

Preparing Proposals in L^AT_EX with `proposal.cls`*

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

September 21, 2017

Abstract

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

Contents

1	Introduction	3
2	The User Interface	3
2.1	Package Options	3
2.2	Proposal Metadata	4
2.3	Proposal Appearance	5
2.4	The proposal Environment and Title Page	5
2.5	Objectives	5
2.6	Work Areas and Work Packages	5
2.7	Tasks	6
2.8	Work Phase Metadata	6
2.9	Milestones and Deliverables	7
2.10	Project Data, Referencing, and Hyperlinking	7
2.11	The Work Package Table	8
2.12	Gantt Charts	8
2.13	Coherence	9
2.14	Localization	9
2.15	Project Management	9
3	Limitations and Enhancements	9
4	The Implementation	11
4.1	Package Options and Format Initialization	11
4.2	Proposal Metadata	13
4.3	Proposal Appearance	14
4.4	The proposal Environment and Title Page	14
4.5	Objectives	16
4.6	Work Areas and Work Packages	17
4.7	Tasks	21
4.8	Work Phase Metadata	23
4.9	Milestones and Deliverables	23
4.10	Project Data, Referencing & Hyperlinking	27

*Version ? (last revised ?)

4.11 The Work Package Table	28
4.12 Gantt Charts	33
4.13 Coherence	36
4.14 Relevant Papers & References	38
4.15 Miscellaneous	39

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 repository 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`, `numericcites`, and `public`.

<code>submit</code>	The <code>submit</code> option will disable various proposal management decorations which are enabled by default for submission.
<code>noworkareas</code>	The <code>noworkareas</code> option specifies that we do not want to structure our work plan into work areas (see section 2.6).
<code>RAM</code>	The <code>RAM</code> option specifies that we specify research assistant months in the effort tallies (see section 2.6).
<code>deliverables</code>	The <code>deliverables</code> 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.
<code>wpsubsection</code>	The <code>wpsubsection</code> option specifies that we want to see subsections headings for the WPs (and WAs, if we have them).
<code>report</code>	The <code>report</code> option specifies that we want to use the <code>report.cls</code> class as a basis for <code>proposal</code> instead of the default <code>article.cls</code> .
<code>keys</code>	The <code>keys</code> option specifies that we want to see the values of various keyval arguments in the

margin.

svninfo The **svninfo** option 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 **[gitinfo2:on]** for this to work.

numericcites The **numericcites** option changes citations to numeric from the default **alphabetic**.

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 L^AT_EX 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** • **acronym** for the proposal acronym, possibly accompanied by an **acrolong** that explains it.
- acrolong** The acronym will also be used in the page headings.
- start** • **start** for the start date of the proposed fragment of the project, and **months** for the length
- months** of the proposal in months. Both have to be specified for the **proposal** class to work.
- since** • If the proposal only concerns a part of a longer-running project, the **since** key allows to
- fundsuntil** 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 **[Kohlhase:workaddress:ctan]** 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.
- Sometimes it makes sense to document the proposal number in the metadata, e.g. to use the generated metadata file **\langle main \rangle.pdata** for project reports. The **proposalnumber** can be used for that.
- \pn** If the **acronym** and **acrolong** are given, then they automatically define the macros **\pn** and
- \pnlong** **\pnlong** which allow to use the project acronym (project **urname**) and its long version in the text. Note that these macros use **\xspace** internally, 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, or 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` supports both of these. Support for the first is configured via the `topdownPM` key and for the other via the `botupPM` key. They add respective lines for planning in the WA/WP figure (see 2.6).

2.3 Proposal Appearance

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

- If the `compactht` key is given (it does not need a value), then the header tables² are made 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.

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. 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 objectives can be referenced via `\OBJref{<id>}` by their label and via `\OBJtref{<id>}` by label 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` macro) and the Gantt Chart (see Section 2.12).

To achieve the automation, work plan is marked up by the `workplan` environment, which sets up various internal counters and bookkeeping macros. It contains texts and `workpackage` environments for the work packages.

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:

- The `id` key is used to specify a label for cross-referencing the work package or work area, it

¹EdNOTE: move the RAM, wpsectionheadings,... options here.

²EdNOTE: describe them somewhere and reference here

³EdNOTE: add documentation

	must be document-unique.
<code>title</code>	• The <code>title</code> and <code>short</code> keys are used for the work package/group title. The short title is used
<code>short</code>	in tables and should not be longer than 15 characters.
<code>wphases</code>	• The <code>wphases</code> key is used according to Section 2.8
<code>requires</code>	• The <code>requires</code> key can be used to mark, up dependencies between tasks. If <code>requires=\taskin{<rid>}{<wp>}</code> is given in a task with <code>id=<t></code> , then task <code><rid></code> in work package <code><wp></code> must be completed for task <code><t></code> to become possible. This key will draw an arrow into the gantt chart from the end of task <code><rid></code> to <code><t></code> . 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.
<code>RM</code>	• In single-site proposals, the <code>RM</code> (and <code>RAM</code> if the <code>RAM</code> 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. <code>RM</code> is used for “researcher months” (wissenschaftlicher Mitarbeiter) and <code>RAM</code> for “research assistant months” (wissenschaftliche Hilfskraft).
<code>*RM</code>	• In multi-site proposals, the <code>proposal</code> package generates the keys <code><site>RM</code> (and <code><site>RAM</code>) where <code><site></code> is any site label declared via the <code>site</code> key in the top-level <code>proposal</code> 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 <code>L^AT_EX</code> twice for this)).
<code>*RAM</code>	
<code>lead</code>	• In multi-site proposals the <code>lead</code> key specifies the work package or work area lead, the value of this feature should be the short name of the respective partner.
<code>swsites</code>	• For work packages with many prospers the <code>swsites</code> key can be given (no value needed) to turn the site names sideways to conserve (horizontal) space.
<code>workarea</code>	It is often useful to group the work packages in a proposal further (especially for larger, collaborative proposals). This can be done via the <code>workarea</code> environment, which groups work packages. This environment takes the same keys as the <code>workpackage</code> environment, except for the efforts, which can be computed automatically from the work packages it groups.
	As the author of the <code>proposal</code> class likes more structured proposals, using work areas is the default, but the <code>proposal</code> class can also be used with the <code>noworkareas</code> option for less structured (smaller) proposals.

2.7 Tasks

<code>tasklist</code>	In the work packages we can list tasks that need to be undertaken with the <code>tasklist</code> environment.
<code>task</code>	The individual tasks are marked up with the <code>task</code> environment. This takes a keyval argument with the keys <code>id</code> for identification, <code>title</code> for a title, and the workphase keys <code>wphases</code> , <code>start</code> , <code>end</code> , and <code>force</code> (see Section 2.8). For planning involvement we can specify the overall person months via the <code>PM</code> key, the task lead via <code>lead</code> , and the partners involved via the <code>partners</code> key. Instead of just listing the partners, we can also specify the contributions of the partners with <code>RM<site></code> and <code>RAM<site></code> keys. Finally task dependencies can be specified via the <code>requires</code> key.
<code>\taskref</code>	Tasks can be referenced by the <code>\taskref</code> 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
<code>\tasktref</code>	variant with work package title: <code>\tasktref</code> . Finally, <code>\localtaskref</code> references a task in the local
<code>\localtaskref</code>	work package by the identifier in its argument.

2.8 Work Phase Metadata

<code>wphases</code>	The <code>task</code> and <code>workpackage</code> allow the <code>wphases</code> 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 <code><start>-<end></code> or <code><start>-<end>!<force></code> , where <code><start></code> and <code><end></code> delimit the run time of the work phase and the optional <code>!<force></code> 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 table¹, which we markup up with the `milestones` environment that takes care of initialization and numbering issues. This contains a list of 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 be customized by redefining `\milestone@label` and referenced by the `\mileref{⟨id⟩}` and `\miletreref{⟨id⟩}` for a reference with milestone title. `\pdatacount{all}{miles}` gives the number of milestones.

wpdelivs 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 an environment `wpdelivs` that contains the deliverable descriptions. These are marked up via 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. The `status` key gives the status of the deliverable. If it has the value `canceled`, then the deliverable is grayed out and it is not mentioned in the deliverables table given by `\inputdelivs` below.

\deliv@label Deliverables are numbered by labels whose shape can be customized by number, where the shape of the label can be specified by redefining `\deliv@label` and referenced by `\delivref{⟨wp⟩}{⟨id⟩}` where `⟨wp⟩` is the work package identifier and `⟨id⟩` that if the deliverable and `\delivtreref{⟨wp⟩}{⟨id⟩}` for a reference with title. `\localdelivref` can be used to reference deliverables in the same work 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).

\inputdelivs 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 with the `\inputdelivs` macro. This works index generation in L^AT_EX. 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. Finally, the `issue` key can be used to bind the deliverable to an issue identifier in a project management system.

wadelivs In some proposals, also work areas can have deliverables, then the above hold analogously for `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 be referenced via the `\pdataref` macro, which takes three arguments.

¹this 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.

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 `WP n` where n is the work package number for referencing), `title` (the work package title), `lead` 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` `\pdataRefFB` The `\pdataRef` and `\pdataRefFB` macros are variant of `\pdataref` and `\pdatarefFB` that also 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 areas. The `\WPref` macro takes a work package identifier as an argument and makes a reference: `\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}`. Unless the `noworkareas` macro is set, we also have the variants `\WAref` and `\WAtref` for work areas.

2.11 The Work Package Table

`\wpfig` One of the most useful features of the `proposal` class is that we can generate an overview table for 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` Gantt charts are used in proposals to show the distribution of activities in work packages over time. A gantt chart is represented by the `gantt` environment that takes a 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 page. The `step` key allows to specify the steps (in months) of the vertical auxiliary lines. Finally, the `draft` key specifies that plausibility checks (that can be expensive to run) are carried out. 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 L^AT_EX 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 offers 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`, `\jointsub`, `\jointsoft`, and `\jointsup` macros before. These macros all take a comma-separated list of site identifiers as an argument. Use for instance `\jointproj{a,b,c}` to specify that the sites with the identifiers `a`, `b` and `c` have a joint project. `\coherencetable` is a variant which packages the coherence table in a table figure with label `tab:collaboration`.

`\coherencetable` The symbols used can be configured by redefining `\jpub`, `\jproj`, and `\jorga`, `\jsoft`, and `\jsup`.

`\jointpub`
`\jointsub`
`\jointorga`
`\jointsoft`
`\jointsup`
`\coherencetable`
`\jpub`
`\jproj`
`\jorga`
`\jsoft`
`\jsup`

2.14 Localization

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 [`Kohlhase:pdrp:ctan`] provides an option `german` for german-language proposals and project reports that triggers a redefinition of all of these macros at read time.

2.15 Project Management

Much of the metadata that is explicitly represented in proposals written with the `proposal` class is very useful for project management. For instance, it is possible to use the metadata in the `<main>.pdata` file to generate issues for all the tasks, work packages, and deliverables automatically. The L^AT_EX-`proposal` repository [`LaTeX-proposal:git`] contains an experimental script that automates that. After that, we can cross-reference them using the `issue` key to get extra mileage⁴

`issue`
EdN:4

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.

⁴EdNOTE: MK: how to use this?

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 PIs to `sites`, 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

The author is indebted to Jake Hartenstein, Christoph Lange, Florian Rabe, Lutz Schröder, and Tsanko Tsankov for error reports, feature suggestions, and code snippets.

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 \newif\if@numericcites\@numericcitesfalse
13 \def\proposal@class{article}
14 \DeclareOption{wpsubsection}{\@wpsubsectiontrue}
15 \DeclareOption{submit}{\submittrue}
16 \DeclareOption{grantagreement}{\grantagreementtrue}
17 \DeclareOption{gitinfo}{\@gitinfotrue}
18 \DeclareOption{numericcites}{\@numericcitesttrue}
19 \DeclareOption{svninfo}{\@svninfotrue}
20 \DeclareOption{public}{\publictrue}
21 \DeclareOption{noworkareas}{\work@areasfalse\PassOptionsToClass{\CurrentOption}{pdata}}
22 \DeclareOption{RAM}{\@RAMtrue}
23 \DeclareOption{report}{\def\proposal@class{report}}
24 \DeclareOption{keys}{\keystrue}
25 \DeclareOption{deliverables}{\delivstrue}
26 \DeclareOption*{\PassOptionsToClass{\CurrentOption}{article}}
27 \ProcessOptions
28 \LoadClass[a4paper,twoside]{\proposal@class}
29 \RequirePackage{proposal}
30 </cls | reporting>

```

EdN:5

5

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

```

31 (*sty)
32 \RequirePackage{amssymb}
33 \RequirePackage{wasysym}
34 \RequirePackage{url}
35 \RequirePackage{graphicx}
36 \RequirePackage{colortbl}
37 \RequirePackage{xcolor}
38 \RequirePackage{rotating}
39 \RequirePackage{fancyhdr}
40 \RequirePackage{array}
41 \RequirePackage{xspace}
42 \RequirePackage{comment}
43 \AtBeginDocument{\ifpublic\excludacomment{private}\fi}
44 \RequirePackage{tikz}
45 \RequirePackage{paralist}
46 \RequirePackage[a4paper,margin=18mm]{geometry}

```

⁵EdNOTE: We should probably try to move all the grantagreement stuff into the euproposal class.

