

# Multi-agent Simulation to Implementation: A Practical Engineering Methodology for Designing Space Flight Operations

William J. Clancey<sup>1</sup>, Maarten Sierhuis<sup>2</sup>, Chin Seah<sup>3</sup>, Chris Buckley<sup>4</sup>,  
Fisher Reynolds<sup>4</sup>, Tim Hall<sup>5</sup>, and Mike Scott<sup>3</sup>

<sup>1</sup> NASA Ames Research Center, Intelligent Systems Division, Moffett Field,  
CA 94035, USA

<sup>2</sup> RIACS, NASA Ames Research Center

<sup>3</sup> QSS, NASA Ames Research Center

<sup>4</sup> USA, NASA Johnson Space Center

<sup>5</sup> NASA Johnson Space Center

{William.J.Clancey, Maarten.Sierhuis-1, Christopher.B.Buckley,  
f.f.reynolds, Timothy.A.Hall}@NASA.Gov, CSeah@mail.arc.nasa.gov,  
mscott@ptolemy.arc.nasa.gov

**Abstract.** OCAMS is a practical engineering application of multi-agent systems technology, involving redesign of the tools and practices in a complex, distributed system. OCAMS is designed to assist flight controllers in managing interactions with the file system onboard the International Space Station. The “simulation to implementation” development methodology combines ethnography, participatory design, multiagent simulation, and agent-based systems integration. We describe the model of existing operations and how it was converted into a future operations simulation that embeds a multiagent tool that automates part of the work. This hybrid simulation flexibly combines actual and simulated systems (e.g., mail) and objects (e.g., files) with simulated people, and is validated with actual data. A middleware infrastructure for agent societies is thus demonstrated in which agents are used to link arbitrary hardware and software systems to distributed teams of people on earth and in space—the first step in developing an interplanetary multiagent system.

**Keywords:** Work Systems Design, Work Practice Simulation, Decision Support System, Multi-Agent System, Agent-based Systems Integration, Space Flight Operations.

## 1 Introduction

OCAMS (Orbital Communications Adapter Mirroring System) is a practical engineering application of multi-agent systems technology, using the Brahms modeling and simulation tool [1-6], involving redesign of the tools and practices in a complex, distributed system. The purpose of the project was to automate some of the tasks involved in mission operations at the Mission Control Center (MCC) at NASA Johnson Space Center (JSC), supporting the International Space Station (ISS).

The combination of people and systems involved in JSC mission operations support is complex and distributed [7]. When long-term complex programs like ISS evolve, new systems and processes are introduced that interact with legacy systems. This creates a growing distributed systems environment that can be taxing on the flight controllers and introduce more risk of human error. The OCAMS solution bridges these complex distributed systems and automates processes that are repetitive and time consuming. This simplification helps improve operator productivity and safety and reliability by reducing the chances of human error.

The OCAMS tool is complex because it is embedded in the infrastructure of geographically and temporally distributed people and systems for which it facilitates communication:

- 1) **People and Organizations:** Flight controllers, “backroom” support teams, and customers (planners, human factors specialists, etc.) located in different rooms at MCC and at other NASA centers in other states; and the crew on-board ISS.
- 2) **Computer Systems:** File servers, PCs communicating with the ISS, support PC, the PC that mirrors the ISS file directories (MirrorLAN), and PC laptops onboard ISS.
- 3) **Communication Media:** Voice communications system (“voice loop”) at MCC, telephone, email, “flight notes,” “change requests,” and log documents.
- 4) **Space Communication Network:** Communication between ground and ISS using the TDRS satellite system is short-term, periodic and irregular (from human perspective).
- 5) **Out of this world geographic distribution:**
  - a) Multiple NASA centers
  - b) In Houston: Highly secure Mission Control Center with flight control rooms and support backrooms; offices in different buildings at JSC
  - c) International partners’ (Russia, Europe, Japan) control rooms and offices
  - d) ISS orbiting earth about every 90 minutes.
- 6) **Regulations relating to safety and control have over time produced disconnected, legacy systems:**
  - a) no ISS link to earth’s Internet
  - b) no cell phones in MCC
  - c) no network connection between MirrorLAN and MCC file servers
  - d) multiple versions of operating systems and file generation and handling programs at different NASA centers and onboard ISS.
- 7) **Work Practices and Protocols:**
  - a) Diversity of methods for delivering files, notifying support personnel (called “officers”) of work to be done (see Communication Media), and notifying customers (usually “flight controllers”) of completed tasks and problems
  - b) Continuous 24-7 coverage in three shifts (called “orbits”)
  - c) Shift handovers relying on detailed logs created manually documenting the work done on the previous shift, anticipated work, and ongoing issues.

The OCAMS tool was designed in a collaboration between two NASA centers, JSC (an operations center) and Ames Research Center (ARC). The objectives included: 1) Developing new mission operations design and automation capabilities that would reduce the need for ISS ground support on a 24-7 schedule, and 2) Shifting MCC's concept of operations from controlling systems directly onboard the ISS to supporting astronauts living and working in space.

The project approach was to automate tasks to improve operator productivity, increase accuracy of the process, and eventually enable consolidation of this position into other console disciplines. A year was allowed to demonstrate a new methodology and automation capability, in which we would use a multiagent system to simulate and implement an automation tool. This paper describes the methodology and presents the results of the project in the first half year, including partial implementation of a prototype tool within a simulation of the new work system (called a "future operations simulation").

