACI CA. The LEGACI CA will retrieve the results from the LEGACI spreadsheet and return them to the Dialog System. The Dialog System uses those results to generate the appropriate verbal response.
6. Other functions in iMAS are also available; for example, plans can be loaded from the Compendium CA, sent to the Plan Assistant Agent (via a Plan Manager - not shown here). The Plan Assistant is used to start activities. The Navigation Assistant provides current location information of the person or mobile system/robot using GPS (not shown in the diagram). The Science Data Assistant Agent is used to support the logging of "sample bags," voice notes, and photographs, including their association in the database with the astronaut who created them, the activity in the plan, and time-stamped locations (Clancey et al. 2007).

[1] The term "assistant agent" is somewhat redundant; the term "agent" is appended here to emphasize that the Medical Assistant is implemented as an agent in the Brahms language.

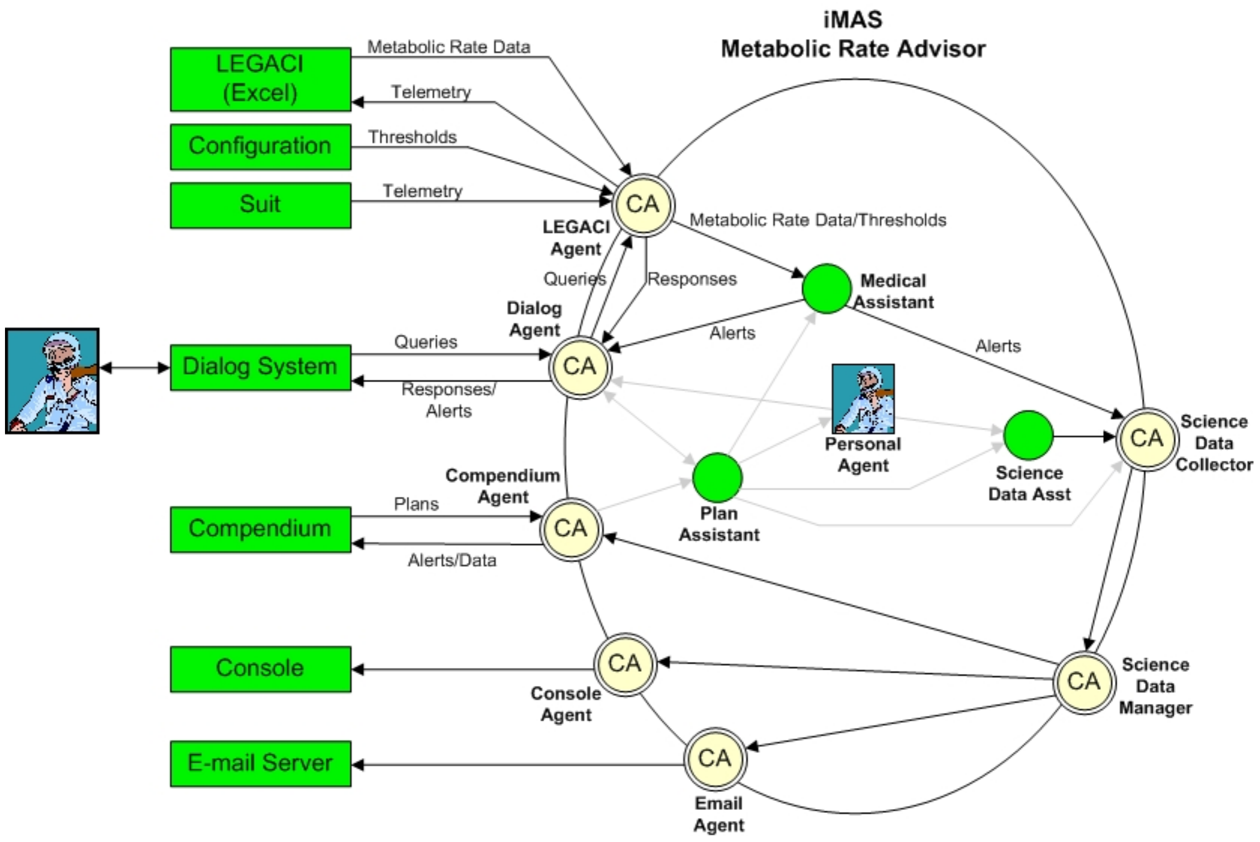

Figure 2: iMAS Metabolic Rate Advisor Architecture and Process Flow (circa 2007)