```

47 \RequirePackage{boxedminipage}
48 % so that ednotes in wps do not run out of symbols
49 \renewcommand{\thempfootnote}{\roman{mpfootnote}}
50 \renewcommand{\familydefault}{\sfdefault}
51 \RequirePackage[scaled=.90]{helvet}
52 \RequirePackage{textcomp}
53 \if@numericcites
54 \RequirePackage[style=numeric,hyperref=auto,defernumbers=true,backend=bibtex,firstinits=true,maxbibnames=9,ma
55 \else
56 \RequirePackage[style=alphabetic,hyperref=auto,defernumbers=true,backend=bibtex,firstinits=true,maxbibnames=9
57 \fi
58 \RequirePackage{csquotes}
59 \RequirePackage{mdframed}

```

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

```

60 \RequirePackage{pdata}
61 \definecolor{darkblue}{rgb}{0,0,.7}
62 \ifsubmit\def\prop@link@color{darkblue}\else\def\prop@link@color{blue}\fi
63 \RequirePackage[bookmarks=true,linkcolor=\prop@link@color,
64 citecolor=\prop@link@color,urlcolor=\prop@link@color,colorlinks=true,
65 breaklinks=true, bookmarksopen=true]{hyperref}

```

the `ed` package [[Kohlhase:ed:ctan](#)] 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.

```

66 \ifsubmit
67 \RequirePackage[hide]{ed}
68 \if@svninfo\RequirePackage[final,today]{svninfo}\fi
69 \else
70 \RequirePackage[show]{ed}
71 \if@svninfo\RequirePackage[eso-foot,today]{svninfo}\fi
72 \if@gitinfo\RequirePackage[mark]{gitinfo2}\fi
73 \fi
74 \renewcommand\ednotesshape{\sl\footnotesize}

```

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

```

75 \ifpublic\excludecomment{private}\else\includecomment{private}\fi

```

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

```

76 \setcounter{secnumdepth}{3}

```

We specify the page headings.

```

77 \let\prop@gen@acronym\@empty
78 \newif\ifofpage\ofpagefalse
79 \ifgrantagreement
80 \fancyhead{}
81 \renewcommand{\headrulewidth}{0pt}
82 \renewcommand{\footrulewidth}{0.4pt}
83 \else
84 \fancyhead[RE,L0]{\ifx\prop@gen@acronym\@empty\else\prop@gen@acronym\fi}
85 \fancyhfoffset{0pt}
86 \fi
87 \fancyfoot[C]{}
88 \newcommand\prop@of@pages[2]{page~\#1\ifofpage~of~\#2\fi}
89 \ifgrantagreement
90 \fancyfoot[L]{\prop@gen@proposalnumber%
91 \ifx\prop@gen@acronym\@empty\else\quad \prop@gen@acronym\fi\quad --\quad Part B}

```

```

92 \fancyfoot[R]{\thepage}
93 \else
94 \fancyhead[LE,R0]{\prop@of@pages\thepage{\pdataref@num{prop}{page}{last}}}
95 \fi
96 \pagestyle{fancyplain}
97 \endsty

```

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.

```

98 (*pdata)
99 \RequirePackage{workaddress}[2016/07/06]
100 \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.

```

101 \newif\if@sites\@sitesfalse\let\prop@gen@sites=\relax%
102 \newcounter{@site}%
103 \define@key{prop@gen}{site}{\@sitestrue\@dmp{site=#1}%
104 \stepcounter{@site}\pdata@def{site}{#1}{number}{\the@site}%
105 \@ifundefined{prop@gen@sites}{\xdef\prop@gen@sites{#1}}{\xdef\prop@gen@sites{\prop@gen@sites,#1}}%
106 \define@key{prop@gen}{#1RM}{\pdata@def{site}{#1}{intendedRM}{##1}}%
107 \if@RAM\define@key{prop@gen}{#1RAM}{\pdata@def{site}{#1}{intendedRAM}{##1}}\fi
108 \define@key{workpackage}{#1RM}{\pdata@def{wp@id}{#1}{RM}{##1}}%
109 \if@RAM\define@key{workpackage}{#1RAM}{\pdata@def{wp@id}{#1}{RAM}{##1}}\fi
110 \define@key{task}{#1RM}{\pdata@def{wp@id @\task@id}{#1}{RM}{##1}}%
111 \if@RAM\define@key{task}{#1RAM}{\pdata@def{wp@id @\task@id}{#1}{RAM}{##1}}\fi
112 \define@key{prop@gen}{#1employed}{\let\tabularnewline\relax\let\hline\relax\let\wa@ref\relax%
113 \@ifundefined{prop@gen@employed@lines}%
114 {\xdef\prop@gen@employed@lines{\wa@ref3{institution}{#1}{shortname} & ##1\tabularnewline\hline}}%
115 {\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.

```

116 \define@key{prop@gen}{RM}{\@dmp{RM=#1}\if@sites%
117 \PackageWarning{Do not use the RM key in the presence of sites}\else%
118 \pdata@def{all}{intended}{RM}{#1}\fi}
119 \define@key{prop@gen}{RAM}{\@dmp{RAM=#1}\if@sites%
120 \PackageWarning{Do not use the RAM key in the presence of sites}\else%
121 \pdata@def{all}{intended}{RAM}{#1}\fi}

```

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

```

122 \define@key{prop@gen}{PI}{\@dmp{PI=#1}%
123 \@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`.

```

124 \define@key{prop@gen}{pubspage}{\@ifundefined{prop@gen@pubspages}%
125 {\xdef\prop@gen@pubspages{#1}}{\xdef\prop@gen@pubspages{\prop@gen@pubspages,#1}}}

```

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

```

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

```

The rest of the keys just store their value.

```

127 \define@key{prop@gen}{instrument}{\def\prop@gen@instrument{#1}%

```

```

128 \pdata@def{prop}{gen}{instrument}{#1}\@dmp{inst=#1}}
129 \define@key{prop@gen}{title}{\def\prop@gen@title{#1}%
130 \pdata@def{prop}{gen}{title}{#1}}
131 \define@key{prop@gen}{acronym}{\gdef\prop@gen@acronym{#1}%
132 \pdata@def{prop}{gen}{acronym}{#1}\@dmp{acro=#1}}
133 \define@key{prop@gen}{acrolong}{\def\prop@gen@acrolong{#1}%
134 \pdata@def{prop}{gen}{acrolong}{#1}}
135 \define@key{prop@gen}{proposalnumber}{\def\prop@gen@proposalnumber{#1}%
136 \pdata@def{prop}{gen}{proposalnumber}{#1}}
137 \define@key{prop@gen}{discipline}{\def\prop@gen@discipline{#1}%
138 \pdata@def{prop}{gen}{discipline}{#1}}
139 \define@key{prop@gen}{areas}{\def\prop@gen@areas{#1}%
140 \pdata@def{prop}{gen}{areas}{#1}}
141 \define@key{prop@gen}{start}{\def\prop@gen@start{#1}%
142 \pdata@def{prop}{gen}{start}{#1}}
143 \define@key{prop@gen}{months}{\def\prop@gen@months{#1}%
144 \pdata@def{prop}{gen}{months}{#1}}
145 \define@key{prop@gen}{since}{\def\prop@gen@since{#1}%
146 \pdata@def{prop}{gen}{since}{#1}}
147 \define@key{prop@gen}{totalduration}{\def\prop@gen@totalduration{#1}%
148 \pdata@def{prop}{gen}{totalduration}{#1}}
149 \define@key{prop@gen}{fundsuntil}{\def\prop@gen@fundsuntil{#1}%
150 \pdata@def{prop}{gen}{fundsuntil}{#1}}
151 \define@key{prop@gen}{topdownPM}[true]{\def\prop@gen@topdownPM{#1}}
152 \define@key{prop@gen}{botupPM}[true]{\def\prop@gen@botupPM{#1}}
153 \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.

```

154 \newcommand\prop@gen@acro@default{ACRONYM}
155 \def\prop@gen@acro{\prop@gen@acro@default}
156 \newcommand\prop@gen@months@default{???months???}
157 \def\prop@gen@months{\prop@gen@months@default}
158 \newcommand\prop@gen@title@default{???Proposal Title???}
159 \def\prop@gen@title{\prop@gen@title@default}
160 \newcommand\prop@gen@instrument@default{??? Instrument ???}
161 \def\prop@gen@instrument{\prop@gen@instrument@default}

```

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

```

162 \newcommand\prop@tl[2]{\xdef\tab@line{
163 \for\tl@ext:={#1}\do{\xdef\tab@line{\tab@line&#2}}
164 \tab@line}

```

4.3 Proposal Appearance

We define the keys for the proposal appearance

```

165 \def\prop@gen@compactht{false}
166 \define@key{prop@gen}{compactht}[true]{\def\prop@gen@compactht{#1}}
167 \pdata

```

`emphbox`

```

168 (*sty)
169 \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.

```

170 \newenvironment{prop@proposal}
171 {\thispagestyle{empty}}%
172 \begin{center}
173   {\LARGE \prop@gen@instrument}\[.2cm]
174   {\LARGE\textbf{\prop@gen@title}}\[.3cm]
175   \ifx\prop@gen@acronym\empty\else{\LARGE Acronym: {\prop@gen@acronym}}\[.2cm]\fi
176   {\large\today}\[1em]
177   \begin{tabular}{c*{\the@PIs}{c}}
178     \prop@tl\prop@gen@PIs{\wa@ref3{person}\tl@ext{name}}\\
179     \prop@tl\prop@gen@PIs{\wa@ref3{institution}\wa@ref3{person}\tl@ext{affiliation}}{name}}
180   \end{tabular}\[2cm]
181 \end{center}
182 \setcounter{tocdepth}{2}\tableofcontents\newpage\setcounter{page}{1}}

Now we come to the end of the environment:

183 {\section{List of Attachments}
184 \begin{itemize}
185 \@for\@I:=\prop@gen@PIs\do{%
186 \item Curriculum Vitae and list of publications for
187   \wa@ref3{person}\@I{personaltitle} \wa@ref3{person}\@I{name}}
188 \end{itemize}\newpage
189 \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.

```

190 \newenvironment{proposal}[1][\readpdata\jobname
191 \ofpagetrue\setkeys{prop@gen}{#1}
192 \pdata@open\jobname
193 \if@sites\else
194 \define@key{workpackage}{RM}{\pdata@def{wp}\wp@id{RM}{##1}\@dmp{RM=##1}}
195 \if@RAM\define@key{workpackage}{RAM}{\pdata@def{wp}\wp@id{RAM}{##1}\@dmp{RAM=##1}}\fi
196 \define@key{task}{RM}{\pdata@def{task}\wp@id @\task@id{RM}{##1}\@dmp{RM=##1}}
197 \if@RAM\define@key{task}{RAM}{\pdata@def{wp}\wp@id @\task@id{RAM}{##1}\@dmp{RAM=##1}}\fi
198 \fi
199 \newcounter{@PIs}
200 \@ifundefined{prop@gen@PIs}{\@for\@I:=\prop@gen@PIs\do{\stepcounter{@PIs}}}
201 \newcounter{@sites}
202 \@ifundefined{prop@gen@sites}{\@for\@I:=\prop@gen@sites\do{\stepcounter{@sites}}}
203 \setcounter{page}{0}
204 \begin{prop@proposal}

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

205 {\end{prop@proposal}
206 \pdata@def{prop}{page}{last}{\thepage}\ofpagefalse
207 \pdata@close}
208 \end{sty}

```

The `report` environment is similar, but somewhat simpler

report

```

209 (*reporting)
210 \newif\if@report\@reportfalse
211 \newenvironment{report}[1][\%
212 {\@reporttrue\readpdata\jobname%
213 \ofpagetrue\setkeys{prop@gen}{#1}%

```

```

214 \pdata@open\jobname%
215 \@ifundefined{prop@gen@PIs}{\newcounter{@PIs}\@for\@I:=\prop@gen@PIs\do{\stepcounter{@PIs}}}%
216 \@ifundefined{prop@gen@sites}{\newcounter{@sites}\@for\@I:=\prop@gen@sites\do{\stepcounter{@sites}}}%
217 \setcounter{page}{0}%
218 \begin{prop@report}}
219 {\end{prop@report}}%
220 \pdata@def{prop}{page}{last}{\thepage}\ofpagefalse\newpage
221 \printbibliography[heading=warnpubs]
222 \pdata@close}

```

prop@report

