Preparing Proposals in LaTeX with `proposal.cls`*

Michael Kohlhase
Computer Science, Jacobs University Bremen
http://kwarc.info/kohlhase

November 24, 2016

Abstract

The `proposal` class supports many of the generic elements of Grant Proposals. It is optimized towards collaborative projects, and should specialized to particular funding agencies.

Contents

---

*Version ? (last revised ?)

1 Introduction

Writing grant proposals is a collaborative effort that requires the integration of contributions from many individuals. The use of an ASCII-based format like LaTeX allows to coordinate the process via a source code control system like GIT or SUBVERSION, allowing the proposal writing team to concentrate on the contents rather than the mechanics of wrangling with text fragments and revisions. In fact the `proposal` package has evolved out of a series of collaborative proposal writing efforts, where large teams (up to 30 individuals from up to 20 sites) have written a 100-page proposal in three weeks (with over 2000 commits). Such collaborative writing sprints are impossible without a revision control system and a "semantic" document class that generates tables, charts, and deliverable lists from content markup and thus takes care of many of the routine tasks of keeping information consistent.

The `proposal` class supports many of the generic elements of Grant Proposals. The package documentation is still preliminary, fragmented and incomplete.

The `proposal` class is distributed under the terms of the LaTeX Project Public License from CTAN archives in directory `macros/latex/base/lppl.txt`. Either version 1.0 or, at your option, any later version.

The CTAN archive always contains the latest stable version, the development version can be found on GitHub at https://github.com/KWARC/LaTeX-proposal. For bug reports please use the issue tracker there. Please feel free to fork the repository and provide extensions and improvements.

The development version also contains example proposals and a very useful script that generates GitHub issues for all the workpackages, tasks, and deliverables. This is a great way of starting up a project and controlling its progress. The OpenDreamKit EU project (see http://opendreamkit.org) uses this for its (very public) project planning on the issue tracker at https://github.com/OpenDreamKit after (also publicly) developing the proposal on GitHub.

Finally, the GitHub repositry contains example project proposals and specialized Makefiles that help start off the proposal development process. These are not part of the CTAN/TeXLive distributions.

2 The User Interface

In this section we will describe the functionality offered by the `proposal` class along the lines of the macros and environments the class provides.

2.1 Package Options

The `proposal` package takes the options `submit`, `noworkareas`, `RAM`, `deliverables`, `wpsubsection`, `keys`, `svninfo`, `gitinfo`, and `public`.

submit   The `submit` option will disable various proposal management decorations which are enabled by default for submission.

noworkareas   The `noworkareas` option specifies that we do not want to structure our work plan into work areas (see section 2.6).

RAM   The `RAM` option specifies that we specify research assistant months in the effort tallies (see section 2.6).

deliverables   The `deliverables` option specifies that we specify deliverables in the grant proposal (see section 2.9). As the deliverables management needs extra support, we only activate them via this option.

wpsubsection   The `wpsubsection` option specifies that we want to see subsections headings for the WPs (and WAs, if we have them).

report   The `report` option specifies that we want to use the `report.cls` class as a basis for `proposal` instead of the default `article.cls`.

keys   The `keys` option specifies that we want to see the values of various keyval arguments in the

margin.

svninfo   The `svninfo` option specifies specifies that we want to use the `svninfo` package for displaying version control metadata in the document (except when the `submit` option is also given). For this we need the `svninfo` metadata line of the form

```
\SVN $Id: proposal.tex 13610 2007-07-11 04:30:16Z kohlhase $
\svnKeyword $HeadURL: https://svn.kwarc.info/../proposal.tex $
```

at the beginning of each file (or in the preamble).

gitinfo   Analogously, the `gitinfo` option uses the `gitinfo2` package for GIT metadata. Note that you will need to install the post-commit hooks in your working copy according to [Lon] for this to work.

public   Finally, the `public` option allows to hide certain sensitive (e.g. financial) parts of the proposal.
private   For this, the `proposal` class provides the `private` environment. If the option `public` is set, the parts of the document between `\begin{private}` and `\end{private}` do not produce output. This is useful for producing public versions of the proposal that hide confidential parts. Note that both `\begin{private}` and `\end{private}` *have to be on lines of their own may not have any leading whitespace* otherwise an error occurs and LaTeX gives error messages that are difficult to comprehend. An alternative way to distinguish private and public sections are to use the
\ifpublic   `\ifpublic` conditional: `\ifpublic{3}\else{5}\fi` will result in "5" in the submitted draft and "3" in the public document.

2.2 Proposal Metadata

proposal   The metadata of the proposal is specified in the `proposal` environment, which also generates the title page and the first section of the proposal as well as the last pages of the proposal with the signatures, enclosures, and references. The `proposal` environment should contain all the mandatory parts of the proposal text. The `proposal` environment uses the following keys to specify metadata.

title
- `title` for the proposal title (used on the title page),

instrument
- `instrument` for the instrument of funding that you would like to apply for,

acronym
acrolong
- `acronym` for the proposal acronym, possibly accompanied by an `acrolong` that explains it. The acronym will also be used in the page headings.

start
months
- `start` for the start date of the proposed fragment of the project, and `months` for the length of the proposal in months. Both have to be specified for the `proposal` class to work.

since
fundsuntil
- If the proposal only concerns a part of a longer-running project, the `since` key allows to specify the date since when the overall project runs. Finally, the `fundsuntil` allows to specify a date until which the funds last.

discipline
- `discipline` for the academic discipline and `areas` for the research areas in that discipline.

PI
- `PI` to declare the principal investigator. For collaborative proposals we can use the `PI` key multiple times. The `proposal` package uses the `workaddress` package for representation of personal metadata, see [Koh16c] or the file `proposal.tex` for details.

site
- Many collaborative proposals are shared between two institutions, which we can declare with the `site` key. As this changes the interface this should not be used for single-institution proposals. We will describe the setup for a single-site proposal below and point out the differences. The example `proposal.tex` is a two-site proposal.

\pn
\pnlong   If the `acronym` and `acrolong` are given, then they automatically define the macros `\pn` and `\pnlong` which allow to use the project acronym (p̲roject n̲name) and its long version in the text. Note that these macros use `\xspace` internallly, so they do not have to be enclosed in curly braces.

There are two ways of organizing the distribution of personnel resources when developing a proposal. Either the coordinator takes a *top-down approach* where she assigns person months (PM) to the respective site, of she takes a *bottom-up approach*, where the sites "request" personnel resources by marking them up in the CVs of the researchers in the site descriptions. `proposal.cls`
topdownPM   supports both of these. Support for the first is configured via the `topdownPM` key and for the other

botupPM via the `botupPM` key. They add respective lines for planning in the WA/WP figure (see 2.6).

2.3 Proposal Appearance

EdN:1 The `proposal` environment takes a second set of keyval arguments that allow to fine-tune the appearance of the proposal document. [1]

compactht • If the `compactht` key is given (it does not need a value), then the header tables[2] are made
EdN:2 compact, i.e. the sites that do not have a contribution to the work package or work area do not get listed. This is useful for proposals with more than 8 partners.

emphbox The `proposal` package supplies the `emphbox` environment to create boxes of emphasized material we want to call attention to.

2.4 The proposal Environment and Title Page

EdN:3 [3]

2.5 Objectives

The work plan starts with a discussion of objectives, which may be referenced in the text later.
objective The `proposal` package provides the `objective` environment that allows to mark up individual objectives. It takes a keyval argument with the keys `id` for identification, `title` for the objective title, and `short` for a short title that can be used for referencing when the title is too long. The
\OBJref objectives can be referenced via `\OBJref{`$\langle id \rangle$`}` by their label and via `\OBJtref{`$\langle id \rangle$`}` by label
\OBJtref and (short if it was specified) title.

2.6 Work Areas and Work Packages

Grant proposals have another part that is often highly stylized; the work plan. This is usually structured into "work packages" — i.e. work items that address a cohesive aspect of the proposed work. These work packages are usually consecutively numbered, have a title, and an associated effort estimation. As work packages are the "atomic" planning units, they are usually heavily cross-referenced. A well-written proposal usually contains a table giving an overview over the work packages and their efforts and a Gantt chart showing the temporal distribution of the proposed work to allow the reviewers to get a clear picture of the feasibility of the research and development proposed. But this picture is also essential during the development of a proposal (which the `proposal` package aims to support), when the work packages (and their estimated efforts) usually change considerably. Therefore the `proposal` class standardizes markup for work packages and automatically computes the work package table (which can be inserted into the table via the
\wpfig `\wpfig` macro) and the Gantt Chart (see Section 2.12).
workplan To achieve the automation, work plan is marked up by the `workplan` environment, which sets up various internal counters and bookeeping macros. It contains texts and `workpackage` environments for the work packages.
workpackage The purpose of the `workpackage` environment is to mark up a fragment of text as a work package description and specify the metadata so that it can be used in the work package table and Gantt chart generation. The metadata is specified by the following keys:
id • The `id` key is used to specify a label for cross-referencing the work package or work area, it must be document-unique.
title • The `title` and `short` keys are used for the work package/group title. The short title is used
short in tables and should not be longer than 15 characters.
wphases • The `wphases` key is used according to Section 2.8
requires • The `requires` key can be used to mark, up dependencies between tasks. If `requires=\taskin{`$\langle rid \rangle$`}{`$\langle wp \rangle$`}`

---

[1]EdNote: move the RAM, wpsectionheadings,... options here.
[2]EdNote: describe them somewhere and reference here
[3]EdNote: add documentation

is given in a task with `id=`$\langle t \rangle$, then task $\langle rid \rangle$ in work package $\langle wp \rangle$ must be completed for task $\langle t \rangle$ to become possible. This key will draw an arrow into the gantt chart from the end of task $\langle rid \rangle$ to $\langle t \rangle$. Note that dependencies should always point forward in time. Furthermore, note that the fact that dependencies always go from the end of the source to the beginning of the target work phase is intentional, if this does not meet your needs, then you should probably break a work phase into pieces that can be addressed separately.

RM
RAM
- In single-site proposals, the `RM` (and `RAM` if the `RAM` option was given) keys are used to specify the estimated efforts to be expended on research and development in this work package. Both are specified in person months. `RM` is used for "researcher months" (wissenschaftlicher Mitarbeiter) and `RAM` for "research assistant months" (wissenschaftliche Hilfskraft).

*RM
*RAM
- In multi-site proposals, the `proposal` package generates the keys $\langle site \rangle$`RM` (and $\langle site \rangle$`RAM`) where $\langle site \rangle$ is any site label declared via the `site` key in the top-level `proposal` environment. This can be used to specify the person months that the site spends on this work package (the value for work areas is automatically computed (remember to run LATEX twice for this)).

lead
- In multi-site proposals the `lead` key specifies the work package or work area lead, the value of this feature should be the short name of the respective partner.

swsites
- For work packages with many prospers the `swsites` key can be given (no value needed) to turn the site names sideways to conserve (horizontal) space.

It is often useful to group the work packages in a proposal further (especially for larger, collab-
workarea orative proposals). This can be done via the `workarea` environment, which groups work packages. This environment takes the same keys as the `workpackage` environment, except for the efforts, which can be computed automatically from the work packages it groups.

As the author of the `proposal` class likes more structured proposals, using work areas is the default, but the `proposal` class can also be used with the `noworkareas` option for less structured (smaller) proposals.

2.7 Tasks

tasklist In the work packages we can list tasks that need to be undertaken with the `tasklist` environment.
task The individual tasks are marked up with the `task` environment. This takes a keyval argument with the keys `id` for identification, `title` for a title, and the workphase keys `wphases`, `start`, `end`, and `force` (see Section 2.8). For planning involvement we can specify the overall person months via the `PM` key, the task lead via `lead`, and the partners involved via the `partners` key. Finally task dependencies can be specified via the `requires` key.

\taskref   Tasks can be referenced by the `\taskref` macro that takes two arguments: the work package identifier and the task identifier. As for work packages and work areas, there is a long reference
\tasktref variant with work package title: `\tasktref`. Finally, `\localtaskref` references a task in the local
\localtaskref work package by the identifier in its argument.

2.8 Work Phase Metadata

wphases The `task` and `workpackage` allow the `wphases` key to specify the a list of work phases. The value of this key is comma-separated list of work phase specifications of the form $\langle start \rangle$-$\langle end \rangle$ or $\langle start \rangle$-$\langle end \rangle$!$\langle force \rangle$, where $\langle start \rangle$ and $\langle end \rangle$ delimit the run time of the work phase and the optional !$\langle force \rangle$ specifies the work force, i.e. the intensity of work as a number between 0 and 1. If no force is given, the default is 1. The main reason for specifying this metadata for tasks is to generate a Gantt chart (see Section 2.12).

2.9 Milestones and Deliverables

Many proposal formats foresee that project progress will be tracked in the form of *milestones* – points in the project, where a predefined state of affairs is reached – and *deliverables* – tangible project outcomes that have to be delivered. Correspondingly, milestones and deliverables have to be specified in the proposal and accounted for in the project reports. To facilitate this the

proposal class and its instances provide a simple infrastructure for dealing with milestones and deliverables.

milestones   Milestones are usually given in a special table[1], which we markup up with the milestones environment that takes care of initialization and numbering issues. This contains a list of milestone
\milestone   descriptions via the \milestone macro which is invoked as \milestone[⟨keys⟩]{⟨title⟩}{⟨desc⟩}, where ⟨keys⟩ supports the keys id for identification month for specifying the milestone date (in months of the project duration). Milestones are numbered with labels whose shape can
\milestone@label be customized by redefining \milestone@label and referenced by the \mileref{⟨id⟩} and
\mileref   \miletref{⟨id⟩} for a reference with milestone title. \pdatacount{all}{miles} gives the num-
\miletref   ber of milestones.

Deliverables are usually defined as part of the work package descriptions (see Section 2.6) and listed in an overview table in a separate of the proposal. As for the milestones, we use
wpdelivs   an environment wpdelivs that contains the deliverable descriptions. These are marked up via
wpdeliv   the environment which takes an optional keyval argument for the deliverable metadata a regular argument for the title and contains the description of the deliverable as the body. For the metadata we have the keys id for the deliverable identifier, due for the target date (a number that denotes the project month), nature and dissem for specifying the deliverable nature and dissemination status (usually as short strings prescribed by the proposal template), and miles for the milestone this deliverable is targeted for (specified by the milestone identifier). For repeating deliverables (e.g. project reports), both due and miles can contain comma-separated lists. Deliverables are numbered by labels whose shape can be customized by number, where the shape of the label can
\deliv@label be specified by redefining \deliv@label and referenced by \delivref{⟨wp⟩}{⟨id⟩} where ⟨wp⟩
\delivref   is the work package identifier and ⟨id⟩ that if the deliverable and \delivtref{⟨wp⟩}{⟨id⟩} for
\delivtref   a reference with title. \localdelivref can be used to reference deliverables in the same work
\localdelivref package. \pdatacount{⟨wp⟩}{delivs} gives the number of milestones of the work package ⟨wp⟩
\pdatacount{all}{delivs} that of all deliverables (aggregating over all work packages).

Some proposal templates ask for an overview table of the deliverables which aggregates the deliverables of the respective work packages and areas ordered by due date. This can be generated
\inputdelivs with the \inputdelivs macro. This works index generation in LaTeX. The wpdeliv environment writes the deliverable data to a file ⟨main⟩.delivs, which can be processed externally (usually just sorting with sort in Unix is sufficient) into ⟨main⟩.deliverables, which is then input via the \inputdelivs macro.

In some proposals, also work areas can have deliverables, then the above hold analogously for
wadelivs   wpdelivs and wadeliv environments.
wadeliv   Note that handling deliverables adds considerable overhead to proposal formatting and adds auxiliary files, so they are only activated if the deliverables option is given (see Section 2.1).

2.10 Project Data, Referencing, and Hyperlinking

The proposal package extends the hyperlinking provided by the hyperref package it includes to work packages, work areas, ... Whenever these are defined using the proposal infrastructure, the class saves the relevant information in the auxiliary file ⟨proposal⟩.aux. This information can
\pdataref   be referenced via the \pdataref macro, which takes three arguments.

