POLITECHNIKA GDAŃSKA

Software Management Plan

Plan Zarządzania Oprogramowaniem – podstawy, dobre praktyki, narzędzia

Szkolenie finansowane w ramach "Działań na rzecz promowania i zwiększenia świadomości w zakresie Otwartych Danych Badawczych", które stanowią element realizacji programu „Inicjatywa Doskonałości – Uczelnia Badawcza” (IDUB) w zakresie podniesienia poziomu jakości działalności naukowej uczelni w ramach Działania I.4 (Działania na rzecz zwiększenia liczby publikacji w prestiżowych czasopismach i wydawnictwach).

POLITECHNIKA GDAŃSKA

CEL SZKOLENIA

- Koncepcja i znaczenie Software Management Plan
- Kluczowe elementy SMP i ich rola
- FAIR4RS
- Narzędzia i praktyki

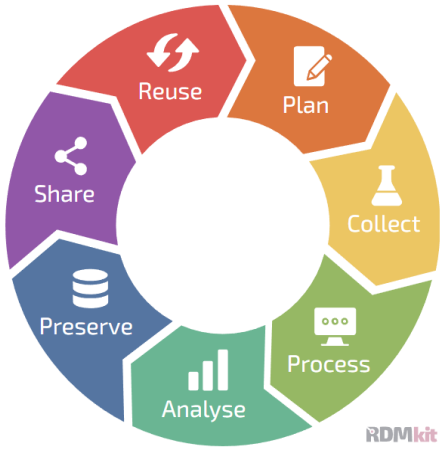
Szkolenie finansowane w ramach "Działań na rzecz promowania i zwiększenia świadomości w zakresie Otwartych Danych Badawczych", które stanowią element realizacji programu „Inicjatywa Doskonałości – Uczelnia Badawcza” (IDUB) w zakresie podniesienia poziomu jakości działalności naukowej uczelni w ramach Działania I.4 (Działania na rzecz zwiększenia liczby publikacji w prestiżowych czasopismach i wydawnictwach).

POLITECHNIKA GDAŃSKA

Software Management Plan (SMP)

Software Management Plan (SMP) to dokument opisujący, w jaki sposób konkretny projekt oprogramowania /badawczego/ będzie:

- ✓ Rozwijany i utrzymywany – strategia tworzenia, aktualizacji i wspierania kodu
- ✓ Licencjonowany i publikowany – wybór licencji i kanały dystrybucji
- ✓ Dokumentowany dla użytkowników – instrukcje instalacji, użytkowania
- ✓ Archiwizowany i utrzymywany długoterminowo



Grafika: https://rdmkit.elixir-europe.org

- ➢ SMP opracowywany jest na początku projektu razem z DMP.
- ➢ Obecny i kluczowy w całym Cyklu zarządzania danymi / tam gdzie występuje oprogramowanie/
- ➢ **SMP to żywy dokument,** który ewoluuje wraz z projektem i powinien być regularnie aktualizowany.

Practical guide to Software Management Plans: https://doi.org/10.5281/zenodo.7589725

POLITECHNIKA GDAŃSKA

Miejsce SMP w Research Data Management Lifecycle

PLAN

Definiuje strategię zarządzania oprogramowaniem na całą długość projektu.

COLLECT

Może być aktualizowany – gdy oprogramowanie jest faktycznie rozwijane, mogą pojawić się nowe wymagania (np. nowe narzędzia, zmiany w zespole)

PROCESS & ANALYZE

Wspiera jakość – określa standardy testowania, dokumentacji i wersjonowania, które są stosowane w trakcie analizy.

PRESERVE & SHARE

SMP określa sposób archiwizacji, DOI, licencjonowania i ogólnego udostępniania oprogramowania.

