Du script artisanal à l'infrastructure ouverte : genèse, enjeux et fondements de *calendario_liturgico*

Andrés Felipe Echavarría Peláez[1]

[1] Consortium ARIANE- IR* Huma-Num

Abstract

The annotation of liturgical dates in early modern archival sources is a central methodological challenge for XML-TEI digital editions. The Python library `calendario_liturgico` automates the computation of Christian liturgical dates from the sixth century to the present by combining the Gauss–Meeus algorithm for the Gregorian calendar with pre-calculated paschal tables for the Julian period (532–1582). This article situates the genesis of this tool within the scholarly context of editing inquisitorial trial records, analyses its mathematical and computational foundations, and assesses its validation protocols. It finally discusses how the transition from an *ad hoc* script to a shared software infrastructure reshapes scientific reproducibility, the interoperability of historical data, and the relationship between humanities scholarship and computation.

Mots-clés: édition numérique, TEI, calendrier liturgique, *computus*, science ouverte, reproductibilité, humanités numériques, algorithme de Gauss, Python
Keywords: digital edition, TEI, liturgical calendar, computus, open science, reproducibility, digital humanities, Gauss algorithm, Python

1 Introduction

L'annotation temporelle des documents d'Ancien Régime pose des difficultés spécifiques aux projets d'édition numérique structurée. La norme ISO 8601 fournit un cadre de normalisation pour les dates grégoriennes modernes, mais elle se révèle insuffisante face aux complexités chronologiques des sources historiques ; les extensions ISO 8601-2 :2019 autorisent la représentation de dates incertaines ou approximatives sans offrir de mécanisme de conversion entre systèmes calendaires [14]. Les documents procéduraux de l'Inquisition espagnole, rédigés entre les XV^e^ et XIX^e^ siècles, recourent fréquemment à des repères liturgiques qui ne peuvent être normalisés sans calcul préalable de la date de la Pâques chrétienne et des fêtes mobiles qui en dépendent [9].

La librairie Python `calendario_liturgico`, publiée sur PyPI, apporte une réponse computationnelle à ce problème [8]. Elle implémente l'algorithme de Gauss–Meeus pour le calcul de Pâques dans le calendrier grégorien et intègre 1 052 dates pascales pré-calculées pour la période julienne (532–1582). Issu d'un script développé dans le cadre d'une thèse de doctorat sur l'éditorialisation des procès de foi espagnols [9], cet outil s'insère dans les chaînes de traitement XML-TEI pour produire des dates conformes aux attributs `@when`, `@notBefore` et `@notAfter` du standard TEI P5 [4 ; 21].

Cet article retrace en premier lieu la genèse de la librairie en la situant dans le champ de l'éditorialisation numérique de corpus inquisitoriaux ; il analyse ensuite les fondements mathématiques et computationnels du *computus* pascal tels qu'ils sont implémentés dans `calendario_liturgico` ; enfin, il discute quelques points pour cette infrastructure mutualisée, dans la perspective de la science ouverte et des principes FAIR appliqués au logiciel de recherche [5].

Andrés Felipe Echavarría Peláez. "Du script artisanal à l'infrastructure ouverte : genèse, enjeux et fondements de *calendario_liturgico*." *Actes de la Conférence Humanistica*, éd. par Serena Crespi, Simon Gabay, Martin Grandjean, Ariane Pinche, Marie Puren et Léa Saint-Raymond. Vol. 4. Anthology of Computers et the Humanities. 2026, 85–91. https://doi.org/10.63744/Sik2vxas691B.

2 Contexte disciplinaire : éditorialisation des procès inquisitoriaux

2.1 Le corpus et ses spécificités temporelles

Ce projet prend sa source dans l'annotation XML-TEI d'un prototype de procès pour cause de foi instruit par l'Inquisition espagnole. Produits par les tribunaux de district, ces procès recourent fréquemment aux fêtes du calendrier liturgique pour dater les faits dans les témoignages, avec des mentions telles que *después de la Navidad próxima pasada veinte días* (« vingt jours après le prochain Noël passé ») ou *el miércoles de las octavas pasadas* (« le mercredi des octaves passées »). À notre connaissance, aucune librairie Python disponible sur PyPI ne combine un *computus* julien fondé sur des tables historiques validées, le calcul systématique des fêtes mobiles dérivées et la production de sorties compatibles dans des attributs TEI P5. [1]

Cette pratique de datation reflète la place centrale des dates religieuses dans l'organisation du temps civil et ecclésiastique de l'Espagne moderne. L'annotation de ce type de source selon les recommandations du TEI P5 impose la normalisation des mentions temporelles à l'aide d'attributs structurés conformes à la norme ISO 8601 [4]. L'élément `<date>` permet notamment l'utilisation de `@when` pour une date précise, de `@notBefore` et `@notAfter` pour un intervalle temporel, et de `@from`/`@to` pour une période [21].

