# Hierarchy-in-flux: Co-evolving a distributed user interface for orbiting robots

Barba, Evan, Miller, Christopher and Majeed, Yasir

# Hierarchy in Flux

# Scenario: Retrieve a airplane Blackbox

* Tele-operated robot with toolkit (controlled by umbilicus 6sec delay)
* Robot operators in control room (video, telemetry, sensor arrays)
* “Dry” testing environment

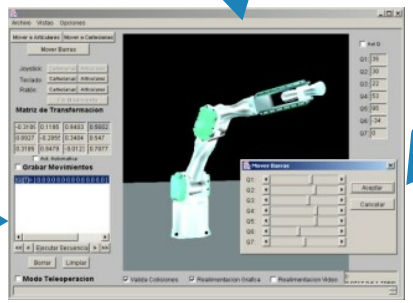
# Sociotechnical System

Leaders

Robot operators

Tele-operated robot

Dry-testing



Interface

# Emergence: higher scale effects

- Strong Emergence

  "effects you could not anticipate or deduce"

- Weak Emergence

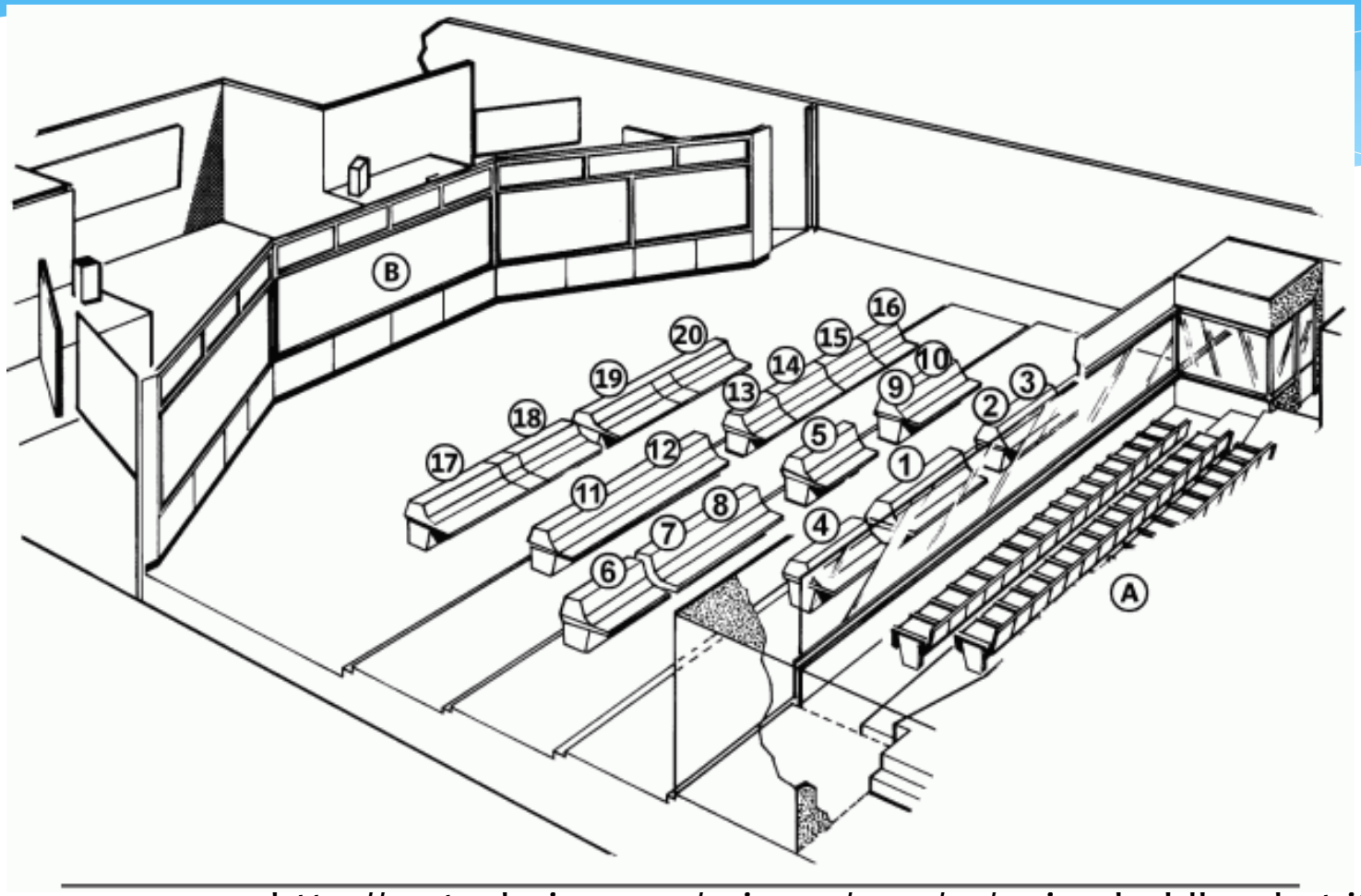
"predictable collective action"

