
Read the Docs Template Documentation

Release 1.0

Read the Docs

Jan 12, 2022

1	The Standardization Survival Kit: presentation	1
1.1	Why standards after all?	1
1.2	The SSK: a toolkit for Humanities scholars	2
1.3	Design principles & features of the SSK	3
1.4	SSK components: Scenarios < Steps < Resources	4
2	How to create a scenario for the SSK? (a SSK Tutorial)	5
2.1	Choose explicit titles starting with a verb or a gerund	6
2.2	Describing a scenario or a step	6
2.3	Associate keywords to the scenario or the step	7
2.4	Choose an illustration for the scenario	8
2.5	Identify relevant resources	8
2.6	Link the resources to the step	9
2.7	Advanced SSK functions (1) : customize a step	11
2.8	Advanced SSK functions (2) : link scenarios	11
3	The SSK data model (TEI)	13
3.1	The TEI specification	13
3.2	Files naming conventions	14
3.3	Scenarios and steps structure	15
3.4	Content of scenarios and steps	18
3.5	Reuse/customize a step	20
4	Technical architecture	23
4.1	SSK's input	24
4.2	SSK Back-end	24
4.3	SSK Front-End	25
4.4	Deployment	26
5	The PARTHENOS project	29
6	The Standardization Survival Kit	31
7	The SSK on Github	33

The Standardization Survival Kit: presentation

To support the digital evolution within Social Sciences and Humanities research, it is necessary to stabilize knowledge on standards and research good practices. The goal of the **Standardization Survival Kit (SSK)**, developed within the [PARTHENOS project](#), is to accompany researchers along this route, giving access to standards and best practices in a meaningful way, by the mediation of research scenarios. A research scenario is a (digital) workflow practiced by researchers, that can be repeatedly applied to a task that will help to gain material or insights in view of a research question. These scenarios are at the core of the SSK, as they embed resources with contextual information and relevant examples on standardized processes and methods in a research context. The SSK is an open tool where users are able to publish new scenarios or adapt existing ones. These scenarios can be seen as a living memory of what should be the best research practices in a given community, made accessible and reusable for other researchers.

1.1 Why standards after all?

The main issue in defining a policy about standards is to understand what they actually are. In the context of research, standards usually take the form of documents informing about practices, protocols, artefact characteristics or data formats that can be used as reference for two parties working in the same field of activity to be able to produce comparable (or interoperable) results. This will also foster innovative, cross-disciplinary research paths, and eventually contribute to bridge the gap between the different cultures that are represented in the wide landscape of the Arts, Humanities and Cultural Heritage studies.

Standards are usually published by standardization organisations (such as ISO, W3C or the TEI Consortium) which ensure that the following three requirements for standards are actually fulfilled:

- **Expression of a consensus:** the standard should reflect the expertise of a wide (possibly international) group of experts in the field
- **Publication:** the standard should be accessible to anyone who wants to know its content
- **Maintenance:** the standard is updated, replaced or deprecated depending on the evolution of the corresponding technical field.

Standards are not regulations. There is no obligation to follow them for research except when one actually wants to produce results that can be compared with those of a wider community. This is why a standardization policy

for an infrastructure in the Arts and Humanities should include recommendations as to what attitude the scholarly communities could or should adopt with regard to specific standards.

The preceding characteristics outlined for standards put a strong emphasis on the role of communities of practice and the corresponding bodies that represent them. Ideally, a good standard reflects the work of the relevant community and is maintained by the appropriate body. This is exactly the case of the [Text Encoding Initiative](#) with respect to text representation standards and, to a lesser extent, of [EAD \(Encoded Archival Description\)](#), whose maintenance is taken up by the [Library of Congress](#) with support of the [Society of American Archivists](#).

Because there is no obligation to use a given standard, it is essential to provide potential users with:

1. awareness about the appropriate standards and the interest to adopt them,
2. the cognitive tools to help them identify the optimal use of standards through selection and possibly customization of a reference portfolio.

The experience gained within the various communities and infrastructure represented in [PARTHENOS](#) that have been in the need of adopting existing standards, is that there is always an initial phase during which scholars should be made aware of some core standards that are systematically related to the definition of interoperable digital objects. This is basically what has lead us to identify the notion of Standardization Survival Kit (SSK). In the table below, for instance, we can see a first group of standards without which it is more or less impossible to deal with digital content in a proper way.

ISO 639 series	Codes for the representation of languages and language families
ISO 15924	Codes for the representation of scripts
ISO 3166	Codes for the representation of country names
IETF BCP 47	Standard for encoding linguistic content, combining ISO 639, ISO 15924 and ISO 3166
ISO 10646	Universal encoding of characters (unicode)
ISO 8601	Representation of dates and times
Extensible Markup Language (XML)	Provides the basic technical concept related to XML documents

The concept of SSK goes far beyond these baseline examples and aims at covering reference digital scenarios in the Arts and Humanities: a role of the SSK is to help communities participate in standardization activities where they exist, or at least document and spread the best practices as de facto standardized guidelines. Such a strategy will also contribute to the actual stabilization of existing conceptual and technical knowledge within ongoing projects, and provide a channel for the wider dissemination of the corresponding results.

1.2 The SSK: a toolkit for Humanities scholars

The work carried out by the SSK covers four types of activities related to the deployment and use of standards in the Humanities and Cultural Heritage fields:

- **Documenting** existing standards to provide a reference for scholars who want to find out more about their role and content. This relates to the specific provision of bibliographic sources, available documentation, specific targeted introductions, as well as providing prototypical examples which can serve as models for similar work, possibly made available through focused Virtual Research Environments within the [PARTHENOS](#) infrastructure;
- **Supporting** the actual adoption of standards by identifying how they relate to research scenarios and gathering the essential materials for controlling their deployment (e.g. schemas);
- **Communicating** with research communities so that they can be made aware of both the need to apply standards in their digital scholarly practices but also be informed of the essential standards for their own fields.

- **Training** for researchers, by giving them access to complete frameworks so that they may acquire knowledge and know-how on standardized methodologies.

In order to apply these four principles, the SSK focuses on giving researchers access to standards in a meaningful way. That is why it is built around research scenarios.

These scenarios are the core of the SSK because they aim at providing **contextual information** and relevant **examples** on how standards can be applied in a given research project. They potentially cover **all the domains of the Humanities**, from Literature to Heritage science, including History, Social sciences, Linguistics, etc.

They have been created and they are added to the SSK by domain experts, from **real life researcher-oriented use cases**), divided into different steps, and involving specific tasks.

These scenarios can be seen as a living memory of what should be the best research practices in a given community, made accessible and reusable for other researchers wishing to carry out a similar project but unfamiliar with the recommended tools, formats, methods to use, etc. For that reason, the SSK can be considered as a **complete framework** showing concrete use of standards, rather than simply a catalogue of resources.

