Researcher Support System (RSS)
Application Requirements and Design Specifications
(version 0.1)

Fritz Solms, Vreda Pieterse, Stacey Omeleze & Antonia Michael
Dept Computer Science, University of Pretoria

March 13, 2016

Contents

The application requirements and design specify the the primary functional requirements, i.e. the user's requirements around use cases and not secondary functional requirements arising from realizing non-functional requirements.

1 Publications

The publications module provides the functionality of maintaining information to track publications through their life cycle from being envisaged to ultimately being either published or abandoned.

1.1 Scope

The scope of the *publications* module is shown in Figure 1.

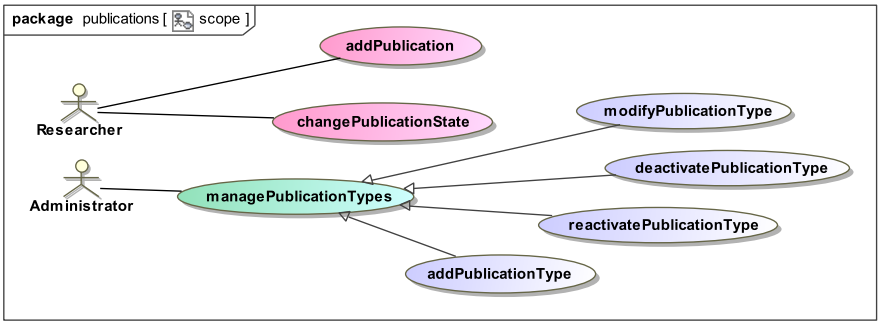

Figure 1: The scope of functionality required from the publications module.

Administrators can manage publication types and the associated accreditation points whilst researchers, research group leaders and research management can add and change the state of publications within their domain.

1.2 Domain model

The information about a publication is stored as a sequence of state-entries, each capturing the state of a publication at a particular date/time[1].

At any stage each publication has publication details (title, author, owner), a publication type, a life cycle state, and optionally a publication target. With each new state entry any aspect of the paper can change.

1.2.1 Publication types

The publication types can be configured with each publication type having a name (e.g. accreditedJournal, conferencePositionPaper, conferenceMainPaper, book, ...). Each publication type has at any stage associated with it an `accreditationsPointChange` with the first one being created

[1] When adding a state-entry one starts with a copy of the previous entry and just applies the appropriate changes.

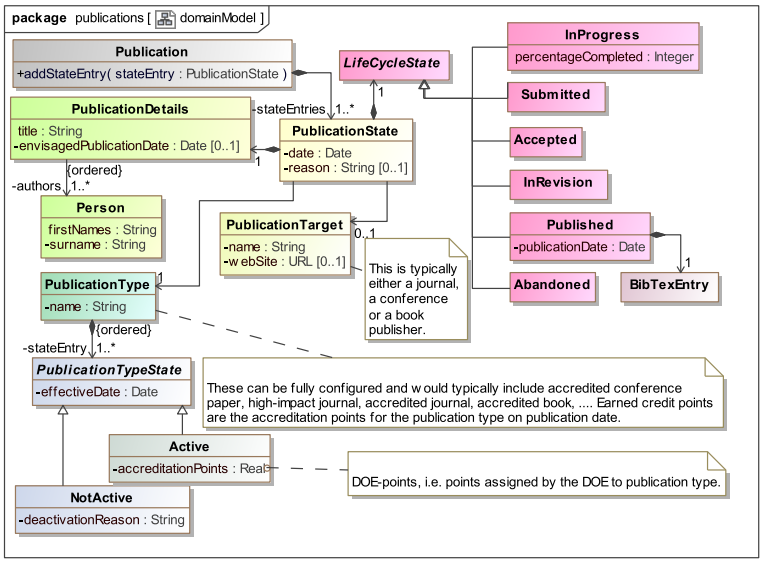

Figure 2: The domain model for publications.

when the publication type is created. The effective date specifies the date on which the assigned accreditation points become have become effective for that publication type.

If, at some stage, the rules for accreditation points change, a new `AccreditationsPointChange` is added to the sequence of accredited point changes for that publication type. Any papers who's publication is between the original effective date and the new effective date will still accumulate the original accreditation points whilst any papers published after the effective date of the last `AccreditationsPointChange` will get the accreditation points specified in that last accreditations point change.