Green Boxes = External systems (includes the RIALIST Dialog System, the suit telemetry system, Microsoft Excel, the planning program called Compendium, Console, and Email server. In non-POGO configurations, includes a camera, GPS, and ERA robot (Hirsh et al. 2006).

CA = Communication Agent (Comm Agent) = JAVA component that communicates commands and data between an external system (using its API) and other Brahms Agents. Serves as a wrapper that makes the external system function as a Brahms agent, hence the "dual-API" architecture. Translates between the data–control language of the external system and the structured language of the task used by the agents (corresponding to the voice commands of the people). Communication between Brahms agents is via TELL and ASK actions that comprise queries, responses, and alerts, plus other state information.

Green Circles = Standard Brahms Agents that perform functions that coordinate with the Personal Agent of the crew member. In a particular configuration of Mobile Agents each person (crew member, mission support officer) has a personal agent that customizes their interaction with external systems and other people (e.g., it "knows" their plan and current location).

Speech Interface

The Dialog System used in the MRA is RIALIST, a research variant of speech recognition software commercialized by Nuance (Dowding, et al. 2006). RIALIST both recognizes and generates speech. Unlike products such as Dragon Naturally Speaking, RIALIST uses a "semantic unification grammar" that pre-enumerates the (potentially thousands) of utterances that can be recognized. Thus it is not prone to the misunderstandings common in systems like Apple's Siri (iOS 7, circa 2013) that recognize on the basis of individual words or short phrases.

RIALIST is integrated with the Mobile Agents system through the Dialog Agent CA (Figure 2), which transforms structured queries from the astronaut into ASK/TELL actions that are directed to other agents for processing, and then back to the Dialog Agent CA to generate a response.

Most of the utterances recognized by the MRA that are relevant to the metabolic rate, including alternative phrasings, are summarized in Table 1. Alternative phrasings are advantageous for several reasons:

- ❖ Individual pronunciation variation will result in some phrases being more reliably recognized than others for some speakers.
- ❖ A system may be too brittle if there is only one way to speak each command, and the user doesn't remember it.
- ❖ Users decide which phrasings are preferred. As we collect speech data of the system in use, the phrases that get used more frequently are given higher probability and thus become more reliably recognized improving the system's performance. Thus, the system learns from the population of users how they prefer to state requests.

Testing with POGO indicated that the queries in Table 1 were sufficient. In general, somewhat longer phrases are more reliably recognized than shorter phrases. A voice command of a single syllable would be especially difficult to recognize, but very few of the preferred phrasings were that short (despite an initial bias by some of the engineers to use single-word commands rather than sentences).

Table 1: Metabolic Rate Advisor alerts and voice commands with responses

Query Parameter	Units	Verbal Command	Verbal Response	Alert Threshold	Alert Response
Average Metabolic Rate*	btu/hr	met rate metabolic rate query met rate what is my met rate	"x btu/hr"	3000 btu/hr	"3000 Met"
Scheduled time to repress	minute s	time left how much time is left how much time is remaining query time left	"x minutes"	30	"30 minutes"
Limiting consumable,	minute s	lim con limiting consumables	"x Consumable	AR=30	"LimCon30"

