Deployment Document for Rhapsody Scripts

Checklist for deployment (all Done) :

- ☐ Port the scripts to run on Linux
- ☐ Error handling and usage help
- ☐ Documentation on which scripts does what (this document)
- ☐ Test with various conditions
- ☐ Add project name to environment variables

Deployment steps:

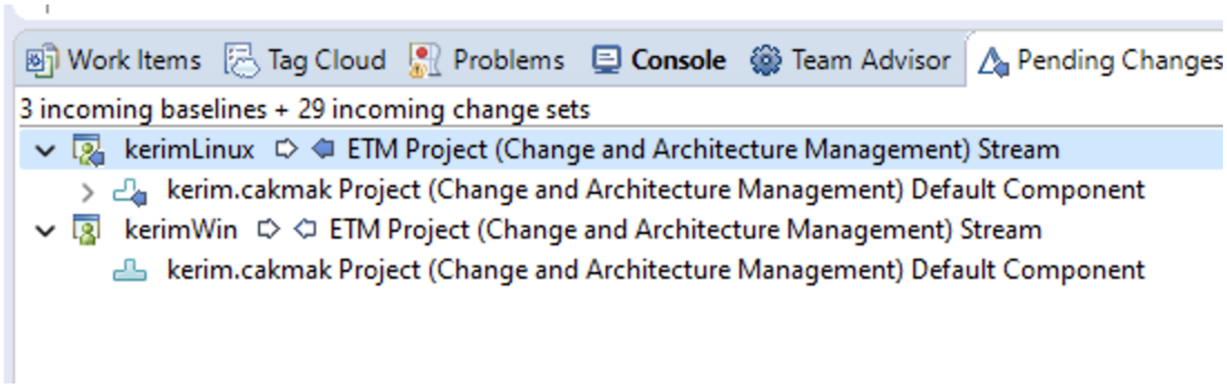
- Copy the libs.pm file under a directory in @INC. List the "include" as follows:

> *perl -E "for @INC"*

- Install EWM command line interface

- Copy the "scripts" directory on Linux
  - Test if the libs and everything works by executing the command:
    > *perl 1-createChildBlock.pl*

- Create a repository workspace to use in Linux for the EWM checkout



3 incoming baselines + 29 incoming change sets
kerimLinux ⇨ ETM Project (Change and Architecture Management) Stream
kerim.cakmak Project (Change and Architecture Management) Default Component
kerimWin ⇨ ETM Project (Change and Architecture Management) Stream
kerim.cakmak Project (Change and Architecture Management) Default Component

- Create a workspace directory for Rhapsody EWM and make a note for its path

- Create environment variables:

WORKSPACE_<*rhapsody_project_name*>:
<path_to_EWM_workspace_including_rhapsody_project_file_directory>
Example: WORKSPACE=/home/parallels/Downloads/workspace/NVL/SETitanic

RHAPSODY_FILE_DIR_<*rhapsody_project_name*>:
<name_of_the_directory_where_rhapsody_files_exist>
Example: RHAPSODY_FILE_DIR=Projekt_SETitanic_rpy/

PROJECTAREA_<*rhapsody_project_name*>: <the ccm id of the project area on the intended use environment and project>
Example: PROJECTAREA=iOjfYfj9Eeyg2Yb4jthRGQ

- Create GUIDs and RMIDs for prospecting new model elements
    - **There is a predefined workflow for this step:**
    1. Import several hundred blocks in the model without delivering the change to CCM server.
    2. Export GUIDs and RMIDs allocated for the new blocks (run extractIDs.pl script)
    3. Discard the change (import of all the blocks)
    4. Check the line endings it should be LF instead of CRLF

- In the template XMLs (in scripts directory), there are GUIDs set for the static types
  Eg:
  <BROWSER-ICON>GUID
  f685432f-691e-4ff1-be70-4d09d19457e1~ALL~BrowserIcon</BROWSER-ICON>
  <GROUP-ICON>GUID
  f685432f-691e-4ff1-be70-4d09d19457e1~ALL~GroupIcon</GROUP-ICON>

  For those GUIDs, identify how these GUIDs are set per Rhapsody project and then edit the "**GUID_Repo.txt**" file to record, each GUID per template, per rhapsody project, as follows:
  Eg:
  template,rhp_project,guid_field,guid
  block_index_template.xml,SETitanic,BROWSER-ICON,GUID
  f685432f-691e-4ff1-be70-4d09d19457e1~ALL~BrowserIcon

- Test if scm commands are working (command line interface of EWM)
- For setStereotype script, the stereotype package should be a single file in Rhapsody project (MBGrV.sbsx).
- Port stereotypes package should be placed under a single file (Ports.sbsx)

Scripts:

1-createChildBlock.pl

Usage:
```
# perl 1-createChildBlock.pl <existing_parent_block> <name_of_the_new_block>
<rhapsody_project_name>
```

Input Parameters:

1. Existing Parent Block:

This parameter should exist in the model. There is a control in the script that compares this parameter with the existing blocks to make sure it exists as a block. If not, then script execution ends with a message.

