SJSU SAN JOSÉ STATE UNIVERSITY

traclabs

Usability Issues and Guidance for Flexible Execution of Procedural Work

Function Allocation with Smart Technology

Dorrit Billman
San Jose State University
@ NASA Ames Research Center
Debra Schreckenghost
TRAClabs

The motivating problem

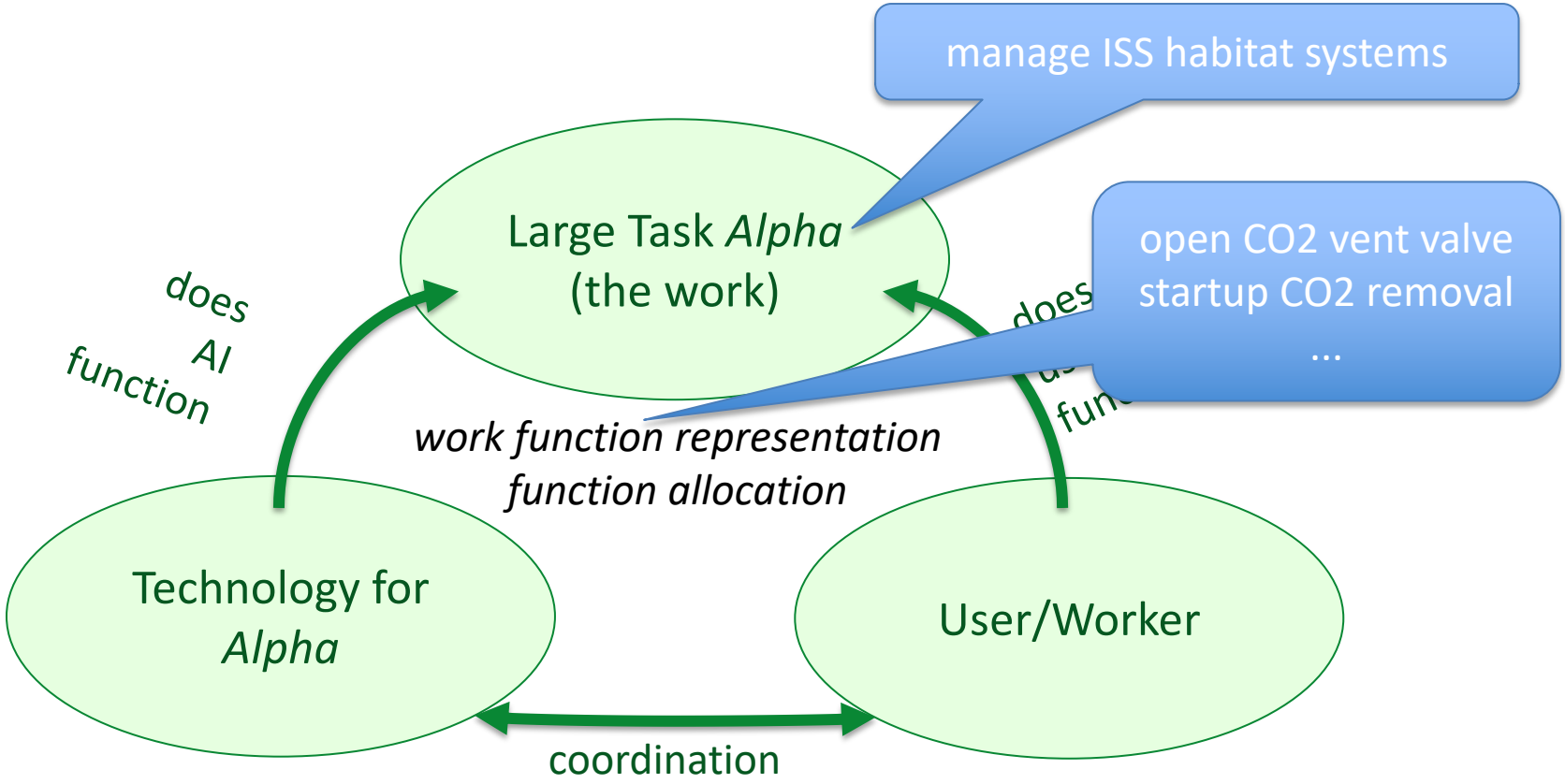
- Increased automation for future long distance crewed missions.
- Human-Automation teaming : UX in AI.
- Difficult to predict needs.

