Object-Process Methodology as an Alternative for Human Factors Task Analysis

Dov Dori[1,2], Ahmad Jbara[1,3], Yongkai E. Yang[2], Andrew M. Liu[2], and Charles M Oman[2]

[1] Technion, Israel Institute of Technology

[2] Massachusetts Institute of Technology

[3] Netanya Academic College

ahmadj@technion.ac.il

Objective: We define and demonstrate the use of OPM-TA—a general model-based Task Analysis (TA) framework that uses Object-Process Methodology (OPM, ISO 19450) as an alternative for traditional TA techniques.

Background: A variety of different Task Analysis (TA) methods exist in Human Factors Engineering, and several are often applied successively for a broad task representation.

Method: Using OPM-TA, we modeled how an International Space Station (ISS) astronaut would support extravehicular activities using the existing robotic arm workstation with a new control panel and an electronic procedure system during both nominal and off-nominal operation conditions. The modeling employed both traditional TA methods and the new OPM-TA approach, enabling a comparison between the two approaches.

Results: While the initial stages of modeling with OPM-TA are the same as those of traditional TA, OPM-TA modeling yields an executable and logically verifiable model of the entire human-robot system. The OPM hierarchical set of diagrams and equivalent, automatically generated statements in a subset of natural language text specify how objects and processes relate to each other at increasingly detailed levels, from abstract to physical. Both the graphic and textual OPM modalities specify the system's architecture, which enables its function and benefits its users. To verify the logical correctness of the model, we executed the model repeatedly using OPM's animated simulation capability.

Conclusion: OPM-TA enabled unifying traditional TA methods and expanding their capabilities. The formal yet intuitive OPM-TA unifies and extends traditional TA methods that are not amenable to simulation, so it can potentially become a widely used tool for Task Analysis and human-machine procedure development and testing.

Keywords—Human Factors Engineering, Task Analysis, Electronic Procedures, Object Process Methodology, ISO 19450, Conceptual Modeling

1 Introduction

Human Factors Engineering (HFE) has long used Task Analysis (TA) methods to describe how humans perform specific physical and mental tasks. The typical TA goal in human-machine interface and procedure design is to improve task allocation, efficiency, safety, and productivity. Kirwan and Ainsworth (1992) defined TA as the study of the actions and/or cognitive processes required of an operator or a team to achieve a system goal. Task Analysis is used in Human-Computer Interface design (Crystal & Ellington, 2004) at each stage of the system development (Diaper & Stanton, 2003). TA has been extensively reviewed in the literature, and over 100 variations of TA techniques exist (Diaper & Stanton, 2003). Ritter (2019) noted that the use of models of human cognition to help design systems, indicating that these models can combine fixed mechanisms (the architecture) with task knowledge (the learned subtasks) to generate behavior. Despite TA's longstanding application in HFE, it has limitations. Practitioners often apply several complementary TA methods successively to produce a useful, comprehensive analysis. This requires them to be familiar with multiple techniques and their applications, advantages, and disadvantages, so they can select the most appropriate ones for the task. Moreover, the most appropriate combination is often hard to achieve.

As we demonstrate, TA analysis results are typically presented in spreadsheets, tables, or drawings. Each method utilizes a different, sometimes unique, format. Conceptually integrating the analyses across multiple methods can be difficult, and errors or omissions within a single representation or across different representations cannot always be identified. Detection of errors and omissions is important in the design stage, because if they go unnoticed, and are discovered only later, e.g., during prototype testing, the impact on project cost and schedule can be overwhelming. Hierarchical Task Analysis (HTA) is considered the most widely used TA technique. Stanton et al. (2013)

summarized the disadvantages of TA methods in general and those of HTA in particular. The motivation for selecting HTA as the primary method to be compared with OPM-TA is the fact that despite its most widely use, it suffers from limitations.

We can overcome major limitations of traditional TA methods by adopting a more formal and comprehensive system modeling approach and adapting it for HFE TA applications. Object-Process Methodology, OPM (Dori, 2002, 2016), is a Model-Based Systems Engineering (MBSE) approach and language, originally developed in the Systems Engineering domain and recognized as ISO 19450. HTA and other TA methods perform analysis at the conceptual level. Conceptual modeling is OPM's major forte and we leverage it to be effectively utilized for TA.

In subsequent sections, we review basic OPM concepts for HFE readers who are not familiar with OPM. Next, we present a systematic framework that is analogous to the traditional TA framework but utilizes OPM instead. The OPM-TA process begins with system and task definition and data collection, which are familiar to HFE practitioners. The next stages of OPM-TA are different, requiring the development of a formal OPM model of the entire system, including not only tasks, but also the animate and inanimate objects, both physical and informatical, as well as the processes that transform them. Model construction typically begins with a definition of top-level system goals and values and proceeds iteratively downward through successive levels of increasing detail, defining, and refining the ways processes (e.g., tasks) transform (create, consume, or change the state of) objects at each detail level. The OPM modeling software we used, OPCloud (https://www.opcloud.tech/, Enterprise Systems Modeling Laboratory, 2018; Dori et al., 2020) is a collaborative cloud-based software for OPM modeling that checks on the fly the graphical model's internal consistency and generates equivalent English text statements across all levels The URL https://sandbox.opm.technion.ac.il/ enables free and immediate experimentation with OPCloud, and readers are encouraged to experiment with it by creating the models presented in this paper. If appropriate, initial or final conditions are specified. OPCAT (Dori, 2010) is our older modeling desktop software (http://esml.iem.technion.ac.il/opcat-installation/), which can be used for the same purpose. OPCloud (Dori et al., 2020) replaces OPCAT and is aligned with both the ISO 19450 OPM and its enhancements, specified in Dori (2016).

To illustrate the OPM-TA framework, we applied it to model a robotic task performed on the International Space

Station (ISS). We briefly describe the nature of the robotic arm and the operator's task, and then walk through the OPM model at several levels of increasing detail, so readers can better understand the method. We then demonstrate the model's animated simulation process and how we used the model to design a new control panel interface required for the simulation. As some systems engineering readers may not be familiar with HFE TA methods, next, we review three established techniques, Hierarchical Task Analysis (HTA), Tabular Task Analysis (TTA), and Abstraction Hierarchy (AH). We illustrate each by applying it to our telerobotic manipulation task and compare the representations with our OPM system model. We conclude with a discussion of the relative advantages and limitations of the traditional methods for analysis, engineering, and procedure design compared with OPM-TA.

This work contributes in proposing OPM-TA—a holistic TA method that combines and extends three leading traditional TA methods. OPM-TA eliminates the efforts practitioners have to invest in finding the correct TA methods and their application order and provides a formal model of the task expressed bimodally in graphics and text, which can be simulated and validated. It also provides for model-based checking of the task at hand, verifying that the human performs the task as the system operator under both nominal and off-nominal scenarios.

2 Methods

This section introduces OPM and OPM-TA. The former is a language and methodology, and the latter is the framework that this work proposes and bases it on OPM.

2.1 Object Process Methodology

Object Process Methodology (OPM) (Dori, 2002, 2016) is a holistic conceptual modeling language and approach to the design and development of systems. OPM is defined by ISO standard 19450, providing practitioners with a normative reference document that is common to all OPM users, enabling anyone to understand any OPM model. Applications of OPM range from satellite control software (Dori & Thipphayathetthana, 2016) to large, complex socio-technical systems (Osorio et al., 2011).

OPM captures the functional, structural, and procedural aspects of the system in a single unifying model, which is bimodal: Models are expressed by both a graphical modality and a textual modality. The graphical view consists of a set of hierarchically organized, interconnected, and cross-validated Object-Process Diagrams (OPDs). Each OPD

refines and extends the OPD at the level above it and adds more details, providing for inherent complexity management and reduction. This hierarchical nature helps modelers to start modeling the underlying system with a high-level, abstract OPD at level 0, called the System Diagram (SD) and gradually drill down into more refined OPDs using the in-zooming mechanism. The same compact set of elements (entities and links) holds for all the levels of the OPD hierarchy, making all the OPDs self-similar in terms of the minimal set of concepts (and symbols) they use. A subset of English, Object-Process Language (OPL), is a structured textual specification language that is generated on the fly from the modeler's graphic input. Each OPD uses a compact set of modeling elements, which define system entities and relations among them. Fig. 1 shows the only three OPM entities: a process (ellipse), an object (rectangle), and object state ("rountangle").

Fig. 1 OPM entities. From left to right: object, process, and object state

Objects and processes are OPM things. Rectangles representing physical objects are shaded to distinguish them from informatical objects. Processes transform objects by creating them, consuming them, or changing their state. By OPM convention, process names use the gerund form and thing names are capitalized, while state names are not. Structural and procedural links graphically define relations among entities. Structural links, shown in Fig. 2 left, are between objects or between processes and support the description of static model aspects. Fig. 2 left is a screenshot of OPCloud, showing the possible structural links when attempting to connect two objects and the OPL sentence which would be generated as a result of selecting each link.

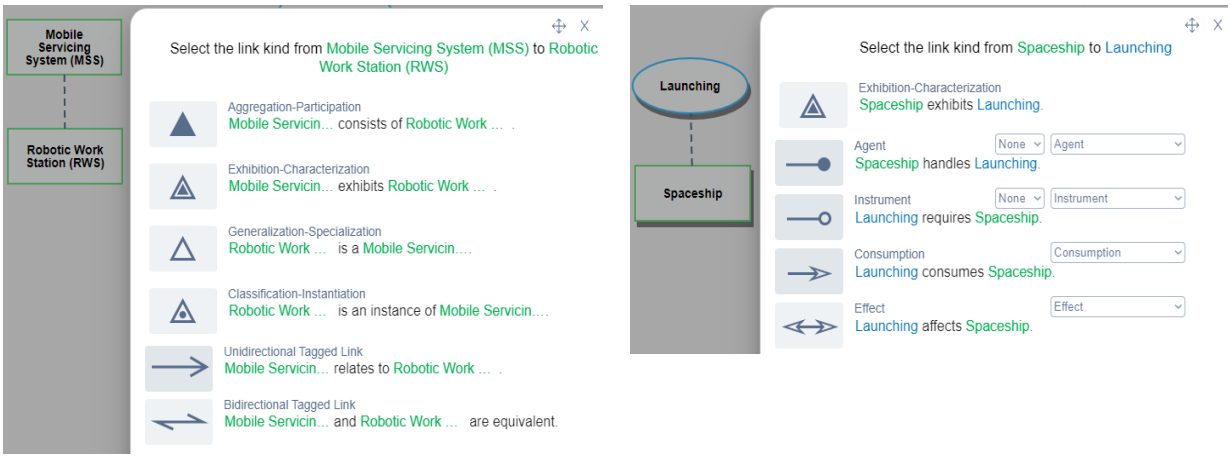


Fig. 2 Left: structural links: aggregation, exhibition, generalization, classification, and tagged links. Right: procedural links: agent, instrument, consumption, and effect

Procedural links, shown in Fig. 2 right, are between an object and a process, and they are used to model the dynamics of the system and causalities within it (Mordecai & Dori, 2014). Examples of these entities, links, and the in-zooming and unfolding refinement mechanisms appear later in this paper. OPM has only one kind of diagram and the total set of OPM elements is small – less than 20, making this graphical language highly compact and easy to learn.