```

223 \newenvironment{prop@report}
224 {\begin{center}
225   {\LARGE Final Project Report}\[\.2cm]
226   {\LARGE\textbf{\prop@gen@title}}\[\.3cm]
227   \ifx\prop@gen@acronym\@empty\else{\LARGE Acronym: {\prop@gen@acronym}}\[\.2cm]\fi
228   {\large\today}\[1em]
229   \begin{tabular}{c*{\the@PIs}{c}}
230     \prop@tl\prop@gen@PIs{\wa@ref3{person}\tl@ext{name}}\
231     \prop@tl\prop@gen@PIs{\wa@ref3{institution}\wa@ref3{person}\tl@ext{affiliation}}{name}}
232   \end{tabular}\[2cm]
233 \end{center}
234 \setcounter{tocdepth}{2}\tableofcontents\newpage\setcounter{page}{1}}
235 {}
236 \reporting}

```

\site*

```

237 (*sty)
238 \newcommand\site[1]{\hyperlink{site@#1@target}{\wa@ref3{institution}{#1}{acronym}}}
239 \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

```

240 \newcommand\objective@label[1]{0#1}

```

We define the keys for the objectives environment

```

241 \define@key{obj}{id}{\def\obj@id{#1}\@dmp{id=#1}}
242 \define@key{obj}{title}{\def\obj@title{#1}\@dmp{title=#1}}
243 \define@key{obj}{short}{\def\obj@short{#1}\@dmp{short=#1}}

```

And a counter for numbering objectives

```

244 \newcounter{objective}

```

objective

```

245 \newenvironment{objective}[1]{}
246 {\let\obj@id\relax\let\obj@title\relax\let\obj@short\relax%
247 \setkeys{obj}{#1}\stepcounter{objective}%
248 \goodbreak\smallskip\par\noindent%
249 \textbf{\objective@label{\arabic{objective}}}%
250 ~\pdata@target{obj}{\obj@id}{\pdata@ref{obj}{\obj@id}{title}}\ignorespaces}%
251 \pdata@def{obj}{\obj@id}{label}{\objective@label\theobjective}%
252 \@ifundefined{obj@title}{\pdata@def{obj}{\obj@id}{title}\obj@title}%
253 \@ifundefined{obj@short}{\pdata@def{obj}{\obj@id}{short}\obj@short}}
254 {}

```


\OBJref

```
255 \newcommand\OBJref[1]{\pdataRef{obj}{#1}{label}}
256 \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).

```
257 \ifwork@areas
258 \define@key{workarea}{id}{\def\wa@id{#1}\@dmp{id=#1}}
259 \define@key{workarea}{title}{\pdata@def{wa}\wa@id{title}{#1}}
260 \define@key{workarea}{short}{\pdata@def{wa}\wa@id{short}{#1}}
261 \define@key{workarea}{lead}{\pdata@def{wa}\wa@id{lead}{#1}}
262 \fi
```

work packages have similar ones.

```
263 \define@key{workpackage}{id}{\def\wp@id{#1}\@dmp{id=#1}}
264 \define@key{workpackage}{title}{\pdata@def{wp}\wp@id{title}{#1}}
265 \define@key{workpackage}{short}{\pdata@def{wp}\wp@id{short}{#1}}
266 \define@key{workpackage}{lead}{\pdata@def{wp}\wp@id{lead}{#1}\def\wp@lead{#1}\@dmp{lead=#1}}
267 \define@key{workpackage}{type}{\def\wp@type{#1}\pdata@def{wp}\wp@id{type}{#1}}
268 \define@key{workpackage}{wphases}{\def\wp@wphases{#1}\pdata@def{wp}\wp@id{wphases}{#1}}
269 \define@key{workpackage}{swsites}[true]{\def\wp@swsites{#1}}
```

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

```
270 \newcommand\wp@mk@title[1]{Work Package {#1}}
271 \newcommand\wp@label[1]{WP{#1}}
272 \ifwork@areas
273 \newcommand\wa@label[1]{WA{#1}}
274 \newcommand\wa@mk@title[1]{Work Area {#1}}
275 \fi
```

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

```
276 \ifwork@areas\newcounter{wa}\newcounter{wp}[wa]\else\newcounter{wp}\fi
277 \ifdelivs\newcounter{deliv}[wp]\fi
278 \newcounter{allwp}
```

EdN:6 \update@* 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.⁶

```
279 \newcommand\update@wps[1]{\@ifundefined{wps}{\xdef\@wps{#1}}{\xdef\@wps{\@wps,#1}}}
280 \newcommand\update@tasks[1]{\@ifundefined{tasks}{\xdef\@tasks{#1}}{\xdef\@tasks{\@tasks,#1}}}
281 \newcommand\update@deps[1]{\@ifundefined{task@deps}{\xdef\@task@deps{#1}}{\xdef\@task@deps{\@task@deps,#1}}}
282 \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 $\langle force \rangle$.

```
283 \newcommand\decode@wphase[1]{\expandafter\decode@p@start#1@%
284 \local@count\wphase@end\advance\local@count by -\wphase@start%
285 \def\wphase@len{\the\local@count}}
286 \def\decode@p@start#1-#2@{\def\wphase@start{#1}\decode@p@end#2!@}
```

⁶EdNOTE: 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`

```

287 \def\decode@p@end#1!#2@{\def\wphase@end{#1}\def\@test{#2}%
288 \ifx\@test\empty\def\wphase@force{1}\else\decode@p@force#2\fi}
289 \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`

```

290 \def\wphases@start#1-#2@{\def\wphase@start{#1}}
291 \newcommand\startend@wphases[1]{\def\@test{#1}
292 \ifx\@test\empty\def\wphase@start{0}\def\wphase@end{0}\else%
293 \@for\@I:=#1\do{\expandafter\decode@p@start\@I @}
294 \expandafter\wphases@start#1@\fi}

```

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

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`.

```

295 \newcounter{wp@RM}
296 \if@RAM\newcounter{wp@RAM}\fi
297 \newenvironment{work@package}[1] []%
298 {\def\wp@wphases{0-0}% default values
299 \def\wp@swsites{false}
300 \setkeys{workpackage}{#1}\stepcounter{wp}\stepcounter{allwp}%
301 \startend@wphases\wp@wphases%
302 \pdata@def{wp}\wp@id{start}\wphase@start\pdata@def{wp}\wp@id{end}\wphase@end%
303 \@ifundefined{wp@type}{\pdata@def{wp}\wp@id{type}\wp@type}%
304 \let\@tasks=\relax%
305 \edef\wp@num{\ifwork@areas\thewa.\fi\thewp}%
306 \pdata@def{wp}\wp@id{label}\wp@label\wp@num}%
307 \pdata@def{wp}\wp@id{number}\thewp}%
308 \pdata@def{wp}\wp@id{page}\thepage}%
309 \update@wps\wp@id%
310 \edef\wp@num{\ifwork@areas\thewa.\fi\thewp}%
311 \pdata@def{wp}\wp@id{num}\thewp}%

```

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

```

312 \if@sites%
313 \setcounter{wp@RM}{0}\if@RAM\setcounter{wp@RAM}{0}\fi%
314 \@for\@site:=\prop@gen@sites\do{%
315 \edef\@RM{\pdataref@num\wp@id\@site{RM}}\addtocounter{wp@RM}{\@RM}%
316 \if@RAM\edef\@RAM{\pdataref@num\wp@id\@site{RAM}}\addtocounter{wp@RAM}{\@RAM}\fi}
317 \pdata@def{wp}\wp@id{RM}\thewp@RM}%
318 \if@RAM\pdata@def{wp}\wp@id{RAM}\thewp@RAM}\fi%
319 \fi}% \if@sites
320 {\@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

```

321 \newenvironment{workpackage}[1] []%
322 {\begin{work@package}[#1]%
323 \ifgrantagreement\else%
324 %\if@wpsubsection\subsection*{\wp@mk@title\thewp}: \pdataref{wp}\wp@id{title}}\fi
325 \if@sites\goodbreak\medskip\wpheadertable%
326 \else\subsection*{\wp@title} (\wprm)}\fi%
327 \addcontentsline{toc}{paragraph}{\wp@mk@title\thewp}: \pdataref{wp}\wp@id{title}}%
328 \noindent\ignorespaces%
329 \fi}
330 {\end{work@package}}

```

EdN:7\wptitle 7

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

EdN:8 \wprm 8

```
332 \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` $\neq 0$)

```
333 \newcount\site@contribution%
334 \newcommand\if@site@contributes[2]{%
335 \ifx\prop@gen@compactht\@true
336 \if@RAM\ifnum\pdataref@num\wp@id{#1}{RM} > 0 \ifnum \pdataref@num\wp@id{#1}{RAM} > 0 #2\fi\fi
337 \else\ifnum\pdataref@num\wp@id{#1}{RM} > 0 #2\fi\fi
338 \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` (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.

```
339 \newcounter{wp@sites@num}
340 \newcommand\wp@sites@efforts@lines{%
341 \setcounter{wp@sites@num}{0}
342 {\let\G@refundefinedtrue=\relax\let\@latex@warning=\relax\let\@sw\relax%
343 \let\site\relax\let\textbf\relax\let\sum\style\relax\let\lead\style\relax%
344 \let\pn\relax\let\sys\relax%
345 \xdef\wp@sites@line{\wp@legend@site}\xdef\wp@efforts@line{\wp@legend@effort}%initialize lines
346 \@for\@site:=\prop@gen@sites\do{\if@site@contributes\@site{\stepcounter{wp@sites@num}}%
347 \xdef\wp@sites@line{\wp@sites@line%
348 \if@site@contributes\@site{&%
349 \ifx\wp@swsites\@true%
350 \@sw{\ifx\@site\wp@lead\lead@style{\site{\@site}}\else\site{\@site}\fi}%
351 \else\ifx\@site\wp@lead\lead@style{\site{\@site}}\else\site{\@site}\fi}%
352 \fi}}%
353 \xdef\wp@efforts@line{\wp@efforts@line%
354 \if@site@contributes\@site{&%
355 \ifx\@site\wp@lead%
356 \lead@style{\pdataref@safe\wp@id\@site{RM}\if@RAM+\pdataref@safe\wp@id\@site{RAM}\fi}
357 \else\pdataref@safe\wp@id\@site{RM}\if@RAM+\pdataref@safe\wp@id\@site{RAM}\fi\fi}}%
358 }% do
359 \xdef\wp@sites@line{\wp@sites@line&\sum@style{\wp@legend@all}}%
360 \xdef\wp@efforts@line{\wp@efforts@line&
361 \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.

```
362 \newcommand\wpheadertable{%
363 \wp@sites@efforts@lines%
364 \par\noindent\begin{tabular}{|l|l|l|*{\thep@sites@num}{c|}|c|}\hline%
365 \textbf{\wp@mk@title{\wp@num}}&\wp@sites@line\\\hline%
366 \textsf{\pdata@target{wp}{\wp@id}{\pdataref{wp}\wp@id{title}}} &\wp@efforts@line\\\hline%
367 \end{tabular}\smallskip\par\noindent\ignorespaces}
```

and now multilinguality support

```
368 \newcommand\wp@legend@site{Site}
```

⁷EDNOTE: document above

⁸EDNOTE: document above

```

369 \newcommand\wp@legend@effort{Effort\if@RAM{ (RM+RAM)}\fi}
370 \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

```

371 \newcounter{prop@RM}\if@RAM\newcounter{prop@RAM}\fi
372 \ifwork@areas
373 \newcounter{wa@RM}\if@RAM\newcounter{wa@RAM}\fi\newcounter{wa@wps}
374 \newenvironment{workarea}[1] []
375 {\setkeys{workarea}{#1}
376 \let\@wps=\relax
377 \stepcounter{wa}
378 \pdata@def{wa}{\wa@id}{\label}{\wa@label\thewa}
379 \pdata@def{wa}{\wa@id}{\number}{\thewa}
380 \pdata@def{wa}{\wa@id}{\page}{\thepage}
381 \update@was{\wa@id}
382 \pdata@def{wa}{\wa@id}{\num}{\thewa}
383 \setcounter{wa@RM}{0}\if@RAM\setcounter{wa@RAM}{0}\fi\setcounter{wa@wps}{0}
384 \edef\@wps{\pdata@ref@aux\wa@id{wp}{ids}}
385 \for\@wp:=\@wps\do{\stepcounter{wa@wps}%
386 \if@sites
387 \@for\@site:=\prop@gen@sites\do{%
388 \edef\@RM{\pdata@ref@num\@wp\@site{RM}}
389 \if@RAM\edef\@RAM{\pdata@ref@num\@wp\@site{RAM}}\fi
390 \addtocounter{wa@RM}{\@RM}\addtocounter{prop@RM}{\@RM}
391 \if@RAM\addtocounter{wa@RAM}{\@RAM}\addtocounter{prop@RAM}{\@RAM}\fi}
392 \else
393 \edef\@RM{\pdata@ref@num{wp}\@wp{RM}}
394 \if@RAM\edef\@RAM{\pdata@ref@num{wp}\@wp{RAM}}\fi
395 \addtocounter{wa@RM}{\@RM}\addtocounter{prop@RM}{\@RM}
396 \if@RAM\addtocounter{wa@RAM}{\@RAM}\addtocounter{prop@RAM}{\@RAM}\fi
397 \fi}
398 \pdata@def{wa}{\wa@id{RM}}\thewa@RM
399 \pdata@def{prop}{all}{RM}\theprop@RM
400 \if@RAM
401 \pdata@def{wa}{\wa@id{RAM}}\thewa@RAM
402 \pdata@def{prop}{all}{RAM}\theprop@RAM
403 \fi
404 \subsubsection*{\wa@mk@title\thewa}: {\pdata@target{wa}\wa@id{\pdata@ref{wa}\wa@id{title}}}}
405 \addcontentsline{toc}{subsubsection}{\wa@mk@title\thewa}: \pdata@ref{wa}\wa@id{title}}%
406 \ignorespaces}
407 {\@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.