2. Name of the new block:

Can be any name. Expected format is BL_<system_level>_code (WXYZ)
W: number for 1st level
X: number for 2nd level
Y: number for 3rd level
Z: number for 4th level

3. Rhapsody Project Name: (eg: SETitanic)

This is required, cause working on more than one Rhapsody projects, the workspace and project directory will be distinguished based on the project.

Environment Variables Used in the Script :
Should be created upfront:
Eg:
```
# sudo -H gedit /etc/environment (for ubuntu)
WORKSPACE_SETitanic=/home/parallels/Downloads/workspace/NVL/SETitanic
RHAPSODY_FILE_DIR_SETitanic=Projekt_SETitanic_rpy/
PROJECTAREA_SETitanic=iOjfYfj9Eeyg2Yb4jthRGQ
```

4. WORKSPACE_<*rhapsody_project*>

This is an environment variable which should be set for each rhapsody project. For instance:

```
$ echo $WORKSPACE_SETitanic
/home/parallels/Downloads/workspace/NVL/SETitanic
```

The input parameter given in “3” is appended to the end of the “WORKSPACE_” to create an environment variable for this project.

5. RHAPSODY_FILE_DIR_<*rhapsody_project*>

This is an environment variable which should be set for each rhapsody project's file directory. For instance:

```
$ echo $RHAPSODY_FILE_DIR_SETitanic
Projekt_SETitanic_rpy/
```

The input parameter given in "3" is appended to the end of the "RHAPSODY_FILE_DIR_" to create an environment variable for this project.

6. PROJECTAREA_<*rhapsody_project*>

This is an environment variable which should be set for each rhapsody project's EWM project area id. For instance:

```
$ echo $PROJECTAREA_SETitanic
iOjfYfj9Eeyg2Yb4jthRGQ
```

The input parameter given in "3" is appended to the end of the "PROJECTAREA_" to create an environment variable for this project.

If the Project name is not provided, the script outputs a message and doesn't continue.

If the Project name is given wrong, again the script outputs a message and doesn't continue.

Important Reminders:

There is no check in the script to control if for instance, BL_HBGR_2450 is created as the child of BL_BA_2400. The script allows a command like:

```
# perl 1-createChildBlock BL_HBA_1000 BL_BGR_3455
```

Then, BL_BGR_3455 will be created as a child block to BL_HBA_1000. Or it can even be visa versa.

If this script will be used through command line, the child and parent blocks should be checked before running. The intended use is through automation and these parameters will be passed to the script, from DNG menus.

The usage in DNG is assumed to be as follows:

- Heading 1 (BL_HBA_2000)
    - Heading 1.1 (BL_BA_2400)
        - Heading 1.1.1 (BL_HBGR_2340)

When Heading 1.1.1 is selected in DNG and create block command is issued with this selection, assumption is that the widget will identify the parent automatically (BL_BA_2400) and be sent a the <existing_parent_block> parameter to the script. So the correct hierarchy will be kept automatically. As the user is not expected to select parent. The parent is identified automatically from the DNG module structure.

The only check script needs to do is whether the parent exists or not. If not, the widget should fire an error and will not continue. User should select BL_BA_2400 this time and first create BL_BA_2400 as the child to BL_HBA_2000. Then he/she selects BL_HBGR_2455 cause henceforth, BL_BA_2400 exists.