SMP (Software Management Plan) porządkuje cały cykl życia oprogramowania: od pomysłu, przez rozwój i udostępnienie, po utrzymanie i wycofanie. Zwiększa jakość, dostępność i ponowne wykorzystanie kodu, a więc również przejrzystość i powtarzalność badań

Grafiki: https://rdmkit.elixir-europe.org

POLITECHNIKA GDAŃSKA

Kluczowe cechy SMP

- **SMP to nie biurokracja** – to praktyczne narzędzie do lepszego zarządzania oprogramowaniem
- **SMP wspiera produktywność** – gdy oprogramowanie jest dobrze udokumentowane i archiwizowane, wyniki mają szansę być powtarzalne
- **SMP pomaga wdrożyć najlepsze praktyki** podczas tworzenia oprogramowania i zapewnia, że będzie ono dostępne oraz możliwe do ponownego użycia w krótkim i dłuższym okresie
- **SMP przyczynia się do odtwarzalności** wyników badań
- **SMP dotyczy KAŻDEGO oprogramowania** – zarówno prostych skryptów, jak i rozbudowanych aplikacji
- **SMP ewoluuje** – początkowy plan może być prosty (8 pytań), a następnie rozbudowywany

SMP uzupełnia DMP – razem tworzą pełną strategię badawczą dla projektu

POLITECHNIKA GDAŃSKA

Osiem kluczowych pytań SMP

8 pytań stanowi „minimum SMP” – zestaw, na który powinien umieć odpowiedzieć każdy zespół tworzący oprogramowanie badawcze, niezależnie od skali projektu.

1 Jakie oprogramowanie będziesz tworzyć/rozwijać?

Jaki problem rozwiązuje, jaka jest jego główna funkcja i w jakim kontekście badawczym będzie używane?

Typ: skrypt, biblioteka, narzędzie CLI, aplikacja webowa, workflow.
Zakres: prototyp na czas projektu czy oprogramowanie o znaczeniu długoterminowym?, dedykowane dla danego badania i danych
Języki i technologie: np. Python + R + Docker – istotne dla późniejszej interoperacyjności.

2 Kim są docelowi użytkownicy?

Czy oprogramowanie jest tylko „dla mnie”, dla zespołu, dla szerszej społeczności naukowej, czy „dla każdego”?

Poziom umiejętności: programiści, analitycy danych, naukowcy, „zwykli ludzie”
Dziedziny: biomedycyna, inżynieria, nauki społeczne
Czy przewidujesz zewnętrznych współtwórców?

3 Jak udostępnisz oprogramowanie?

Gdzie będzie dostępny kod lub pakiety (repozytorium, platforma), na jakiej licencji i w jakiej formie (kod źródłowy, paczki, aplikacja/narzędzie web)

Repozytoria: GitHub, GitLab, instytucjonalny Git,
Licencje: MIT, Apache-2.0, GPL – kompatybilność z wymogami grantodawców.

4 Jak oprogramowanie przyczyni się do realizowanych badań?

Jaką wartość naukową wnosi, jakie zagadnienia badawcze rozwiązuje, jakie ograniczenia usuwa?

POLITECHNIKA GDAŃSKA

Osiem kluczowych pytań SMP

8 pytań stanowi „minimum SMP” – zestaw, na który powinien umieć odpowiedzieć każdy zespół tworzący oprogramowanie badawcze, niezależnie od skali projektu.

5 Jak będziesz mierzyć wkład/znaczenie oprogramowania?

Jakie wskaźniki wykorzystasz, aby ocenić użycie i wpływ oprogramowania?

Miary ilościowe: cytowania w publikacjach, pobrania pakietu, znaczniki ilościowe na GitHubie.
Wykorzystanie w innych projektach.
Czy raportowanie jest konieczne i potrzebne?

6 Jak będziesz wspierać użytkowników?

Jaką dokumentację, kanały kontaktu i procedury wsparcia zapewnisz?

Dokumentacja użytkownika: README, tutorial
Kanały wsparcia: issue tracker, dyskusje, e-mail
Polityka wsparcia: jaki jest deklarowany poziom wsparcia po zakończeniu projektu i czy jest konieczny?
README to minimum

7 Jak Twoje oprogramowanie jest powiązane…?

Jakie są zależności od innych bibliotek, usług i danych oraz jak oprogramowanie powiązane jest z publikacjami i zbiorami danych?

Zależności od bibliotek i usług (np. bazy danych, REST API) oraz ich licencje.
Powiązania z danymi: jakie zbiory danych wymagane są do działania, jak są wersjonowane i cytowane.
Powiązania z publikacjami i innym oprogramowaniem

8 Jak zapewnisz długoterminową dostępność i archiwizację?

Gdzie zarchiwizujesz oprogramowanie, jak zadbasz o wersjonowanie i identyfikatory i dostępność długoterminową?