```

408 \ifdelivs\newwrite\wpg@delivs\fi
409 \newenvironment{workplan}%
410 {\ifdelivs\immediate\openout\wpg@delivs=\jobname.delivs\fi
411 \ifwork@areas\let\@was=\relax\else\let\@wps=\relax\fi}%
412 {\@ifundefined{task@deps}{\pdata@def{all}{task}{deps}{\task@deps}}
413 \pdata@def{all}{task}{count}{\thealltasks}
414 \ifwork@areas
415 \@ifundefined{@was}{\pdata@def{all}{wa}{ids}\@was}
416 \else
417 \@ifundefined{@wps}{\pdata@def{all}{wp}{ids}\@wps}
418 \fi

```

```

419 \ifdelivs\@ifundefined{mile@stones}{%
420 {\@for\@I:=\mile@stones\do{%
421 \pdata@def{mile}\@I{delivs}{\@ifundefined{\@I delivs}{\csname\@I delivs\endcsname}}}\fi
422 \ifwork@areas\pdata@def{all}{wa}{count}{\thewa}\fi
423 \pdata@def{all}{wp}{count}{\theallwp}
424 \ifdelivs
425 \pdata@def{all}{deliverables}{count}{\thedeliverable}
426 \pdata@def{all}{milestones}{count}{\themilestone}
427 \fi
428 \ifdelivs\closeout\wpg@delivs\fi}

```

4.7 Tasks

tasklist

```

429 \newenvironment{tasklist}
430 {\smallskip\begin{compactenum}}{\end{compactenum}\smallskip}

```

The next step is to

```

431 \ifwork@areas
432 \newcommand\task@label[3]{\textbf{T#1.#2.#3}}
433 \else
434 \newcommand\task@label[2]{\textbf{T#1.#2}}
435 \fi

```

We define the keys for the task macro

```

436 \define@key{task}{id}{\def\task@id{#1}\@dmp{id=#1}}
437 \define@key{task}{wphases}{\def\task@wphases{#1}\@dmp{wphases=#1}}
438 \define@key{task}{requires}{\@requires\task@id{#1}\@dmp{req=#1}}
439 \define@key{task}{title}{\def\task@title{#1}}
440 \define@key{task}{lead}{\def\task@lead{#1}}
441 \define@key{task}{partners}{\def\task@partners{#1}}
442 \define@key{task}{PM}{\def\task@PM{#1}}
443 \define@key{task}{issue}{\def\task@issue{#1}}
444 \def\task@set#1{\edef\task@id{task\thetask@all}}
445 \def\task@wphases{0-0}\def\task@partners{}\def\task@lead{}\def\task@PM{}\def\task@title{}
446 \setkeys{task}{#1}}

```

@post@title@space make the space after the title tweakable

```

447 \def\task@post@title@space{\;}

```

task The task environment. We first set up config stuff

```

448 \newcounter{alltasks}
449 \def\task@post@title@space{ }
450 \newcommand\task@legend@partners{Sites: }
451 \newcommand\task@legend@PM{PM}

```

now comes the environment proper. We first call \@task on the keyval argument to do the metadata handling. Then we start formatting the task as an item in the `description` list from the `tasklist` environment, and print the title if there is one

```

452 \newenvironment{task}[1][]%
453 {\stepcounter{alltasks}%
454 \@task{#1}%
455 \item[\pdata@target{task}{\taskin\task@id\wp@id}%
456 {\ifwork@areas\task@label\thewa\thewp\thetask@wp\else\task@label\thewp\thetask@wp\fi}]}%
457 \ifx\task@title\@empty\else\textbf\task@title\fi\task@post@title@space%

```

now we decode and show the work phases on the task, if they have been specified.

```

458 \def\@initial{0-0}%
459 \ifx\task@wphases\@initial\else%

```

```

460 \let\@sep=\relax\@for\@I:=\task@wphases%
461 \do{\decode@wphase\@I%
462 @@sep\show@wphase\wphase@start\wphase@end\wphase@force%
463 \let\@sep=\sep@wphases}%
464 \fi% initial

in non-submit mode we give the specified PM for cross-checking
465 \ifsubmit\else\ifx\task@PM\@empty\else\task@PM~\task@legend@PM;\fi\fi%

and we list the partners who contribute if they are specified.
466 \if@sites%
467 \ifx\task@lead\@empty\else\ \task@legend@partners\site\task@lead~(\legend@lead)%
468 \@for\@I:=\task@partners\do{, \site\@I}\fi%

if there are no partners, then we show the RM/RAM contributions specified (if any)
469 \ifx\task@partners\@empty
470 \xdef\@involvement{}\xdef\@inv{}\%
471 \xdef\@sep{, }\def\msep{}\% do not show the sep the first time around
472 \edef\@sites{\prop@gen@sites}%
473 {\let\site\relax% to to render it inert here
474 \@for\@site:=\@sites\do{%
475 \edef\@@RM{\pdataref@safe{\wp@id \@task@id}\@site{RM}}}%
476 \ifx\@@RM\@empty\else\xdef\@@inv{showit}%
477 \xdef\@involvement{\@involvement% and
478 \msep\site{\@site}: \@@RM\if@RAM\ifx\@@RAM\@empty\else/\@@RM\fi\fi}
479 \let\msep=\@sep% but the second time show it.
480 \fi}}% \@@RM empty
481 \ifx\@@inv\@empty\else(RM{\if@RAM/RAM\fi} distribution: \@involvement)\strut\\\fi
482 \fi% no partners key
483 \fi% sites

finally, we ignore any spaces that may follow the the task environment
484 \ignorespaces}
485 {\smallskip}

now the multilingual support and presentation configuration
486 \newcommand\month@label[1]{M#1}
487 \newcommand\show@wphase[3]{\edef\@test{#3}\def\@one{1}%
488 \month@label{#1}-\month@label{#2}%
489 \ifx\@test\@empty\else\ifx\@test\@one\else \@3\fi\fi}
490 \newcommand\sep@wphases{; }
491 \newcommand\legend@partners{Partners}
492 \newcommand\legend@lead{lead}
493 \newcommand\task@label@long{Task}

```

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

```

494 \newcounter{task@all}\newcounter{task@wp}[wp]
495 \newcount\task@@end
496 \def\@task#1{\stepcounter{task@all}\stepcounter{task@wp}%
497 \task@set{#1}%
498 \pdata@def{task}{\taskin\task@id\wp@id}{title}{\task@title}
499 \pdata@def{task}{\taskin\task@id\wp@id}{lead}{\task@lead}
500 \pdata@def{task}{\taskin\task@id\wp@id}{partners}{\task@partners}
501 \pdata@def{task}{\taskin\task@id\wp@id}{PM}{\task@PM}
502 \pdata@def{task}{\taskin\task@id\wp@id}{wphases}{\task@wphases}
503 \@ifundefined{task@issue}{}
504 {\pdata@def{task}{\taskin\task@id\wp@id}{issue}{\task@issue}}%
505 \ifwork@areas
506 \pdata@def{task}{\taskin\task@id\wp@id}{label}{\task@label\thewa\thewp\thetask@wp}%

```

```

507 \else
508 \pdata@def{task}{\taskin\task@id\wp@id}{label}{\task@label\thewp\thetask@wp}%
509 \fi
510 \pdata@def{task}{\taskin\task@id\wp@id}{number}{\thetask@wp}%
511 \pdata@def{task}{\taskin\task@id\wp@id}{page}{\thepage}%
512 \update@tasks{\taskin\task@id\wp@id}

```

4.8 Work Phase Metadata

`\workphase`

```

513 \newcommand\workphase[1]{\PackageError{proposal}
514   {The \protect\workphase macro is deprecated,\MessageBreak
515     use the attributes wphase on the workpackage environment instead!}}

```

`*task*ref`

```

516 \newcommand\taskin[2]{#2@#1}
517 \newcommand\taskref[2]{\pdataRef{task}{#1@#2}{label}}
518 \newcommand\taskreflong[2]{\pdataRef{task}{#2}{label}}
519 \newcommand\taskrtref[2]{\taskref{#1}{#2}: \pdataRefFB{task}{#1@#2}{short}{title}}
520 \newcommand\localtaskref[1]{\taskref{\wp@id}{#1}}
521 \newcommand\localtaskrtref[1]{\taskrtref{\wp@id}{#1}}

```

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

```

522 \newcounter{ganttd@deps}
523 \def\@requires#1#2{\stepcounter{ganttd@deps}%
524 \edef\dep@id{taskdep\theganttd@deps}%
525 \pdata@def{taskdep}\dep@id{from}{\taskin{#1}\wp@id}%
526 \pdata@def{taskdep}\dep@id{to}{#2}%
527 \update@deps\dep@id}

```

4.9 Milestones and Deliverables

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

```

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

```

`wpdelivs`

```

530 \newenvironment{wpdelivs}{\begin{wp@delivs}}{\end{wp@delivs}}

```

`wp@delivs`

```

531 \newenvironment{wp@delivs}
532 {\ifdelivs\textbf{\deliv@legend@delivs:\ll[-3ex]}%
533 \begin{compactdesc}\else\deliv@error\fi}
534 {\ifdelivs\end{compactdesc}\fi}

```

and now multilinguality support

```

535 \newcommand\deliv@legend@delivs{Deliverables}

```

`\wadelivs`

```

536 \newenvironment{wadelivs}
537 {\textbf{\deliv@legend@delivs:\ll[-3ex]}\begin{wp@delivs}}
538 {\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.

```

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

```

`\deliv@label`

```
540 \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.

```
541 \newcommand\delivref[2]{\pdataRef{deliv}{#1@#2}{label}}
542 \newcommand\localdelivref[1]{\delivref{\wp@id}{#1}}
543 \newcommand\delivtreref[2]{\delivref{#1}{#2}: \pdataRefFB{deliv}{#1@#2}{short}{title}}
544 \newcommand\localdelivtreref[1]{\delivtreref{\wp@id}{#1}}
```

`\wpg@deliv` We first define the keys

```
545 \define@key{deliv}{id}{\def\deliv@id{#1}}
546 \define@key{deliv}{due}{\def\deliv@due{#1}}
547 \define@key{deliv}{dissem}{\def\deliv@dissem{#1}}
548 \define@key{deliv}{nature}{\def\deliv@nature{#1}}
549 \define@key{deliv}{miles}{\def\deliv@miles{#1}}
550 \define@key{deliv}{short}{\def\deliv@short{#1}}
551 \define@key{deliv}{lead}{\def\deliv@lead{#1}}
552 \define@key{deliv}{issue}{\def\deliv@issue{#1}}
553 \define@key{deliv}{status}{\def\deliv@status{#1}}
554 \define@key{deliv}{blog}{\def\deliv@blog{#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.

```
555 \newcounter{deliverable}
556 \newcommand{\wpg@deliv}[3]{% keys, title, type
557 \stepcounter{deliverable}
558 \let\deliv@miles=\relax% clean state
559 \def\@type{#3}\def\@wp{\wp}% set up ifx
560 \def\wpg@id{\csname #3@id\endcsname}
561 \setkeys{deliv}{#1}\stepcounter{deliv}% set state
562 \ifx\@type\@wp\def\current@label{\deliv@label{\ifwork@areas\thewa.\fi\thewp.\thedeliv}}
563 \else\def\current@label{\deliv@label{\thewa.\thedeliv}}\fi
564 \pdata@def{deliv}{\taskin\deliv@id\wpg@id}{label}{\current@label}
565 \pdata@def{deliv}{\taskin\deliv@id\wpg@id}{title}{#2}
566 \pdata@def{deliv}{\taskin\deliv@id\wpg@id}{page}{\thepage}%
567 \ifundefined{deliv@short}
568 {\pdata@def{deliv}{\taskin\deliv@id\wpg@id}{short}{#2}}
569 {\pdata@def{deliv}{\taskin\deliv@id\wpg@id}{short}{\deliv@short}}
570 \ifundefined{deliv@nature}
571 {\protect\G@refundefinedtrue\@latex@warning{key 'nature' for Deliv \wpg@id undefined}}
572 {\pdata@def{deliv}{\taskin\deliv@id\wpg@id}{nature}{\deliv@nature}}
573 \ifundefined{deliv@dissem}
574 {\protect\G@refundefinedtrue\@latex@warning{key 'dissem' for Deliv \wpg@id undefined}}
575 {\pdata@def{deliv}{\taskin\deliv@id\wpg@id}{dissem}{\deliv@dissem}}
576 \ifundefined{deliv@lead}
577 {\protect\G@refundefinedtrue\@latex@warning{key 'lead' for Deliv \wpg@id undefined}}
578 {\pdata@def{deliv}{\taskin\deliv@id\wpg@id}{lead}{\deliv@lead}}
579 \ifundefined{deliv@due}{\pdata@def{deliv}{\taskin\deliv@id\wpg@id}{due}{\deliv@due}}
580 \ifundefined{deliv@issue}{\pdata@def{deliv}{\taskin\deliv@id\wpg@id}{issue}{\deliv@issue}}
581 \ifundefined{deliv@status}{\pdata@def{deliv}{\taskin\deliv@id\wpg@id}{status}{\deliv@status}}
582 \ifundefined{deliv@blog}{\pdata@def{deliv}{\taskin\deliv@id\wpg@id}{blog}{\deliv@blog}}
```