More broadly, in terms of engineering agent societies, this project illustrates the following themes:

**1) Highly-interdisciplinary methodology for the engineering of complex distributed applications**

- a) Ethnography
- b) Mission Operations (flight controllers & protocols)
- c) IT Administration: tools and constraints (OCA—Orbital Communications Adapter wireless card, servers, FTP, email, multiple networks, security, file types, mirroring, GUI, agents)
- d) Brahms: Work Practice Simulation
- e) MA: Agent-based Systems Integration
- f) Java platform

**2) Analysis, Design, Development & Verification of Agent System**

The simulation to implementation methodology enables dealing with complexity by using a simulation to design and largely implement a tool that is integrated with a simulation of the work setting and practices:

- a) *The future tool is embedded in the Future Operations Simulation*
  - i) Simulated people (Brahms agents)
  - ii) Actual software agents (e.g., personal agents)
  - iii) Actual external systems (e.g., email, FTP, file system, office tools)
  - iv) External system APIs used by Brahms Communication (COM) Agents
- b) *The work system design and tool is tested with actual data*
  - i) Simulation is driven by the same inputs used by the future tool
  - ii) Develop using part of the data set (e.g., a month's input)
  - iii) Continue to test during the implementation phase by using new data as it becomes available.

**3) Middleware infrastructures for agent societies: Use of Brahms agents and external system APIs to link arbitrary hardware/software to teams of people**

Brahms provides a promising candidate for answering the question: How will we build practical complex agent systems on a variety of platforms using arbitrary external software and hardware devices?

This paper describes the OCAMS project's origin and scope, the Current Operations Model and simulation output and the consequential design and partial implementation of a multiagent tool (OCAMS) that automates operations in what we call the Future Operations Simulation. Conclusions review how the development of OCAMS provides methods and insights for engineering agent societies.

## 2 Simulation to Implementation Approach

Our methodology makes multiagent simulation of work practices an integral part of creating agent software, an approach we call "simulation to implementation" (Fig. 1). The approach starts with the creation of a *Current Work Practice Simulation Model* (CSM), using the Brahms language. The purpose of this simulation is to help frame an organization's problems and prioritize relationships and trade-offs. For example, how will the functionality of NASA's new spacecraft vehicle—the CEV—impact mission operations, and how will the vehicle to ground split in functionality impact communications and in turn performance? Framing the problem to be addressed, metrics and scenarios are developed to create a work practice model in Brahms. Simulating the model in Brahms will generate simulation data that can be used to interpret the outcome and validate the answers to the framing questions. One important additional aspect is the use of such a simulation to generate new ideas for formulating the problem, and ultimately identifying solutions to the problems that can be addressed in the next design phase. Methods used in this phase are work practice observation (including videotaping and still photography), collaborative modeling with the workers from the organization, and interview techniques [8].

In the *Participatory Design of Future Work System* phase we work closely with the workers from the organization to design a solution to the problems identified in the current work practice simulation [9]. In this phase we generate user-driven requirements and turn these requirements into a functional and technical design of a multiagent workflow tool. Following a principle of participatory design—transforming current practices rather than believing one can start from scratch—leads to the *Future Work System Simulation Model* (FSM) phase, in which the CSM model is adapted to include the proposed tool(s). The data, metrics, and scenarios from Phase 1 are used to drive the future work simulation, allowing comparison of the CSM with the FSM models and validating the improvements of the new design. Because the future tool is embodied in the FSM model, from the workers' perspective it is actually a prototype tool that runs in an automatic mode driven by historical data, simulating human actions. By providing interactive control of the simulation in a prototype GUI, the tool's operation can be demonstrated and its automatic features inspected and hence refined.

In the *Work System Implementation* phase we transform the Brahms FSM model into a distributed real-time multi-agent system (MAS). The Brahms simulation engine in runtime mode will shift the discrete event simulation from being driven by an internal clock to being coupled to the time and events in the real world, thus transforming simulated agents into real-time software agents. Brahms can both simulate or execute its agent models over the internet, enabling a seamless transformation from an agent-based simulation environment to a distributed multi-agent system environment.