Archiwa: Zenodo (publikacja, wersje i DOI), instytucjonalne repozytoria (Research Software Directory), Software Heritage.
Wersjonowanie: tagi w Git, opis zmian (CHANGELOG, release notes)

POLITECHNIKA GDAŃSKA

Wymagania dla SMP

Figure 1. Software Management Plan requirements grouped by their focus.

- ✓ **Cel (Purpose) i kontekst użycia**
  Krótki opis oprogramowania: co robi, jaki problem rozwiązuje, dla kogo jest przeznaczone i jakie ma ograniczenia.
- ✓ **Wersjonowanie i jakość (Version control, Testing, Software engineering quality)**
  Zastosowanie systemu kontroli wersji (np. Git), strategia tagowania/releasów, podstawowe testy, zasady przeglądu kodu.
- ✓ **Licencjonowanie i kompatybilność (Software licensing and compatibility)**
  Wybór licencji, zgodność z innymi licencjami, zależności, polityka ponownego użycia, zgodność z wymaganiami instytucji finansujących
- ✓ **Cytowanie i uznanie (Citation, Software compatibility)**
  Zapewnienie sposobu cytowania (np. CITATION.cff, DOI), powiązanie z publikacjami i danymi badawczymi
- ✓ **Dokumentacja (User, Developer, Deployment documentation)**
  Dokumentacja użytkownika (README, podręcznik), dokumentacja wdrożeniowa (jak uruchomić/instalować), dokumentacja deweloperska (jak rozwijać, struktura kodu).
- ✓ **Repozytorium i archiwizacja (Repository, Packaging, Preservation)**
  Wybór repozytorium (GitHub/GitLab), sposób pakowania (pakiety, kontenery), miejsce i forma archiwizacji długoterminowej (Zenodo, Software Heritage).
- ✓ **Wsparcie, zasoby, utrzymanie (Support/Resources, Maintenance)**
  Plan wsparcia użytkowników, kto odpowiada za projekt, jaki jest poziom wsparcia i przez jaki czas; dostępne zasoby (czas, ludzie, infrastruktura).
- ✓ **Zarządzanie ryzykiem (Risk analysis)**
  Identyfikacja głównych ryzyk (brak personelu, zależności, infrastruktura, bezpieczeństwo, koszty)

Martinez-Ortiz, C., Martinez Lavanchy, P., Sesink, L., Olivier, B. G., Meakin, J., de Jong, M., & Cruz, M. (2023). Practical guide to Software Management Plans (1.1). Zenodo. https://doi.org/10.5281/zenodo.7589725

POLITECHNIKA GDAŃSKA

SMP - FAIR4RS

FAIR4RS (FAIR for Research Software) to zbiór zaleceń opracowany w 2022 roku przez społeczność ok. 500 osób z 110 organizacji w 34 krajach. Stanowi rozszerzenie oryginalnych zasad FAIR (Findable, Accessible, Interoperable, Reusable) specjalnie dla oprogramowania naukowego.

Findability

Oprogramowanie i jego metadane muszą być łatwe do znalezienia dla ludzi i maszyn

- Unikalne, trwałe identyfikatory (DOI)
- Bogate metadane (CodeMeta)
- Rejestracja w repozytoriach (Zenodo, GitHub)

✓ SMP: Pytanie 3 (Jak udostępnisz?) + Pytanie 8 (Gdzie zdeponujesz?)

Accessibility

Oprogramowanie musi być łatwo dostępne

- Jasne warunki licencji (MIT, Apache, GPL)
- Dostęp przez standaryzowane protokoły (HTTP, Git)
- Archiwizacja długoterminowa (Software Heritage, Zenodo)
- Dokumentacja dla instalacji i użycia (README)

˘ SMP: Pytanie 3 (licencjonowanie) + Pytanie 6 (wsparcie)

Interoperability

Oprogramowanie powinno współdziałać z innymi narzędziami i danymi

- Metadane w standardowych formatach (JSON-LD, RDF)
- Udokumentowane deklaracje zależności (requirements.txt)
- Zgodność ze standardami dziedzinowymi (Bioschemas)
- Interfejsy publiczne i dokumentacja API

˘ SMP: Pytanie 7 (relacje, powiązania)

Reusability

Oprogramowanie powinno być zrozumiałe, modyfikowalne i mieć możliwość ponownego użycia

- Bogate metadane opisujące funkcjonalność
- Dokumentacja deweloperska i kodu
- Informacje o cytowaniu (CITATION.cff)
- Jasne warunki licencji i referencje do zależności

˘ Pytania 4–6 (wpływ, pomiary, wsparcie) + dokumentacja

SMP odpowiada na pytania, które bezpośrednio wspierają osiągnięcie zasad FAIR4RS w konkretnym projekcie.

