MISSION PREPARATION SUPPORT OF THE EUROPEAN ROBOTIC ARM (ERA)

Fennanda Doctor[(1)], André Glas[(2)], Zeholij Pronk[(3)]

[(1)]Software Engineer
National Aerospace Laboratory NLR, Informatics Division,
P.O. Box 90502, 1006 BM Amsterdam, The Netherlands
Phone: +31 527 8381, fax: +31 527 8210,
email: doctor@nlr.nl

[(2)]Software Engineer
National Aerospace Laboratory NLR, Informatics Division,
P.O. Box 90502, 1006 BM Amsterdam, The Netherlands
Phone: +31 527 8372, fax: +31 527 8210,
email: glas@nlr.nl

[(3)]System Engineer
National Aerospace Laboratory NLR, Informatics Division,
P.O. Box 90502, 1006 BM Amsterdam, The Netherlands
Phone: +31 527 8223, fax: +31 527 8210,
email: pronk@nlr.nl

1. INTRODUCTION

The European Robotic Arm will be one of the major European contributions to the operational capabilities of the Russian Segment (RS) of the International Space Station (ISS). ERA is developed under contract with the European Space Agency (ESA). Dutch Space is leading a European consortium developing the arm and its ground support facilities.

A major part of the operational ground-infrastructure of the ERA system is called the ERA Mission Preparation and Training Equipment (MPTE). MPTE will be used to plan, prepare, train and support ERA operations. MPTE is designed and built by the Dutch National Aerospace Laboratory (NLR) in close co-operation with Spacebel Trasys Association (STA) of Belgium, and Dutch Space.

The mission preparation subsystem is one of the major parts of the MPTE. For the operational phase, it is used to prepare flight missions, training missions and verification missions (the latter for maintenance purposes).

This paper focuses on the mission preparation software design, functions and features, in particular, the handling of a generic mission database, using the COTS software Oracle as a relational database. Furthermore the preparation of an ERA Operations Plan (EOP), preparation of onboard database parameter sets, verification of an EOP and conversion to flight compatible data will be discussed. But first a short introduction of the ERA system is given, followed by an introduction of the main MPTE functions.

2. EUROPEAN ROBOTIC ARM

The operational ERA system is composed of the European Robotic Arm itself, together with Man-Machine Interfaces (MMI) for control of ERA from inside (IVA-MMI or IMMI) and outside (EVA-MMI or EMMI) the space station. The system will initially be used to support assembly and servicing operations on the Russian Segment of the ISS.

ERA is an 11 meter long robotic manipulator arm with 7 Degrees Of Freedom (DOF) that will operate on the outside of the space station. Its main elements are one elbow joint (1 DOF), two carbon-fibre limbs, two wrist joints (3 DOF) and two end-effectors (ERA's "hands"), configured symmetrically with respect to the elbow joint. This configuration allows ERA to move across the station by grappling any one of a number of basepoints with either one of its two end-effectors.

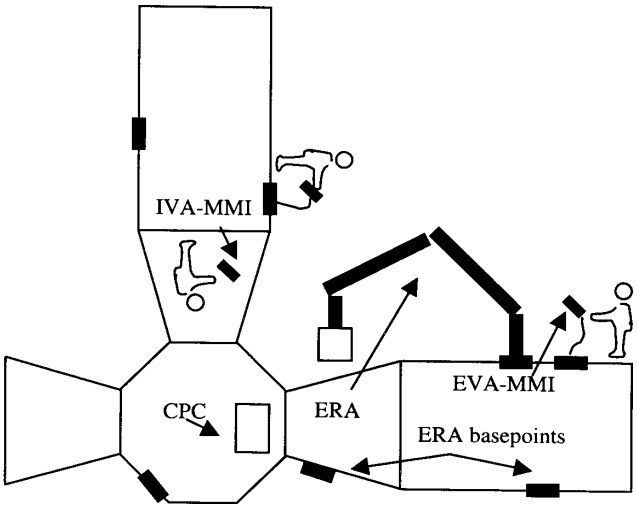

Fig. 1: ERA flight segment configuration

A limited number of basepoints will be available on the Russian Segment of ISS. Each basepoint will be able to provide ERA with power and communication lines.