Query Parameter	Units	Verbal Command	Verbal Response	Alert Threshold	Alert Response
time left*		query consumable what are my limiting consumables?	, x minutes”		
Heart Rate	Beat/ min	heart rate query heart rate what is my heart rate	“x bpm”	175 bpm	“175 bpm”
Consumables Utilization Efficiency (Planned vs. Actual)*	percen t	red line consumable usage query red line what is my consumable usage	“% consumable used, planned, x lbs vs actual, y lbs”		
Power Remaining*	watt- hrs	power power remaing query power query power remaining how much power is remaining	“x minutes power”	AR=30	“Power 30”
LiOH Remaining*	minute s	Scrubber Query scrubber	“x minutes scrubber”	AQ=30	“Scrubber 30”
O2 Remaining*	minute s	oh two query oh two oxygen query oxygen how much oh two is left what is my oh two	“x minutes O2”	AJ=30	“O2 30”
Feedwater Remaining*	minute s	feed water query feed water how much feed water is left what is my feed water	“x minutes feedwater”	AH=30	“Feedwater 30”
O2 Pressure	psi	oh two press oh two pressure oxygen press oxygen pressure query oh two pressure query oxygen pressure query oh two press query oxygen press what is my oh two pressure	“x psi”	85 psi (10%)	“O2press”
Walkback Range*	km	hab how far is the hab where is the hab habitat how far is the habitat where is the habitat	“x kilometers”	10 km	“Ten K”
Environmental heat load	btu/hr	heat leak query heat leak what is my heat leak	“x btu/hr”	300	“Heat leak”
Suit Leak Rate*	lb/min	suit leak query suit leak what is the suit leak	“x lb/min”	0.01 (10*spec)	“Suit leak”
Heat Storage*	btu	storage query storage heat storage query heat storage	“x btu”	BT=300 btu	“Overheat”

Query Parameter	Units	Verbal Command	Verbal Response	Alert Threshold	Alert Response
		what is my heat storage			
LCVG T inlet advisory	deg F	t in query t in what is my temperature in	" x deg F actual " y deg F advised	ABS(G-BZ) .GT 2***	"Colder or Warmer"
Dehydration advisory	lbs	sweat query sweat what is my sweat	"x lbs"	0.5, 1, 1.5, 2	"Drink"
Nutrition advisory	kcal	kay kal nutituion what is my k cal	"x kcals"	500, 1000, 2000	"Eat"
Exertion alert	pC02, HR			C=175, E=15	overexert
Acute overheat alert	btu/hr			P=4000	slow down

Table 2. Selected additional voice commands available in MRA from previous versions of Mobile Agents

Query/Command	Verbal Command
Stop talking	Stop talking please Shut up Stop talking
Change volume	Increase volume Decrease volume
Repeat	Say that again Repeat that
Current Time	What time is it?
Acknowledge	Acknowledge Hello
Start Activity	Start first activity Start next activity
Query activity	What is my current activity? What is my next activity?
Record voice note	Record a voice note Create a voice note Take a voice note
Create sample	Create sample bag
Play voice note	Play that voice note Play voice note <N>
Entertainment	Tell me a joke Play a song by <N>

Integration with EVA Path Planner

The MRA was extended subsequently by an MIT masters student, Aaron Johnson, in a dissertation program called SEXTANT (Johnson 2010; Johnson et al. 2010). As shown in Figure 3, an additional external system was included, a program that plans the EVA route using terrain data. An additional CA enables communication between the route planner (using its API) and other agents. This provided access to suit data, the LEGACI model, the Dialog System, etc. such that paths could be planned with respect to the original activity plan, terrain, and the astronaut's physiological status (Johnson et al. 2009).

The SEXTANT project illustrates how the Mobile Agents Architecture enables interoperability of new subsystems that can incorporate and augment capabilities provided by existing agents. For example, SEXTANT received access to the astronaut's location, plan, and metabolic rate information directly, and was able to used the Dialog System to communicate with the astronaut. In such integration, design decisions need to made of course about the flow of information among the agents, especially which agent(s) are responsible for proactively monitoring and generating alerts. Other agents passively transform and log data, and communicate when requested.

