PTCRISync Specification

An ORCID-based Synchronization Framework for PTCRIS

Version 0.4.3
September 2016

About PTCRIS

The PTCRIS (*Portuguese Current Research Information System*) program from the FCT|FCCN (*Fundação para a Computação Científica Nacional* of the Portuguese Foundation for Science and Technology) aims to ensure the creation and sustained development of a national integrated information ecosystem, to support research management according to the best international standards and practices. This report specifies a synchronization framework (PTCRISync) that relies on ORCID as a central hub for information exchange between the various national systems and international systems. PTCRISync will enable researchers to register a given research output at one of the interconnected national systems, and automatically propagate that output to the remaining ones, thus ensuring global consistency of the stored information. For more information about PTCRIS, see `http://ptcris.pt`.

Who should read this document

This document is aimed at PTCRIS service managers and developers who wish to use ORCID as a central hub for synchronization. The document may also prove useful to the users of the services who wish to understand in detail how their records are being synchronized among the services.

Conformity to ORCID

The current version of this specification (0.4.3) is based on version 2.0, rc1, of the ORCID API.

What changed in this version

Version 0.4.3 introduces a new scenario (19) for the two-staged export update and another (20) for empty profiles.

Authors

This PTCRISync specification was commissioned by the FCT|FCCN to the High-assurance Software Laboratory (HASLab), from INESC TEC and Universidade do Minho, Portugal, and developed by:

- Alcino Cunha
- Nuno Macedo

Acknowledgments

This specification benefited a lot from the input and suggestions of the following people, to whom the authors would like to express their appreciation:

- João Mendes Moreira (FCT|FCCN, Portugal)
- Carlos Sousa Pinto (DSI, Universidade do Minho, Portugal)
- Luís Valério, Pedro Lopes (DeGóis, Portugal)
- José Carvalho (SDUM, Universidade do Minho, Portugal)
- António Lopes (ISCTE, Portugal)
- Andrea Bollini (CINECA, Italy)

License

1 Introduction

PTCRIS (*Portuguese Current Research Information System*) is a program, officially initiated in May 2014, which aims to ensure the creation and sustained development of a national integrated information ecosystem, to support research management according to the best international standards and practices. One of PTCRIS' goals is to reduce the burden of research output management, by adopting an "input once, re-use often" principle. This report describes a synchronization framework, PTCRISync, that relies on ORCID[1] – a community-based service that aims to provide a registry of unique researcher identifiers (an ORCID iD) and a method of linking research outputs to these identifiers, based on data collected from external sources – as a central hub for information exchange between the various national systems (including CV management systems such as DeGóis, open access repositories such as RCAAP or SARI, and local CRIS systems) and international systems (WoK, Scopus, Datacite, etc), that shall enable researchers (or managers) to register once a given research output at one of the interconnected national systems, and automatically propagate that output to the remaining ones, thus ensuring global consistency of the stored information. For an overview of the expected PTCRIS architecture and the interaction between the different services, the reader is redirected to [2].

The main challenge in the design of this framework stems from fundamental differences between the data model of ORCID and that of most PTCRIS services. Thus, the report starts by describing (an abstract view of) the data model used by ORCID API v2.0 (Section 2) and the requirements that a PTCRIS service must satisfy in order to comply with this specification (Section 3). The specified synchronization framework operates at the user profile level, that is, it synchronizes user profiles from different PTCRIS services with the corresponding user profile from ORCID. The matching of users across these systems is outside the scope of this report, but it is usually a trivial issue, since PTCRIS services can (and most already do) store the ORCID iD of the researcher in his profile. The effective synchronization procedures are presented in Section 4. These are designed to keep the user profiles at ORCID and PTCRIS synchronized according to well-defined consistency constraints, and satisfy several "well-behavedness" properties, such as correctness or stability[2]. The report ends with the presentation of interesting synchronization scenarios (Section 5), that can be used both to clarify the synchronization procedures with all stakeholders, and to be used as test cases to verify the conformity of the implementation.

2 An overview of ORCID

A distinctive feature of ORCID is the possibility of using different external sources to automatically populate a user profile. As a consequence, a user profile typically contains different works that actually describe the same research output (possibly containing different or even contradictory meta-data). To ease the management of such works, the ORCID service groups together works that describe the same output, showing only the preferred one in the web interface overview. The grouping mechanism is quite simple, and just assumes two works w_1 and w_2 are *similar* if, and only if, they share an external identifier (UID) or there is another work w_3 that is similar to w_1 and similar to w_2. Essentially, this recursive definition considers two works to be similar if, and only if, they share directly or indirectly (via transitivity) some UID. As of version 2.0 of ORCID, these groups are now intrinsic to the ORCID data schema[3]. The UML class diagram from Fig. 1 depicts a portion of this data model, focusing on the components relevant to this report. Besides works, an ORCID user profile contains information

[1] http://orcid.org/.

[2] Both the data models and the synchronization procedures were formalized under the formal specification language Alloy [1] that allowed for the automatic verification of the properties. This also allowed the automatic generation of the presented scenarios.

[3] The current release candidate 1 for the 2.0 schema is available in https://github.com/ORCID/ORCID-Source/tree/master/orcid-model/src/main/resources/record_2.0_rc1.

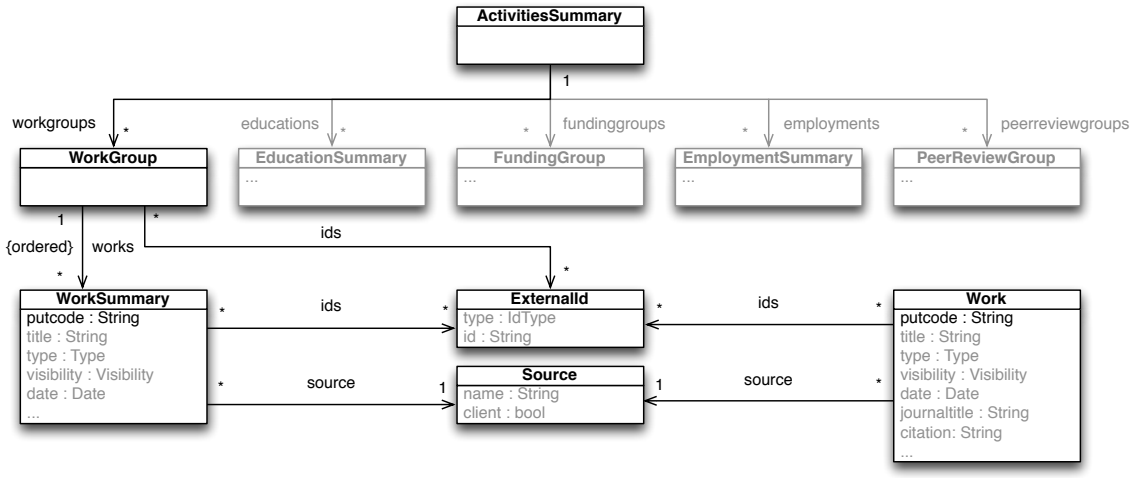


Figure 1: Overview of the ORCID data model.

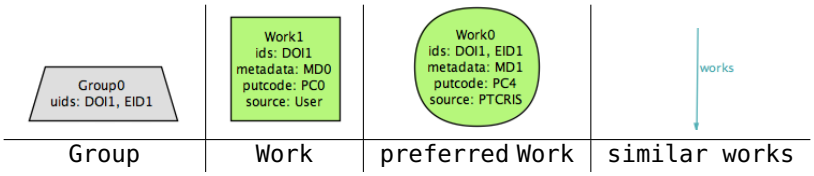


Table 1: Graphical representation of ORCID profiles.

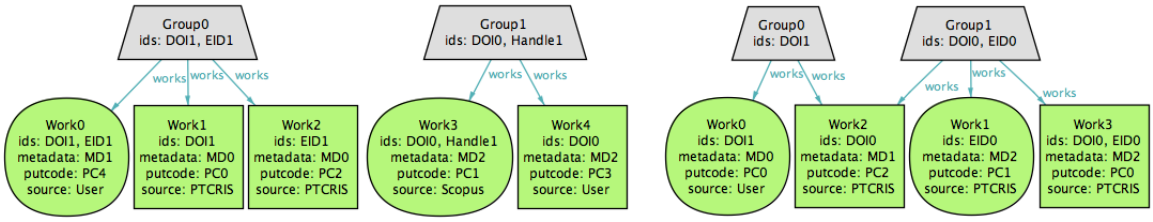
regarding other research activities, which PTCRIS services may also be interested in synchronizing. This information is either trivial to synchronize (e.g., education entries) or may be synchronized with the technique presented in this report for works (e.g., funding information). Components disregarded in this report are grayed out in the class diagram.

Taking these design decisions into consideration, this report assumes that an ORCID user profile consists of a set of *work groups*. Each group is comprised by a set of work *summaries* and a set of UIDs: this set aggregates the external identifiers from all the works in the group. The collection of work summaries within a group is ordered: the first work of the collection is considered to be the one *preferred* by the user. To retrieve the complete meta-data of a work, the user must first find its internal identifier and then explicitly request it (more on the ORCID API below). However, the synchronization framework is, in general, oblivious to the meta-data and focuses mainly on harvesting UIDs from ORCID profiles. Thus, it is in general not required to be aware of the complete meta-data of the work. Concretely, the algorithm shall require the work's *putcode*, its internal identifier which uniquely identifies it in an ORCID profile; its *external identifiers*, a (possibly empty) set of external UIDs, such as a DOI, EID or ISBN; and its *source* of information, which can be the user himself or any other external source associated with ORCID, such as Scopus, CrossRef, or one of the PTCRIS services. The ORCID service forbids works from the same source to share UIDs: if an external source tries to add a work with a UID that is shared with a previous work an error is returned[4]. The ORCID team also expects most of the works in its database have at least one UID associated, so the API forces every work that is introduced to have some UID assigned, even if it is an identifier that only makes sense for the external source[5].

Figure 2 depicts possible ORCID profiles, following the notation from Table 1 (the additional meta-

[4]Currently it is still possible for the user to introduce works with shared UIDs through the web interface.

[5]Although it is still possible for the user to introduce works without UIDs through the web interface, this has been flagged as a bug by the ORCID team, and will be eventually fixed.



(a) Valid profile. (b) Invalid profile.

Figure 2: Example ORCID profiles.

data fields are abstracted by the `metadata` field). The profile at Figure 2a is a valid one, where all the defined constraints hold, with two groups of similar works. Note how similar works are defined by the shared UIDs and not by the associated meta-data. Also, even though two works from the same source must have disjoint UIDs (`Work1` and `Work2`), they may be grouped together by a third similar work (`Work0`). Figure 2b shows instead an invalid profile where several constraints are broken: putcode `PC0` is not unique, `Work1`, `Work2` and `Work3` are from the same source (a `PTCRIS` service) but have overlapping UIDs and the groups are not correctly aggregating similar works. The synchronization framework assumes that such invalid profiles never occur and that these constraints are enforced by the ORCID service.

The synchronization framework propagates information to the user's ORCID profile through its RESTful API. ORCID provides two distinguished APIs: one that is public and another that is reserved for members. This report abstracts some of the technicalities of the API access, including the connection to the ORCID server, the OAuth authentication protocol for accessing the member API and the manipulation of the returned data in XML format[6]. The public API allows any user or service to read the public profile of a user, given his ORCID iD, an operation that will be denoted by $\texttt{getActivities}(oid : \texttt{String}) : \texttt{ActivitiesSummary}$. As already stated, the user profile contains only summaries of the research activities; once a putcode of a work is known, it can be used to retrieve the complete work record, which will be denoted by $\texttt{getWork}(oid : \texttt{String}, putcode : \texttt{String}) : \texttt{Work}$. The member API allows member services to add and remove information from the user's ORCID profile. It also allows reading information from the user's profile that the user has set as semi-private, unlike the public API that reads only public items. As of version 2.0 of the ORCID API, the member API allows services to add, update and delete works, represented respectively, by $\texttt{addWork}(oid : \texttt{String}, work : \texttt{Work})$, $\texttt{updateWork}(oid : \texttt{String}, putcode : \texttt{String}, work : \texttt{Work})$ and $\texttt{deleteWork}(oid : \texttt{String}, putcode : \texttt{String})$. Works added by a member service will automatically have their source set to that member; a fundamental constraint is that a member service may only update and delete works whose source is itself. The service is also unable to directly modify the set of preferred works selected by the user, although that may happen indirectly if a preferred work is deleted from the ORCID profile, or if a new work unifies two groups of similar works, which may only have one preferred work. Due to this fact, the impact of an update is not the same as a delete/create sequence of operations. In the latter case, it is not always clear how the new preferred work of the group is selected by the ORCID service (i.e., how the grouped works are ordered).