During operations, ERA is controlled using the IMMI and/or EMMI. The IMMI is a laptop based user interface for use inside the space station (see Fig. 1). It provides the operator with command and control capabilities, a telemetry representation, and an artificial view on ERA and its operating environment. The EMMI is a space-qualified and robust user interface for commanding ERA from outside the space station. It has less monitoring capabilities than the IMMI but similar commanding capabilities.

ERA has three operational modes:
- Fully automatic mode; the operator uses mission specific command datasets (prepared using MPTE) built up from single commands and "Auto Sequences" (sets of closely related commands) to control ERA.
- Partially manual mode; the operator uses pre-defined, generic "Mini Auto Sequences".
- Fully manual mode; the operator directly controls the rotations of ERA's joints (one at a time) or a Cartesian motion parameter.

Under nominal conditions ERA is operated in fully automatic mode. A more thorough description of ERA operations can be found in [1].

3. MISSION PREPARATION AND TRAINING EQUIPMENT

The ERA Mission Preparation and Training Equipment provides ground support functions for ERA operations before, during and after the actual execution of a mission. In nominal situations, ERA missions will be prepared and validated on the ground, using MPTE. An ERA mission is defined as a complete end-to-end sequence of ERA operations from one period of hibernation to the next.

The following main functions of MPTE can be distinguished:
- Mission Preparation,
- Operations Training,
- Online Mission Support, and
- Mission Evaluation,
- Facility Management
- Ground Operational Software Maintenance.

Fig. 2: European Robotic Arm

Mission Preparation and operations Training would normally be done before the execution of a mission, the online mission support tools are used during mission execution, and mission evaluation is performed after the mission has finished. To support mission validation and training and to manage the MPTE as a stand-alone system, the MPTE also includes simulation, visualisation, and facility management. In addition, the MPTE supports flight and ground operational software maintenance.

The MPTE will be installed at three locations:
- S.P. Korolev Rocket and Space Corporation Energia and Mission Control Centre (RSC/E-MCC), Korolev, Moscow Region, Russia; to be used for support of flight operations (preparation, support and evaluation of flight missions)
- Gagarin Cosmonaut Training Centre (GCTC), Star City, Moscow Region, Russia; to be used for training of ERA operators (preparation of training missions, operator training based on flight and training missions, evaluation of trainees).
- ESTEC, Noordwijk, the Netherlands; to be used for training of Russian instructors and other MPTE operators, and for maintenance of flight and ground operational software.

4. INTRODUCTION TO MISSION PREPARATION

The purpose of Mission Preparation is to prepare ERA Missions. Three types of ERA Missions can be prepared:
- Flight Missions, which are the real ERA Missions that actually will be executed on the ISS.
- Training Missions, which serve only to train MPTE operators and cosmonauts for their future tasks.
- Verification Missions, which are used only to test, verify and validate ERA On-board software, for example after an update of this software.

The MPTE ESTEC facility provides the flight operational software as well as flight software and related data, which is stored in the ERA-DD. The purpose of the ERA-DD (ERA Data Dictionary) is to keep the MPTE ground software independent from the flight software. The flight software is the embedded software running inside ERA on the ERA Control Computer (ECC). The ERA-DD is an Access database that defines the ERA Command interface in a generic format. If the flight software of ERA needs to be updated, it suffices for MPTE if the ERA-DD is updated. This way, MPTE supports flight software updates, which is a crucial capability for software running in space.

The results of the Mission Preparation function are:
- ERA Data sets, which are binary uplinkable code that can be sent to the Central Post Computer (CPC).
- ERA Operations plan, the readable list of ERA Commands to be executed by the ECC.

The sub-functions of Mission Preparation can be divided into three groups. The first group is involved in data and software transfer. The second group of sub-functions is involved in geometry data handling while the third group concerns the construction of an ERA Operations Plan.

The scope of this paper is to present the implementation of manipulation, configuration control, user interface aspects, and interpretation of the building blocks of ERA operations, roughly identified by actions (based on generic ERA commands), instructions, tasks, and missions. Therefore, the functions that are directly involved in construction of an ERA Operations Plan are presented.