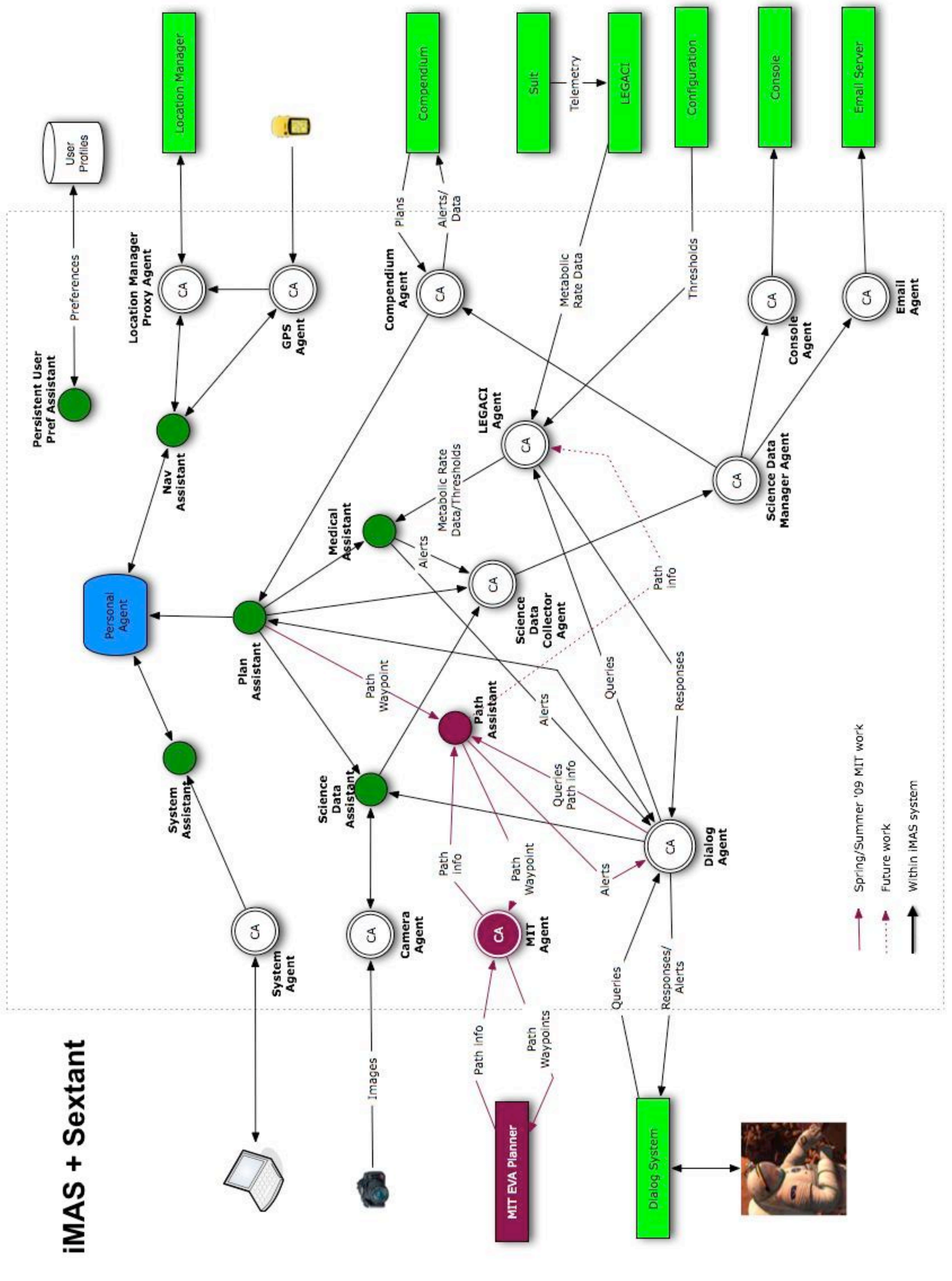

Figure 3. Metabolic Rate Advisor combined with Sextant (EVA Path Planner and its Communication agent). Diagram shows configuration as of 2009, with integration to Mobile Agents Plan Assistant Agent but not yet to the LEGACI data.

Discussion of R&D Perspectives and Methodology

On striking lesson we learned during the Mobile Agents project is that conceiving, designing, developing, and experimenting with interoperating networked subsystems (e.g. a robot, space suit, camera, science database) is fundamentally different from the "breadboard demonstrator" approach formalized by NASA engineers in the 1970s. By that methodology, R&D went through many phases of component development and laboratory testing, called Technology Readiness Levels (2014), whereby deploying a system in the field—a space flight—was the final step. Table 3 shows one version of the transitions from TRL 3, studies and focused designs/experiments, to TRL 7, the operational space environment.