3 Requirements on the PTCRIS service

Unlike ORCID, we assume the data model of typical PTCRIS services (depicted in Fig. 3) does not recognize multiple versions of the same research output and thus does not group similar outputs together. Typically, the profile of a user in a PTCRIS service consists of sets of different research

[6] For a tutorial on the use of ORCID's RESTful API the reader is redirected to the ORCID support center (`http://members.orcid.org/api/`).

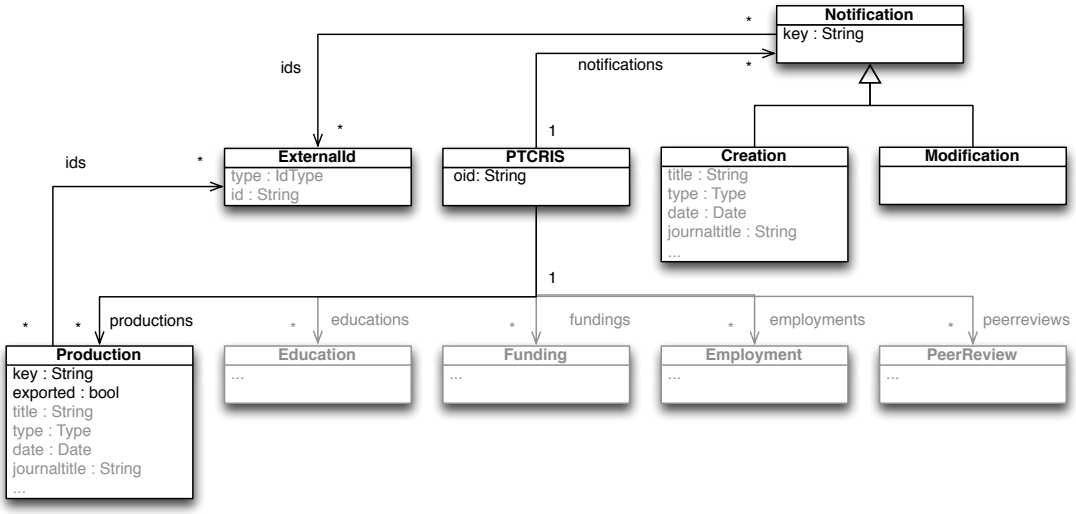


Figure 3: Overview of the PTCRIS data model.

activities, although this report is concerned only with research *productions*. It is also assumed that this profile contains the ORCID iD of the user. The data contained in each production also varies with the PTCRIS service. For the synchronization framework, a production is assumed to contain at least a *key*, an internal identifier that uniquely identifies it within a PTCRIS user profile; a (possibly empty) set of UIDs (the supported types are assumed to be the same as ORCID); and a boolean field indicating whether it is selected to be *exported* to ORCID. A production may only be selected to be exported if it contains at least one UID (in order to follow the ORCID guidelines to avoid works without UIDs), and if two productions share UIDs, only one of them may be selected (due to the restriction on unique UIDs from the same source). Maintaining these constraints may not be a trivial affair: for example, when adding a UID to a production two productions that previously shared no UIDs may now share some. There are several valid alternatives for enforcing this constraint: for example, ask the user which of the productions should no longer be exported, or simply deselect all conflicting productions from being exported. A production will typically also contain varied additional meta-data such as its title, publication year, publication type, authors, etc.

As will be presented in the next section, the synchronization framework is semi-automatic and notification based. Thus, each service will be required to support two kinds of notifications in a user profile: *creation* notifications, to alert the user that a new production has been found in ORCID; and *modification* notifications, to alert the user that new UIDs for an existing production have been found. The latter are particularly useful for propagating UIDs between different PTCRIS services, in particular from open access repositories, that provide *handles* for research outputs, to academic CV management services, such as DeGóis. The notification mechanism relies on the keys of modification notifications to point to the production that is to be modified, while the keys of creation notifications are unique and must not overlap with those of the existing productions, as they result in new productions if accepted. Notifications have also a set of UIDs, which must not be empty for modification notifications, since their goal is precisely to propagate newly found UIDs. Additionally, a creation notification contains the meta-data associated with the new production. The shape of the meta-data supported by the PTCRIS service may not be the same as the one supported by ORCID and it may not be trivial to convert one format to the other. However, as will be shown below, the synchronization procedure is oblivious to the meta-data information and relies only on UIDs to match research outputs. Thus, no specific conversion procedure from one format to the other will be imposed, leaving to each PTCRIS service the decision of how to do so, which should be clarified as precisely as possible. In this report, this is abstracted by assuming that the constructor of creation notifications takes into consideration an

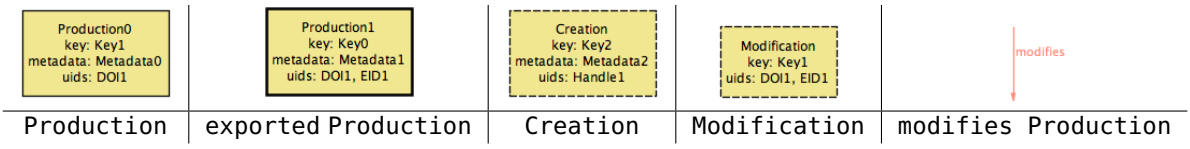
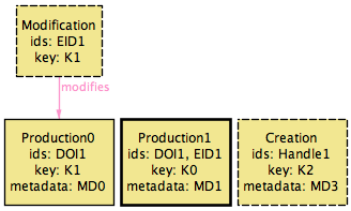
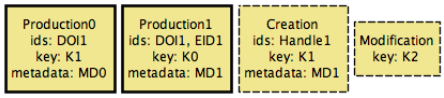


Table 2: Graphical representation of PTCRIS profiles.



(a) Valid profile.



(b) Invalid profile.

Figure 4: Example PTCRIS profiles.

ORCID work from which the meta-data is extracted. If the notification system is to be used for other purposes in the PTCRIS service, this data model must be adapted accordingly.

Figure 4 depicts possible PTCRIS profiles automatically generated after its formalization, following the notation from Table 2. Figure 4a depicts a valid PTCRIS profile with two productions, one of which (`Production1`) is selected to be exported. There is also a creation notification and a modification notification, set to modify `Production0`. In contrast, Fig. 4b depicts an invalid instance, since the two productions with shared unique identifiers are set to be exported, the key of the creation notification is not unique, the modification does not refer to an existing production and does not have any UIDs assigned. These constraints should be maintained by the PTCRIS service internally, and the synchronization procedures assumes that the provided user's profiles are indeed valid.

In this report no specific implementation for PTCRIS services is imposed, as different services may wish to pursue different approaches. Thus, the synchronization procedures presented below take as input a user profile, and are not concerned with how it was obtained nor how it can be incorporated back to into the PTCRIS service database. Nonetheless, the PTCRIS services are assumed to at least implement the following methods over a profile: `resetNotifications`(), that deletes all existing notifications, `addCreation`(ids : `Set(ExternalId)`, $work$: `Work`), that adds a creation notification with given set of external identifiers and meta-data extracted from a given ORCID work, and `addModification`(key : `String`, ids : `Set(ExternalId)`) that adds a modification notification for a given production (identified by its key) with a given set of external identifiers. Since the role of the synchronization framework is solely to manage the notifications in the user's profile, the synchronizer is not allowed to modify the productions nor the selection of productions to be exported. The service is also assumed to implement a constructor `newWork`(ids : `Set(ExternalId)`, $production$: `Production`) : `Work` to create a new ORCID work with a given set of external identifiers and with meta-data extracted from a given PTCRIS production.

4 Synchronization procedures

This section specifies the synchronization procedures that allows a PTCRIS service to keep a user's profile consistent with the one at ORCID. There are two modular procedures that can be used to synchronize the profiles according to two different modes:

Import This mode aims to harvest new research outputs from ORCID, namely new publications and new UIDs of known publications. The general principle is that every UID in an ORCID profile

should be harvested. The synchronization procedure supporting this mode is semi-automatic, based on a notification system, allowing the user to select which outputs or UIDs he wishes to add to his PTCRIS profile.

Export This mode is targeted for PTCRIS services that intend to be ORCID sources and export their productions to ORCID, ensuring that other PTCRIS services can harvest them. The general principle is that every production selected to be exported in the PTCRIS profile should be inserted as a new work in the ORCID profile and then automatically kept up-to-date.

These modes are supported by separate synchronization procedures, named $\texttt{IMPORT}(p : \texttt{PTCRIS})$ and $\texttt{EXPORT}(p : \texttt{PTCRIS})$. The IMPORT procedure does not change the ORCID user profile, managing only the notifications of the input PTCRIS profile p. This semi-automatic approach provides the user with valuable information while still allowing him to control the updates that are effectively applied to the profile. The option for a notification based semi-automatic approach is due to the fact that the ORCID user profile may contain erroneous information (for example, erroneous meta-data), and, as such, we avoid propagating such error to the PTCRIS profile, giving the opportunity for the user to clean-up his ORCID profile beforehand (for example, deleting incorrect works or creating new versions with corrected meta-data). The EXPORT procedure does not change the input PTCRIS user profile p and manages only works on the ORCID profile whose source is the PTCRIS service. The ORCID profile is updated through its API, given the ORCID iD stored in p. A PTCRIS service may choose to implement only one of the modes (e.g., RCAAP is only concerned with exporting outputs, while the SARIs are concerned with harvesting outputs) or the conjunction of both (e.g., the CV management system DeGóis). In the latter case, EXPORT must be executed prior to IMPORT, since running the EXPORT procedure may change the grouping of works (see Scenario 12). This procedure is denoted by:

$$\texttt{SYNC}(p : \texttt{PTCRIS}) \doteq \texttt{EXPORT}(p); \texttt{IMPORT}(p)$$

The consistency ensured by both modes is precisely stated in the companion formal specification (with a precise set of constraints that instantiates the above general principles), and the synchronization procedures were designed to satisfy several "well-behavedness" properties concerning such consistency[7]. The most important of those is *correctness*, namely ensuring that after running the synchronization procedures the user profiles in ORCID and in the PTCRIS service are consistent according to the specification. Another important "well-behavedness" law is *stability*, ensuring that if the synchronization procedures are tun on already consistent profiles the result is the same (modulo differences in the internal identifiers). Having stable synchronization procedures ensures that there is no need to explicitly check the consistency to determine whether they should be run, since running the synchronizers will not affect them. In fact, the checking procedures have the same approximate complexity as the synchronizers, and thus, no significant performance gains would be achieved by running them beforehand. It could even cause a performance degradation if the user profiles happen to be inconsistent.

Each service is free to choose when to run these synchronization procedures, as long as inconsistencies in the profiles are eventually resolved within a reasonable delay. One possible choice would be to run them periodically in the specified order in batch mode, thus avoiding possible delays that can negatively affect the user interface. Premium ORCID members could also trigger the synchronization based on Webhooks Change Notifications from ORCID, by registering to be notified when a user profile changes[8]. Another sensible choice would be to run IMPORT at the begin of a user session and EXPORT at the end. This ensures that the visible parts of the profiles are consistent when the user is logged out, but that whenever he logs in again the correct notifications are shown. We believe that invoking the synchronization procedures every time the user performs an edit within a session may

[7] The interested reader can consult [2] and the Alloy specification for an understanding on how the framework was formally specified and checked for "well-behavedness".

[8] See tutorial at http://members.orcid.org/api/tutorial-webhooks.

be counterproductive, as new notifications might keep popping-in and confuse the user. Similarly to distributed systems, the goal of the synchronization framework is to ensure eventual consistency and not necessarily real-time strong consistency among all services.

4.1 Import

As discussed above, the general principles of this mode are that it should harvest all UIDs present in the ORCID profile and present them to the user in the form of notifications. Whenever a new research output is found (one without previously known UIDs) the user should be notified by a creation notification, and whenever new UIDs for an existing research output are found, the user should be notified by a modification notification that lists the newly found UIDs of the output. To enforce these principles, creation notifications should not share UIDs with exiting productions and have at least a UID assigned: since our research output matching procedure is based just on UIDs instead of meta-data, there is no way to determine precisely whether a work without UIDs is already in the PTCRIS profile. If the creation of such works was notified and then later accepted by the user, undesirable notifications for the same works would reappear. The choice to implement a matching procedure based just on UIDs is mainly to preserve uniformity in the design, more specifically, to use the same matching procedure that ORCID currently uses: we believe that having a different matching procedure in the synchronization framework would be confusing for the end user, and should be avoided. Modification notifications should only be applied to an existing production that shares UIDs with the harvested output but always introduce new UIDs, otherwise it is a spurious notification. Moreover, a modification notification should be created for every production in that situation. Finally, it is expected that there are no spurious notifications other than those arising from newly found UIDs.