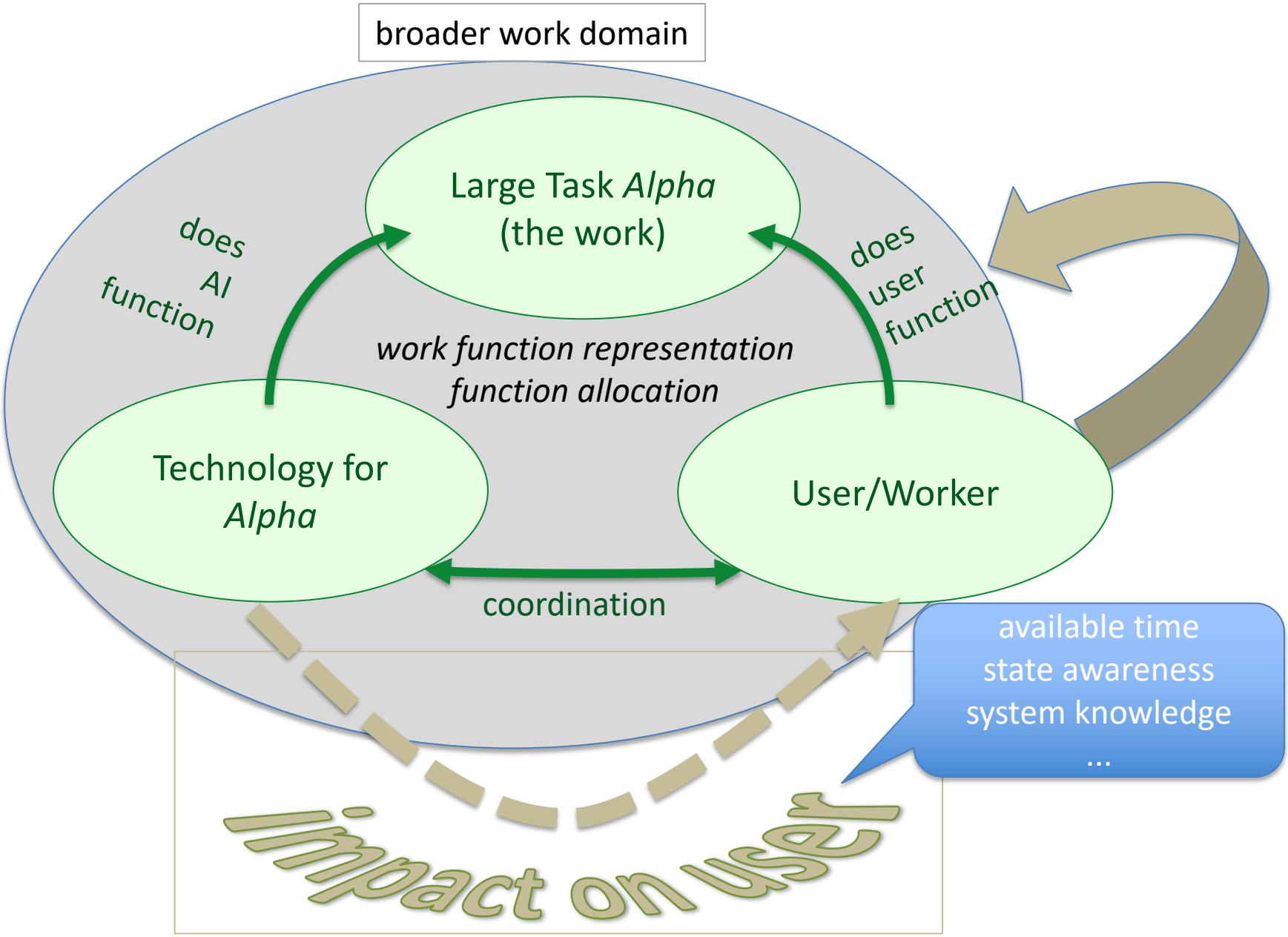
- Flexibility valuable, what does smart technology do, what does user do?