- **Preparation of Generic Items**. This function constructs generic procedures that serve as re-usable building blocks for ERA Operations Plans.
- **Preparation of an ERA Operations Plan**. This sub-function constructs the actual ERA Operations Plan including a dedicated link to the ERA path planning which provides the possibility to support planning and verification of detailed ERA paths.
- **Preparation of On-board Database Parameter Sets**. Setting values to ERA's many parameters.
- **Verification of an ERA Operations Plan**. This function checks if an Operations Plan is syntactically correct.
- **Mission Composition**. This sub-function generates the uplinkable code for the ERA Operations Plan.

5. SOFTWARE DESIGN OF MISSION PREPARATION

As described above, the Mission Preparation function of MPTE consists of several sub-functions. In the software design of Mission Preparation these sub-functions have been developed as separate units. Furthermore the sub-functions are divided in several layers (see Fig. 3).

The following layers have been identified:

GUI (Graphical User Interface)
The user interface functionality from the units Build ERA Operations Plan, Compose Mission and Manage Generic Items is programmed in Tcl/Tk and is brought together in the GUI layer

API (Application Programmers Interface)
The API provides functionality to the GUI in the form of high-level procedures. The functionality activated by the GUI can be found in the Application Programmers Interface layer. All non GUI functionality is programmed in C. The API layer consists of functions for the manipulation of data items in the stores ERA Mission and Generic Items. These data stores contain the ERA mission to be up-linked to the International Space Station and they contain data and building blocks for the generation of the ERA mission.
In the stores data items are defined. Per data item special functions can be identified for their manipulation such as open, close, get, set, etc. They are used by the functions in the units of Mission Preparation. To avoid the development of many functions for the manipulation of the same data items, a layer has been developed that contains these more generic functions for data item manipulation.

Data manipulation layer
The data manipulation layer contains the library functions that are used in the API layer. The library functions have an object-based approach. The data may only be accessed through the functions offered by that module. Almost every module defines a data structure, which acts as a class. The instances of such a class have only private members. Functions are provided to get and set elements from this data structure. How and where the physical data is stored is decided in the data abstraction layer.

The library offers a straightforward and consistent point of view towards the data abstraction layer. Whenever data storage items are moved or changed, this will only affect the access to the library in a minor way.

Data abstraction layer.
This is the lowest software level. It is a data abstraction layer, which must hide the physical data storage from the library functions. This means that the functions in the library are not aware of the physical source from where data is read and the physical destination (file or database) to which the data is written.

General manipulation layer.
The general manipulation layer is a vertical layer accessible to all other layers. It contains general programming modules for file, memory and string manipulation, not specific for MPTE. The error handler module is also part of support. All layers use this module to maintain consistent error handling.

Physical Data Storage
Data is stored either in files or in a database. Access to the physical data is not provided directly but through the use of the special functions in the data abstraction layer. In this way, changes in the data storage hardly affect the functions in the other layers.

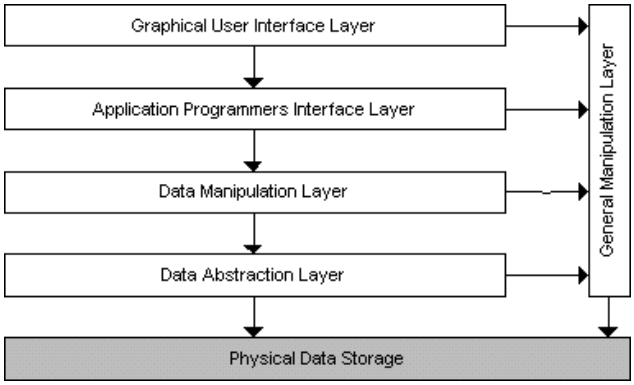

Fig. 3: Layers

6. PREPARATION OF GENERIC ITEMS

Generic Items are the basic building blocks from which ERA Missions are constructed. Generic Items are re-usable. More precisely, they can perform the same kind of operation under different circumstances, just by changing their parameter settings. An example of the application of Generic Items would be to implement the Generic Nominal functions. An example of such a procedure is to transfer a payload from one Payload Mounting Unit to another. The procedures can be divided into procedure steps, also called Generic Tasks. An example of a Generic Task would be to detach the ERA End Effector from its Base Point. It is also possible to define own generic procedures. Generic Items can range in size from very small to quite large. The smallest Generic Item is just a single ERA Command; the largest is a complete mission. The input for the Generic Items sub-function is the ERA Command definitions. The output of the Manage Generic Items sub-function flows into the sub-function Build ERA Operations Plan.