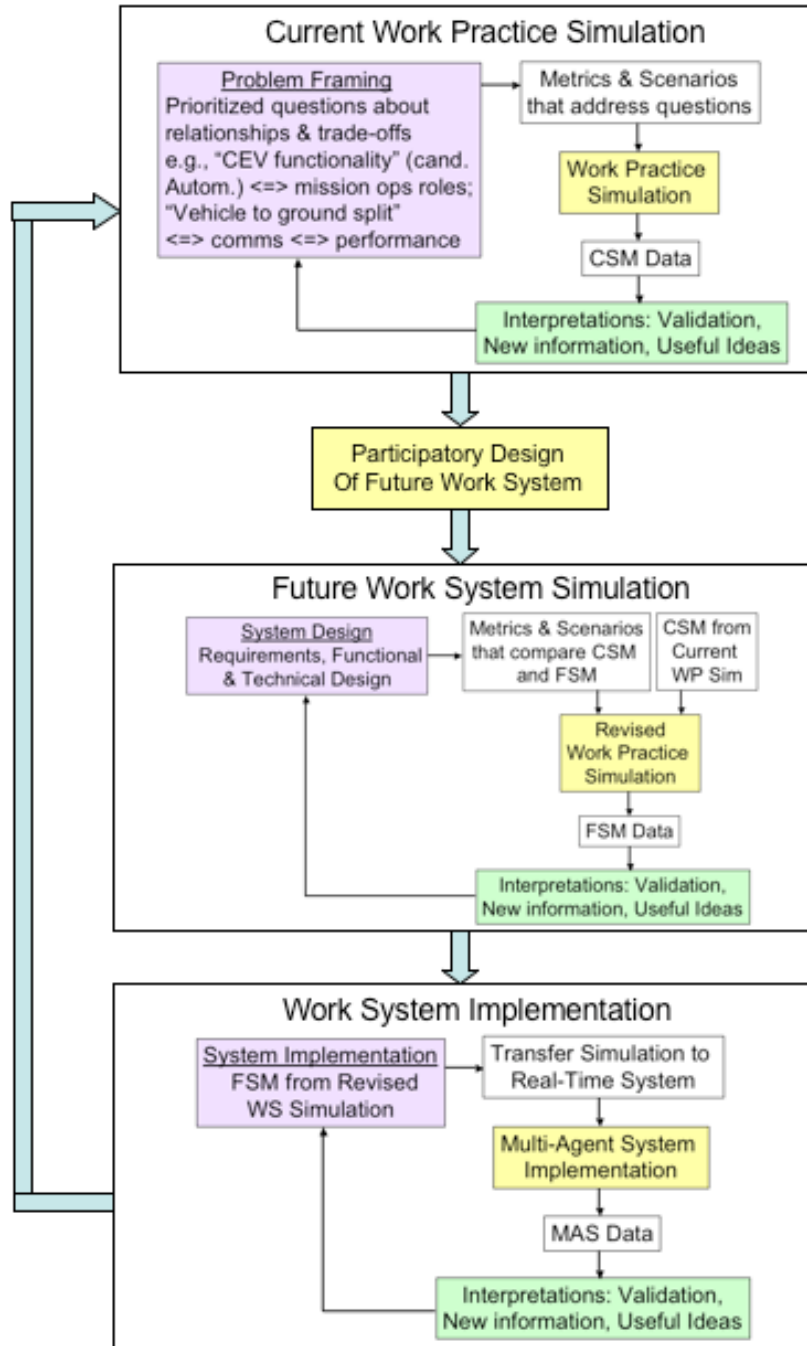


Fig. 1. Simulation to Implementation Approach

### 3 Project Origin and Scope

One purpose of this project was to demonstrate the use of an agent-based simulation-to-implementation methodology in Mission Control at NASA's Johnson Space Center. Program management chose the Orbital Communications Adapter (OCA) backroom group, which provides file transfer support to the ground team and astronauts onboard the ISS.

Applying the simulation-to-implementation methodology to the OCA setting involved the following activities:

- 1) Observation of OCA operations and interviews with OCA officers
- 2) Creation of a baseline simulation of current operations using Brahms
- 3) Collaborative redesign of the work system (documented in a functional specification)
- 4) Creation of a future operations simulation that embeds an agent-based workflow automation tool, implementing the functional specification (documented in a technical design)
- 5) Validation of the tool and revised work processes by driving the simulation with actual data
- 6) Integration of the agent-based workflow tool in the MOD work environment.

In this methodology, multiagent simulations serve multiple roles for understanding, communication, formalization, specification, validation, and implementation.

The purpose of the OCA current operations model was to create a baseline understanding and formal description of an aspect of the OCA work process that could be redesigned. Early observations of OCA operations and discussions with OCA officers indicated that mirroring of ISS files was a good candidate for improvement. Given time constraints and modelers available, our strategy in developing the current operations model was to understand and simulate enough of the system to provide confidence that we could develop a functional specification for automating the mirroring process. Consequently, the simulation does not attempt to capture any of the timings or activities of the OCA officer in any detail, except for the mirroring activity. The simulation showed that the OCA officers spend about 6% of their work time on the mirroring activity.

The development of a current operations simulation has also served as tool for management to understand the Brahms agent-based architecture and to grasp how a future operations simulation could be converted into a workflow tool. The future operations simulation is described at the end of this paper. By virtue of formalizing the future design with a prototype GUI, it serves the additional role of a management decision support tool for redesigning mission operations.

### 4 Model of the OCA Current Operations Work System

This section provides an overview of the OCA current operations model, which represents the typical actions of OCA officers during a shift. The activity model describes what the OCA officer does during the shift; only the mirroring activity is modeled in any detail in this current operations simulation. The main components of the model

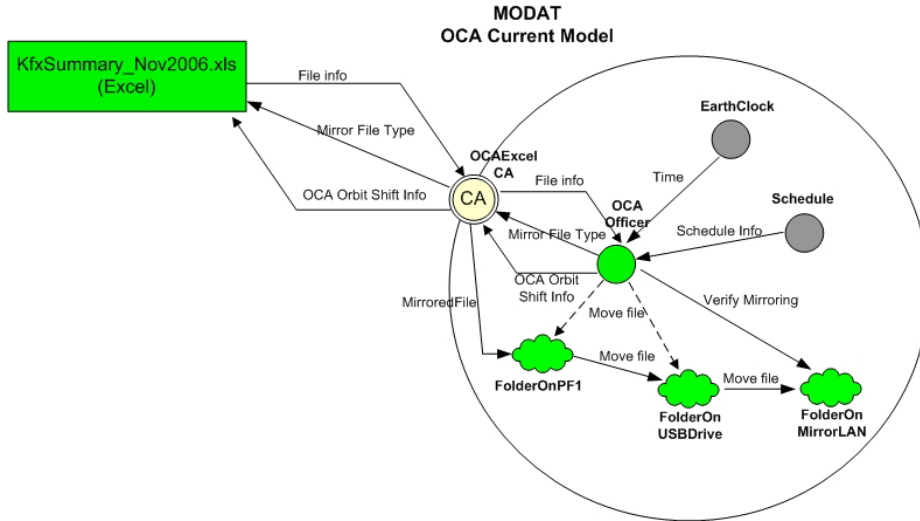


Fig. 2. Agents and Objects in the OCA Current Operations Model

Table 1. Example of Kfx Summary Log data that drives the current operations simulation