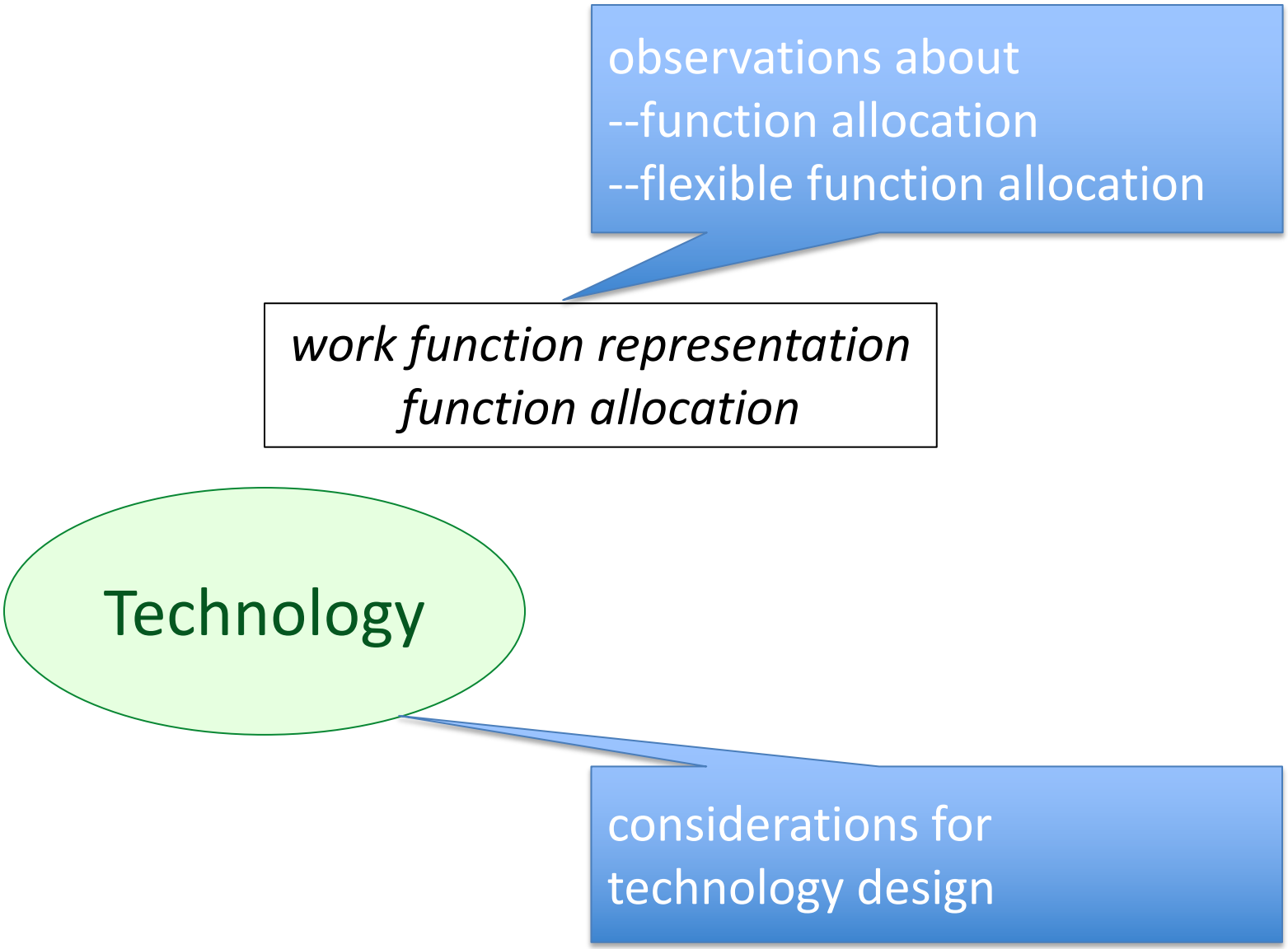
Design for Given vs Open Goals

Work: Set of prior, extrinsic goals, constraints, activities driving technology design.

