**Refactoring and Porting of the TEP**

|  |  | Concepts and Training |
| --- | --- | --- |
| Author | Richard Klees |
| Version | V1.0 |
| Date | 2014-04-16 |

# **Versions**

|  |  |  |  |
| --- | --- | --- | --- |
| **Changes** | **Version** | **Editor** | **Date** |
| first draft | V0.1 | Richard Klees |  |
| added mock up for filters, description of filter behaviour | V1.0 | Richard Klees | 2014-04-16 |

**Settings**

|  |  |
| --- | --- |
| Package Shortname | TEP |
| ILIAS-Version | 4.4 |
| Branch | svn.ilias.de/svn/ilias/branches/CaT/extensions/TEP |
| CaT Contact | Richard Klees <richard.klees@concepts-and-training.de> |
| Development by | Leifos GmbH |
| Developers Contact | Jörg Lützenkirchen <luetzenkirchen@leifos.com> |
| Required Documentation | classes, public and protected methods including parameters, inline documentation of intricate parts of code |

# Intent

The TEP (Trainereinsatzplanung = Trainer Operation Planing) is a component for ILIAS, developed for the Volksfürsorge Insurance. With the tep it is possible to view and manage appointments of tutors, especially their assignment to seminars.

The solution for the Volksfürsorge is based on ILIAS 4.4 and strongly coupled to the seminar managment processes of the Volksfürsorge. This solution needs to be generalized and reviewed to make it possible to use the TEP for other customers. There are also some enhancements to be done regarding the design of the code, which ease the adoption of the TEP in different scenarios.

# Overall changes

Since the TEP was implemented for the Volksfürsorge, it uses vf as prefix for classes. This prefix should be changed to the standard il prefix for all classes included in the TEP. Accordingly the prefixes for settings and language variables should be changed from vofue to tep.

The current implementation of the TEP uses a custom organisation managment implemented by Databay. This should be replaced by the new standard organisational managment of ILIAS 4.4..

Since the „create new seminar“ functionality in the TEP is strongly coupled to the seminar managment process of the Volksfürsorge, all buttons and actions that create seminars from within the TEP should be removed

# Changes in ilTEP

The permission handling done in the ilTEP-class should be factored out to a new class ilTEPPermissions. The TEP should instantiate that class and use it for permission checks instead of directly calling the RBAC or determine a permission by other means. This especially covers the currently implemented methods starting with may. Other permission checks in the TEP should also be delegated to methods of the new ilTEPPermissions class, where the methods have descriptive names similar to the current may\*-methods.The instance of ilTEPPermissions should only be created by ilTEP and queried from there if it is needed in another class of the TEP.

The logic to create, update and delete entries in the TEP should be moved from ilTEPGUI to ilTEP. In this way it is easier to create entries in the TEP programmatically, where the GUI of the TEP is only one way to create entries.

To reduce the coupling to other components, the TEP should react to events and announce events by itself, instead of being called by other classes to often. Regarding the class ilTEP, this means, that ilTEP::deleteUser should be called on the event Services/User/deleteUser. ilTEP should throw events when an entry is created, updated or deleted. (TEP/create, TEP/update, TEP/delete).The historizing of the TEP for reporting purpose could then be invoked, when a TEP event is announced. The code for historizing tep entries should therefore be removed from ilTEP.

The mechanism for operation days of tutors as it is currently implemented for tep entries related to courses should be made generic so it could be used for every type of entry. There is no need to make this mechanism accessible via the GUI for general entries. The solution for setting operation days of tutors for courses is outlined below.

In general, all course specific handling of tep entries should be removed from ilTEP. This handling should be moved to an extra singleton class ilTEPCourseEntries, that uses the methods of ilTEP to create, delete and update entries in the TEP. The column for courses without tutors should not be removed.

The standard title for personal calendars created for the use by the TEP should be queried from a language variable.

# Changes in ilCustomCalEntryType

In the current implementation of TEP, the types for TEP entries are hardcoded in the class ilCustomCalEntryType. This hardcoded types should be replaced by a database based retrieval of the available categories. There is no need to introduce a GUI to access or modify the entry types in the database.

The constants defined to identify entry types should be replaced the possibility of a string based identification of categories.

To determine whether an entry type could be created, modified or deleted through a GUI oft he TEP, a flag to determine that possibility should be added to the entry types.

# Changes in ilTEPView

The rendering of entries in the TEP should be made template based rather then being created inline in the code.

If an entry has a context\_id, that id is a valid object\_id and the user has view permission on that object, the title of a TEP entry should link to that object in the repository.

The logic to determine holidays and weekends should be factored out in a new class, that is capable of telling whether a given date is a weekend or holiday for a given country, where the country should be identified by an ISO 3166 country identifier.

This class should be implemented for germany only and raise an error for other country codes. The TEP should use the query with german language code.