1.3 Design principles & features of the SSK

From the very start of the project, the aim has been to build an **easy-to-use** online and collaborative platform with a **user-friendly** design. The idea of having general, end-to-end scenarios to help researchers carry out their project by following best practices and clear methods in their area of expertise is the most important design principle for the SSK.

The second principle is to add **context**: a “story-telling” approach to the use of standards in the Humanities and Social Sciences. The goal is to avoid providing yet another catalogue of resources, and offer instead contextual, **activity-based information** on how to use standards for researchers who are unfamiliar with them, but could see how they are used and what workflow they help achieve by following a scenario.

With these principles in mind, the SSK should enable the user, to perform two main actions:

1. **Consult and follow the guidelines expressed in the scenarios** you are interested in for your project. Finding the most relevant ones should be easy since the navigation relies on strong taxonomies covering the different aspects of research: the disciplines, the type of techniques and objects involved, the concrete activities carried out, the standards needed.
2. **Propose new scenarios** of your own by following a predefined model, with the possibility of both adding new content (steps as well as resources) and reusing existing content (to avoid duplication if a general step is already available in another scenario).

The first feature is fully operational. It was tested for the first time in April 2018, and iterations with test users have contributed to improve the **information readability** and **attractiveness**, in particular the exploration and search of scenarios.

The work on the second feature, allowing the user to contribute, is still ongoing. It is possible to create research scenarios with the SSK underlying data model, the Text Encoding Initiative, or TEI (see the dedicated section for more information). However, as we are aware that this solution requires some technical knowledge, a user-friendly interface is currently under development and should be released during the first trimester 2019.

Anyone who has registered and agreed to the “**Terms of use**” clearly stated will be able to contribute. This option has been chosen due to the difficulties of setting up some kind of editorial board in charge of reviewing any scenarios submitted. The **quality check** of the contributions should come from the very strict model that has to be followed in the scenario creation process. In addition, by contributing to the SSK, the user accepts to be visible and citable as an author, to be responsible for the work that he/she decides to share with others.

Why would you, as a researcher, want to contribute to the SSK? There are three main reasons:

- to make your research project align with the best practices in your community

- to get peer review and visibility
- to share a project in another form than the usual blog / article (a new way to disseminate your work).

1.4 SSK components: Scenarios < Steps < Resources

The SSK is a web platform builded on three main layers nested within each other following a specific order: Research scenarios, steps and resources.

Each **scenario** within the SSK works like a high-level research guide for scholars. They are made up of successive **steps** or tasks, and can be followed as a complete process to solve a given problem with the most standardized means. For each step, the appropriate **resources** to perform the given task are proposed, divided into two categories: the “**general resources**” that include the primary documentation and tools; and the “**project-specific resources**” that point to concrete use cases in which a similar task was accomplished. The material contained in these sections is of various kinds:

- The most important is the **state-of-the-art bibliography**, which includes all the documentation needed to carry out a given task. The bibliographical references are up-to-date and gathered within a [Zotero library](#), which was specially created for this project. This choice was made to ease the resource selection process and to allow for a collaborative watch and curation of relevant information. When the resource is available online, a direct link is provided; otherwise, the user is given all the necessary metadata.
- The SSK also offers the possibility to point to more **technical resources**, such as stylesheets, code samples, software or services.
- **Training materials** such as tutorials.

How to create a scenario for the SSK? (a SSK Tutorial)

The following instructions help users create a scenario for the SSK. These instructions are themselves conceived as a “scenario”, a step-by-step tutorial. First, contributors should be aware that:

- they can submit their new scenarios directly in TEI - see *The SSK data model (TEI)* section - and upload them on the SSK GitHub (<http://github.com/ParthenosWP4/SSK>), or by using the dedicated SSK contribution workspace (still work in progress).
- scenarios and steps follow the same data model. The difference is that a scenario points to a set of steps whereas a step points to a set of external resources.
- It is possible to create a scenario in any language but ideally, we would advocate for at least bilingual scenarios, with an additional version in English, in order to make it more visible. The forthcoming user interface will implement such behaviour and in TEI, it is possible to translate the prose, by duplicating the elements head, desc or term, and adding *xml:lang* attributes.

- For both, it is important to extend the acronyms cited and to briefly present the projects mentioned.

The form of this text should respect the following constraints:

- It shouldn't exceed 1500 characters (but should not be too short either).
- It is possible to point to external links. In TEI, use the following code:

```
<ref target="//url here//">text of the link</ref>
```

- Lists are also available. The TEI elements are `<list>` and `<item>`

2.2.1 References

- Universitat Autònoma de Barcelona. 'Describing a Process'. Coursera. Accessed 29 June 2018. <https://www.coursera.org/lecture/teaching-english/3-1-1-describing-a-process-mjuio>.
- Documentation of the TEI element `<desc>` *element*

2.3 Associate keywords to the scenario or the step

In order to enhance discoverability and search relevance, the SSK resources are described with a set of controlled vocabularies, particularly created for describing Humanities research. They are:

- **Research activities**, taken from Tadirah;
- **Research techniques**, taken from Tadirah;
- **Research objects**, taken from Tadirah;
- **Standards**, taken from the SSK Standard Knowledge base (supported by DARIAH-IT);
- **Disciplines**, taken from aureHAL

When editing the description of a scenario, the available keywords are:

- Research Techniques
- Research objects
- Standards
- Disciplines

For the steps, the most important keyword is the **Activity**, that should be unique for each step. It also possible to pick some **techniques**, **objects** and **standards**. For each keyword type, we recommend to choose between **1 and 4** terms.

2.3.1 References

- TaDiRAH - Taxonomy of Digital Research Activities in the Humanities
- Documentation of the TEI element `<term>` *element*
- List of all available terms: *Taxonomies*

2.4 Choose an illustration for the scenario

- The illustration must closely relates with the purpose of the scenario, i.e. not only with the discipline or the period studied.
- Screenshots are accepted.
- Landscape orientation image are recommended
- Maximum size : 2 Mo
- Accepted formats : png or jpg
- It must be published under the licence CC-BY or CC-0.

2.4.1 References

- [Unsplash](#), a gallery of free images and photos

2.5 Identify relevant resources

Identifying state of the art references is a prerequisite before actually add the resources to the steps. When we are talking about resources, we mean a standardized tool, service or document helpful for the task completion.

They take the form of a digital object and have to be of one the following types:

- Specification (normative document) of a standard (*spec*);
- Technical reports (*report*);
- Scholarly papers (*paper*);
- A single scholarly monograph (*book*);
- Lists of bibliographic references (*bibliography*);
- Scripts and code samples (*code*);
- Computing libraries (*library*);
- Collection of structured data (*database*);
- Computing tool, software (*tool*);
- Curating or hosting service (*service*);
- Tutorials or guidelines (*tutorial*);
- Blog posts (*blog*);
- Specialised mailing list, forum, etc (*community*).