Up/Down	Downlink	Uplink
<b>GMT</b>	300/23:59:00	301/00:00:05
<b>Bytes</b>	1,364	40,539,150
<b>FileName</b>	d:\oca-down\Updates.log	U:\COSS\ILRT\Ref CD42\TrainingManuals\English\01(0)T0008E.pdf
<b>Extension</b>	log	Pdf
<b>Client</b>	Plan D	Plan B
<b>Year</b>	2006	2006

are: OCA Officers, the OCA computers and drives, Building 30S (the MCC) work areas, computer files and folders, and work schedules. Fig. 2 shows the agents and main flow of data and commands between the agents and objects in the OCA current operations model. For simplicity, the folders are represented as geographic areas (shown as clouds).

To drive the simulation, we used a spreadsheet provided by an OCA officer, KfxSummary\_Nov2006.xls, which had been derived using macros from a log created automatically by the ISS uplink/downlink software of the November 2006 file transfers. Table 1 shows an entry from the spreadsheet.

Referring to Fig. 2 notice that a special agent, called the OCA Excel Com Agent (Excel CA, hereafter ECA) provides information about what files were transferred during a particular shift (as shown in Table 1). The simulated OCA Officer agent determines whether a given file needs to be mirrored based on its type. The ECA simulates the file being placed in the location FolderOnPF1. The OCA Officer agent then operates on the file using PF1, the USB Drive, and the MirrorLAN.

1. The OCA Officer agent sends the ECA its shift information at the beginning of its shift just after handover:
  - GMT Date, e.g. 305 is Nov 1st
  - GMT Start/Stop Hour and Minute
2. ECA sends file information back to OCA agent:
  - File Extension, e.g. pdf, xml, zip
  - File Name without Path, e.g. nfhWednesday.pdf
  - File Path, e.g. /BHPG/Crew/News/
  - File Direction (Uplink or Downlink, where “up” means to the ISS)
3. OCA Officer agent applies thoughtframes that use file type information and sends back to ECA: Decision to mirror or not (true or false) and File Type symbol, e.g. OSTPV\_Snapshot\_File\_Type.
4. If the file is being mirrored, the ECA then puts the file in the location FolderOnPF1, and informs the OCA Officer agent of the location and File Type symbol.

This part of the simulation is not a model of work practice, but rather a method of driving the simulation to use actual file transfer data. The effect is that the simulated OCA Officer agent will mirror the same files during a given simulated shift that were mirrored in the corresponding actual shift, by virtue of processing the files listed for that time period in the Kfx Summary Log file.

The simulation constitutes a model of work practice (i.e., has fidelity) by virtue of including the following:

- Data about file transfers that can be derived from the Kfx logs, including file names, paths, sizes, and transfer direction.
- Relationship between file path/name and type of file (Table 1), e.g., ACKBAR files, BEV updates files, DOUG files.
- OCA officer activities of transferring files from PF1 to USB Drive to MirrorLAN, in which the duration of these activities is estimated by the actual byte size of the files being transferred at any time.
- OCA officer activities of monitoring file processing by services running on the MirrorLAN, in which the duration of these activities were estimated by OCA officers, based on file type.

Consequently, statistics can be generated from the model regarding how much time the OCA agent spends mirroring files. Furthermore, by virtue of recognizing file types, procedures for handling different types (e.g., providing notification) can be modeled more easily in the Future Operations Simulation.

The Brahms Current Operations model completely describes an OCA shift. However, only the shift handover, file transfer, mail synchronization, and mirroring operations are modeled in any detail. The behaviors of people (modeled as Brahms agents), systems (modeled as Brahms objects), and software agents (modeled as Brahms agents) are represented as Brahms workframes and thoughtframes.

Here is an example of one of the actions performed by the OCA Officer agent beginning the shift (ReadOCAHandoverLog is abbreviated ROHL):

```

workframe Read_OCA_Handover_Log {
  variables:
    forone(int) maxTime; minTime; actPriority;
  when(
    knownval(current.currentIndividualAct = ROHL) and
    knownval(ROHL.isDone = false) and
    knownval(maxTime = ROHL.maxDuration) and
    knownval(minTime = ROHL.minDuration) and
    knownval(actPriority = ROHL.activityPriority))
  do {
    moveToIndividualActivityLocation();
    ROHL(actPriority, minTime, maxTime,
          Statistics_Understanding);
    conclude((ROHL.isDone = true), fc:0);}
} // workframe Read_OCA_Handover_Log

```

In the conditional or “when” part of the workframe, the durations are read from the activity schedule object ReadOCAHandoverLog. In the action or “do” part of the workframe the agent does the following: 1) moves to an appropriate location, 2) reads the log (a primitive activity performed for the specified time), and 3) concludes that the activity of reading the log has been done. (Such propositions become part of the individual agent’s model of the world and are called *beliefs*, contrasted with the Brahms global model of the world, consisting of *facts*, which are only accessible to agents via uncertain *observables* that occur during activities [1][5].)