Main actions in the script

1. **Check if child block exists** with the same name in the rhapsody file contents where the parent block is defined. So, if the child block exists under this specific file, script doesn't continue. But a block with the same name can still exist in some other Rhapsody file within the same model. The script doesn't check that.
2. **Check if Parent block exists** with the same name in the Rhapsody files location. Searches all Rhapsody files for the same name. If it doesn't exist, script outputs a messages and doesn't continue. If it does, the Rhapsody file that the block is found in, will become the main file that the child block will be created in.
3. **Create a package for the child block** on the same level with the parent block. Replace "BL_" with "ST_". Keep the rest the same. If there is no "BL_" just append "ST_" in the front of the new block name. To create the package, we use two templates:
    a. package_template.xml
    b. package_index_template.xml
4. **Append both package definitions** in the rhapsody file where the parent block is defined.
5. **Append new package name** under the parent block's main package to maintain the package hierarchy
6. **Create a new block** under the new package created in step 3. To create this, use two XML templates:
    a. block_template.xml
       Definition of the block in the local Rhapsody model
    b. block_index_template.xml
       Definition of the block for the RMM server
7. **Append both block definitions** in the rhapsody file where the parent block is defined.
8. **Append the new block name** under the newly created package to maintain package hierarchy
9. **Create a directed composition** between parent block and new child block
10. Fix Rhapsody indices.

How to Verify:

Open the Rhapsody model and check if

- The package exists (that should be created with the block) under the parent block’s package
- The block exists under the new package
- There exists a new part under the parent block with the name of the new child block.

2-linkToRequirement.pl

Usage:
perl 2-linkToRequirement.pl <*model_element_name*> <*model_element_type*> <*rhapsody_project_name*>

Input Parameters:

1. Existing Model Element:

This parameter should exist in the model. There is a control in the script that compares this parameter with the existing elements to make sure it exists. If not, then script execution ends with a message. This could be any model element, a block, package, part, port etc..

2. Type of the Model Element :

Type is required to create the link. There is a check to see if the entered value for type is valid or not. If not, system prints a message:

```
Model element or type not Found. Please make sure you entered right element and type:
please enter Class for Block
Subsystem for Package
Port for Proxy Ports
Part for Parts
etc...
```

3. Rhapsody Project Name: (eg: SETitanic)

This is required, cause working on more than one Rhapsody projects, the workspace and project directory will be distinguished based on the project.

Environment Variables Used in the Script :
Should be created upfront:
Eg:

```
# sudo -H gedit /etc/environment (for ubuntu)
WORKSPACE_SETitanic=/home/parallels/Downloads/workspace/NVL/SETitanic
RHAPSODY_FILE_DIR_SETitanic=Projekt_SETitanic_rpy/
PROJECTAREA_SETitanic=iOjfYfj9Eeyg2Yb4jthRGQ
```

4. WORKSPACE_<*rhapsody_project*>

This is an environment variable which should be set for each rhapsody project. For instance:

```
$ echo $WORKSPACE_SETitanic
/home/parallels/Downloads/workspace/NVL/SETitanic
```

The input parameter given in “3” is appended to the end of the “WORKSPACE_” to create an environment variable for this project.

5. RHAPSODY_FILE_DIR_<*rhapsody_project*>

This is an environment variable which should be set for each rhapsody project’s file directory. For instance:

```
$ echo $RHAPSODY_FILE_DIR_SETitanic
Projekt_SETitanic_rpy/
```

The input parameter given in “3” is appended to the end of the “RHAPSODY_FILE_DIR_” to create an environment variable for this project.

6. PROJECTAREA_<*rhapsody_project*>

This is an environment variable which should be set for each rhapsody project’s EWM project area id. For instance:

```
$ echo $PROJECTAREA_SETitanic
iOjfYfj9Eeyg2Yb4jthRGQ
```

The input parameter given in “3” is appended to the end of the “PROJECTAREA_” to create an environment variable for this project.

If the Project name is not provided, the script outputs a message and doesn’t continue.

If the Project name is given wrong, again the script outputs a message and doesn’t continue.

Important Reminders:

Since the DNG part of this functionality not developed yet, the “requirement” end of this link is right now hard coded in the code as :

```
my $targetLink = "https://jazz.net/sandbox01-rm/resources/BI_H6ar9SnLEe2zB-tVgYH0_w";
```

This is the address of an artifact in DNG in the jazz.net sandbox. But in real life scenario, it should be passed as a parameter to the script through the widget.

3-setStereotype.pl

Usage:
perl 3-setStereotype.pl <*existing_block_name*> <*rhapsody_project_name*>

Input Parameters:

1. Existing Block Name:

This parameter should exist in the model. There is a control in the script that compares this parameter with the existing blocks to make sure it exists. If not, then script execution ends with a message.