The resources should be separated into several groups:

- First, general resources like standard specifications
- Second, project-related resources, i.e. how the standards are used in a real research project.

2.5.1 References

- [State of the art Wikipedia article](#)

2.6 Link the resources to the step

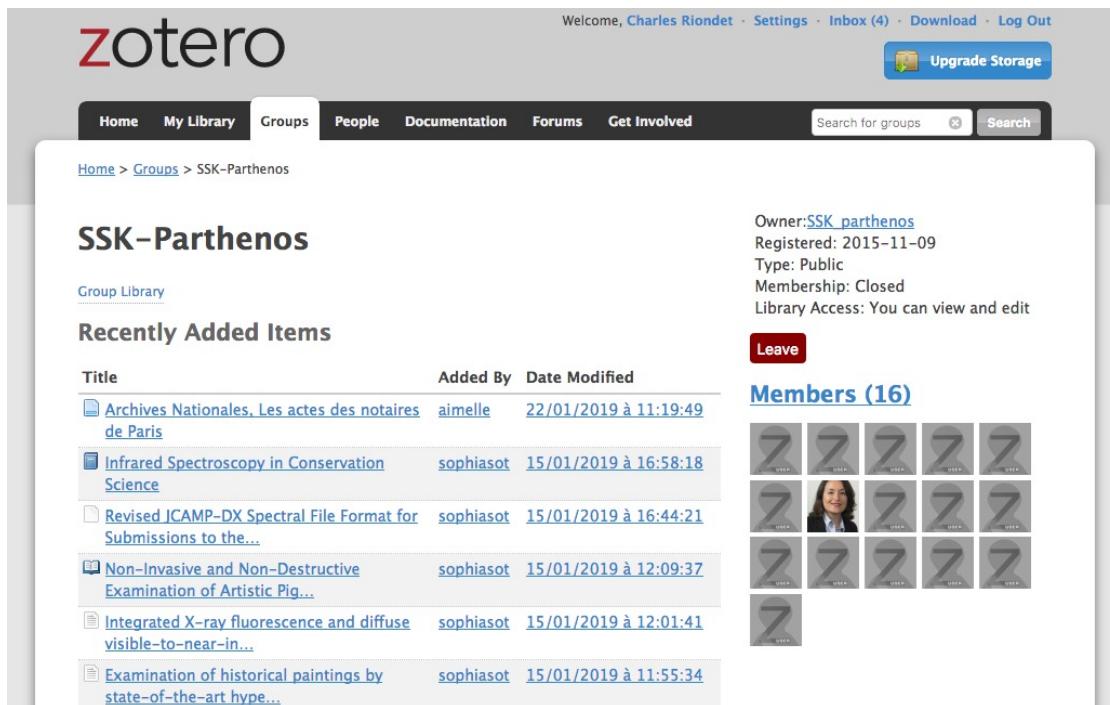
There is different ways to link resources to a step. The one we favour is the recording of the resource metadata in the dedicated SSK Zotero Library (see [here](#)).

The Zotero fields required by the SSK are:

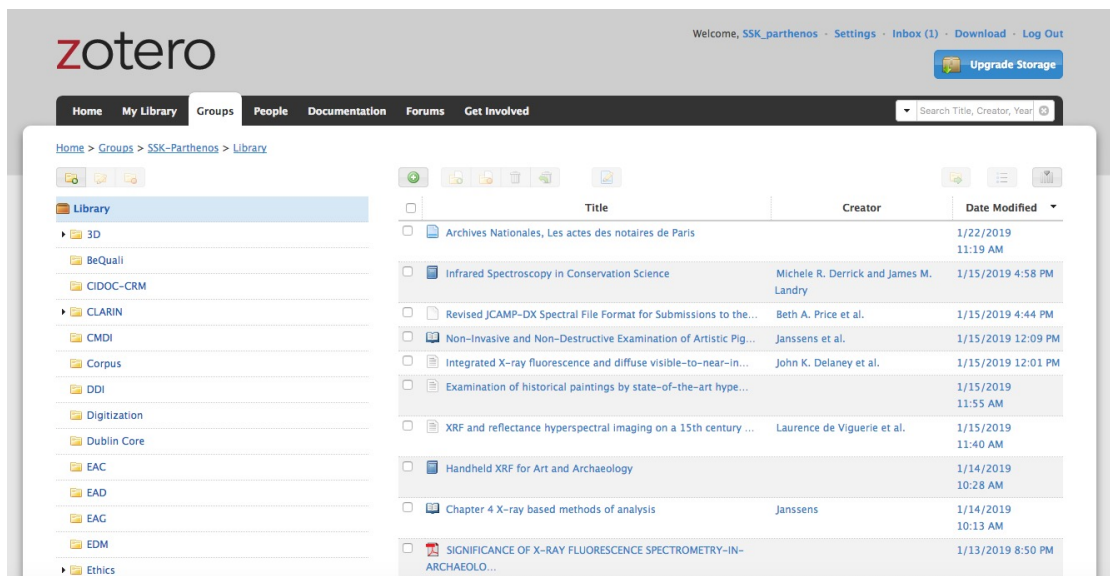
- `Item type`: The item type is most of the time identified by Zotero but it's important to check it. The most used item types are:
 - webpage
 - blogpost
 - journal article
 - book section
 - book
 - presentation
 - conference Paper
 - document
- `Title`
- `Author`
- `Date`
- `Url`: The url of the resource is most of the time mandatory (except when the resource is a printed document unavailable online).
- `abstract`: An abstract is very important to give an overview of what the user will find under the linked resource. Zotero is very often able to get this abstract from the webpage, but it may be necessary to check it, and sometimes create it.
- `Language`: the language of the resource

2.6.1 Add resource to the Zotero Library

1. To populate it, a Zotero account is necessary (create it [here](#)) as well as a membership in the SSK group (apply [here](#)).



The SSK library is organized in collections and sub-collections, by domains or standards. To learn more about how to use Zotero, many tutorial and learning resources are available [here](#).



2. Each group of resources (general and project specific) should be gathered: One group for the general resources and one group for each project. In the TEI, these groups are represented by `<linkGrp>` elements:
 1. One `<linkGrp>` for the general resources;
 2. one `<linkGrp>` for each project.
3. The references added to the Zotero Library are linked to the step with the help of their Zotero key, i.e. the last part of the URL of the resource record on the Zotero website.