# Engineering Emergence

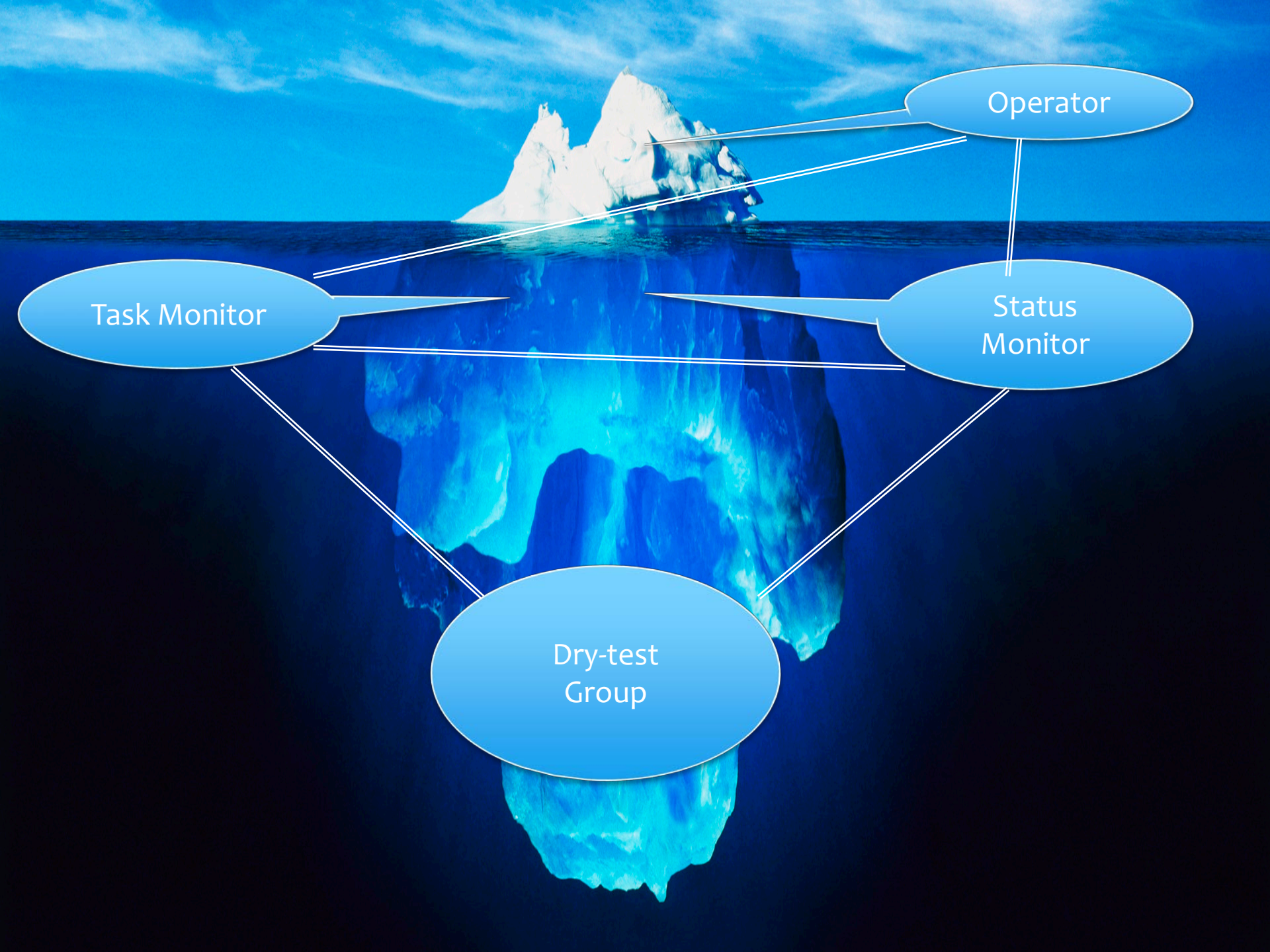
* Maximize Weak Emergence
* Minimize (eliminate) strong emergence