Reading the log is defined as a primitive activity as follows:

```

primitive_activity ReadOCAHandoverLog(int priorityNum,
int minDuration, int maxDuration, Statistics statObj) {
  display: "Read OCA Handover Log";
  priority: priorityNum; random: true;
  min_duration: minDuration; max_duration: maxDuration;
  resources: statObj; }

```

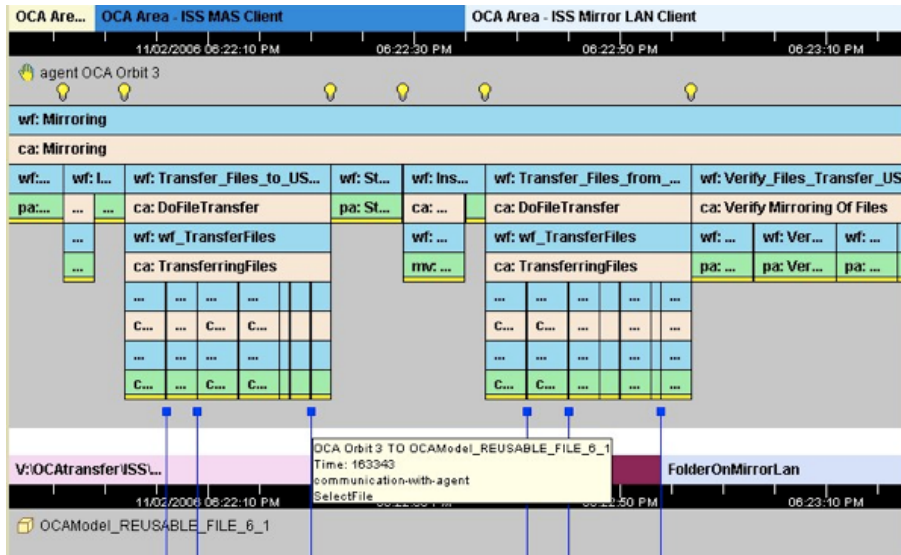
The resources property indicates that statistical information about this activity should be logged by the simulation engine. See Section 4.2 below on Statistical Charts. The min\_duration, max\_duration and random facets of the activity definition specify a random duration at runtime.

## 5 OCA Current Operations Simulation Output

The Current Operations model was run for 31 simulated days, corresponding to the OCA officers’ shifts in November 2006. The simulation result can be verified and validated using two methods, the AgentViewer and statistical charts.

### 5.1 AgentViewer

A Brahms simulation produces a history file in the form of a database that can be diagrammed and studied in the AgentViewer. This allows us to understand the behavior of agents and objects during the simulation. Fig. 3 shows agent behaviors chronologically as activities; Workframes (darker shade) shown with “wf”, Primitive



**Fig. 3.** Simulated OCA agent’s actions during Orbit 3 (starting 5:50 PM CST; each white mark is a clock tick = 5 minutes). Behaviors are modeled as workframes with composite activities (CA) that invoke workframes, ending in primitive activities, such as communications (shown as vertical lines connecting to file objects that are not visible here).

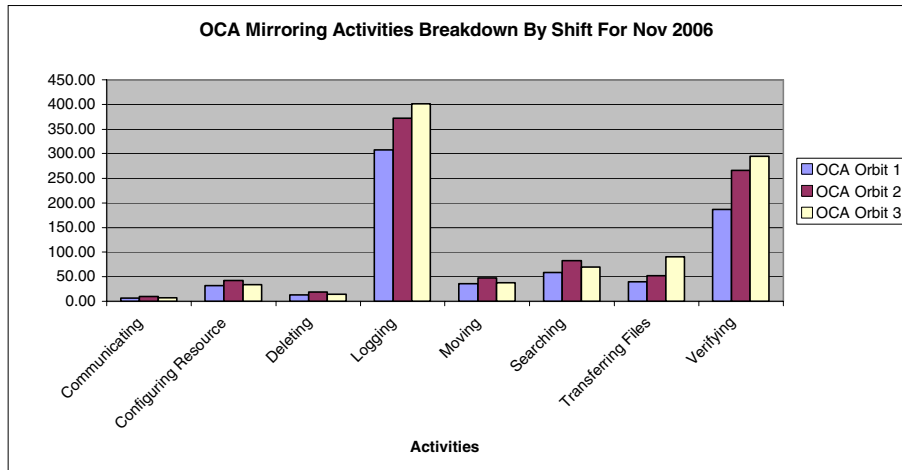
Actions (at the bottom) shown with “pa:”, Composite Activities (light shade) shown with “ca:”, Communications (vertical lines), and Thoughtframe conclusions (light bulbs). Locations appear in the bar above the black timeline for each agent or object.

The agent is transferring files to a USB drive (sitting at ISS MAS client area) and then mirroring the files to the MirrorLAN (at the MirrorLAN area). Labels that won’t fit are shown as three dots (...). One communication has been selected, causing details to pop up. Light bulbs may also be selected for details about the belief concluded by a Thoughtframe.

**5.2 Statistical Charts**

A Brahms simulation can be “instrumented” by defining a resources property for primitive activities (i.e. actions). For example, the communication action SelectFile has the resources property Statistics\_Transferring\_Files, an object. The primitive activity DraggingFile has the resources property “file,” which is the object being manipulated. When the Brahms executive (simulation engine) encounters a resources property, it logs data about the agents, objects, and durations during which that resource was worked on. The current operations model is annotated by 20 statistics categories. These categories represent general work “chunks” classifying the different activities of the OCA officer for which we want the simulation to generate decision-metrics. Fig. 4 illustrates the kinds of charts that can be generated from the resulting statistics.