Chue Hong, N. P., Katz, D. S., Barker, M., Lamprecht, A.-L., Martinez, C., Psomopoulos, F. E., Harrow, J., Castro, L. J., Gruenpeter, M., Martinez, P. A., Honeyman, T., Struck, A., Lee, A., Loewe, A., van Werkhoven, B., Jones, C., Garijo, D., Plomp, E., Genova, F., … RDA FAIR4RS WG. (2022). FAIR Principles for Research Software (FAIR4RS Principles) (1.0). Zenodo. https://doi.org/10.15497/RDA00068

POLITECHNIKA GDAŃSKA

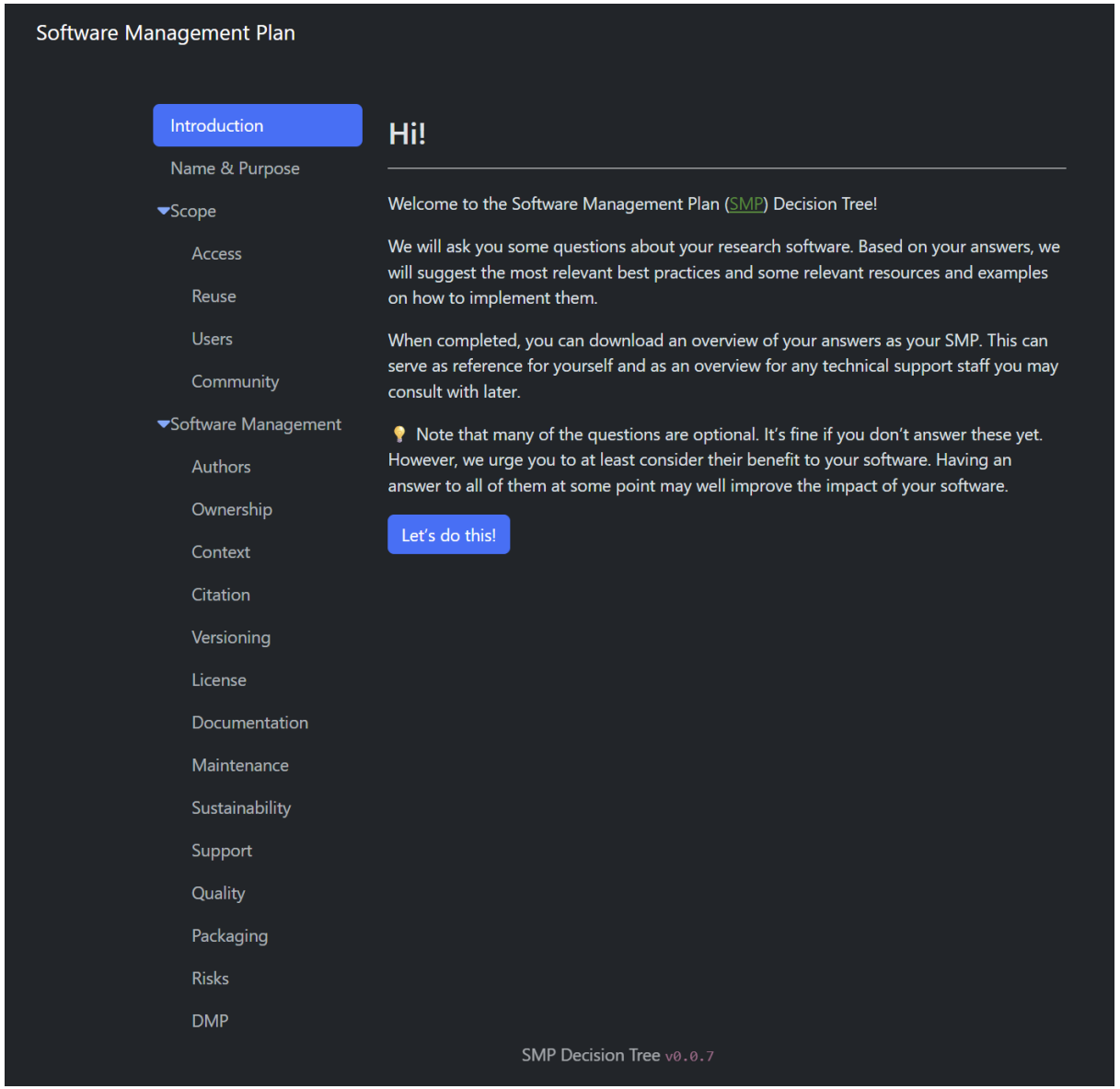
SMP - Narzędzia