Work-focused UX of AI : work success→ UX

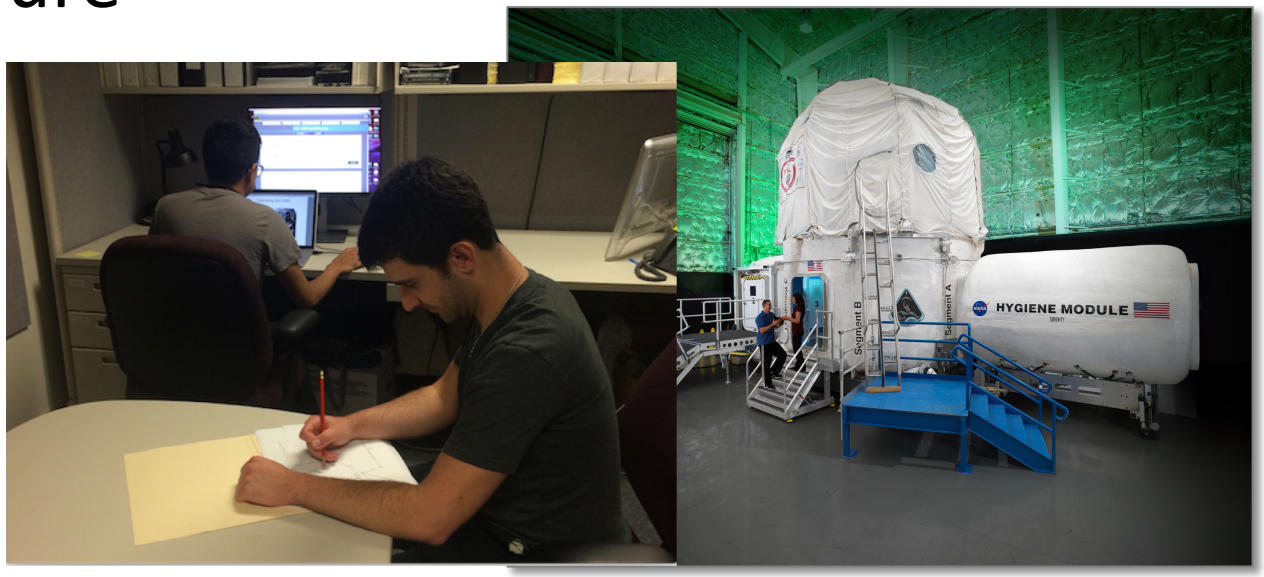

manage ISS habitat systems
Large Task *Alpha*
(the work)
open CO2 vent valve
startup CO2 removal
...
does
AI
function
work function representation
function allocation
Technology for
Alpha
User/Worker
coordination


broader work domain
Large Task *Alpha*
(the work)
does
AI
function
does
user
function
work function representation
function allocation
Technology for
Alpha
User/Worker
coordination
available time
state awareness
system knowledge
...
impact on user


observations about
--function allocation
--flexible function allocation
work function representation
function allocation
Technology
considerations for
technology design

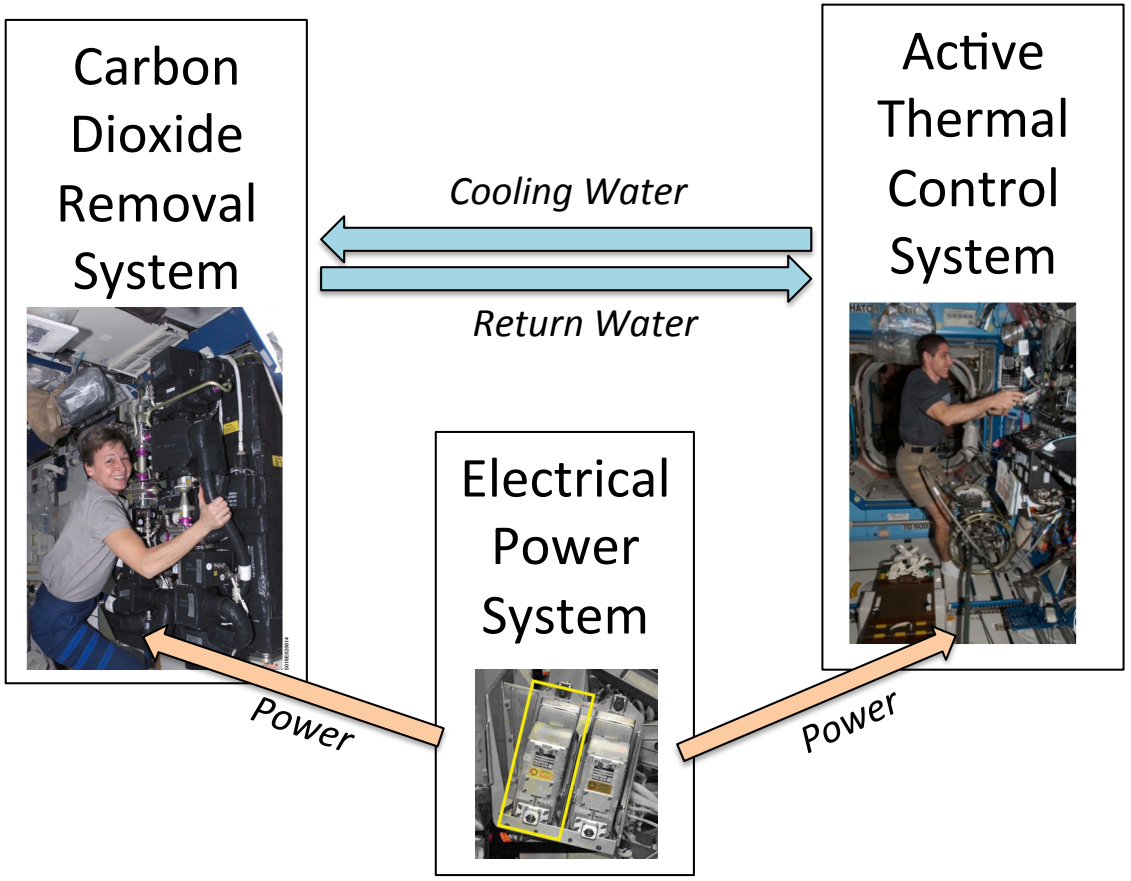
Source of Observations

- Lab studies
- Simulation of International Space Station (ISS) habitat systems
- Using procedure automation software.