# Entries for Courses: ilTEPCourseEntries

The logic to create TEP entries for courses should be moved to a new singleton class ilTEPCourseEntries. This class should use the generic methods of ilTEP to create entries in the TEP.

To create, delete and modify entries for courses in the TEP, ilTEPCourseEntries should react on the events Modules/Course/create, Modules/Course/delete, Modules/Course/update, Modules/Course/addParticipant and Modules/Course/deleteParticipant.

Since there is no notion of start- and enddate of a course in the ILIAS standard, the class should have two methods to determine that values. For testing purpose those methods should be filled with mockup code. The methods will be implemented properly later.

The notion of a halfday date (saved in the column vf\_halfday in the table cal\_entries) should be removed completely.

# Trainer Operation Days for Courses

A new GUI class to set operation days for all tutors of a course should be introduced. It should be possible to give a ref\_id of a course as get parameter to that class. The tutors to set operation days for should be determined based on the ref\_id. There should be a back button which links to the members page of the given course. The start and enddate of the course should be determined by the new class ilTEPCourseEntries.

The permission to access the GUI for a specific course should be querid at the new permission class (see below). For testing purpose those method should be filled with mockup code. It will be implemented properly later.

There also should be GUI class to set the operation days for an individual tutor of a course. The GUI-class should take the id of the course and of the user

The form to set the operating days of tutors and the form element to set the operating days for an individual tutor should be implemented in a way that they could easily be reused in larger forms or pages..

# Filters

The „Seminartitel“-filter in the TEP should be removed completely, since there should be now general distinction between TEP entries of courses and other TEP entries after the changes proposed in this paper.

The large org-unit filter used at the TEP of the Volksfürsorge should be removed in favour of a smaller org unit multi select filter and a checkbox for including sub units of the selected org units. The new org-unit filter should be a part of the filter section on the page as all other filters. In the multi select part of the filter, all org-units where the user has the permission tep\_view\_others and all org-units and subjacent org-units where the user hast he tep\_view\_others\_recursive should be offered for selection, where the names of the org-units are ordered alphabetically.

If the user did not select an org unit in ist current session, the default setting for the filter is „all tutors“. The filter

When the user clicks on the Select-Link in the org-unit filters, he should be provided with an overlay on the right side of the screen, that shows all org-units the user can filter under the root node „all trainers“. A user can filter an org-unit if he has the permission *tep\_view\_others* or *tep\_view\_others\_recursive* in that org-unit. If he has the permission *tep\_view\_others* in the org-unit, only the org-unit itself should be shown in the selection. If he has the permission *tep\_view\_others\_recursive* in the org-unit, the org-unit and all the subjacent org-units should be shown in the selection.

The user can select multiple org-units by selecting them in the multi select box and decide whether he wants to see only direct members of the selected org units or the members of the subjacent org-units as well by using the checkbox beneath the org-unit multi select box. The subjacent org-units should only be taken into account for org-units where the user has the permission *tep\_view\_recursive*.

If the user is in the month view of the TEP and clicks on Filter the TEP should display the month view of all tutors that are members in the selected org-units.

If the user is in the halfyear view of the TEP, the filters should show a tutor selection dropdown instead of the „Show courses without tutors“-filter. The user should be able to select one of the tutors in the currently selected org-units, where the tutors are ordered alphabetically by their lastname. The filter should be filled with a previously selected tutor, if that tutor is a tutor in the selected org-units. If the that is not the case, the first tutor in the dropdown should be selected. The tutor selection should also submit the filters directly as if the user clicked on Filter and the halfyear view should be displayed for the selected tutor.

# ilTEPPermissions

The granting of permission used by the TEP should be encapsulated in a new class ilTEPPermissions. This would make it easy, to change the logic to grant a specific permission since this could be done in one place.

To replace the permissions in the Databay org-managment, which are currently in use, three permissions in the ILIAS standard org-managment are required:

* **tep\_is\_tutor:** The user may view the TEP with ist own entries and create, delete and modify entries for himself if he has that permission in any org-unit. The user counts as tutor in any org-unit where he has this permission.
* **tep\_view\_other:** The user may view tep entries for every tutor in the org-unit where he has this permission.
* **tep\_view\_other\_recursive:** The user may view tep entries for every tutor in the org-unit where he has this permission and in the org-units below.
* **tep\_edit\_other:** The user may edit tep entries for for every tutor in the org-unit where he has this permission.
* **tep\_edit\_other\_recursive:** The user may edit tep entries for for every tutor in the org-unit where he has this permission and in the org-units below.

This permissions should be used in the may\*-functions, moved from ilTEP to the new ilTEPPermissions-class.

The new permissions for organisational units should be included in the role templates.

# Install Script

To install the TEP in an ILIAS-Installation, an idempotent install skript for the TEP should be created. This skript should:

* create new org-unit permissions
* create the calendar for courses
* add a column for the entry type to the table cal\_entries
* reload the control structure
* bind events to handlers in ilTEPCourseEntries