In a reference \pdataref{⟨type⟩}{⟨id⟩}{⟨aspect⟩} the first argument ⟨type⟩ specifies the type of the object (currently one of wp, wa, and partner) to be referenced, ⟨id⟩ specifies the identifier of the referenced object (it matches the identifier given in the id key of the object), and ⟨aspect⟩ specifies the aspect of the saved information that is referenced.
\pdatarefFB   \pdatarefFB{⟨type⟩}{⟨id⟩}{⟨a1⟩}{⟨a2⟩} tries first \pdataref{⟨type⟩}{⟨id⟩}{⟨a1⟩} and if that is not given \pdataref{⟨type⟩}{⟨id⟩}{⟨a2⟩}.

For a work package ⟨aspect⟩ can be number, (the work package number), label (the label WPn where n is the work package number for referencing), title (the work package title), lead

---

[1]this is the default provided by the base proposal class, it can be specialized for proposal class instances by redefining the @milestones environment and correspondingly the milestone macro.

the work package leader, `short` (a short version of the WP title for tables). For work areas we have the same aspects with analogous meanings. In all cases, the referenced information carries a hyperlink to the referenced object.

\pdataRef The `\pdataRef` and `\pdataRefFB` macros are variant of `\pdataref` and `\pdataRef` that also
\pdataRefFB carry a hyperlink (if the `hyperref` package is loaded).

\pdatacount The `\pdatacount` macro gives access to the numbers of certain aspects. For instance, the number of work packages in the proposal can be cited by `\pdatacount{all}{wp}`, similarly for work areas (if they are enabled), and finally, `\pdatacount{`⟨*wa*⟩`}{wp}` gives the number of work packages for a work area ⟨*wa*⟩. This is very useful for talking about work plans in a general way. Other objects that can be counted are deliverables (`\pdatacount{all}{deliverables}`) and milestones (`\pdatacount{all}{milestones}`).

Note that since the referencable information is written into the project data file ⟨*proposal*⟩`.pdata` file, it is available for forward references. However, it will only become available when the project data file is read, so the proposal has to be formatted twice for references to be correct.

Finally, the `proposal` package supplies specialized reference macros for work packages and
\WPref areas. The `\WPref` macro takes a work package identifier as an argument and makes a reference:
\WPtref `\WPref{`⟨*id*⟩`}` abbreviates `\pdataRef{wp}{`⟨*id*⟩`}{label}`. The `\WPtref` macro is similar, but also prints out the (short) title: `\WPref{`⟨*id*⟩`}` abbreviates `\pdataRef{wp}{`⟨*id*⟩`}{label}: \pdataRef{wp}{`⟨*id*⟩`}{title}`.
\WAref Unless the `noworkareas` macro is set, we also have the variants `\WAref` and `\WAtref` for work
\WAtref areas.

2.11 The Work Package Table

One of the most useful features of the `proposal` class is that we can generate an overview table for
\wpfig the distribution of workloads in the project fully automatically. All it takes is the `\wpfig` macro. We invoke this as `\wpfig[`⟨*opt*⟩`]`, where ⟨*opt*⟩ contains the following keywords:

pages makes a column with page numbers of the respective work package/area description.

type makes a column with work package/area types

start, **end**, and **length** makes a columns with work package/area start/end months and length (in months).

if **caption** is given then the table contains an explicatory caption.

label allows to specify a label other than the default `fig:wplist`.

For instance `\wpfig[pages,start,length,caption=Overview of Work Packages]` gives a table with columns for page references, duration information, and a special caption.
\wpfigstyle The general appearance of the table `\wpfigstyle` macro takes a token sequence to specialize the global appearance (mostly used for text sizes and color) of the work package table. Cell styling can be tweaked by redefining special internal macros; see section **??**.

2.12 Gantt Charts

Gantt charts are used in proposals to show the distribution of activities in work packages over time.
gantt A gantt chart is represented by the `gantt` environment that takes a on optional keyval argument.
xscale The keys `xscale` and `yscale` are used to specify a scale factors for the chart so that it fits on the
yscale page. The `step` key allows to specify the steps (in months) of the vertical auxiliary lines. Finally,
step the `draft` key specifies that plausibility checks (that can be expensive to run) are carried out.
draft Note that the value does not have to be given, so `\begin{gantt}{draft,yscale=.5,step=3}` is a perfectly good invocation.
\ganttchart Usually, the `gantt` environment is not used however, since it is part of the macro that takes the same keys. This generates a whole Gantt chart automatically from the work phase specifications in the work packages. As above we have to run LaTeX two times for the work phases to show up.

2.13 Coherence

Many proposals require ways to show coherence between the partners. The `proposal` class of-
`\coherencematrix` fers the macro `\coherencematrix` for this which generates a matrix of symbols specifying joint publications, project organization, software/resource development, and supervision of students by the project partners that have been declared by the `\jointpub`, `\jointproj`, `\jointorga`
`\jointpub`
`\jointsub`   `\jointsoft`, and `\jointsup` macros before. These macros all take a comma-separated list of site
`\jointorga` identifiers as an argument. Use for instance `\jointproj{a,b,c}` to specify that the sites with
`\jointsoft` the identifiers `a`, `b` and `c` have a joint project. `\coherencetable` is a variant which packages the
`\jointsup`   coherence table in a table figure with label `tab:collaboration`.
`\coherencetable`   The symbols used an be configured by redefining `\jpub`, `\jproj`, and `\jorga`, `\jsoft`, and
`\jpub`   `\jsup`.
`\jproj`
`\jorga` ## 2.14 Localization
`\jsoft`
`\jsup`   The `proposal` class offers some basic support for localization. This is still partial though, and I am not sure that this is the best way of setting things up. What I do is to define macros for all generated texts that can be redefined in the proposal classes that build in `proposal`. For instance the `dfgproposal` class [Koh16b] provides an option `german` for german-language proposals and project reports that triggers a redefinition of all of these macros at read time.

3 Limitations and Enhancements

The `proposal` is relatively early in its development, and many enhancements are conceivable. We will list them here.

1. macros cannot be used in work package and work area titles. They really mess up our `\wpfig` automation. The problem is that they are evaluated too early, and our trick with making them undefined while collecting the parts of the table-rows only works if we know which macros we may expect. We might specify all "allowable" macros in an optional key `protectmacro`, which is defined via

   ```
   \define@key{wpfig}{protectmacro}{\epandafter\let\csname #1\endcsname=\relax}
   ```

   But I am not sure that this will work.

2. It would be great, if in the Gantt Charts, we could include some plausibility checks (for draft = not `submit` mode). I can see two at the moment:

   - calculating the effort (i.e. the weight of the black area) and visualizing it. Then we could check whether that is larger than the effort declared for the work package.
   - calculating (and visualizing) the monthly effort. That should be kind of even (or it has to be explained in the positions requested).

3. we currently do not have a way to relate `PI`s to `site`s, but we do not really need to.

If you have other enhancements to propose or feel you can alleviate some limitation, please feel free to contact the author.

Acknowledgements

4 The Implementation

In this section we describe the implementation of the functionality of the `proposal` package.

4.1 Package Options and Format Initialization

We first set up the options for the package.

```
1 ⟨∗cls | reporting⟩
2 \newif\if@wpsubsection\@wpsubsectionfalse
3 \newif\ifsubmit\submitfalse
4 \newif\ifgrantagreement\grantagreementfalse
5 \newif\ifpublic\publicfalse
6 \newif\ifkeys\keysfalse
7 \newif\ifdelivs\delivsfalse
8 \newif\ifwork@areas\work@areastrue
9 \newif\if@RAM\@RAMfalse
10 \newif\if@svninfo\@svninfofalse
11 \newif\if@gitinfo\@gitinfofalse
12 \def\proposal@class{article}
13 \DeclareOption{wpsubsection}{\@wpsubsectiontrue}
14 \DeclareOption{submit}{\submittrue}
15 \DeclareOption{grantagreement}{\grantagreementtrue}
16 \DeclareOption{gitinfo}{\@gitinfotrue}
17 \DeclareOption{svninfo}{\@svninfotrue}
18 \DeclareOption{public}{\publictrue}
19 \DeclareOption{noworkareas}{\work@areasfalse\PassOptionsToClass{\CurrentOption}{pdata}}
20 \DeclareOption{RAM}{\@RAMtrue}
21 \DeclareOption{report}{\def\proposal@class{report}}
22 \DeclareOption{keys}{\keystrue}
23 \DeclareOption{deliverables}{\delivstrue}
24 \DeclareOption*{\PassOptionsToClass{\CurrentOption}{article}}
25 \ProcessOptions
26 \LoadClass[a4paper,twoside]{\proposal@class}
27 \RequirePackage{proposal}
28 ⟨/cls | reporting⟩
```

For `proposal.sty` we load the packages we make use of

```
29 ⟨∗sty⟩
30 \RequirePackage{amssymb}
31 \RequirePackage{wasysym}
32 \RequirePackage{url}
33 \RequirePackage{graphicx}
34 \RequirePackage{colortbl}
35 \RequirePackage{xcolor}
36 \RequirePackage{rotating}
37 \RequirePackage{fancyhdr}
38 \RequirePackage{array}
39 \RequirePackage{xspace}
40 \RequirePackage{comment}
41 \AtBeginDocument{\ifpublic\excludecomment{private}\fi}
42 \RequirePackage{tikz}
43 \RequirePackage{paralist}
44 \RequirePackage[a4paper,margin=18mm]{geometry}
45 \RequirePackage{boxedminipage}
46 % so that ednotes in wps do not run out of symbols
47 \renewcommand{\thempfootnote}{\roman{mpfootnote}}
48 \renewcommand{\familydefault}{\sfdefault}
49 \RequirePackage[scaled=.90]{helvet}
```

```
50 \RequirePackage{textcomp}
51 \RequirePackage[hyperref=auto,style=alphabetic,defernumbers=true,backend=bibtex,firstinits=true,maxbibnames=6
52 \RequirePackage{csquotes}
53 \RequirePackage{mdframed}
```

in submit mode, we make the links a bit darker, so they print better.

```
54 \RequirePackage{pdata}
55 \definecolor{darkblue}{rgb}{0,0,.7}
56 \ifsubmit\def\prop@link@color{darkblue}\else\def\prop@link@color{blue}\fi
57 \RequirePackage[bookmarks=true,linkcolor=\prop@link@color,
58  citecolor=\prop@link@color,urlcolor=\prop@link@color,colorlinks=true,
59  breaklinks=true, bookmarksopen=true]{hyperref}
```

the ed package [Koh16a] is very useful for collaborative writing and passing messages between collaborators or simply reminding yourself of editing tasks, so we preload it in the class. However, we only want to show the information in draft mode. Furthermore, we adapt the options for the svninfo and gitinfo2 packages.

```
60 \ifsubmit
61 \RequirePackage[hide]{ed}
62 \if@svninfo\RequirePackage[final,today]{svninfo}\fi
63 \else
64 \RequirePackage[show]{ed}
65 \if@svninfo\RequirePackage[eso-foot,today]{svninfo}\fi
66 \if@gitinfo\RequirePackage[mark]{gitinfo2}\fi
67 \fi
68 \renewcommand\ednoteshape{\sl\footnotesize}
```

private We configure the comment package, so that it provides the private environment depending on the status of the public option.

```
69 \ifpublic\excludecomment{private}\else\includecomment{private}\fi
```

And we set up the appearance of the proposal. We want numbered subsubsections.

```
70 \setcounter{secnumdepth}{3}
```

We specify the page headings.

```
71 \newif\ifofpage\ofpagefalse
72 \ifgrantagreement
73 \fancyhead{}
74 \renewcommand{\headrulewidth}{0pt}
75 \renewcommand{\footrulewidth}{0.4pt}
76 \else
77 \fancyhead[RE,LO]{\prop@gen@acronym}
78 \fancyhfoffset{0pt}
79 \fi
80 \fancyfoot[C]{}
81 \newcommand\prop@of@pages[2]{page~#1\ifofpage~of~#2\fi}
82 \ifgrantagreement
83 \fancyfoot[L]{\prop@gen@proposalnumber\quad \prop@gen@acronym\quad --\quad Part B}
84 \fancyfoot[R]{\thepage}
85 \else
86 \fancyhead[LE,RO]{\prop@of@pages\thepage{\pdataref@num{prop}{page}{last}}}
87 \fi
88 \pagestyle{fancyplain}
89 ⟨/sty⟩
```

4.2   Proposal Metadata

pdata Most of the metadata functionality is encapsulated into the pdata package, which is shared by

the proposal and report classes. `pdata.sty` first loads the `workaddress` package from sTeX and supplies the Euro symbol.

```
 90 ⟨∗pdata⟩
 91 \RequirePackage{workaddress}[2016/07/06]
 92 \RequirePackage{eurosym}
```

We define the keys for metadata declarations in the `proposal` environment, they park their argument in an internal macro for use in the title page. The `site` key is the most complicated, so we take care of it first: We need a switch `\if@sites` that is set to true when the `site` key is used. Furthermore `site=⟨site⟩` makes new keys ⟨site⟩`RM` and ⟨site⟩`RAM` (if the `RAM` option was set) for the `workpackage` environment and records the sites in the `\prop@gen@sites` token register.

```
 93 \newif\if@sites\@sitesfalse\let\prop@gen@sites=\relax%
 94 \newcounter{@site}%
 95 \define@key{prop@gen}{site}{\@sitestrue\@dmp{site=#1}%
 96 \stepcounter{@site}\pdata@def{site}{#1}{number}{\the@site}%
 97 \@ifundefined{prop@gen@sites}{\xdef\prop@gen@sites{#1}}{\xdef\prop@gen@sites{\prop@gen@sites,#1}}%
 98 \define@key{prop@gen}{#1RM}{\pdata@def{site}{#1}{intendedRM}{##1}}%
 99 \if@RAM\define@key{prop@gen}{#1RAM}{\pdata@def{site}{#1}{intendedRAM}{##1}}\fi
100 \define@key{workpackage}{#1RM}{\pdata@def\wp@id{#1}{RM}{##1}}%
101 \if@RAM\define@key{workpackage}{#1RAM}{\pdata@def\wp@id{#1}{RAM}{##1}}\fi
102 \define@key{prop@gen}{#1employed}{{\let\tabularnewline\relax\let\hline\relax\let\wa@ref\relax%
103 \@ifundefined{prop@gen@employed@lines}%
104 {\xdef\prop@gen@employed@lines{\wa@ref3{institution}{#1}{shortname} & ##1\tabularnewline\hline}}%
105 {\xdef\prop@gen@employed@lines{\prop@gen@employed@lines \wa@ref3{institution}{#1}{shortname} & ##1\tabularnew
```

If there are no sites, then we have to define keys `RM` and `RAM` that store the intended research (assistant months). Unfortunately, we cannot just include this in the `\if@sites` conditional here, since that is only set at runtime.

```
106 \define@key{prop@gen}{RM}{\@dmp{RM=#1}\if@sites%
107 \PackageWarning{Do not use the RM key in the presence of sites}\else%
108 \pdata@def{all}{intended}{RM}{#1}\fi}
109 \define@key{prop@gen}{RAM}{\@dmp{RAM=#1}\if@sites%
110 \PackageWarning{Do not use the RAM key in the presence of sites}\else%
111 \pdata@def{all}{intended}{RAM}{#1}\fi}
```

similarly, the PI keys are registered in `\prop@gen@PIs`.

```
112 \define@key{prop@gen}{PI}{\@dmp{PI=#1}%
113 \@ifundefined{prop@gen@PIs}{\xdef\prop@gen@PIs{#1}}{\xdef\prop@gen@PIs{\prop@gen@PIs,#1}}}
```

and the `pubspage` keys in `\prop@gen@pubspages`.

```
114 \define@key{prop@gen}{pubspage}{\@ifundefined{prop@gen@pubspages}%
115 {\xdef\prop@gen@pubspages{#1}}{\xdef\prop@gen@pubspages{\prop@gen@pubspages,#1}}}
```

the `importfrom` key reads the proposal data from its argument.

```
116 \define@key{prop@gen}{importfrom}{\message{importing proposal data from #1.pdata}\readpdata{#1}}
```