1.2.2 The life-cycle state of a publication

A publication is in one of a set of states as shown in Figure 2. The system will allow transition from any one state to any other state. Users can capture a reason for any state entry and can thus explain why, for example, progress might be negative.

1.2.3 Publication targets

The system will store for publication targets only the name (e.g. conference, journal or book publisher name) as well as the web site of the publication target.

1.3 addPublication

Adding a new publication creates a new publication with its own state trace represented by a time-ordered sequence of state entries.

1.3.1 Service contract

Figure 3 depicts the service contract for the `addPublication` service.

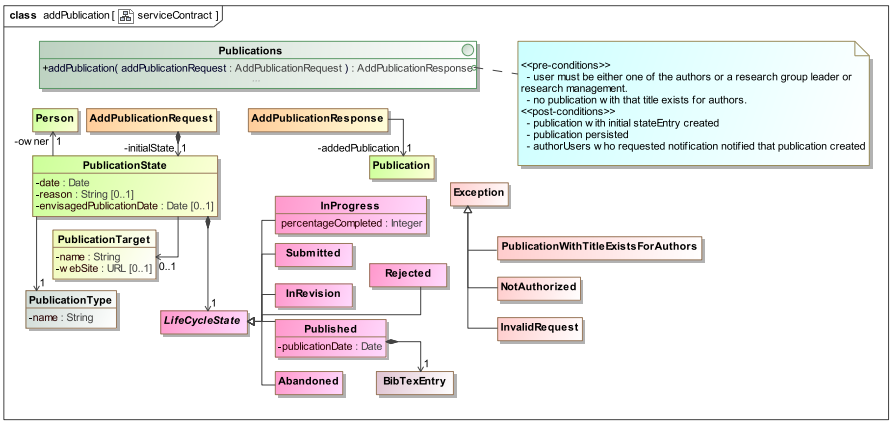

Figure 3: The service contract for the *addPublication* use case.

The use case creates a new publication with a single `PublicationState` encapsulating the initial state of the publication. Future state changes will add further `PublicationState`s to the `stateEntries` of a publication, i.e. to the time-ordered sequence of `PublicationState` representing the complete state history for the publication.

For a user to be able to add a publication one of the following must hold true:

1. He/she is one of the authors of the publication.
2. The user is a research group leader and at least one of the authors is a member of the research group.
3. The user has the role of research manager.

If any of the above does not hold a `NotAuthorized` exception is raised.

In addition, the service will be refused if either

- the request does not comply to the data structure specification or if
- the system has already a publication with that same title and the same authors.

In either case the user will be notified by a corresponding exception being raised.

1.3.2 Functional requirements

Figure 4 shows the lower level services required by the `addPublication` service to either check the pre-conditions or address the post-conditions.

Note that the `getPublication` service is required to check whether a publication with that title already exists for the specified authors. The other services

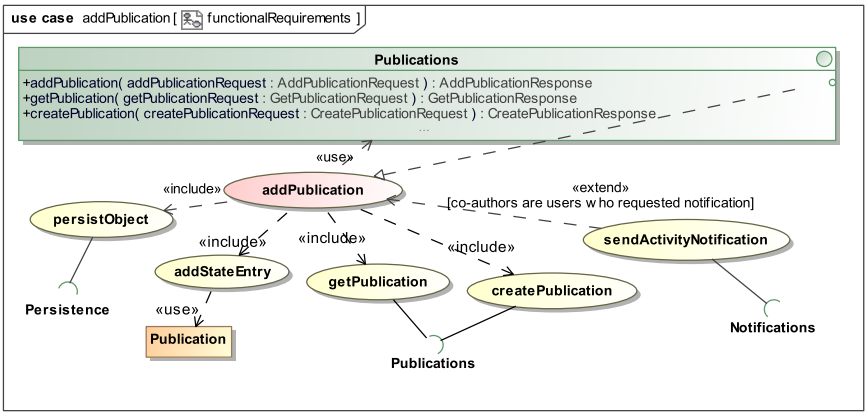

Figure 4:

- create a publication which is really just an identifier for a state trace containing the full state information of a publication over time as a sequence of mementos,
- adding the `stateEntry` representing the publication's initial state to the state trace,
- persisting the publication and
- fulfilling any notification obligations for the creation of the publication.