SMP Decision Tree

url: https://smp.research.software/
Git: https://github.com/SS-NES/docassemble-SMPDecisionTree

Interaktywne narzędzie stworzone przez Netherlands eScience Center, które przeprowadza przez serię pytań wraz ze wskazówkami w celu opracowania spersonalizowanego Software Management Plan.

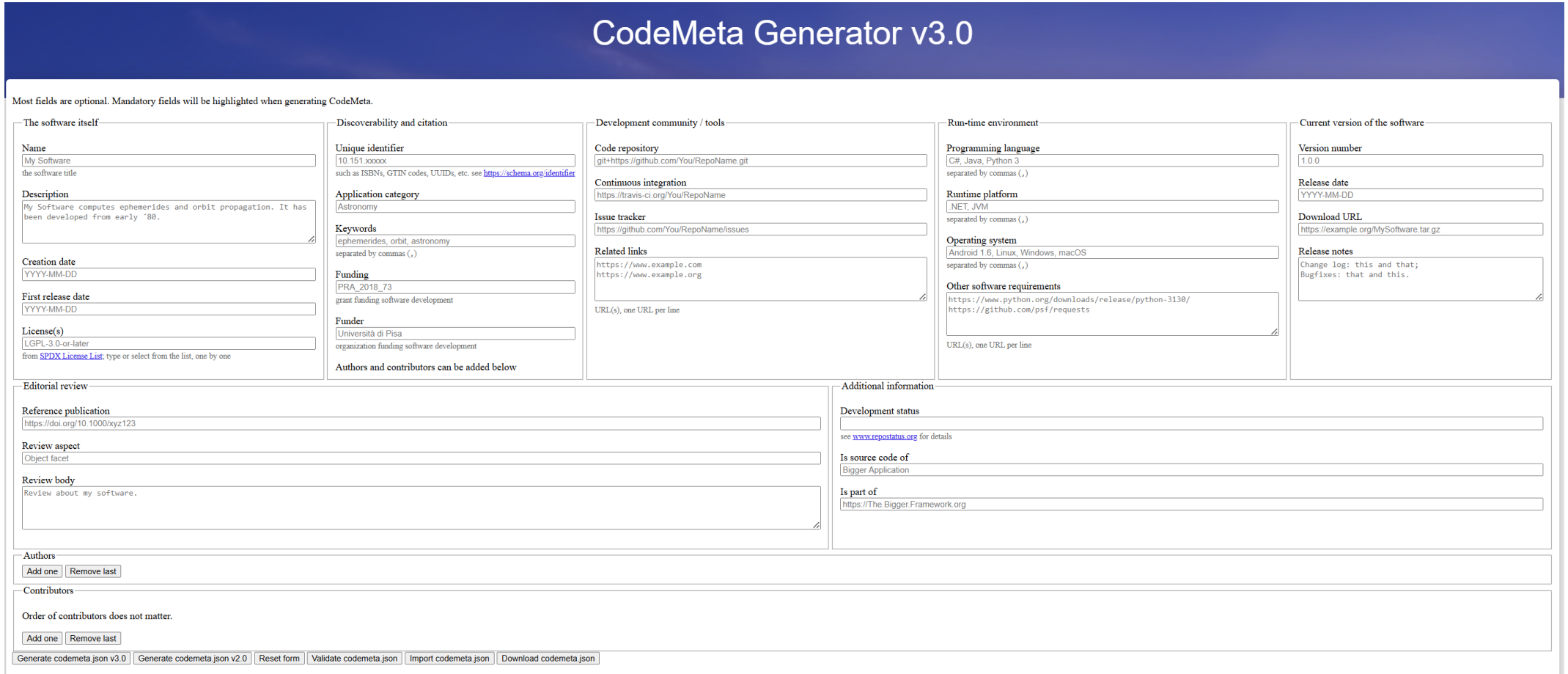
- Dokument zawiera rekomendacje dotyczące praktyk, narzędzi i zasobów
- Plan można dowolnie edytować, aktualizować oraz eksportować