Then we iterate over the due dates and generate an entry for each of them in the `*.deliverables` file; but only if the status is not canceled.

```
583 \def\@status@canceled{canceled}
584 \ifx\deliv@status\@status@canceled\else
585 \ifundefined{deliv@due}{\%
586 \for\@I:=\deliv@due\do{\protected@write\wpg@delivs}{\string\deliverable%
```



```

587 {\ifnum\@I<10 0\@I\else\@I\fi}% sort key
588 {\@I}% due date
589 {\current@label}% label
590 {\@ifundefined{deliv@id}{??}{\taskin\deliv@id\wpg@id}}% id
591 {\@ifundefined{deliv@dissem}{??}{\deliv@dissem}}% dissemination level
592 {\@ifundefined{deliv@nature}{??}{\deliv@nature}}% nature
593 {#2}
594 {\ifx\@type\@wp{WP\ifwork@areas\thewa.\fi\thewp}\else{WA\thewa}\fi}%WP
595 {\@ifundefined{deliv@lead}{??}{\string\site{\deliv@lead}}}% lead
596 }%deliv@due defined
597 \fi% status != canceled

```

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

```

598 {\ifx\deliv@status\@status@canceled\color{lightgray}\fi
599 \item[\current@label\ (%
600 \delivs@legend@due: \@ifundefined{deliv@due}{??}{\deliv@due},
601 \delivs@legend@nature: \@ifundefined{deliv@nature}{??}{\deliv@nature},
602 \delivs@legend@dissem: \@ifundefined{deliv@dissem}{??}{\deliv@dissem},
603 \delivs@legend@lead: \@ifundefined{deliv@lead}{??}{\site{\deliv@lead}})]
604 \pdata@target{deliv}{\taskin\deliv@id\wpg@id}{\textit{#2}}
605 \@ifundefined{deliv@miles}{\}{% print the milestones and update their deliverables
606 \let\m@sep=\relax% do not print the separator the first time round
607 \lec{\@for\@I:=\deliv@miles\do{% Iterate over the milestones mentioned
608 \m@sep\pdataRef{mile}{\@I}{label}}% print the milestone reference
609 \let\m@sep=,}%set the separator for the next times
610 \def\d@sep{,}
611 \@for\@I:=\deliv@miles\do{% Iterate over the milestones mentioned
612 \expandafter\ifx\csname\@I delivs\endcsname\relax% Check that the miles@delivs is empty
613 {\expandafter\xdef\csname\@I delivs\endcsname{\wpg@id \@deliv@id}}% if so, skip the separator
614 \else\expandafter\xdef\csname\@I delivs\endcsname{if not add it
615 {\csname\@I delivs\endcsname\d@sep\wpg@id \@deliv@id}\fi}}%
616 }% end gray color
617 }

```

Now, we only need to instantiate

wadeliv

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

wpdeliv

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

\milestone@label

```
620 \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.

```

621 \newcommand\mileref[1]{\pdataRef{mile}{#1}{label}}
622 \newcommand\miletrf[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.

```

623 \newcounter{milestone}
624 \define@key{milestone}{id}{\gdef\mile@id{#1}}
625 \define@key{milestone}{month}{\gdef\mile@month{#1}}
626 \newcommand\milestone[3][\%
627 \ifdelivs%
628 \setkeys{milestone}{#1}\stepcounter{milestone}%

```

```

629 \pdata@def{mile}\mile@id{label}\milestone@label{\themilestone}}%
630 \pdata@def{mile}\mile@id{month}\mile@month}%
631 \pdata@def{mile}\mile@id{title}{#2}%
632 \pdata@def{mile}\mile@id{description}{#3}%
633 \ifundefined{mile@stones}%
634 {\xdef\mile@stones{\mile@id}}%
635 {\xdef\mile@stones{\mile@stones,\mile@id}}%
636 \@milestone{\mile@id}{#2}{#3}% presentation
637 \else\deliv@error\fi}

\@milestone the corresponding presentation macro.

638 \newcommand\@milestone[3]{% id, title, description
639 \item \textbf{\miles@legend@milestone\hspace\pdata@target{mile}\mile@id{\pdataref{mile}{#1}{label}}}
640 (\miles@legend@month \pdataref{mile}\mile@id{month})
641 \textbf{#2} #3}
642 \newcommand\miles@legend@month{Month}
643 \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.

644 \newenvironment{milestones}%
645 {\ifdelivs\begin{\@milestones}\else\deliv@error\fi}
646 {\ifdelivs\pdata@def{all}{mile}{ids}{\mile@stones}%
647 \pdata@def{all}{mile}{count}{\themilestone}%
648 \end{\@milestones}\fi}

\@milestones here we do the work.

649 \newenvironment{\@milestones}{\begin{enumerate}}{\end{enumerate}}

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

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

deliverables

651 \newenvironment{deliverables}[1]{\ifdelivs\begin{longtable}{|l|p{#1}|l|l|l|l|l|}\hline%
652 \#&\textbf{\delivs@legend@name}&%
653 \textbf{\delivs@legend@wp}&%
654 \textbf{\delivs@legend@lead}&%
655 \textbf{\delivs@legend@nature}&%
656 \textbf{\delivs@legend@level}&%
657 \textbf{\delivs@legend@due}\\\hline\hline%
658 \endhead%
659 \else\deliv@error\fi}
660 {\ifdelivs\end{longtable}\fi}

now the multilingual support

661 \newcommand\delivs@legend@name{Deliverable name}
662 \newcommand\delivs@legend@wp{WP}
663 \newcommand\delivs@legend@nature{Type}
664 \newcommand\delivs@legend@level{Level}
665 \newcommand\delivs@legend@due{Due}
666 \newcommand\delivs@legend@dissem{Dissem.}
667 \newcommand\delivs@legend@lead{Lead}

\inputdelivs

668 \newcommand{\inputdelivs}[1]{%
669 \begin{deliverables}{#1}%
670 \IfFileExists{\jobname.deliverables}%
671 {\input{\jobname.deliverables}}%

```

```

672 {\IfFileExists{\jobname.delivs}{\input{\jobname.delivs}}{}}
673 \end{deliverables}}
674 \end{sty}

```

4.10 Project Data, Referencing & Hyperlinking

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

```

675 (*pdata)
676 \newif\ifwork@areas\work@areastrue
677 \DeclareOption{nowork@areas}{\work@areasfalse}
678 \ProcessOptions
679 \RequirePackage{xspace}
680 \newwrite\pdata@out
681 \newcommand\pdata@open[1]{\immediate\openout\pdata@out=#1.pdata}
682 \newcommand\pdata@close{\closeout\pdata@out}

```

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

```

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

```

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

```

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

```

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

```

687 \newcommand\pdata@def[4]{%\@pdata@def{#1}{#2}{#3}{#4}%
688 \protected@write\pdata@out{\string\@pdata@def{#1}{#2}{#3}{#4}}%

```

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

```

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

```

`\pdataref`

```

690 \newcommand\pdataref[3]{\@ifundefined{#1@#2@#3}%
691 {\protect\G@refundefinedtrue\@latex@warning{#3 for #1 #2 undefined}??}%
692 {\csname #1@#2@#3\endcsname}}%
693 \newcommand\pdataref@aux[3]{\@ifundefined{#1@#2@#3}{??}{\csname #1@#2@#3\endcsname}}%
694 \newcommand\pdataref@num[3]{\@ifundefined{#1@#2@#3}{0}{\csname #1@#2@#3\endcsname}}%
695 \newcommand\pdataref@safe[3]{\@ifundefined{#1@#2@#3}{\csname #1@#2@#3\endcsname}}%

```

`\pdatarefFB` a variant with fallback field,

```

696 \newcommand\pdatarefFB[4]{\@ifundefined{#1@#2@#3}%
697 {\@ifundefined{#1@#2@#4}%
698 {\protect\G@refundefinedtrue\@latex@warning{both #3 and its fallback #4 undefined for #1 #2}??}%
699 {\csname #1@#2@#4\endcsname}}
700 {\csname #1@#2@#3\endcsname}}

```

`\pdataRef`

```

701 \newcommand\pdataRef[3]{\@ifundefined{#1@#2@#3}%
702 {\protect\G@refundefinedtrue\@latex@warning{#3 for #1 #2 undefined}??}%
703 {\hyperlink{#1@#2@target}{\csname #1@#2@#3\endcsname}}}

```

`\pdataRefFB` a variant with fallback field,

```

704 \newcommand\pdataRefFB[4]{\@ifundefined{#1@#2@#3}%
705 {\@ifundefined{#1@#2@#4}%
706 {\protect\G@refundefinedtrue\@latex@warning{both #3 and its fallback #4 undefined for #1 #2}??}%
707 {\hyperlink{#1@#2@target}{\csname #1@#2@#4\endcsname}}}
708 {\hyperlink{#1@#2@target}{\csname #1@#2@#3\endcsname}}}

```

```

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

\pn*
712 \newcommand\pn{\pdataref{prop}{gen}{acronym}\xspace}
713 \newcommand\pnlong{\pdataref{prop}{gen}{acrolong}\xspace}

\W*ref
714 \newcommand\WPref[1]{\pdataRef{wp}{#1}{label}}
715 \newcommand\WPtref[1]{\WPref{#1}: \pdataRefFB{wp}{#1}{short}{title}}
716 \ifwork@areas
717 \newcommand\WAref[1]{\pdataRef{wa}{#1}{label}}
718 \newcommand\WAtref[1]{\WAref{#1}: \pdataRefFB{wa}{#1}{short}{title}}
719 \fi
720 \end{pdata}

```

4.11 The Work Package Table

EdN⁹ These macros⁹ determine the styling of cells in the work package table. That can be tweaked by redefining them.

```

721 \sty
722 \definecolorset{gray/rgb/hsb/cmyk}{}{}%
723 {leadgray,.90/.90,.90,.90/0,0,.90/0,0,0,.10;%
724 wagray,.70/.70,.70,.70/0,0,.70/0,0,0,.30;%
725 ganttgray,.60/.60,.60,.60/0,0,.60/0,0,0,.40}
726 \newcommand\sum@style[1]{\cellcolor{wagray}{\textbf{#1}}}
727 \newcommand\wa@style[1]{\cellcolor{wagray}{\textbf{#1}}}
728 \newcommand\wp@style[1]{#1}
729 \newcommand\lead@style[1]{\cellcolor{leadgray}{\textit{#1}}}
730 \newcommand\wp@lead@style@explained{light gray italicised}

```

\wpfigstyle

```

731 \def\wpfig@style{}
732 \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.