1.3.3 processDesign

The process for adding a new publication is straight forward. It is depicted in Figure 5.

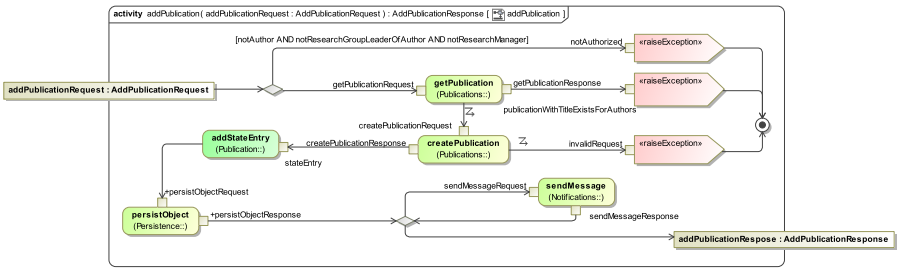

Figure 5:

Pre-condition violations result in the exception associated with that pre-condition being raised. This includes trying to retrieve a publication with that same name and authors. If such a publication is provided an `PublicationWithTitleExistsForAuthors` exception is raised.

Otherwise the service is provided such that all post-conditions hold true. This includes

- creating the actual publication (which is really just creating a publication identifier against which state entries are provided),
- adding the state entry capturing the initial state for the publication to the publication's state trace,
- persisting the publication to database, and
- if notification is required by any of the authors (user authors), to send the creation message to them.

1.4 changePublicationState

A publication has a sequence of state entries representing the state trace for that publication. Effectively it is an implementation of the *memento* pattern with each state entry being a memento capturing a snapshot of the state of the publication.

1.4.1 Services contract

Figure 6 depicts the service contract for the `changePublicationState` service.

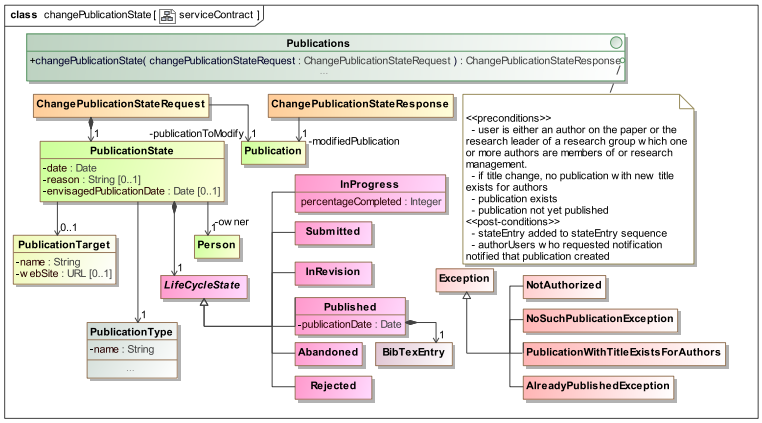

Figure 6: Service contract for the *changePublicationState* use case.

Changing the state of a publication results in adding a new state entry to the sequence of state entries of a publication. The sequence of state entry provides a full state trace for the publication.

For a user to be able to change the state of a publication one of the following must hold true:

1. He/she is one of the authors of the publication.
2. The user is a research group leader and at least one of the authors is a member of the research group.
3. The user has the role of research manager.

If any of the above does not hold a `NotAuthorized` exception is raised.

In addition, the service will be refused if either

- the request does not comply to the data structure specification or if
- the system has already a publication with that same title and the same authors.

In either case the user will be notified by a corresponding exception being raised.

1.4.2 Functional requirements

Figure 7 shows the lower level services required by the `changePublicationState` service to either check the pre-conditions or address the post-conditions.

