Rapport d'Activité

Didier Verna
Epita/ Lre
14–16 rue Voltaire,
94276 Le Kremlin-Bicêtre

didier.verna@epita.fr
https://lrde.epita.fr/~didier

0000-0002-6315-052X profile/Didier_Verna in/didierverna

Distinctions

Reviewers Choice Award 2018 du Art, Science and Engineering of Programming Journal
(Verna, 2018a, n° 3[publication choisie])

Keynote Speaker à ACCU 2014
(Verna, 2014, n° 51)

Best Paper Award 2006 du European Lisp Workshop
(Verna, 2006a, n° 43)

Actuellement

Position :	Enseignant-Chercheur habilité à diriger les recherches
Établissement d'enseignement :	École pour l'Informatique et les Techniques Avancées (Epita)
Établissement de Recherche :	Laboratoire de Recherche de l'Epita (Lre)

Parcours

2024 **Qualification Professeur des Universités**

Section : 27 — Informatique
Jury : Olga Kouchnarenko, Céline Rouveirol

2020 **Habilitation à Diriger les Recherches, Edite de Paris, Sorbonne Université, Epita/Lre**

Titre : `(dynamic (programming paradigms)) ;; performance and expressivity`
Jury : Manuel Serrano, Inria, Sophia Antipolis, France (rapporteur)
Robert Strandh, Université de Bordeaux, France (rapporteur)
Nicolas Neuß, Fau, Erlangen-Nürnberg, Allemagne (rapporteur)
Marco Antoniotti, Université de Milan-Bicocca, Italie (examinateur)
Ralf Möller, Université de Lübeck, Allemagne (examinateur)
Gérard Assayag, Ircam, Paris, France (examinateur).

2000 **Docteur de l'École Nationale Supérieure des Télécommunications de Paris (Enst)**

Titre : Téléopération et Réalité Virtuelle — Assistance à l'Opérateur par Modélisation Cognitive de ses Intentions
Jury : Philippe Coiffet, Jean-François Richard (rapporteurs)
Jean-Paul Papin (examinateur)
Alain Grumbach (directeur)

1994 Ingénieur de l'Enst, spécialisation Informatique et Réseaux
1988 Baccalauréat scientifique (C)

Expérience Professionnelle

2014–… Membre de l'Edite de Paris (École Doctorale Informatique, Télécommunications et Électronique)
2005–2018 Enseignant vacataire à l'Enst, l'Ensta, l'Upmc, l'École Nationale des Beaux Arts et Ionis stm
Cf. détails dans la partie « Activités d'Enseignement »
1999–… Enseignant-Chercheur à l'Epita (CDI)

À Noter

Je travaille à temps partiel (3/5e, puis récemment 4/5e). Toutes les données chiffrées qui suivent sont exprimées en valeur absolue. Les heures d'enseignement sont accompagnées d'un calcul d'équivalent temps plein, mais le reste (densité de publication, taux de participation à l'écosystème, *etc.*) est à évaluer relativement à mon taux d'activité.

1 Responsabilités Administratives

2020–… Référent national cours d'informatique fondamentale, *cf.* section « Enseignement »
2009–… Particiaption au processus de recrutement des étudiants de l'Epita (salons, JPO, oraux d'admission, *etc.*)
Environ 90h / an
2008–… Co-fondateur et président du comité de pilotage d'ELS (European Lisp Symposium)
2008 Fondateur et premier organisateur du séminaire Performance et Généricité du Lre

2 Activités d'Enseignement

Je suis le concepteur de tous les cours que je donne depuis 1999. Tous mes supports de cours sont disponibles publiquement sur la page enseignement[1] de mon site web scientifique, y compris les anciens cours archivés.

2.1 Actuellement à l'Epita

En Bref

- Référent national cours
- 600+ étudiants/an niveau L3
- 116h CM + 42h TP = 158h/an

⇒ Équivalent temps plein = 200h/an

Anciennement vacataire à l'Enst, l'Ensta, l'Upmc, et l'École des Beaux Arts.