2. Rhapsody Project Name: (eg: SETitanic)

This is required, cause working on more than one Rhapsody projects, the workspace and project directory will be distinguished based on the project.

Environment Variables Used in the Script :
Should be created upfront:
Eg:

```
# sudo -H gedit /etc/environment (for ubuntu)
WORKSPACE_SETitanic=/home/parallels/Downloads/workspace/NVL/SETitanic
RHAPSODY_FILE_DIR_SETitanic=Projekt_SETitanic_rpy/
PROJECTAREA_SETitanic=iOjfYfj9Eeyg2Yb4jthRGQ
```

3. WORKSPACE_<*rhapsody_project*>

This is an environment variable which should be set for each rhapsody project. For instance:

```
$ echo $WORKSPACE_SETitanic
/home/parallels/Downloads/workspace/NVL/SETitanic
```

The input parameter given in “2” is appended to the end of the “WORKSPACE_” to create an environment variable for this project.

4. RHAPSODY_FILE_DIR_<*rhapsody_project*>

This is an environment variable which should be set for each rhapsody project’s file directory. For instance:

```
$ echo $RHAPSODY_FILE_DIR_SETitanic
Projekt_SETitanic_rpy/
```

The input parameter given in “2” is appended to the end of the “RHAPSODY_FILE_DIR_” to create an environment variable for this project.

5. PROJECTAREA_<*rhapsody_project*>

This is an environment variable which should be set for each rhapsody project's EWM project area id. For instance:

```
$ echo $PROJECTAREA_SETitanic
iOjfYfj9Eeyg2Yb4jthRGQ
```

The input parameter given in "3" is appended to the end of the "PROJECTAREA_" to create an environment variable for this project.

If the Project name is not provided, the script outputs a message and doesn't continue.

If the Project name is given wrong, again the script outputs a message and doesn't continue.

Important Reminders:

This script assumes that the format of the block name applies to the standard naming convention: *BL_<system_level>_code.*

If that is the case, then querying the MBGrV package in Rhapsody Project, the corresponding stereotype is found and set for the relevant block.

One main prerequisite for this script is that MBGrV package should be a single Rhapsody file. It should not be broken down to units.

If the block name is not compliant with the naming convention, the stereotype setting will not work and the script will return null.

4-createPorts.pl

Usage:
perl 4-createPorts.pl <*existing_block_name*> <*new_port_name*> <*port_stereotype*> <*rhapsody_project_name*>

Input Parameters:

1. Existing Block Name:

This parameter should exist in the model. There is a control in the script that compares this parameter with the existing blocks to make sure it exists. If not, then script execution ends with a message.

2. New Port Name:

Any name that can be used to define the new proxy port to be created in the existing block.

3. Port Stereotype:

Can only have the following values:

- IF_Mechanik
- IF_Hardware
- IF_Software
- IF_Weitere
- IF_Daten
- IF_Fluid

Any value other than these, is not allowed. The script will exit with an output message.

4. Rhapsody Project Name: (eg: SETitanic)

This is required, cause working on more than one Rhapsody projects, the workspace and project directory will be distinguished based on the project.

Environment Variables Used in the Script :
Should be created upfront:
Eg:

```
# sudo -H gedit /etc/environment
```
(for ubuntu)
```
WORKSPACE_SETitanic=/home/parallels/Downloads/workspace/NVL/SETitanic
RHAPSODY_FILE_DIR_SETitanic=Projekt_SETitanic_rpy/
PROJECTAREA_SETitanic=iOjfYfj9Eeyg2Yb4jthRGQ
```

5. WORKSPACE_<*rhapsody_project*>

This is an environment variable which should be set for each rhapsody project. For instance:

```
$ echo $WORKSPACE_SETitanic
/home/parallels/Downloads/workspace/NVL/SETitanic
```

The input parameter given in “2” is appended to the end of the “WORKSPACE_” to create an environment variable for this project.

6. RHAPSODY_FILE_DIR_<*rhapsody_project*>

This is an environment variable which should be set for each rhapsody project’s file directory. For instance:

```
$ echo $RHAPSODY_FILE_DIR_SETitanic
Projekt_SETitanic_rpy/
```

The input parameter given in “2” is appended to the end of the “RHAPSODY_FILE_DIR_” to create an environment variable for this project.

7. PROJECTAREA_<*rhapsody_project*>

This is an environment variable which should be set for each rhapsody project’s EWM project area id. For instance:

```
$ echo $PROJECTAREA_SETitanic
iOjfYfj9Eeyg2Yb4jthRGQ
```

The input parameter given in “3” is appended to the end of the “PROJECTAREA_” to create an environment variable for this project.