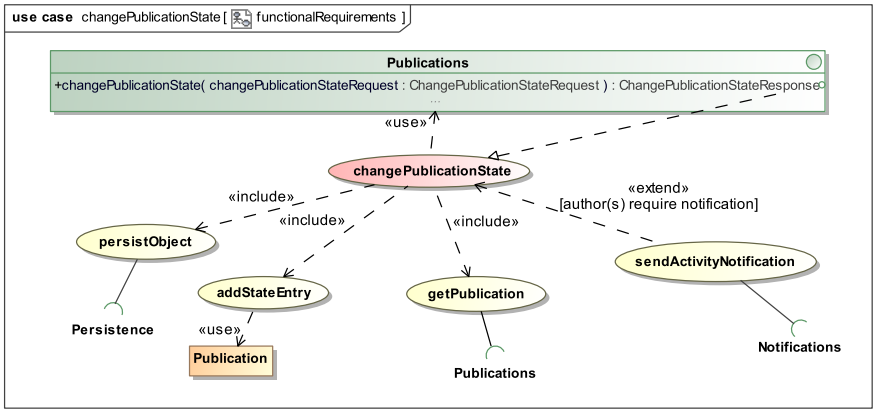

Figure 7:

Note that the `getPublication` service is required to both,

- retrieve the publication which is to be modified, and
- to check whether a publication with that title already exists for the specified authors.

The other services

- adding the `stateEntry` representing the publication's initial state to the state trace,
- persisting the publication, and
- fulfilling any notification obligations for the creation of the publication.

1.4.3 Process design

The process for changing the state of a publication is is depicted in Figure 8.

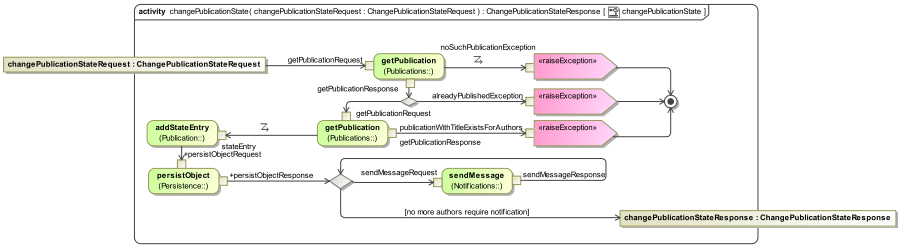

Figure 8:

1.5 Publication types

The main reason for assigning publication types to publications are

- that different publication types earn different accreditation points which result in different amount of funding for research outputs and
- for more meaningful reporting.

The publication types and the accreditation points they earn are determined by the funding organization (e.g. the *Department of Education* (DoE)). The mini-project will assume that there is only a single funding organization.

The application must maintain the full history of the publication types, i.e.

- which publication types were available and active at which time, as well as
- the accreditation points a publication type earned on any particular date from a funding organization.

1.5.1 addPublicationType

Administrators will be able to add new publication types, each with a unique name and an initial state which is typically an active state to which zero or more accreditation points are assigned, i.e. a publication published after the effective date will earn the specified accreditation points with the DOE.

1.5.1.1 Services contract Figure 9 depicts the service contract for the `addPublicationType` service.

Each publication type is created with an initial `PublicationTypeState` which has an effective date, and, if initially active, the number of accreditation points a publication of that publication type will earn on publication date. Any publication of that publication type published after the effective date of the initial `PublicationTypeState` and before the state is overridden by a new `PublicationTypeState` at its effective date will earn the credit points as specified by the initial `PublicationTypeState` created for the `PublicationType` when it was added.

Later, when the `modifyPublicationType` service is called, new `PublicationTypeState`s are added to the `PublicationType`, each with their own effective date. The effective period of a state

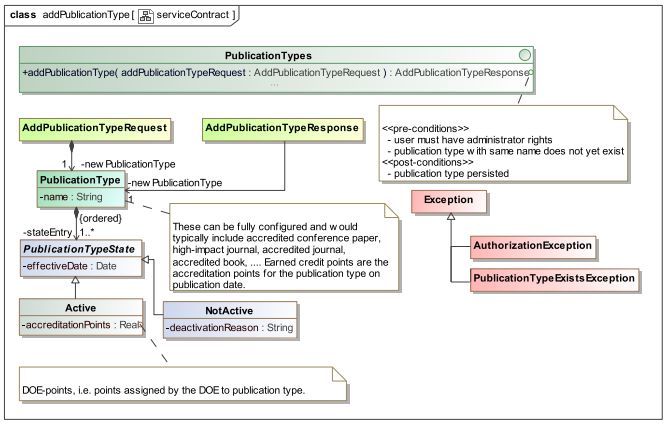

Figure 9: The service contract for the *addPublicationType* use case.

entry, i.e. a `PublicationTypeState`, is from its effective date to the effective date of the next `PublicationTypeState`s in the time-ordered sequence of state entries. The last entry is effective from its effective date to the current date.