Les cours ci-dessous concernent plus de 600 étudiants en tronc commun du cycle ingénieur (niveau L3) chaque année. J'en suis le « référent national » (*cf.* attestation jointe) : je conçois et donne ces cours moi-même dans l'antenne parisienne de l'école (environ 400 étudiants / an en 3 groupes) et je gère par ailleurs des équipes de 4 à 6 enseignants qui les dispensent, avec mes supports, dans les quatre antennes régionales : Lyon, Rennes, Strasbourg, et Toulouse.

À cela se rajoutent de nombreux encadrements d'étudiants sur des projets divers et variés, du coaching, du suivi d'étudiants en filière apprentissage, et d'étudiants chercheurs (*cf.* Section 5.3 page 10). Ces activités étant très variables d'une année sur l'autre, elles ne sont pas détaillées ici, bien que représentant quelques dizaines d'heures de face à face pédagogique supplémentaires chaque année.

Intitulé		Cours Magistraux	TPs	Depuis
Igpp	Introduction Générale aux Paradigmes de Programmation	2h		2016
Aop 1	Approche Objet Classique de la Programmation	12h	3 (2h) = 6h	2016
Afp	Approches Fonctionnelles de la Programmation	12h	4 (2h) = 8h	2008
Aop 2	Approche Objet Avancée de la Programmation	12h		2017
Typo	Typographie : Art, Artisanat, et Science	12h		2023

Le volume horaire est donné en heures par étudiant

2.1.1 Syllabus

Igpp – Introduction Générale aux Paradigmes de Programmation Il s'agit d'une introduction générale à la notion de paradigme de programmation, qui sert de préambule aux trois cours suivants : Aop 1, Aop 2, et Afp. On y évoque la naissance de l'informatique, l'émergence de la notion de paradigme dans le contexte historique de la programmation impérative et procédurale et de l'architecture de Von Neumann, ainsi que les aspects linguistiques et langagiers de cette notion (expressivité, lien entre langage et pensée, *etc.*).

1. https://www.lrde.epita.fr/~didier/lectures/

Aop 1 – Approche Objet Classique de la Programmation Ce cours traite de l'approche orientée objet classique (industrielle). C'est en particulier l'approche des langages Java et C++ qui sont tous deux utilisés en guise d'illustration. Les caractéristiques de cette approche sont la notion de « classe » pour la structuration statique de l'information, l'« envoi de message » pour la modélisation du comportement dynamique, le tout dans le context général des langages statiquement typés.
On détaille le contexte historique dans lequel ce paradigme est né, son évolution et son développement, ainsi que les concepts fondamentaux au-dessus desquels il s'articule. On aborde les grands principes de modélisation dans cette approche, et on en montre également les limites, en mettant l'accent sur les liens étroits avec les caractéristiques des langages sous-jacents.

- **00. Introduction** Rappels sur les paradigmes impératifs et procéduraux. Genèse, histoire, évolution et développement de l'approche orientée objet. Situation actuelle.
- **01. Classes et Objets** Modélisation à base de classe. Notion d'objet, instantiation, destruction (*vs.* langages à ramasse-miettes). Diagrammes uml. Portée (attributs et méthodes d'instance ou de classe) et accessibilité (attributs et méthodes privées ou publiques) de l'information.
- **02. Structuration Statique** Relations entre classes : agrégation, composition, héritage. Caractéristiques de l'héritage : sous-classage *vs.* sous-typage, hiérarchies de classes (héritage simple et multiple), classes abstraites et finales, accessibilité de l'information (attributs et méthodes protégés). Problèmes liés à l'héritage : héritage *vs.* instantiation, héritage *vs.* agrégation, ambivalence, héritage en losange, héritage publique et privé.
- **03. Comportement Dynamique** Polymorphisme statique : surcharge, masquage. Polymorphisme dynamique : réécriture (méthodes virtuelles). Méthodes abstraites. Interfaces *vs.* héritage multiple. Limitations de la pseudo-équivalence sous-classe ⇔ sous-type : covariance, contravariance, principe de substitution de Liskov.