2.2 La problématique de la conversion temporelle

La conversion de références liturgiques mobiles en dates absolues suppose le calcul préalable de la date de Pâques pour l'année concernée. Pâques constitue l'événement fondamental du calendrier chrétien : plus d'une vingtaine de fêtes mobiles en dépendent, parmi lesquelles le Mercredi des Cendres (46 jours avant Pâques), les Rameaux (7 jours avant), l'Ascension (39 jours après), la Pentecôte (49 jours après) et le *Corpus Christi* (la Fête-Dieu, 60 jours après). Depuis le concile de Nicée (325), la règle fixe Pâques au dimanche suivant la première pleine lune ecclésiastique après l'équinoxe de printemps, conventionnellement fixé au 21 mars [18].

Les difficultés s'accentuent pour les documents antérieurs à octobre 1582, date de l'adoption du calendrier grégorien en Espagne. La transition entre calendriers julien et grégorien crée une discontinuité calendaire qui affecte directement le *computus* pascal. Faute d'outils unifiés, les encodeurs doivent consulter manuellement des tables pascales dispersées ou recourir à des calculatrices dont la fiabilité et la pérennité ne sont pas garanties.

3 Fondements mathématiques : le *computus* pascal

3.1 Historique du calcul de Pâques

Le terme *computus* désigne l'ensemble des méthodes de calcul de la date de Pâques, science qui a mobilisé les savants ecclésiastiques pendant plus de quinze siècles [2]. Denys le Petit (*Dionysius Exiguus*) a composé en 525 une table pascale de 95 ans couvrant la période 532–626, qui introduit le système de datation *Anno Domini* et prolonge une tradition alexandrine antérieure [18]. Le cycle de 532 ans, produit du cycle métonique de 19 ans et du cycle solaire de 28 ans, forme la base du système de calcul pascal adopté par l'Église latine. Bède le Vénérable diffuse ce système dans son traité *De temporum ratione* (725), qui fournit une justification du calcul pascal et intègre le système *Anno Domini* dans l'historiographie occidentale [1].

1. Sur PyPI existent déjà des bibliothèques de calendriers liturgiques ou de jours fériés, telles que `workalendar` (calendriers civils incluant certains jours fériés religieux), `liturgical-calendar` et `liturgical-colour` (calendrier et couleurs liturgiques pour l'Église d'Angleterre), ou encore des liaisons Python vers la bibliothèque *romcal* qui génère des calendriers liturgiques catholiques à partir d'un cœur écrit en Rust [3 ; 12 ; 16]. Aucune ne propose toutefois, à ce jour, une table pascale julienne 532–1582 validée sur sources historiques ni une production directe de dates normalisées pour l'annotation TEI.

3.2 Le cycle métonique et l'arithmétique modulaire

Le cycle de Méton, découvert par l'astronome grec Méton vers 432 av. J.-C., établit que 19 années solaires correspondent approximativement à 235 mois synodiques, avec un écart d'environ 2 heures. Le rang d'une année dans ce cycle est appelé nombre d'or (*Golden Number*), calculé par la formule :

$$G = (A \bmod 19) + 1 \tag{1}$$

où A désigne l'année considérée [6]. L'épacte, c'est-à-dire l'âge de la Lune au 1er janvier, dérive du nombre d'or. Pour le calendrier julien, l'épacte E_j se calcule à partir du nombre d'or n par la relation :

$$E_j = (11\,n - 10) \bmod 30 \tag{2}$$

Pour le calendrier grégorien, des corrections séculaires interviennent pour compenser l'écart entre le cycle métonique et la durée réelle des mois synodiques, qui produit un décalage d'un jour tous les 312,5 ans environ [2].