```

733 \newcounter{wpfig@options}
734 \define@key{wpfig}{size}{\def\wpfig@size{#1}\@dmp{size=#1}}
735 \def\@true{true}
736 \def\wpfig@pages{false}
737 \define@key{wpfig}{pages}[true]{\def\wpfig@pages{#1}\stepcounter{wpfig@options}}
738 \def\wpfig@type{false}
739 \define@key{wpfig}{type}[true]{\def\wpfig@type{#1}\stepcounter{wpfig@options}}
740 \def\wpfig@start{false}
741 \define@key{wpfig}{start}[true]{\def\wpfig@start{#1}\stepcounter{wpfig@options}}
742 \def\wpfig@length{false}
743 \define@key{wpfig}{length}[true]{\def\wpfig@length{#1}\stepcounter{wpfig@options}}
744 \def\wpfig@end{false}
745 \define@key{wpfig}{end}[true]{\def\wpfig@end{#1}\stepcounter{wpfig@options}}
746 \define@key{wpfig}{label}{\def\wpfig@label{#1}}
747 \define@key{wpfig}{caption}{\def\wpfig@caption{#1}}

```

⁹EdNOTE: 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.¹⁰ Depending on the various class and `wpfig` options, we make header and footer line for the table.

EdN:10

```

748 \def\@sw#1{\begin{sideways}#1\end{sideways}}
749 \newenvironment{wp@figure}{\begin{table}[ht]\wpfig@style\begin{center}}
750 {\let\@sw\relax\let\textbf\relax\let\site\relax\let\pn\relax\let\sys\relax%
751 \gdef\wpfig@headline{\wpfig@legend@wap&\wpfig@legend@title%
752 \ifx\wpfig@type\@true&\wpfig@legend@type\fi%
753 \ifx\wpfig@pages\@true&\@sw{\wpfig@legend@page}\fi%
754 \ifx\wpfig@start\@true&\@sw{\wpfig@legend@start}\fi%
755 \ifx\wpfig@length\@true&\@sw{\wpfig@legend@length}\fi
756 \ifx\wpfig@end\@true&\@sw{\wpfig@legend@end}\fi}%
757 \if@sites%
758 \@for\@site:=\prop@gen@sites\do{%
759 \xdef\wpfig@headline{\wpfig@headline&\@sw{\wpfig@legend@siteRM{\@site}}}%
760 \if@RAM\xdef\wpfig@headline{\wpfig@headline&\@sw{\wpfig@legend@siteRAM{\@site}}}\fi}%
761 \xdef\wpfig@headline{\wpfig@headline&\@sw{\wpfig@legend@totalRM}}%
762 \if@RAM\xdef\wpfig@headline{\wpfig@headline&\@sw{\wpfig@legend@totalRAM}}\fi%
763 \else% \if@sites
764 \xdef\wpfig@headline{\wpfig@headline &\@sw{\wpfig@legend@RM}\if@RAM&\@sw{\wpfig@legend@RAM}\fi}
765 \fi}%\if@sites
766 \if@RAM\begin{tabular}{|l|l|*{\thepwfig@options}{r|}*{\the@sites}{r|r|}{r|r|}\hline
767 \else\begin{tabular}{|l|l|*{\thepwfig@options}{r|}*{\the@sites}{r|}{r|}\hline\fi%
768 \wpfig@headline\\\hline\hline}
769 {\end{tabular}}\smallskip\\
770 \wpfig@legend@RAM@expl\if@sites; \wpfig@legend@lead@expl\fi
771 \@ifundefined{wpfig@label}{\caption{\wpfig@legend@caption}}{\caption{\wpfig@caption}}
772 \@ifundefined{wpfig@label}{\label{fig:wplist}}{\label{\wpfig@label}}
773 \end{center}\end{table}}

```

and now multilinguality support

```

774 \newcommand\wpfig@legend@wap{\textbf{\ifwork@areas{WA/P}\else{WP}\fi}}
775 \newcommand\wpfig@legend@title{\textbf{Title}}
776 \newcommand\wpfig@legend@type{\textbf{type}}
777 \newcommand\wpfig@legend@page{\textbf{page}}
778 \newcommand\wpfig@legend@start{\textbf{start}}
779 \newcommand\wpfig@legend@length{\textbf{length}}
780 \newcommand\wpfig@legend@end{\textbf{end}}
781 \newcommand\wpfig@legend@siteRM[1]{\site{#1}\if@RAM\ RM\fi}
782 \newcommand\wpfig@legend@siteRAM[1]{\site{#1}\ RAM}
783 \newcommand\wpfig@legend@totalRM{total\if@RAM\ RM\fi}
784 \newcommand\wpfig@legend@totalRAM{total RAM}
785 \newcommand\wpfig@legend@RM{RM}
786 \newcommand\wpfig@legend@RAM{RAM}
787 \newcommand\wpfig@legend@RAM@expl{\if@RAM R(A)M $\widehat{=}$ Researcher (Assistant) Months\else\ Efforts in PM}
788 \newcommand\wpfig@legend@lead@expl{WP lead efforts \wp@lead@style@explained}
789 \newcommand\wpfig@legend@caption{\ifwork@areas Work Areas and \fi}Work Packages}

```

EdN:11\wpfig 11

```

790 \newcount\local@count
791 \newcount\@@@RM\if@RAM\newcount\@@@RAM\fi
792 \newcount\all@@@RM\if@RAM\newcount\all@@@RAM\fi
793 \newcommand{\wpfig}[1][\setcounter{wpfig@options}{0}\setkeys{wpfig}{#1}

```

¹⁰EdNOTE: this is a bit of misnomer, it does not do the figure bit.

¹¹EdNOTE: 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.

```

794 {\gdef\wp@lines{}}initialize
795 \let\tabularnewline\relax\let\hline\relax\let\lead@style\relax% so they
796 \let\wa@style\relax\let\wp@style\relax \let\sw@style\relax\let\textbf\relax% do not
797 \let\G@refundefinedtrue=\relax\let\@latex@warning=\relax\let\hyperlink=\relax% bother
798 \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.

799 \ifwork@areas
800 \edef\@was{\pdataref@safe{all}{wa}{ids}}%
801 \@for\@wa:=\@was\do{% iterate over the work areas
802 \xdef\@wa@line{\wa@style{\pdataref{wa}\@wa{label}}}%
803 &\wa@style{\@ifundefined{wa@\@wa @short}{\pdataref{wa}\@wa{title}}{\pdataref{wa}\@wa{short}}}%
804 \ifx\wpfig@type\@true&\wa@style{\pdataref{wa}\@wa{type}}\fi%
805 \ifx\wpfig@pages\@true&\wa@style{\pdataref{wa}\@wa{page}}\fi%
806 \ifx\wpfig@start\@true&\wa@style{\pdataref{wa}\@wa{start}}\fi%
807 \ifx\wpfig@length\@true&\wa@style{\pdataref{wa}\@wa{len}}\fi%
808 \ifx\wpfig@end\@true&\wa@style{\pdataref{wa}\@wa{end}}\fi%
809 \if@sites
810 \@for\@site:=\prop@gen@sites\do{%
811 \edef\@wps{\pdataref@safe{\@wa{wp}{ids}}}%
812 \local@count 0%
813 \@for\@wp:=\@wps\do{\advance\local@count by \pdataref@num{\@wp\@site{RM}}}%
814 \pdata@def\@wa\@site{RM}{\the\local@count}%
815 \xdef\@wa@line{\@wa@line&\wa@style{\the\local@count}}%
816 \if@RAM
817 \local@count 0%
818 \@for\@wp:=\@wps\do{\advance\local@count by \pdataref@num{\@wp\@site{RAM}}}%
819 \pdata@def\@wa\@site{RAM}{\the\local@count}%
820 \xdef\@wa@line{\@wa@line&\wa@style{\the\local@count}}%
821 \fi}
822 \local@count0\relax%
823 \@for\@site:=\prop@gen@sites\do{\global\advance\local@count by \pdataref@num{\@wa\@site{RM}}}%
824 \xdef\@wa@line{\@wa@line &\wa@style{\textbf{\the\local@count}}}
825 \if@RAM
826 \local@count0\relax%
827 \@for\@site:=\prop@gen@sites\do{\global\advance\local@count by \pdataref@num{\@wa\@site{RAM}}}%
828 \xdef\@wa@line{\@wa@line &\wa@style{\textbf{\the\local@count}}}
829 \fi
830 \else% if@sites
831 \edef\@wps{\pdataref@safe{all}{wp}{ids}}%
832 \xdef\@wa@line{\@wa@line&\wa@style{\pdataref{wa}\@wa{RM}}}%
833 \if@RAM&\wa@style{\pdataref{wa}\@wa{RAM}}\fi%
834 \fi% if@sites
835 \xdef\wp@lines{\wp@lines\@wa@line\tabularnewline\hline}% add the line for the workarea
836 \edef\@wps{\pdataref@safe{\@wa{wp}{ids}}}%
837 \@for\@wp:=\@wps\do{% iterate over its work packages
838 \xdef\@wp@line{\pdataref{wp}\@wp{label}}%
839 &\@ifundefined{wp@\@wp @short}{\pdataref{wp}\@wp{title}}{\pdataref{wp}\@wp{short}}}%
840 \ifx\wpfig@type\@true&\pdataref{wp}\@wp{type}\fi%
841 \ifx\wpfig@pages\@true&\pdataref{wp}\@wp{page}\fi%
842 \ifx\wpfig@start\@true&\pdataref{wp}\@wp{start}\fi%
843 \ifx\wpfig@length\@true&\pdataref{wp}\@wp{len}\fi%
844 \ifx\wpfig@end\@true&\pdataref{wp}\@wp{end}\fi}

```

```

845 \if@sites
846 \@for\@site:=\prop@gen@sites\do{%
847 \edef\@@lead{\pdataref@safe{wp}\@wp{lead}}
848 \edef\@@RM{\ifx\@@lead\@site\lead@style{\pdataref@safe\@wp\@site{RM}}\else\wp@style{\pdataref@safe\@wp\@site{RM}}\fi}
849 \xdef\@@wp@line{\@wp@line&\@@RM}
850 \if@RAM
851 \edef\@@RAM{\ifx\@@lead\@site\lead@style{\pdataref@safe\@wp\@site{RAM}}\else\wp@style{\pdataref@safe\@wp\@site{RAM}}\fi}
852 \xdef\@@wp@line{\@wp@line&\@@RAM}
853 \fi}
854 \local@count0\relax%
855 \@for\@site:=\prop@gen@sites\do{\global\advance\local@count by \pdataref@num\@wp\@site{RM}}%
856 \xdef\@@wp@line{\@wp@line &\textbf{\the\local@count}}
857 \if@RAM
858 \global\local@count0\relax%
859 \@for\@site:=\prop@gen@sites\do{\global\advance\local@count by \pdataref@num\@wp\@site{RAM}}%
860 \xdef\@@wp@line{\@wp@line &\textbf{\the\local@count}}
861 \fi% if@RAM
862 \else% if@sites
863 \xdef\@@wp@line{\@wp@line&\wp@style{\pdataref@safe{wp}\@wp{RM}}}%
864 \if@RAM\xdef\@@wp@line{\@wp@line&\wp@style{\pdataref@safe{wp}\@wp{RAM}}}\fi
865 \fi% if@sites
866 \xdef\@wp@lines{\@wp@lines\@wp@line\tabularnewline\hline}}

Now the case where we do not have work areas.

867 \else% ifwork@areas
868 \edef\@@wps{\pdataref@safe{all}{wp}{ids}}%
869 \@for\@wp:=\@wps\do{% iterate over its work packages
870 \xdef\@@wp@line{\pdataref{wp}\@wp{label}}%
871 &\ifundefined{wp\@@wp\@short}{\pdataref{wp}\@wp{title}}{\pdataref{wp}\@wp{short}}
872 \ifx\wpfig@type\@true&\pdataref{wp}\@wp{type}\fi%
873 \ifx\wpfig@pages\@true&\pdataref{wp}\@wp{page}\fi%
874 \ifx\wpfig@start\@true&\pdataref{wp}\@wp{start}\fi%
875 \ifx\wpfig@length\@true&\pdataref{wp}\@wp{len}\fi%
876 \ifx\wpfig@end\@true&\pdataref{wp}\@wp{end}\fi}
877 \if@sites
878 \@for\@site:=\prop@gen@sites\do{%
879 \edef\@@lead{\pdataref@safe{wp}\@wp{lead}}
880 \edef\@@RM{\ifx\@@lead\@site\lead@style{\pdataref@safe\@wp\@site{RM}}\else\wp@style{\pdataref@safe\@wp\@site{RM}}\fi}
881 \xdef\@@wp@line{\@wp@line&\@@RM}
882 \if@RAM
883 \edef\@@RAM{\ifx\@@lead\@site\lead@style{\pdataref@safe\@wp\@site{RAM}}\else\wp@style{\pdataref@safe\@wp\@site{RAM}}\fi}
884 \xdef\@@wp@line{\@wp@line&\wp@style\@@RAM}
885 \fi}
886 \global\local@count0\relax%
887 \@for\@site:=\prop@gen@sites\do{\global\advance\local@count by \pdataref@num\@wp\@site{RM}}%
888 \xdef\@@wp@line{\@wp@line &\textbf{\the\local@count}}
889 \if@RAM
890 \global\local@count0\relax%
891 \@for\@site:=\prop@gen@sites\do{\global\advance\local@count by \pdataref@num{#1}\@site{RAM}}%
892 \xdef\@@wp@line{\@wp@line &\textbf{\the\local@count}}
893 \fi
894 \else% if@sites
895 \xdef\@@wp@line{\@wp@line&\wp@style{\pdataref@safe{wp}\@wp{RM}}}%
896 \if@RAM\xdef\@@wp@line{\@wp@line&\wp@style{\pdataref@safe{wp}\@wp{RAM}}}\fi
897 \fi% if@sites
898 \xdef\@wp@lines{\@wp@lines\@wp@line\tabularnewline\hline}}
899 \fi%ifwork@areas

```

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