Afp – Approches Fonctionnelles de la Programmation Le paradigme de programmation fonctionnelle est très ancien, solidement ancré dans des fondations théoriques, et pourtant beaucoup moins populaire que le paradigme objet, bien qu'il ait éprouvé un regain d'intérêt ces deux dernières décennies. Ce cours traite du paradigme de programmation fonctionnelle au sens large, c'est-à-dire non réduit au seul fonctionnel pur. En guise d'illustration, nous utilisons deux langages en parallèle : Lisp (le père du paradigme), et Haskell (un langage plus récent). Ces deux langages sont fonctionnels et pourtant complètement différents sur pratiquement tous leurs aspects. C'est précisément ce qui rend leur juxtaposition intéressante.
On détaille le contexte historique dans lequel ce paradigme est né, les principes applicatifs qu'il offre (principalement les fonctions d'ordre supérieur) ainsi que leurs fondements théoriques. On met l'accent sur les différents contextes langagiers d'intégration de ce paradigme, et en particulier avec typage statique ou dynamique, avec ou sans effets de bords (pur / impur), et dans le cadre d'évaluateurs à l'ordre normal ou applicatif.

- **00. Introduction** Histoire, évolution et développement du paradigme : Lisp et le Lambda calcul, autres langages. Comparaison avec la programmation impérative. Caractéristiques de l'approche fonctionnelle : ordre supérieur, fonctionnel pur ou impur (avec ou sans effet de bord), formes d'évaluation (ordre normal ou applicatif).
- **01. Lisp et Haskell** Tutoriel des différences : éléments syntaxiques, notions d'expression et d'évaluation, interactivité. Typage dynamique et statique. Conditionnels, arithmétique, listes. Homoïconicité, réflexivité, méta-programmation. Comparatif final : 4 caractéristiques de l'approche fonctionnelle sur l'exemple de Newton-Raphson.
- **02. Ordre Supérieur** Rappels et généralités : définition, nommage, affectation, localité. Fonctions anonymes. Arguments fonctionnels et design patterns : mapping, filtrage, réduction. Retours fonctionnels et design patterns : composition, coupure d'opérateur, application partielle. Objets fonctionnels : structures de données fonctionnelles et lien avec le paradigme objet.
- **04. Évaluation et Scoping** Techniques d'évaluation : ordres normal et applicatif, conséquences sur la sémantique des langages. Scoping lexical et dynamique, structure de blocs, localité, fermetures lexicales, hygiène, capture, et collision.

Aop 2 – Approche Objet Avancée de la Programmation Ce cours traite d'une approche objet avancée, alternative à l'approche classique d'Aop 1. Bien que toujours basée sur la notion de classe, cette approche possède un certain nombre de traits distinctifs : typage dynamique, multi-méthodes, et réflexivité au travers d'un protocole méta-objet en sont les principaux. Le but est à la fois de montrer qu'il n'existe pas qu'une seule vision du paradigme objet, fût-il à base de classe, mais aussi que l'approche classique peut être vue comme un cas particulier de cette approche plus générale et donc plus expressive.

- **01. Clos – le Common Lisp Object System** Introduction : historique, caractéristiques distinctives. Classes et objets, instantiation, portée et accessibilité de l'information. Intégration classe – type. Héritage : rappels et caractéristiques distinctives. Polymorphisme : rappels, fonctions génériques et méthodes. Covariance, contravariance et displatch multiple (multi-méthodes).
- **02. Clos Avancé** Protocoles méta-objet : combinaison de méthodes, instantiation, `eql-specialiseurs`. Méta-classes et application : classes abstraites, finales, singleton et comptables. Réflexivité et application : problème du cercle elliptique.
- **03. Étude de Cas : Méthodes Binaires** Comparaison avec l'approche classique : covariance, contravariance. Résolution avec l'approche dynamique et le dispatch multiple. Validation du concept niveau utilisateur : introspection et méta-classes. Validation du concept niveau implémenteur : méta-programmation d'une extension de la couche objet.

04. Étude de cas : Design Patterns (le Visiteur) Introduction et historique des design patterns. Exemple du visiteur en C++. Réimplémentation naïve en Lisp. Déconstruction des idiotismes de l'approche classique. Mise en évidence de motifs fonctionnels plutôt qu'orientés objet. Extension aux visiteurs à état.

Typo – Typographie : Art, Artisanat, et Science Ce cours est une introduction au monde de la typographie, c'est-à-dire l'art de mettre en forme des documents essentiellement textuels, et à son histoire. L'objectif est de fournir une vision très large de la discipline, à la fois sur les plans scientifiques, techniques, historiques, et esthétiques. On y aborde en particulier les liens entre la beauté et la lisibilité d'un texte, les styles et usages typographiques, les outils permettant de composer en très haute qualité, et les enjeux techniques et algorithmiques sous-jacents.