Table 3. Traditional Technology Readiness Levels—from start of R&D through first prototype in a operational environment (from Wikipedia, 2014; emphasis added).

3. Analytical and experimental critical function and/or characteristic proof of concept	At this step in the maturation process, active research and development (R&D) is initiated. This must include both analytical studies to set the technology into an appropriate context and *laboratory-based studies to physically validate that the analytical predictions are correct.*
4. Component and/or breadboard validation in laboratory environment	Following successful "proof-of-concept" work, basic technological elements must be integrated to establish that the "pieces" will work together to achieve *concept-enabling levels of performance for a component and/or breadboard*. This validation must be devised to support the concept that was formulated earlier, and should also be consistent with the requirements of potential system applications. *The validation is "low-fidelity" compared to the eventual system: it could be composed of ad hoc discrete components in a laboratory.*
5. Component and/or breadboard validation in relevant environment	At this level, the fidelity of the component and/or breadboard being tested has to increase significantly. *The basic technological elements must be integrated with reasonably realistic supporting elements so that the total applications (component-level, sub-system level, or system-level) can be tested in a 'simulated' or somewhat realistic environment.*
6. System/subsystem model or prototype demonstration in a relevant environment (ground or space)	At TRL 6, a *representative model or prototype system* or system - which would go well beyond ad hoc, 'patch-cord' or discrete component level breadboarding - would be *tested in a relevant environment.* At this level, if the only 'relevant environment' is the environment of space, then the model/prototype must be demonstrated in space.
7. System prototype demonstration in a space environment	TRL 7 is a significant step beyond TRL 6, requiring *an actual system prototype demonstration in a space environment.* The prototype should be near or at the scale of the planned operational system and *the demonstration must take place in space.*

Notice the emphasis on laboratory testing for levels 3 to 5. In the Mobile Agents project (Clancey et al. 2011), these steps were repeated each year and required about six months. TRL 6, "testing in a relevant environment" occurred in the field tests at MDRS and DRATS; and emphatically, these were *experiments* as part of requirements analysis, not *tests*. Arguably, insofar as using systems like iMAS to survey lava flows in Hawaii, Idaho, and New Mexico constituted the actual work of the geologists, these were also TRL 7, demonstrations of practical use in operational environments. Indeed, TRL 7 is only possible in Earth analog environments for many complex interactive technologies today. As exemplified by the "sky crane" of the Mars Science Laboratory, the first demonstrations in true operational environments can only occur when the spacecraft operates during the space mission itself.

Today the combination and availability of the Internet, wireless communication, object-oriented message-passing distributed architectures (e.g., agent systems), and a variety of digital peripherals (sensors, cameras, robotic devices etc.)— all available in portable "supercomputer" packages—has fundamentally changed the complexity of systems that can be built and how they are developed. Integrated systems can be prototyped from the ground up (i.e., TRL 4) as portable, end-to-end systems (Clancey et al. 2007; 2011) and thus can be used experimentally in authentic work contexts. Also, system development can be iterative, cycling many times through the phases of requirements specification, design, development, testing, and experimentation/evaluation. In particular, Earth locations that are analogs of the moon or Mars, such as Devon Island in the Canadian Arctic, have enabled field scientists and computer scientists to carry out "analog missions" using prototype systems experimentally over multiple field seasons.

The NASA life support systems engineers who worked on the Metabolic Rate Advisor project and were not part of the Mobile Agents team focused on the POGO test environment (Figure 1) as if it embodied the requirements for a metabolic advisor. The objective became packaging a "flight ready" system, rather than better understanding the requirements for the MRA and discovering how new mission operations capabilities might be enabled by new kinds of technology. In effect, the Ames Mobile Agents engineers were engaged in a requirements discovery and technology invention project, while the JSC life support systems engineers, strongly focusing on the space suit as a product, were engaged in a packaging project that presupposed the MRA vocabulary circa 2009 (Table 1) was final.