The rest of the keys just store their value.

```
117 \define@key{prop@gen}{instrument}{\def\prop@gen@instrument{#1}%
118 \pdata@def{prop}{gen}{instrument}{#1}\@dmp{inst=#1}}
119 \define@key{prop@gen}{title}{\def\prop@gen@title{#1}%
120 \pdata@def{prop}{gen}{title}{#1}}
121 \define@key{prop@gen}{acronym}{\gdef\prop@gen@acronym{#1}%
122 \pdata@def{prop}{gen}{acronym}{#1}\@dmp{acro=#1}}
123 \define@key{prop@gen}{acrolong}{\def\prop@gen@acrolong{#1}%
124 \pdata@def{prop}{gen}{acrolong}{#1}}
125 \define@key{prop@gen}{proposalnumber}{\def\prop@gen@proposalnumber{#1}%
126 \pdata@def{prop}{gen}{proposalnumber}{#1}}
127 \define@key{prop@gen}{discipline}{\def\prop@gen@discipline{#1}%
128 \pdata@def{prop}{gen}{discipline}{#1}}
129 \define@key{prop@gen}{areas}{\def\prop@gen@areas{#1}%
```

```
130 \pdata@def{prop}{gen}{areas}{#1}}
131 \define@key{prop@gen}{start}{\def\prop@gen@start{#1}%
132 \pdata@def{prop}{gen}{start}{#1}}
133 \define@key{prop@gen}{months}{\def\prop@gen@months{#1}%
134 \pdata@def{prop}{gen}{months}{#1}}
135 \define@key{prop@gen}{since}{\def\prop@gen@since{#1}%
136 \pdata@def{prop}{gen}{since}{#1}}
137 \define@key{prop@gen}{totalduration}{\def\prop@gen@totalduration{#1}%
138 \pdata@def{prop}{gen}{totalduration}{#1}}
139 \define@key{prop@gen}{fundsuntil}{\def\prop@gen@fundsuntil{#1}%
140 \pdata@def{prop}{gen}{fundsuntil}{#1}}
141 \define@key{prop@gen}{topdownPM}[true]{\def\prop@gen@topdownPM{#1}}
142 \define@key{prop@gen}{botupPM}[true]{\def\prop@gen@botupPM{#1}}
143 \define@key{prop@gen}{keywords}{\def\prop@gen@keywords{#1}}
```

and the default values, these will be used, if the author does not specify something better.

```
144 \newcommand\prop@gen@acro@default{ACRONYM}
145 \def\prop@gen@acro{\prop@gen@acro@default}
146 \newcommand\prop@gen@months@default{???months???}
147 \def\prop@gen@months{\prop@gen@months@default}
148 \newcommand\prop@gen@title@default{???Proposal Title???}
149 \def\prop@gen@title{\prop@gen@title@default}
150 \newcommand\prop@gen@instrument@default{??? Instrument ???}
151 \def\prop@gen@instrument{\prop@gen@instrument@default}
```

\prop@tl   An auxiliary macro that is handy for making tables of WorkAddress data.

```
152 \newcommand\prop@tl[2]{\xdef\tab@line{}
153 \@for\tl@ext:={#1}\do{\xdef\tab@line{\tab@line&#2}}
154 \tab@line}
```

4.3   Proposal Appearance

We define the keys for the proposal appearance

```
155 \def\prop@gen@compactht{false}
156 \define@key{prop@gen}{compactht}[true]{\def\prop@gen@compactht{#1}}
157 ⟨/pdata⟩
```

emphbox

```
158 ⟨*sty⟩
159 \newmdenv[settings=\large]{emphbox}
```

4.4   The proposal Environment and Title Page

prop@proposal   This internal environment is called in the proposal environment from the proposal class. The implementation here is only a stub to be substituted in a specialized class.

```
160 \newenvironment{prop@proposal}
161 {\thispagestyle{empty}%
162 \begin{center}
163   {\LARGE \prop@gen@instrument}\\[.2cm]
164   {\LARGE\textbf{\prop@gen@title}}\\[.3cm]
165   {\LARGE Acronym: {\prop@gen@acronym}}\\[.2cm]
166   {\large\today}\\[1em]
167   \begin{tabular}{c*{\the@PIs}{c}}
168     \prop@tl\prop@gen@PIs{\wa@ref3{person}\tl@ext{name}}\\
169     \prop@tl\prop@gen@PIs{\wa@ref3{institution}{\wa@ref3{person}\tl@ext{affiliation}}{name}}
170   \end{tabular}\\[2cm]
171 \end{center}
172 \setcounter{tocdepth}{2}\tableofcontents\newpage\setcounter{page}{1}}
```

Now we come to the end of the environment:

```
173 {\section{List of Attachments}
174 \begin{itemize}
175 \@for\@I:=\prop@gen@PIs\do{%
176 \item Curriculum Vitae and list of publications for
177   \wa@ref3{person}\@I{personaltitle} \wa@ref3{person}\@I{name}}
178 \end{itemize}\newpage
179 \printbibliography[heading=warnpubs]}
```

proposal The `proposal` environment reads the metadata keys defined above, and if there were no `site` keys, then it defines keys `RM` and `RAM` (unless the `noRAM` package option was given) for the `workpackage` environment. Also it reads the project data file and opens up the project data file `\pdata@out`, which it also closes at the end.

The environment calls an internal version of the environment `prop@proposal` that can be customized by the specializing classes.

```
180 \newenvironment{proposal}[1][]{\readpdata\jobname
181 \ofpagetrue\setkeys{prop@gen}{#1}
182 \pdata@open\jobname
183 \if@sites\else
184 \define@key{workpackage}{RM}{\pdata@def{wp}\wp@id{RM}{##1}\@dmp{RM=##1}}
185 \if@RAM\define@key{workpackage}{RAM}{\pdata@def{wp}\wp@id{RAM}{##1}\@dmp{RAM=##1}}\fi
186 \fi
187 \newcounter{@PIs}
188 \@ifundefined{prop@gen@PIs}{}{\@for\@I:=\prop@gen@PIs\do{\stepcounter{@PIs}}}
189 \newcounter{@sites}
190 \@ifundefined{prop@gen@sites}{}{\@for\@I:=\prop@gen@sites\do{\stepcounter{@sites}}}
191 \setcounter{page}{0}
192 \begin{prop@proposal}}
```

Now we come to the end of the environment, we take care of the last page and print the references.

```
193 {\end{prop@proposal}
194 \pdata@def{prop}{page}{last}{\thepage}\ofpagefalse
195 \pdata@close}
196 ⟨/sty⟩
```

The `report` environment is similar, but somewhat simpler

report

```
197 ⟨*reporting⟩
198 \newif\if@report\@reportfalse
199 \newenvironment{report}[1][]%
200 {\@reporttrue\readpdata\jobname%
201 \ofpagetrue\setkeys{prop@gen}{#1}%
202 \pdata@open\jobname%
203 \@ifundefined{prop@gen@PIs}{}{\newcounter{@PIs}\@for\@I:=\prop@gen@PIs\do{\stepcounter{@PIs}}}%
204 \@ifundefined{prop@gen@sites}{}{\newcounter{@sites}\@for\@I:=\prop@gen@sites\do{\stepcounter{@sites}}}%
205 \setcounter{page}{0}%
206 \begin{prop@report}}
207 {\end{prop@report}%
208 \pdata@def{prop}{page}{last}{\thepage}\ofpagefalse\newpage
209 \printbibliography[heading=warnpubs]
210 \pdata@close}
```

prop@report

```
211 \newenvironment{prop@report}
212 {\begin{center}
213   {\LARGE Final Project Report}\\[.2cm]
214   {\LARGE\textbf{\prop@gen@title}}\\[.3cm]
```

```
215    {\LARGE Acronym: {\prop@gen@acronym}}\\[.2cm]
216    {\large\today}\\[1em]
217    \begin{tabular}{c*{\the@PIs}{c}}
218      \prop@tl\prop@gen@PIs{\wa@ref3{person}\tl@ext{name}}\\
219      \prop@tl\prop@gen@PIs{\wa@ref3{institution}{\wa@ref3{person}\tl@ext{affiliation}}{name}}
220   \end{tabular}\\[2cm]
221  \end{center}
222  \setcounter{tocdepth}{2}\tableofcontents\newpage\setcounter{page}{1}}
223  {}
224  ⟨/reporting⟩
```

\site*

```
225  ⟨*sty⟩
226  \newcommand\site[1]{\hyperlink{site@#1@target}{\wa@ref3{institution}{#1}{acronym}}}
227  \newcommand\sitename[1]{\hyperlink{site@#1@target}{\wa@ref3{institution}{#1}{name}}}
```

4.5   Objectives

We first define a presentation macro for objectives

\objective@label

```
228  \newcommand\objective@label[1]{O#1}
```

We define the keys for the objectives environment

```
229  \define@key{obj}{id}{\def\obj@id{#1}\@dmp{id=#1}}
230  \define@key{obj}{title}{\def\obj@title{#1}\@dmp{title=#1}}
231  \define@key{obj}{short}{\def\obj@short{#1}\@dmp{short=#1}}
```

And a counter for numbering objectives

```
232  \newcounter{objective}
```

objective

```
233  \newenvironment{objective}[1][]
234  {\let\obj@id\relax\let\obj@title\relax\let\obj@short\relax%
235  \setkeys{obj}{#1}\stepcounter{objective}%
236  \goodbreak\smallskip\par\noindent%
237  \textbf{\objective@label{\arabic{objective}}:%
238  ~\pdata@target{obj}{\obj@id}{\pdataref{obj}{\obj@id}{title}}\ignorespaces}%
239  \pdata@def{obj}\obj@id{label}{\objective@label\theobjective}%
240  \@ifundefined{obj@title}{}{\pdata@def{obj}\obj@id{title}\obj@title}%
241  \@ifundefined{obj@short}{}{\pdata@def{obj}\obj@id{short}\obj@short}}
242  {}
```

\OBJref

```
243  \newcommand\OBJref[1]{\pdataRef{obj}{#1}{label}}
244  \newcommand\OBJtref[1]{\OBJref{#1}: \pdataRefFB{obj}{#1}{short}{title}}
```

4.6   Work Areas and Work Packages

We first define keys for work areas (if we are in larger project).

```
245  \ifwork@areas
246  \define@key{workarea}{id}{\def\wa@id{#1}\@dmp{id=#1}}
247  \define@key{workarea}{title}{\pdata@def{wa}\wa@id{title}{#1}}
248  \define@key{workarea}{short}{\pdata@def{wa}\wa@id{short}{#1}}
249  \define@key{workarea}{lead}{\pdata@def{wa}\wa@id{lead}{#1}}
250  \fi
```

work packages have similar ones.

```
251 \define@key{workpackage}{id}{\def\wp@id{#1}\@dmp{id=#1}}
252 \define@key{workpackage}{title}{\pdata@def{wp}\wp@id{title}{#1}}
253 \define@key{workpackage}{short}{\pdata@def{wp}\wp@id{short}{#1}}
254 \define@key{workpackage}{lead}{\pdata@def{wp}\wp@id{lead}{#1}\def\wp@lead{#1}\@dmp{lead=#1}}
255 \define@key{workpackage}{type}{\def\wp@type{#1}\pdata@def{wp}\wp@id{type}{#1}}
256 \define@key{workpackage}{wphases}{\def\wp@wphases{#1}\pdata@def{wp}\wp@id{wphases}{#1}}
257 \define@key{workpackage}{swsites}[true]{\def\wp@swsites{#1}}
```

We define the constructors for the work package and work area labels and titles.

```
258 \newcommand\wp@mk@title[1]{Work Package {#1}}
259 \newcommand\wp@label[1]{WP{#1}}
260 \ifwork@areas
261 \newcommand\wa@label[1]{WA{#1}}
262 \newcommand\wa@mk@title[1]{Work Area {#1}}
263 \fi
```

The `wa` and `wp` counters are for the work packages and work areas, the counter `deliv` for deliverables.

```
264 \ifwork@areas\newcounter{wa}\newcounter{wp}[wa]\else\newcounter{wp}\fi
265 \ifdelivs\newcounter{deliv}[wp]\fi
266 \newcounter{allwp}
```

\update@*

EdN:4

update the list `\@wps` of the work packages in the local group and the list `\@was` work areas for the staff efforts table: if `\@wps` is undefined, then initialize the comma-separated list, otherwise extend it.[4]

```
267 \newcommand\update@wps[1]{\@ifundefined{@wps}{\xdef\@wps{#1}}{\xdef\@wps{\@wps,#1}}}
268 \newcommand\update@tasks[1]{\@ifundefined{@tasks}{\xdef\@tasks{#1}}{\xdef\@tasks{\@tasks,#1}}}
269 \newcommand\update@deps[1]{\@ifundefined{task@deps}{\xdef\task@deps{#1}}{\xdef\task@deps{\task@deps,#1}}}
270 \ifwork@areas\def\update@was#1{\@ifundefined{@was}{\xdef\@was{#1}}{\xdef\@was{\@was,#1}}}\fi
```

\decode@wphase

\decode@wphase decodes a string of the form $\langle start\rangle$-$\langle end\rangle$!$\langle force\rangle$ and defines the macros \wphase@start, \wphase@end, and \wphase@force with the three parts and also computes \wphase@len. The intermediate parsing macro \decode@p@start parses out the start (a number), and passes on to \decode@p@end, which parses out the end (another number) and the force string, which is either empty (if the !$\langle force\rangle$ part is omitted) or of the form !$\langle force\rangle$. In the first case the default value 1 is returned for \decode@force in the second $\langle force\rangle$.

```
271 \newcommand\decode@wphase[1]{\expandafter\decode@p@start#1@%
272 \local@count\wphase@end\advance\local@count by -\wphase@start%
273 \def\wphase@len{\the\local@count}}
274 \def\decode@p@start#1-#2@{\def\wphase@start{#1}\decode@p@end#2!@}
275 \def\decode@p@end#1!#2@{\def\wphase@end{#1}\def\@test{#2}%
276 \ifx\@test\@empty\def\wphase@force{1}\else\decode@p@force#2\fi}
277 \def\decode@p@force#1!{\def\wphase@force{#1}}
```

\startend@wphases

We first iteratively decode the work phases, so that the last definition of \wphase@end remains, then we parse out the start of the first workphase to define \wphase@start

```
278 \def\wphases@start#1-#2@{\def\wphase@start{#1}}
279 \newcommand\startend@wphases[1]{\def\@test{#1}
280 \ifx\@test\@empty\def\wphase@start{0}\def\wphase@end{0}\else%
281 \@for\@I:=#1\do{\expandafter\decode@p@start\@I @}
282 \expandafter\wphases@start#1@\fi}
```

with these it is now relatively simple to define the interface macros.