```

900 \gdef\@totals{}
901 \ifwork@areas
902 \if@sites
903 \@for\@site:=\prop@gen@sites\do{% iterate over the sites
904 \@@@RM=0\if@RAM\@@@RAM=0\fi
905 \edef\@@was{\pdataref@safe{all}{wa}{ids}}%
906 \@for\@wa:=\@was\do{% iterate over the work areas
907 \edef\@@wps{\pdataref@safe\@wa{wp}{ids}}%
908 \@for\@wp:=\@wps\do{% iterate over the work packages
909 \advance\@@@RM by \pdataref@num\@wp\@site{RM}%
910 \if@RAM\advance\@@@RAM by \pdataref@num\@wp\@site{RAM}\fi}
911 \pdata@def{all}\@site{RM}\the\@@@RM\if@RAM\pdata@def{all}\@site{RAM}\the\@@@RAM\fi
912 \advance\all\@@@RM by \the\@@@RM\if@RAM\advance\all\@@@RAM by \the\@@@RAM\fi
913 \xdef\@totals{\@totals & \textbf{\the\@@@RM}\if@RAM& \textbf{\the\@@@RAM}\fi}
914 \xdef\@totals{\@totals & \textbf{\the\all\@@@RM}\if@RAM&\textbf{\the\all\@@@RAM}\fi}
915 \pdata@def{all}{total}{RM}\the\all\@@@RM\if@RAM\pdata@def{all}{total}{RAM}\the\all\@@@RAM\fi
916 \else% if@sites
917 \@@@RM=0\if@RAM\@@@RAM=0\fi
918 \edef\@@was{\pdataref@safe{all}{wa}{ids}}%
919 \@for\@wa:=\@was\do{\edef\@@wps{\pdataref@safe\@wa{wp}{ids}}%
920 \@for\@wp:=\@wps\do{% iterate over the work packages
921 \advance\@@@RM by \pdataref@num{wp}\@wp{RM}%
922 \if@RAM\advance\@@@RAM by \pdataref@num{wp}\@wp{RAM}\fi}
923 \pdata@def{all}{total}{RM}\the\@@@RM\if@RAM\pdata@def{all}{total}{RAM}\the\@@@RAM\fi
924 \xdef\@totals{\&\the\@@@RM\if@RAM &\the\@@@RAM\fi}
925 \fi% if@sites
926 \else%i.e. no work@areas
927 \if@sites
928 \@for\@site:=\prop@gen@sites\do{%iterate over the sites
929 \@@@RM=0\if@RAM\@@@RAM=0\fi%
930 \edef\@@wps{\pdataref@safe{all}{wp}{ids}}%
931 \@for\@wp:=\@wps\do{% iterate over the work packages
932 \advance\@@@RM by \pdataref@num\@wp\@site{RM}%
933 \if@RAM\advance\@@@RAM by \pdataref@num\@wp\@site{RAM}\fi}
934 \pdata@def{all}\@site{RM}\the\@@@RM\if@RAM\pdata@def{all}\@site{RAM}\the\@@@RAM\fi
935 \xdef\@totals{\@totals & \textbf{\the\@@@RM}\if@RAM& \textbf{\the\@@@RAM}\fi}
936 \advance\all\@@@RM by \the\@@@RM\if@RAM\advance\all\@@@RAM by \the\@@@RAM\fi}
937 \xdef\@totals{\@totals &\textbf{\the\all\@@@RM}\if@RAM&\textbf{\the\all\@@@RAM}\fi}
938 \pdata@def{all}{total}{RM}\the\all\@@@RM\if@RAM\pdata@def{all}{total}{RAM}\the\all\@@@RAM\fi
939 \else% if@sites
940 \@@@RM=0\if@RAM\@@@RAM=0\fi
941 \edef\@@wps{\pdataref@safe{all}{wp}{ids}}%
942 \@for\@wp:=\@wps\do{% iterate over the work packages
943 \advance\@@@RM by \pdataref@num{wp}\@wp{RM}%
944 \if@RAM\advance\@@@RAM by \pdataref@num{wp}\@wp{RAM}\fi}
945 \pdata@def{all}{total}{RM}\the\@@@RM\if@RAM\pdata@def{all}{total}{RAM}\the\@@@RAM\fi
946 \xdef\@totals{\&\the\@@@RM\if@RAM &\the\@@@RAM\fi}
947 \fi% if@sites
948 \fi

```

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

```

949 \gdef\intended@totals{}\gdef\requested@totals{}
950 \if@sites
951 \@for\@site:=\prop@gen@sites\do{
952 \xdef\intended@totals{\intended@totals&\textbf{\pdataref@safe{site}\@site{intendedRM}}}
953 \xdef\requested@totals{\requested@totals&\pdataref@safe{site}\@site{reqPM}}
954 \if@RAM\xdef\intended@totals{\intended@totals&\textbf{\pdataref@safe{site}\@site{intendedRAM}}}\fi}
955 \if@RAM\xdef\intended@totals{\intended@totals&\else%
956 \xdef\intended@totals{\intended@totals&}%

```



```

957 \xdef\requested@totals{\requested@totals&}%
958 \fi
959 \else% if@sites
960 \xdef\intended@totals{\intended@totals&\textbf{\pdataref@safe{all}{intended}{RM}}}%
961 \if@RAM\xdef\intended@totals{\intended@totals&\textbf{\pdataref@safe{all}{intended}{RAM}}}%\fi
962 \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.

```

963 \local@count\thepfig@options\advance\local@count by 2
964 \begin{wp@figure}
965 \@wp@lines\hline%
966 \multicolumn{\the\local@count}{|c|}{\prop@legend@totals}\@totals\\\hline%
967 \ifsubmit\else%
968 \ifx\prop@gen@topdownPM@true%
969 \multicolumn{\the\local@count}{|c|}{\prop@legend@intendedtotals}\intended@totals\\\hline%
970 \fi% topdownPM
971 \ifx\prop@gen@botupPM@true%
972 \multicolumn{\the\local@count}{|c|}{\prop@legend@requestedtotals}\requested@totals\\\hline%
973 \fi% botupPM
974 \fi% submit
975 \end{wp@figure}}

```

and now multilinguality support

```

976 \newcommand\prop@legend@totals{\textbf{totals}}
977 \newcommand\prop@legend@intendedtotals{\textbf{intended totals}}
978 \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

```

979 \newif\ifgantt@draft\gantt@draftfalse
980 \newif\ifgantt@miles\gantt@milesfalse
981 \define@key{gantt}{xscale}{\def\gantt@xscale{#1}}
982 \define@key{gantt}{yscale}{\def\gantt@yscale{#1}}
983 \define@key{gantt}{step}{\def\gantt@step{#1}}
984 \define@key{gantt}{size}{\def\gantt@size{#1}}
985 \define@key{gantt}{draft}[true]{\ifsubmit\else\gantt@drafttrue\fi}
986 \define@key{gantt}{milestones}[true]{\gantt@milestrue}

```

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

```

987 \def\gantt@set#1{\gantt@draftfalse\def\gantt@xscale{1}\def\gantt@yscale{.35}\def\gantt@step{3}
988 \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.

```

989 \newenvironment{gantt}[2]{}
990 {\gantt@set{#1}\gdef\gantt@height{#2}
991 \def\@test{\prop@gen@months@default}
992 \ifx\@test\prop@gen@months
993 \ClassError{proposal}{Need overall project months to draw gantt
994 chart - expect trouble;\MessageBreak specify
995 \protect\begin{proposal}[...months=??,...] to fix}\fi
996 \@ifundefined{gantt@size}{\curname\gantt@size\endcurname}

```

```

997 \newdimen\gantt@ymonths
998 \gantt@ymonths=\gantt@height cm
999 \advance\gantt@ymonths by .8cm
1000 \begin{tikzpicture}[xscale=\gantt@xscale,yscale=\gantt@yscale]}
1001 {\draw[xstep=\gantt@step,very thin] (0,0) grid (\prop@gen@months,\gantt@height);
1002 \foreach \x in {0,\gantt@step,...,\prop@gen@months} \node at (\x,\gantt@ymonths) {\x};
1003 \ifgantt@miles
1004 \newdimen\gantt@ymiles\gantt@ymiles=\gantt@height cm
1005 \advance\gantt@ymiles by 2cm
1006 \newdimen\gantt@ymiles@top\gantt@ymiles@top=\gantt@height cm
1007 %\advance\gantt@ymiles@top by 2cm
1008 \edef\@@miles{\pdataref@safe{all}{mile}{ids}}
1009 \@for\@I:=\@@miles\do{%
1010 \edef\@@month{\pdataref@safe{mile}{\@I}{month}}
1011 \draw[very thick,blue] (\@@month,\gantt@ymiles@top) -- (\@@month,0);
1012 \node[blue] at (\@@month,\gantt@ymiles) {\pdataref{mile}{\@I}{label}};
1013 \fi %gantt@miles
1014 \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.

```

1015 \newdimen\gantt@ymid\newdimen\gantt@yinc\newdimen\gantt@xend
1016 \newcommand{\@action}[6][\def\@test{#1}%
1017 \ifx\@test\@empty\def\@@color{ganttgray}\else\def\@@color{#1}\fi
1018 \gantt@ymid=#3 cm\gantt@yinc=\gantt@yscale cm
1019 \gantt@xend=#4 cm\advance\gantt@xend by #5 cm
1020 \advance\gantt@ymid by \gantt@yinc
1021 \fill[\@@color] (#4,#3) rectangle +(#5,#6);
1022 \node (#2@left) at (#4,\gantt@ymid) {};
1023 \node (#2@right) at (\gantt@xend,\gantt@ymid) {};

```

\@dependency

```

1024 \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**

```

1025 \newcommand\gantt@compute@effort[3]{% start, len, force
1026 \@@e=#1\advance\@@e by #2
1027 \ifnum\thegantt@month<#1\else
1028 \ifnum\thegantt@month<\@@e
1029 \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.

```

1030 \newcommand{\ganttchart}[1][\begin{figure}[ht]\centering
1031 \gantt@set{#1}
1032 \def\gantt@wps{\pdataref@num{all}{wp}{count}}
1033 \begin{gantt}[#1]{\gantt@wps}
1034 \newcounter{taskwps}\newcount\@@line
1035 \edef\@@was{\pdataref@safe{all}{wa}{ids}}
1036 \ifwork@areas
1037 \@for\@wa:=\@@was\do{% iterate over work areas
1038 \edef\@@wps{\pdataref@safe\@wa{wp}{ids}}
1039 \@for\@wp:=\@@wps\do{% iterate over work packages
1040 \stepcounter{taskwps}
1041 \@@line=\gantt@wps\advance\@@line by -\thetaskwps
1042 \edef\@@tasks{\pdataref@safe\@wp{task}{ids}}

```

```

1043 \node at (-1/\gantt@xscale,\@@line) [above=-2pt] {\pdataRef{wp}\@wp{label}};
1044 \edef\@wphases{\pdataref@safe{wp}\@wp{wphases}}
1045 \@for\@@ft:=\@wphases\do{%wp-level work phases
1046 \decode@wphase\@@ft
1047 \@action\@wp\@@line\wphase@start\wphase@len\wphase@force}
1048 \@for\@@task:=\@tasks\do{% tasks
1049 \edef\@wphases{\pdataref@safe{task}\@task{wphases}}
1050 \@for\@@ft:=\@wphases\do{%task-level work phases
1051 \decode@wphase\@@ft
1052 \@action\@task\@@line\wphase@start\wphase@len\wphase@force}}}}
1053 \else% ifwork@areas false
1054 \edef\@wps{\pdataref@safe{all}{wp}{ids}}
1055 \@for\@@wp:=\@wps\do{% iterate over work packages
1056 \stepcounter{taskwps}
1057 \@@line=\gantt@wps\advance\@@line by -\thetaskwps
1058 \edef\@tasks{\pdataref@safe\@wp{task}{ids}}
1059 \node at (-1/\gantt@xscale,\@@line) [above=-2pt] {\pdataRef{wp}\@wp{label}};
1060 \edef\@wphases{\pdataref@safe{wp}\@wp{wphases}}
1061 \@for\@@ft:=\@wphases\do{%iterate over the wp-level work phases
1062 \decode@wphase\@@ft
1063 \@action\@wp\@@line\wphase@start\wphase@len\wphase@force}
1064 \@for\@@task:=\@tasks\do{% task-level work phases
1065 \edef\@wphases{\pdataref@safe{task}\@task{wphases}}
1066 \@for\@@ft:=\@wphases\do{%iterate over the task-level work phases
1067 \decode@wphase\@@ft
1068 \@action\@task\@@line\wphase@start\wphase@len\wphase@force}}}}
1069 \fi% ifwork@areas end
1070 \edef\@@deps{\pdataref@safe{all}{task}{deps}}
1071 \@for\@@dep:=\@@deps\do{%
1072 \@dependency{\pdataref@safe{taskdep}\@dep{from}}{\pdataref@safe{taskdep}\@dep{to}}}

```

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