01. Histoire de l'Écriture Systèmes de proto-écriture et écriture : non alphabétiques et alphabétiques, taxonomie (logogrammes, phonogrammes). Supports d'écriture : volumen et rotulus, codex, livre, manuscrit *vs.* imprimé, supports contemporains (physiques et dématérialisés).

02. Histoire de l'Impression Préhistoire : estampillage, xylographie, impression par bloc. Histoire : caractères mobiles en extrême-orient et en Europe. Presses : lit-plat, cylindriques, rotatives. Linotypes, monotypes, lumitype.

03. Histoire des Fontes Histoire : antique (influences médiévales et humanistes, naissance du romain), transition (romain moderne), contemporain (égyptiennes, sans-serif, art nouveau). Taxonomie : classifications, vox atypi. Anatomie d'un caractère, chasse, ligature, crénage.

04. L'Ère Numérique Introduction : mondes typographiques pré- et post-numériques. Premiers systèmes : machines dédiés, temps partagé et markup, systèmes typographiques contemporains. Tracé de caractères : par segments, plotters, trames et bitmaps, par contour, ductus. Formats de polices : Adobe et PostScript, TEX et METAFONT, guerre des polices, Type 1, TrueType, OpenType, polices variables.

05. Justification de Paragraphe : le Knuth-Plass Introduction et historique. Le modèle de TEX : boîtes, glues, critères esthétiques, tolérance, pénalités, démérites. Modélisation du problème : graphe des solutions, plus court chemin, optimisation par programmation dynamique.

05. L'Art de la Composition Introduction : critères typographiques, finalité, notion de gris typographique. Macro-typographie : rectangle d'empagement, interlignage. Ortho-typographie : règles, conventions et usages, internationalisation, multilinguisation. Micro-typographie : modèle de TEX, saillie, expansion, interlettrage, intermottage, lignes lavées, à voleur, trompeuses, veuves, orphelines, similarités.

2.2 Anciens Cours

Les cours suivants ne sont plus donnéset ne sont décrits que brièvement. Pour la plupart, les supports de cours correspondants sont archivés mais continuent d'être accessibles publiquement. Le nombre d'étudiants concernés variait d'une cinquantaine (Enst, Ensta, Upmc, Beaux Arts) à 350 (Epita) en plusieurs groupes.

Intitulé	Lieu	Type	Niveau	Volume	Période
Synthèse d'Images	Ensta/ Enst/ Upmc	CM + TD	M2	12h	2000 – 2010
Systèmes d'Exploitation	Epita	CM	L3	30h	2000 – 2017
Réalité Virtuelle	Enst/ Beaux Arts	CM + TD	L3	9h	2000 – 2010
Analyse Lexico-Syntaxique	Epita	TD	L1	15h	1999 – 2005
Programmation Lisp	Ionis stm	TD	MBA	15h	2011 – 2018

Le volume horaire est donné en heures par étudiant

2.3 Approche Pédagogique

Sans que cela ne devienne une doctrine, ma vision personnelle de la pédagogie est influencée par la théorie du socioconstructivisme, et l'approche par compétences. En particulier, je suis convaincu que les étudiants doivent occuper une place très active dans leurs apprentissages, et que le savoir théorique doit être consolidé par la mise en pratique.

Mes cours magistraux sont extrêmement interactifs. Ils sont conduits sur un mode conversationnel plus que magistral : les étudiants sont amenés à anticiper sur la suite par des sollicitations et un questionnement quasi-permanent. Les échanges entre étudiants sont également favorisés, ce qui est d'autant plus bénéfique qu'à l'Epita, école spécialisée en informatique, nombre d'entre eux sont passionnés et arrivent par conséquent avec un bagage autodidacte non négligeable (bien qu'empirique, donc sans fondation solide).