Our context: The core domain

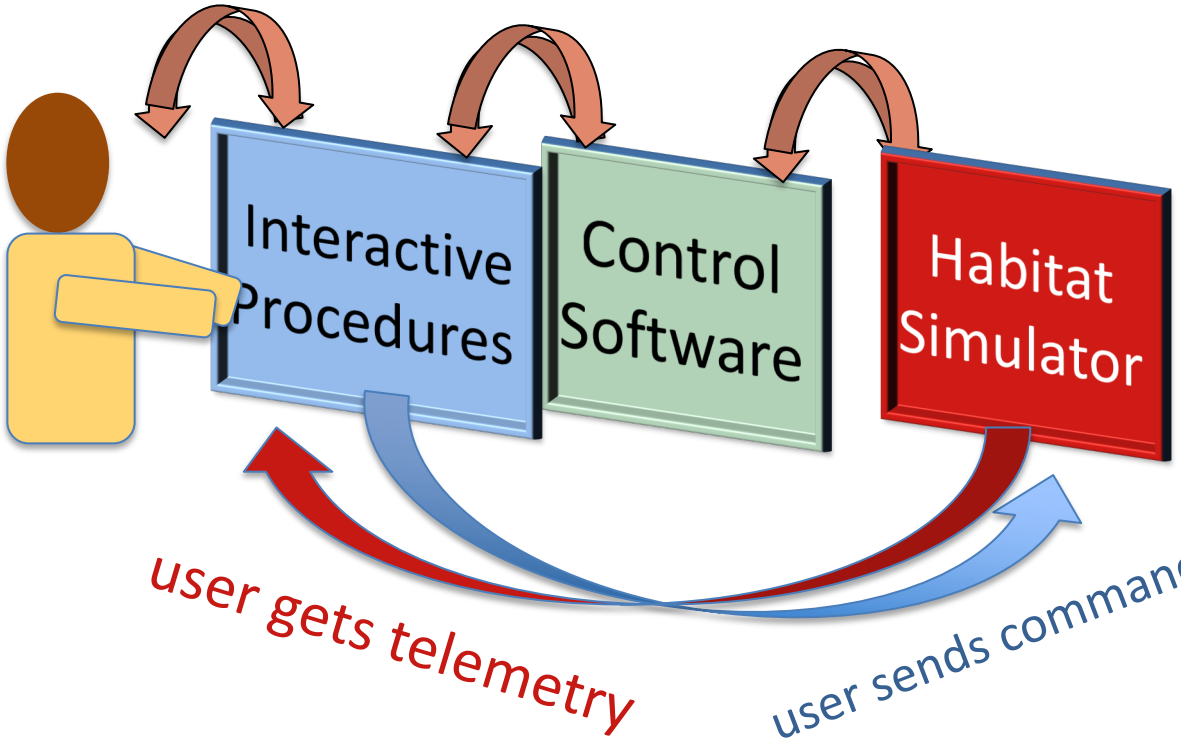
Users supervise procedures for operating ISS systems.



Goal of Our Empirical Studies

- Understand human-system integration
- Formative evaluation
- Explore flexible function allocation