1.5.1.2 Functional requirements Figure 10 shows the lower level services required by the `addPublicationType` service to either check the pre-conditions or address the post-conditions.

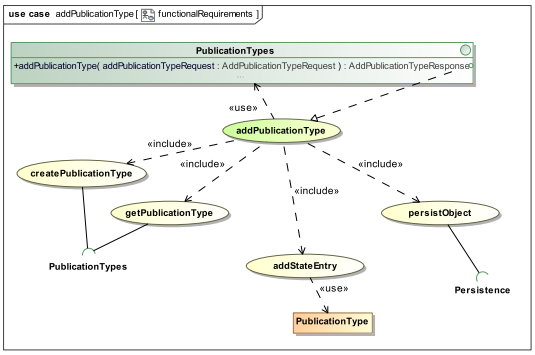

Figure 10:

1.5.1.3 Process design The process for adding a new publication type is straight forward. It is depicted in Figure 11.

Note that the `getPublicationType` service is required to check whether a publication type with that name already exists. If that service returns a publication type with that name, then the

publication type exists already and a `PublicationTypeExistsException` is raised.

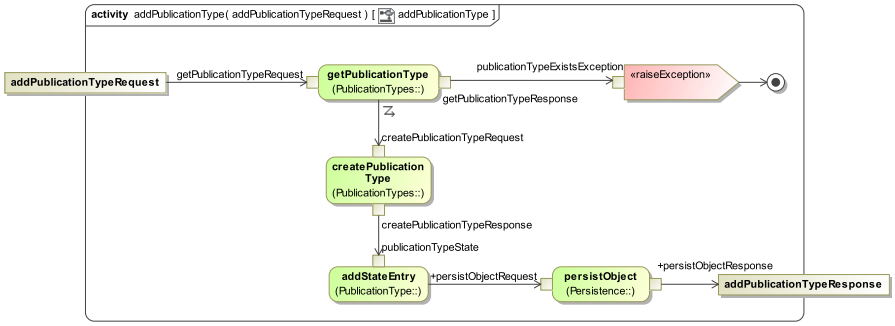

Figure 11:

On the other hand, if the `getPublicationType` service raises a `NoSuchPublicationTypeException`, then the publication type does not yet exist and it is created and persisted with its initial state.

1.5.2 Modify publication type

The state of a publication type can change at any stage, for example, by earning more or less accreditation points from a new effective date onwards (whilst still remaining active) or by being deactivated on some effective date. The state sequence captures the full state history of publication types and makes it possible to calculate the number of credits a publication of a particular publication type will have earned on some date.

1.5.2.1 Services contract Figure 12 depicts the service contract for the `modifyPublicationType` service.

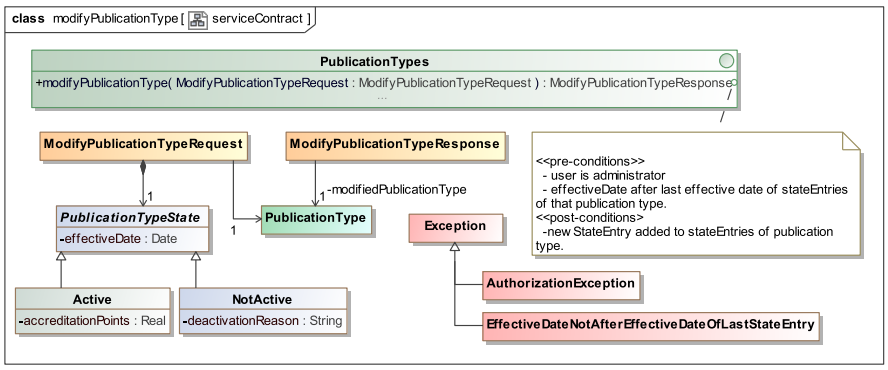

Figure 12: The service contract for the *modifyPublicationType* use case.