There are four types of Generic Items:

- **Instructions.** They are basically just messages, consisting of three lines of text. They are used when some sort of manual action is required before the ERA Control Computer can execute the next ERA command.
- **Actions**. They are derived from the ERA Commands definition. An Action and an ERA Command are basically the same thing e.g. FREE_MOVE (107) or GO_TO_CONTR_HOLD (200).
- **Tasks**. They from logical groupings of Instructions and Actions.
- **Missions**. They consist of Tasks, Actions, and Instructions.

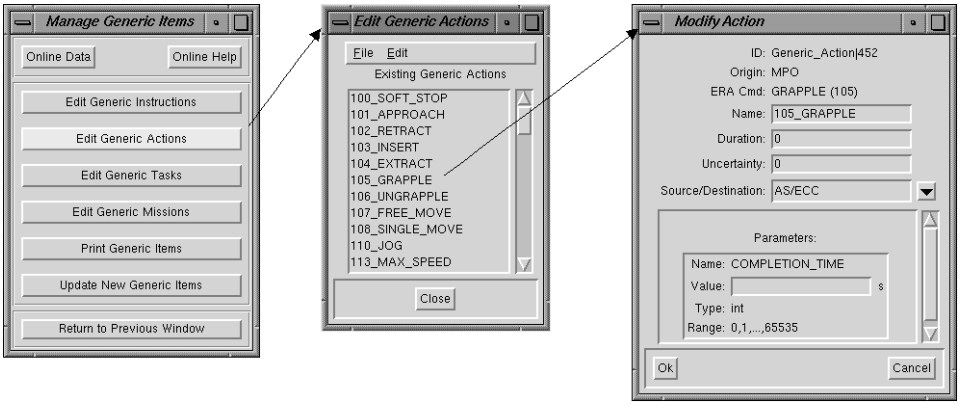

Fig. 4: Edit Generic Actions

The Generic Items are organised into a strict hierarchy. The Generic Instructions, Actions, Tasks and Missions can be created, modified, copied and deleted by the MPTE operator via a graphical user interface (see Fig. 4).
MPTE is able to receive a new version of the ERA Command definitions upon which Generic Actions are based. Importing a new version gives rise to a new version of the Generic Items. It is not possible to immediately use all the current Generic Items with the new ERA Command definitions. The current, user-defined Generic Items first have to be migrated to the new version. The objective of this migration process, called "Update New Generic Items" is to check for any inconsistencies between the Generic Items and the new ERA Command Definitions. In this process each Generic Item is checked for conflicts and if there are conflicts the operator is informed. If possible, the operator is given a change to resolve the conflicts.

7. PREPARATION OF AN ERA OPERATIONS PLAN

The ERA Operations Plan is the part of a mission that determines what the mission will actually do. An ERA Operations Plan (EOP) basically consists of a list of ERA tasks/actions divided into Auto Sequences. An auto sequence is a part of an Operations Plan that is loaded into the memory of the ECC as a single unit. The sub-function "Compose Mission" translates an EOP into a binary dataset that can be sent to the CPC and ECC.
The basic steps in building an EOP are:

1. **Initialise Mission**. Before the operator can start working on an EOP, a new mission must first be created, based upon an RS Mission Plan.
2. **Update Mission Definition**. Several "configuration parameters" of the mission can be set, such as which version of the ECC Default Database is to be used.
3. **Edit EOP**. To construct the actual EOP (see Fig. 5).
4. **Verify Mission**. After completion of an EOP, verify the mission for syntactical correctness and check several operational constraints.

These four steps will be executed sequentially, but it is possible to switch back and forth between steps 2, 3 and 4.

In the step "Edit EOP", the ERA Operations Plan is constructed by copy and paste of the Generic Tasks. The Generic Tasks implement high level operations. The advantage of this approach is that it saves a lot of effort during mission preparation when the ERA Operations Plan is build.