---

[4]EdNote: with the current architecture, we cannot have work areas that do not contain work packages, this leads to the error that wps is undefined in endworkplan

work@package   The `workpackage` environment collects the keywords, steps the counters, writes the metadata to the aux file, updates the work packages in the local group, generates the work package number `\wp@num`.

```
283 \newcounter{wp@RM}
284 \if@RAM\newcounter{wp@RAM}\fi
285 \newenvironment{work@package}[1][]%
286 {\def\wp@wphases{0-0}% default values
287 \def\wp@swsites{false}
288 \setkeys{workpackage}{#1}\stepcounter{wp}\stepcounter{allwp}%
289 \startend@wphases\wp@wphases%
290 \pdata@def{wp}\wp@id{start}\wphase@start\pdata@def{wp}\wp@id{end}\wphase@end%
291 \@ifundefined{wp@type}{}{\pdata@def{wp}\wp@id{type}\wp@type}%
292 \let\@tasks=\relax%
293 \edef\wp@num{\ifwork@areas\thewa.\fi\thewp}%
294 \pdata@def{wp}\wp@id{label}{\wp@label\wp@num}%
295 \pdata@def{wp}\wp@id{number}{\thewp}%
296 \pdata@def{wp}\wp@id{page}{\thepage}%
297 \update@wps\wp@id%
298 \edef\wp@num{\ifwork@areas\thewa.\fi\thewp}%
299 \pdata@def{wp}{\wp@id}{num}{\thewp}%
```

If we have sites, we have to compute the total RM and RAM for this WP.

```
300 \if@sites%
301 \setcounter{wp@RM}{0}\if@RAM\setcounter{wp@RAM}{0}\fi%
302 \@for\@site:=\prop@gen@sites\do{%
303 \edef\@RM{\pdataref@num\wp@id\@site{RM}}\addtocounter{wp@RM}{\@RM}%
304 \if@RAM\edef\@RAM{\pdataref@num\wp@id\@site{RAM}}\addtocounter{wp@RAM}{\@RAM}\fi}
305 \pdata@def{wp}\wp@id{RM}{\thewp@RM}%
306 \if@RAM\pdata@def{wp}\wp@id{RAM}{\thewp@RAM}\fi%
307 \fi}% if@sites
308 {\@ifundefined{@tasks}{}{\pdata@def{\wp@id}{task}{ids}\@tasks}}
```

workpackage   With this, it becomes simple to define a work package environment. We consider two cases, if we have sites, then we make a header table. If not, we can make things much simpler: we just generate a subsection

```
309 \newenvironment{workpackage}[1][]%
310 {\begin{work@package}[#1]%
311 \ifgrantagreement\else
312 %\if@wpsubsection\subsubsection*{{\wp@mk@title\thewp}: \pdataref{wp}\wp@id{title}}\fi
313 \if@sites\goodbreak\medskip\wpheadertable%
314 \else\subsubsection*{{\wptitle} (\wprm)}\fi%
315 \addcontentsline{toc}{paragraph}{{\wp@mk@title\thewp}: \pdataref{wp}\wp@id{title}}%
316 \noindent\ignorespaces%
317 \fi}
318 {\end{work@package}}
```

EdN:5 \wptitle   [5]

```
319 \newcommand\wptitle{\wp@mk@title{\wp@num}: \pdata@target{wp}{\wp@id}{\pdataref{wp}\wp@id{title}}}
```

EdN:6 \wprm   [6]

```
320 \newcommand\wprm{\pdataref@safe{wp}\wp@id{RM}\if@RAM\ RM+\pdataref{wp}\wp@id{RAM} RAM\fi}
```

@site@contributes   Called as `\if@site@contributes{`⟨*site*⟩`}{`⟨*tokens*⟩`}` the following happens: If `\prop@gen@compactht` is `\@true` (set by the `compactht` attribute on the `proposal` environment), then ⟨*tokens*⟩ is processed. Otherwise, ⟨*tokens*⟩ is only processed if ⟨*site*⟩ contributes to the current work package (i.e the $RM \neq 0$ and $RAM \neq 0$)

---

[5]EDNOTE: document above

[6]EDNOTE: document above

```
321 \newcount\site@contribution%
322 \newcommand\if@site@contributes[2]{%
323 \ifx\prop@gen@compactht\@true
324 \if@RAM\ifnum\pdataref@num\wp@id{#1}{RM} > 0 \ifnum \pdataref@num\wp@id{#1}{RAM} > 0 #2\fi\fi
325 \else\ifnum\pdataref@num\wp@id{#1}{RM} > 0 #2\fi\fi
326 \else #2\fi}
```

\wp@sites@line The following macro computes the sites line (in the token register \wp@sites@line), the efforts
\wp@efforts@line line (in \wp@efforts@line), and the sites number (in the counter \sites@num) for later inclusion
\wp@sites@num in the \wpheadertable. If \prop@gen@compactht is \@true, then no sites without contributions
are listed in the table.

```
327 \newcounter{wp@sites@num}
328 \newcommand\wp@sites@efforts@lines{%
329 \setcounter{wp@sites@num}{0}
330 {\let\G@refundefinedtrue=\relax\let\@latex@warning=\relax\let\@sw\relax%
331 \let\site\relax\let\textbf\relax\let\sum@style\relax\let\lead@style\relax%
332 \let\pn\relax\let\sys\relax%
333 \xdef\wp@sites@line{\wp@legend@site}\xdef\wp@efforts@line{\wp@legend@effort}%initialize lines
334 \@for\@site:=\prop@gen@sites\do{\if@site@contributes\@site{\stepcounter{wp@sites@num}}%
335 \xdef\wp@sites@line{\wp@sites@line%
336 \if@site@contributes\@site{&%
337 \ifx\wp@swsites\@true%
338 \@sw{\ifx\@site\wp@lead\lead@style{\site{\@site}}\else\site{\@site}\fi}%
339 \else\ifx\@site\wp@lead\lead@style{\site{\@site}}\else\site{\@site}\fi%
340 \fi}}%
341 \xdef\wp@efforts@line{\wp@efforts@line%
342 \if@site@contributes\@site{&%
343 \ifx\@site\wp@lead%
344 \lead@style{\pdataref@safe\wp@id\@site{RM}\if@RAM+\pdataref@safe\wp@id\@site{RAM}\fi}
345 \else\pdataref@safe\wp@id\@site{RM}\if@RAM+\pdataref@safe\wp@id\@site{RAM}\fi\fi}}%
346 }% do
347 \xdef\wp@sites@line{\wp@sites@line&\sum@style{\wp@legend@all}}%
348 \xdef\wp@efforts@line{\wp@efforts@line&
349 \sum@style{\textbf{\pdataref{wp}\wp@id{RM}\if@RAM+\pdataref{wp}\wp@id{RAM}\fi}}}}}
```

\wpheadertable This macro computes the default work package header table, if there are sites.

```
350 \newcommand\wpheadertable{%
351 \wp@sites@efforts@lines%
352 \par\noindent\begin{tabular}{|l||l|*{\thewp@sites@num}{c|}|c|}\hline%
353 \textbf{\wp@mk@title{\wp@num}}&\wp@sites@line\\\hline%
354 \textsf{\pdata@target{wp}{\wp@id}{\pdataref{wp}\wp@id{title}}} &\wp@efforts@line\\\hline%
355 \end{tabular}\smallskip\par\noindent\ignorespaces}
```

and now multilinguality support

```
356 \newcommand\wp@legend@site{Site}
357 \newcommand\wp@legend@effort{Effort\if@RAM{ (RM+RAM)}\fi}
358 \newcommand\wp@legend@all{\textbf{all}}
```

workarea the workarea environment for work areas is almost the same, but we also have to initialize the
work package counters. Also, the efforts can be computed from the work packages in this group
via the wa@effort counter

```
359 \newcounter{prop@RM}\if@RAM\newcounter{prop@RAM}\fi
360 \ifwork@areas
361 \newcounter{wa@RM}\if@RAM\newcounter{wa@RAM}\fi\newcounter{wa@wps}
362 \newenvironment{workarea}[1][]
363 {\setkeys{workarea}{#1}
364 \let\@wps=\relax
365 \stepcounter{wa}
```

```
366 \pdata@def{wa}{\wa@id}{label}{\wa@label\thewa}
367 \pdata@def{wa}{\wa@id}{number}{\thewa}
368 \pdata@def{wa}{\wa@id}{page}{\thepage}
369 \update@was{\wa@id}
370 \pdata@def{wa}{\wa@id}{num}{\thewa}
371 \setcounter{wa@RM}{0}\if@RAM\setcounter{wa@RAM}{0}\fi\setcounter{wa@wps}{0}
372 \edef\@@wps{\pdataref@aux\wa@id{wp}{ids}}
373 \@for\@wp:=\@@wps\do{\stepcounter{wa@wps}%
374 \if@sites
375 \@for\@site:=\prop@gen@sites\do{%
376   \edef\@RM{\pdataref@num\@wp\@site{RM}}
377   \if@RAM\edef\@RAM{\pdataref@num\@wp\@site{RAM}}\fi
378   \addtocounter{wa@RM}{\@RM}\addtocounter{prop@RM}{\@RM}
379   \if@RAM\addtocounter{wa@RAM}{\@RAM}\addtocounter{prop@RAM}{\@RAM}\fi}
380 \else
381 \edef\@RM{\pdataref@num{wp}\@wp{RM}}
382 \if@RAM\edef\@RAM{\pdataref@num{wp}\@wp{RAM}}\fi
383 \addtocounter{wa@RM}{\@RM}\addtocounter{prop@RM}{\@RM}
384 \if@RAM\addtocounter{wa@RAM}{\@RAM}\addtocounter{prop@RAM}{\@RAM}\fi
385 \fi}
386 \pdata@def{wa}\wa@id{RM}\thewa@RM
387 \pdata@def{prop}{all}{RM}\theprop@RM
388 \if@RAM
389 \pdata@def{wa}\wa@id{RAM}\thewa@RAM
390 \pdata@def{prop}{all}{RAM}\theprop@RAM
391 \fi
392 \subsubsection*{{\wa@mk@title\thewa}: {\pdata@target{wa}\wa@id{\pdataref{wa}\wa@id{title}}}}
393 \addcontentsline{toc}{subsubsection}{{\wa@mk@title\thewa}: \pdataref{wa}\wa@id{title}}%
394 \ignorespaces
395 {\@ifundefined{@wps}{}{\pdata@def\wa@id{wp}{ids}\@wps}\pdata@def\wa@id{wp}{count}\thewa@wps}\fi
```

workplan   The `workplan` environment sets up the accumulator macros `\@wps`, `\@was`, for the collecting the
identifiers of work packages and work areas. At the end of the workplan description it writes out
their content to the aux file for reference.

```
396 \ifdelivs\newwrite\wpg@delivs\fi
397 \newenvironment{workplan}%
398 {\ifdelivs\immediate\openout\wpg@delivs=\jobname.delivs\fi
399 \ifwork@areas\let\@was=\relax\else\let\@wps=\relax\fi}%
400 {\@ifundefined{task@deps}{}{\pdata@def{all}{task}{deps}{\task@deps}}
401 \pdata@def{all}{task}{count}{\thealltasks}
402 \ifwork@areas
403 \@ifundefined{@was}{}{\pdata@def{all}{wa}{ids}\@was}
404 \else
405 \@ifundefined{@wps}{}{\pdata@def{all}{wp}{ids}\@wps}
406 \fi
407 \ifdelivs\@ifundefined{mile@stones}{}
408 {\@for\@I:=\mile@stones\do{%
409 \pdata@def{mile}\@I{delivs}{\@ifundefined{\@I delivs}{}{\csname\@I delivs\endcsname}}}}\fi
410 \ifwork@areas\pdata@def{all}{wa}{count}{\thewa}\fi
411 \pdata@def{all}{wp}{count}{\theallwp}
412 \ifdelivs
413 \pdata@def{all}{deliverables}{count}{\thedeliverable}
414 \pdata@def{all}{milestones}{count}{\themilestone}
415 \fi
416 \ifdelivs\closeout\wpg@delivs\fi}
```

4.7 Tasks

```
417 \newenvironment{tasklist}
418 {\begin{compactenum}}{\end{compactenum}}
```

The next step is to

```
419 \newcommand\task@label[2]{\textbf{T#1.#2}}
```

We define the keys for the task macro

```
420 \define@key{task}{id}{\def\task@id{#1}\@dmp{id=#1}}
421 \define@key{task}{wphases}{\def\task@wphases{#1}\@dmp{wphases=#1}}
422 \define@key{task}{requires}{\@requires\task@id{#1}\@dmp{req=#1}}
423 \define@key{task}{title}{\def\task@title{#1}}
424 \define@key{task}{lead}{\def\task@lead{#1}}
425 \define@key{task}{partners}{\def\task@partners{#1}}
426 \define@key{task}{PM}{\def\task@PM{#1}}
427 \define@key{task}{issue}{\def\task@issue{#1}}
428 \def\task@set#1{\edef\task@id{task\thetask@all}
429 \def\task@wphases{0-0}\def\task@partners{}\def\task@lead{}\def\task@PM{}\def\task@title{}
430 \setkeys{task}{#1}}
```

make the space after the title tweakable

```
431 \def\task@post@title@space{\;}
```

```
432 \newcounter{alltasks}
433 \def\task@post@title@space{\quad}
434 \newcommand\task@legend@partners{Sites: }
435 \newcommand\task@legend@PM{PM}
436 \newenvironment{task}[1][]%
437 {\stepcounter{alltasks}%
438 \@task{#1}\item[\pdata@target{task}{\taskin\task@id\wp@id}{\task@label\thewp\thetask@wp}]%
439 \ifx\task@title\@empty\else\textbf\task@title\fi\task@post@title@space%
440 \def\@initial{0-0}\ifx\task@wphases\@initial\else%
441 \let\@@sep=\relax\@for\@I:=\task@wphases%
442 \do{\decode@wphase\@I%
443 \@@sep\show@wphase\wphase@start\wphase@end\wphase@force%
444 \let\@@sep=\sep@wphases}%
445 \fi% initial
446 \hfill%
447 \ifsubmit\else\ifx\task@PM\@empty\else\task@PM~\task@legend@PM;\fi\fi%
448 \ifx\task@lead\@empty\else\ \task@legend@partners\site\task@lead~(\legend@lead)\fi%
449 \@for\@I:=\task@partners\do{, \site\@I}\\%
450 \ignorespaces}
451 {\medskip}
```

now the multilingual support and presentation configuration

```
452 \newcommand\month@label[1]{M#1}
453 \newcommand\show@wphase[3]{\edef\@test{#3}\def\@one{1}%
454 \month@label{#1}-\month@label{#2}%
455 \ifx\@test\@empty\else\ifx\@test\@one\else @#3\fi\fi}
456 \newcommand\sep@wphases{; }
457 \newcommand\legend@partners{Partners}
458 \newcommand\legend@lead{lead}
459 \newcommand\task@label@long{Task}
```

The \@task macro is a internal macro which takes a bunch of keyword keys and writes their values to the aux file.