OPM copes with complexity via detail-level decomposition rather than aspect-oriented decomposition, which is the common way to handle complexity in other conceptual modeling languages (Dori, 2002). Specifically, OPM's detail-level decomposition approach is in contrast with the decomposition into structure, behavior, state transitions, activities, time flow, etc. that Unified Modeling Language (UML) with its 14 diagram types and System Modeling Language (SysML) with its nine diagram types advocate. In most systems, processes and objects usually have inherently hierarchical structures that can be exposed through refinement.

In OPM, refinement is conveniently and intuitively represented graphically by providing the capability to select a process and specify or inspect its constituent subprocesses, a procedure known as in-zooming. Unfolding is another refinement mechanism, in which a specific object exposes its constituent parts, features (attributes and operations), specializations, and/or instances. Thus, the hierarchically organized OPDs are derived from each other using one of the following refinement-abstraction mechanisms: (1) Unfolding and folding of structural hierarchies of things, primarily objects, (2) Zooming into or out of the inner details of things, primarily processes, and (3) Expressing additional information through dedicated views. Every OPM model fact needs to be defined at least once, in any of

134 the OPDs in the OPD set, in order for it to be true for the entire model. OPM models can be constructed using
135 OPCAT or OPCloud. These tools can provide a simple English description in OPL for the whole model or an OPL
136 paragraph for each OPD separately (Fig. 7). They also provide a built-in visual simulation engines to support model
137 validation, verification, and testing, which is especially useful in visualizing various execution scenarios. OPM's
138 characteristics and capabilities enable clear and concise modeling of complex processes and structures in the same
139 and only diagram kind, facilitating cognitive integration of the system's structure and behavior. A basic and an
140 advanced online OPM MOOCS are available at https://www.edx.org/professional-certificate/israelx-model-based-
141 systems-engineering.

2.2 The OPM-Based Human Factors Engineering Analysis and Design Framework

142 Analogous to Stanton's framework for HTA (Stanton et al., 2013), we propose an OPM-TA as an OPM-based
143 framework for HFE task analysis and design. OPM-TA provides for including both the human and the technological
144 system she or he operates in a single conceptual model, producing a holistic hierarchical representation of the entire
145 system's architecture—the combination at all levels of its structure and behavior. The system's architecture enables
146 its function, which is performing the task at hand. The same OPM model enables simulating the system behavior in
147 nominal ("sunny day") and off-nominal, abnormal and contingency conditions. The major steps of our OPM-TA
148 framework are provided below. Examples of each step are provided in the next section.

149 The fact that the proposed approach models both the human and the technology in the same system compels the
150 modeler to think about the human role and potentially recognize additional tasks that otherwise would have been
151 missed.

A. Define the task.

152 Determine the purpose and function of the human-machine system to be designed and the conditions for its successful
153 operation.

B. Determine the system boundary.

154 Decide what is included in the human-machine system, what the system boundary is, and what objects are

environmental, i.e., what are the external objects with which the system interacts but does not control.

C. Collect data.

Below is a non-exhaustive list for guiding the collection of the required data or information:

1) The high-level processes to be performed, their order and dependencies, and their inputs and outputs,
2) The systems and subsystems required to support those processes,
3) The conditions for performing the task and each process comprising it,
4) The subprocesses (at the appropriate level of depth) involved in each process, the objects they transform and their enabling object,
5) The systems and activities outside the system boundary that might affect the activity outcome,
6) The distribution of work and responsibilities between the human or human team members and the machine,
7) The human and machine decision processes,
8) The preconditions and post-conditions of each activity, and
9) Principles, guidelines, and best practices in the pertinent domain.

D. Create the OPM System Diagram (SD).

SD is the bird-eye's view of the system, "level zero", and the root of the model's hierarchical OPD tree. In our context, it aims to present a top-level view of the human-machine task and the objects involved in performing it.

1) Start with modeling the task as the system's main function—the main process, what it is supposed to do and achieve, and the main operands, i.e., the objects which that process transforms (creates, consumes, or changes their states).
2) Add the main objects involved as enablers, i.e., agents – humans, and instruments – non-human objects.
3) Connect the objects to the top-level process using the appropriate procedural link, e.g., agent, instrument, consumption, result or effect links (see Fig. 2 right).
4) Connect objects to objects using the appropriate structural link: aggregation-participation, exhibition-

179 characterization, or generalization-specialization (see Fig. 2 left).

180 5) Add condition and event (c and e) control modifiers to appropriate procedural links to ensure correct
181 operational semantics.

182 6) Check the newly created or edited OPL sentence to verify that the graphical edit of the model is correctly
183 reflected by the sentence. If not, correct the graphical model until the text reflects your modeling intent.

E. Continuously perform animated simulation.

184 The simulation ensures that the model executes correctly, so it has to be performed after each significant graphic edit
185 operation of the OPM model, otherwise it becomes difficult to track the logical error introduced since the last correct
186 animated simulation.

F. Zoom into the process.

187 Zoom into processes that require further elaboration in order to model the next level of detail.

188 1) Arrange the subprocesses within the in-zoomed process context according to their temporal order of
189 execution. By OPM convention, the subprocesses temporal order within the in-zoomed process is top-to-
190 bottom, taking the top-most ellipse point of each subprocess as the reference point. Position parallel processes
191 (if any) at the same height. The model textual modality, OPL, also supports this temporal order.

192 2) In order to avoid clutter that makes an OPM visually too complex to comprehend, limit the total number
193 of subprocesses in each in-zoomed process to about five. The suggested number, five, is roughly based on the
194 limit of the human short-term memory capacity determined by (Miller, 1956) and mentioned in (Embley,
195 2011) in the context of OPM. If there are more, consider performing out zooming: review the process
196 definitions and assess whether some could be combined and detailed at the next, lower level. Overall, the
197 specific number is not important, but rather clutter avoidance.

G. Connect existing objects to processes.

198 The objects in predecessor OPDs are automatically depicted in the in-zoomed OPD and are connected to the outer
199 ellipse of the in-zoomed process. If an object should be linked to all the subprocesses in the in-zoomed process with

the same procedural link, leave it connected to the outer process ellipse. Otherwise, connect the object to specific relevant subprocesses using the appropriate procedural links.

H. Add lower-level objects.

1) Determine if new objects related to the subprocesses need to be added.
2) If so, create each such object, including its states if relevant, and connect the object or its appropriate state to the subprocess(es) using the appropriate procedural link.
3) If needed, add the correct control modifier (event or condition) to the procedural link.
4) Connect the newly created objects to their ancestors as parts, attributes, or specializations, using the appropriate structural link.

I. Ensure consistency.

1) Check that every process in the OPD has at least one operand, i.e., that the process transforms (creates, consumes, or changes the state of) at least one object.
2) Check the correctness of the two kinds of enabling links, instrument and agent links, which connect the enabled process and the enabling object. Use an instrument link for a non-human object that the process requires in order to execute, or an agent link if a human (agent) or a group of humans initiate or perform the process.
3) Check the newly created or edited OPL sentence to verify that the sentence reflects the graphical edit operation correctly. If not, modify the graphical model until the text reflects your modeling intent.

J. Verify pre- and post-conditions.

1) Check the correctness and completeness of the objects and their states required for performing each subprocess and the links from these objects or their states.
2) For each combination of improper precondition set for each subprocess (if any), prepare a contingency subprocess to take care of this off-nominal situation, or at least issue an informative message, specifying what prevents that subprocess from starting its execution.
3) For each combination of improper postcondition set for each subprocess (if any), issue an informative

message, specifying what prevented that subprocess from properly terminating its execution and what post-condition was violated.

K. Continue model refinement.

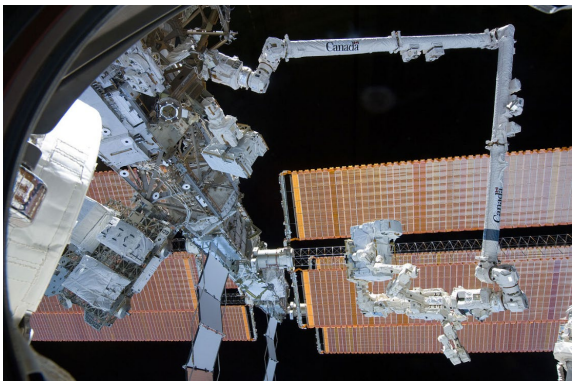
1) Recursively perform refinement operations mainly of process in-zooming and parallel object unfolding by repeating steps F through J, until the model reaches the level of detail that the modeler deems appropriate.

2) Stop the refinement when the level of detail is sufficient to fully specify the system's structure and behavior, such that it can be implemented with a minimal (ideally no) need for further explanations or interpretations.

2.3 Space Station Remote Manipulator System and MIT Training Simulator

The International Space Station is equipped with a Space Station Remote Manipulator System (SSRMS), also known as the Canadarm2 (Fig. 3). When fully extended, Canadarm2, made from titanium with exterior cladded with Kevlar fabric, is 17.6 m long. It has seven motorized joints, its mass is 1,800 kg, and its diameter is 35 cm (https://en.wikipedia.org/wiki/Mobile_Servicing_System).

Astronaut Stephen K. Robinson anchored to the end of Canadarm2 during STS-114, 2005



Canadarm2 – view of the whole arm grappling containers while near the massive solar arrays

Fig. 3 SSRMS also known as Canadarm2

Credit: Wikipedia https://en.wikipedia.org/wiki/Mobile_Servicing_System

The arm can handle large payloads of up to 116,000 kg, and it was able to assist with docking the space shuttle. The

arm has Latching End Effectors (LEEs) at both ends, each capable of grappling with special fixtures mounted on space vehicles floating nearby. The LEE at the other end grapples with power and data fixtures on the outside of the ISS or on a mobile platform that runs along the station's 60.96-meter-long truss. The SSRMS can be "walked" from one power and data fixture to another by reversing the function of the LEEs at each end. The LEE can also grapple a portable foot restraint fixture, allowing the arm to transport and stabilize astronauts who perform an extravehicular activity (EVA). The SSRMS, the power and data fixtures, and the mobile transporter are parts of the ISS Mobile Servicing System (MSS).