Analysis of such charts revealed that the OCA Officer spends most of the shift logging and verifying file transfers. Our design has therefore focused on automating the



**Fig. 4.** Example of simulation results for mirroring subactivities. Note: Not definitive data; these charts represent work in progress. They are not necessarily accurate and do not reflect later changes made to the model.

mirroring activity in such a way that these subactivities in the future do not have to be performed by the OCA officer. Besides eliminating a manual error-prone process, automating the mirror activity will result in the OCA officer saving time and potentially enable the position to be given less tedious responsibilities.

More specific charts compare shifts according to the percent of total files transferred to the percentage of files in a given shift (called an orbit) that are mirrored. For example, Orbit 2 (the daytime shift) processed 36% of the files transferred between ground and ISS during November 2006. Of the files that Orbit 2 transferred about 31% were mirrored. Thus, on average about a third of the files processed by an OCA officer are mirrored. This represents a significant workload (about 2500 files manually manipulated) and further justifies automation.

In developing the future operations simulation, which will change the work design by including a workflow tool for mirroring, we could choose to model additional activities, such as communications with flight controllers, and instrument these events to gather statistics from the simulation. These statistics can then be compared to observations we make of OCA operations, leading us to refine the model or gain confidence in the simulation's predictions. In particular, we could use the simulation to predict that the OCA officer could take on other responsibilities, and include these in another future operations simulation. In this way, we proceed through observation, collaborative design, simulation, and redesign to incrementally improve how the work is done, gaining efficiency and reliability.

## 6 Creating a Future Operations Simulation

The steps in creating an OCA future operations simulation include: 1) Creating a functional design of the revised OCA work system, including automation of mirroring,

2) Revising the current operations model to more accurately represent the aspects of work pertinent to mirroring automation, 3) Implementing the functional design as Brahms agents, revised OCA Officer agent activities, and a simulated GUI for human-agent interactions, 4) Validating the simulation using existing Kfx Summary logs and carrying out “what if” simulations that introduce problems (e.g., unavailable systems or errors during mirroring).

At the time of this writing, the project is in step 3, implementing the revised work system design as a Future Operations Simulation. The simulation will include approximately 80% of the OCAMS tool. Using the tool, the work process is modified, such that the OCA Officer will perform the following operations:

1. Select files to mirror (reviewing Mirroring Decision Agent’s selections)
2. Submit session (a batch) of files to mirror
3. Review and verify results; delete session of mirrored files
4. Handle files not mirrored by OCAMS manually
5. Handle files with MirrorLAN errors identified by OCAMS
6. Review mirrored files automatically logged in handover document
7. Notify flight controllers mirroring completed.

Fig. 5 shows the first version of the future operations simulation. It shows the flow of shift information between the OCA Officer, the Mirroring Decision Agent, and the Monitoring Agent. Files to be mirrored are transmitted by FTP to a staging area on a separate computer where the Monitoring Agent individually moves files to the MirrorLAN (via a drive mapping) and inspects the outcome of batch file execution.

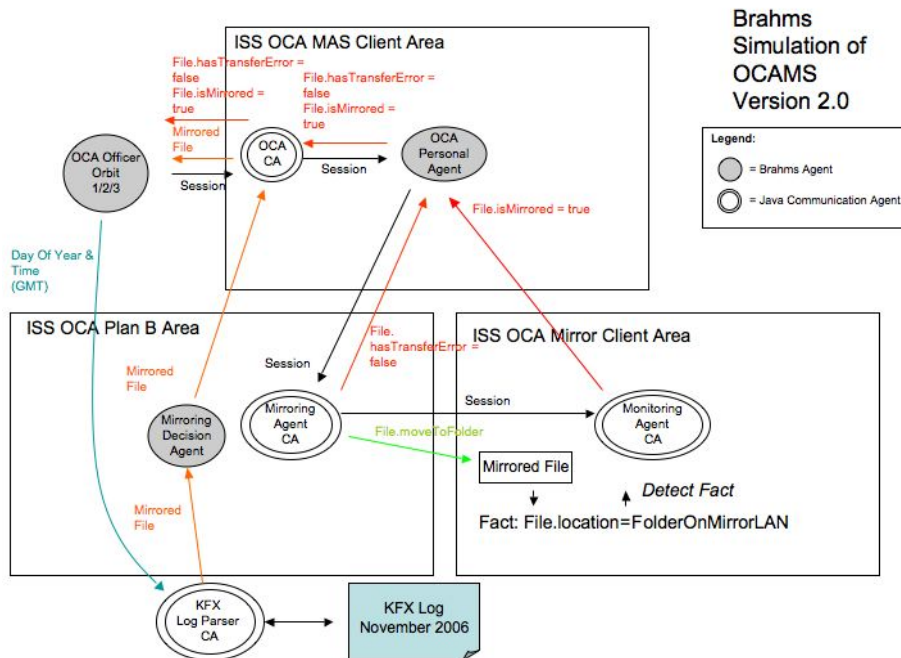


Fig. 5. Future Operations Model, derived from Current Operations Model (Fig. 2)

Through collaboration among the ARC project team, JSC OCA officers, and management, the Future Operations Simulation will be modified to adjust the design of the OCA work system (e.g., as may be required for implementation in the MCC). After it is agreed that the design is complete and validated through simulations, a runtime distributed Brahms agent system can be extracted from the Future Operations Simulation and packaged as the OCAMS tool, to run in the Brahms virtual machine on computers and the network in the OCA backroom area of the MCC. A certification process, to be defined, might include constructing a mockup of this network and operations.