For instance, in the following example, the key is 4B62GJ5I: <https://www.zotero.org/groups/427927/ssk-parthenos/items/itemKey/4B62GJ5I>. In TEI, the Zotero key should be used like this:

```
<ref type="zotero" key="4B62GJ5I" / >
```

2.6.2 References

- Documentation of the TEI element `<linkGrp>` *element*
- Documentation of the TEI element `<ref>` *element*

2.7 Advanced SSK functions (1) : customize a step

The SSK is adaptable by nature and contributors don't have to start from scratch their scenario. It is possible to create a scenario with existing steps as basis. But if the content of the step doesn't exactly fit, it is also possible to modify it, by updating the initial step (but with care), or, more safely, directly in the new scenario.

In TEI, the update of some elements is made with the help of the attribute `@mode`. See more in the section: *Reuse/customize a step*.

2.8 Advanced SSK functions (2) : link scenarios

Link scenarios together, or in other words, include a scenario (entirely or partially) into another is an interesting possibility when a scenario is a pre-condition or the continuation of another one. For instance, a scenario related to the preservation of 3D models can be preceded by a scenario explaining how to create such models.

The most common use cases are the following:

- Add a prerequisite scenario (as a first step)
- Associate a scenario that can be the follow-up of the current (as a last step)
- Insert a scenario (totally or partially) inside the current scenario, with the use of parameters that allows the user to choose which step of the external scenario should be included. See param.

The SSK data model (TEI)

The underlying data model of the SSK respects itself a standard, the Text Encoding Initiative. Each scenario and each step is encoded in TEI documents that are linked together with referencing mechanisms. This choice was made in order to ensure that the scenarios and the steps can be easily extended, reused and customized. The data model allows scenario creators to modify the structure of their research scenarios on the fly, by creating, removing or reordering steps. As steps are considered as autonomous objects in the architecture, they can be used in several scenarios. Customisation mechanisms are added to make sure that the information displayed is linked to the context of the scenarios as much as possible, namely according to disciplines, research objects and techniques.

3.1 The TEI specification

3.1.1 Schema

The SSK TEI specification takes the form of an ODD document, accessible on GitHub (https://github.com/ParthenosWP4/SSK/tree/master/spec/TEI_SSK_ODD.xml). This ODD specification allows us to generate a RelaxNG schema, also hosted on GitHub (https://github.com/ParthenosWP4/SSK/tree/master/spec/TEI_SSK_ODD.rng). This schema needs to be referenced in every TEI file, with this XML declaration:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-model href="https://raw.githubusercontent.com/ParthenosWP4/SSK/master/spec/TEI_
↪SSK_ODD.rng"
      type="application/xml"
      schematypens="http://relaxng.org/ns/structure/1.0"?>
```

3.1.2 Contribute to the SSK on GitHub with the TEI

Users willing to create scenarios in TEI can follow these instructions:

- Download or fork the SSK data repository in GitHub. It is necessary to have an account on GitHub: <https://github.com/ParthenosWP4/SSK/tree/master/> (NB: to fork a repository, a GitHub user account is necessary);

- Create your files with your favourite XML editor. Don't forget to validate them against the SSK schema (see above);
- To publish scenarios on the SSK, the TEI files need to be in the `scenarios` and `steps` folders;
- Users with a GitHub account can make a pull request to ask for the update of the repository. Users without an account can contact the SSK team at `ssk [at] inria [dot] fr`.

More info on GitHub [documentation](#).

3.1.3 Main Principles

A scenario is a list of events (`<listEvent>`) where each scenario step is an event (`<event>`). In order to use steps in several scenarios, they are stored in separated files.

- Scenario files gather `<event>` in a `<listEvent>` element, by referencing them with a `@ref` attribute. It is however possible to modify the content of the event called in a scenario (see *Reuse/customize a step*).
- The `<event >` element is the core of SSK scenarios. It contains the full description of scenario step:
 - A `<head>` that contains any label or heading used to identify part of a text, typically but not exclusively in a list or glossary.
 - A `<desc type="definition">` that contains a brief description of the object documented by its parent element, typically a documentation element or an entity.
 - Some descriptive terms following controlled vocabularies: `<term >` elements that contains a single-word, multi-word, or symbolic designation which is regarded as a technical term.
 - Bibliographical references with `<ref >`. They are gathered in `<linkGrp>` elements, that can be of two types :
 - * *general resources*
 - * *project specific* (one `<linkGrp>` per project)
- The attribute `xml:lang` is mandatory in all the content elements, namely `<head>` and `<desc type="definition">`. The authoritative list of registered language subtags is maintained by IANA and is available at <http://www.iana.org/assignments/language-subtag-registry>.

Example:

```
<head xml:lang="en">Create associated documentation</head>
<head xml:lang="fr">Création de la documentation associée</head>
```

3.2 Files naming conventions

These conventions have to be used when creating TEI files by hand, to avoid name duplicates.

3.2.1 For scenarios

1. SSK_sc for scenario
2. an underscore : _
3. a condensed title of the scenario in camel case: `myScenarioTitle`

example:

```
sc_myScenarioTitle.xml
```

3.2.2 For steps

1. the string step
2. an underscore; _
3. the initials of the step name, with the liaison words in lower case, and the meaningful words in upper case. For example, if a step title is : Searching for a relevant step title, it would give : SfaRST.
4. an underscore and the date (optional)

example:

```
step_SfaRST_10092018.xml
```

3.3 Scenarios and steps structure

3.3.1 Header

The `<teiHeader>` of each scenario and step files records important metadata, displayed in the SSK web application

Title

Note that the `<title>` element contain the title of the TEI document, and not the title of the scenario. This information is required by the TEI model.

Example:

```
<title>Scenario \"Creation of a TEI-based corpus\"</title>
```

Authors

The `<author>` element is repeatable. It contains:

- `<persName>`: the forename and surname of the author
- `<affiliation>`: the institution of the author.

Example:

```
<author>
  <persName>Mary Researcher-Name</persName>
  <affiliation>University of City</affiliation>
</author>
```

Licence

For scenarios and steps, it is possible to choose between two licenses:

- CC-BY (Creative Commons Attribution)
- CC-0 (Public Domain)

In TEI files, the license declaration is presented like this:

CC-BY

```
<availability>
  <licence target="http://creativecommons.org/licenses/by/4.0/">
    <p>The Creative Commons Attribution 4.0 Unported (CC BY 4.0) Licence applies to
      this document.</p>
  </licence>
</availability>
```

CC-0 (Public domain)

```
<availability>
  <licence target="https://creativecommons.org/publicdomain/zero/1.0/">
    <p>This document is in the public domain.</p>
  </licence>
</availability>
```

Revision history

It is possible to record the History of modification of the TEI files: addition of an author, of a step, etc.