Fig. 4 The ISS SSRMS Robotic Workstation

An astronaut operator located inside the ISS controls the SSRMS by using one of two Robotic Workstations (RWS). Each RWS (Fig. 4) consists of a Display and Control Panel (DCP) and a pair of joystick hand controllers that provide rotational and translational manual control commands to a computer called Control Electronic Unit (CEU). The CEU transforms manual inputs from the hand controllers, or pre-programmed ("autosequence") inputs entered onboard or uplinked from the ground into appropriate commands to the motor controllers and brakes located at each arm joint and in each LEE. The operator can select a control frame of reference fixed to the LEE or payload and "fly" the arm while monitoring the movement from a camera attached to the LEE. Alternatively, the arm can be controlled in a station fixed frame.

While the SSRMS is moving, the operator must avoid joint motion singularities and continuously monitor that the position and clearance of the entire arm and attached payload relative to all ISS structures is at least 1 meter. The operator accomplishes this by using three video monitors and pre-selecting an appropriate set of camera views for

each maneuver. Interpretation of the images can be difficult, because exterior lighting is sometimes uneven, and the control frame of reference is not always aligned with the view of any particular camera. For this reason, emergency stop, and joint braking are provided. Currie et al. (2002) have considered the human factors perspective of the ISS robotic systems operations, in particular factors associated with workstation layout, human-computer interface, and adequacy of alignment cues. A communications system allows the operator to talk directly with other astronauts observing the arm motion and with others on the ground. To help maintain situation awareness, in addition to the RWS, several Portable Computer System (PCS) laptops display electronic procedures and a synthetic 3D perspective view of the arm position. The cognitive challenges and RWS procedural complexities mandate that the primary operator be highly qualified, and she or he is required to practice several hundreds of training hours.

Over the past decade, we developed a desktop computer research simulation system of the ISS SSRMS (Fig. 5). It includes 3D models of the robot arm, ISS modules and truss, attached and floating payloads, and EVA astronauts. It rendered ISS video camera scenes using a Python-based VR software called Vizard (http://www.worldviz.com/about-worldviz-virtual-reality-software/). Control modes and braking correspond to those on the real SSRMS. The simulator initially supported our NASA funded research on the effects of camera view-control axis disparity, mental rotation abilities, and sleep deprivation on operator performance and workload. We also incorporated simulations of the RWS DCP, SSRMS CEU, Fault Detection (FDIR) and video camera selection subsystems. Because the PCS laptop used as a procedure viewer on the ISS cannot communicate with the SSRMS, it must function entirely autonomously, using only keyboard entries by the crew.

The OPM-TA analysis method and model described in this paper were part of a NASA project, whose objective was to define a different kind of electronic procedure system that could directly sense and control RWS subsystem states. Through a new System Control Panel, overlaid on one of the video screens, the new electronic procedure system should provide the crew with procedure execution capability in a stepwise mode or a macro mode. Employing the OPM model, the system could detect procedural errors and direct failure recovery.

For this project, we first performed traditional task analyses of SSRMS operations. We then modeled the procedures using the OPM-based method detailed below. Because our OPM model was executable and our modeling software detected logical errors, we could validate our representation of existing procedures. As a third step, we extended the

OPM model by incorporating elements of the electronic procedure system and used it to define information requirements for the new System Control Panel. This approach is useful and beneficial especially for long-duration space flight missions with limited communications with ground control, as the astronauts can consult the model to gain insights into possible solutions to problems they encounter.

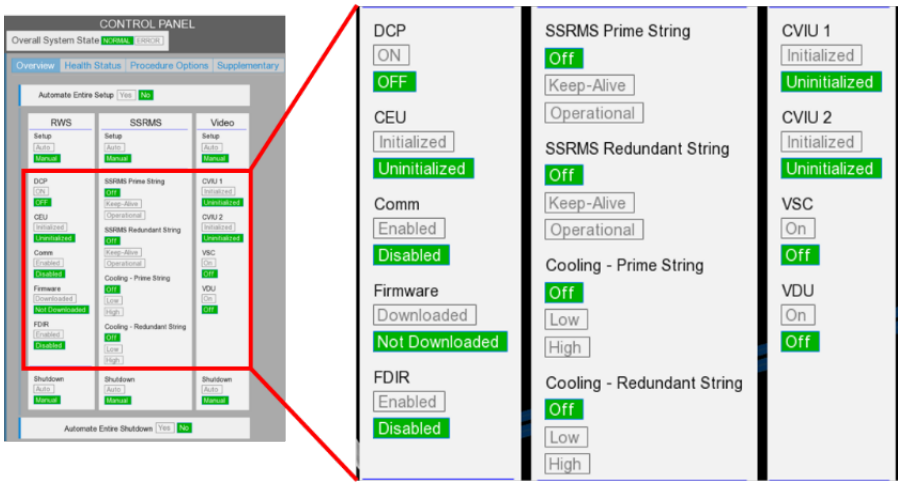


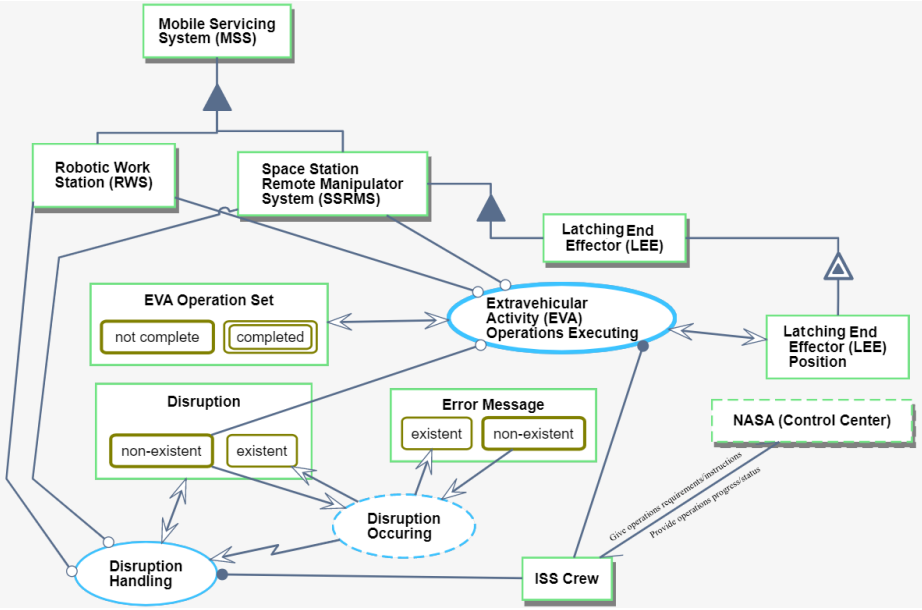
Fig. 5 MIT telerobotic training simulator

3. RESULTS

In order for readers to understand the OPM modeling process in detail, the following sections specify the SSRMS operations for supporting astronauts in performing EVAs. The modeling objectives are to (1) correctly represent the RWS procedures to be performed at an appropriate level of detail, (2) determine which new subsystems and components are needed on a new Control Panel for our improved electronic procedures system, and (3) define the necessary preconditions for each step to be successfully executed.

3.1 OPM Analysis of Telerobotic Training Simulator EVA Procedures

OPM model entities, denoted in **bold Arial** text below, refer to the corresponding objects, processes and states in OPD figures. Fig. 6 shows the OPM model System Diagram (SD)—the top-level diagram and the root of the OPD tree. Fig. 7 shows a portion of the corresponding OPL text. This SD describes **Extravehicular (EVA) Operations Executing** the function (and purpose) of the entire system.



297

298 **Fig. 6** System Diagram of Extravehicular Activity. OPM entities are exemplified in Fig. 1

NASA is environmental.
Error Message is informatical.
Error Message can be existent and non-existent.
State non-existent of Error Message is initial.
Disruption is informatical.
Disruption can be non-existent and existent.
State non-existent of Disruption is initial.
EVA Operation Set is informatical.
EVA Operation Set can be not complete and completed.
State not complete of EVA Operation Set is initial.
State completed of EVA Operation Set is final.
Mobile Servicing System consists of Robotic Work Station and Space Station Remote Manipulator System.
299 Space Station Remote Manipulator System consists of Laching End Effector.

300

301 **Fig. 7** The OPL text corresponding to the SD in Fig. 6

302 Moreover, Fig. 6 shows that the **Space Station Remote Manipulator System (SSRMS)** is clearly an essential object,

303 and it is part of the **Mobile Service System (MSS)**, which is composed of four other subsystems that are not modeled
304 here: the Mobile Base System (MBS), the Mobile Transporter (MT), the Robotic Work Station (RWS), and the
305 Special Purpose Dexterous Manipulator (SPDM). The **Latching End Effector (LEE)**, attached on the free end,
306 exhibits the attribute **LEE Position** – the location of **LEE** in 3D space, which changes during the **Extravehicular**
307 **Activity (EVA) Operations Executing** process. At this high abstraction level, a failure or error of any type is
308 represented generically as a **Disruption**, which, at any moment, is either **existent** or **non-existent**.

309 The **EVA Operations Executing** process requires the **ISS Crew** (who are the agents – human enablers in the OPM
310 terminology), the **Robotic Work Station (RWS)**, **SSRMS** (even though there are some EVAs that do not require the
311 SSRMS) and **Disruption** in its **non-existent** state. This process changes the state of **EVA Operation Set**. The **RWS**
312 provides the control interface for the robotic system onboard the **ISS**. It has two hand controllers, three video
313 monitors, a **Display & Control Panel (DCP)** with a variety of switch controls, and a **Portable Computer System (PCS)**
314 laptop. The initial state of **EVA Operation Set** is **not complete**, and the final state is **completed**. Robotics experts who
315 monitor **ISS** operations from the ground in the **NASA (Control Center)** are not part of the system, but rather part of the
316 environment, as marked by its dashed box contour. This is because the system boundary is the **ISS**. **NASA (Control**
317 **Center)** receives status updates from the **ISS Crew** and provides operational advice. This statement does not
318 essentially change the modeling and could be changed by making the contour solid.

319 If there is a **Disruption Occurring**, an external event as denoted by the dashed ellipse in Fig. 6, the state of the object
320 **Disruption** changes from **non-existent** to **existent**. **Disruption Occurring** also results in an **Error Message**. A
321 disruption halts the normal **EVA Operations Executing** process and triggers the **Disruption Handling** process. The
322 **ISS Crew** then performs **Disruption Handling** in order to return **Disruption** to its **non-existent** state, so they can
323 resume normal operations.