Modifying a publication type results in a new `PublicationTypeState` (i.e. a new stateEntry) being added to the stateTrace of the publication type. The previous state is valid from its effective date to the effective date of the new state.

1.6 getPublicationsForPerson

This service returns all publications for an author which either have been published, accepted or are envisaged to be published for a user within a specified time period. If no time period is specified, then all matching publications over all time are returned.

1.6.1 Services contract

Figure 13 depicts the service contract for the `getPublicationsForPerson` service.

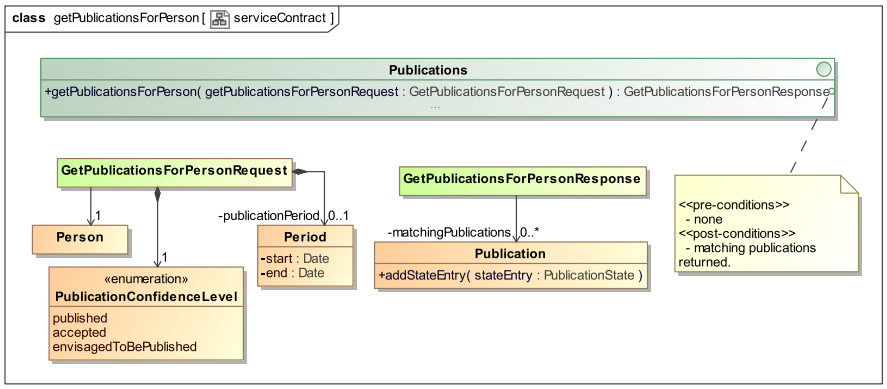

Figure 13: Service contract for the *getPublicationsForPerson* use case.

1.7 getPublicationsForGroup

This service is similar to the *getPublicationsForPerson* service except that it returns all published, accepted or envisaged publications for a period for a group. The group may be a first level group (e.g. a specific research group) or a higher level group (e.g. the department).

1.8 calcAccreditationPointsForPerson

This service finds all publications published, accepted or envisaged to be published for a period by a person and sums up the accreditation points earned. The contribution from each individual paper is the accreditation points for the respective publication type divided by the number of authors on that paper.

1.9 calcAccreditationPointsForGroup

This service is similar to the *calcAccreditationPointsForPerson* service except that it accumulates the accreditation points for all persons who are part of that group. The group may be a first level group (e.g. a specific research group) or a higher level group (e.g. the department). All members of a sub-group are also members of any higher level grouping.

2 Persons

The *persons* module is responsible for maintaining demographic information about the persons (researchers) themselves, some of which may be users of the system and others not as well as groupings of persons into research groups.

2.1 Scope

The scope of the *persons* module is shown in figure 14

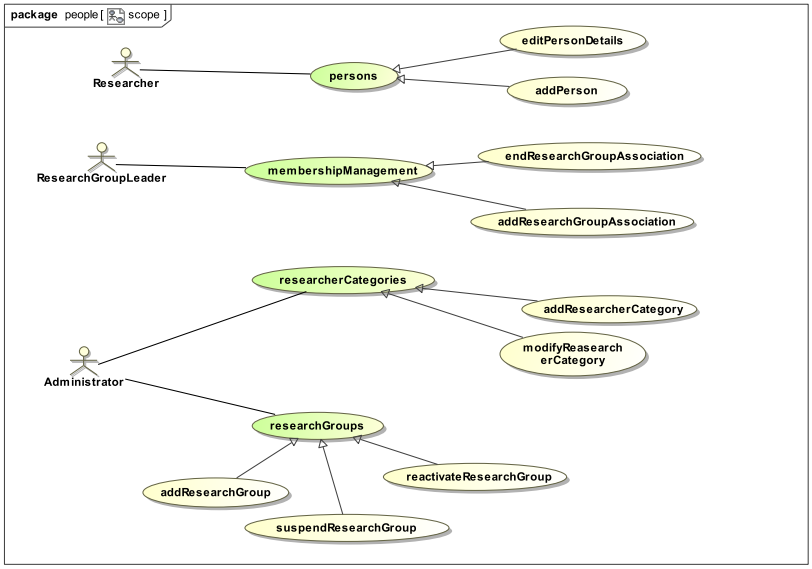

Figure 14: The scope of functionality required from the persons module.