CodeMeta Generator

url: https://codemeta.github.io/codemeta-generator

Narzędzie webowe do tworzenia pliku CodeMeta.json – standardowego formatu metadanych dla oprogramowania. Plik CodeMeta umożliwia opisanie oprogramowania w sposób zrozumiały dla maszyn i ludzi (JSON-LD).



CITATION.cff

url: https://citation-file-format.github.io/
url [generator] https://citation-file-format.github.io/cff-initializer-javascript

Tekstowy plik opisujący jak cytować oprogramowanie. Umieszczany w głównym katalogu repozytorium.
Wspierany przez:
✓ GitHub
✓ Zenodo
✓ Zotero
✓ Software Heritage

This is an example of a simple `CITATION.cff` file:

```
cff-version: 1.2.0
message: "If you use this software, please cite it as below."
authors:
  - family-names: Druskat
    given-names: Stephan
    orcid: https://orcid.org/1234-5678-9101-1121
title: "My Research Software"
version: 2.0.4
identifiers:
  - type: doi
    value: 10.5281/zenodo.1234
date-released: 2021-08-11
```

SMP - Narzędzia

GitHub

url: https://github.com/

Platforma kontroli wersji, która wspiera wiele elementów SMP:

- ✓ Repozytorium – centralny punkt dostępu do kodu
- ✓ Issues – system śledzenia problemów i wsparcia
- ✓ Releases – publikacja wersji
- ✓ README – dokumentacja dla użytkowników
- ✓ Contributing guide – instrukcje dla współtwórców
- ✓ License – deklaracja licencji
- ✓ Actions (CI/CD) – automatyczne testy

```
repo/
├── README.md              # Dokumentacja użytkownika
├── CONTRIBUTING.md        # Jak współtworzyć
├── LICENSE                # Licencja
├── CITATION.cff           # Jak cytować
├── codemeta.json          # Metadane
├── requirements.txt       # Zależności
├── .github/
│   └── workflows/
│       └── tests.yml      # CI/CD (testowanie)
└── docs/                  # Dokumentacja rozszerzona
```