324 OPM is not just a language; it is also a methodology. As such, the proposed OPM-TA method is general and can be
325 applied to other tasks and systems. Fig. 6 shows that SD, the top-level diagram of the system at hand, is not just a
326 collection of graphical elements but is rather based on OPM modeling principles: Every OPM model starts with
327 determining the function of the system, which includes primarily the main process, in our case – **Extravehicular**
328 **Activity (EVA) Operations Executing.** This main process is expected to benefit a person (or a group) which is called

the beneficiary, in our case – the **ISS Crew.** Moreover, the main process of the function operates on the main object – the other part of the function, called the operand, in our case – **EVA Operation Set**. Additional stakeholders and environmental objects can also be modeled at this level, providing the context of the system operation.

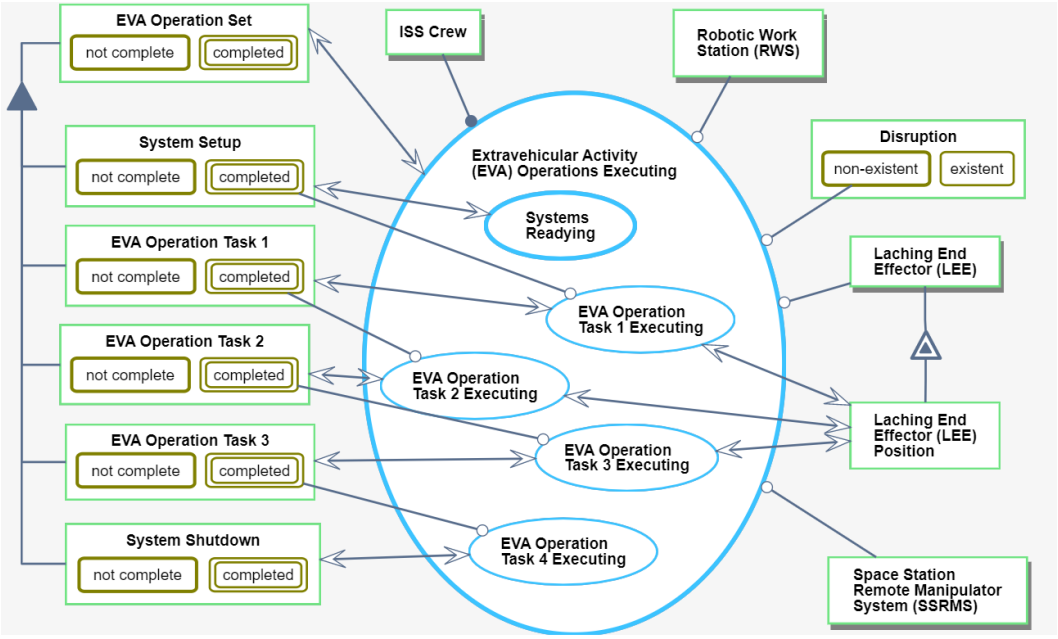


Fig. 8 EVA Operations Executing in-zoomed at SD1 – a new OPD at the next level of detail below SD

In Fig. 8, **EVA Operations Executing** is zoomed into, creating a new OPD called SD1, which shows the subprocesses of **EVA Operations Executing**, beginning with **Systems Readying**. Following OPM convention, subprocesses within an in-zoomed process are ordered temporally in a top-to-bottom order, with the top-most point of each subprocess ellipse serving as the reference point. Each of the subprocesses in Fig. 8 affects (changes the state of) a corresponding object. For example, **Systems Readying** affects **System Setup**, which is a part (subset) of the **EVA Operation Set**. Each subprocess shown in Fig. 8 is further refined at lower level OPDs by zooming into it, as indicated by the thick line of each subprocess ellipse contour in Fig. 8.

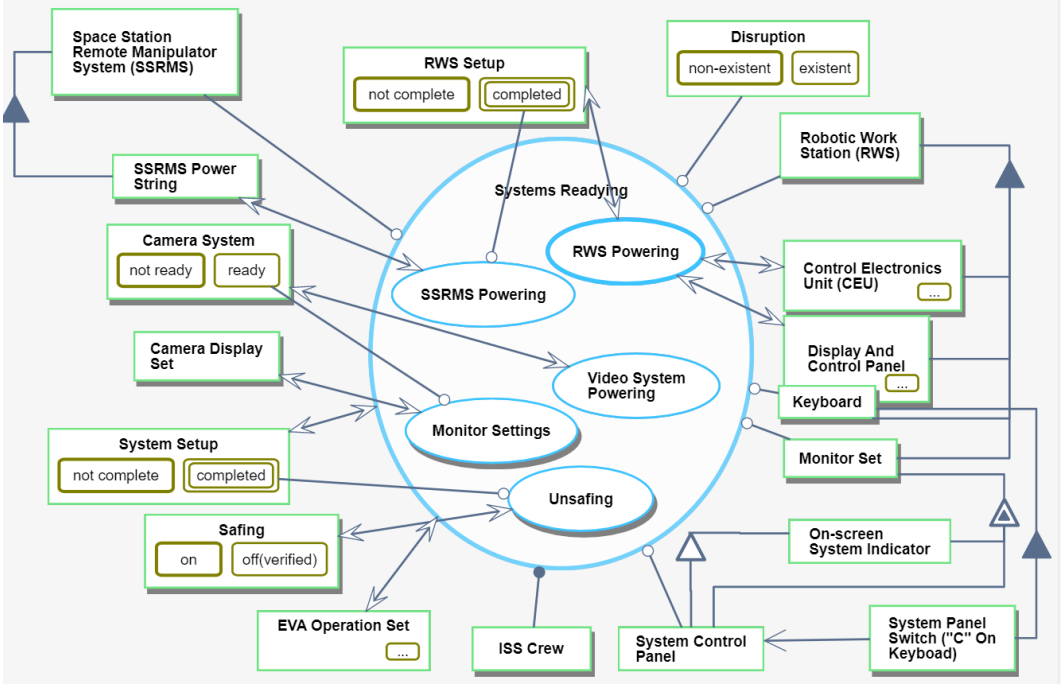
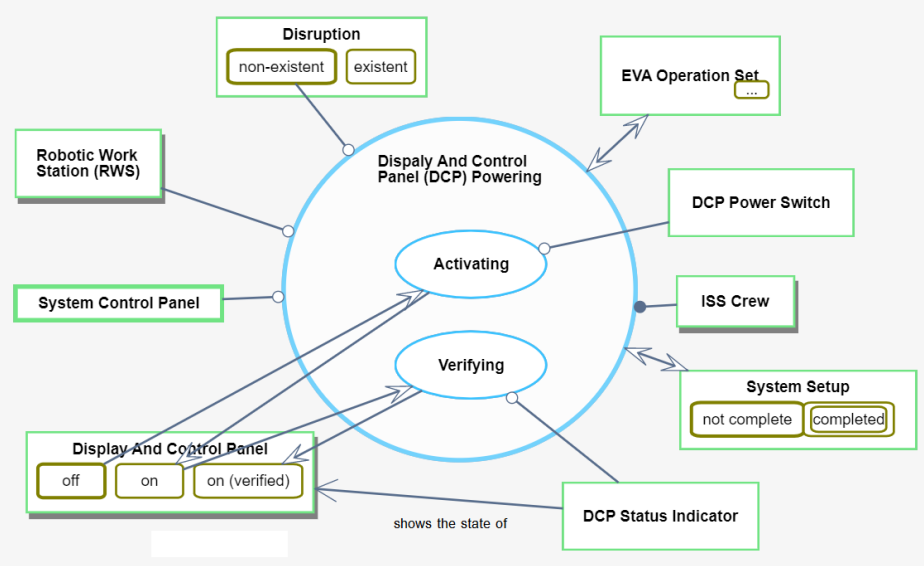


Fig. 9 System Readying in-zoomed at SD1.1 – a new OPD at the next level of detail below SD1

Fig. 9 shows an OPD in which one of these subprocesses, **System Readying**, is zoomed into. **System Readying** has its own five constituent subprocesses and appropriate lower-level physical objects, including **Keyboard**, **Monitor Set** and **System Control Panel**, with aggregation-participation relations between them and their aggregate – the **Robotic Work Station (RWS)**. Preconditions on processes are defined by procedural links in Fig. 2 right. For example, **SSRMS Powering** requires that the **RWS Setup** object be in its **completed** state, as indicated by the instrument link (the white lollipop) from that state to the process.

The methodology part of OPM recommends a top-down approach, starting with SD as an abstract, bird's-eye view of the system's function, beneficiary, and benefit. It then advocates iteratively refining SD using one of the refinement mechanisms, in-zooming or unfolding, to specify details at increasing levels till the modeler deems the level of detail sufficient. Throughout the model development, OPM principles streamline the process, making OPM-TA a general and widely applicable method well beyond the task demonstrating it in this work.



354

355 **Fig. 10** Display and Control Panel Powering in-zoomed

356 We proceed with iterative zooming into each subprocess, adding the involved objects and connecting the structural
357 and procedural links in a similar fashion until the lowest "leaf" or "atomic" level is reached, where no further
358 refinement is deemed necessary. For example, zooming into **RWS Powering** (Fig. 9) results in an intermediate OPD
359 (not shown) defining the new processes **Display & Control Panel (DCP) Powering**, **RWS CEU Initializing**, **Comm**
360 **Enabling**, **Workstation Host Software Downloading**, and **Failure Detection Enabling**. Zooming into one of these
361 processes, the **Display & Control Panel (DCP) Powering** (Fig. 10) shows the **Activating** subprocess, which requires a
362 **DCP Power Switch** and changes the **DCP** from **off** to **on**. The other subprocess, **Verifying**, confirms that the correct
363 state is set and changes the state of **DCP** from **on** to **on (verified)**.

364 The verification of the **DCP** state requires the **DCP Status Indicator**. Both processes require a human agent – the **ISS**
365 **Crew**. Thus, this OPD defines two steps for **System Readying**: **Activating** followed by **Verifying**, both denoted as
366 atomic (leaf-level) processes by their thin light blue ellipse contours. The union of the leaf OPDs, the lowest in the
367 OPD hierarchy is a flat huge OPD with no levels. It is very hard to read and follow by humans, but it can be used
368 effectively by a machine. Performed in the correct temporal sequence, it explicitly describes the entire detailed
369 **System Readying** procedure at the atomic, most detailed level.