20

```
460 \newcounter{task@all}\newcounter{task@wp}[wp]
461 \newcount\task@@end
462 \def\@task#1{\stepcounter{task@all}\stepcounter{task@wp}%
463 \task@set{#1}%
464 \pdata@def{task}{\taskin\task@id\wp@id}{title}{\task@title}
465 \pdata@def{task}{\taskin\task@id\wp@id}{lead}{\task@lead}
466 \pdata@def{task}{\taskin\task@id\wp@id}{partners}{\task@partners}
467 \pdata@def{task}{\taskin\task@id\wp@id}{PM}{\task@PM}
468 \pdata@def{task}{\taskin\task@id\wp@id}{wphases}{\task@wphases}
469 \@ifundefined{deliv@issue}{}
470 {\pdata@def{task}{\taskin\task@id\wp@id}{issue}{\task@issue}}%
471 \pdata@def{task}{\taskin\task@id\wp@id}{label}{\task@label\thewp\thetask@wp}%
472 \pdata@def{task}{\taskin\task@id\wp@id}{number}{\thetask@wp}%
473 \pdata@def{task}{\taskin\task@id\wp@id}{page}{\thepage}%
474 \update@tasks{\taskin\task@id\wp@id}}
```

4.8   Work Phase Metadata

```
475 \newcommand\workphase[1]{\PackageError{proposal}
476   {The \protect\workphase macro is deprecated,\MessageBreak
477     use the attributes wphase on the workpackage environment instead!}}
```

```
478 \newcommand\taskin[2]{#2@#1}
479 \newcommand\taskref[2]{\pdataRef{task}{#1@#2}{label}}
480 \newcommand\taskreflong[2]{\pdataRef{task}{#2}{label}}
481 \newcommand\tasktref[2]{\taskref{#1}{#2}: \pdataRefFB{task}{#1@#2}{short}{title}}
482 \newcommand\localtaskref[1]{\taskref{\wp@id}{#1}}
483 \newcommand\localtasktref[1]{\tasktref{\wp@id}{#1}}
```

now we initialize experimental infrastructure for task dependencies (not very well used/tested)

```
484 \newcounter{gantt@deps}
485 \def\@requires#1#2{\stepcounter{gantt@deps}%
486 \edef\dep@id{taskdep\thegantt@deps}%
487 \pdata@def{taskdep}\dep@id{from}{\taskin{#1}\wp@id}%
488 \pdata@def{taskdep}\dep@id{to}{#2}%
489 \update@deps\dep@id}
```

4.9   Milestones and Deliverables

this macro raises an error if deliverable commands are used without the `deliverables` option being set.

```
490 \newcommand\deliv@error{\PackageError{proposal}
491 {To use use deliverables, you have to specify the option 'deliverables'}}
```

```
492 \newenvironment{wpdelivs}{\begin{wp@delivs}}{\end{wp@delivs}}
```

```
493 \newenvironment{wp@delivs}
494 {\ifdelivs\textbf\deliv@legend@delivs:\\[-3ex]%
495 \begin{compactdesc}\else\deliv@error\fi}
496 {\ifdelivs\end{compactdesc}\fi}
```

and now multilinguality support

```
497 \newcommand\deliv@legend@delivs{Deliverables}
```

```
498 \newenvironment{wadelivs}
499 {\textbf\deliv@legend@delivs:\\[-3ex]\begin{wp@delivs}}
500 {\end{wp@delivs}}
```

\lec This macro is generally useful to put a comment at the end of the line, possibly making a new one if there is not enough space.

```
501 \newcommand\lec[1]{\strut\hfil\strut\null\nobreak\hfill\hbox{$\leadsto$#1}\par}
```

```
502 \newcommand\deliv@label[1]{D{#1}}
```

*deliv*ref This macro is generally useful to put a comment at the end of the line, possibly making a new one if there is not enough space.

```
503 \newcommand\delivref[2]{\pdataRef{deliv}{#1@#2}{label}}
504 \newcommand\localdelivref[1]{\delivref{\wp@id}{#1}}
505 \newcommand\delivtref[2]{\delivref{#1}{#2}: \pdataRefFB{deliv}{#1@#2}{short}{title}}
506 \newcommand\localdelivtref[1]{\delivtref{\wp@id}{#1}}
```

\wpg@deliv We first define the keys

```
507 \define@key{deliv}{id}{\def\deliv@id{#1}}
508 \define@key{deliv}{due}{\def\deliv@due{#1}}
509 \define@key{deliv}{dissem}{\def\deliv@dissem{#1}}
510 \define@key{deliv}{nature}{\def\deliv@nature{#1}}
511 \define@key{deliv}{miles}{\def\deliv@miles{#1}}
512 \define@key{deliv}{short}{\def\deliv@short{#1}}
513 \define@key{deliv}{lead}{\def\deliv@lead{#1}}
514 \define@key{deliv}{issue}{\def\deliv@issue{#1}}
```

The \wpdeliv macro cycles over the due dates and generates the relevant entries into the deliverables file. The first step is to write the general metadata to the pdata file.

```
515 \newcounter{deliverable}
516 \newcommand{\wpg@deliv}[3]{% keys, title, type
517 \stepcounter{deliverable}
518 \let\deliv@miles=\relax% clean state
519 \def\@type{#3}\def\@wp{wp}% set up ifx
520 \def\wpg@id{\csname #3@id\endcsname}
521 \setkeys{deliv}{#1}\stepcounter{deliv}% set state
522 \ifx\@type\@wp\def\current@label{\deliv@label{\ifwork@areas\thewa.\fi\thewp.\thedeliv}}
523 \else\def\current@label{\deliv@label{\thewa.\thedeliv}}\fi
524 \pdata@def{deliv}{\taskin\deliv@id\wpg@id}{label}{\current@label}
525 \pdata@def{deliv}{\taskin\deliv@id\wpg@id}{title}{#2}
526 \pdata@def{deliv}{\taskin\deliv@id\wpg@id}{page}{\thepage}%
527 \@ifundefined{deliv@short}
528 {\pdata@def{deliv}{\taskin\deliv@id\wpg@id}{short}{#2}}
529 {\pdata@def{deliv}{\taskin\deliv@id\wpg@id}{short}{\deliv@short}}
530 \@ifundefined{deliv@nature}
531 {\protect\G@refundefinedtrue\@latex@warning{key 'nature' for Deliv \wpg@id undefined}}
532 {\pdata@def{deliv}{\taskin\deliv@id\wpg@id}{nature}{\deliv@nature}}
533 \@ifundefined{deliv@dissem}
534 {\protect\G@refundefinedtrue\@latex@warning{key 'dissem' for Deliv \wpg@id undefined}}
535 {\pdata@def{deliv}{\taskin\deliv@id\wpg@id}{dissem}{\deliv@dissem}}
536 \@ifundefined{deliv@lead}
537 {\protect\G@refundefinedtrue\@latex@warning{key 'lead' for Deliv \wpg@id  undefined}}
538 {\pdata@def{deliv}{\taskin\deliv@id\wpg@id}{lead}{\deliv@lead}}
```

Then we iterate over the due dates and generate an entry for teach of them.

```
539 \@ifundefined{deliv@due}{}{%
```

```
540 \@ifundefined{deliv@issue}{}{\pdata@def{deliv}{\taskin\deliv@id\wpg@id}{issue}{\deliv@issue}}
541 \@for\@I:=\deliv@due\do{\protected@write\wpg@delivs{}{\string\deliverable%
542 {\ifnum\@I<10 0\@I\else\@I\fi}% sort key
543 {\@I}% due date
544 {\current@label}% label
545 {\@ifundefined{deliv@id}{??}{\taskin\deliv@id\wpg@id}}% id
546 {\@ifundefined{deliv@dissem}{??}{\deliv@dissem}}% dissemination level
547 {\@ifundefined{deliv@nature}{??}{\deliv@nature}}% nature
548 {#2}
549 {\ifx\@type\@wp{WP\ifwork@areas\thewa.\fi\thewp}\else{WA\thewa}\fi}%WP
550 {\@ifundefined{deliv@lead}{??}{\string\site{\deliv@lead}}}}}}% lead
```

And finally, we generate the entry into the deliverables table.

```
551 \item[\current@label\ (%
552 \delivs@legend@due: \@ifundefined{deliv@due}{??}{\deliv@due},
553 \delivs@legend@nature: \@ifundefined{deliv@nature}{??}{\deliv@nature},
554 \delivs@legend@dissem: \@ifundefined{deliv@dissem}{??}{\deliv@dissem},
555 \delivs@legend@lead: \@ifundefined{deliv@lead}{??}{\site{\deliv@lead}})]
556  \pdata@target{deliv}{\taskin\deliv@id\wpg@id}{\textit{#2}}
557 \@ifundefined{deliv@miles}{}{% print the milestones and update their deliverables
558 \let\m@sep=\relax% do not print the separator the first time round
559 \lec{\@for\@I:=\deliv@miles\do{% Iterate over the milestones mentioned
560 \m@sep\pdataRef{mile}{\@I}{label}% print the milestone reference
561 \let\m@sep=,}}%set the separator for the next times
562 \def\d@sep{,}
563  \@for\@I:=\deliv@miles\do{% Iterate over the milestones mentioned
564 \expandafter\ifx\csname\@I delivs\endcsname\relax% Check that the miles@delivs is empty
565  {\expandafter\xdef\csname\@I delivs\endcsname{\wpg@id @\deliv@id}}% if so, skip the separator
566   \else\expandafter\xdef\csname\@I delivs\endcsname%if not add it
567      {\csname\@I delivs\endcsname\d@sep\wpg@id @\deliv@id}\fi}}}
```

Now, we only need to instantiate

wadeliv

```
568 \newenvironment{wadeliv}[2][]{\ifdelivs\wpg@deliv{#1}{#2}{wa}\else\deliv@error\fi}{}
```

wpdeliv

```
569 \newenvironment{wpdeliv}[2][]{\ifdelivs\wpg@deliv{#1}{#2}{wp}\else\deliv@error\fi}{}
```

\milestone@label

```
570 \newcommand\milestone@label[1]{M{#1}}
```

\mileref   This macro is generally useful to put a comment at the end of the line, possibly making a new
one if there is not enough space.

```
571 \newcommand\mileref[1]{\pdataRef{mile}{#1}{label}}
572 \newcommand\miletref[1]{\mileref{#1}: \pdataRefFB{mile}{#1}{short}{title}}
```

\milestone   create a new milestone, initialize its deliverables accumulator macro, set up hyperlinking, and
extend the milestones list.

```
573 \newcounter{milestone}
574 \define@key{milestone}{id}{\gdef\mile@id{#1}}
575 \define@key{milestone}{month}{\gdef\mile@month{#1}}
576 \newcommand\milestone[3][]{%
577 \ifdelivs%
578 \setkeys{milestone}{#1}\stepcounter{milestone}%
579 \pdata@def{mile}\mile@id{label}{\milestone@label{\themilestone}}%
580 \pdata@def{mile}\mile@id{month}{\mile@month}%
581 \pdata@def{mile}\mile@id{title}{#2}%
```

```
582 \pdata@def{mile}\mile@id{description}{#3}%
583 \@ifundefined{mile@stones}%
584 {\xdef\mile@stones{\mile@id}}%
585 {\xdef\mile@stones{\mile@stones,\mile@id}}%
586 \@milestone{\mile@id}{#2}{#3}% presentation
587 \else\deliv@error\fi}
```

\@milestone   the corresponding presentation macro.

```
588 \newcommand\@milestone[3]{% id, title, description
589 \item \textbf{\miles@legend@milestone\xspace\pdata@target{mile}\mile@id{\pdataref{mile}{#1}{label}}
590 (\miles@legend@month \pdataref{mile}\mile@id{month})
591 \textbf{#2}} #3
592 \newcommand\miles@legend@month{Month}
593 \newcommand\miles@legend@milestone{Milestone}
```

milestones   This does the metadata bookkeeping, the layout is delegated to the presentation environment
@milestones and the legend macros that can be customized for specific proposals.

```
594 \newenvironment{milestones}%
595 {\ifdelivs\begin{@milestones}\else\deliv@error\fi}
596 {\ifdelivs\pdata@def{all}{mile}{ids}{\mile@stones}%
597 \pdata@def{all}{mile}{count}{\themilestone}%
598 \end{@milestones}\fi}
```

@milestones   here we do the work.

```
599 \newenvironment{@milestones}{\begin{enumerate}}{\end{enumerate}}
```

\deliverable   the first argument is an extended due date to facilitate sorting.

```
600 \newcommand{\deliverable}[9]{\pdataRef{deliv}{#4}{label}&#7&#8&#9&#6&#5&#2\\\hline}%sortkey,due,label,id,titl
```

deliverables

```
601 \newenvironment{deliverables}[1]{\ifdelivs\begin{longtable}{|l|p{#1}|l|l|l|l|l|}\hline%
602 \#&\textbf{\delivs@legend@name}&%
603 \textbf{\delivs@legend@wp}&%
604 \textbf{\delivs@legend@lead}&%
605 \textbf{\delivs@legend@nature}&%
606 \textbf{\delivs@legend@level}&%
607 \textbf{\delivs@legend@due}\\\hline\hline%
608 \endhead%
609 \else\deliv@error\fi}
610 {\ifdelivs\end{longtable}\fi}
```

now the multilingual support

```
611 \newcommand\delivs@legend@name{Deliverable name}
612 \newcommand\delivs@legend@wp{WP}
613 \newcommand\delivs@legend@nature{Type}
614 \newcommand\delivs@legend@level{Level}
615 \newcommand\delivs@legend@due{Due}
616 \newcommand\delivs@legend@dissem{Dissem.}
617 \newcommand\delivs@legend@lead{Lead}
```

\inputdelivs

```
618 \newcommand{\inputdelivs}[1]{%
619 \begin{deliverables}{#1}%
620 \IfFileExists{\jobname.deliverables}%
621 {\input{\jobname.deliverables}}%
622 {\IfFileExists{\jobname.delivs}{\input{\jobname.delivs}}{}}
623 \end{deliverables}}
624 ⟨/sty⟩
```

4.10 Project Data, Referencing & Hyperlinking

\pdata@* \quad \pdata@out is the file handle for the project data file, we define internal macros to open and close it.

```
625 ⟨*pdata⟩
626 \newif\ifwork@areas\work@areastrue
627 \DeclareOption{noworkareas}{\work@areasfalse}
628 \ProcessOptions
629 \RequirePackage{xspace}
630 \newwrite\pdata@out
631 \newcommand\pdata@open[1]{\immediate\openout\pdata@out=#1.pdata}
632 \newcommand\pdata@close{\closeout\pdata@out}
```

\readpdata \quad This macro reads the project data file and its error handling

```
633 \newcommand\readpdata[1]{\IfFileExists{#1.pdata}
634 {\message{proposal: Reading Project Data}\makeatletter\input{#1.pdata}\makeatother}
635 {proposal: No Project Data found, (forward) references may be compromised}}
```

\pdata@target \quad This internal macro makes a hyper-target: \pdata@target{⟨cat⟩}{⟨id⟩}{⟨label⟩} prints ⟨label⟩ with a target name ⟨cat⟩@⟨id⟩@target attached to it.

```
636 \newcommand\pdata@target[3]{\hypertarget{#1@#2@target}{#3}}
```

\pdata@def \quad This macro writes an \@pdata@def command to the current aux file and also executes it.

```
637 \newcommand\pdata@def[4]{%\@pdata@def{#1}{#2}{#3}{#4}%
638   \protected@write\pdata@out{}{\string\@pdata@def{#1}{#2}{#3}{#4}}}
```

\@pdata@def \quad This macro stores the value of its last argument in a custom macro for reference.

```
639 \newcommand\@pdata@def[4]{\expandafter\gdef\csname #1@#2@#3\endcsname{#4}}
```

\pdataref

```
640 \newcommand\pdataref[3]{\@ifundefined{#1@#2@#3}%
641           {\protect\G@refundefinedtrue\@latex@warning{#3 for #1 #2 undefined}??}%
642            {\csname #1@#2@#3\endcsname}}%
643 \newcommand\pdataref@aux[3]{\@ifundefined{#1@#2@#3}{??}{\csname #1@#2@#3\endcsname}}%
644 \newcommand\pdataref@num[3]{\@ifundefined{#1@#2@#3}{0}{\csname #1@#2@#3\endcsname}}%
645 \newcommand\pdataref@safe[3]{\@ifundefined{#1@#2@#3}{}{\csname #1@#2@#3\endcsname}}%
```

\pdatarefFB \quad a variant with fallback field,

```
646 \newcommand\pdatarefFB[4]{\@ifundefined{#1@#2@#3}%
647 {\@ifundefined{#1@#2@#4}%
648 {\protect\G@refundefinedtrue\@latex@warning{both #3 and its fallback #4 undefined for #1 #2}??}%
649 {\csname #1@#2@#4\endcsname}}
650 {\csname #1@#2@#3\endcsname}}
```

\pdataRef

```
651 \newcommand\pdataRef[3]{\@ifundefined{#1@#2@#3}%
652 {\protect\G@refundefinedtrue\@latex@warning{#3 for #1 #2 undefined}??}%
653 {\hyperlink{#1@#2@target}{\csname #1@#2@#3\endcsname}}}
```

\pdataRefFB \quad a variant with fallback field,

```
654 \newcommand\pdataRefFB[4]{\@ifundefined{#1@#2@#3}%
655 {\@ifundefined{#1@#2@#4}%
656 {\protect\G@refundefinedtrue\@latex@warning{both #3 and its fallback #4 undefined for #1 #2}??}%
657 {\hyperlink{#1@#2@target}{\csname #1@#2@#4\endcsname}}}
658 {\hyperlink{#1@#2@target}{\csname #1@#2@#3\endcsname}}}
```

```
659 \newcommand\prop@count[1]{\ifcase #1 zero\or one\or two\or three\or four\or five\or six\or seven \or
660   eight\or nine\or ten\or eleven \or twelve\else#1\fi}
661 \newcommand\pdatacount[2]{\prop@count{\pdataref@num{#1}{#2}{count}}}
```