Since the PTCRIS services do not support grouping likewise to ORCID, some care must be made to avoid the proliferation of productions and notifications that actually describe the same research output. In particular, when an ORCID group of works is unknown to the PTCRIS service, the existence of a single creation notification in the user profile, grouping all UIDs of similar works should suffice to ensure consistency. Of course, this raises the issue of how the meta-data of such creation notification should be filled in, as the meta-data of similar works can (and often does) differ. For the moment, we assume that such meta-data is extracted from the preferred work of each group.

That being said, the `IMPORT` procedure can be roughly implemented by deleting all existing notifications, and then for every ORCID group, add a creation notification if no UID of the group is shared by a production in the PTCRIS profile, or create a modification notification with the missing UIDs for every production that shares a UID. Figure 5 presents a possible algorithm . It starts by deleting all existing notifications (line 1) and retrieving the set gs of all work groups that should be considered for importing, namely those with some UIDs (line 2). For each group g to be processed, it computes the set ps of all productions in the PTCRIS profile that share UIDs with g (line 4). If this set is empty then a new creation notification is added to the set of notifications (line 6). The method for adding a creation notification takes as input the UIDs of the new production (in this case the UIDs of the group) and the work from which the meta-data must be computed (in this case the preferred one, the first work of the group). If the set ps is not empty, then for all productions in ps that do not contain all the UIDs in g, a new modification notification is added to insert the missing UIDs (line 9). The method for adding a modification notification that takes as input the key the existing production to be modified and the set of UIDs to be added.

The `IMPORT` procedure can be implemented quite efficiently. The most critical step that requires some care implementing is the one on line 4, determining the set of productions that share some UIDs with the group. A naïve implementation of this step, just by iterating over all productions for every group, would lead to an approximate complexity $\mathcal{O}(|\texttt{workgroups}| \times |\texttt{productions}|)$ (assuming there is an average constant number of UIDs per research output, independent of the number of works and productions present in the user profiles). This complexity can be improved by maintaining a reverse map in the PTCRIS service that, for each UID returns the productions that contain it: this would allow

```
IMPORT(p : PTCRIS) ≐
1 :   p.resetNotifications()
2 :   gs := getActivities(p.oid).workgroups → select(x | x.ids → notEmpty())
3 :   foreach g in gs
4 :       ps := p.productions → select(x | x.ids → intersection(g.ids) → notEmpty())
5 :       if ps → isEmpty()
6 :           p.addCreation(g.ids, getWork(p.oid, g.first().putcode))
7 :       else foreach x in ps
8 :           if not (x.ids → includesAll(g.ids))
9 :               p.addModification(r.key, g.ids − x.ids)
```

Figure 5: The IMPORT procedure.

an implementation of the step in line 5 that quickly narrows the set of matching productions[9]. This optimization would lower the complexity to the class $\mathcal{O}(|\texttt{workgroups}|)$, that is, linear in the number of groups of works currently in the ORCID user profile.

The specified IMPORT procedure is, in fact, the only admissible one, given the desired notion of consistency. In a sense, the import mode is *deterministic*: given two inconsistent profiles there is only one possible update to the PTCRIS profile that will render them consistent (affecting only the notifications and modulo differences in the keys of creation notifications).

4.2 Export

This mode is considerably simpler than the previous one. As mentioned above, the general principle is that every production selected to be exported in a PTCRIS profile should be inserted as a new work in the ORCID profile and then automatically kept up-to-date. In fact, it is expected that a one-to-one correspondence between productions selected to be exported and works whose source is the PTCRIS service is maintained.

On first glance, it seems that the EXPORT procedure could trivially be implemented by just deleting all works whose source is the PTCRIS service and then iterate over all productions selected to be exported and, for each, create a new matching work in the ORCID profile. This naïve procedure would ensure correctness (i.e., the resulting user profiles would be consistent), but would not be stable, since it could unnecessarily change the status of works selected as preferred (if works whose source is the PTCRIS service were set as preferred). Concretely, to avoid unnecessary changes in the preferred selection and preserve stability, the EXPORT synchronization procedure must at least guarantee that preferred works that are to be preserved in ORCID are updated and not deleted and re-introduced, which is possible as of version 2.0 of the ORCID API.

Figure 6 presents a possible algorithm that implements such correct and stable procedure. The algorithm starts by computing the set es of productions selected as exported in the PTCRIS profile (line 1) and the set ws of works in the ORCID profile whose source is the PTCRIS service (line 2). Then, for each work w in ws, it computes the set ps of productions that share some UIDs with w (line 5): if this set is empty, meaning that no matching exported production exists, the work w is deleted from the ORCID profile (line 7); if it is non-empty, it contains at least one matching exported production e, and the pair (w, e) is added to the set us, the set of works that must later be updated (line 9). In the latter case, the production e is also removed from the set es containing the exported productions that still need to be processed. After deleting the works that are no longer exported, the algorithm updates

[9] Notice that, in a standard relational database implementation, assuming that a table associating each production key with its UIDs exists, this reverse mapping can easily be implemented by maintaining an index on the UIDs column.

```
    EXPORT(p : PTCRIS) ≐
1 :     es := p.productions → select(exported)
2 :     ws := getActivities(p.oid).workgroups → collect(works) → select(x | x.source = PTCRIS)
3 :     us := Set{}
4 :     foreach w in ws
5 :         ps := es → select(x | x.ids → intersection(w.ids) → notEmpty())
6 :         if ps → isEmpty()
7 :             deleteWork(p.oid, w.putcode)
8 :         else e := ps → any()
9 :             us := us → including(w, e)
10 :            es := es → excluding(e)
11 :    foreach (w, e) in us
12 :        if (e.ids\w.ids) → notEmpty()
13 :            updateWork(p.oid, w.putcode, newWork(e.ids ∩ w.ids, e.metadata))
14 :    foreach (w, e) in us
15 :        if (w.ids\e.ids) → notEmpty() ∨ (w.ids\e.ids) → isEmpty()
16 :            updateWork(p.oid, w.putcode, newWork(e.ids, e.metadata))
17 :    foreach e in es
18 :        addWork(p.oid, newWork(e.ids, e.metadata))
```

Figure 6: The EXPORT procedure.

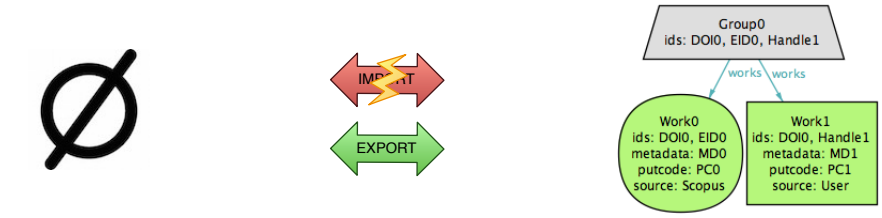
every remaining work w with source PTCRIS to have their meta-data and UIDs consistent with the respective exported production e. However, these updates may cause conflicts with each other and thus must be performed in two stages: the first (line 13) removes obsolete UIDs that may conflict with new UIDs that are introduced in the second (line 16). Finally, the algorithm proceeds by adding a new work for each remaining exported production e (line 18). Likewise to IMPORT, concerning efficiency, the most critical step of this procedure is line 4, and similar measures to those proposed above should be taken to ensure an efficient implementation. The specified EXPORT synchronization procedure is, up to differences in putcode identifiers, the only admissible one, given the desired consistency notion for the export mode.

5 Synchronization Scenarios

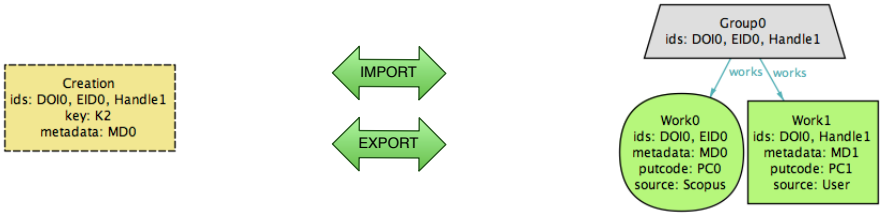
This section presents some scenarios illustrating the specified synchronization procedures, which can be used to validate implementations of the synchronization framework. The notation follows the one already presented at Tables 1 and 2. A more exhaustive set of scenarios is presented in Appendix A.

5.1 IMPORT scenarios

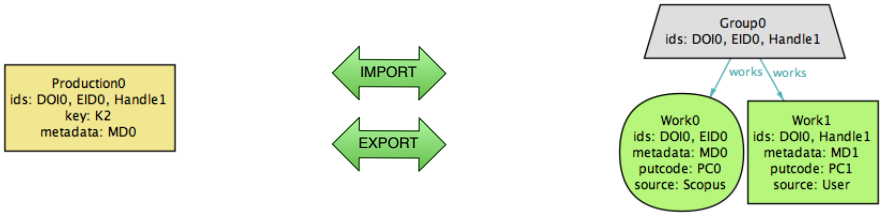
Let us assume a basic scenario where both the PTCRIS and the ORCID profiles of the user are empty (denoted by the $\emptyset$ symbol), until a group of similar works from different external sources is introduced in the ORCID profile, resulting in the following configuration:



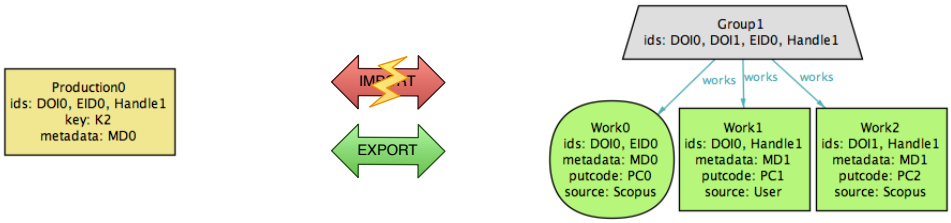
The synchronization procedure will detect a group works in ORCID whose UIDs do not overlap with any of the PTCRIS productions. To synchronize the profiles, the `IMPORT` procedure will create a single creation notification with the UIDs of the group, and with the meta-data extracted from the preferred one (`Work0`) (Scenario 1):



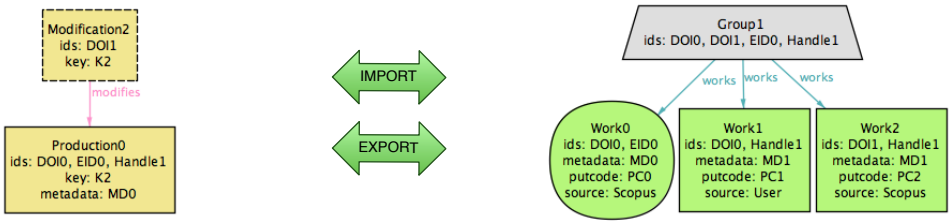
Eventually, the PTCRIS user accepts this creation notification, which results in a new production in his PTCRIS profile. The profiles remain synchronized after this user update:



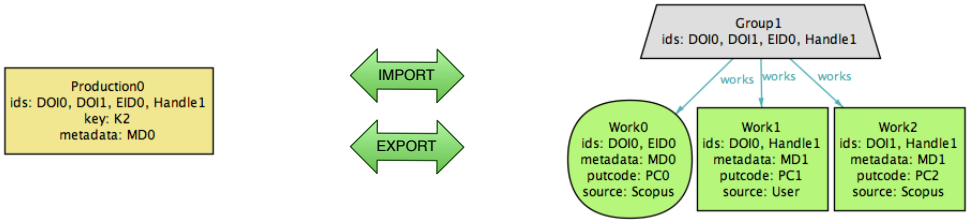
Consider now that one of the external sources introduced a new work in the ORCID profile with a UID (`DOI1`) that is not yet known in the PTCRIS profile:



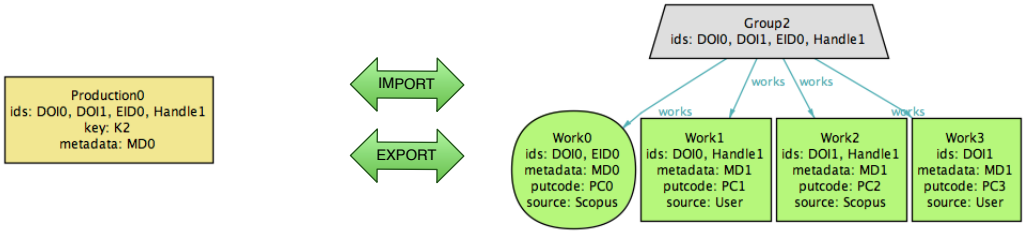
The UIDs of this work are unique among the works from that source (`Scopus`), but it is similar to the existing group by transitivity. The synchronization framework detects this similarity and proposes only a modification notification to introduce the new UIDs in the existing production, rather than a creation notification (Scenario 2):



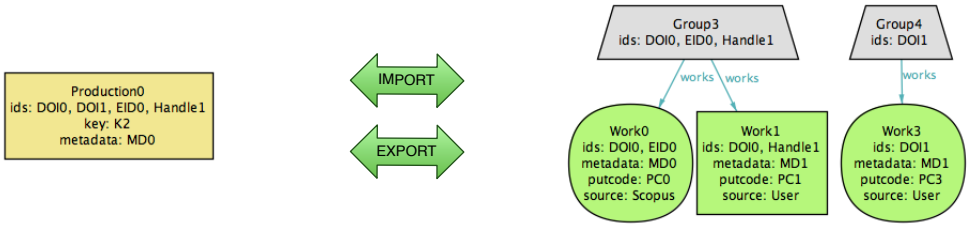
The PTCRIS user eventually accepts the modification notification and introduces the new UIDs into the existing production:



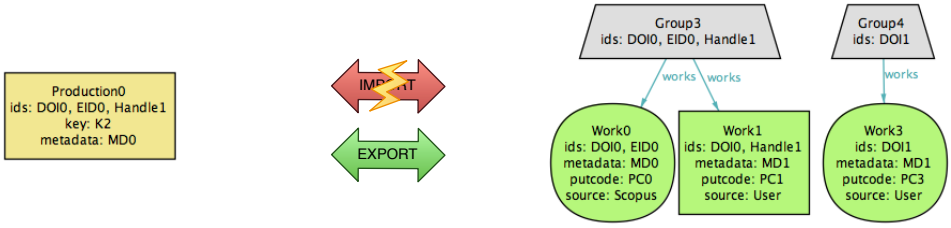
At some point, another similar work is introduced in the ORCID profile (`Work3`), whose UIDs already occur in the PTCRIS profile. The profiles remain synchronized under this update, and thus the PTCRIS profile is not updated, preserving the stability of the system (Scenario 3):



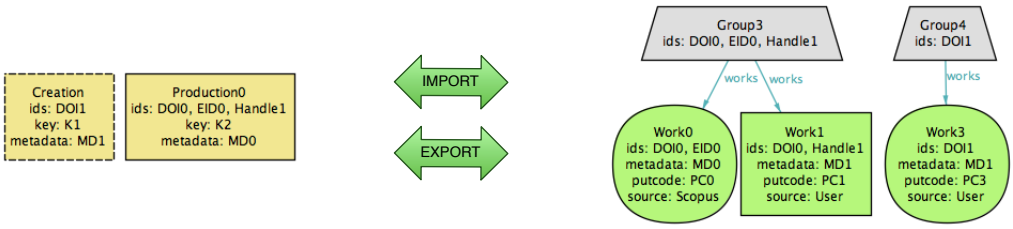
Now, imagine that for some reason (maybe the user finds the meta-data incorrect, maybe he does not trust the source), the user deletes a work that connected all the works into a single group (`Work2`). While there are now two distinct groups of similar works in the ORCID profile, since the production already contains every UID present in them, the profiles remain synchronized (Scenario 4):



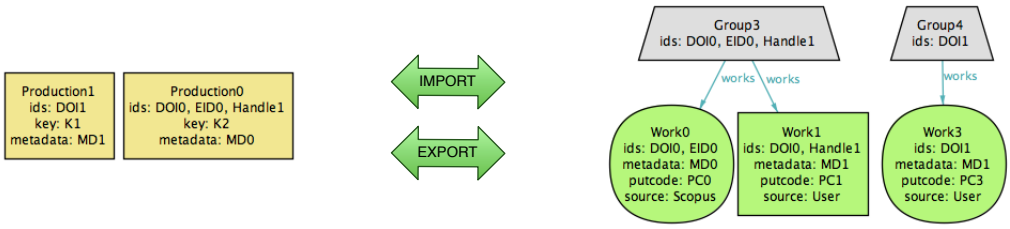
Yet, noticing this new disposition of groups, the user decides to manually remove the UID from the production that connected the two groups (`DOI1`), which will break the constraints:



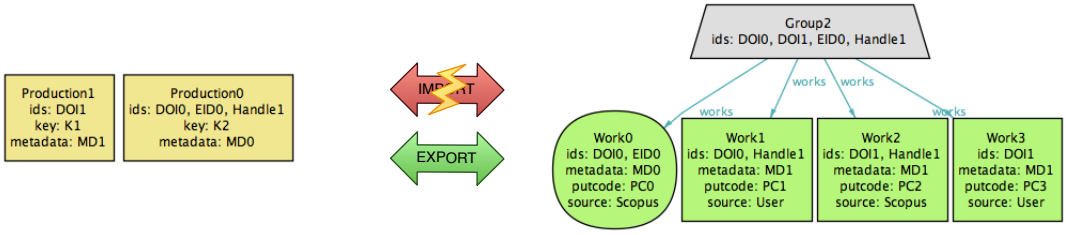
The removed UID occurs in the ORCID profile but is no longer known to the PTCRIS profile. Moreover, there are now no shared UIDs between the production and one of the groups (comprised solely by `Work3`), so the synchronization procedure adds a creation notification, rather than a modification notification (Scenario 5):



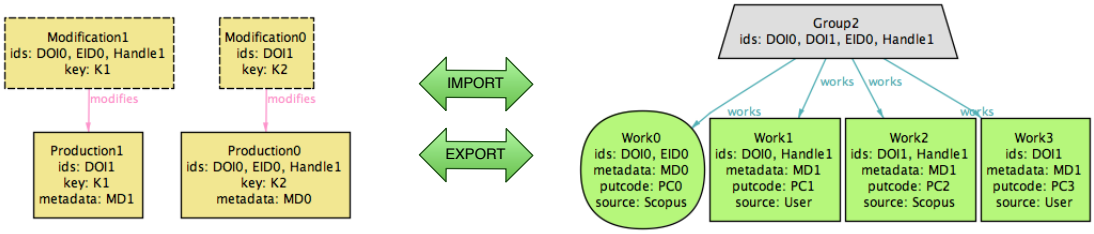
The PTCRIS user eventually accepts the creation notification resulting in a new production in his profile:



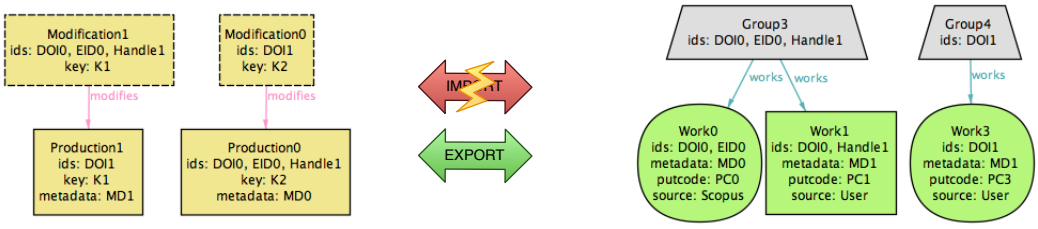
Then, perhaps by distraction (or because he allowed the external sources to automatically insert works), the work previously removed is reintroduced in the ORCID profile, connecting again all the works into a single group of similar works:



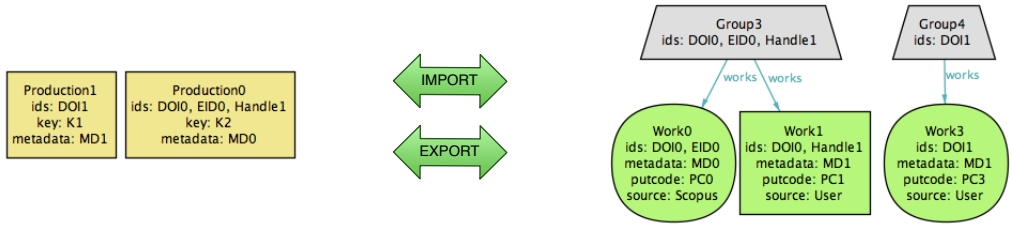
It is not clear how the preferred work is selected when groups are merged since the work order is internal to the groups; here `Work0` was preserved as preferred, but preserving `Work3` would also be a possibility. Either way, both productions now share UIDs with the unified group, so a modification notification must be created for each production in order to synchronize the profiles (Scenario 6):



However, aware that this ORCID work is not to be trusted, the user chooses not to accept the notifications at the PTCRIS profile, and again removes the work from the ORCID profile:

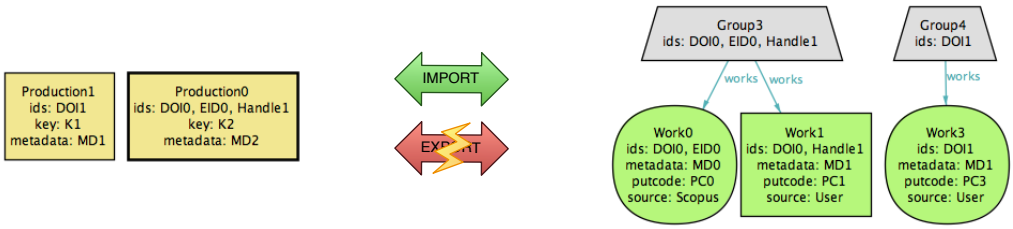


The procedure should then remove the obsolete modification notifications from the PTCRIS profile (Scenario 7):

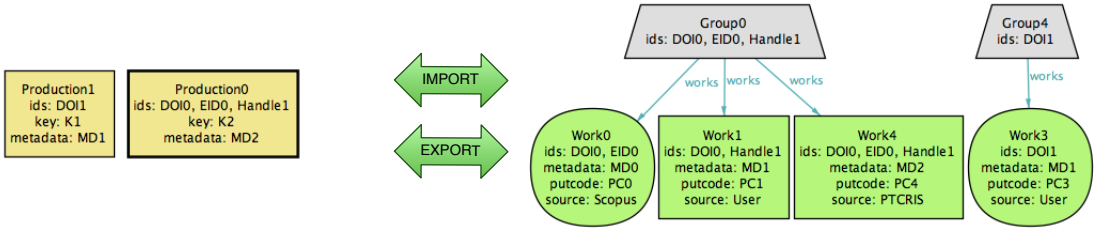


5.2 EXPORT scenarios

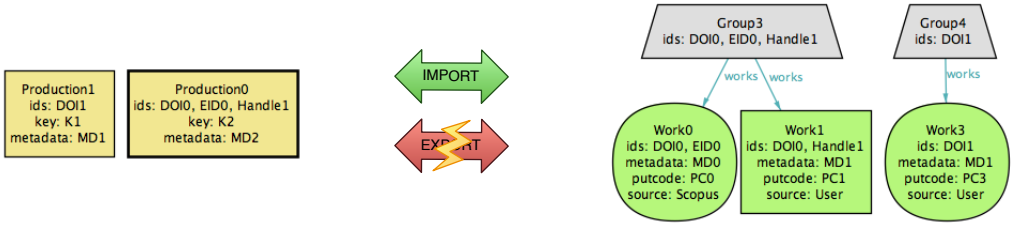
At this point, the user decided to clean up the meta-data (`MD2` in `Production0`) of a production on the PTCRIS side, and have it propagated to the ORCID profile. To do so, he sets it to be exported:



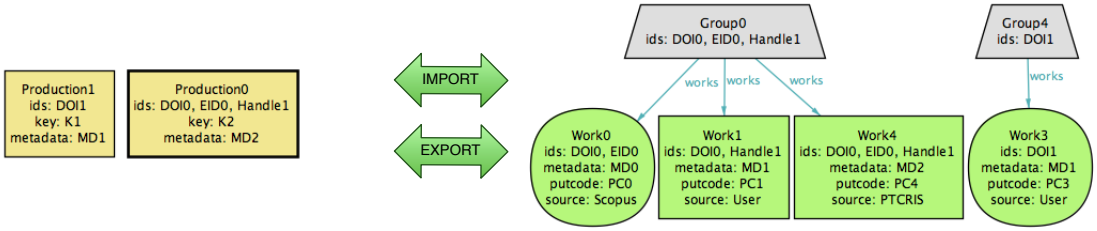
To synchronize, `EXPORT` must create a new work (`Work4`) in the ORCID profile identical to the exported production (Scenario 8):



The EXPORT procedure has only the ability to update the ORCID profile, so, when a work that has the PTCRIS service as source is deleted by the user using the ORCID web interface:

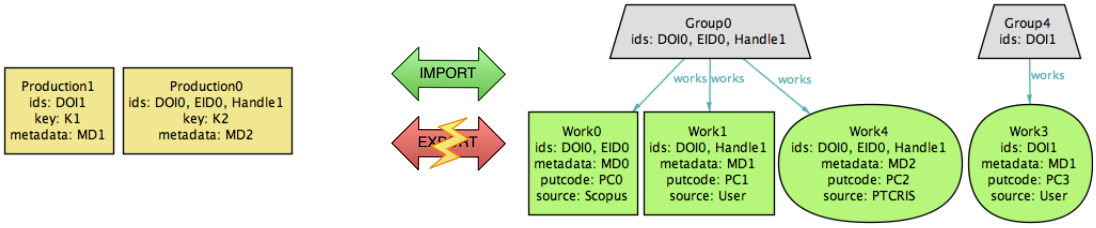


The procedure will re-introduce it, since it is still selected to be exported in the PTCRIS profile (Scenario 9):

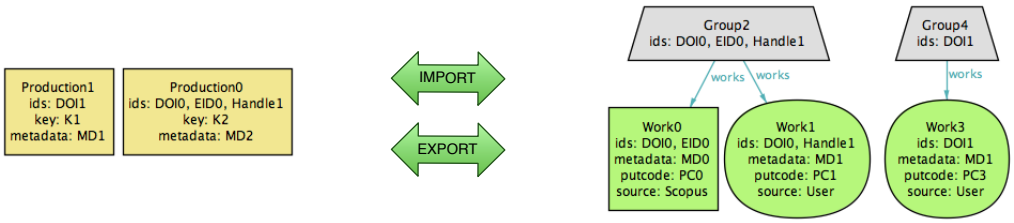


To effectively remove works from the ORCID profile inserted by the PTCRIS service, the user must deselect them as exported in the PTCRIS profile.

Since the user manually updated the meta-data from this production, he assumes that it is the best one, and sets the respective work as preferred in the ORCID profile. This does not affect the consistency of the profiles. If he now deselects the production from being exported:

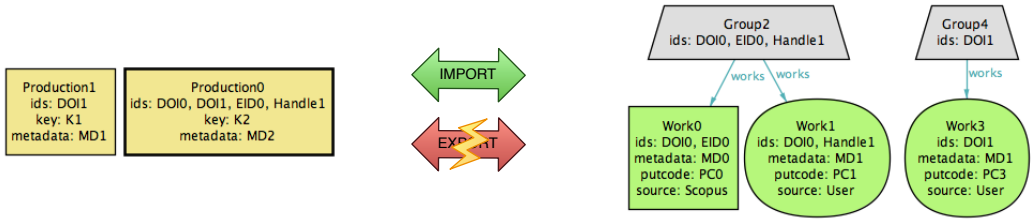


The procedure will remove the work whose source was the PTCRIS service form the ORCID profile. Since this work was selected as the preferred one, the succeeding work is promoted to preferred. The order on works inside a group is not represented, but assuming that `Work1` was the succeeding work, the result would be (Scenario 10):

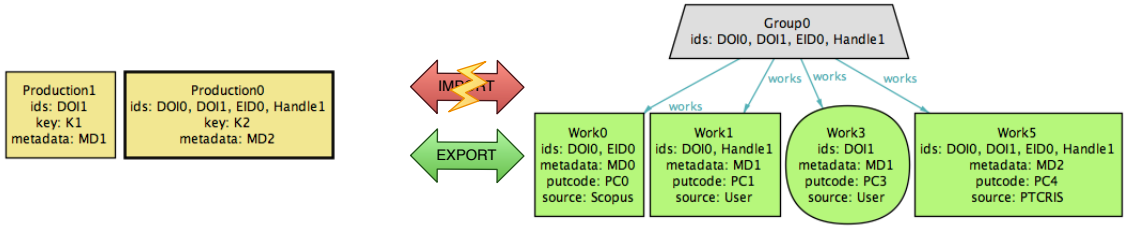


5.3 SYNC scenarios

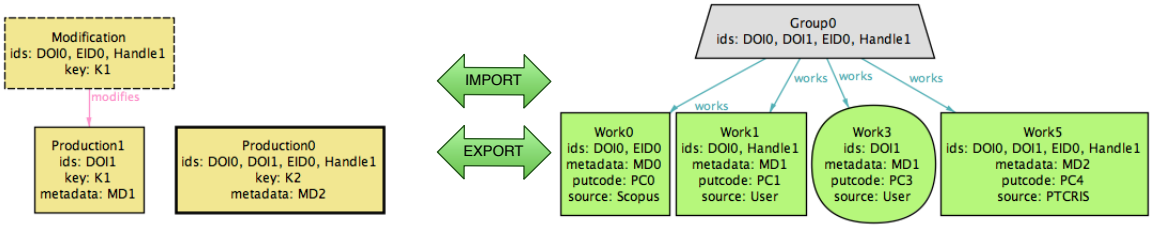
Consider that the user introduced the ambiguous UID in the production and selected it to be exported:



As expected, EXPORT will create a new identical work in the ORCID profile:



However, since this new work unifies the two pre-existing groups of the ORCID profile, the user should now be notified to introduce the remainder UIDs in the other production of the PTCRIS profile, so IMPORT must be subsequently executed (Scenario 12):



References

[1] Daniel Jackson. *Software Abstractions: Logic, Language, and Analysis*. MIT Press, revised edition, 2012.

[2] João Mendes Moreira, Alcino Cunha, and Nuno Macedo. An ORCID based synchronization framework for a national CRIS ecosystem. *F1000Research*, 4(181), 2015.

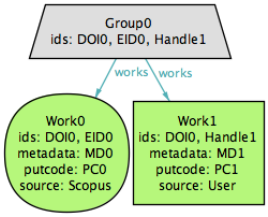
A Scenarios

Scenario 1

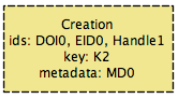
This scenario depicts the introduction of group of works in the ORCID profile whose UIDs are not shared with any production in the PTCRIS profile.

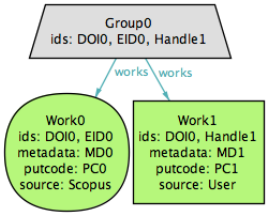
Consistent profiles: Empty PTCRIS and ORCID profiles.

ORCID update (⇒`IMPORT` inconsistency): A group of similar works (`Work0` and `Work1` through `DOI0`) is introduced in the ORCID profile.



`IMPORT` executed: `IMPORT` introduces a creation notification in the PTCRIS profile to synchronize the profiles.





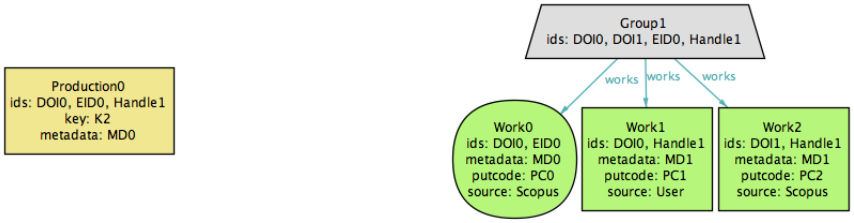
Observations: The meta-data is assumed to have been extracted directly from the preferred work (`Work0`).

Scenario 2

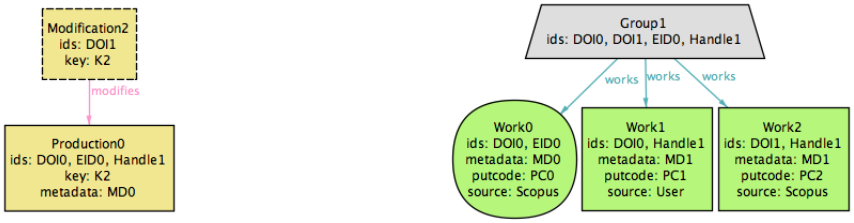
This scenario depicts the introduction of new work in the ORCID profile that is similar to existing works but introduces new UIDs.

Consistent profiles: A production in the PTCRIS profile matches a group of similar works in the ORCID profile.

ORCID update (⇒`IMPORT` inconsistency): A new work (`Work2`), similar to the existing works through `Handle1`, is added to the ORCID profile, introducing a new UID (`DOI1`).



`IMPORT` executed: Since the existing `Production0` in the PTCRIS profile shares UIDs with the group at the ORCID profile, `IMPORT` adds only a modification notification over the production so that the new UID (`DOI1`) is introduced.

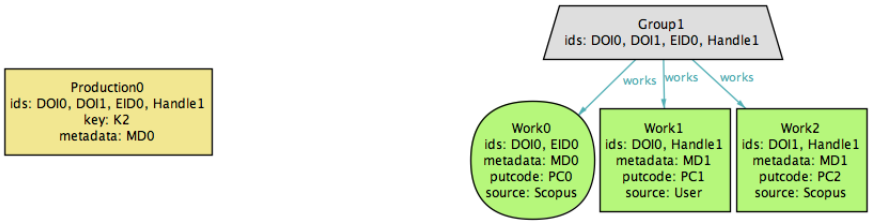


Observations: The modification notifications are assumed to contain only the newly found UIDs.

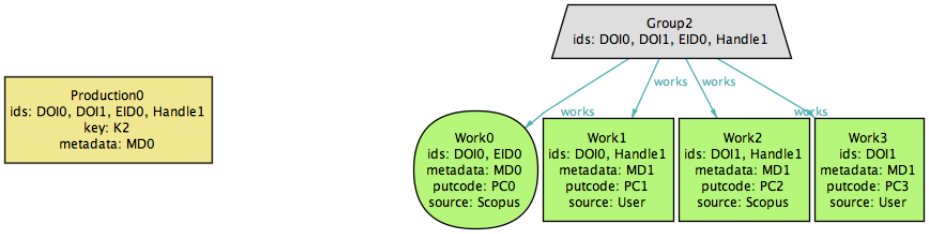
Scenario 3

This scenario depicts the introduction of new work in the ORCID profile that is similar to existing works and does not introduce new UIDs.

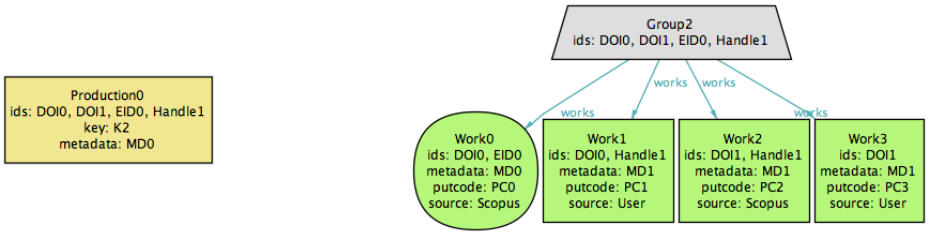
Consistent profiles: A production in the PTCRIS profile matches a group of similar works in the ORCID profile.



ORCID update (⇒`IMPORT` inconsistency): A new work (`Work3`), similar to the existing works through `DOI1`, is added to the ORCID profile, without introducing new UIDs.



`IMPORT` executed: Since `Production0` in the PTCRIS profile already contains all the UIDs of the group, the profiles are still consistent, so `IMPORT` does not perform any change.