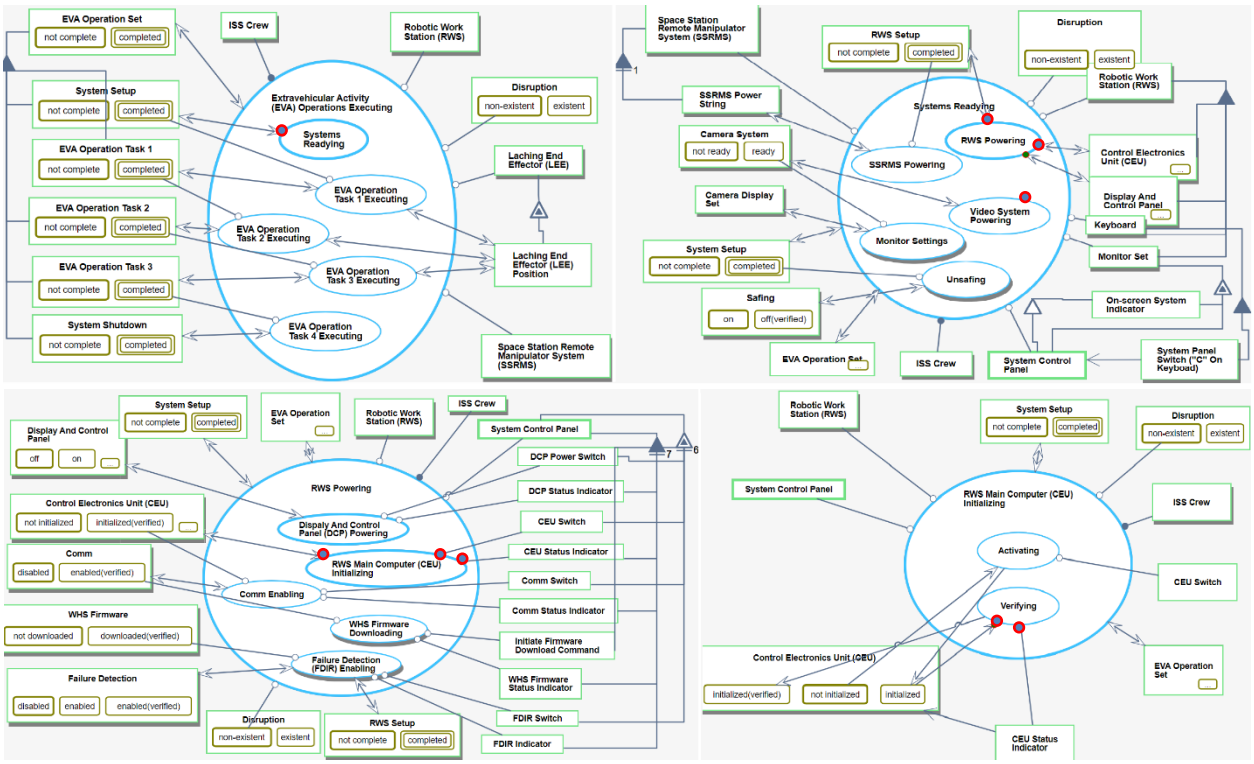


Fig. 11 Animated simulation of the EVA Operations Executing system OPM model

Using the animated execution of OPM models, Fig. 11 illustrates an animated simulation execution of the **EVA Operations Executing** model. Active procedural links, representing interactions between objects and processes, are shown with red dots running along them. The OPD at the top left is SD1 – the OPD at level 1, in which the function of the system – the main process, **EVA Operations Executing**, at the root of the OPD tree – the System Diagram (SD; level 0) was in-zoomed. To the right of SD1 is SD1.1 – the OPD in the next detail level, in which **System Readying** was in-zoomed.

Similarly, the next level down appears at the bottom left of Fig. 11, and the most detailed level is at the bottom right – this is the same OPD shown larger in Fig. 10. At this level, we can see that the simulation is currently performing the **Verifying** leaf process: There is a dot running from the state on of **Control Electronic Unit** and another dot running from the **Verifying** subprocess to the **Control Electronic Unit Status Indicator** object. The OPD tree is traversed in a depth-first manner. When a process is completed, the simulation automatically moves to the next process based on the top-to-bottom graphical arrangement of the processes.

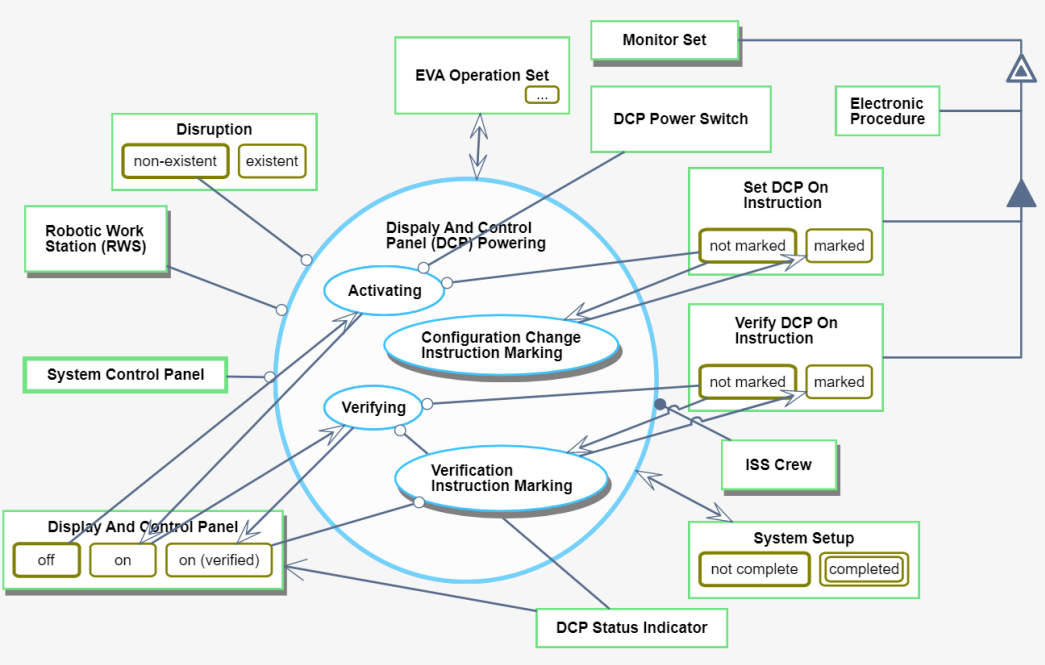


Fig. 12 DCP Processing modified with instruction marking

In the third phase of our project, the goal was to define a new Electronic Procedure system incorporating a **System Control Panel** via video camera overlay that would allow the SSRMS operator to choose between step-by-step or macro procedure execution. The system needs to keep track of what leaf steps in the procedure have already been performed. To accomplish this, we modified our OPM EVA procedure models. For example, each of the two subprocesses in the original **Display and Control Panel Powering** OPD (Fig. 10) was followed by a corresponding **Instruction Marking** step, as shown in Fig. 12.

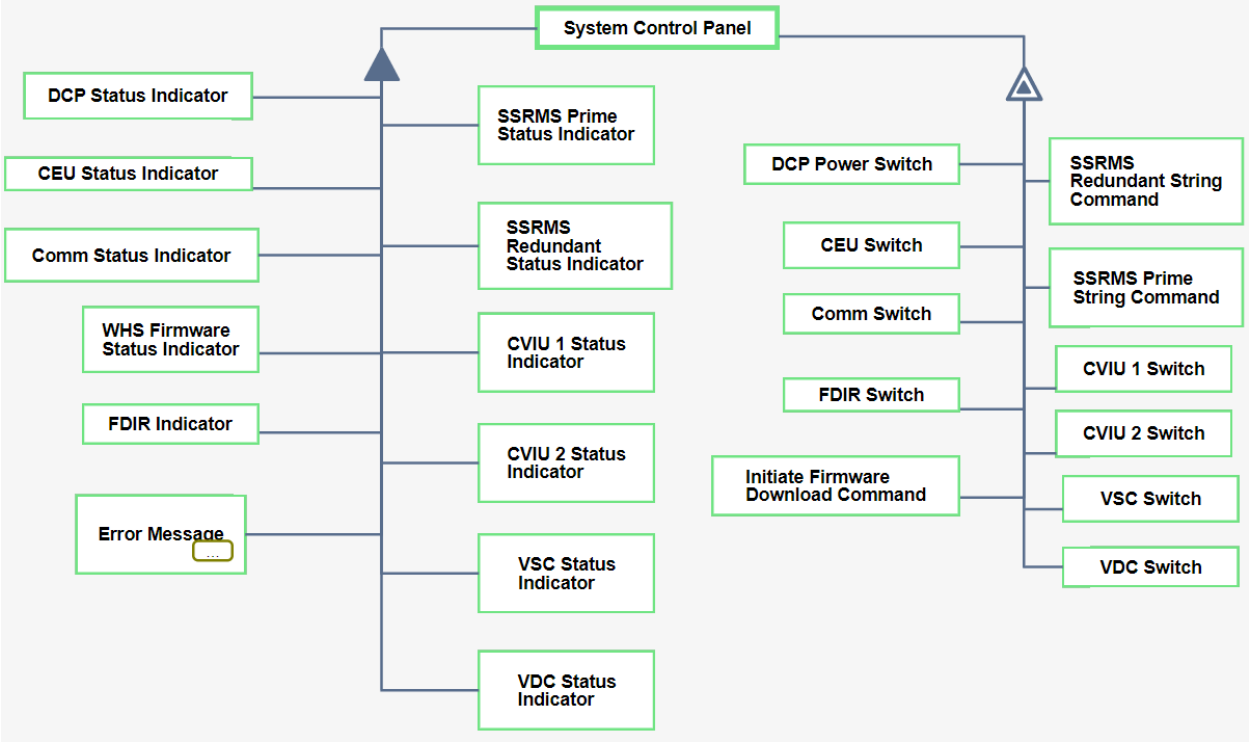


Fig. 13 System Control Panel unfolded

OPCloud's capability to unfold objects into their constituent objects proved to be useful for defining requirements for the new **System Control Panel**. An unfolded view (Fig**.** 13) defined the essential physical and informatical objects required to display as identified by the aggregation-participation symbol – the black triangle) and control (as identified by the exhibition-characterization symbol – the black-in-white triangle).