Consequently, the MRA vocabulary was treated as if it were the only voice interaction that would occur between astronaut and subsystems, without considering how the MRA capabilities might need to interact with other components. This specific "life support system" focus was in stark contrast to our experience in developing the Mobile Agents systems (Clancey et al. 2007). The original requirements for the Mobile Agents voice commanding were derived by analyzing interactions between CapCom and the Apollo astronauts (Clancey 2004). Through iterative experimentation from 2002–2008 with nine fielded system configuration prototypes, we identified eleven categories of voice commands supporting an EVA (e.g., navigation, plan management, science data logging) including 134 voice-commanded "workflow capabilities" (Clancey et al. 2011; Clancey & Lowry 2012; Clancey et al. 2012b)—and none of the recognizable commands were fewer than three words (compare to Table 1).

Confining the astronaut in the JSC POGO test harness restricted his activity. Exploring was impossible and there was nothing to explore. The MRA test harness had a "work station," but tests focused on the mobility in the suit, ability to grasp and manipulate tools, and associated metabolic rate for different positions of bending, grabbing materials, etc. It was not possible to walk to other sites, so there was no site planning or navigation functionality required, no logging of observations at different sites using a variety of cameras and other instruments, and no robotic assistant to be commanded. For example, during Apollo EVAs astronauts had difficulty relating craters to their maps and finding a route (Clancey 2004).

From the start, the MRA was conceived by the life support system engineers as the "audio component of LEGACI" or "the speech system." But this would be like calling a laptop computer "an LCD display system." By focusing on a single integration—*ARC shall integrate their iMAS speech recognition system into the EVA Advisory Algorithm Demonstrator*—the agent system was invisible (as it should be), and thus the open architecture and interoperability it provided were not appreciated.

In part, the JSC test and package perspective reflects the 1960s mental model that the product of research was a packaged flight-ready system. It was inconceivable in the 1960s that the software might not be written or tested prior to launch (as is common on planetary missions today). Software of the day was more coupled to hardware "registers" and "commands," and if updating were possible it would be risky: The TRLs required that the entire system must be built and operational before flight.

Integration of early hardware was more likely to occur through soldered connections than programs. The flexibility of today's systems—both for reconfiguration and adaption during operation—has mostly enabled through the modularity afforded by object-oriented programming and the adaptive capability of model-based programming (also known in AI research as "symbolic programming" or "rule-based systems").

The MRA was conceived by the JSC engineers who developed spacesuits as being a self-contained subsystem (i.e., suit/sensors + LEGACI interpreting software + voice interface) that would be packaged with the life support system. That is, MRA would be the entire voice commanding system that would be used for EVAs, a system that focused exclusively on metabolic rate advising and no other EVA concerns. They called the system they packaged Violet, following perhaps from the phrase "Voice Interface for Operations…." (Kuznetz 2008; Mackin et al. 2010).[2]

Because metabolic rate inquires could be reduced to one word or short phrases (e.g., "query storage") rather than actual sentences used in field tests with geologists who were actually exploring and documenting a site of interest and new to them (Clancey et al. 2007), the engineers viewed most of the MRA architecture and Dialog System as

[2] Mackin (2010) builds on the Metabolic Rate Advisor, referring to "the user interface implemented by Kuznetz," with no reference to the original Ames project.

superfluous. They determined that a off-the-shelf voice recognizer developed for automobiles would be sufficient for Violet, abandoning the RIALIST subsystem that allowed full sentences, grammar-based recognition, and alternative phrasings. Whereas RIALIST in the Mobile Agents EVA configuration incorporated a grammar with over one hundred rules enabling thousands of questions or statements averaging X words (with response time usually less than two seconds), Violet reduced the MRA to seventeen one or two word commands. Whereas RIALIST at MRA built on and incorporated all of the EVA grammar— enabling the same system to be taken directly to the field with a robot, habitat, cameras, etc.—Violet would only be able to provide data for 17 metabolic parameters. Without these other subsystems, there was no need for the flexibility of the agent "plug and play" architecture, so it was discarded. Consequently, future extensions rather than being reconfigurable modules would need to be hardwired in code.