Cette vision pédagogique influence également le contenu de mes cours, et ma façon de les construire et de les organiser. Dans le cas des cours de paradigmes de programmation par exemple, je ne fais pas de cours classiques de programmation *orientée objet* ou *fonctionnelle dans tel ou tel langage* comme on trouve le plus fréquemment dans la littérature. Ma vision du tronc commun d'un cycle ingénieur est qu'il doit ouvrir l'esprit des étudiants plutôt que de les rendre opérationnels sur telle ou telle technologie (mais avec des œillères). C'est pour cette raison que mes cours ne se basent jamais sur un seul langage de programmation (*cf.*

leurs noms : « Approches … » au pluriel). Au contraire, mon objectif est de leur montrer qu'il existe toujours une multitude de déclinaisons possibles d'un même concept, mais aussi de mettre en évidence les biais cognitifs et culturels dont ils sont souvent déjà victime (en particulier le biais des langages impératifs) afin d'éveiller leur vigilance.

Enfin, mes cours sont également conçus en lien étroit avec mes activités académiques de recherche fondamentale ou appliquée, ainsi qu'avec mon goût très prononcé pour la transversalité. À titre d'exemple, les chapitres 3 et 4 d'Aop 2 sont directement tirés de deux publications (Verna, 2008a, 2010a, nº 4, nº 5). Il en va de même du cours de typographie qui découle directement de mon travail de recherche appliquée (Verna, 2025a, 2024a,b, nº 6 [publication choisie], nº 48, nº 49), et qui est constitué pour moitié de contenu non technique (paléographie, considérations esthétiques, *etc.*).

2.4 Mise en Œuvre

En l'état, la mise en œuvre de l'enseignement tient nécessairement compte d'un certain nombre de contraintes telles le nombre d'étudiants, la logistique, la synchronisation avec les antennes régionales, *etc.* Globalement et dans la mesure du possible, elle est implémentée comme suit.

Les cours sont disponibles sur une plate-forme Moodle qui donne aux étudiants accès à l'ensemble des ressources correspondantes. Mes supports de cours sont fournis en mode « animé » aux autres enseignants, et en mode « figé » aux étudiants. Chaque chapitre de cours magistral dispose d'une bibliographie qui est rassemblée sous forme d'hyperliens en fin de support. Il ne s'agit que des références citées explicitement pendant les cours ; le reste (p.ex. ouvrages de référence) est listé sur la page Moodle du cours directement.

Les cours magistraux sont évalués en contrôle continu par des QCM. La plupart des chapitres de cours sont enrichis de « code companion », disponibles publiquement avec les supports. Ce sont des versions intégrales et fonctionnelles de tous les exemples que je présente en cours sous forme d'extraits. Ce matériel permet aux étudiants de s'exercer en autonomie. Par ailleurs, les cours sont intégralement disponibles en vidéo (internes à l'école). En fonction du temps et grâce à toutes ces modalités, les étudiants sont à même de reprendre les cours à tête reposée et de revenir vers les enseignants dans un format de remédiation.

Pour la consolidation des savoirs, chaque chapitre de cours magistral est accompagné d'une séance de travaux pratiques ou dirigés qui sont évalués et notés (les sujets ne sont donc pas publiques). Bien que les rendus soient individuels, les étudiants sont encouragés à travailler en binômes afin de favoriser les échanges de points de vue. Dans le cas de la programmation objet ou fonctionnelle par exemple, une expérience récente montre qu'un format particulièrement bien reçu des étudiants est celui où le binôme tente de faire le même exercice, chacun dans un langage différent, tout en confrontant les solutions choisies, leurs caractéristiques, et leurs limites.

Dans le cas du cours de typographie qui est moins technique que les cours de paradigme, mon choix s'est porté sur une validation par mini-projet. Les projets s'étendent sur 2 à 4 semaines, sont effectués en groupes de 3 à 5, et font l'objet d'une soutenance orale dans laquelle chaque membre du groupe doit prendre part activement à la présentation.

3 Activités de Recherche

3.1 Stratégie et Trajectoire

Mon travail de recherche fondamentale se concentre sur l'expressivité et la performance des paradigmes de programmation dynamiques (« dynamique » dans le sens des langages dynamiquement typés). Dans un contexte multi-paradigme en particulier, les questions d'orthogonalité sont prépondérantes, ainsi que les aspects de généricité, de réflexivité et de méta-programmation. En parallèle, je poursuis un travail de recherche appliquée en typographie numérique et en ingénierie documentaire. Enfin, quand le contexte y est favorable, je m'intéresse également aux aspects les plus transversaux et pluridisciplinaires de l'informatique.