3.3 L'algorithme de Gauss (1800)

Carl Friedrich Gauss publie en 1800, dans la revue *Monatliche Correspondenz*, un algorithme pour le calcul de la date de Pâques qui remplace les tables traditionnelles d'épactes et de lettres dominicales [11]. L'algorithme procède en deux étapes. La première calcule le nombre de jours d séparant le 21 mars de la pleine lune pascale. Soit Y l'année considérée :

$$a = Y \bmod 19 \tag{3}$$
$$k = \lfloor Y/100 \rfloor \tag{4}$$
$$p = \lfloor (13 + 8k)/25 \rfloor \tag{5}$$
$$q = \lfloor k/4 \rfloor \tag{6}$$
$$M = (15 - p + k - q) \bmod 30 \tag{7}$$
$$d = (19a + M) \bmod 30 \tag{8}$$

La variable a positionne l'année dans le cycle métonique. L'indice séculaire k et les corrections p et q ajustent le calcul pour les siècles grégoriens : p intègre le décalage du cycle métonique, tandis que q corrige pour les jours bissextiles non observés [2]. La constante M fournit le point de départ pour chaque siècle dans l'arithmétique modulo 30, qui correspond à la durée approximative du mois synodique. Le facteur 19 dans l'équation (8) reflète le décalage annuel de 11 jours entre année solaire et année lunaire : comme $19 \times 11 = 209 \equiv -1 \pmod{30}$, ce coefficient est fonctionnellement équivalent à -11 en arithmétique modulo 30 [2].

La seconde étape calcule le nombre de jours e séparant la pleine lune pascale du dimanche suivant :

$$N = (4 + k - q) \bmod 7 \tag{9}$$
$$b = Y \bmod 4 \tag{10}$$
$$c = Y \bmod 7 \tag{11}$$
$$e = (2b + 4c + 6d + N) \bmod 7 \tag{12}$$

L'expression $2b + 4c$ encode le décalage du jour de la semaine d'une année à l'autre, d'un jour pour les années ordinaires et de deux jours pour les années bissextiles [2]. Deux corrections

historiques s'appliquent : si $d = 29$ et $e = 6$, ou si $d = 28$, $e = 6$ et $a > 10$, alors e est remplacé par -1, ce qui déplace Pâques d'une semaine en arrière. La date finale de Pâques est

$$P = 22 + d + e \tag{13}$$

Si $P > 31$, Pâques tombe en avril au jour $(d + e - 9)$; sinon, Pâques tombe en mars au jour P. Reinhold Bien a documenté l'évolution de cet algorithme et ses corrections successives, de la version initiale de 1800 à la version corrigée par Peter Paul Tittel en 1816 [2]. Dans le cadre du calendrier grégorien standard, ces variantes corrigées concordent avec les algorithmes de Butcher–Meeus et avec les fonctions de calcul de Pâques proposées par Dershowitz et Reingold pour la période postérieure à 1583 [6].

3.4 L'algorithme de Meeus et la synthèse computationnelle

L'algorithme publié par Jean Meeus dans *Astronomical Algorithms* (1991) synthétise des formules antérieures : il réunit l'algorithme de Delambre (1814) pour le calendrier julien et l'algorithme de Butcher (1877) pour le calendrier grégorien [17]. L'algorithme de Butcher, diffusé à l'origine par une note dans *Nature* et démontré sans limite de date par Samuel Butcher, a ensuite servi de base à de nombreuses implémentations, notamment dans les langages de programmation étudiés par Donald Knuth [2 ; 15]. Meeus fournit des variantes explicites pour les calendriers julien et grégorien dont la complexité temporelle est $O(1)$: elles ne comportent qu'un nombre constant d'opérations arithmétiques, sans boucle ni récursion.

3.5 L'approche hybride de `calendario_liturgico`

La librairie `calendario_liturgico` adopte une stratégie hybride [8]. Pour le calendrier grégorien (à partir de 1582), elle implémente l'algorithme de Gauss–Meeus, qui calcule la date de Pâques en temps constant. Pour le calendrier julien (532–1582), la librairie stocke 1 052 dates pascales pré-calculées, validées contre les tables historiques de Denys le Petit, de Bède le Vénérable et des calendriers officiels du Vatican. La table julienne (532–1582) a été construite par transcription des tables de Denys le Petit et de Bède, puis vérifiée automatiquement par comparaison avec les algorithmes de conversion de Dershowitz et Reingold pour la même période [6].