```

1073 \ifgantt@draft
1074 \newcounter{gantt@month}
1075 \newcount\@e\newdimen\gantt@effort\newdimen\gantt@plus
1076 \@whilenum\thegantt@month<\prop@gen@months\do{% step over months
1077 \gantt@effort=0cm
1078 \ifwork@areas
1079 \edef\@was{\pdataref@safe{all}{wa}{ids}}
1080 \@for\@@wa:=\@was\do{% iterate over work areas
1081 \edef\@wps{\pdataref@safe\@wa{wp}{ids}}
1082 \@for\@@wp:=\@wps\do{% iterate over work packages
1083 \edef\@wphases{\pdataref@safe{wp}\@wp{wphases}}
1084 \@for\@@ft:=\@wphases\do{%iterate over the wp-level work phases
1085 \decode@wphase\@@ft
1086 \gantt@compute@effort\wphase@start\wphase@len\wphase@force}
1087 \edef\@tasks{\pdataref@safe\@wp{task}{ids}}
1088 \@for\@@task:=\@tasks\do{% iterate over tasks
1089 \edef\@wphases{\pdataref@safe{task}\@task{wphases}}
1090 \@for\@@ft:=\@wphases\do{%iterate over the wp-level work phases
1091 \decode@wphase\@@ft
1092 \gantt@compute@effort\wphase@start\wphase@len\wphase@force}}}}
1093 \fill[ganttgray] (\thegantt@month,-5) rectangle +(1,\gantt@effort);
1094 \else% ifwork@areas
1095 \edef\@wps{\pdataref@safe{all}{wp}{ids}}
1096 \@for\@@wp:=\@wps\do{% iterate over work packages
1097 \edef\@wphases{\pdataref@safe{wp}\@wp{wphases}}
1098 \@for\@@ft:=\@wphases\do{%iterate over the wp-level work phases
1099 \decode@wphase\@@ft

```

```

1100         \gantt@compute@effort\wphase@start\wphase@len\wphase@force}
1101         \edef\@tasks{\pdataref@safe\@wp{task}{ids}}
1102         \for\@task:=\@tasks\do{% iterate over tasks
1103         \edef\@wphases{\pdataref@safe{task}\@task{wphases}}
1104         \for\@ft:=\@wphases\do{%iterate over the wp-level work phases
1105         \decode@wphase\@ft
1106         \gantt@compute@effort\wphase@start\wphase@len\wphase@force}}}
1107         \fill[ganttgray] (\thegantt@month,-5) rectangle +(1,\gantt@effort);
1108         \fi% ifwork@areas
1109         \stepcounter{gantt@month}}
1110     \fi% ifgantt@draft
1111 \end{gantt}
1112 \caption{\gantt@caption}\label{fig:gantt}
1113 \end{figure}\footnotetext{\gantt@footnote}

now the multilingual support
1114 \newcommand\gantt@caption@main{Gantt Chart: Overview Work Package Activities}
1115 \newcommand\gantt@caption@lower{lower bar shows the overall effort \if@RAM (RM only) \fi per month}
1116 \newcommand\gantt@caption{\gantt@caption@main\ifgantt@draft\xspace
1117 -- \gantt@caption@lower\fi}
1118 \newcommand\gantt@footnote{Bars shown at reduced height (e.g. 50\%) indicate reduced
1119 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 projects¹²

```

1120 \newcommand{\gantttaskchart}[1] [] {\begin{figure} [hbt] \centering \gantt@set{#1}
1121 \newcounter{gantt@all@tasks}%
1122 \setcounter{gantt@all@tasks}{\pdataref@num{all}{task}{count}}
1123 \addtocounter{gantt@all@tasks}{\pdataref@num{all}{wp}{count}}
1124 \begin{gantt}[#1]{\thegantt@all@tasks}
1125     \newcounter{gantt@tasks}\newcount\@line
1126     \edef\@wps{\pdataref@safe{all}{wp}{ids}}
1127     \for\@wp:=\@wps\do{% iterate over work packages
1128         \stepcounter{gantt@tasks}
1129         % \action[white]{} \@line0{48}1
1130         \edef\@tasks{\pdataref@safe\@wp{task}{ids}}
1131         \for\@task:=\@tasks\do{% iterate over the tasks
1132             \stepcounter{gantt@tasks}
1133             \@line=\thegantt@all@tasks\advance\@line by -\thegantt@tasks
1134             \node at (-.5/\gantt@xscale,\@line) [above=-2pt] {\footnotesize\taskreflong\@wp\@task}};
1135             \edef\@wphases{\pdataref@safe{task}\@task{wphases}}
1136             \for\@ft:=\@wphases\do{%iterate over the task-level work phases
1137                 \decode@wphase\@ft
1138                 \action\@task\@line\wphase@start\wphase@len\wphase@force
1139             }}}} % end all iterations
1140     \end{gantt}
1141     \caption{\gantt@caption@main{} -- \emph{\gantt@footnote}}\label{fig:gantt}
1142 \end{figure}}

```

4.13 Coherence

`\j*`

```

1143 \newcommand\jpub{\textcolor{\prop@link@color}{\textbf{\Large{$\star$}}}}
1144 \newcommand\jpro{\textcolor{\prop@link@color}{\textbf{\Large{$\bullet$}}}}
1145 \newcommand\jsoft{\textcolor{\prop@link@color}{\textbf{@}}}
1146 \newcommand\jorga{\textcolor{\prop@link@color}{\textbf{\Large{$\circ$}}}}

```

¹²EDNOTE: this should be incorporated with the gantt chart above, but I am currently too scared to do it so close to the deadline

```
1147 \newcommand\jsup{\textcolor{\prop@link@color}{\textbf{\smiley}}}
```

`\add@joint` `\add@joint{<first>}{<second>}{<sym>}` adds $\langle sym \rangle$ to the the `\coherence@<first>@<second>` macro for the coherence table.

```
1148 \newcommand\add@joint[3]{\@ifundefined{coherence@#1@#2}%
1149 {\@namedef{coherence@#1@#2}{#3}}%
1150 {\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.

```
1151 \newcommand\prop@joint[2]{\@for\@first:=#2\do{%
1152 \@for\@second:=#2\do{\ifx\@first\@second\else\add@joint\@first\@second{#1}\fi}}}
```

`\joint*` Now, some instances that use these.

```
1153 \newcommand\jointproj[1]{\prop@joint\jpro{#1}}
1154 \newcommand\jointpub[1]{\prop@joint\jpub{#1}}
1155 \newcommand\jointorga[1]{\prop@joint\jorga{#1}}
1156 \newcommand\jointsoft[1]{\prop@joint\jsoft{#1}}
1157 \newcommand\jointsup[1]{\prop@joint\jsup{#1}}
```

`\coherencematrix`

```
1158 \newcommand{\coherencematrix}{
1159 {\let\tabularnewline\relax\let\hline\relax\let\site\relax% so they do
1160 \let\@sw\relax\let\jpub\relax\let\jpro\relax\let\jorga\relax% not bother
1161 \let\jsoft\relax\let\jsup\relax\let\cellcolor\relax% us
1162 \gdef\@ct@head{%
1163 \@for\@site:=\prop@gen@sites\do{\xdef\@ct@head{\@ct@head%
1164 &\ifx\cht@swsites\@true\@sw{\site{\@site}}\else\site{\@site}\fi}}%
1165 \gdef\@ct@lines{\@ct@head\tabularnewline\hline\hline} %initialize with head line
1166 \@for\@site:=\prop@gen@sites\do{\xdef\@ct@line{\site{\@site}}%
1167 \@for\@site:=\prop@gen@sites\do{%
1168 \xdef\@ct@line{\@ct@line&\ifx\@site\@site{\cellcolor{wagray}}}\fi%
1169 \@ifundefined{coherence@\@site @\@site}{\@nameuse{coherence@\@site @\@site}}}%
1170 \xdef\@ct@lines{\@ct@lines\@ct@line\tabularnewline\hline}}}%
1171 \begin{tabular}{|l|l|*{\the@site}{c|}}\hline%
1172 \@ct@lines\hline%
1173 joint&\multicolumn{\the@site}{l}{\jpub $\hat{=}$ publication, \jpro $\hat{=}$ project,
1174 \jorga $\hat{=}$ organization, \jsoft $\hat{=}$ software/resource dev,
1175 \jsup $\hat{=}$ supervision}\hline
1176 \end{tabular}}
```

`\coherencetable`

```
1177 \newskip\@bigflushglue \@bigflushglue = -100pt plus 1fil
1178 \def\bigcenter{\trivlist \bigcentering\item\relax}
1179 \def\bigcentering{\let\\\@centercr\rightskip\@bigflushglue%
1180 \leftskip\@bigflushglue
1181 \parindent\z@\parfillskip\z@skip}
1182 \def\endbigcenter{\endtrivlist}
1183 \define@key{coherencetable}{swsites}[true]{\def\cht@swsites{#1}}
1184 \define@key{coherencetable}{stretch}{\def\cht@stretch{#1}}
1185 \newcommand\coherencetable[1][{}]{%
1186 \def\cht@swsites{false}%
1187 \def\cht@stretch{1}%
1188 \setkeys{coherencetable}{#1}%
1189 \begin{table}[ht]%
1190 \small\setlength{\tabcolsep}{.5em}%
1191 \renewcommand{\arraystretch}{\cht@stretch}%
1192 \begin{bigcenter}%
```

```

1193 \coherencematrix%
1194 \end{bigcenter}%
1195 \caption{\coherence@caption}\label{tab:collaboration}
1196 \end{table}}

now the multilinguality support
1197 \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.

```
1198 \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.

```

1199 \newif\if@allpapers\@allpaperstrue
1200 \newcommand\prop@ppl[3][]{\@allpapersfalse\message{ppl processing: #2}%
1201 \printbibliography[heading=subbibliography,type=#2,title=#3#1]%
1202 \@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.

```

1203 \newcommand\prop@ppl[1]{\message{unclassified: #1}%
1204 \printbibliography[heading=subbibliography,title=Unclassified,#1]}%
1205 \define@key{paperlist}{unclassified}[true]{\message{unclass: \prop@rl}\prop@ppl\prop@rl}

```

with this, we define a couple of keys that generate

```

1206 \define@key{paperlist}{articles}[true]{\prop@ppl{article}{Articles}}
1207 \define@key{paperlist}{chapters}[true]{\prop@ppl{inbook}{Book Chapters}}
1208 \define@key{paperlist}{confpapers}[true]{\prop@ppl[,keyword=conference]{inproceedings}{Conference Papers}}
1209 \define@key{paperlist}{wspapers}[true]{\prop@ppl[,notkeyword=conference]{inproceedings}{Workshop Papers}}
1210 \define@key{paperlist}{theses}[true]{\prop@ppl{thesis}{Theses}}
1211 \define@key{paperlist}{submitted}[true]{\prop@ppl[,keyword=submitted]{unpublished}{Submitted}}
1212 \define@key{paperlist}{books}[true]{\prop@ppl{book}{Monographs}}
1213 \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.

```
1214 \DeclareBibliographyCategory{featured}

```

`\prop@paperlist` `\prop@paperlist{<keys>}{<refs>}` generates a paper list from a list `<keys>` of bibliography keys. It makes some local adaptations 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 bibliography¹³

EdN:13

```

1215 \newcommand\prop@paperlist[2][]{%
1216 \let\bibboldfont\bibfont%
1217 \renewcommand{\bibfont}{\footnotesize}%
1218 \renewcommand{\baselinestretch}{.9}%
1219 \nocite{#2}\def\do##1{\addtocategory{featured}{##1}}\docsvlist{#2}%
1220 \setkeys{paperlist}{#1}
1221 \@ifundefined{prop@rl}{}{\@latex@warning{some papers are not classified!}}
1222 \if@allpapers\printbibliography[category=featured,heading=empty]\fi%
1223 \let\bibfont\bibboldfont}

```

¹³EdNOTE: MK: we may want to make this optional controlled by a package option eventually.

We define the warnpubs heading constructor.

```

1224 \def\prop@warnpubs@message{Many of the proposers' publications are online at one of the following URIs:}
1225 \def\prop@warnpubs@title{References}
1226 \defbibheading{warnpubs}{\section*{\prop@warnpubs@title}%
1227   \@ifundefined{prop@gen@pubspages}
1228   {\@latex@warning{No publication pages specified;
1229     use the pubspage key in the proposal environment!}}
1230   {\prop@warnpubs@message%
1231   \@for\@I:=\prop@gen@pubspages\do{\par\noindent\csname\@I\endcsname}}}
```

Finally, we tweak bibL^AT_EX to not give DOIs and URLs at the same time.

```

1232 \renewbibmacro*{event+venue+date}{%
1233 \renewbibmacro*{doi+eprint+url}{%
1234   \iftoggle{bbx:doi}
1235     {\printfield{doi}\iffieldundef{doi}{}\clearfield{url}}
1236     {}%
1237   \newunit\newblock
1238   \iftoggle{bbx:eprint}
1239     {\usebibmacro{eprint}}
1240     {}%
1241   \newunit\newblock
1242   \iftoggle{bbx:url}
1243     {\usebibmacro{url+urldate}}
1244     {}
1245 \}
```

4.15 Miscellaneous

`\signatures`

```

1246 \pdata
1247 \newcommand{\signatures}[1]{\section{#1}
1248 \qqquad\number\day. \number\month. \number\year\\[6ex]
1249 \strut\qqquad Date\hfill\@for\@p:=\prop@gen@PIs\do{%
1250 \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.

```

1251 \def\@dmp#1{\ifkeys\marginpar{#1}\fi}
```

`\euro`

```

1252 \renewcommand\euro{\officialeguro\xspace}
1253 \pdata
```