The Software:
PRIDEview, PRIDE, & BioSim

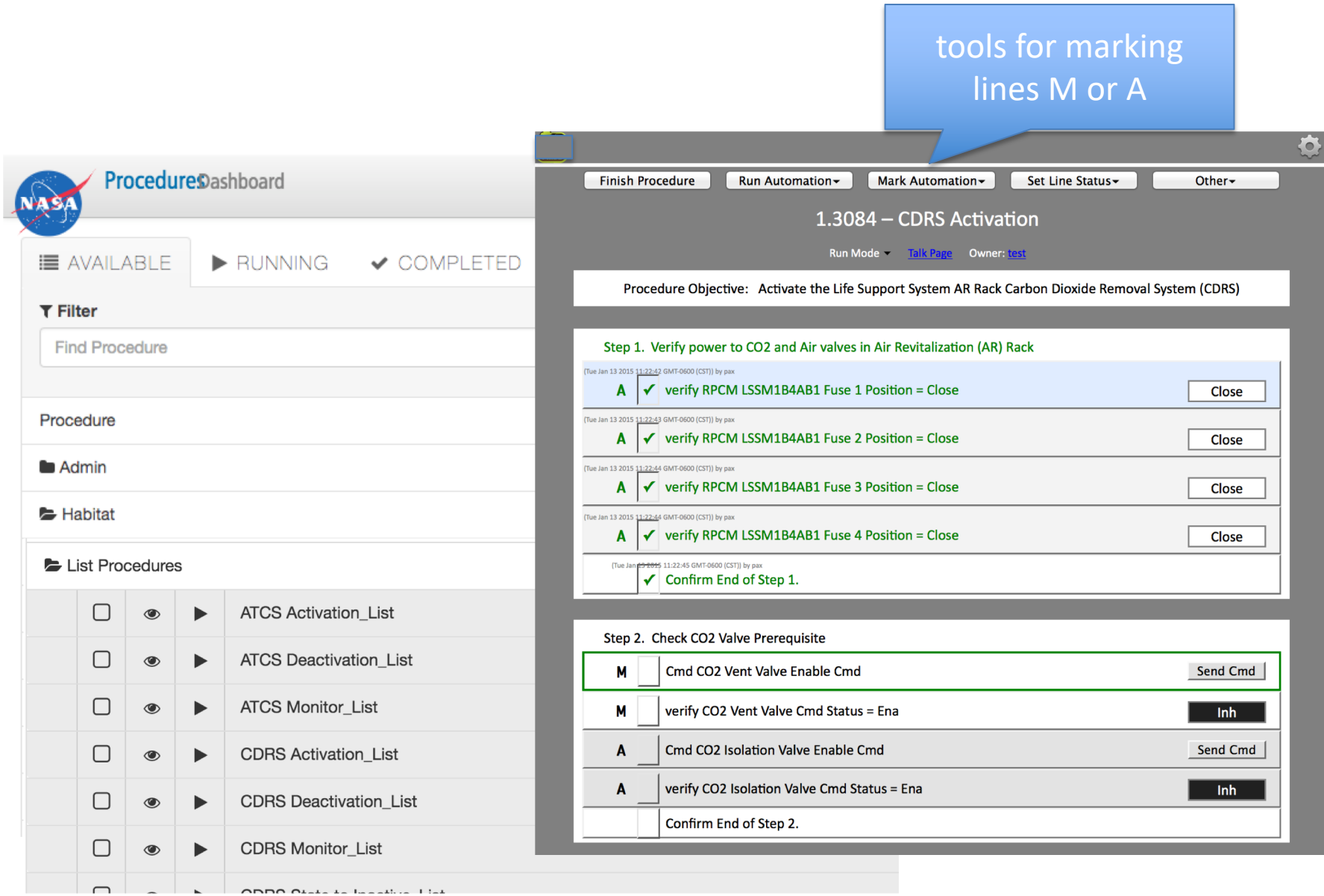


get telemetry

send command

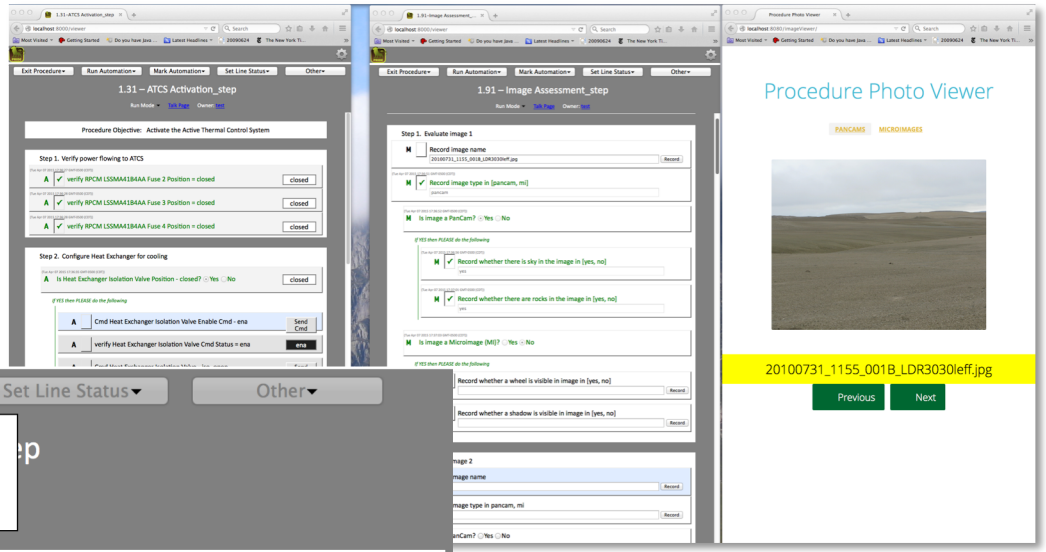
A	✓	verify RPCM LSSMA41B4AA Fuse 1 Position Status = closed	closed
A		Cmd RPCM LSSMA41B4AA Fuse 1 Position - open	Send Cmd
A		verify RPCM LSSMA41B4AA Fuse 1 Position = open	closed