Ce choix repose sur une exigence de fiabilité historique. Les tables pascales juliennes sont le fruit de calculs ecclésiastiques séculaires qui constituent une autorité documentaire ; substituer un algorithme moderne à ces tables manuscrites comporterait un risque d'anachronisme méthodologique : les sources primaires médiévales (tables de Bède, de Grosseteste) représentent l'état de la pratique computistique telle qu'elle était effectivement appliquée [1]. En revanche, l'algorithme de Gauss–Meeus pour le calendrier grégorien relève d'un modèle mathématique explicite et sa concordance avec les algorithmes de Dershowitz et Reingold a été établie pour l'ensemble de la période grégorienne [6]. L'ouvrage de référence *Calendrical Calculations* de Dershowitz et Reingold présente en outre des implémentations algorithmiques complètes pour la conversion entre une trentaine de systèmes calendaires, y compris les calendriers ecclésiastiques [6]. La librairie `calendario_liturgico` se distingue de cette approche généraliste par sa spécialisation dans le contexte liturgique catholique et son compatibilité pour la TEI.

4 Architecture logicielle et choix de conception

4.1 Structure modulaire

L'architecture de `calendario_liturgico` repose sur une séparation fonctionnelle en trois modules [7]. Le module `core` expose la classe principale `LiturgicalCalendar`, point d'entrée pour les utilisateurs. Le module `calculators` implémente l'algorithme de Gauss–Meeus, tandis que le

module `data` stocke les dates pascales pré-calculées pour la période julienne. À partir de la date de Pâques, la librairie calcule l'ensemble des principales fêtes liturgiques mobiles (Mercredi des Cendres, Dimanche des Rameaux, Triduum pascal, Ascension, Pentecôte, *Corpus Christi*, Christ Roi), ainsi que les dimanches de l'Avent et Noël [8]. Chaque fête est dérivée par un décalage arithmétique constant par rapport à Pâques, ce qui garantit la cohérence interne des résultats.

La librairie utilise exclusivement la bibliothèque standard Python, sans recourir à tout autre paquet tiers [8]. Ce choix de conception vise à limiter les dépendances transitives, qui peuvent introduire des ruptures en cascade lors des mises à jour, et à assurer une stabilité à long terme pour un outil destiné à s'intégrer dans des chaînes de traitement académiques pérennes. Il renforce également la portabilité : un code sans dépendances externes fonctionne dans tout environnement Python standard, facilitant le déploiement sur des serveurs de recherche, des postes de travail de chercheurs en sciences humaines ou des environnements interactifs.

La librairie intègre des annotations de types complètes (*type hints*) conformes aux *PEP 484* et suivantes, ce qui permet la vérification statique du code par des outils comme `mypy` et clarifie les interfaces pour les utilisateurs avancés [7]. Une suite de tests automatisés, implémentée avec `pytest`, couvre des cas critiques : années bissextiles, transitions séculaires, limites du calendrier julien (532) et du calendrier grégorien (1582), cas historiques documentés [20]. Ces tests sont exécutés en intégration continue afin de prévenir les régressions[2].

5 Validation, documentation et science ouverte

La librairie `calendario_liturgico` a été confrontée à trois corpus de référence [8] : les dates pascales officielles publiées par le Vatican pour la période 1583-2100, les tables historiques médiévales (Bède, Grosseteste) et les calendriers liturgiques contemporains de plusieurs diocèses. Par ailleurs, une validation croisée avec les algorithmes décrits par Reingold et Dershowitz fournit un contrôle supplémentaire.

L'ensemble du code source est documenté en espagnol, français et anglais [7]. Ce choix linguistique vise les communautés scientifiques concernées : hispanistes travaillant sur les sources inquisitoriales, chercheurs francophones des humanités numériques et communauté internationale du TEI Consortium.

La librairie est publiée sous Licence Ouverte Etalab 2.0, conçue par la mission Etalab pour faciliter la réutilisation libre et gratuite des informations publiques [10][3].

La mise en œuvre des principes FAIR4RS [5 ; 19] dans `calendario_liturgico` se traduit par plusieurs choix : la *trouvabilité* est assurée par la publication sur PyPI et sur Zenodo avec un DOI ; l'*accessibilité* par le code source ouvert hébergé sur GitHub [7] ; l'*interopérabilité* par la production de dates au format ISO 8601 et la compatibilité avec les attributs TEI ; la *réutilisabilité* par la licence ouverte, la documentation trilingue et l'architecture modulaire.