Continuing ethnographic observations of OCA operations will be a key part of our work while simulating the future operations to verify the simulations and to understand how mirroring operations interact with other aspects of work practice, such as notification to other flight controllers. Furthermore, we know from observation that mirroring is a useful training ground for new OCA officers. Therefore, we want to implement the system in such a way that manual operations on the MirrorLAN are still possible and that automated processes adapt accordingly. Similarly, after implementation of the OCAMS tool, an important new phase of observation will begin to understand changes to the work practice, emergent uses of the tool, and ways to improve it.

## 7 Related Work

In this section we compare and contrast the Brahms simulation framework to Workflow Management Systems and Agent-Based Modeling and Simulation (ABMS) to show the modeling requirements of the simulation to implementation approach.

### 7.1 Workflow Management Systems

Workflow Management Systems (WfMS) have evolved from business management, business process reengineering, business process modeling and simulation, and to a lesser extent artificial intelligence. OCAMS is a WfMS, where the automated process is not a business, but mission operations.

Recently, workflow modeling languages and tools have been developed as industry standards. For example, Business Process Execution Language (BPEL) is an XML-based language with structured, executable programming concepts that can be integrated with web services. The most common language in academia is the Petri-Net language [10, 11], which uses complex state transition diagrams to model programs and parallel processes such as concurrent tasks. A multi-agent system can be modeled as parallel Petri-Nets.

Workflow models are based on a functional flow-based abstraction of the work, modeling defined tasks and operations such as those formalized in business procedures. In contrast, Brahms' activity-based approach [2] enables modeling the complexity of activities, communication practices, relationships, and circumstantial details of coordination and workarounds, together constituting *work practice*, by which functions are accomplished [12] [9]. Although I-X [13] also uses the concept of an activity, its framework uses a task-planning approach. A Brahms activity is a

broader concept, including more than goal-directed problem solving, such as resting by informally conversing with co-workers [2].

In systems with run-time capabilities, the work process model is or can be automatically transformed into an executable language. For example, BPNM, IDEF3, colored Petri-Net and YAWL languages are imperative programming languages. In contrast, Brahms is an agent-oriented Belief-Desire-Intention (BDI) language which represents processes as an organization of agents with individual beliefs, coordinating group and agent-specific activities represented as situation-action rules [14]. Rather than only expressing functional transformations, Brahms enables representing roles, points of view, habits, temporal rhythms of behavior, contextual factors, communication media, tools, conversations, etc. This level of specificity enables a Brahms agent that automates work to *fit* into the practices of the people who must interact with it, an understanding encouraged by and enabling the embodiment of participatory design within a simulation-to-implementation engineering approach.

## 7.2 Agent-Based Modeling and Simulation

Agent-based Modeling and Simulation (ABMS) is a term mostly used by researchers in complex adaptive systems to model systems of relatively simple agents that derive their emergent behavior from the system as a whole, instead of from complexity within the agents themselves. Tools for ABMS such as Swarm [15] and Repast [16] are not based on any particular human behavior theory and are not BDI languages; agent methods are driven by a global scheduler. In contrast, Brahms models cognitive agents; their internal state (possible and incomplete activities, plus beliefs, which can represent plans and goals) combined with a complex modeled environment determines the agents' next behaviors. Thus the Brahms language is both a BDI agent language and an ABMS language, which is important for example in representing decision making in mirroring files and handling errors.

In the category of BDI languages [17], Brahms is distinguished from systems such as Jason and AgentsSpeak by its use of a subsumption-based architecture [18] for representing an agent's conceptualization of activities as parallel-hierarchical processes [1, 2]. This allows modeling how activities are like identities that blend and contextually change what is perceived, how communications are interpreted, and how tasks are prioritized. See [14] for additional comparisons to agent-oriented languages.

## 8 Conclusions

The OCAMS agent system is designed to automate workflow deterministically, under OCA officer control to develop trust, enable customization, manage problems/shortcomings, and retain a manual approach for use in training. A key aspect of this practical engineering project is the highly interdisciplinary team that partners operations personnel with researchers and combines specialized knowledge from computer science, anthropology, spaceflight operations, and work systems design.

The use of Brahms demonstrates how agents can be used in a "simulation to implementation" methodology by which a model of current operations is converted into a future operations model that incorporates both essential aspects of an agent-based tool and a simulation of how the tool interacts with people and other systems. This

hybrid simulation enables flexible, incremental development of an implementation, such that actual systems (e.g., email, FTP, files) replace simulated systems and objects. The simulations are driven by logs of the actual work performed in the past, and the future operations simulation operates upon the actual files manipulated by the OCA officers. By running the simulation subsequently with data from other months, we can validate the generality of the mirroring rules and special handling designed into the tool.

OCAMS is one of the first steps in developing an interplanetary multiagent system that integrates people on earth and astronauts with a diversity of hardware and software systems. The combination of agent-based simulation and systems integration enables great efficiency in designing, validating, and deploying practical tools.