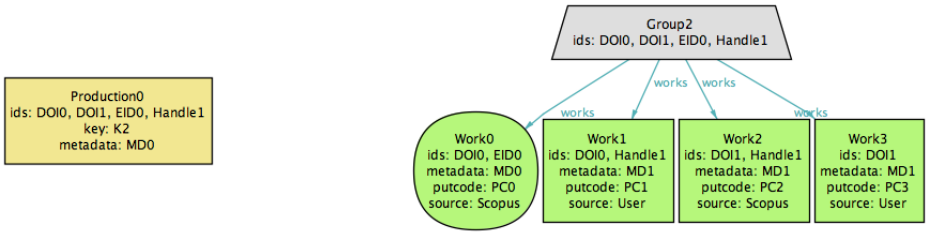


Observations: NA

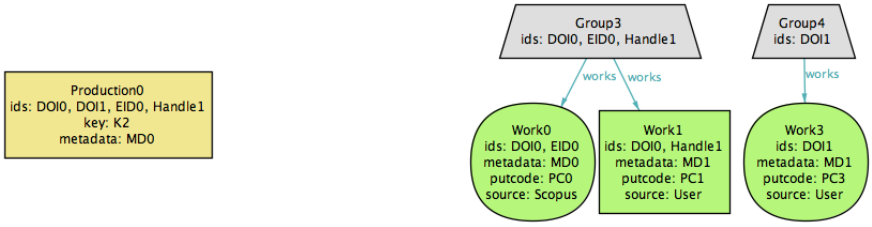
Scenario 4

This scenario depicts the deletion of a work in the ORCID profile that splits a group of similar works into two distinct works.

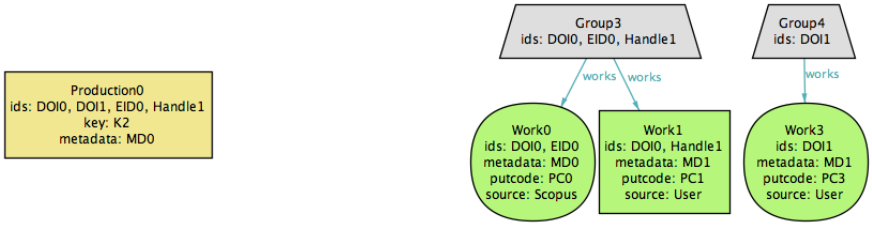
Consistent profiles: A production in the PTCRIS profile matches a group of similar works in the ORCID profile.



ORCID update (⇒`IMPORT` inconsistency): The work that rendered all works similar (`Work2` through `DOI1` and `Handle1`) is removed, splitting the group into two.



`IMPORT` executed: Since `Production0` in the PTCRIS profile already contains all the UIDs of the two groups, the profiles are already consistent and `IMPORT` does not perform any change.

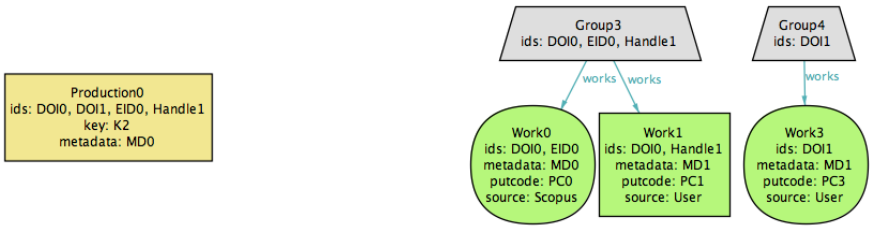


Observations: When a group of works is split, one of the works must be promoted to the preferred of the new group. This procedure relies on the internal order of the works in the original group (not represented in the diagrams).

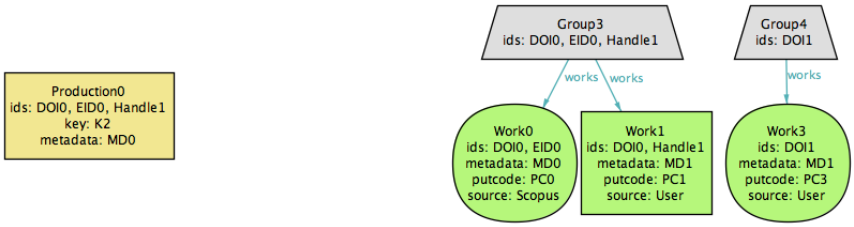
Scenario 5

This scenario depicts the deletion of a UID in a production of the PTCRIS profile, so that it no longer represents every group at the ORCID profile.

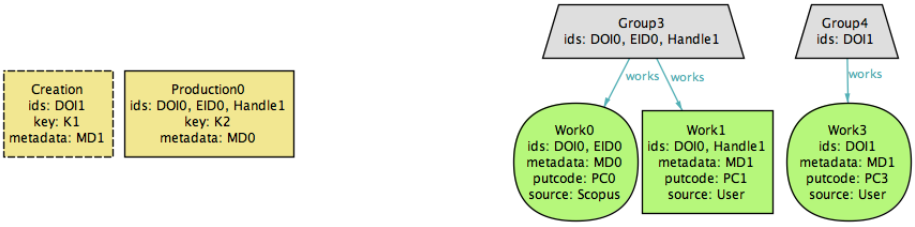
Consistent profiles: A production in the PTCRIS profile matches two distinct groups of similar works in the ORCID profile.



PTCRIS update (⇒`IMPORT` inconsistency): A UID (`DOI1`) is removed from the production, so that it no longer matches with one of the groups in the ORCID profile (`Work3`).



`IMPORT` executed: Since the production does not share any UID with `Work3` (`Group4`), `IMPORT` creates a new creation notification in the PTCRIS profile.

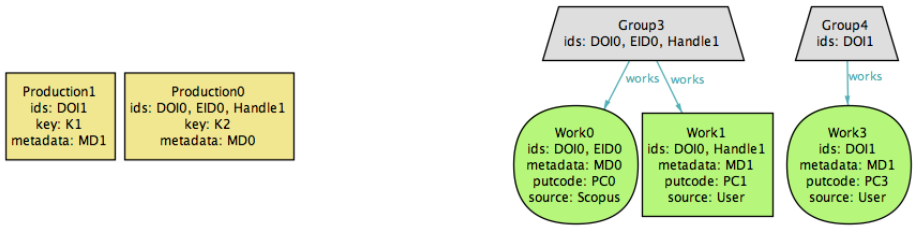


Observations: NA

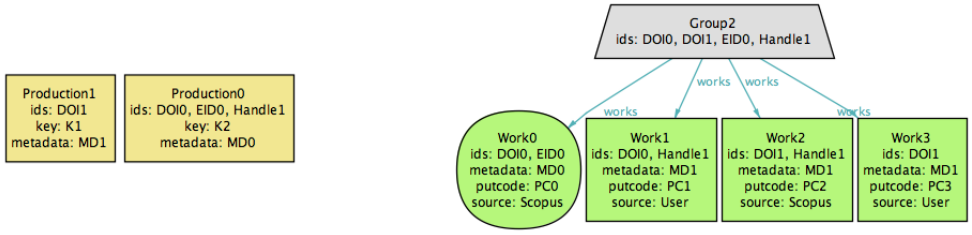
Scenario 6

This scenario depicts the introduction of an ORCID work that unifies two previously distinct groups of similar works into one, that now shares UIDs with two productions at the PTCRIS profile.

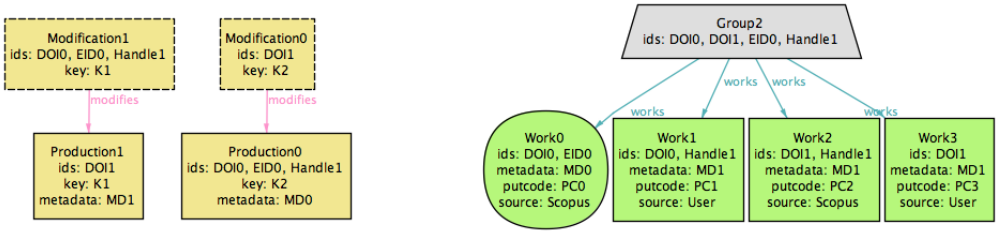
Consistent profiles: Two groups of similar works in the ORCID profile, each matched by a production in the PTCRIS profile.



ORCID update (⇒`IMPORT` inconsistency): A new work (`Work2`) is introduced in the ORCID profile that connects the two distinct groups of similar works.



`IMPORT` executed: Since the group of similar works now shares UIDs with both productions at the PTCRIS profile, `IMPORT` creates a modification notification to introduce the missing UIDs for each of the productions.

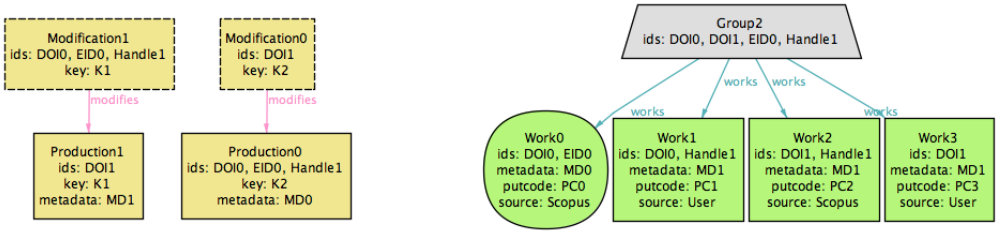


Observations: The synchronization framework imposes a modification notification for every production that shares UIDs. The outcome of this scenario depends on the behavior of the ORCID service when groups are merged: since the order on works is internal to the groups, it is not clear how the preferred work of the group is selected. In this scenario `Work0` was selected, but `Work3` would also have been a valid option.

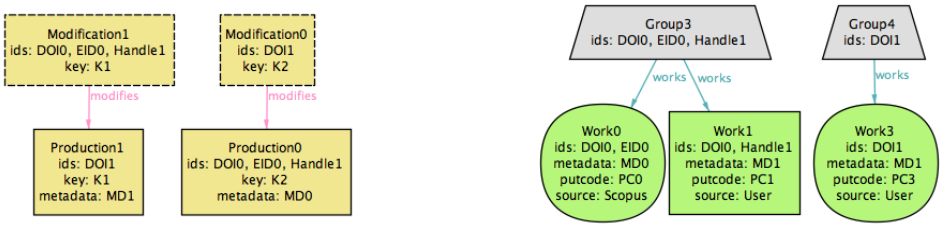
Scenario 7

This scenario depicts the removal of works from the ORCID profile that renders some notifications in the PTCRIS profile obsolete.

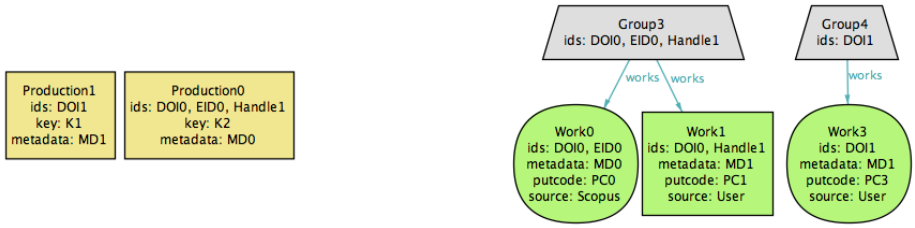
Consistent profiles: A group of similar works in the ORCID profile shares UIDs with two productions in the PTCRIS profile, to which modification notifications were assigned to introduce the additional UIDs.



ORCID update (⇒`IMPORT` inconsistency): A work was removed, splitting the group into two, such that the UIDs of each group now match exactly one production in the PTCRIS profile.



`IMPORT` executed: Since the UIDs of each group match exactly the UIDs of a production, the modification notifications are no longer valid, and must be removed by `IMPORT`.

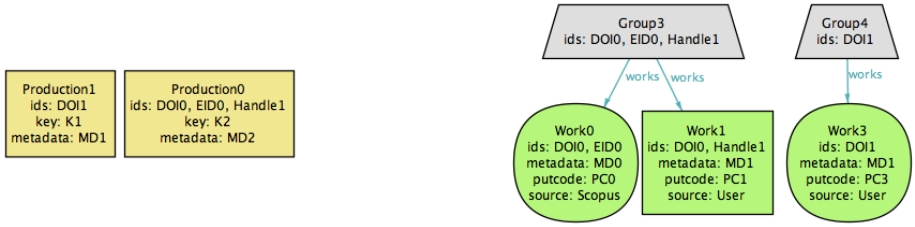


Observations: This scenario depicts the occurrence of an obsolete creation notification. See Scenario 4 for the discussion on splitting groups.