Perhaps because of its legacy and traditional engineering methods, systems developed by JSC for field tests are packaged resemble flight-ready hardware. For example, see the Space Exploration Vehicle (2014) and the gold-plated Robonaut (2014). Packaging for public appearance is viewed as part of the R&D process; it is part of the JSC culture. For some systems like the SEV, which appeared in the US President's inauguration parade in January 2009, there are clear benefits for stimulating public interest. For R&D involving EVA support systems such as the MRA, the packaging process leads to far too early concerns about weight, power, and miniaturization that require reducing system capabilities—when the entire effort should focus on what might be possible in ten or twenty years. Imagine designing communications for Mars in the 1980s using early PCs with floppy disks and less than 1 MB RAM, before the advent of GPS, cell phones, wireless networks, web browsers, etc. The emphasis of mission support systems R&D should instead be on requirements analysis and inventing new technologies, particularly in analog environments.

The irony of "packaging for flight" is that in 2007 we were decades from having boots on the ground of Mars or an asteroid where the MRA might be used. During that time much will change in computing technology, particularly in voice commanding; a great deal of reconfiguration and experimentation will be possible and desirable. Thus, the reduction of the MRA to Violet reduced capabilities of the system as well as its ability to be reconfigured and extended in ongoing research, which had been inherent in the original Mobile Agents architecture.

In fact, putting Mobile Agents into NASA mission operations does not require such stripping down, as is proven by the OCAMS system (Clancey et al. 2008). Implemented in the Mobile Agent Architecture (though with the Brahms agents "compiled" into Java modules), this system currently automates all routine file transfers between the International Space Station computers and ground support teams.

In summary, our experience developing the MRA revealed that a component-based "develop, test, package" approach does not exploit the capabilities of today's system architectures and affordable mobile systems; nor does it allow exploring what might be possible and useful in practice by an incremental, rapid prototyping methodology.

Premature concerns with packaging may affect R&D in other contexts, such as developing something like the MRA for a "smart watch" that monitors the physiology of diabetes patients. The following pitfalls can inhibit the creative process in designing technology:

1) **Requirements:** Designing for a testbed as opposed for an end-to-end system prototype in an authentic work context
2) **System Architecture:** Viewing each subsystem (e.g., a robot) as a separately designed, developed, tested, and packaged component instead of developing integrated prototypes for early-in-design field experimentation.
3) **Development:** Grounding design in automation functions (e.g., alerting using parameter thresholds) as opposed to human activities (e.g., helping the astronaut replan an EVA)
4) **Packaging:** Preparing the test system for public demonstrations and beginning the process of "hardening" it for flight instead of retaining system architectures that allow substitution or integration with more advanced technologies.

In contrast with the component-based R&D approach, "human-centered computing" begins with authentic work settings (e.g., Mars analog sites) to understand how people work, including their chronological activities, behaviors, perceptions, conception of situations, communications, movements, etc. From this understanding one develops tools and protocols incrementally that fit the physical, conceptual, and social work context. The work system is collaboratively designed as whole through experience in operational settings.

References

Clancey, W. J., Sachs, P., Sierhuis, M., and van Hoof, R. 1998. Brahms: Simulating practice for work systems design. *International Journal of Human-Computer Studies*, 49: 831-865.

Clancey, W. J. 2004. Roles for agent assistants in field science: Understanding personal projects and collaboration, *IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews*, 34 (2) 125-137. Special Issue on Human-Robot Interaction, May.

Clancey, W. J., Sierhuis, M., Damer, B., Brodsky, B. 2005. Cognitive modeling of social behaviors. In R. Sun (Ed.), *Cognition and Multi-Agent Interaction: From Cognitive Modeling to Social Simulation*, pp. 151-184. New York: Cambridge University Press.