**Acknowledgments.** Brahms was originally developed 1992-1997 as a joint project between NYNEX Science & Technology and The Institute for Research on Learning [1, 5], and reengineered in Java at ARC from 1998-2001. The runtime form of Brahms was developed in the Mobile Agents project [10]. We are grateful to Tom Diegelman at NASA JSC for promoting applications of Brahms to mission operations design and securing seed funding for this project in 2006 [11]; Brian Anderson, Dennis Webb, and Ernie Smith also provided essential support. Several other OCA officers not listed as co-authors reviewed and commented on the OCAMS specifications, including Skip Moore and Karen Wells. This project has been supported in part by funding from NASA's Constellation Program.

## References

1. Clancey, W.J., Sachs, P., Sierhuis, M., van Hoof, R.: Brahms: Simulating Practice for Work Systems Design. *International Journal on Human-Computer Studies* 49, 831-865 (1998)
2. Clancey, W.J.: Simulating Activities: Relating Motives, Deliberation, and Attentive Coordination. *Cognitive Systems Research* 3(3), 471-499 (2002)
3. van Hoof, R., Sierhuis, M.: Brahms Language Reference (2000), [http://www.agentisolutions.com/documentation/language/ls\\_title.htm](http://www.agentisolutions.com/documentation/language/ls_title.htm)
4. Seah, C., Sierhuis, M., Clancey, W.J.: Multi-agent Modeling and Simulation Approach for Design and Analysis of MER Mission Operations. *SIMCHI: Human-computer interface advances for modeling and simulation*, January 2005, pp. 73-78 (2005)
5. Sierhuis, M.: Modeling and Simulating Work Practice; Brahms: A Multiagent Modeling and Simulation Language for Work System Analysis and Design. In: *Dissertation in Social Science Informatics (SWI)*, The Netherlands. SIKS Dissertation 10, University of Amsterdam, Amsterdam (2001)
6. Sierhuis, M., Clancey, W.J., Seah, C., Trimble, J.P., Sims, M.H.: Modeling and Simulation for Mission Operations Work System Design. *Journal of Management Information Systems* 19(4), 85-129 (2003)
7. Sierhuis, M., Clancey, W.J., Seah, C., Acquisti, A., Bushnell, D., Damer, B., Dorigi, N., Edwards, L., Faithorn, L., Flueckiger, L., van Hoof, R., Lees, D., Nandkumar, A., Neukom, C., Scott, M., Sims, M., Wales, R., Wang, S.-Y., Wood, J., Zhang, B.: Agent-based Mission Modeling and Simulation. In: *Agent Directed Simulation 2006*; part of the 2006 Spring Simulation Multiconference, Huntsville, AL (2006)

8. Blomberg, J., Giacomini, J., Mosher, A., Swenton-Wall, P.: Ethnographic Field Methods and Their Relation to Design. In: Schuller, A.N.D. (ed.) *Participatory Design: Principles and Practices*, pp. 123–155. Lawrence Erlbaum Associates, Hillsdale (1993)
9. Greenbaum, J., Kyng, M. (eds.): *Design at Work: Cooperative design of computer systems*, Hillsdale. Lawrence Erlbaum, NJ (1991)
10. van der Aalst, W.M.P.: Putting Petri Nets to Work in the Workflow Arena. In: van der Aalst, J.M.C.W., Kordon, F., Kotsis, G., Moldt, D. (eds.) *Petri Net Approaches for Modeling and Validation* (pp. pp. 125–143. Lincom Europa, Munich (2003)
11. van der Aalst, W.M.P., ter Hofstede, A.H.M.: YAWL: Yet Another Workflow Language. *Information Systems* 30(4), 245–275 (2005)
12. Sachs, P.: Transforming Work: Collaboration, Learning, and Design. *Communications of the ACM* 38(9), 36–44 (1995)
13. Wickler, G., Tate, A., Hansberger, J.: Supporting Collaborative Operations within a Coalition Personnel Recovery Center. In: Paper presented at the International Conference on Integration of Knowledge Intensive Multi-Agent Systems, Waltham, MA (2007)
14. Sierhuis, M.: It's not just goals all the way down – It's activities all the way down. In: *Engineering Societies in the Agents World, Seventh International Workshop (ESAW 2006)*. Springer, Dublin (in press, 2006)
15. Minar, M., Burkhart, R., Langton, C.: *Swarm Development Group* (1996), <http://www.swarm.org>
16. Tatara, E., North, M.J., Howe, T.R., Collier, N.T., Vos, J.R.: An Introduction to Repast Modeling by Using a Simple Predator-Prey Example. In: *Agent 2006 Conference on Social Agents: Results and Prospects*, Argonne National Laboratory, Argonne, IL (2006)
17. Bordini, R.H., Dastani, M., Dix, J., Seghrouchni, A.E.F. (eds.): *Multi-Agent Programming: Languages, Platforms and Applications*. Springer Science+Business Media, Inc, New York (2005)
18. Brooks, R.A.: A Robust Layered Control System for a Mobile Robot. *IEEE Journal of Robotics and Automation* 2(1), 14–23 (1986)
19. 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., Rupert, S.: Automating CapCom Using Mobile Agents and Robotic Assistants. In: *American Institute of Aeronautics and Astronautics 1st Space Exploration Conference*, Orlando, FL (2005)
20. Sierhuis, M., Diegelman, T.E., Seah, C., Shalin, V., Clancey, W.J., Selvin, A.M.: Agent-based Simulation of Shuttle Mission Operations. In: *Agent-Directed Simulation 2007; part of the 2007 Spring Simulation Multiconference*, Norfolk, VA, pp. 53–60 (2007)