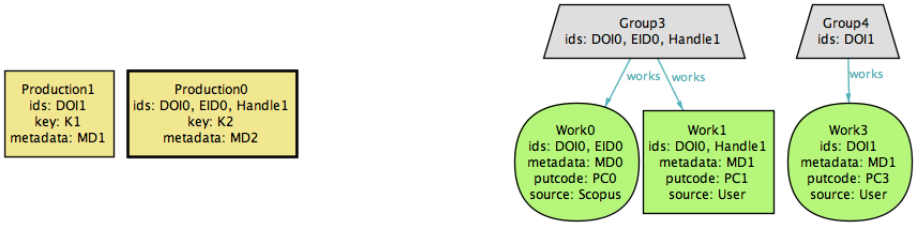
Scenario 8

This scenario depicts the selection of a production to be exported when there is already a group int the ORCID profile that shares its UIDs.

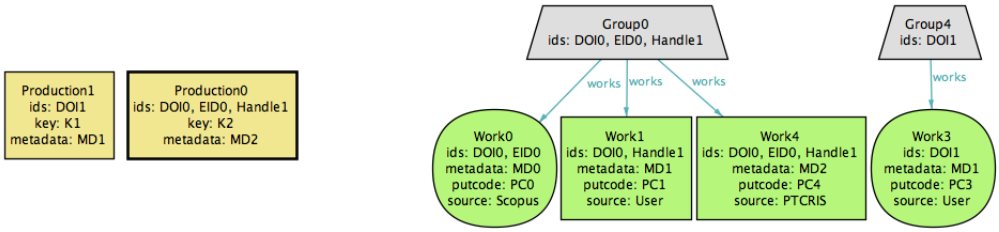
Consistent profiles: Two productions in the PTCRIS profile, each sharing UIDs with a groups of similar works in the ORCID profile.



PTCRIS update (⇒`EXPORT` inconsistency): One of the productions (`Production0`) is selected to be exported, requiring the creation of an identical work in the ORCID profile.



`EXPORT` executed: The `EXPORT` procedure creates the new work (`Work4`) with the PTCRIS service as the source, that is grouped with the existing works that share UIDs.

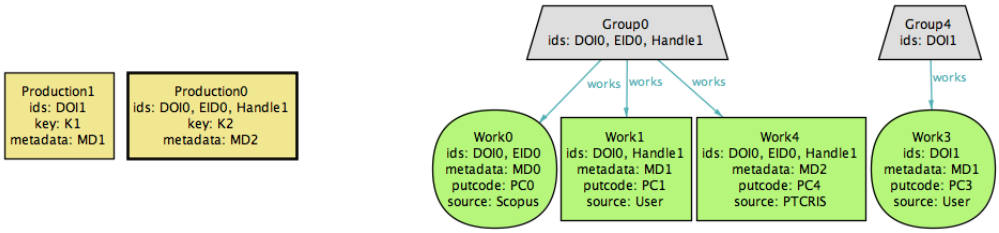


Observations: NA

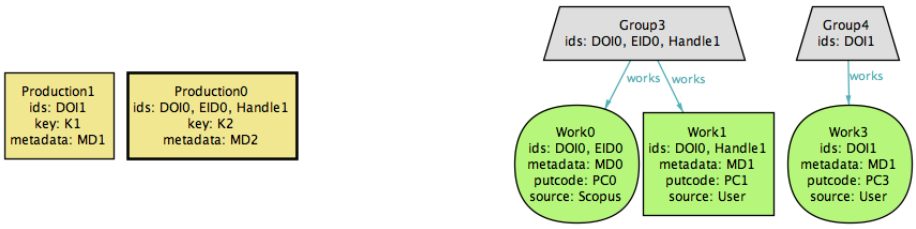
Scenario 9

This scenario depicts the removal of a work from the ORCID profile that was created by the PTCRIS service from a production selected to be exported.

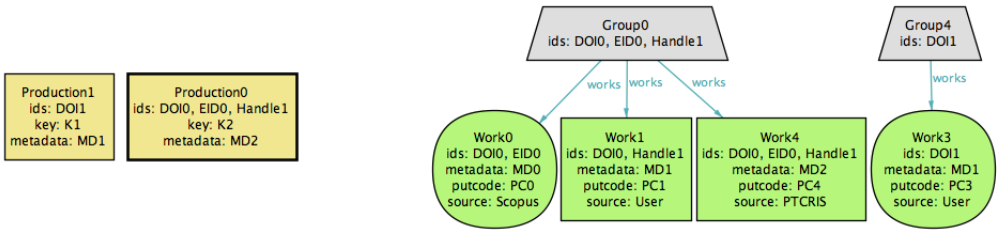
Consistent profiles: A production from the PTCRIS profile is selected to be exported (`Production0`) and the matching work is already present in the ORCID profile (`Work4`).



ORCID update (⇒`EXPORT` inconsistency): The user removes the work whose source was the PTCRIS service from the ORCID profile (`Work4`).



`EXPORT` executed: Since the production selected to be exported no longer has a matching work at the ORCID profile, the work is re-introduced by `EXPORT` (`Work4`).

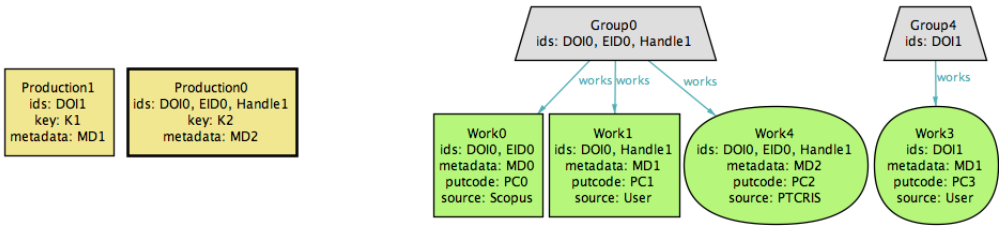


Observations: Reflects the design decision to not allow `EXPORT` to change the selection of exported productions.

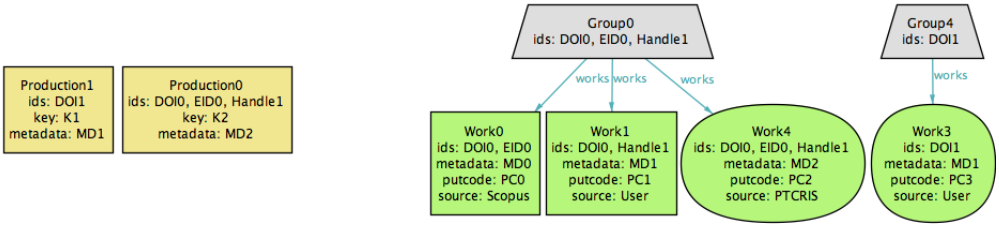
Scenario 10

This scenario depicts the deselection of an exported work when the matching work at the ORCID profile was selected as the preferred work.

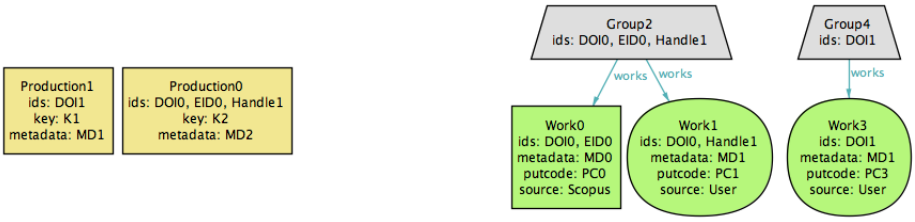
Consistent profiles: A production from the PTCRIS profile is selected to be exported (`Production0`) and the matching work is already created in the ORCID profile (`Work4`). This work was selected by the user as the preferred in its group (`Group0`).



PTCRIS update (⇒`EXPORT` inconsistency): The user deselects the production to be exported (`Production0`).



`EXPORT` executed: The work whose source was the PTCRIS service (`Work4`) no longer has a matched exported production in the PTCRIS profile, and thus must be deleted by `EXPORT`. Since it was the preferred of the group, one of the other similar works must be promoted.

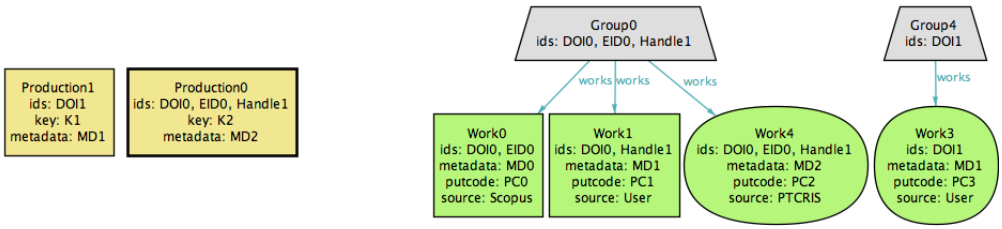


Observations: When the preferred work of a group is deleted, the succeeding work of the group's internal order is promoted. This order is not depicted in the scenarios, but `Work1` was assumed to be the work succeeding `Work4`.

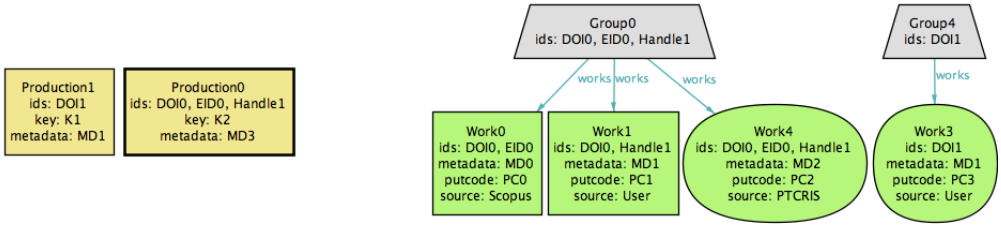
Scenario 11

This scenario depicts an update on an exported production in the PTCRIS profile when the matching work at the ORCID profile was selected as the preferred work.

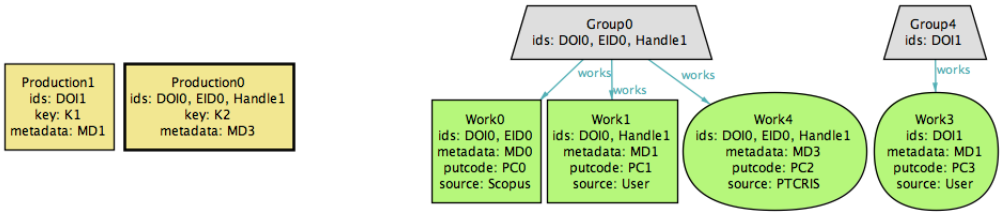
Consistent profiles: A production from the PTCRIS profile is selected to be exported (`Production0`) and the matching work is already present in the ORCID profile (`Work4`). This work was selected by the user as the preferred in its group (`Group0`).



PTCRIS update (⇒`EXPORT` inconsistency): The user updates the production to be exported (`Production0` through `MD3`).



`EXPORT` executed: The work whose source was the PTCRIS service (`Work4`) is no longer identical to the exported production, and must be updated by `EXPORT` with meta-data `MD3`.

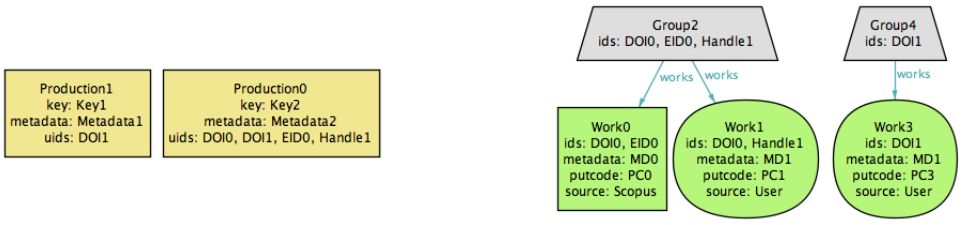


Observations: This scenario assumes that the ORCID API allows services to update their works, not requiring the deletion and re-insertion of the work, thus preserving its selection as preferred. It denotes an ORCID update that needs only the update of the meta-data, relevant for the two-stage work update procedure.

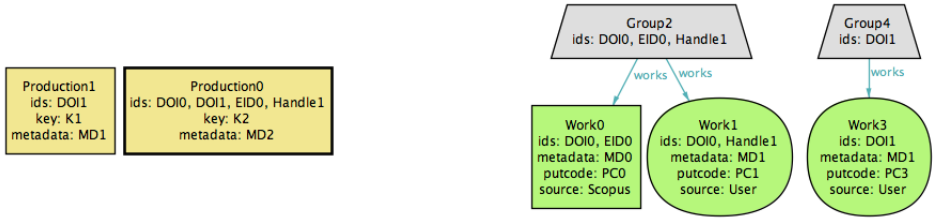
Scenario 12

This scenario depicts the exportation of a production that unifies two groups of works in the ORCID profile, leading to the need of further UIDs updates in the PTCRIS profile, and thus to the subsequent execution of the `IMPORT` procedure.