```
<revisionDesc>
  <change when="2019-01-31">A new step between ...</change>
  <change when="2019-02-01">Addition of a new author</change>
</revisionDesc>
```

Full example of a <teiHeader>

```
<TEI type="researchScenario" xmlns="http://www.tei-c.org/ns/1.0">
  <teiHeader>
    <fileDesc>
      <titleStmt>
        <title>
          <!-- Title of the tei document, not title of the scenario -->
        </title>
        <author>
          <persName>...</persName>
          <affiliation>...</affiliation>
        </author>
        <sponsor>PARTHENOS</sponsor>
      </titleStmt>
```

(continues on next page)

(continued from previous page)

```

<publicationStmt>
  <authority>...</authority>
  <availability>
    <licence target="http://creativecommons.org/licenses/by/4.0/">
      <p>The Creative Commons Attribution 4.0 Unported
        (CC BY 4.0) Licence applies to this document.</p>
    </licence>
  </availability>
</publicationStmt>
<sourceDesc>
  <p>Created from scratch</p>
</sourceDesc>
</fileDesc>
<revisionDesc>
  <change when="2019-01-31">A new step between ...</change>
  <change when="2019-02-01">Addition of a new author</change>
</revisionDesc>
</teiHeader>

```

3.3.2 Scenarios

In a scenario file, `<event>` elements are used as pointers to link to full event elements stored in external files.

```

<listEvent>
  <event xml:id="s1" type="researchStep" ref="step_EaXswO_290517"/>
  <event xml:id="s2" type="researchStep" ref="step_Eprimrf_300517"/>
  <event xml:id="s3" type="researchStep" ref="step_Cad_300517"/>
  <event xml:id="s4" type="researchStep" ref="step_Tdats_300517"/>
  <event xml:id="s5" type="researchStep" ref="step_Sapditnf_300517"/>
</listEvent>

```

It is also possible to refer to another scenario, as a preliminary or a follow-up

```

<listEvent>
  <event type="researchScenario" ref="SSK_sc_digitization.xml" subtype="preliminary"/>
  <event xml:id="s1" type="researchStep" ref="step_KedKep_170717"/>
  ...
  <event type="researchScenario" ref="SSK_sc_Preservation.xml" subtype="followUp"/>
</listEvent>

```

It is possible to modify the content of an existing step directly in the scenario file. See *Reuse/customize a step* for more information.

3.3.3 Steps

Step files record the full description of the scenario step. Several elements have the same meaning and behaviour than those in scenario files. The main difference is the content of the `<event>` element. The main components of a `<event>` element are the description of the event, and the resources related to it.

- The description is recorded in the elements: `head` (see below) and `desc`;
- The resources are contained by one or several `linkGrp`.

3.4 Content of scenarios and steps

3.4.1 <head> element

The TEI head element record the title of a scenario or a step. It can be repeated to give as many translated versions as possible, with the attribute `xml:lang`.

```
<head xml:lang="en" type="scenarioTitle">Title of the scenario</head>
<head xml:lang="fr" type="scenarioTitle">Titre du scénario</head>
```

3.4.2 <desc> element

The element `<desc>` is used in two ways for the description of the scenarios and the steps. The distinction is made with the attribute `type`

- When the value of `type` is *definition*, the content of `<desc>` is a short text describing the scenario or the step;
- When the value of `type` is *term*, the content of `<desc>` is a set of term elements.

```
<desc type="definition" xml:lang="en">Description of the scenario</desc>
<desc xml:lang="en" type="terms">
  <term type="discipline" source="aurehal" key=""/>
  <term type="object" source="tadirah" key=""/>
  <term type="technique" source="tadirah" key=""/>
</desc>
```

3.4.3 <term> element

`<term>` elements are used to tag the scenarios, the steps and the resources, according to the SSK taxonomies, that are:

- Tadirah activities, objects and techniques
- the Dariah-IT Standard Knowledge base
- aureHAL disciplines

Functioning

These taxonomies are declared with the attributes `type` and `source`. The attributes of `<term>` elements are:

- The `type` attribute gives an information about the kind of term used.
- The `source` attribute sets a reference link for the taxonomy.
- The `key` attribute gives either an URI when the label of the term can be taken from or directly a label

Taxonomies

For each kind of `<term>`, the values of the attributes `type`, `source` and `key` are:

Term	type	source	List of possible terms: key
Research activities	activity	tadirah	http://ssk.huma-num.fr/#/glossary/activities
Research techniques	technique	tadirah	http://ssk.huma-num.fr/#/glossary/techniques
Research objects	object	tadirah	http://ssk.huma-num.fr/#/glossary/objects
Standards	standard	ssk	http://ssk.huma-num.fr/#/glossary/standards
Disciplines	discipline	aurehal	http://ssk.huma-num.fr/#/glossary/disciplines

NB: The value of the `key` attribute must be the exact same string the one displayed on the Glossary page. Use *copy & paste* to avoid trouble.

```
<term type="activity" source="tadirah" target="Encoding" />
```

3.4.4 <linkGrp> element

<linkGrp> is the container for the resources associated to a given step. It can have three attributes:

- The attribute `type` is required and can have two values:
 - *generalResources*: for resources that give general input about a standard, a protocol, ...
 - *projectResources*: for resources that show examples of real projects using the described standard, protocol, ...
- When `type` has *projectResources* for value, two more attributes are required:
- `source` for the name of the project mentioned
- `corresp` for a url pointing to or identifying the project

```
<linkGrp type="generalResources">
  <ref type="report" source="zotero" target="ZQVB6CIP" />
</linkGrp>
<linkGrp type="projectResources" source="CODATA" corresp="http://www.codata.org/">
  <ref type="report" source="zotero" target="G4UPDPG3" />
</linkGrp>
```

3.4.5 <ref> element

The <ref> elements gathered in <linkGrp> are used to point to resources of the SSK Zotero Library. See the section *How to create a scenario for the SSK? (a SSK Tutorial)*, to learn how to work with Zotero and the SSK. The attributes for <ref> are `type`, `source` and `target`.

- The attribute `type` is required. Its values are taken from the Zotero item types, plus SSK specific values. Possible values are:
 - *spec*: the specification (normative document) of a standard;
 - *report*: technical reports;
 - *blog*: blog posts;
 - *tutorial*: tutorials or guidelines;
 - *code*: Scripts and code samples;
 - *paper*: Scholarly papers;
 - *library*: Computing libraries;

- *bibliography*: A list of bibliographic references
 - *database*: collection of structured data
 - *tool*: Computing tool, software;
 - *service*: Curating or hosting service.
 - *community*: Specialised mailing list, forum, etc.
 - *book*: A single scholarly monograph.
- The `source` attribute in `<ref>` records that the `target` refers to a zotero ID. The value of `source` is most of the time `zotero`
 - The `target` attribute specifies the destination of the reference, i.e. the ID of the Zotero record. This ID is last part of the URL of the Zotero online record:

<https://www.zotero.org/groups/427927/ssk-parthenos/items/itemKey/BEVAWMPX>

A relatively easy way to quickly get this ID when you just added the resource to zotero is to go to the SSK group page, that lists the recently added items: <https://www.zotero.org/groups/427927/ssk-parthenos>

Example:

```
<ref type="spec" source="zotero" target="BEVAWMPX"/>
```

3.5 Reuse/customize a step

One of the main features of the SSK is the possibility of reusing a step in several scenarios. However, it may be needed to update slightly the scope of an existing step when reusing it in a new scenario. When updating a step, it is important to understand that every changed element must be expressly specified.

To declare that the selected step is updated, an attribute `mode`, with the value `change` is added to the `<event>` element:

```
<TEI type="researchScenario" xmlns="http://www.tei-c.org/ns/1.0">
  ...
  ...
  <listEvent>
    ...
    <event xml:id="s4" type="researchStep" ref="step_Tdats_300517"/>
    <event xml:id="s5" type="researchStep" ref="step_Sapditnf_300517" mode="change">
      ...
    </event>
  </listEvent>
```

The elements that can be updated, also specified with the `mode` attribute, are the following:

- The Standards (`<term type="standard">`, within `<desc type="terms">`);
- The Resources (`<linkGrp>` and `<ref>`).

For these elements, the modification must be explicit :

- each new element must be specified with `mode="add"`;
- every removed element from the original step must be specified with `mode="delete"`.

3.5.1 Update standards

```
<event xml:id="s5" type="researchStep" ref="step_Sapditnf_300517" mode="change">
  <desc xml:lang="en" type="terms" mode="change">
    <term source="standardList" type="standard" key="CIDOC-CRM" mode="add"/>
    <term source="standardList" type="standard" key="LIDO" mode="delete"/>
  </desc>
```

3.5.2 Update resources

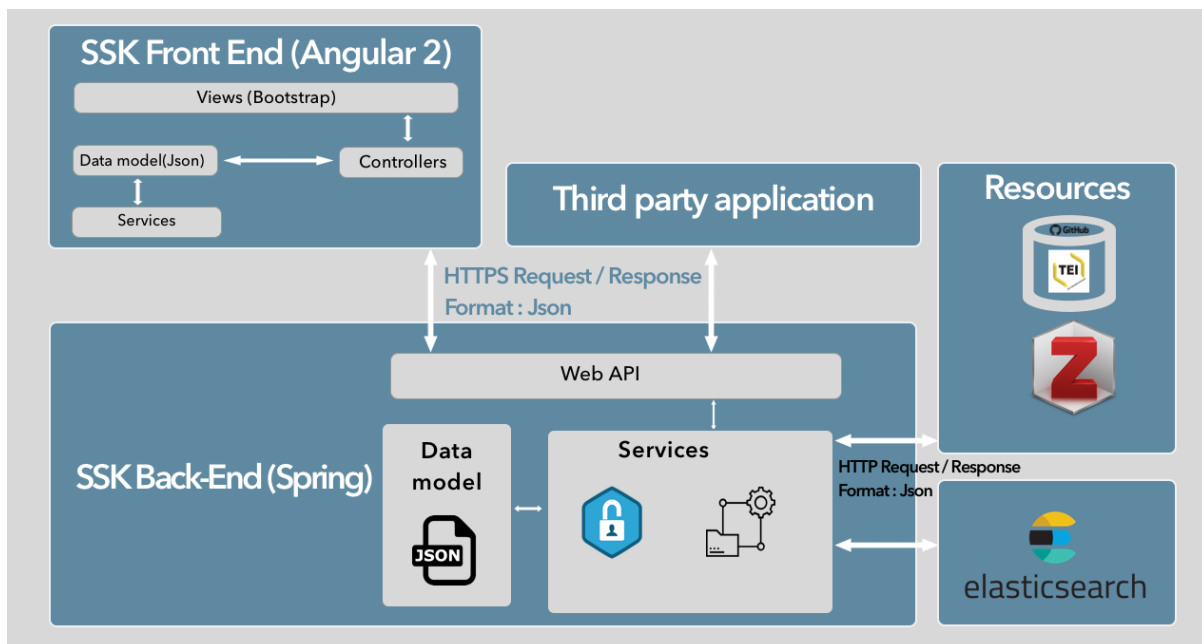
```
<event xml:id="s5" type="researchStep" ref="step_Sapditnf_300517" mode="change">
  <linkGrp type="generalResources">
    <ref type="code" source="zotero" target="9SKJDJKS" mode="add"/>
    <ref type="code" source="zotero" target="9SKORJKS" mode="delete"/>
  </linkGrp>
  <linkGrp type="projectResources" source="CODATA" corresp="http://www.codata.org/">
    <ref type="code" source="zotero" target="9SKJDJKS" mode="add"/>
    <ref type="code" source="zotero" target="9SKORJKS" mode="delete"/>
  </linkGrp>
</event>
```

 Technical architecture

The implementation of the SSK is based on a flexible, easy to deploy and maintain architecture. It is composed of independent entities that communicate together through services (REST / JSON).

- The main entity, the core of the SSK, is the back-end, which makes queries to our data repositories (Github, Zotero, etc.) and processes retrieved data.
- This core/back-end communicates with a search engine, part of the architecture, based on Apache Lucene: ElasticSearch.
- The data processed from the core part and from the search engine are all delivered via an API to third-party applications like the SSK interface (the front-end), which is an entity of our architecture.

The architecture of the SSK is depicted in the following schema:



4.1 SSK's input

The SSK processes TEI files stored on Github and divided into two folders, `scenarios` and `steps`. For more information about the data model, check the dedicated section: *The SSK data model (TEI)*.

4.2 SSK Back-end

This part is the main component of the SSK, it has been built using **Spring Boot version 1.5.4.RELEASE**, a Java based framework (more details [here](#)). It contains modules for :

4.2.1 1 - Processing SSK data

This means retrieving TEI content from SSK Github repository. A very important step is to validate the retrieved content according to the `RELAX NG` schema defined for SSK files.

If the file is validated, TEI content is converted in JSON format using the built-in scripts provided by the TEI consortium : <https://github.com/ParthenosWP4/SSK/blob/dev/SSK-Server/src/main/resources/lib/bin/teitojson>.

The references contained in TEI files are resolved to complete the data:

- For keywords, in particular the `standards`, the data retrieved in the TEI files is used to get extra information about the terms. For standards, a knowledge base of standards is queried to retrieve more informations (standard complete name, multilingual standard description and links).
- For resources, queries are made on platforms such as `Zotero` and `GitHub` (for project resources). **Website scraping** is also used to make resources more consistent.

Once the data is completed, it is then pushed into for indexing and search.

Note that each scenario and its steps are also pushed on . Each step is linked to the scenarios it is part of, by a parent attribute, directly in . Resources and descriptive metadata have been also targeted with their parent identifier in the same way.

4.2.2 2 - API serving

The SSK Back-End makes its data available via a REST API, built with Spring boot. This API allows third party applications to retrieve scenarios, steps, resources and their descriptive metadata.

The API V1 is accessible via the URL: http://ssk.huma-num.fr/ssk_services-0.0.1/ssk

Example queries:

- Get all the standards mentioned in scenarios and steps: http://ssk.huma-num.fr/ssk_services-0.0.1/standard/all
- To get the descriptive metadata of a scenario, the query is composed of the keyword `scenario`, the Id of the scenario, and, as parameters, the metadata the user wants to get. For instance, the following query serves the title, description, image and descriptive terms for the scenario whose ID is `SSK_sc_DisseminationFieldSurveys`:

```
http://ssk.huma-num.fr/ssk_services-0.0.1/scenario/SSK_sc_DisseminationFieldSurveys?fields=title,desc,image,scenario_metadata,author&fromSSK=true
```

A V2 is planned in order to serve more easy-to-handle content, for instance giving all the scenarios with a given author or related to a given institution, discipline or standard, etc.

4.2.3 3 - User management

This part is still work in progress. We plan to deliver it by the **first trimester 2019**.

Creating an account will allow the user to:

- bookmark scenarios and steps in order to facilitate future navigations or stay in touch with some specific research fields.
- Create their own scenarios based on existing ones or by starting from scratch.

4.2.4 4 - Creation and update of scenarios

We mentioned that it will be soon possible to create or update scenarios from forms in the Front-End part of SSK. Meanwhile, it is also possible to do so on our GitHub repository, via directly updating the TEI files.

To take into account these modifications or updates in our Elasticsearch search engine, we use [GitHub webhooks](#) which allows us to make POST requests to the Back-End. Thus an end-point in this module of SSK (Back-End) receives the GitHub request and its data, processes these data so that it can be pushed into Elasticsearch.

4.2.5 5 - Search Engine

The search engine module has been built in order to allow refined information retrieval. It relies on [Elasticsearch](#), version 6.2.4, a full-text search and analytics engine, that allows us to easily propose multi-criteria and full-text queries to the users, but also autocomplete suggestions.

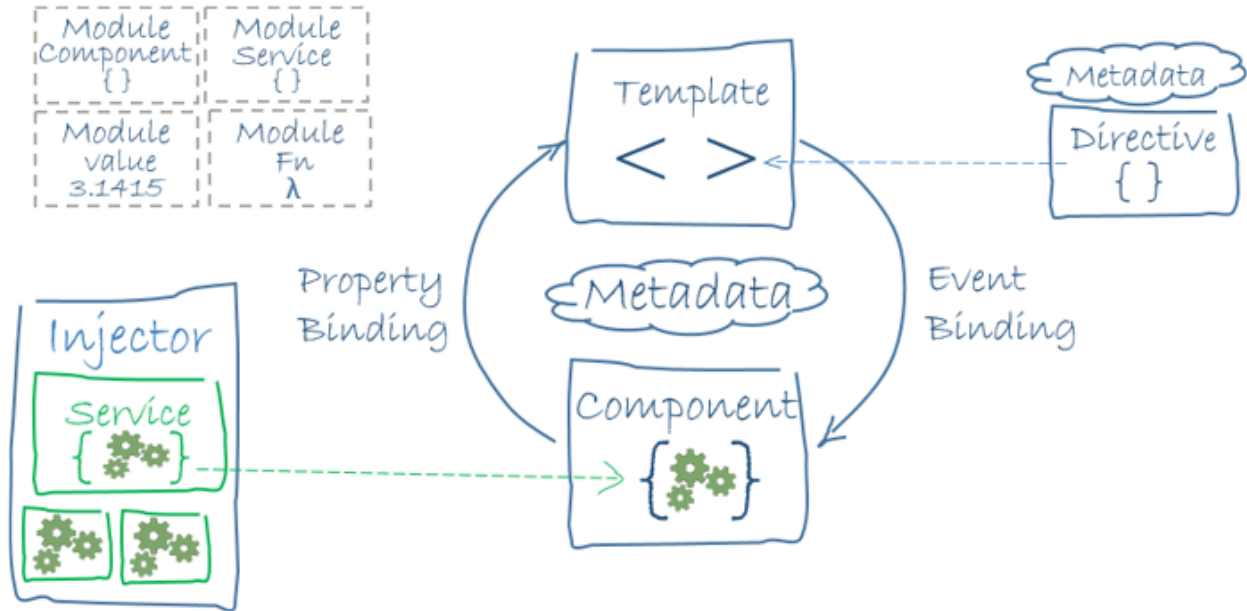
The indexed data is not only the information contained in the scenarios and steps descriptions stored in TEI files, but also the data hosted on the [Zotero database](#) and the [SSK Standards Knowledge Base](#).

4.3 SSK Front-End

The SSK Front-End is the client part of the SSK, where users can see the SSK data (TEI files + Zotero references).

It is built with [AngularJS](#), a framework using for building web applications. Angular proposes to set a hierarchy of components (or classes), associated with HTML templates. Components use services (or functions) to communicate with the server (to fetch the data for example) and to link components between them.

The image below (taken from <https://angular.io/guide/architecture>) shows the architecture of an Angular application.



To display SSK’s data on the web interface, we created several components, services and templates.

Components, combined with templates, are used to represent the different SSK layers : scenarios, steps, resources.

Services are used to share data between these layers, but they also allowed us to design functions that queries data from main modules of the SSK (Core SSK or Back-End) via a REST API.

4.4 Deployment

As the SSK is based on three main parts, each of its modules (Elasticsearch, Front-End, Back-End) need to be deployed independently. The communication between the modules is made with dedicated endpoints. The Elasticsearch endpoint serve the data to the Back-end, and in the same way, the Back-End also offers an endpoint to the Front-end so that it can get SSK’s data for display.

4.4.1 1 - Install Elasticsearch


The binary packages of Elasticsearch have only one dependency: Java. The oldest supported version is Java 8. To download and install Elasticsearch, use the commands that work with your system (deb for Debian/Ubuntu, rpm for Redhat/Centos/Fedora, mac for OS X, and win for Windows). Follow for more details. After installing Elasticsearch, run it with the `./elasticsearch` command from the `/bin` installation folder of Elasticsearch. and After that run this `curl -XPUT http://localhost:9200/ssk?pretty` to create SSK’s index in Elasticsearch.

4.4.2 2 - Run SSK in Local

First be sure you have **gradle** installed on your computer, the actual version on which the ssk is running on local computer is **4.1**. Clone the `dev` branch on <https://github.com/ParthenosWP4/SSK> and you’ll have many folder but the more updated version of the SSK are respectively **SSK-Server** and **SSK_Client** for the Back-end and the Front-end.

Open each part with your favorite IDE. *IntelliJ IDEA* is good for Back-end and *VisualStudio code* for the Front-End. Use gradle to install dependencies and run the Back-end with your IDE. The server-side of the SSK will then be available as an API service for the Front-End. To run the Front-end part, insure to have the following configuration on your computer for angular project by running `ng -version`.