\pn*

```
662 \newcommand\pn{\pdataref{prop}{gen}{acronym}\xspace}
663 \newcommand\pnlong{\pdataref{prop}{gen}{acrolong}\xspace}
```

\W*ref

```
664 \newcommand\WPref[1]{\pdataRef{wp}{#1}{label}}
665 \newcommand\WPtref[1]{\WPref{#1}: \pdataRefFB{wp}{#1}{short}{title}}
666 \ifwork@areas
667 \newcommand\WAref[1]{\pdataRef{wa}{#1}{label}}
668 \newcommand\WAtref[1]{\WAref{#1}: \pdataRefFB{wa}{#1}{short}{title}}
669 \fi
670 ⟨/pdata⟩
```

4.11   The Work Package Table

EdN:7 *@style   These macros[7] determine the styling of cells in the work package table. That can be tweaked by redefining them.

```
671 ⟨∗sty⟩
672 \definecolorset{gray/rgb/hsb/cmyk}{}{}%
673 {leadgray,.90/.90,.90,.90/0,0,.90/0,0,0,.10;%
674 wagray,.70/.70,.70,.70/0,0,.70/0,0,0,.30;%
675 ganttgray,.60/.60,.60,.60/0,0,.60/0,0,0,.40}
676 \newcommand\sum@style[1]{\cellcolor{wagray}{\textbf{#1}}}
677 \newcommand\wa@style[1]{\cellcolor{wagray}{\textbf{#1}}}
678 \newcommand\wp@style[1]{#1}
679 \newcommand\lead@style[1]{\cellcolor{leadgray}{\textit{#1}}}
680 \newcommand\wp@lead@style@explained{light gray italicised}
```

\wpfigstyle

```
681 \def\wpfig@style{}
682 \newcommand\wpfigstyle[1]{\def\wpfig@style{#1}}
```

We first define the options for the \wpfig macro, they specify what columns we have in the table.

```
683 \newcounter{wpfig@options}
684 \define@key{wpfig}{size}{\def\wpfig@size{#1}\@dmp{size=#1}}
685 \def\@true{true}
686 \def\wpfig@pages{false}
687 \define@key{wpfig}{pages}[true]{\def\wpfig@pages{#1}\stepcounter{wpfig@options}}
688 \def\wpfig@type{false}
689 \define@key{wpfig}{type}[true]{\def\wpfig@type{#1}\stepcounter{wpfig@options}}
690 \def\wpfig@start{false}
691 \define@key{wpfig}{start}[true]{\def\wpfig@start{#1}\stepcounter{wpfig@options}}
692 \def\wpfig@length{false}
693 \define@key{wpfig}{length}[true]{\def\wpfig@length{#1}\stepcounter{wpfig@options}}
694 \def\wpfig@end{false}
695 \define@key{wpfig}{end}[true]{\def\wpfig@end{#1}\stepcounter{wpfig@options}}
696 \define@key{wpfig}{label}{\def\wpfig@label{#1}}
697 \define@key{wpfig}{caption}{\def\wpfig@caption{#1}}
```

---

[7]EdNote: maybe add "wpfig" in the name to show dependency

wp@figure This environment makes legend for the table (but not the contents) for the `\wpfig` macro. The main work achieved here is to generate the head line (sideways) and the footer in the various cases given by the package options.[8] Depending on the various class and `wpfig` options, we make header and footer line for the table.

EdN:8

```
698 \def\@sw#1{\begin{sideways}#1\end{sideways}}
699 \newenvironment{wp@figure}{\begin{table}[ht]\wpfig@style\begin{center}
700 {\let\@sw\relax\let\textbf\relax\let\site\relax\let\pn\relax\let\sys\relax%
701 \gdef\wpfig@headline{\wpfig@legend@wap&\wpfig@legend@title%
702 \ifx\wpfig@type\@true&\wpfig@legend@type\fi%
703 \ifx\wpfig@pages\@true&\@sw{\wpfig@legend@page}\fi%
704 \ifx\wpfig@start\@true&\@sw{\wpfig@legend@start}\fi%
705 \ifx\wpfig@length\@true&\@sw{\wpfig@legend@length}\fi
706 \ifx\wpfig@end\@true&\@sw{\wpfig@legend@end}\fi}%
707 \if@sites%
708 \@for\@site:=\prop@gen@sites\do{%
709 \xdef\wpfig@headline{\wpfig@headline&\@sw{\wpfig@legend@siteRM{\@site}}}%
710 \if@RAM\xdef\wpfig@headline{\wpfig@headline&\@sw{\wpfig@legend@siteRAM{\@site}}}\fi%
711 \xdef\wpfig@headline{\wpfig@headline&\@sw{\wpfig@legend@totalRM}}%
712 \if@RAM\xdef\wpfig@headline{\wpfig@headline&\@sw{\wpfig@legend@totalRAM}}\fi%
713 \else% if@sites
714 \xdef\wpfig@headline{\wpfig@headline &\@sw{\wpfig@legend@RM}\if@RAM&\@sw{\wpfig@legend@RAM}\fi}
715 \fi}%if@sites
716 \if@RAM\begin{tabular}{|l|l|*{\thewpfig@options}{r|}*{\the@sites}{r|r|}|r|r|}\hline
717 \else\begin{tabular}{|l|l|*{\thewpfig@options}{r|}|*{\the@sites}{r|}|r|}\hline\fi%|
718 \wpfig@headline\\\hline\hline}
719 {\end{tabular}\smallskip\\
720 \wpfig@legend@RAM@expl\if@sites; \wpfig@legend@lead@expl\fi
721 \@ifundefined{wpfig@label}{\caption{\wpfig@legend@caption}}{\caption{\wpfig@caption}}
722 \@ifundefined{wpfig@label}{\label{fig:wplist}}{\label{\wpfig@label}}
723 \end{center}\end{table}}
```

and now multilinguality support

```
724 \newcommand\wpfig@legend@wap{\textbf{\ifwork@areas{WA/P}\else{WP}\fi}}
725 \newcommand\wpfig@legend@title{\textbf{Title}}
726 \newcommand\wpfig@legend@type{\textbf{type}}
727 \newcommand\wpfig@legend@page{\textbf{page}}
728 \newcommand\wpfig@legend@start{\textbf{start}}
729 \newcommand\wpfig@legend@length{\textbf{length}}
730 \newcommand\wpfig@legend@end{\textbf{end}}
731 \newcommand\wpfig@legend@siteRM[1]{\site{#1}\if@RAM\ RM\fi}
732 \newcommand\wpfig@legend@siteRAM[1]{\site{#1}\ RAM}
733 \newcommand\wpfig@legend@totalRM{total\if@RAM\ RM\fi}
734 \newcommand\wpfig@legend@totalRAM{total RAM}
735 \newcommand\wpfig@legend@RM{RM}
736 \newcommand\wpfig@legend@RAM{RAM}
737 \newcommand\wpfig@legend@RAM@expl{\if@RAM R(A)M $\widehat=$ Researcher (Assistant) Months\else\ Efforts in PM
738 \newcommand\wpfig@legend@lead@expl{WP lead efforts \wp@lead@style@explained}
739 \newcommand\wpfig@legend@caption{{\ifwork@areas Work Areas and \fi}Work Packages}
```

EdN:9 `\wpfig` [9]

```
740 \newcount\local@count
741 \newcount\@@@RM\if@RAM\newcount\@@@RAM\fi
742 \newcount\all@@@RM\if@RAM\newcount\all@@@RAM\fi
743 \newcommand{\wpfig}[1][]{\setcounter{wpfig@options}{0}\setkeys{wpfig}{#1}
```

---

[8]EDNOTE: this is a bit of misnomer, it does not do the figure bit.

[9]EDNOTE: The computation can be distributed much more efficiently (by intermingling the counter advances with the row creation), but this works now

the first thing to do is to build the body of the table programmatically by (globally) extending the `\@wp@lines` token register inside a bracket group which locally redefines all macros we are using in the extensions, so that they do not get into the way. We start this group now.

```
744 {\gdef\@wp@lines{}%initialize
745 \let\tabularnewline\relax\let\hline\relax\let\lead@style\relax% so they
746 \let\wa@style\relax\let\wp@style\relax \let\@sw\relax\let\textbf\relax% do not
747 \let\G@refundefinedtrue=\relax\let\@latex@warning=\relax\let\hyperlink=\relax% bother
748 \let\pn\relax\let\xspace\relax% us
```

The code that follows now, could be more elegant, if we had a better way of organizing the data, but this works for now, we have four cases: with/without work areas and with/without sites. All do something very similar.

```
749 \ifwork@areas
750 \edef\@@was{\pdataref@safe{all}{wa}{ids}}%
751 \@for\@@wa:=\@@was\do{% iterate over the work areas
752 \xdef\@@wa@line{\wa@style{\pdataRef{wa}\@@wa{label}}%
753 &\wa@style{\@ifundefined{wa@\@@wa @short}{\pdataref{wa}\@@wa{title}}{\pdataref{wa}\@@wa{short}}}%
754 \ifx\wpfig@type\@true&\wa@style{\pdataref{wa}\@@wa{type}}\fi%
755 \ifx\wpfig@pages\@true&\wa@style{\pdataref{wa}\@@wa{page}}\fi%
756 \ifx\wpfig@start\@true&\wa@style{\pdataref{wa}\@@wa{start}}\fi%
757 \ifx\wpfig@length\@true&\wa@style{\pdataref{wa}\@@wa{len}}\fi%
758 \ifx\wpfig@end\@true&\wa@style{\pdataref{wa}\@@wa{end}}\fi}
759 \if@sites
760 \@for\@site:=\prop@gen@sites\do{%
761 \edef\@@wps{\pdataref@safe\@@wa{wp}{ids}}%
762 \local@count 0%
763 \@for\@@wp:=\@@wps\do{\advance\local@count by \pdataref@num\@@wp\@site{RM}}%
764 \pdata@def\@@wa\@site{RM}{\the\local@count}%
765 \xdef\@@wa@line{\@@wa@line&\wa@style{\the\local@count}}%
766 \if@RAM
767 \local@count 0%
768 \@for\@@wp:=\@@wps\do{\advance\local@count by \pdataref@num\@@wp\@site{RAM}}
769 \pdata@def\@@wa\@site{RAM}{\the\local@count}%
770 \xdef\@@wa@line{\@@wa@line&\wa@style{\the\local@count}}%
771 \fi}
772 \local@count0\relax%
773 \@for\@site:=\prop@gen@sites\do{\global\advance\local@count by \pdataref@num\@@wa\@site{RM}}%
774 \xdef\@@wa@line{\@@wa@line &\wa@style{\textbf{\the\local@count}}}
775 \if@RAM
776 \local@count0\relax%
777 \@for\@site:=\prop@gen@sites\do{\global\advance\local@count by \pdataref@num\@@wa\@site{RAM}}%
778 \xdef\@@wa@line{\@@wa@line &\wa@style{\textbf{\the\local@count}}}
779 \fi
780 \else% if@sites
781 \edef\@@wps{\pdataref@safe{all}{wp}{ids}}%
782 \xdef\@@wa@line{\@@wa@line&\wa@style{\pdataref{wa}\@@wa{RM}}
783 \if@RAM&\wa@style{\pdataref{wa}\@@wa{RAM}}\fi}%
784 \fi% if@sites
785 \xdef\@wp@lines{\@wp@lines\@@wa@line\tabularnewline\hline}% add the line for the workarea
786 \edef\@@wps{\pdataref@safe\@@wa{wp}{ids}}%
787 \@for\@@wp:=\@@wps\do{% iterate over its work packages
788 \xdef\@@wp@line{\pdataRef{wp}\@@wp{label}%
789 &\@ifundefined{wp@\@@wp @short}{\pdataref{wp}\@@wp{title}}{\pdataref{wp}\@@wp{short}}}%
790 \ifx\wpfig@type\@true&\pdataref{wp}\@@wp{type}\fi%
791 \ifx\wpfig@pages\@true&\pdataref{wp}\@@wp{page}\fi%
792 \ifx\wpfig@start\@true&\pdataref{wp}\@@wp{start}\fi%
793 \ifx\wpfig@length\@true&\pdataref{wp}\@@wp{len}\fi%
794 \ifx\wpfig@end\@true&\pdataref{wp}\@@wp{end}\fi}
```

```
795 \if@sites
796 \@for\@site:=\prop@gen@sites\do{%
797 \edef\@@lead{\pdataref@safe{wp}\@@wp{lead}}
798 \edef\@@RM{\ifx\@@lead\@site\lead@style{\pdataref@safe\@@wp\@site{RM}}\else\wp@style{\pdataref@safe\@@wp\@sit
799 \xdef\@@wp@line{\@@wp@line&\@@RM}
800 \if@RAM
801 \edef\@@RAM{\ifx\@@lead\@site\lead@style{\pdataref@safe\@@wp\@site{RAM}}\else\wp@style{\pdataref@safe\@@wp\@s
802 \xdef\@@wp@line{\@@wp@line&\@@RAM}
803 \fi}
804 \local@count0\relax%
805 \@for\@site:=\prop@gen@sites\do{\global\advance\local@count by \pdataref@num\@@wp\@site{RM}}%
806 \xdef\@@wp@line{\@@wp@line &\textbf{\the\local@count}}
807 \if@RAM
808 \global\local@count0\relax%
809 \@for\@site:=\prop@gen@sites\do{\global\advance\local@count by \pdataref@num\@@wp\@site{RAM}}%
810 \xdef\@@wp@line{\@@wp@line &\textbf{\the\local@count}}
811 \fi% if@RAM
812 \else% if@sites
813 \xdef\@@wp@line{\@@wp@line&\wp@style{\pdataref@safe{wp}\@@wp{RM}}}
814 \if@RAM\xdef\@@wp@line{\@@wp@line&\wp@style{\pdataref@safe{wp}\@@wp{RAM}}}\fi
815 \fi% if@sites
816 \xdef\@wp@lines{\@wp@lines\@@wp@line\tabularnewline\hline}}}
```

Now the case where we do not have work areas.

```
817 \else% ifwork@areas
818 \edef\@@wps{\pdataref@safe{all}{wp}{ids}}%
819 \@for\@@wp:=\@@wps\do{% iterate over its work packages
820 \xdef\@@wp@line{\pdataRef{wp}\@@wp{label}%
821 &\@ifundefined{wp@\@@wp @short}{\pdataref{wp}\@@wp{title}}{\pdataref{wp}\@@wp{short}}
822 \ifx\wpfig@type\@true&\pdataref{wp}\@@wp{type}\fi%
823 \ifx\wpfig@pages\@true&\pdataref{wp}\@@wp{page}\fi%
824 \ifx\wpfig@start\@true&\pdataref{wp}\@@wp{start}\fi%
825 \ifx\wpfig@length\@true&\pdataref{wp}\@@wp{len}\fi%
826 \ifx\wpfig@end\@true&\pdataref{wp}\@@wp{end}\fi}
827 \if@sites
828 \@for\@site:=\prop@gen@sites\do{%
829 \edef\@@lead{\pdataref@safe{wp}\@@wp{lead}}
830 \edef\@@RM{\ifx\@@lead\@site\lead@style{\pdataref@safe\@@wp\@site{RM}}\else\wp@style{\pdataref@safe\@@wp\@sit
831 \xdef\@@wp@line{\@@wp@line&\@@RM}
832 \if@RAM
833 \edef\@@RAM{\ifx\@@lead\@site\lead@style{\pdataref@safe\@@wp\@site{RAM}}\else\wp@style{\pdataref@safe\@@wp\@s
834 \xdef\@@wp@line{\@@wp@line&\wp@style\@@RAM}
835 \fi}
836 \global\local@count0\relax%
837 \@for\@site:=\prop@gen@sites\do{\global\advance\local@count by \pdataref@num\@@wp\@site{RM}}%
838 \xdef\@@wp@line{\@@wp@line &\textbf{\the\local@count}}
839 \if@RAM
840 \global\local@count0\relax%
841 \@for\@site:=\prop@gen@sites\do{\global\advance\local@count by \pdataref@num{#1}\@site{RAM}}%
842 \xdef\@@wp@line{\@@wp@line &\textbf{\the\local@count}}
843 \fi
844 \else% if@sites
845 \xdef\@@wp@line{\@@wp@line&\wp@style{\pdataref@safe{wp}\@@wp{RM}}}
846 \if@RAM\xdef\@@wp@line{\@@wp@line&\wp@style{\pdataref@safe{wp}\@@wp{RAM}}}\fi}
847 \fi% if@sites
848 \xdef\@wp@lines{\@wp@lines\@@wp@line\tabularnewline\hline}}
849 \fi%ifwork@areas
```