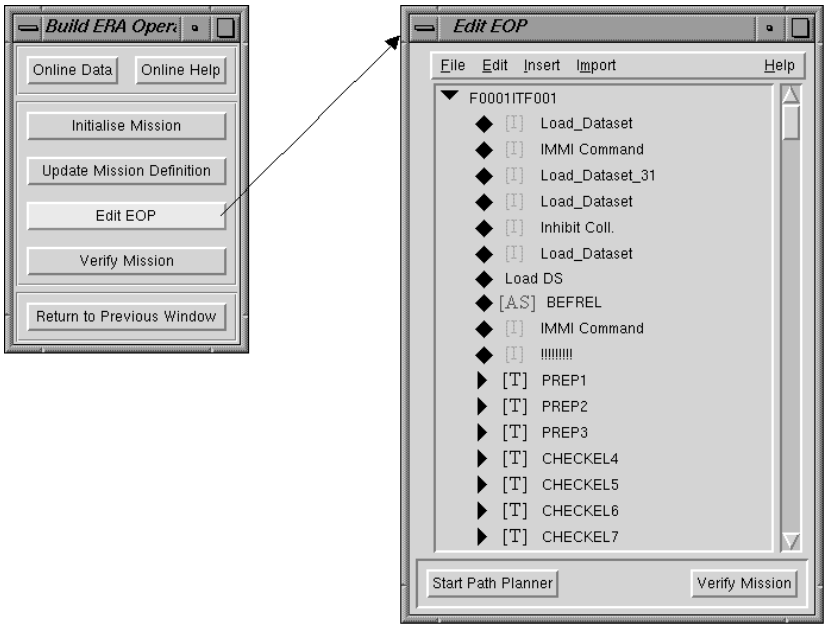

Fig. 5: Edit ERA Operations Plan

8. PREPARATION OF ON-BOARD DATABASE PARAMETERS SETS

To make ERA as flexible as possible, the behaviour of the arm is heavily parameterised. This allows the Mission Planner to adjust most of ERA's characteristics to the specific needs of an ERA Mission. At present, the total number of parameters is 8661, which gives an indication of the degree to which ERA has been parameterised. An example of an ERA parameter is IST_DB_Popin_Duration, which indicates the expected time it takes to activate the Integrated Service Tool.

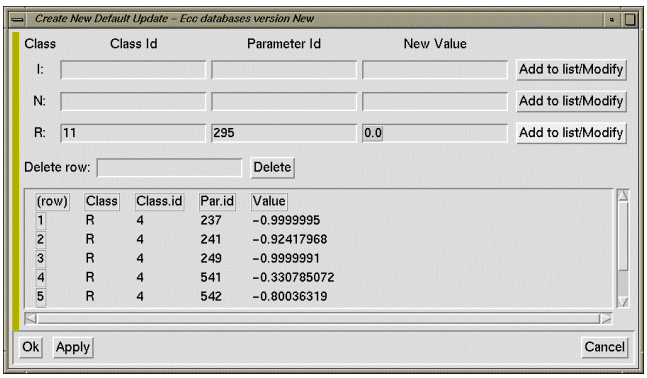

Fig. 6: Editing Default Parameters

Usually, the default parameter values will be adequate. However, it may be necessary to change some parameters for specific needs imposed by an ERA Mission. For this purpose, MPTE supports the construction of ECC Parameter Sets. The Mission Planner can interactively provide new values for any number of ERA parameters, see Fig. 6. MPTE will check if the user specified an existing parameter and if the new value lies within the acceptable range for the parameter. To do so, MPTE uses the ERA parameter definitions from the ERA-DD. The user can specify parameter updates in any order and MPTE will automatically select the most efficient way to upload them to the ECC.

9. VERIFICATION OF AN ERA OPERATIONS PLAN

MPTE provides several means of checking whether an ERA Mission is correct before sending it to the space station. One of those means is 'Mission Verification'. This process checks an ERA Operations Plan on a syntactical level. Several straightforward checks aside, the most important check made during Mission Verification is that of the ERA state vector. The ERA state vector represents the current state of the arm throughout an Operations Plan. It starts with an initial state that is updated after each ERA Command in the Operations Plan to reflect the effects of that particular command to ERA's state. A state vector consists of a list of states, which in turn consists of a unique identification and a current value. An example of a state is GRAPPLE_STATUS with the possible values of 'Grappled' and 'Ungrappled'. This state indicates whether ERA's End Effector (its hand) is holding something or not.