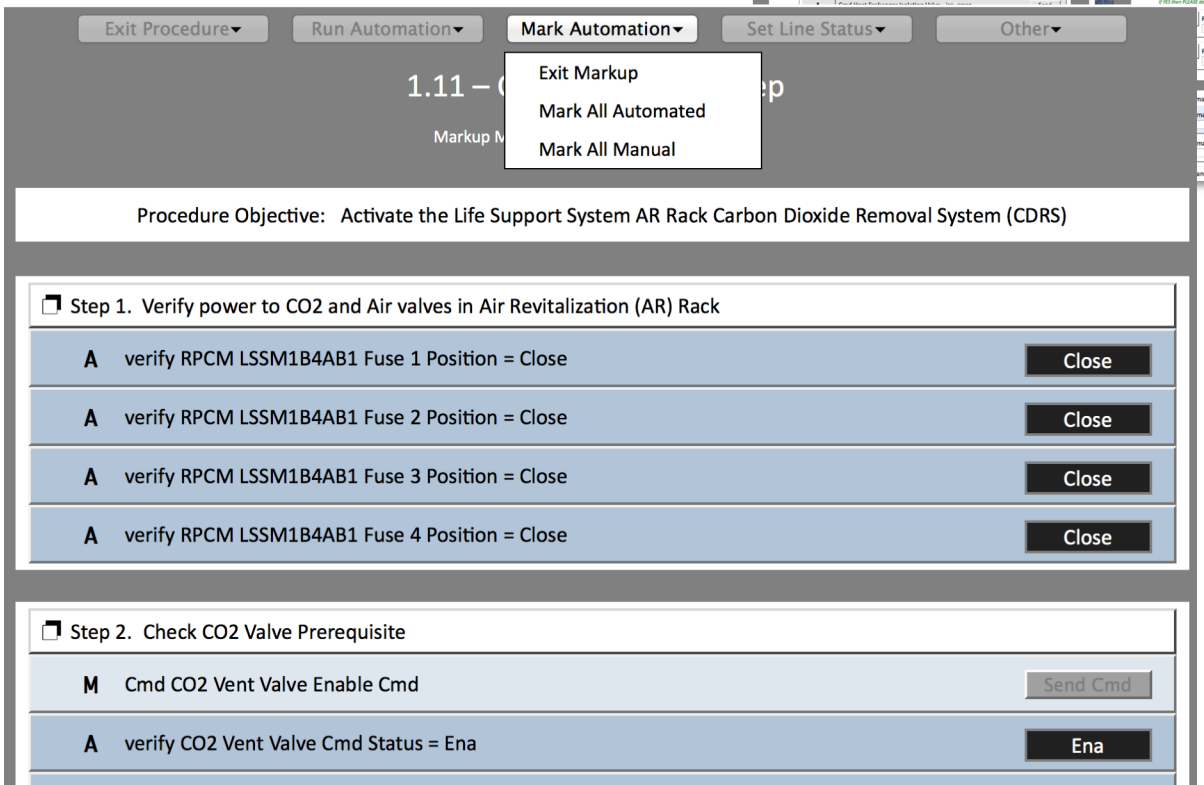
Interface


tools for marking lines M or A
Procedures Dashboard
AVAILABLE
RUNNING
COMPLETED
Filter
Find Procedure
Procedure
Admin
Habitat
List Procedures
ATCS Activation_List
ATCS Deactivation_List
ATCS Monitor_List
CDRS Activation_List
CDRS Deactivation_List
CDRS Monitor_List
Finish Procedure
Run Automation
Mark Automation
Set Line Status
Other
1.3084 – CDRS Activation
Run Mode
Talk Page
Owner: test
Procedure Objective: Activate the Life Support System AR Rack Carbon Dioxide Removal System (CDRS)
Step 1. Verify power to CO2 and Air valves in Air Revitalization (AR) Rack
A verify RPCM LSSM1B4AB1 Fuse 1 Position = Close
Close
A verify RPCM LSSM1B4AB1 Fuse 2 Position = Close
Close
A verify RPCM LSSM1B4AB1 Fuse 3 Position = Close
Close
A verify RPCM LSSM1B4AB1 Fuse 4 Position = Close
Close
Confirm End of Step 1.
Step 2. Check CO2 Valve Prerequisite
M Cmd CO2 Vent Valve Enable Cmd
Send Cmd
M verify CO2 Vent Valve Cmd Status = Ena
Inh
A Cmd CO2 Isolation Valve Enable Cmd
Send Cmd
A verify CO2 Isolation Valve Cmd Status = Ena
Inh
Confirm End of Step 2.

Variety of Tasks

Dual tasks
Normal or problems
Flexible function allocation





Change assignment of manual vs automatic execution.

Factors guiding function allocation

- representation of work
- capabilities of agents
  - (technology & humans)
- automation goals

How are units of work represented? What are the functions to be allocated?

In our case:
Procedures designed for manual execution represent units of work for human-automation "team".
Provides common ground for user-AI communication.

Procedures, steps, actions

How are units of work represented? What are the functions to be allocated?

Types of Work	Continuous Control	Discrete Sequence
examples	autoflight, lane following	surgery procedures, flight checklists, ISS habitat operation
type of unit	parameters, constraints	action sequences
size of unit	{heading} vs {heading, altitude, speed}	**Procedures, steps, actions**

Hierarchical Procedure Representation

- Procedure, step, action units
- Units clearly related, comprehensible

Recommendation: discrete-sequence domains,

use human-originated, hierarchical procedures: they a) provide solution method

b) aid comprehension

c) aid coordination

Given units of work, what guides function allocation? Capability

- Agent must have threshold capability to be allocated the function. Only choices if multiple capable agents.