Now we compute the totals lines in the \@totals macros; again there are four cases to consider

```
850 \gdef\@totals{}
851 \ifwork@areas
852 \if@sites
853 \@for\@site:=\prop@gen@sites\do{% iterate over the sites
854 \@@@RM=0\if@RAM\@@@RAM=0\fi
855 \edef\@@was{\pdataref@safe{all}{wa}{ids}}%
856 \@for\@@wa:=\@@was\do{% iterate over the work areas
857 \edef\@@wps{\pdataref@safe\@@wa{wp}{ids}}%
858 \@for\@@wp:=\@@wps\do{% iterate over the work packages
859 \advance\@@@RM by \pdataref@num\@@wp\@site{RM}%
860 \if@RAM\advance\@@@RAM by \pdataref@num\@@wp\@site{RAM}\fi}}
861 \pdata@def{all}\@site{RM}{\the\@@@RM}\if@RAM\pdata@def{all}\@site{RAM}{\the\@@@RAM}\fi
862 \advance\all@@@RM by \the\@@@RM\if@RAM\advance\all@@@RAM by \the\@@@RAM\fi
863 \xdef\@totals{\@totals & \textbf{\the\@@@RM}\if@RAM& \textbf{\the\@@@RAM}\fi}}
864 \xdef\@totals{\@totals & \textbf{\the\all@@@RM}\if@RAM&\textbf{\the\all@@@RAM}\fi}
865 \pdata@def{all}{total}{RM}{\the\all@@@RM}\if@RAM\pdata@def{all}{total}{RAM}{\the\all@@@RAM}\fi
866 \else% if@sites
867 \@@@RM=0\if@RAM\@@@RAM=0\fi
868 \edef\@@was{\pdataref@safe{all}{wa}{ids}}%
869 \@for\@@wa:=\@@was\do{\edef\@@wps{\pdataref@safe\@@wa{wp}{ids}}%
870 \@for\@@wp:=\@@wps\do{% iterate over the work packages
871 \advance\@@@RM by \pdataref@num{wp}\@@wp{RM}%
872 \if@RAM\advance\@@@RAM by \pdataref@num{wp}\@@wp{RAM}\fi}}
873 \pdata@def{all}{total}{RM}{\the\@@@RM}\if@RAM\pdata@def{all}{total}{RAM}{\the\@@@RAM}\fi
874 \xdef\@totals{&\the\@@@RM\if@RAM &\the\@@@RAM\fi}
875 \fi% if@sites
876 \else%i.e. no work@areas
877 \if@sites
878 \@for\@site:=\prop@gen@sites\do{%iterate over the sites
879 \@@@RM=0\if@RAM\@@@RAM=0\fi%
880 \edef\@@wps{\pdataref@safe{all}{wp}{ids}}%
881 \@for\@@wp:=\@@wps\do{% iterate over the work packages
882 \advance\@@@RM by \pdataref@num\@@wp\@site{RM}%
883 \if@RAM\advance\@@@RAM by \pdataref@num\@@wp\@site{RAM}\fi}
884 \pdata@def{all}\@site{RM}{\the\@@@RM}\if@RAM\pdata@def{all}\@site{RAM}{\the\@@@RAM}\fi
885 \xdef\@totals{\@totals & \textbf{\the\@@@RM}\if@RAM& \textbf{\the\@@@RAM}\fi}
886 \advance\all@@@RM by \the\@@@RM\if@RAM\advance\all@@@RAM by \the\@@@RAM\fi}
887 \xdef\@totals{\@totals &\textbf{\the\all@@@RM}\if@RAM&\textbf{\the\all@@@RAM}\fi}
888 \pdata@def{all}{total}{RM}{\the\all@@@RM}\if@RAM\pdata@def{all}{total}{RAM}{\the\all@@@RAM}\fi
889 \else% if@sites
890 \@@@RM=0\if@RAM\@@@RAM=0\fi
891 \edef\@@wps{\pdataref@safe{all}{wp}{ids}}%
892 \@for\@@wp:=\@@wps\do{% iterate over the work packages
893 \advance\@@@RM by \pdataref@num{wp}\@@wp{RM}%
894 \if@RAM\advance\@@@RAM by \pdataref@num{wp}\@@wp{RAM}\fi}
895 \pdata@def{all}{total}{RM}{\the\@@@RM}\if@RAM\pdata@def{all}{total}{RAM}{\the\@@@RAM}\fi
896 \xdef\@totals{&\the\@@@RM\if@RAM &\the\@@@RAM\fi}
897 \fi% if@sites
898 \fi
```

And we finally have a line for the intended totals which we use in draft mode.

```
899 \gdef\intended@totals{}\gdef\requested@totals{}
900 \if@sites
901 \@for\@site:=\prop@gen@sites\do{
902 \xdef\intended@totals{\intended@totals&\textbf{\pdataref@safe{site}\@site{intendedRM}}}
903 \xdef\requested@totals{\requested@totals&\pdataref@safe{site}\@site{reqPM}}
904 \if@RAM\xdef\intended@totals{\intended@totals&\textbf{\pdataref@safe{site}\@site{intendedRAM}}}\fi}
905 \if@RAM\xdef\intended@totals{\intended@totals&&}\else%
906 \xdef\intended@totals{\intended@totals&}%
```

```
907 \xdef\requested@totals{\requested@totals&}%
908 \fi
909 \else% if@sites
910 \xdef\intended@totals{\intended@totals&\textbf{\pdataref@safe{all}{intended}{RM}}}
911 \if@RAM\xdef\intended@totals{\intended@totals&\textbf{\pdataref@safe{all}{intended}{RAM}}}\fi
912 \fi}% if@sites
```

finally, we make all of this into a figure, computing the colspan of the the legend cells for the totals via `\local@count` from the optional columns.

```
913 \local@count\thewpfig@options\advance\local@count by 2
914 \begin{wp@figure}
915 \@wp@lines\hline%
916 \multicolumn{\the\local@count}{|c|}{\prop@legend@totals}\@totals\\\hline%
917 \ifsubmit\else%
918 \ifx\prop@gen@topdownPM\@true%
919 \multicolumn{\the\local@count}{|c|}{\prop@legend@intendedtotals}\intended@totals\\\hline%
920 \fi% topdownPM
921 \ifx\prop@gen@botupPM\@true%
922 \multicolumn{\the\local@count}{|c|}{\prop@legend@requestedtotals}\requested@totals\\\hline%
923 \fi% botupPM
924 \fi% submit
925 \end{wp@figure}}
```

and now multilinguality support

```
926 \newcommand\prop@legend@totals{\textbf{totals}}
927 \newcommand\prop@legend@intendedtotals{\textbf{intended totals}}
928 \newcommand\prop@legend@requestedtotals{\textbf{requested totals}}
```

4.12   Gantt Charts

Gantt Charts are done with help of the the `tikz` package. The `gantt` environments pick up on the declared duration of the proposal in months stored in the `\prop@gen@months` macro.

We define the keys for Gantt tables

```
929 \newif\ifgantt@draft\gantt@draftfalse
930 \newif\ifgantt@miles\gantt@milesfalse
931 \define@key{gantt}{xscale}{\def\gantt@xscale{#1}}
932 \define@key{gantt}{yscale}{\def\gantt@yscale{#1}}
933 \define@key{gantt}{step}{\def\gantt@step{#1}}
934 \define@key{gantt}{size}{\def\gantt@size{#1}}
935 \define@key{gantt}{draft}[true]{\ifsubmit\else\gantt@drafttrue\fi}
936 \define@key{gantt}{milestones}[true]{\gantt@milestrue}
```

Then we define an auxiliary function that provides defaults for these keys and sets the internal macros.

```
937 \def\gantt@set#1{\gantt@draftfalse\def\gantt@xscale{1}\def\gantt@yscale{.35}\def\gantt@step{3}
938 \setkeys{gantt}{#1}}
```

Finally, the Gantt Chart environment itself.

gantt   The gantt[⟨*keyvals*⟩]{⟨*height*⟩} environment sets up the grid and legend for a gantt chart. The grid is `\prop@gen@months` wide and ⟨*height*⟩ high.

```
939 \newenvironment{gantt}[2][]
940 {\gantt@set{#1}\gdef\gantt@height{#2}
941 \def\@test{\prop@gen@months@default}
942 \ifx\@test\prop@gen@months
943 \ClassError{proposal}{Need overall project months to draw gantt
944     chart - expect trouble;\MessageBreak specify
945     \protect\begin{proposal}[...,months=??,...] to fix}\fi
946 \@ifundefined{gantt@size}{}{\csname\gantt@size\endcsname}
```

```
947 \newdimen\gantt@ymonths
948 \gantt@ymonths=\gantt@height cm
949 \advance\gantt@ymonths by .8cm
950 \begin{tikzpicture}[xscale=\gantt@xscale,yscale=\gantt@yscale]}
951 {\draw[xstep=\gantt@step,very thin] (0,0) grid (\prop@gen@months,\gantt@height);
952 \foreach \x in {0,\gantt@step,...,\prop@gen@months} \node at (\x,\gantt@ymonths) {\x};
953 \ifgantt@miles
954 \newdimen\gantt@ymiles\gantt@ymiles=\gantt@height cm
955 \advance\gantt@ymiles by 2cm
956 \newdimen\gantt@ymiles@top\gantt@ymiles@top=\gantt@height cm
957 %\advance\gantt@ymiles@top by 2cm
958 \edef\@@miles{\pdataref@safe{all}{mile}{ids}}
959 \@for\@I:=\@@miles\do{%
960 \edef\@@month{\pdataref@safe{mile}{\@I}{month}}
961 \draw[very thick,blue] (\@@month,\gantt@ymiles@top) -- (\@@month,0);
962 \node[blue] at (\@@month,\gantt@ymiles) {\pdataref{mile}{\@I}{label}};}
963 \fi %gantt@miles
964 \end{tikzpicture}}
```

\@action In this we have used the macro that does the actual painting. `\@action{⟨name⟩}{⟨line⟩}{⟨start⟩}{⟨len⟩}{⟨force⟩}` creates a gantt node with name ⟨name⟩ in line ⟨line⟩ starting at month ⟨month⟩ with length ⟨len⟩ that is ⟨force⟩ thick.

```
965 \newdimen\gantt@ymid\newdimen\gantt@yinc\newdimen\gantt@xend
966 \newcommand{\@action}[6][]{\def\@test{#1}%
967 \ifx\@test\@empty\def\@@color{ganttgray}\else\def\@@color{#1}\fi
968 \gantt@ymid=#3 cm\gantt@yinc=\gantt@yscale cm
969 \gantt@xend=#4 cm\advance\gantt@xend by #5 cm
970 \advance\gantt@ymid by \gantt@yinc
971 \fill[\@@color] (#4,#3) rectangle +(#5,#6);
972 \node (#2@left) at (#4,\gantt@ymid) {};
973 \node (#2@right) at (\gantt@xend,\gantt@ymid) {};}
```

\@dependency

```
974 \def\@dependency#1#2{\draw[->,line width=2pt,color=red] (#1@right) -- (#2@left);}
```

tt@compute@effort A helper function that updates the dimension `\gantt@effort` according to whether the counter `\gantt@month` is in the range. It is used in `\gantt@chart`

```
975 \newcommand\gantt@compute@effort[3]{% start, len, force
976   \@@e=#1\advance\@@e by #2
977   \ifnum\thegantt@month<#1\else
978   \ifnum\thegantt@month<\@@e
979   \gantt@plus=#3cm\advance\gantt@effort by \gantt@plus\fi\fi}
```

\ganttchart This macro iterates over the work areas, their work packages, and finally their work phases to use the internal macro `\@action`. All of this in the gantt setting.

```
980 \newcommand{\ganttchart}[1][]{\begin{figure}[ht]\centering
981 \gantt@set{#1}
982 \def\gantt@wps{\pdataref@num{all}{wp}{count}}
983 \begin{gantt}[#1]{\gantt@wps}
984 \newcounter{taskwps}\newcount\@@line
985 \edef\@@was{\pdataref@safe{all}{wa}{ids}}
986 \ifwork@areas
987 \@for\@@wa:=\@@was\do{% iterate over work areas
988   \edef\@@wps{\pdataref@safe\@@wa{wp}{ids}}
989   \@for\@@wp:=\@@wps\do{% iterate over work packages
990     \stepcounter{taskwps}
991     \@@line=\gantt@wps\advance\@@line by -\thetaskwps
992     \edef\@@tasks{\pdataref@safe\@@wp{task}{ids}}
```

```
993    \node at (-1/\gantt@xscale,\@@line) [above=-2pt] {\pdataRef{wp}\@@wp{label}};
994    \edef\@@wphases{\pdataref@safe{wp}\@@wp{wphases}}
995    \@for\@@ft:=\@@wphases\do{%wp-level work phases
996      \decode@wphase\@@ft
997      \@action\@@wp\@@line\wphase@start\wphase@len\wphase@force}
998    \@for\@@task:=\@@tasks\do{% tasks
999      \edef\@@wphases{\pdataref@safe{task}\@@task{wphases}}
1000     \@for\@@ft:=\@@wphases\do{%task-level work phases
1001       \decode@wphase\@@ft
1002       \@action\@@task\@@line\wphase@start\wphase@len\wphase@force}}}}
1003 \else% ifwork@areas false
1004 \edef\@@wps{\pdataref@safe{all}{wp}{ids}}
1005 \@for\@@wp:=\@@wps\do{% iterate over work packages
1006   \stepcounter{taskwps}
1007   \@@line=\gantt@wps\advance\@@line by -\thetaskwps
1008   \edef\@@tasks{\pdataref@safe\@@wp{task}{ids}}
1009   \node at (-1/\gantt@xscale,\@@line) [above=-2pt] {\pdataRef{wp}\@@wp{label}};
1010  \edef\@@wphases{\pdataref@safe{wp}\@@wp{wphases}}
1011  \@for\@@ft:=\@@wphases\do{%iterate over the wp-level work phases
1012    \decode@wphase\@@ft
1013    \@action\@@wp\@@line\wphase@start\wphase@len\wphase@force}
1014  \@for\@@task:=\@@tasks\do{% task-level work phases
1015    \edef\@@wphases{\pdataref@safe{task}\@@task{wphases}}
1016    \@for\@@ft:=\@@wphases\do{%iterate over the task-level work phases
1017      \decode@wphase\@@ft
1018      \@action\@@task\@@line\wphase@start\wphase@len\wphase@force}}
1019 \fi% ifwork@areas end
1020 \edef\@@deps{\pdataref@safe{all}{task}{deps}}
1021 \@for\@@dep:=\@@deps\do{%
1022   \@dependency{\pdataref@safe{taskdep}\@@dep{from}}{\pdataref@safe{taskdep}\@@dep{to}}}}
```