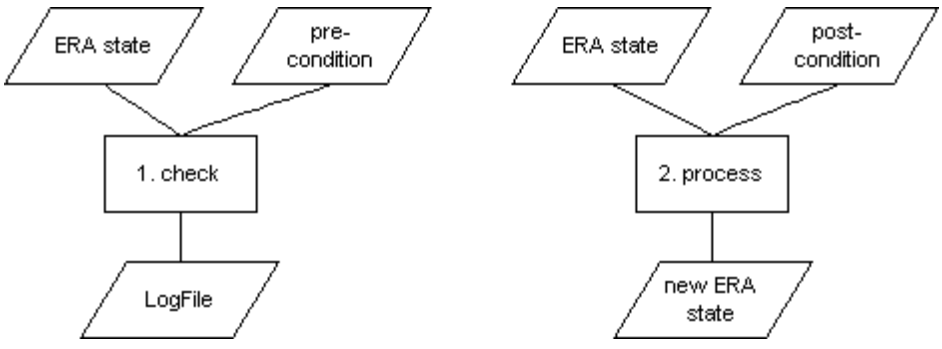

Fig. 7: Overview of mission verification

The figure above shows how the process of checking state vectors works. First, the current ERA state is checked against the pre-condition of the next ERA Command. The pre-condition of an ERA Command is in itself a state vector that specifies the allowed range of values for each relevant state. For example, the UNGRAPPLE command is only allowed if ERA's End Effector is currently attached to something (payload or space station). The pre-condition of that command, therefore, contains the state GRAPPLE_STATUS = 'Grappled'. If the current value of an ERA state is not in the range specified by a pre-condition, a warning message is written to the Mission Verification log file. Next, the effects of the ERA Command have to be processed in the current ERA state vector. To this end, each command has a post-condition that specifies the new value for each state it affects. It will not be hard to understand that, for example, the post-condition of the UNGRAPPLE command contains a state GRAPPLE_STATUS = 'Ungrappled'. After updating the ERA state vector, the pre-condition of the next command in the ERA Operations Plan can be verified.
Pre- and post-conditions of all ERA Commands are read from the ERA-DD. MPTE does not depend on the specific values of these pre- and post-conditions. Therefore, if there is an update of ERA's on-board software, MPTE can simply process the new accompanying state vectors by providing a new ERA-DD. This independence between ground and on-board software is crucial for the capability of changing ERA's software after it has been launched.

While Mission Verification can automatically perform many checks, it does not exceed the syntactical level. It remains necessary to validate a mission on a more semantical level. To this end, MPTE provides ample functionality for interactive, real-time simulation of a mission [2].

10. MISSION COMPOSITION

An ERA Operations Plan can be seen as a program to be executed by the ERA Control Computer (ECC). However, an Operations Plan is just a list of commands either in ASCII format or as a set of records in an Oracle database. Before the ECC can actually execute an Operations Plan it needs to be converted into the binary format that it understands. MPTE has a fully automatic process called 'Compose Mission' to do this. It takes each command in an Operations Plan and converts it into its binary form. The binary form of each ERA Command is defined in the ERA-DD, which provides the flexibility to change ERA Commands without having to change the software for the MPTE ground system.

11. CONCLUSIONS

The description of the MPTE functions and the MPTE design depicts a multi-purpose system for all ground operations needed to support ERA flight operations. The most important features of MPTE that were presented in this paper are listed below.

- Missions are prepared and checked on the ground, which minimises the risk of errors during flight.
- The software design of mission preparation, using a layered structure, makes the software easy to maintain.
- The design of the building blocks used within mission preparation has been done as such that the operator is not confronted with configuration management of these building blocks; this is hidden from the operator. Furthermore, the MPTE graphical user interface presented to the operator is straightforward and easy to use, which minimises the risks of making errors.
- Through the ERA-DD, flight software characteristics become adjustable parameters to MPTE. In other words, the MPTE ground software is independent of the ERA flight software. This allows MPTE to handle flight software updates, which is a crucial capability for space qualified software.
- Generic Tasks implement standard, high-level ERA operations. Constructing an ERA Operations Plan from these tasks frees the operator from the low-level details of ERA. This considerably reduces the effort of mission preparation.

12. REFERENCES

[1] Ph. Schoonejans et al., "ERA, the Flexible Robot Arm," *i-SAIRAS 1999 conference*.
[2] Z. Pronk, F.J.P. Wokke, H. Knobbout, R. Vidal, "Simulation Support of the ERA Operational Phase," *SESP 2002 conference*.