The **System Control Panel** (Fig. 5) employed a tabbed format to conserve space. The setup status for the RWS, SSRMS, and Video systems are depicted in columns defined by the object subcategories color coded in the panel's Fig. 13 informatical requirements view. A novel feature is the Operation Resume Assistant (ORA), used for failure recovery. Traditional failure recovery procedures require the operator to reset the system to the pre-failure state. This is not always straightforward, and it can require considerable prior experience. However, the OPM logic, programmed into the Electronic Procedure system, defines the preconditions, post-conditions, and instructions for every configuration change, because in OPM only a process can cause any change. By comparing the configuration before the failure and after recovery, the ORA can display a set of instructions to be performed in the correct sequence in order to resume normal operation. Further details on the Electronic Procedure version of the extended OPM model

and the System Control Panel are available in Yang (2017). Catastrophic failure modes from which there is no recovery are naturally beyond the scope of this model, and they are irreversible and can be caused by myriad reasons.

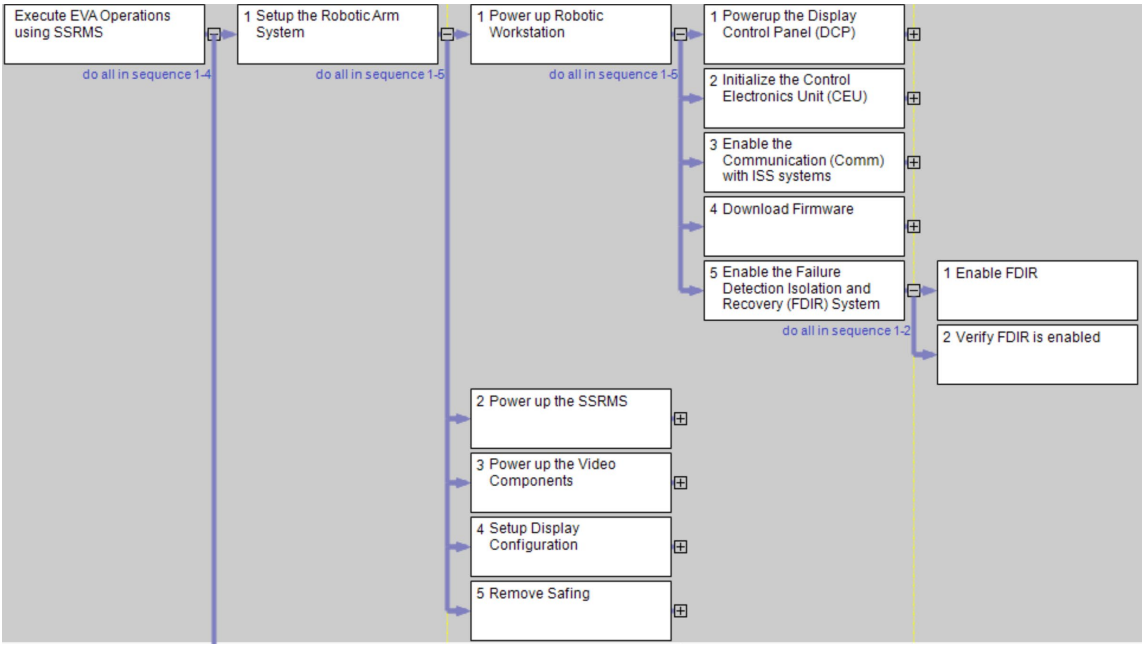
4. Discussion

For purposes of comparison, we applied also three traditional HFE analysis methods to represent the ISS SSRMS EVA operations: Hierarchical Task Analysis (HTA), Tabular Task Analysis (TTA), and Abstraction Hierarchy (AH). We briefly review these methods using our application as an example.

4.1 SSRMS EVA Analyses Using Traditional Task Analysis Methods

Hierarchical Task Analysis was introduced in 1967 to evaluate an organization's training needs (Annett & Duncan, 1967). HTA is the methodology ergonomists in the UK, and probably worldwide, use most frequently (Annett & Stanton, 2000). The key feature of HTA is that complex tasks – assignments that the person seeks to achieve – are defined by goals rather than by actions, and these may be analyzed by decomposing them into a hierarchy of goals and sub-goals, producing an extensive description of the activities required to fulfill the goals.

The framework of HTA, as suggested by Stanton and colleagues (2013), consists of the following steps: (1) define task(s) under analysis, (2) collect data, (3) determine the overall goal of the tasks, (4) determine sub-goals, (5) decompose sub-goals, and (6) analyze plans. Analyses are documented in a spreadsheet or some commercially available TA software database, and results are presented in either a graphical hierarchical diagram format or as a hierarchical list. For our HTA analysis, we employed the Stanton framework and TaskArchitect (Kern Technology Group https://www.taskarchitect.com/solutions/) to present the results in a hierarchical diagram format or in an equivalent hierarchical tabular format. Examples of both are shown in Fig. 14 and Fig. 15. The complete set of HTA diagrams and tables are available in Yang (2017). Because many HTA variants are in common use, diagram and table formats and content vary considerably across methods and HFE practitioners.



426

427 **Fig. 14** SSRMS EVA Operations HTA result Diagram Format

428

No.	Task	Plan
	Execute Extra-Vehicular (EVA) Operations using Space Station Remote Manipulator System (SSRMS)	do all in sequence 1-4
1	Setup the Robotic Arm System	do all in sequence 1-5
1.1	Power up Robotic Workstation	do all in sequence 1-5
1.1.1	Power up the Display & Control Panel (DCP)	do all in sequence 1-2
1.1.1.1	Turn DCP ON	
1.1.1.2	Verify DCP power is set as 'ON'	
1.1.2	Initialize the Control Electronics Unit (CEU)	do all in sequence 1-2
1.1.2.1	Initialize CEU	
1.1.2.2	Verify that CEU initialization is complete	
1.1.3	Enable the Communication (Comm.) with ISS systems	do all in sequence 1-2
1.1.3.1	Enable Comm.	
1.1.3.2	Verify that Comm. is enabled	
1.1.4	Firmware Download	do all in sequence 1-2
1.1.4.1	Download Firmware	
1.1.4.2	Verify Firmware Download is completed	
1.1.5	Enable the Failure Detection Isolation and Recovery (FDIR) System	do all in sequence 1-2
1.1.5.1	Enable FDIR	
1.1.5.2	Verify FDIR is enabled	

429

430 **Fig. 15** SSRMS EVA Operations HTA result Diagram Format

431 Although HTA delineates task goals and procedures, it does not define the control, display, or informatics
432 requirements of the task. To address some of the limitations of HTA, we applied the Tabular TA (TTA). Introduced
433 by Kirwan (1994), TTA is an HTA variant that analyzes each task step by defining the enablers, feedback or error

signals, displays and controls used, and other information for tasks represented at the physical level. We augmented our HTA by defining it in a columnar format with the additional information. This method results in tables spanning many pages, and though potentially comprehensive, it results in a database that is relatively difficult to navigate and comprehend at the system level. Moreover, the combined detailed HTA and TTA analysis did not enable us to enumerate all the relationships between the subsystem states and the dependencies between tasks. Specifically, we could not use this combined representation to answer questions such as "Is subsystem A being set to 'On' in Task A a prerequisite of Task B, or is it the case that the designer simply preferred Task B to take place after Task A in the procedure with no apparent reason?" Furthermore, neither HTA nor TTA of each subsystem and component explicitly states the purpose of the operations and the means for achieving them.

Abstraction Hierarchy (AH) is the first stage of the Cognitive Work Analysis method developed by Vicente (1999). The goal of AH is to identify means-ends relationships and task constraints. AH usually describes the system at five levels, proceeding from abstract to physical. It starts with the overall purpose, values and functions, and proceeds downward to physical functions and objects (Xiao et al., 2008). Means-ends links in an AH model show how individual components (means) influence the overall abstract objectives (ends) of the system. Usually, a set of resources or constraints at one level support functions, value or purpose at a higher level (Lintern, 2013). Thus, each link reveals the resources or constraints that one must use at some level to satisfy resources or constraints at the preceding level.

Fig. 16 shows a simplified AH of SSRMS EVA Operations. One limitation of AH is that real-world systems have many low-level components. Hence, the AH representation has many lines between these levels crossing each other, making the representation visually complex and difficult to grasp for anyone and especially for uninitiated practitioners. The diagram can be simplified by aggregating elements into broader categories at each level, making the model visually simpler, but this dilutes the information content, reducing the usefulness of AH analysis.

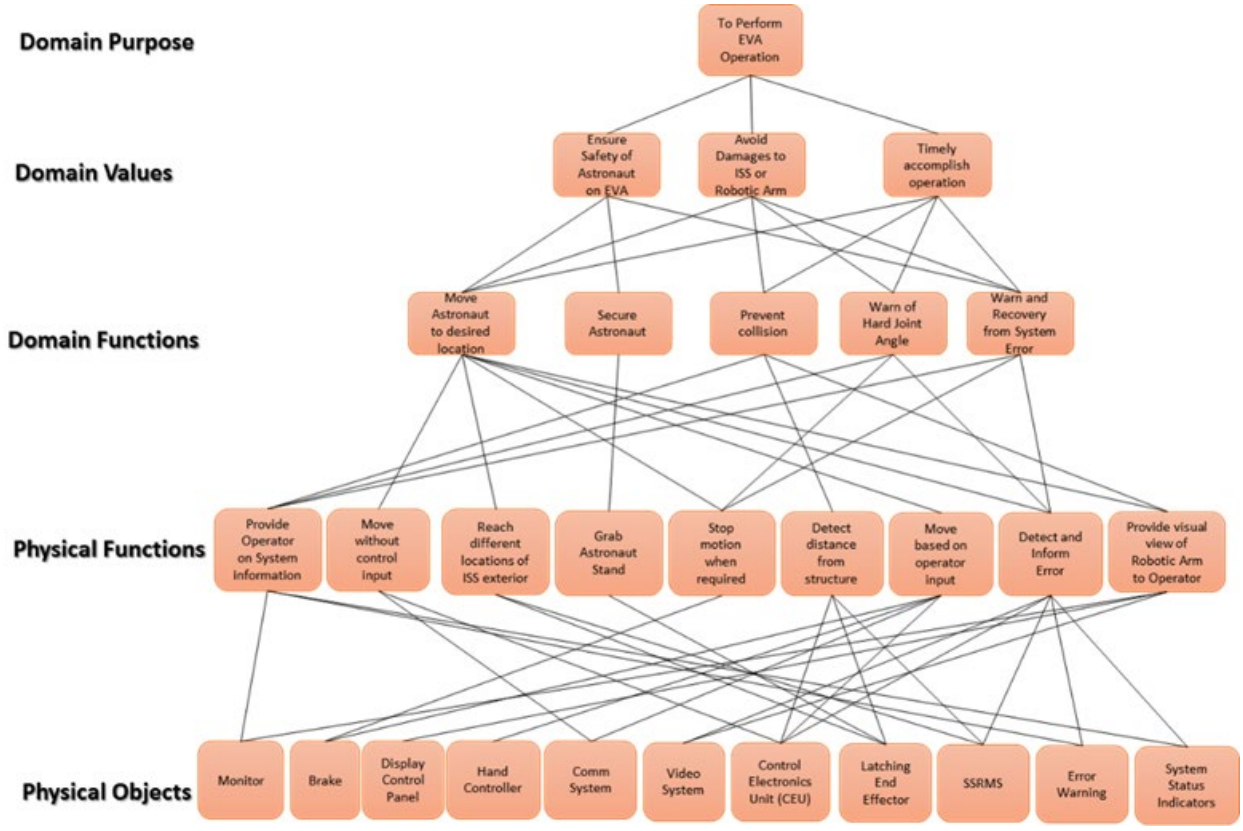


Fig. 16 A simplified Abstraction Hierarchy (AH) representation of SSRMS EVA Operations