```
(base) MP-66090:client ltadonfo$ ng -version
```



```
Angular CLI: 6.2.9
Node: 10.0.0
OS: darwin x64
Angular: 6.1.10
... animations, common, compiler, compiler-cli, core, forms
... http, language-service, platform-browser
... platform-browser-dynamic, router
```

Package	Version
@angular-devkit/architect	0.13.9
@angular-devkit/build-angular	0.13.9
@angular-devkit/build-optimizer	0.13.9
@angular-devkit/build-webpack	0.13.9
@angular-devkit/core	7.3.9
@angular-devkit/schematics	0.8.9 (cli-only)
@angular/cli	<error>
@ngtools/webpack	7.3.9
@schematics/angular	0.8.9 (cli-only)
@schematics/update	0.8.9 (cli-only)
rxjs	6.2.2
typescript	2.9.2
webpack	4.29.0

When all is ok, run **npm install** to install modules for the angular SSK Front-end, run the projet with `ng serve` and the application will be accessible through <http://localhost:4200/>.

4.4.3 3 - Build and deploy SSK on a remotre server (Huma-num in this case)

The Back-End is composed of two main elements : **Spring Boot** and **Tomcat** (Java Servlet Container).

Spring Boot is a *convention over configuration* framework that allows us to set up a production-ready setup of a Spring project. By default, Spring Boot builds a standalone Java application that can run as a desktop application or be configured as a system service. For the SSK, we use it as a service.

Opposite to standalone applications, Tomcat is also installed as a service that can manage multiple applications within the same application process, avoiding the need for a specific setup for each application.

The SSK spring boot application use as build automation system.

3.1 Build and deploy Back-End

To build a Tomcat-deployable WAR application for the Back-end part of the SSK, run the following gradle command `gradle build -PonD4Science=false`.

The WAR will be generated in `target/ssk_services.war` (assuming the Gradle artifactId is `ssk_services`).

To have our WAR file deployed on Huma-num Tomcat SSK virtual machine, we need to complete the following steps:

1. First you will need credentials from Huma-num to access the SSK virtual machine
2. Copy our WAR file from `target/ssk_services.war` to the tomcat folder with a `scp` command like this : “ `scp build/libs/ssk_services-0.0.1.war user_name@ssk.huma-num.fr:resource/tomcat/current/webapps`”
3. The default tomcat of SSK virtual machine is hot deployment means that the server part (war file) will be automatically deployed just at the end of the `scp` copy. Otherwise you could follow [Hummum!](#) for more information on Huma-Num virtual Machine.

This is how the SSK Back-End has been deployed on the Huma-Num infrastructure. There is already a SSK'Elasticsearch server on Huma-Num.

Source:

4.4.4 3.2 - Front-End Deployment (Angular application)

Build and deploy the Front-end of the SSK which is an Angular based application, requires to be installed on your computer.

The steps to follow are:

1. To build angular applications, execute the `ng build --prod --base-href / --configuration=prod-huma-num` command. This will generate files in the `dist` folder located at the root of the application folder.
2. Copy the whole content of `dist` folder to a folder on the server with `scp` as follow : `scp -r dist/* user_name@ssk.huma-num.fr:www/web_main`

More details .

The **Standardization Survival Kit** has been developed in the context of the PARTHENOS project.

The PARTHENOS project

PARTHENOS stands for “Pooling Activities, Resources and Tools for Heritage E-research Networking, Optimization and Synergies”. It is a research infrastructure whose objective is to strengthen the cohesion of research across a number of related fields associated with the humanities. This broad sector includes linguistic studies, cultural heritage, history and archaeology and existing research structures such as ARIADNE (archaeology), CLARIN (languages) and DARIAH (arts and humanities) are members of PARTHENOS.

To know more about the partners please visit the dedicated page.

Within the PARTHENOS infrastructure, the Work Package 4 (WP4) addressed the standards set up by community needs and defined, managed and maintained by international initiatives. The standards concern all the aspects of digital activity, from individual items to documents and collections.

The Standardization Survival Kit

The design and development of the **Standardization Survival Kit** has been carried out since 2016 by [PARTHENOS WP4 members](<http://www.parthenos-project.eu/consortium>) (PIN, CLARIN, KNAW, CNR, CSIC, FORTH, OEAW, MIBACT-ICCU, SISMELE, Academy of Athens, FH Potsdam, Huma-Num, Inria) and lead by Inria (team ALMAnaCH).

- Laurent Romary (Project Leader) Directeur de Recherche at Inria (France), within the team ALMAnaCH, and former director general of DARIAH (2014-2018). He carries out research on the modelling of semi-structured documents, with a specific emphasis on texts and linguistic resources. He has been active in standardisation activities within ISO committee TC 37 and the Text Encoding Initiative. He has also been working since many years on the advancement of open access.
- Charles Riondet (project management, data modelling) History Ph.D. and Archivist. Research Engineer in Digital Humanities at Inria in Paris, member of the team ALMAnaCH. Metadata and standards specialist, with a focus on the standardization of textual data (XML-TEI), archival metadata (EAD, EAC-CPF) and research workflows.
- Marie Puren (until 2018) (project management, data modelling): Postdoctoral researcher in Digital History at the CNRS' Laboratory for Historical Research - Rhône Alpes (LARHRA), and research associate at the Bibliothèque Nationale de France. She received her PhD in History at the Sorbonne University, and is a lecturer in Digital Humanities at the Ecole Nationale des Chartes and at the Université de Versailles - Saint Quentin. She is also a responsible for the French version of the Programming Historian.
- Dorian Seillier (UX designer): Master's degree in Medieval History and in Information Architecture at the ENS de Lyon. He is currently a member of the Team ALMAnaCH at Inria Paris, where he works as a UX Designer / Information Architect.
- Lionel Tadjou (lead developer): Holds a Master Degree in Computer science with major in software engineering. He is currently research engineer at the French National Institute for Computer Science and Applied Mathematics (Inria) in Paris, and member of the ALMAnaCH team where he works as a lead developer. He is involved in European Union funded Horizon 2020 projects PARTHENOS
- Damien Biabiany (developer): Student at the Coding Factory By Itescia, developer at Inria Paris with a work-study contract. I joined the ALMAnaCH team in December 2018 for the PARTHENOS project. I work on the Front-end development of the SSK and the contribution feature.

CHAPTER 7

The SSK on Github

- The branch *master* contains the data,
- The branch *dev* contains the back-end and the front-end.

This project has been developed within the H2020 PARTHENOS project.

 <http://creativecommons.org/licenses/by/4.0/>