* Limit interaction between parts of the system

# An Analogy




http://arstechnica.com/science/2012/10/going-boldly-what-it-was-like-to-be-an-apollo-flight-controller/


Operator
Task Monitor
Status Monitor
Dry-test Group

# Why does this work?

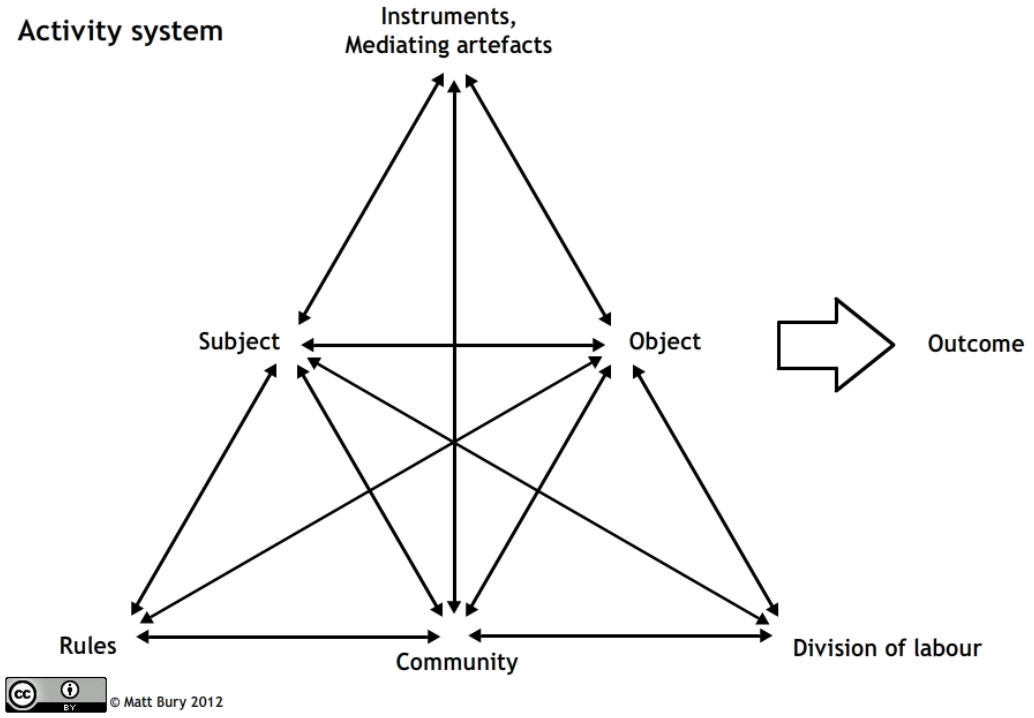
* Every role is specialized
  * Every specialist is focused on one small set of tasks
  * Every task is clearly defined
  * Inputs and outputs only go up or down one level
* It's a rigid and well-defined hierarchy that minimizes interaction and organizes the flow of communication and control
  * It is engineered not self-organizing

# Wave Theory of HCI

* 1 – Human Factors
  * Emphasizes human-machine coupling
  * Treats user as blackbox with inputs and outputs
* 2 – Cognitivist
  * Emphasizes the work/task context
  * Supports the user as an intentional agent
* 3 – Phenomenological
  * Emphasizes emergent uses of technology
  * Understands the user as a source of meanings

# Activity Theory

* Decomposes activity into "Activity, Action, Operation" hierarchy.
* These closely map to Knowledge, Rules, and Skills, respectively.



# Reconfiguring the Social Hierarchy

* How do we turn the rigid engineered system into a lightweight adaptable one?
  * Parsimony *with* variety
    * Co-locate personnel (from iceberg to ice cube)
    * Redundancy and variability of roles (flexibility of interface)
  * Automate skills (build them into the robot)
    * Dry-testing and modeling

# What does this have to do with interface design?

* Understanding context is important, but there is a problem with the unit of analysis (level of description)
  * We've designed a context but not an interface
  * More like a waterfall than co-evolution

* When we begin to look at the design of the interface itself a new set of dynamics begin to dominate
  * Perception, reasoning, situated-ness, communication

# How do we bridge the gap?