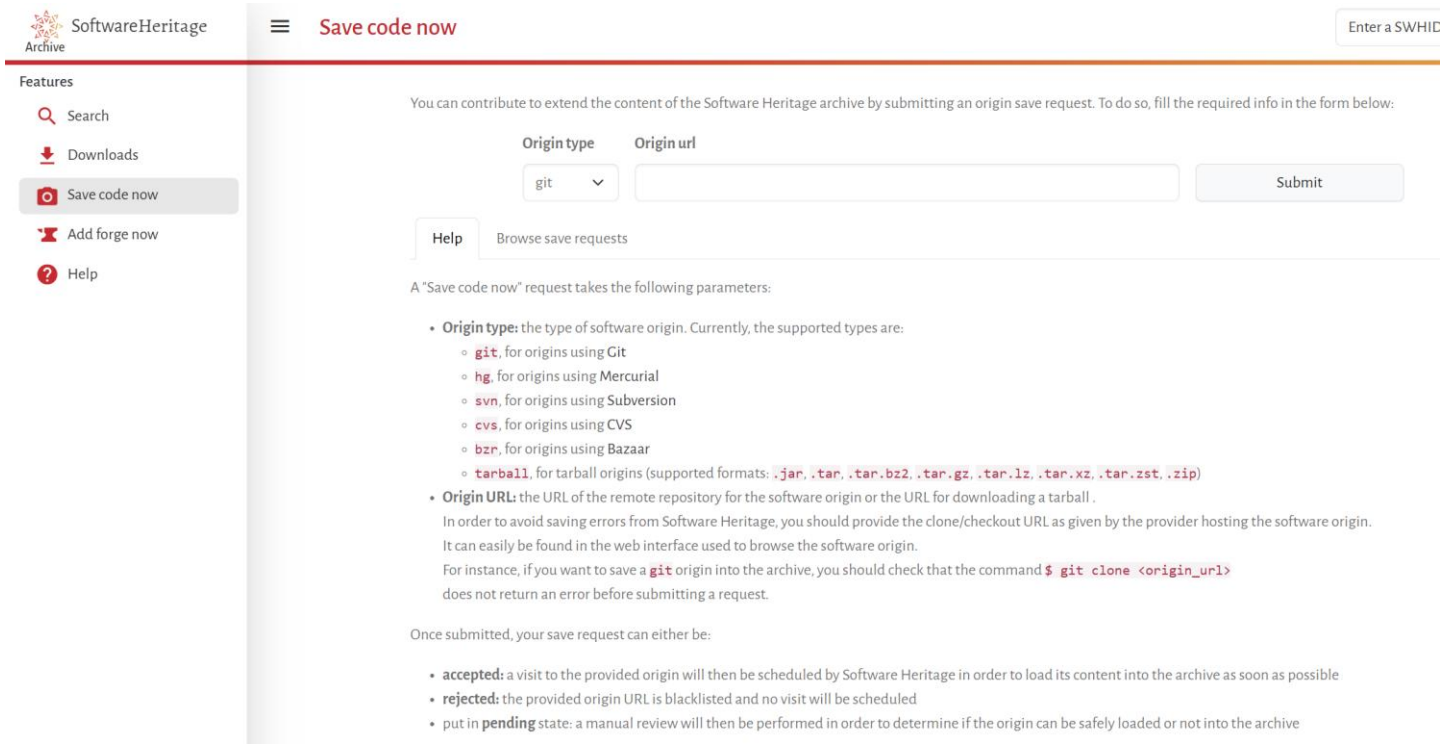
Software Heritage – Uniwersalne Archiwum Kodu

URL: https://www.softwareheritage.org/

Software Heritage to globalne, otwarte archiwum kodu źródłowego, którego celem jest długoterminowe gromadzenie, ochrona i udostępnianie oprogramowania jako części światowego dziedzictwa cyfrowego. Projekt zbiera publicznie dostępny kod z wielu platform (m.in. GitHub, GitLab, Bitbucket, ekosystemy pakietów), zapisując zarówno pliki źródłowe, jak i pełną historię wersji.

Kluczowe funkcje:

- ✓ Automatyczne archiwizowanie publicznych repozytoriów
- ✓ Gwarancja dostępu – nawet jeśli GitHub upadnie
- ✓ Trwały identyfikator (SWHID) dla każdego commit'a/dla każdej wersji
- ✓ Przeglądanie historii kodu bez dostępu do oryginalnego repozytorium