The scope of the persons module includes

- adding persons and modifying their details including the researcher category a person is assigned to,
- adding groupings of persons into research groups as well as grouping groups into higher-level groups,

- defining and changing group memberships of persons, and
- creating and modifying researcher categories with associated research output targets.

2.2 Domain model

The domain model for the `persons` module is shown in Figure 15.

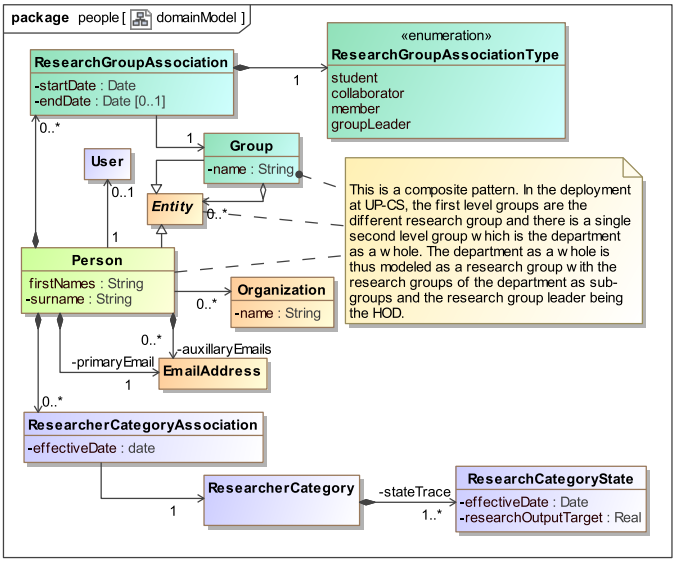

Figure 15:

Each person has first- and surnames, an optional email address and zero or more associations with organizations. In addition each person can have multiple associations with research groups. Each research group association is a role which a person plays within a research group and has a start-date as well as optionally an end date. Hence, over time a person may change from becoming a student member to a staff member and may change his or her affiliations with research groups. The full affiliation history is maintained by the system, i.e. one will be able to determine for any point in time, which research group associations a person had.

For performance tracking, researchers may be associated to a `ResearcherCategory` which has at any point in time a `researchOutputTarget`. The target may vary over time and the system maintains a state trace for any researcher category through which one can determine the state and hence the target outputs at any point in time. When the target output for a researcher category changes a new `ResearcherCategoryState` frame is added with the new effective date and the previous state frame lapses on the effective date of the new state frame.

3 Notifications

The `notifications` module is responsible for sending messages to users. this includes

- once-off or reminder messages

3.1 Scope

The scope of the *notifications* module is shown in Figure 16.

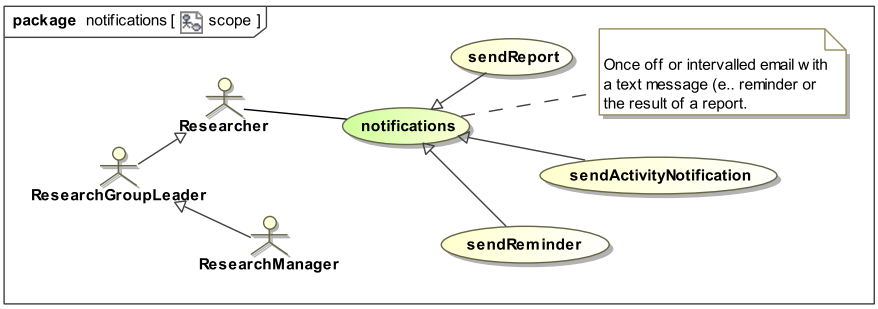

Figure 16: The scope of functionality required from the notifications module.