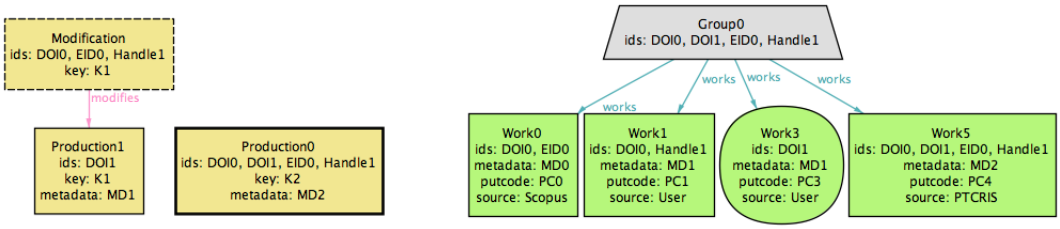
Consistent profiles: Two productions in the PTCRIS profile share UIDs with each other and with two groups of similar works in the ORCID profile.



PTCRIS update (⇒`EXPORT` inconsistency): One of the productions (`Production0`) is selected to be exported, and since it shares UIDs with both groups of works will unify them under a single group of similar works.



`SYNC` executed: To recover full consistency, first `EXPORT` must create the work at the ORCID profile whose source is the PTCRIS service (`Work5`). This will unify the two groups, and thus the UIDs must be introduced in all matched productions in the PTCRIS profile (including `Production1`), which renders the profiles `IMPORTED`-inconsistent. The subsequent execution of `IMPORT` creates the modification notification.



Observations: This scenario depicts the need for a `SYNC` procedure in case both import and export modes are to be enforced. See Scenario 6 for the discussion on merging groups.

Scenario 13

This scenario depicts the merging of two previously exported productions into a single one in the PTCRIS, by joining their UIDs.

Consistent profiles: Two productions from the PTCRIS profile are selected to be exported (`Production0` and `Production1`) and the matching works are already present in the ORCID profile (`Work0` and `Work1`, respectively), forming two distinct groups.

PTCRIS update (⇒`EXPORT` inconsistency): The user adds the UIDs of one of the productions to the other (`Handle1` to `Production0`), removing the other from the exported (`Production1`).

`SYNC` executed: One of the works must be deleted from the ORCID profile since there is now only one production exported; the other must be updated to contain both UIDs (`Handle1` and `DOI0`). The production with a single UID also matched the ORCID group, thus `IMPORT` must be executed to create a modification notification.

Observations: For this scenario to work properly, the PTCRIS service must first remove the extra work from the ORCID profile, otherwise the insertion of the novel UID will fail due to being duplicated. The selection of the preserved work is arbitrary. It also denotes an ORCID update that requires the insertion of UIDs, relevant for the two-stage work update procedure.

Scenario 14

This scenario depicts the splitting of a previously exported production into two productions in the PTCRIS profile, by separating their UIDs.

Consistent profiles: A production from the PTCRIS profile (`Production0`) with two UIDs is selected to be exported and the matching work is already present in the ORCID profile (`Work0`).

PTCRIS update (⇒`EXPORT` and `IMPORT` inconsistency): The user splits the UIDs creating a new production (`Production1` with `Handle1`), setting both to be exported.

`EXPORT` executed: One of the UIDs must be removed from the existing work in the ORCID profile, while a new one must be created with the other UID.

Observations: Likewise the previous scenario, to work properly the PTCRIS service must first update the existing work at the ORCID profile, otherwise the insertion of the new work will fail due to duplicated UIDs. The selection of which UID is preserved is arbitrary. Although the PTCRIS update renders the profiles temporarily `IMPORTED` inconsistent, this is fixed as the PTCRIS-sourced works are updated in the ORCID profile. It also denotes an ORCID update that requires the removal of UIDs, relevant for the two-stage work update procedure.

Scenario 15

This simple scenario depicts the fact that works are matched even if the UIDs of the production are not totally contained in the ORCID group.

Consistent profiles: A production in the PTCRIS profile (`Production0`) with two UIDs.

ORCID update (⇒`IMPORT` inconsistency): A new group is added to the ORCID profile (`Work0`) that matches `Production0` with one, but not every, UID (`DOI0`.

`IMPORT` executed: A modification notification must be created to introduce the additional UID of the ORCID group (`Handle1`).

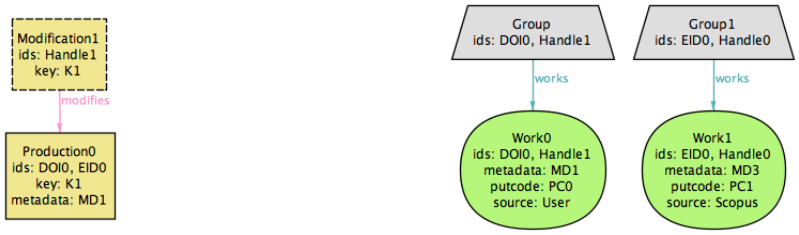
Observations: NA

Scenario 16

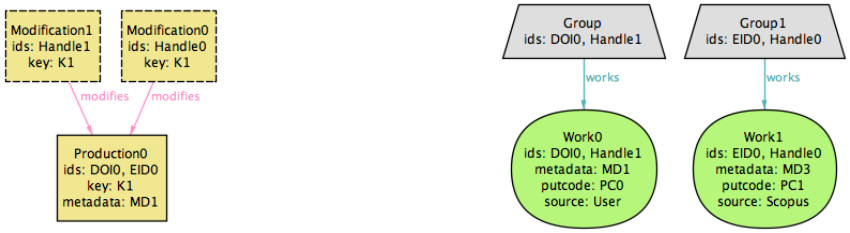
This scenario depicts the matching of a production in the PTCRIS with multiple works in the ORCID profile.

Consistent profiles: A production from the PTCRIS profile (`Production0`) with two UIDs is matched with a group in the ORCID profile (`Work0`) that induces a modification notification.

ORCID update (⇒`IMPORT` inconsistency): A new independent group is add to the ORCID profile (`Work1`) that also matches with `Production0` and introduces new UIDs (`Handle0`).



`IMPORT` executed: Another modification notification must be created to represent the UIDs of the new mathcing group.

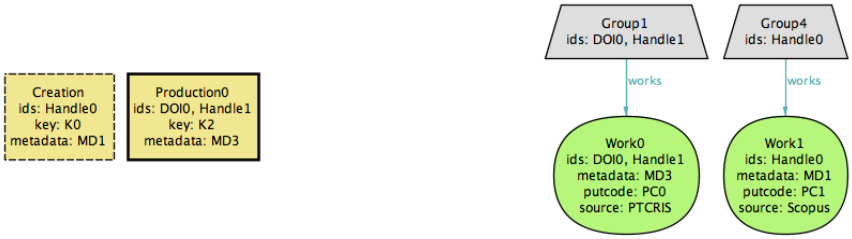


Observations: The specification currently assumes an independent modification notification for each matching group in the ORCID profile, rather than a single one containing every newly found UID.

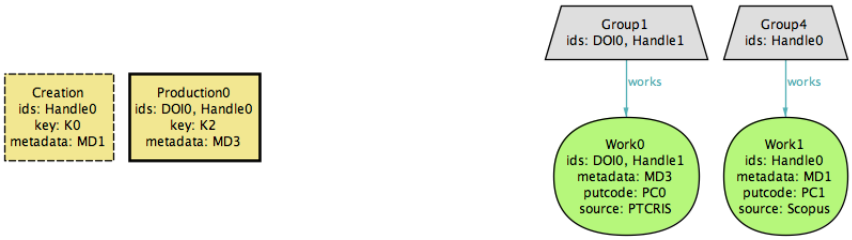
Scenario 17

This scenario depicts an update of an exported production that removes UIDs and introduces new ones.

Consistent profiles: A production in the PTCRIS profile is selected to be exported (`Production0`) and the matching work is already present in the ORCID profile (`Work0`) but is out of date. Another independent group in ORCID (`Work1`) has also induced a creation notification in the PTCRIS profile.



PTCRIS update (⇒`EXPORT` and `IMPORT` inconsistency): The user changes the UIDs of one of the production, matching the independent group, rendering the notification obsolete. The PTCRIS-sourced work must be updated accordingly.



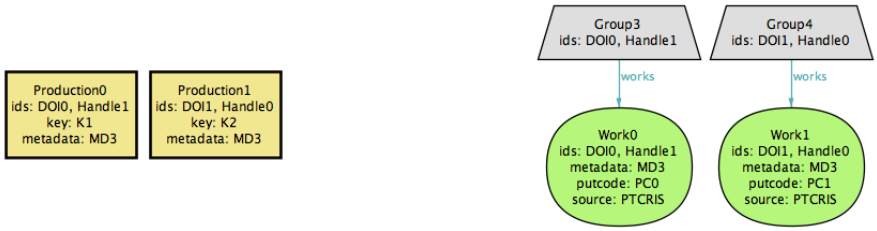
`SYNC` executed: The PTCRIS-sourced work is updated and the creation notification removed.

Observations: This scenario depicts the occurrence of an obsolete creation notification. It also denotes an ORCID update that requires the insertion and removal of UIDs, relevant for the two-stage work update procedure.

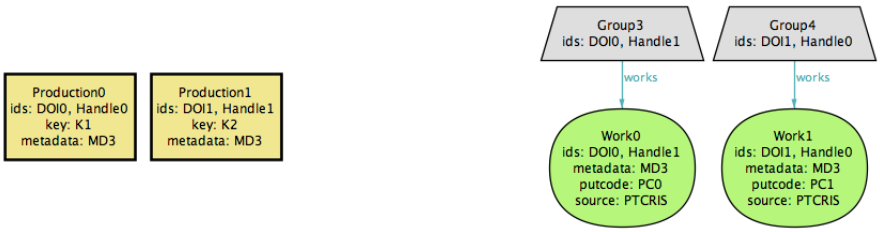
Scenario 18

This scenario depicts simultaneous ORCID updates that may conflict with each other due to UID swapping.

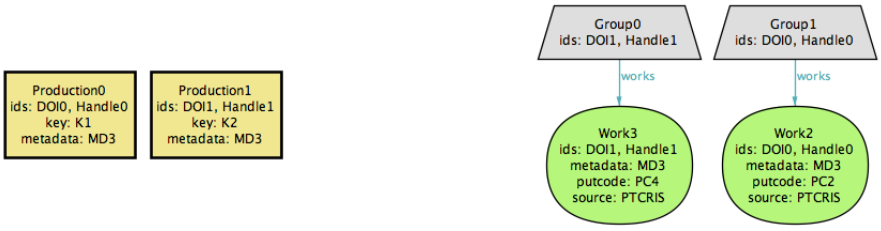
Consistent profiles: Two productions from the PTCRIS profile are selected to be exported (`Production0` and `Production1`) and the matching works are already present in the ORCID profile (`Work0` and `Work1`, respectively), forming two distinct groups.



PTCRIS update (⇒`EXPORT` and `IMPORT` inconsistency): The user swaps the UIDs of the productions (`Handle0` and `Handle1`), which requires the update of the PTCRIS-sourced works.



`EXPORT` executed: Both PTCRIS-source works are updated to the new UID configuration.



Observations: To operate properly, this procedure cannot simply apply two subsequent updates, since this would generate a conflict due to overlapping UIDs. Thus, it must be performed in two phases, the first removing spurious UIDs, the second inserting the new ones.

Scenario 19

This scenario depicts productions in PTCRIS profile selected to be exported, already present in the ORCID profile, that were not updated by the user.

Consistent profiles: A production in the PTCRIS profile selected to be exported (`Production0`), already exported in the ORCID profile (`Work0`).

No updates: No updates are performed by the user, consistency is preserved.

`EXPORT` executed: The execution of the `EXPORT` procedure should not modify the profile.

Observations: This scenario is useful to demonstrate that consistent profiles should not be changed.

Scenario 20

This boundary scenario depicts the relationship between empty PTCRIS and ORCID profiles.

Consistent profiles: Two empty profiles.


No updates: No updates are performed by the user, consistency is preserved.


`EXPORT` executed: The execution of the `EXPORT` procedure should not modify the profile.


Observations: This scenario is useful to test the boundary case.