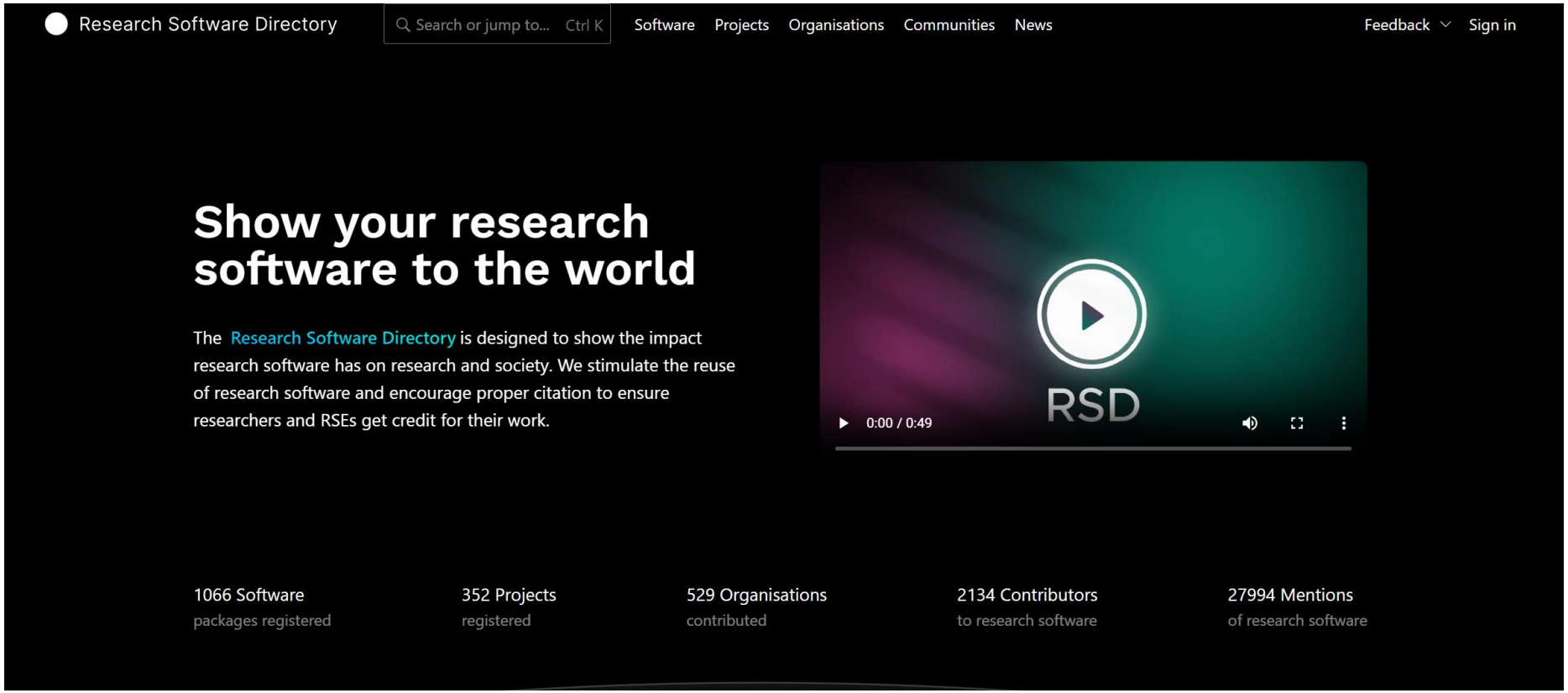
POLITECHNIKA GDAŃSKA

SMP - Narzędzia

Research Software Directory

URL: [Software | Research Software Directory/]

Repozytorium Oprogramowania Badawczego



POLITECHNIKA GDAŃSKA

Software Management Plan (SMP) - źródła wiedzy

- Martinez-Ortiz, C., Martinez Lavanchy, P., Sesink, L., Olivier, B. G., Meakin, J., de Jong, M., & Cruz, M. (2023). Practical guide to Software Management Plans (1.1). Zenodo. https://doi.org/10.5281/zenodo.7589725
- The Software Sustainability Institute. (2018). Checklist for a Software Management Plan (1.0). Zenodo. https://doi.org/10.5281/zenodo.2159713
- Chue Hong, N. P., Katz, D. S., Barker, M., Lamprecht, A.-L., Martinez, C., Psomopoulos, F. E., Harrow, J., Castro, L. J., Gruenpeter, M., Martinez, P. A., Honeyman, T., Struck, A., Lee, A., Loewe, A., van Werkhoven, B., Jones, C., Garijo, D., Plomp, E., Genova, F., … RDA FAIR4RS WG. (2022). FAIR Principles for Research Software (FAIR4RS Principles) (1.0). Zenodo. https://doi.org/10.15497/RDA00068
- Barker, M., Chue Hong, N.P., Katz, D.S. et al. Introducing the FAIR Principles for research software. Sci Data 9, 622 (2022). https://doi.org/10.1038/s41597-022-01710-x
- Grossmann, Y.V., Lanza, G., Biernacka, K., Hasler, T. and Helbig, K. (2024) Software Management Plans – Current Concepts, Tools, and Application. Data Science Journal, 23:43, 1-16. DOI: https://doi.org/10.5334/dsj-2024-043

- Software Sustainability Institute
- The Turing Way
- Centrum Kompetencji Otwartej Nauki Biblioteka PG

Szkolenie finansowane w ramach "Działań na rzecz promowania i zwiększenia świadomości w zakresie Otwartych Danych Badawczych", które stanowią element realizacji programu „Inicjatywa Doskonałości – Uczelnia Badawcza” (IDUB) w zakresie podniesienia poziomu jakości działalności naukowej uczelni w ramach Działania I.4 (Działania na rzecz zwiększenia liczby publikacji w prestiżowych czasopismach i wydawnictwach).