Cependant, l'algorithme de Gauss–Meeus offre une précision adaptée au calendrier grégorien standard, mais ne prend pas en compte toutes les variations régionales historiques. L'adoption du calendrier grégorien a été progressive : l'Espagne l'a adopté en octobre 1582, alors que d'autres pays européens ne l'ont fait que beaucoup plus tard. Les particularismes liturgiques locaux (rites mozarabe, ambrosien) ne sont pas modélisés, pas plus que les calendriers propres à certaines communautés religieuses. Les références temporelles ambiguës (« esta Cuaresma o la anterior ») nécessitent en outre un jugement interprétatif humain que l'outil ne peut et ne doit pas automatiser.

2. En complément, une interface web développée avec le micro-framework Flask offre aux historiens et aux liturgistes non programmeurs un moyen de vérifier les calculs sans écrire de code Python [8 ; 13].

3. Cette licence autorise la reproduction, l'adaptation, la modification et l'exploitation commerciale des données et du code, sous réserve de la mention de la source, et est compatible avec la licence Creative Commons Attribution (CC-BY) et l'Open Data Commons Attribution License (ODC-BY).

6 Perspectives et extensions

Les perspectives d'évolution concernent moins un programme de travail arrêté qu'un potentiel ouvert pour d'autres communautés. Le *computus* n'est pas un phénomène exclusivement latin : les Églises orthodoxes calculent Pâques selon le calendrier julien, ce qui produit des dates différentes de celles du calendrier grégorien, et les calendriers hébraïque et islamique, luni-solaires ou lunaires, posent des problèmes de conversion comparables [6]. L'ouvrage de Dershowitz et Reingold propose des algorithmes pour une trentaine de systèmes calendaires [6] ; l'architecture modulaire de `calendario_liturgico` pourrait, en principe, servir de base à des extensions ou à des projets connexes centrés sur d'autres traditions liturgiques.

De même, l'exposition d'une API stable ouvre la voie à des intégrations ultérieures dans des plateformes d'édition numérique ou des services web, sans que ces développements soient nécessairement portés par l'auteur initial de la librairie. L'enrichissement des métadonnées historiques associées à chaque date (sources liturgiques médiévales, variantes régionales) constituerait une piste de travail précieuse pour documenter les limites de chaque calcul et expliciter les choix éditoriaux.

Enfin, l'articulation entre calcul de dates et traitement automatique du langage naturel (TAL) reste un champ à explorer. La reconnaissance automatique des mentions temporelles liturgiques dans les textes espagnols du XVI^{e} siècle, combinée au calcul des dates correspondantes, permettrait, à terme, une annotation semi-automatique de corpus à grande échelle.

7 Conclusion

La genèse de `calendario_liturgico` témoigne d'une dynamique de mutualisation des infrastructures de recherche en humanités numériques. Partie d'un besoin méthodologique situé — l'annotation TEI d'un corpus inquisitorial espagnol —, elle aboutit à une ressource computationnelle générique, documentée, validée et accessible. Les fondements mathématiques de l'outil, ancrés dans une tradition computistique pluriséculaire (de Denys le Petit à Gauss et Meeus), garantissent la fiabilité des calculs, tandis que l'architecture sans dépendances, les tests automatisés et la documentation trilingue en renforcent la pérennité et la réutilisabilité. La conformité aux principes FAIR4RS et la publication sous licence ouverte inscrivent la librairie dans l'écosystème de la science ouverte [5].

Cette trajectoire illustre une éthique de la recherche où la science ouverte ne se limite pas au partage des publications, mais s'étend à la mise en commun des outils, des protocoles et des données. Le dialogue entre humanités et computation suppose à la fois une acculturation des chercheurs en sciences humaines aux logiques algorithmiques et une attention accrue, du côté des informaticiens, aux subtilités herméneutiques et à l'historicité des objets culturels. Sans prétendre épuiser ces enjeux, `calendario_liturgico` propose un exemple de rigueur computationnelle mise au service de l'intelligence historique, en maintenant une distinction claire entre ce qui relève du calcul et ce qui relève de l'interprétation.

Références

[1] Bède le Vénérable. *De temporum ratione*. Éd. et trad. anglaise par Faith Wallis, *The Reckoning of Time*, Liverpool University Press, 1999. 725.