4.2 Comparing Traditional Task Analysis with OPM-TA

Similar to AH, OPM-TA defines the system's top-level goals and proceeds downward, conceptually defining objects and processes that are both physical and informatical at all levels of detail. Unlike the traditional TA methods, OPM-TA galvanizes the representation at each step by checking for logical completeness and incorporating the following features, which traditional TA methods do not have:

(1) The human agents who participate in each step,

(2) The robotic subsystems and components involved and how their states change by each subprocess,

(3) Supporting information, such as displays and messages they present for each situation,

(4) Required preconditions, such as “Turning System A to on requires that System X be operational”,

(5) Postconditions for processes, such as the creation of a new window panel for performing the subsequent step,

(6) The structural relations between objects and the procedural relations between processes and objects, and in a complete model,

(7) What to do in cases of malfunction for any step and under various conditions.

The last item is most crucial, because the system must be ready for all possible failures, and precise prescription of how to mitigate each problematic situation is especially critical in space systems.

4.3 Traditional Task Analysis Limitations

Applied successively, our combined HTA, TTA and AH representations appeared complementary. However, several limitations and deficiencies were evident: Different off-nominal scenarios, such as recovery from failure, had to be represented separately from the nominal, “sunny day” operation scenario. Therefore, they required separate analyses that were difficult to integrate. Although we had documented the SSRMS EVA tasks in detail, such that the combination of HTA, TTA and AH analysis results seemed comprehensive, we worried that these representations might be incomplete and had no way to ascertain the opposite. Taken together, they did not result in a computable representation that could be checked for logical correctness or for logical completeness. We could not code and execute them to verify that the temporal behavior of the model matched that of the real system in sufficient detail.

When we gave the HTA/TTA/AH databases, diagrams and tables to engineers in our group, they found the heterogeneous materials confusing and difficult to integrate. Lacking a unified, holistic description of the entire SSRMS EVA human-machine system, it was not obvious how to use the information to modify the existing SSRMS EVA system to incorporate the new Electronic Procedure System or to specify the elements required on the new System Control Panel.

More limitations have been mentioned in the literature besides the one that was evident in this work. Even though HTA is considered the most popular task analysis technique and perhaps the most common of all available human

488 factor ones (Annet, 2004; Stanton, 2006), Stanton and colleagues (2013) have mentioned a few disadvantages of
489 HTA. For example, HTA provides mainly descriptive information, but it does not particularly support the details of
490 the cognitive components of the task performance, it can be time consuming for large and complex tasks, and it
491 contains little that can be used for design solutions. The TTA method is considered very time-consuming, which may
492 lead to problems of produced data reliability.

4.4 Advantages of the OPM-TA Approach

493 **Concise distinction between objects and processes:** Both traditional HTA and OPM reduce apparent system
494 complexity by describing the system in a hierarchical fashion. However, unlike HTA, OPM makes a formal and clear
495 distinction between objects and processes in a system. It provides unambiguous representations of the operands,
496 instruments, and the dynamic actions (processes) involved. By using formally defined structural and procedural links,
497 OPM represents all modeled system elements unambiguously, facilitating holistic comprehension of both goals and
498 means. The behavior of some objects – notably humans – varies stochastically, but if necessary, OPCloud can
499 simulate stochastic states in an OPM model. Traditional TA techniques represent objects and processes, but no formal
500 distinction is made between them. Lacking formal definitions, some objects and processes are only implicitly
501 represented in TA, so their existence has to be inferred. Because some elements of the system architecture may
502 inadvertently be missing in the TA description, the representation is not computable, and practitioners may have
503 difficulty understanding or predicting how the system will behave.

504 **Model testing via animated simulation:** Unlike traditional Task Analysis, OPM models in OPCAT or OPCloud are
505 executable in the sense that they can be visually simulated and executed, not just qualitatively but also quantitatively
506 (Li et al., 2019) .

507 One can check the model for logical correctness, and temporal animations can reveal errors at the various model
508 levels, much as one can debug a computer program. An unexpected halt in the simulation highlights potential logical
509 errors or fallacies that must be addressed. As the model building proceeds down to the physical level, where
510 predictions can be compared with real system behavior, the simulation anneals the model in the sense that it stabilizes
511 it and make it converge to the real system. This reduces the risk that designs based on the OPM analyses will contain
512 errors that adversely affect project cost and schedule. The earlier an error is detected, the less costly it is to correct it.

The cost is known to increase exponentially with the system development stage. In safety-critical systems, such as the ISS SSRMS, faulty procedures or controls could lead to hazardous situations if a modeling error goes undetected. The animated simulation can also help all system stakeholders, including designers and users, to visualize and understand system actions, the roles of the various objects and processes involved, and the preconditions and postconditions for each process.

Validation by construction: OPM is both a language and methodology. It has many construction rules that a modeler has to follow. Being a formal language, it is possible to enforce these OPM rules while constructing the model. This validation is the first step where errors can be caught. The next step, animated simulation, should catch additional, more subtle errors that were not detected in previous steps.

Model double-checking: OPM models are represented bimodally: the graphical and textual representations complement each other, as for every graphical fact, a textual representation is generated automatically. It is highly recommended that the modeler continuously inspects these OPL sentences and compare them with the graphical representation to make sure that what has been drawn matches her intent. This double checking process is supposed to detect errors that formal rules and simulations miss.

Improving model comprehension: Systems are inherently complex, and the way modelers reflect facts might add more complicatedness to this inherent complexity. To help modelers build more comprehensible models, we have to be able to measure their complexity. As OPM is a formal language that provides one kind of diagram that represents all three system aspects – function, structure, and behavior – it is easier to define and compute quantitative metrics for model complexity. The compared TA methods are not formal, making it difficult to define such metrics, let alone getting them assessed quantitatively.

Unified analysis method: Because of the limitations of traditional TA, HFE practitioners, as well as the engineers or safety specialists who use their results, must learn a variety of different TA methods and know which combination to apply, without really knowing if the chosen combination is the best, and how to interpret the results for design applications. While students who take human factors or cognitive engineering subjects can learn the basics of a given TA technique in several hours, many more hours of practice on examples and real-world problems are necessary before they become competent practitioners. The same is true of OPM.

Our experience has been that students in a one- or two-day OPM mini-course with OPCAT or OPCloud and example problems produce functioning models quickly. Yet, as with traditional TA, it takes many more hours of experience with a range of problems to learn to "think OPM" and achieve professional level analysis and modeling expertise. On one hand, because OPM models represent the entire human-machine system, not just part of it, as TA often does, OPM model development for a given application is likely to require somewhat more time than traditional TA methods. On the other hand, OPM practitioners and the engineering team model users need to develop proficiency in only one analysis method, not several. Moreover, OPM practitioners benefit and develop proficiency more quickly, as they use the same method repeatedly, rather than switching between different TA methods and their constellations.

Application to engineering design: When HFE practitioners present traditional TA analyses to engineering teams, often the response is: "This analysis represents a huge amount of work and certainly appears comprehensive, but how do I use these diagrams and tables in engineering design and testing?" We utilized the OPM SSRMS EVA model to develop the logic for a new type of electronic checklist system and to specify the required display and control entities on the System Control Panel. Model-based systems engineering techniques are gaining traction. Many of today's engineering students graduate with interdisciplinary competence using MBSE simulation tools, including Matlab/Simulink, SysML, UML, and in certain schools like MIT and the Technion, they use OPM. Going forward, HFE practitioners who utilize the OPM approach will be able to answer the engineering team's question by providing an executable model. They can demonstrate how to unfold OPM objects to define informatical requirements and how to derive logically correct procedures from leaf process descriptions. Matlab extensions to OPCAT (Dori, Renick, & Wengrowicz, 2016) and OPCloud are also available. That is, a modeler can attach computations to leaf processes. These computations can be done, among other options, in Matlab and linked to the model.

Embedded OPM models for failure and error detection: As described in Yang (2017), our SSRMS OPM-based model was used to develop appropriate logic and displays for our new Electronic Procedure System. While the OPM model was not incorporated into the modified RWS logic, it could be embedded into the system software and run in real time. With access to all RWS inputs and outputs, and knowing the necessary preconditions for each step, the model could evaluate all operator actions and SSRMS responses, thereby detecting operator errors and SSRMS subsystem failures. The model would thus function in a manner analogous to the mental model that a ground

566 controller uses to monitor ISS robotics activities in real time. Adding an internal system model to a human-machine
567 automation interface could prove particularly useful in future planetary missions, where real-time monitoring by
568 ground controllers is impossible. For known (and modeled) failures, the system could then either decide or advise
569 what corrective actions need to be performed in order to resume normal operation. Rather than having to recode all
570 the software decision logic, as we did in the current system, any change in the system's behavior could be
571 incorporated by updating the live underlying OPM model.

4.5 Disadvantages of the OPM-TA Approach

572 The main drawback of OPM for human factors task analysis is that the creation and modification of OPM models
573 might be time consuming as they require gaining some proficiency with OPM syntax and semantics before a high-
574 quality model can be achieved. However, considering the potential cost of undetected design or operating procedure
575 errors that are prevented thanks to representing the various task aspects in the OPM-TA model, the effort that needs
576 to be put into creating and executing the model is well-justified. Moreover, as the designer becomes proficient in
577 modeling with OPM and thinking in terms of objects and processes, the modeling process becomes agile, the quality
578 of the model improves, and the benefits increase.