In our case, “all or none” capability executing actions

  - human capable for all actions, automation only some.

- Other cases, degree of inherent capability might vary

Given work units and capabilities, what guides FA? Goals/Purpose of Automation

- Socio- technical focus
  - Safety
  - Speed/Efficiency
  - Standardization/transparency

- User focus
  - experience in use
  - what user carries away

In our case, work success → user experience

Goals of Automation In Our Case & Design Considerations

- safety
- overall efficiency
- robustness/flexibility across unexpected situations
  - flexible function allocation

- freeing user (astronaut) time {1,2,3,4,5}
- ensuring user understanding {5,6,9}
- adapting to the unexpected {7,8,9,10}

1. Design for time on other tasks: useful blocks of time

Minimize total time on manual work?

- Avoid FA plans that fragment blocks of user time below *useful length* for domain.
- Consider grouping required-manual actions together, when position is not constrained by system factors, to gain useful time-blocks.
- Lesson learned

2. Design for useful time on other tasks: interruption management

Handovers interrupt user's activity. Support management of task-switching.

- Consider notification methods for providing information about why automation stopped; balance low enough disruption with high enough salience.
- In our case, stopping of the checklist scrolling when automation stopped was insufficient flag; adding a notification panel was promising.

(Warnings and alarms possible.)

3. Design to minimize # of handovers

Team work heavy around handovers, fewer is better.

- Resist automating actions if this increases handovers. Cost of handover may cancel benefit of automatic execution. (Handovers produce disruptive shift of focus.)
- In our case, when manual and automatic actions were interleaved, some users manually executed actions allocated to automation.

4. Design to minimize cost of handovers: interface time

Interface can reduce handover costs by reducing interaction overhead.

- Consider reducing interaction overhead with auto-resume functions, if this is a prevalent use pattern.
- In our case, this was done in some versions, and may be helpful overall. This behavior was non-obvious for some users, perhaps b/c it depends on status of *following* action. Procedures were revised to prevent behavior when not desired.

5. Design to minimize cost of handovers: user orientation

The interface can reduce handover costs by helping the user orient, both for planned manual execution and when automation execution halts due to a problem.

- Orient user with in-context task & system info included in the procedure
- In our case, procedures show
  - task progress (e.g. checklist) and
  - system status (e.g. verification lines show variable values)

Added monitor-procedures for better view of system states.

6. Design to support understanding

Particularly valuable for mixed initiative systems, particularly when automation halts due to a problem.

- Consider embedding guidance about system operation in execution context.
- In our case, procedure steps annotated with information about
  - action requirements or preconditions and
  - the goal a step is intended to accomplish.

Design for flexibility: Why flexible function allocation?

Multiple information sources:

1) AI/technology capabilities- engineers

2) Procedure specification-authors

*3) Operational conditions- users

Do conditions vary enough to make it worth while to FA add for users?

7. Design for flexibility: variation in operational conditions

- Consider whether operations vary in need for user attention vs AI independence. Is flexibility more valuable than standardization?

If not, don't add unneeded complexity.

If so, consider methods for user to allocate functions among users and technologies.

- In our case: needs change faster than procedures are updated; role of Flight Notes. Need for close monitoring or user action shifts with conditions.

8. Design user FA methods to cover range of allocation purposes needed

- Specific tweak vs overall edit? Both can be useful.
- In our case, we provided a light-weight method to specify a point where automation should stop, and an editing mode for setting function assignment on a procedure.

9. Design user's FA methods for transparency.

- Ensure user can tell what current allocation is and what allocation is possible.
- In our case, initially users had to infer what was or could be automated; revised interface provided explicit (A) or (M) annotation.
- Ensure user can distinguish executing from editing. If FA-editing is in a different mode from execution, minimize mode confusability.
- In our case, mode is marked by different background; noticing and understanding color change must be trained.

10. Revise & repurpose?

- The optimal procedure for all manual vs all automatic execution might differ. Edit procedures to modify order of actions, depending on FA, e.g. to group manual actions together? If so what sort of guidance?

- When conditions, resources, or goals don't match? Productive reuse of components.
- Ongoing research on problem solving and generalization.

(General?) Tradeoffs

- interdependence of different “layers” of design

  <procedure design> before <function allocation>
- goal conflict in order guidance
  - manual actions together vs goal coherence
- flexibility vs stability
- transparency vs TMI
- power to modify vs complexity of software
- design for routine vs design for edge cases

Across design problems,
is it same dimensions, different values?
How does type of work/nonwork frame design problems?