Researchers and group leaders (e.g. research group leaders or leaders of higher level groupings like the department as a whole) can request notification messages like

- *reminder notification* requests (e.g. that a deadline for a conference submission is approaching) a *reminder* that a paper needs to be submitted to a conference,
- *activity notifications* (e.g. that a publication for which they are an author has been added or modified), and
- report notifications (e.g. have certain reports sent to them in regular intervals.

Group leaders and administrators may, in addition, schedule broadcast notifications (e.g. that all research paper information should be updated by a particular date).

3.2 Domain model

The domain model for notifications is shown in figure 17 depicts the domain model for the *notifications* module.

The domain model maintains information around

reminder notifications used by users to schedule general reminders for themselves,

activity notifications used by users to be notified of any changes made to publications on which they are authors on,

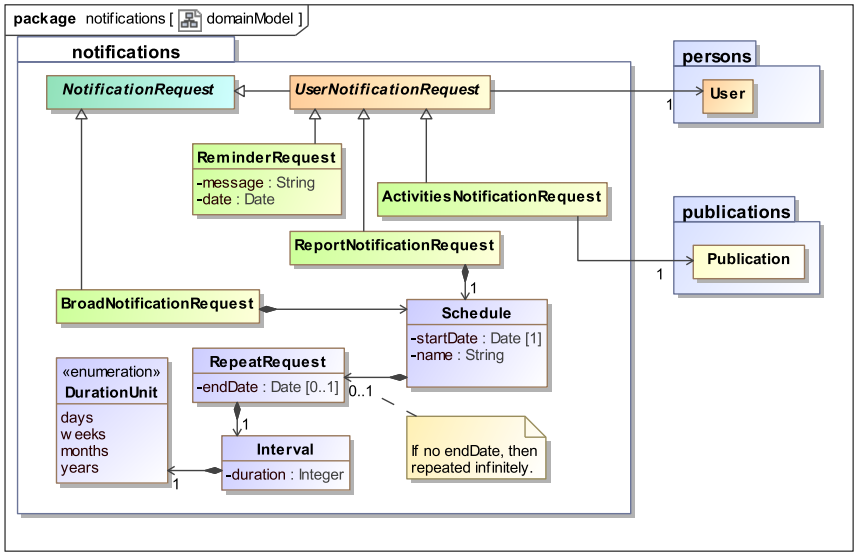

Figure 17:

report notifications used by users (particularly by group leaders like research group leaders and the head of department) to have automatically a publications report on a group (e.g. a research group or a higher level group like the department as a whole) sent, and

broadcast notifications used by group leaders or administration to send scheduled broadcast messages (e.g. that the publications report will be drawn and that suers should add any publications they want to have included in that report).

4 Reporting

Reporting will generate visualization of the following:

1. Information regarding all accreditation units active within a specified time period. The information should be based only on the current state of the relevant publications.

   - per entity (i.e. person or group)
   - per publication type
   - per selection of lifecycle states
   - per combination of any of the above

   When data is aggregated for a group, it is done by calculating the sum of the relevant information as gathered per member of the group.

Each visualisation should include information related to the accreditation unit target for the specific entity. It should be possible to show or hide information related to accreditation unit targets.

Papers which have either no envisaged publication date or whose envisioned or actual publication date is outside the reporting period are not included.

2. Information regarding the progress status of publications. The information should be based on the indicated % complete value for all publications that are currently in the "in progress" life cycle state.
    - per entity (i.e. person or group)
    - per publication type
    - per combination of any of the above

In future work, it should be possible to generate these reports based on a specified instance in time in the past.

5 Import/export

The system will support

- importing and exporting of persons and research groups from a CSV file,
- importing and exporting of publications for a person from a CSV file,
- exporting published papers for a user or a group to a bibtex file.

The required data structures for the import/export from/to CSV files needs to be derived from the domain models for the persons and publications modules. However, importing and exporting will only include the latest state of the entities (e.g. the latest publication state as well as the latest research group associations) and will not include the importing or exporting of historic data.

In cases where the data format for an import is invalid, the system will have to provide useful error messages. The import operation is atomic in the sense that if the import fails, nothing will have been imported.

In either case the target operating system can be specified in order to ensure that the correct end-of-line character is used.