579 Another disadvantage of the OPM-TA approach is that designers are dependent on a specific language, OPM ISO
580 19450, and OPM modeling software tools. This might limit the TA designer's flexibility in manipulating collected
581 data, which in traditional TA methods is done using simpler tools like spreadsheets, simplifying data portability.
582 Furthermore, the fact that practitioners apply several complementary TA methods to produce a useful analysis
583 makes their work flexible, as they can combine the best methods for the case in point. This might be more efficient
584 than following one specific method, OPM-TA, which is expected to be fit for all the task kinds.

5. CONCLUSIONS

585 HFE has traditionally employed multiple Task Analysis (TA) methods for human system analysis and design,
586 including Hierarchical Task Analysis (HTA), Tabular Task Analysis (TTA) and Abstraction Hierarchy (AH). The
587 multiplicity of methods requires practitioners to perform and refer to multiple analyses, consider their outcomes,
588 and try to mentally integrate the disparate pieces of information together into a coherent mental model of the

system. This can be a daunting task, and even taken together, these methods combined do not produce an executable model of the entire system that can be simulated and tested for logical correctness and consistency within the system under development and across its boundaries. The necessary preconditions required for understanding and designing each task and the system as a whole are often unclear, and results are presented in a form that can be difficult for others to comprehend. This is especially true for complex, interdisciplinary and safety-critical systems, where unambiguous specification and comprehension are of paramount importance. As we have shown, OPM ISO 19450, a conceptual modeling language and methodology used in model-based system engineering (MBSE), has the potential to address these limitations.

OPM advantages include the following: (1) Representation of the entire system rather than objects (humans or machines), or processes (tasks) alone, (2) ability to incorporate internal and external constraints, (3) simplicity of the modeling platform with its use of a single diagram kind (OPD) with less than 20 modeling symbols to describe each hierarchical level, and (4) the ability to detect logical errors and execute the model to predict real system behavior. We show how during simulations, various processes in the hierarchically arranged OPDs become active in the designed sequence or in parallel if preconditions are satisfied, and how they change the states of the associated objects.

This paper demonstrated how OPM was applied to the analysis of a procedurally intensive space telerobotic task, and how the model was used to specify the design of a new Electronic Checklist and System Control Panel. We showed examples of the OPM model structure along with some of the corresponding products of analyses using traditional HTA, TTA, and AH analysis methods.

There are other task analysis methods other than the three reviewed here. Readers should not conclude that OPM is invariably superior to all. Nonetheless, at least for the highly demanding robotic space application and electronic checklist design problem we considered, OPM was demonstrably superior over the traditional HTA, TTA and AH. A single OPM model incorporates more information than the three traditional task analyses combined, yielding an executable animated simulation and a comprehensive set of graphical views of the system at increasing levels of detail. The OPM model can be instrumental in making informed engineering design decisions, potentially providing a new useful tool for HFE practitioners and design engineers. The OPM-TA method was described and demonstrated in order for readers to try it themselves and adopt it if they are convinced about its advantages.

Acknowledgements

Supported by NASA under Contract NNX15AW35G. We thank NASA-JSC Robotics Instructor/Flight Controller Heidi Jennings, MIT's Dr. Raquel Galvan-Garza and students Martina Stadler and Ann Abay for all their assistance in the simulator development.

Key points

- HFE employs multiple Task Analysis (TA) methods for human system analysis and design
- This multiplicity requires practitioners to refer to multiple analyses to integrate them into a coherent mental model of the system
- This does not produce an executable model of the entire system that can be simulated and tested for logical correctness and consistency within the system
- Results are presented in a form that can be difficult for others to comprehend
- OPM has the potential to address these limitations

References

Annett, J. (2004). Hierarchical task analysis (HTA). In Handbook of human factors and ergonomics methods. CRC Press.

Annett, J., & Duncan, K. D. (1967). Task analysis and training design. http://eric.ed.gov/?id=ED019566.

Annett, J., & Stanton, N. A. (2000). Task analysis. CRC Press.

Bibliowicz, A. (2017). Object-process programming – a visual programming language for complex systems design and implementation. Ph. D. Thesis, Technion, Israel Institute of technology.

Crystal, A., & Ellington, B. (2004). Task analysis and human-computer interaction: Approaches, techniques, and levels of analysis. AMCIS 2004 Proceedings, December. http://aisel.aisnet.org/amcis2004/391.

Currie, N. J., & Peacock, B. (2002). International space station robotic systems operations - a human factors perspective. Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 46(1), 26–30.

https://doi.org/10.1177/154193120204600106

Diaper, D., & Stanton N. (2003). The handbook of task analysis for human-computer interaction. CRC Press.

Dori, D. (2002). Object-process methodolgy - a holistic systems paradigm. Springer-Verlag Berlin Heidelberg. 2002. Model-Based Systems Engineering with OPM and SysML. New York: Springer. http://www.springer.com/de/book/9781493932948.

Dori, D. (2016). Model-based systems engineering with OPM and SysML. New York: Springer, 2016.

Dori, D., Kohen, H., Jbara, A., Wengrowicz, N., Lavi, R., Levi Soskin, N., Bernstein, K., & Shani, U. (2020). OPCloud: An OPM integrated conceptual-executable modeling environment for Industry 4.0. In Kenneth R, Zoneneshain A, and Swarz RS (eds.) Systems engineering in the Fourth Industrial Revolution: how big data and novel technologies affect modern systems engineering. Wiley.

Dori, D., Linchevski, C., & Manor, R. (2010). OPCAT–An Object-process CASE Tool for OPM-based conceptual modelling. Proc. 1st International Conference on Modelling and Management of Engineering Processes, pp. 1–30. http://esml.iem.technion.ac.il/wp-content/uploads/2011/08/OPCAT-Description-and-Demo-to-MMEP-2010-Cambridge-UK-DOC.pdf.

Dori, D., Renick, A., & Wengrowicz N (2016). When quantitative meets qualitative: Enhancing OPM conceptual systems modeling with MATLAB computational capabilities. research in engineering design, 27(2), pp. 141–164.

Dori, D., & Thipphayathetthana, S. (2016). Model-based guidelines for user-centric satellite control software development. International Journal of Satellite Communications and Networking 34 (2):295–319.

ISO/PAS 19450:2015 Automation System and Integration - Object-Process Methodology. 2015. https://www.iso.org/obp/ui/#!iso:std:62274:en

Embley, D. W., & Thalheim, B. (2011). Handbook of conceptual modeling - theory, practice, and research challenges. Springer Science & Business Media.

Enterprise Systems Modeling Laboratory (2018). OPCloud. http://www.opcloud.tech. Accessed August 24 2019.

Kirwan, B. (1994). A guide to practical human reliability assessment. London: Taylor & Francis.

Kirwan, B., & Ainsworth, L. K. (1992). A guide to task analysis: The task analysis working group. London: Taylor & Francis.

Li, L., & Soskin, N. L., Jbara, A., Karpel, M., & Dori, D. (2019), Model-based systems engineering for aircraft design

with dynamic landing constraints using object-process methodology. *IEEE Access*, pp. 61494-61511. Open Access: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8710233

Lintern, G. (2013). Tutorial: work domain analysis. Cognitive Systems Design. http://ww.w.cognitivesystemsdesign.net/Tutorials/Work%20Domain%20Analysis%20Tutorial.pdf.

Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review, 63*(2), 81–97. https://doi.org/10.1037/h0043158

Mordecai, Y., & Dori, D. (2014). Conceptual modeling of system-based decision-making. *Proc. 24th Annual INCOSE International Symposium*. Las Vegas, NV, USA.

Osorio, C. A., Dori, D., & Sussman, J. (2011). COIM: An object-process based method for analyzing architectures of complex, interconnected, large-scale socio-technical systems." *Systems Engineering* 14 (4):364–82. https://doi.org/10.1002/sys.20185.

Ritter, F. E. (2019). Modeling human cognitive behavior for system design. In S. Scataglini and Đ. Paul (Eds.), DHN and posturography, (pp. 517-525, Ch. 37). London: Academic Press.

Stanton, N., Salmon, P. M., Rafferty, L. A., Walker, G. H., Baber, C., & Jenkins, D. P. (2013). Human factors methods: A practical guide for engineering and design. 2nd Edition. Ashgate Publishing, Ltd.

Stanton, N. A. (2006). Hierarchical task analysis: Developments, applications and extensions. Applied ergonomics 37(1), 55–79.

Stanton, N. (2013). Human factors methods: A practical guide for engineering and design. CRC Press.

Vicente, K. J. (1999). Cognitive work analysis: Toward safe, productive, and healthy computer-based work. CRC Press.

Xiao, T., Sanderson, p. M., Mooij, M., & Fothergill, S. (2008). Work domain analysis for assessing simulated worlds for ATC studies. Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 52:277–281. Sage Publications. http://pro.sagepub.com/content/52/4/277.

Yang, Y. (2017). Using object process methodology to develop interfaces and smart electronic procedures for simulated telerobotic operations. M.S. Thesis, Massachusetts Institute of Technology.

Wikipedia, (2019). Mobile servicing system. https://en.wikipedia.org/wiki/Mobile_Servicing_System#Canadarm2. Accessed 27 August 2019.

Dov Dori (Fellow, IEEE) received the Ph.D. degree in computer science from Weizmann

Institute of Science, Rehovot, Israel, in 1988. He is currently the Head of the Enterprise Systems Modeling Laboratory, Technion, Haifa, Israel, and a Visiting Professor with the Department of AeroAstro, MIT, Cambridge,MA,USA. Since 1993, when he invented Object-Process Methodology (OPM ISO 19450:2015), he has been central to the field of model-based systems engineering.

Ahmad Jbara received the Ph.D. degree in computer science from The Hebrew University, Jerusalem, Israel, in 2016. He is currently a Researcher with the Enterprise Systems Modeling Laboratory, Technion, Haifa, Israel. He is also a Faculty Member with the Computer Science School, Netanya Academic College, Netanya, Israel. His research interests include program comprehension, code complexity metrics, and conceptual modeling using OPM.

Yongkai Eugene Yang holds a master's degree in science in Engineering and management.

Andrew M. Liu received the Ph.D. degree in Bioengineering from University of California in 1993. He is currently a Research Scientist at MIT. His specialization and research interests include Aerospace human factors, human-machine interfaces, human-automation interaction, space telerobotics, spatial memory, learning and training of complex skills, sleep and behavioral research, augmented reality/virtual reality.

Charles M. Oman received the Ph.D. degree from Aeronautics and Astronautics department at MIT in 1972. His specialization and research interests include Aerospace human factors and physiology, human-machine interfaces, manual and supervisory control, aircraft systems and automation, space telerobotics, spatial memory, mathematical models for spatial disorientation and motion sickness