[2] Bien, Reinhold. « Gauß and Beyond : The Making of Easter Algorithms ». In : *Archive for History of Exact Sciences 58*, no. 5 (2004), p. 439-452. DOI : `10.1007/s00407-004-0078-5`.

[3] Bord, Bruno. « Workalendar : Worldwide Holidays and Working Days Helper and Toolkit. » URL : `https://github.com/workalendar/workalendar` (visité le 27/03/2026).

[4] BURNARD, Lou. *What is the Text Encoding Initiative ?* Encyclopédie numérique. Marseille : OpenEdition Press, 2014. DOI : 10.4000/books.oep.426.

[5] CHUE HONG, Neil P. et al. « Introducing the FAIR Principles for Research Software ». In : *Scientific Data 9* (2022), p. 622. DOI : 10.1038/s41597-022-01710-x.

[6] DERSHOWITZ, Nachum et REINGOLD, Edward M. *Calendrical Calculations : The Ultimate Edition*. 4e éd. Cambridge : Cambridge University Press, 2018.

[7] ECHAVARRÍA PELÁEZ, Andrés Felipe. « calendario-liturgico — GitHub repository ». 2025. URL : https://github.com/andresecha/calendario-liturgico.

[8] ECHAVARRÍA PELÁEZ, Andrés Felipe. « calendario_liturgico : a Python library for calculating Christian liturgical dates ». Version 0.1.0. 2025. URL : https://pypi.org/project/calendario-liturgico.

[9] ECHAVARRÍA PELÁEZ, Andrés Felipe. *Éditorialisation des procès de foi espagnols : annotation textuelle et thésaurus documentaire*. Thèse de doct. Université Paul-Valéry Montpellier 3, 2025.

[10] ETALAB. « Licence Ouverte / Open Licence, version 2.0 ». 2017. URL : https://www.etalab.gouv.fr/wp-content/uploads/2017/04/ETALAB-Licence-Ouverte-v2.0.pdf.

[11] GAUSS, Carl Friedrich. « Berechnung des Osterfestes ». In : *Monatliche Correspondenz zur Beförderung der Erd- und Himmelskunde 2* (1800). Réédité dans *Werke*, vol. 6, p. 73–79, p. 121-130.

[12] GAZELEY, Jonathan. « liturgical-calendar : Library to determine liturgical dates and colours for the Anglican Church of England ». 2024. URL : https://pypi.org/project/liturgical-calendar.

[13] GRINBERG, Miguel. *Flask Web Development : Developing Web Applications with python*. O'Reilly, 2018.

[14] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. *ISO 8601-2 :2019 — Date and Time — Representations for Information Interchange — Part 2 : Extensions*. Genève : ISO, 2019.

[15] KNUTH, Donald E. « The Calculation of Easter ». In : *Communications of the ACM 5*, no. 4 (1962), p. 209-210. DOI : 10.1145/366920.366980.

[16] MAGNIER, Étienne. « romcal 4.0.0b6 : Liturgical Calendar Library for Calculating Catholic Liturgical Dates and Calendars ». URL : https://github.com/emagnier/romcal (visité le 27/03/2026).

[17] MEEUS, Jean. *Astronomical Algorithms*. Richmond, VA : Willmann-Bell, 1991.

[18] MOSSHAMMER, Alden A. *The Easter Computus and the Origins of the Christian Era*. Oxford Early Christian Studies. Oxford : Oxford University Press, 2008.

[19] MUNAFÒ, Marcus R., NOSEK, Brian A., BISHOP, Dorothy V. M., BUTTON, Katherine S., CHAMBERS, Christopher D., PERCIE DU SERT, Nathalie, SIMONSOHN, Uri, WAGENMAKERS, Eric-Jan, WARE, Jennifer J. et IOANNIDIS, John P. A. « A Manifesto for Reproducible Science ». In : *Nature Human Behaviour 1*, no. 1 (2017), p. 0021. DOI : 10.1038/s41562-016-0021.

[20] « Python Package Index - PyPI ». URL : https://pypi.org/ (visité le 28/03/2021).

[21] TEI CONSORTIUM. « TEI P5 : Guidelines for Electronic Text Encoding and Interchange ». Version 4.9.0. 2025. URL : https://tei-c.org/release/doc/tei-p5-doc/en/html/index.html.