If the Project name is not provided, the script outputs a message and doesn’t continue.

If the Project name is given wrong, again the script outputs a message and doesn’t continue.

Important Reminders:

The Block should exist but port should not exist. There cannot be two ports with the same name in the same Rhapsody file. The script has a control for this.

The predefined port stereotype will be set to the new port. The port stereotypes in the Rhapsody project should be set as a separate unit and be stored in the “Ports.sbsx” file.

Main Actions in the Script:

1. Identify in which Rhapsody unit file the block exists.
2. Create the new Port definition for local Rhapsody Project from the template called “port_template.xml”.
3. Create the new Port definition for RMM server from the template called “port_index_template.xml”.
4. Append the port definition for local Rhapsody project and RMM server in the unit file where the block is defined.

5. Append the port GUID to the block definition
6. Set the port stereotype given as the input parameter
7. Fix rhapsody indices.

5-connectPorts.pl

Usage:
# perl 5-connectPorts.pl *<existing_from_part_name>* *<existing_from_port_name>* *<existing_to_part_name>* *<existing_to_port_name>* *<rhapsody_project_name>*

Input Parameters:

1. Existing From Part Name

The part name (not the Block but the part from which the block is created). The part should exist under a parent block.

For instance:
BL_BA_2400 has a part called PT_HBGR_2470. This part is the part representation of the block, BL_HBGR_2470.
In this example,

- Part name is PT_HBGR_2470,
- Part Block is BL_HBGR_2470
- Parent Block is, BL_BA_2400

The port for this part is defined under the part block, BL_HBGR_2470.

2. Existing From Port Name

The from port name defined under the part block.

3. Existing To Part Name

The part name (not the Block but the part from which the block is created). The part should exist under a parent block.

4. Existing To Port Name

The from port name defined under the part block.

5. Rhapsody Project Name

This is required, cause working on more than one Rhapsody projects, the workspace and project directory will be distinguished based on the project.

Environment Variables Used in the Script :
Should be created upfront:
Eg:
# sudo -H gedit /etc/environment (for ubuntu)
WORKSPACE_SETitanic=/home/parallels/Downloads/workspace/NVL/SETitanic
RHAPSODY_FILE_DIR_SETitanic=Projekt_SETitanic_rpy/
PROJECTAREA_SETitanic=iOjfYfj9Eeyg2Yb4jthRGQ

6. WORKSPACE_*<rhapsody_project>*

This is an environment variable which should be set for each rhapsody project. For instance:

```
$ echo $WORKSPACE_SETitanic
/home/parallels/Downloads/workspace/NVL/SETitanic
```

The input parameter given in “2” is appended to the end of the “WORKSPACE_” to create an environment variable for this project.

7. RHAPSODY_FILE_DIR_<*rhapsody_project*>

This is an environment variable which should be set for each rhapsody project’s file directory. For instance:

```
$ echo $RHAPSODY_FILE_DIR_SETitanic
Projekt_SETitanic_rpy/
```

The input parameter given in “2” is appended to the end of the “RHAPSODY_FILE_DIR_” to create an environment variable for this project.

8. PROJECTAREA_<*rhapsody_project*>

This is an environment variable which should be set for each rhapsody project’s EWM project area id. For instance:

```
$ echo $PROJECTAREA_SETitanic
iOjfYfj9Eeyg2Yb4jthRGQ
```

The input parameter given in “3” is appended to the end of the “PROJECTAREA_” to create an environment variable for this project.

If the Project name is not provided, the script outputs a message and doesn’t continue.

If the Project name is given wrong, again the script outputs a message and doesn’t continue.

Main Actions in the Script:

1. Find rhapsody unit files where thre following model elements reside:
    a. From Part Name
    b. To Part Name
    c. From Port Name
    d. To Port Name
    e. From Part Block Name
    f. To Part Block Name
    g. From Parent Block Name
    h. To Parent Block Name

The connector definition will be created in the unit file where the “From Parent Block Name” resides.

2. Create port definition for local Rhapsody Project from the template called “connection_template.xml”.
3. Create port definition for RMM Server from the template called “connection_index_template.xml”
4. Append connection name to Parent From Block
5. Append port definition for local Rhapsody Project and RMM server under the Rhapsody unit file where the Parent From Block resides.