The next piece of code generates the effort sum table in draft mode

```
1023 \ifgantt@draft
1024     \newcounter{gantt@month}
1025     \newcount\@@e\newdimen\gantt@effort\newdimen\gantt@plus
1026     \@whilenum\thegantt@month<\prop@gen@months\do{% step over months
1027       \gantt@effort=0cm
1028       \ifwork@areas
1029       \edef\@@was{\pdataref@safe{all}{wa}{ids}}
1030       \@for\@@wa:=\@@was\do{% iterate over work areas
1031         \edef\@@wps{\pdataref@safe\@@wa{wp}{ids}}
1032         \@for\@@wp:=\@@wps\do{% iterate over work packages
1033           \edef\@@wphases{\pdataref@safe{wp}\@@wp{wphases}}
1034           \@for\@@ft:=\@@wphases\do{%iterate over the wp-level work phases
1035             \decode@wphase\@@ft
1036             \gantt@compute@effort\wphase@start\wphase@len\wphase@force}
1037           \edef\@@tasks{\pdataref@safe\@@wp{task}{ids}}
1038           \@for\@@task:=\@@tasks\do{% iterate over tasks
1039           \edef\@@wphases{\pdataref@safe{task}\@@task{wphases}}
1040           \@for\@@ft:=\@@wphases\do{%iterate over the wp-level work phases
1041             \decode@wphase\@@ft
1042             \gantt@compute@effort\wphase@start\wphase@len\wphase@force}}}}
1043     \fill[ganttgray]  (\thegantt@month,-5) rectangle +(1,\gantt@effort);
1044     \else% ifwork@areas
1045     \edef\@@wps{\pdataref@safe{all}{wp}{ids}}
1046     \@for\@@wp:=\@@wps\do{% iterate over work packages
1047         \edef\@@wphases{\pdataref@safe{wp}\@@wp{wphases}}
1048         \@for\@@ft:=\@@wphases\do{%iterate over the wp-level work phases
1049           \decode@wphase\@@ft
```

```
1050              \gantt@compute@effort\wphase@start\wphase@len\wphase@force}
1051          \edef\@@tasks{\pdataref@safe\@@wp{task}{ids}}
1052          \@for\@@task:=\@@tasks\do{% iterate over tasks
1053          \edef\@@wphases{\pdataref@safe{task}\@@task{wphases}}
1054          \@for\@@ft:=\@@wphases\do{%iterate over the wp-level work phases
1055              \decode@wphase\@@ft
1056              \gantt@compute@effort\wphase@start\wphase@len\wphase@force}}}
1057      \fill[ganttgray]   (\thegantt@month,-5) rectangle +(1,\gantt@effort);
1058      \fi% ifwork@areas
1059      \stepcounter{gantt@month}}
1060    \fi% ifgantt@draft
1061    \end{gantt}
1062    \caption{\gantt@caption}\label{fig:gantt}
1063 \end{figure}\footnotetext\gantt@footnote
```

now the multilingual support

```
1064 \newcommand\gantt@caption@main{Gantt Chart: Overview Work Package Activities}
1065 \newcommand\gantt@caption@lower{lower bar shows the overall effort \if@RAM (RM only) \fi per month}
1066 \newcommand\gantt@caption{\gantt@caption@main\ifgantt@draft\xspace
1067    -- \gantt@caption@lower\fi}
1068 \newcommand\gantt@footnote{Bars shown at reduced height (e.g. 50\%) indicate reduced
1069    intensity during  that work phase  (e.g. to 50\%).}
```

\gantttaskchart   This macro is a variant of \ganttchart, but it shows the tasks consecutively, as is useful for EU
    EdN:10   projects[10]

```
1070 \newcommand{\gantttaskchart}[1][]{\begin{figure}[hbtp]\centering\gantt@set{#1}
1071 \newcounter{gantt@all@tasks}%
1072 \setcounter{gantt@all@tasks}{\pdataref@num{all}{task}{count}}
1073 \addtocounter{gantt@all@tasks}{\pdataref@num{all}{wp}{count}}
1074 \begin{gantt}[#1]{\thegantt@all@tasks}
1075    \newcounter{gantt@tasks}\newcount\@@line
1076    \edef\@@wps{\pdataref@safe{all}{wp}{ids}}
1077    \@for\@@wp:=\@@wps\do{% iterate over work packages
1078        \stepcounter{gantt@tasks}
1079 %      \@action[white]{}\@@line0{48}1
1080        \edef\@@tasks{\pdataref@safe\@@wp{task}{ids}}
1081        \@for\@@task:=\@@tasks\do{% iterate over the tasks
1082          \stepcounter{gantt@tasks}
1083          \@@line=\thegantt@all@tasks\advance\@@line by -\thegantt@tasks
1084          \node at (-.5/\gantt@xscale,\@@line) [above=-2pt] {{\footnotesize\taskreflong\@@wp\@@task}};
1085          \edef\@@wphases{\pdataref@safe{task}\@@task{wphases}}
1086          \@for\@@ft:=\@@wphases\do{%iterate over the task-level work phases
1087            \decode@wphase\@@ft
1088            \@action\@@task\@@line\wphase@start\wphase@len\wphase@force
1089          }}}% end all iterations
1090      \end{gantt}
1091      \caption{\gantt@caption@main{} -- \emph{\gantt@footnote}}\label{fig:gantt}
1092 \end{figure}}
```

4.13   Coherence

\j*

```
1093 \newcommand\jpub{\textcolor{\prop@link@color}{\textbf{\Large{$\star$}}}}
1094 \newcommand\jpro{\textcolor{\prop@link@color}{\textbf{\Large{$\bullet$}}}}
1095 \newcommand\jsoft{\textcolor{\prop@link@color}{\textbf{@}}}
1096 \newcommand\jorga{\textcolor{\prop@link@color}{\textbf{\Large{$\circ$}}}}
```

---

[10]EDNOTE: this should be incorporated with the gantt chart above, but I am currently to scared to do it so close to
the deadline

1097 \newcommand\jsup{\textcolor{\prop@link@color}{\textbf{\smiley}}}

\add@joint   \add@joint{⟨first⟩}{⟨second⟩}{⟨sym⟩} adds ⟨sym⟩ to the the \coherence@⟨first⟩@⟨second⟩ macro for the coherence table.

1098 \newcommand\add@joint[3]{\@ifundefined{coherence@#1@#2}%
1099 {\@namedef{coherence@#1@#2}{#3}}%
1100 {\expandafter\g@addto@macro\csname coherence@#1@#2\endcsname{#3}}}

\prop@joint   This iterates over a comma-separated list of names and makes the necessary entries into the coherence table.

1101 \newcommand\prop@joint[2]{\@for\@first:=#2\do{%
1102 \@for\@second:=#2\do{\ifx\@first\@second\else\add@joint\@first\@second{#1}\fi}}}

\joint*   Now, some instances that use these.

1103 \newcommand\jointproj[1]{\prop@joint\jpro{#1}}
1104 \newcommand\jointpub[1]{\prop@joint\jpub{#1}}
1105 \newcommand\jointorga[1]{\prop@joint\jorga{#1}}
1106 \newcommand\jointsoft[1]{\prop@joint\jsoft{#1}}
1107 \newcommand\jointsup[1]{\prop@joint\jsup{#1}}

\coherencematrix

1108 \newcommand{\coherencematrix}{
1109 {\let\tabularnewline\relax\let\hline\relax\let\site\relax% so they do
1110 \let\@sw\relax\let\jpub\relax\let\jpro\relax\let\jorga\relax% not bother
1111 \let\jsoft\relax\let\jsup\relax\let\cellcolor\relax% us
1112 \gdef\@ct@head{}%
1113 \@for\@site:=\prop@gen@sites\do{\xdef\@ct@head{\@ct@head%
1114 &\ifx\cht@swsites\@true\@sw{\site{\@site}}\else\site{\@site}\fi}}%
1115 \gdef\@ct@lines{\@ct@head\tabularnewline\hline\hline} %initialize with head line
1116 \@for\@site:=\prop@gen@sites\do{\xdef\@ct@line{\site{\@site}}%
1117 \@for\@@site:=\prop@gen@sites\do{%
1118 \xdef\@ct@line{\@ct@line&\ifx\@site\@@site{\cellcolor{wagray}{}}\fi%
1119 \@ifundefined{coherence@\@site @\@@site}{}{\@nameuse{coherence@\@site @\@@site}}}}%
1120 \xdef\@ct@lines{\@ct@lines\@ct@line\tabularnewline\hline}}}%
1121 \begin{tabular}{|l||*{\the@site}{c|}}\hline%
1122 \@ct@lines\hline%
1123 joint&\multicolumn{\the@site}{l|}{\jpub $\hat=$ publication, \jpro $\hat=$ project,
1124 \jorga $\hat=$ organization, \jsoft $\hat=$ software/resource dev,
1125 \jsup $\hat=$ supervision}\\\hline
1126 \end{tabular}}

\coherencetable

1127 \define@key{coherencetable}{swsites}[true]{\def\cht@swsites{#1}}
1128 \define@key{coherencetable}{stretch}{\def\cht@stretch{#1}}
1129 \newcommand\coherencetable[1][]{%
1130 \def\cht@swsites{false}%
1131 \def\cht@stretch{1}%
1132 \setkeys{coherencetable}{#1}%
1133 \begin{table}[ht]\centering%
1134 \small\setlength{\tabcolsep}{.5em}%
1135 \renewcommand{\arraystretch}{\cht@stretch}%
1136 \coherencematrix%
1137 \caption{\coherence@caption}\label{tab:collaboration}
1138 \end{table}}

  now the multilinguality support

1139 \newcommand\coherence@caption{Previous Collaboration between {\pn} members}

4.14 Relevant Papers & References

We first define a bibLaTeX bibliography heading that does not create headers, we need it somewhere.

```
1140 \defbibheading{empty}{}
```

We define an internal macro that prints a publication list of a given bibTeX entry type and title for convenience. It also adds a `notype=` to the token register `\prop@rl` to deal with the unclassified entries from the list.

```
1141 \newif\if@allpapers\@allpaperstrue
1142 \newcommand\prop@ppl[3][]{\@allpapersfalse\message{ppl processing: #2}%
1143 \printbibliography[heading=subbibliography,type=#2,title=#3#1]%
1144 \@ifundefined{prop@rl}{\xdef\prop@rl{#2}}{\xdef\prop@rl{\prop@rl, #2}}}
```

The following code does not work yet, it would have been nice to be able to just add a key `unclassified` to catch the unclassified ones. I guess we just have to issue a warning instead.

```
1145 \newcommand\prop@prl[1]{\message{unclassified: #1}%
1146 \printbibliography[heading=subbibliography,title=Unclassified,#1]}%
1147 \define@key{paperlist}{unclassified}[true]{\message{unclass: \prop@rl}\prop@prl\prop@rl}
```

with this, we define a couple of keys that generate

```
1148 \define@key{paperlist}{articles}[true]{\prop@ppl{article}{Articles}}
1149 \define@key{paperlist}{chapters}[true]{\prop@ppl{inbook}{Book Chapters}}
1150 \define@key{paperlist}{confpapers}[true]{\prop@ppl[,keyword=conference]{inproceedings}{Conference Papers}}
1151 \define@key{paperlist}{wspapers}[true]{\prop@ppl[,notkeyword=conference]{inproceedings}{Workshop Papers}}
1152 \define@key{paperlist}{theses}[true]{\prop@ppl{thesis}{Theses}}
1153 \define@key{paperlist}{submitted}[true]{\prop@ppl[,keyword=submitted]{unpublished}{Submitted}}
1154 \define@key{paperlist}{books}[true]{\prop@ppl{book}{Monographs}}
1155 \define@key{paperlist}{techreports}[true]{\prop@ppl{techreport}{Technical Reports}}
```

featured We introduce a new bibLaTeX category `featured` for those papers that were already mentioned in `\prop@paperlist` and the macros defined from it.

```
1156 \DeclareBibliographyCategory{featured}
```

\prop@paperlist `\prop@paperlist{⟨keys⟩}{⟨refs⟩}` generates a paper list from a list ⟨keys⟩ of bibliography keys. It makes some local adaptions to the appearance of the bibliography, and then adds ⟨refs⟩ to the citable papers marks them as `featured`. Then it uses `\printbibliography` to make a bibliography of the cited papers. Note that these are not cited again in the main bibliography[11]

EdN:11

```
1157 \newcommand\prop@paperlist[2][]{%
1158 \let\biboldfont\bibfont%
1159 \renewcommand{\bibfont}{\footnotesize}%
1160 \renewcommand{\baselinestretch}{.9}%
1161 \nocite{#2}\def\do##1{\addtocategory{featured}{##1}}\docsvlist{#2}%
1162 \setkeys{paperlist}{#1}
1163 \@ifundefined{prop@rl}{}{\@latex@warning{some papers are not classified!}}
1164 \if@allpapers\printbibliography[category=featured,heading=empty]\fi%
1165 \let\bibfont\biboldfont}
```

We define the `warnpubs` heading constructor.

```
1166 \def\prop@warnpubs@message{Many of the proposers' publications are online at one of the following URIs:}
1167 \def\prop@warnpubs@title{References}
1168 \defbibheading{warnpubs}{\section*{\prop@warnpubs@title}%
1169   \@ifundefined{prop@gen@pubspages}
1170 {\@latex@warning{No publication pages specified;
1171                 use the pubspage key in the proposal environment!}}
1172   {\prop@warnpubs@message%
1173 \@for\@I:=\prop@gen@pubspages\do{\par\noindent\csname\@I\endcsname}}}
```

---

[11] EDNOTE: MK: we may want to make this optional controlled by a package option eventually.

Finally, we tweak bibLATEX to not give DOIs and URLS at the same time.

```
1174 \renewbibmacro*{event+venue+date}{}
1175 \renewbibmacro*{doi+eprint+url}{%
1176   \iftoggle{bbx:doi}
1177     {\printfield{doi}\iffieldundef{doi}{}{\clearfield{url}}}
1178     {}%
1179   \newunit\newblock
1180   \iftoggle{bbx:eprint}
1181     {\usebibmacro{eprint}}
1182     {}%
1183   \newunit\newblock
1184   \iftoggle{bbx:url}
1185     {\usebibmacro{url+urldate}}
1186     {}}
1187 ⟨/sty⟩
```

4.15   Miscellaneous

\signatures

```
1188 ⟨*pdata⟩
1189 \newcommand{\signatures}[1]{\section{#1}
1190 \qquad\number\day. \number\month. \number\year\\[6ex]
1191 \strut\qquad Date\hfill\@for\@p:=\prop@gen@PIs\do{%
1192 \wa@ref3{person}\@p{personaltitle}~\wa@ref3{person}\@p{name}\hfill}}
```

\@dmp   The \@dmp macro shows metadata information about the keys in the margin if \keystrue is specified. This is a debugging tool.

```
1193 \def\@dmp#1{\ifkeys\marginpar{#1}\fi}
```

\euro

```
1194 \renewcommand\euro{\officialeuro\xspace}
1195 ⟨/pdata⟩
```

References

[Koh16a] Michael Kohlhase. *Editorial Notes for LaTeX*. Self-documenting LaTeX package. Comprehensive TeX Archive Network (CTAN), 2016.

[Koh16b] Michael Kohlhase. *Preparing DFG Proposals and Reports in LaTeX with `dfgproposal.cls`*. Self-documenting LaTeX package. Comprehensive TeX Archive Network (CTAN), 2016. URL: http : / / mirror . ctan . org / macros / latex / contrib / proposal / dfg / dfgproposal.pdf.

[Koh16c] Michael Kohlhase. *`workaddress.sty`: An Infrastructure for marking up Dublin Core Metadata in LaTeX documents*. Self-documenting LaTeX package. Comprehensive TeX Archive Network (CTAN), 2016. URL: http://mirror.ctan.org/macros/latex/contrib/stex/sty/workaddress/workaddress.pdf.

[Lon] Brent Longborough. *gitinfo2.sty. A package for accessing metadata from the git dvcs*. URL: http://mirrors.ctan.org/macros/latex/contrib/gitinfo2/gitinfo2.pdf (visited on 10/26/2014).