Clancey, W.J., Sierhuis, M., Alena, R., Berrios, D., Dowding, J., Graham, J.S., Tyree, K.S., Hirsh, R.L., Garry, W.B., Semple, A., Buckingham Shum, S. J., Shadbolt, N. and Rupert, S. 2007. *Automating CapCom using Mobile Agents and Robotic Assistants.* NASA Technical Publication 2007-214554. Washington, D.C.

Clancey, W.J., Sierhuis, M., Seah, C., Buckley, C., Reynolds, F., Hall, T., Scott, M. 2008. Multi-agent simulation to implementation: A practical engineering methodology for designing space flight operations. In A. Artikis, G. O'Hare, K.

Stathis, & G. Vouros (Eds.), *Engineering Societies in the Agents' World VIII.* Athens, Greece, October 2007. Lecture Notes in Computer Science Series, Volume 4870. Heidelberg Germany: Springer, pp. 108-123.

Clancey, W. J., Lowry, M., Nado, R., Sierhuis, M. 2011. Software productivity of field experiments using the Mobile Agents open architecture with workflow interoperability, *IEEE Space Mission Challenges for Information Technology*, August 2011, Palo Alto, pp. 85-92.

Clancey, W. J. and Lowry, M. 2012. Lunar Surface Systems Software Architecture Study: Interoperability. NASA/TP—2012–216040. Available: http://ti.arc.nasa.gov/publications/

Clancey, W. J., Nado, R., van Hoof, R., Sierhuis, M., Jones, G., Dvorak, D. 2012. Lunar Surface Systems Software Architecture Study: Open Architecture. NASA/TP—2012–216041. Available: http://ti.arc.nasa.gov/publications/

Dowding, J., Alena, R., Clancey, W. J., Graham, J., and Sierhuis, M. 2006. Are you talking to Me? Dialogue Systems Supporting Mixed Teams of Humans and Robots. *AAAI Fall Symposium 2006: Aurally Informed Performance: Integrating Machine Listening and Auditory Presentation in Robotic Systems*, October, Washington, DC.

Hirsh, R., Graham, J., Tyree, K., Sierhuis, M., and Clancey, W. J. 2006. Intelligence for human-assistant planetary surface robots. In A. M. Howard and E. W. Tunstel (Eds.), *Intelligence for Space Robotics*. Albuquerque: TSI Press, 2006, pp. 261-279.

Johnson, A. W., Newman, D. J., Waldie, J. M., Hoffman, J. A. 2009. An EVA mission planning tool based on metabolic cost optimization, SAE 2009-01-2562, *39th International Conference on Environmental Systems*, Savannah, GA, 12-16 July 2009.

Johnson, A. W., Hoffman, J. M., Newman, D. J., Mazarico, E., and Zuber, M. 2010. An Integrated Traverse Planner and Analysis Tool for Planetary Exploration. *AIAA SPACE 2010 Conference & Exposition*, Anaheim, California, AAAI 2010–8829.

Johnson, A.W. 2010. *An Integrated EVA Mission Planner for Future Planetary Exploration*, M.S. thesis, Massachusetts Institute of Technology

Kuznetz, L.H. 2008. "Designing a smart suit for the moon and mars." 38th International Conference on Environmental Systems: SAE 2008-01-1952.

Robonaut. (2014, August 19). In Wikipedia, The Free Encyclopedia. Retrieved 00:02, October 9, 2014, from http://en.wikipedia.org/w/index.php?title=Robonaut&oldid=621949466

Sierhuis, M. 2001. *Modeling and Simulating Work Practice*. Ph.D. thesis, Social Science and Informatics (SWI), University of Amsterdam, SIKS Dissertation Series No. 2001-10, Amsterdam, The Netherlands, ISBN 90-6464-849-2.

Space Exploration Vehicle. (2014, July 3). In Wikipedia, The Free Encyclopedia. Retrieved 00:03, October 9, 2014, from http://en.wikipedia.org/w/index.php?title=Space_Exploration_Vehicle&oldid=615461460

Technology readiness level. (2014, September 26). In Wikipedia, The Free Encyclopedia. Retrieved 00:00, October 9, 2014, from http://en.wikipedia.org/w/index.php?title=Technology_readiness_level&oldid=627180805