3.1.1 Motivation

Pendant de nombreuses années, l'industrie a privilégié une approche « ascendante » des langages de programmation, c'est-à-dire une évolution et un gain en abstraction par incréments successifs au dessus des langages historiques proches de l'architecture de Von Neumann. Dans le même temps, des paradigmes pourtant très anciens, tels la programmation fonctionnelle, ont été délaissés car trop avant-gardistes : soit leur utilité n'était pas comprise, soit le coût de leur implémentation était rédhibitoire. La situation a fini par changer cependant : grâce à l'augmentation de la puissance de calcul, au métier de la compilation, ainsi qu'à la reconnaissance de leur valeur ajoutée, l'avant-garde a fini par rentrer dans le rang (en témoignent par exemple la prolifération des langages de script dynamiques ou encore l'incorporation de traits fonctionnels dans les langages industriels « mainstream »).

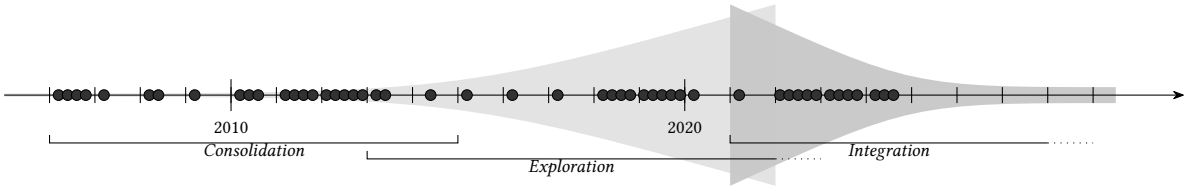
3.1.2 Enjeu

Une caractéristique importante du paysage logiciel contemporain est l'hétérogénéité. Même si elle tend à diminuer aujourd'hui, la dichotomie historique entre paradigmes avant-gardistes et « mainstream » conduit l'industrie à développer des solutions multi-langages (p.ex. « full stack ») pour tirer le meilleur profit des points forts de chacun (ou bien, vu sous un autre angle, pour

éviter d'affronter les points faibles de chacun). Je crois cependant qu'une approche à la fois multi-paradigme et uni-langage est non seulement possible, mais également préférable (ne serait-ce que pour faciliter le développement, la maintenance, et l'évolution du logiciel). Mon projet de recherche se distingue donc par son objectif de créer un cadre homogène et unifié pour l'ingénierie logicielle multi-paradigme (pour les aspects fondamentaux) et documentaire (pour la partie applicative).

3.1.3 Mise en œuvre

Vers la fin des années 2000, il m'est apparu que le langage Lisp est un véhicule pertinent pour mener ce projet à bien. Justifier de ce choix est hors de propos ici, mais une explication détaillée se trouve au début de mon rapport d'habilitation [2]. En bref, c'est un langage qui se prête particulièrement bien à l'expérimentation de par sa grande extensibilité (cœur minimaliste et homoïconicité), qui couvre dès le départ un très large éventail de paradigmes de programmation, et dont le standard Ansi en fait aussi un langage utilisé dans l'industrie. Depuis ce moment, ma stratégie de recherche s'articule donc selon trois axes, dont le développement temporel est figuré sur le diagramme ci-dessous [3].



1. La phase de « consolidation » avait pour but de replacer Lisp dans un contexte de légitimité académique. Lisp est un langage principalement utilisé dans l'industrie, ce qui explique la rareté du matériel académique jusque dans les années 2010. Le but de cette période était donc de créer une base de publication justifiant de l'utilité et de la légitimité de ce langage pour l'expérimentation en génie logiciel et documentaire, ainsi que de justifier de ces forces industrielles auprès du monde de la recherche. En parallèle l'idée était également de donner un nouveau souffle à la communauté, en renforçant notamment les liens entre les mondes académique et industriels, ainsi qu'en attirant la jeune génération.
2. Une fois la légitimité académique de ce langage démontrée, la phase d'« exploration » consiste à développer de nouveaux paradigmes, enrichir ou améliorer les paradigmes existant, étudier leurs mérites comparatifs (expressivité, performance), et analyser leurs interactions (composabilité, orthogonalité). Du point de vue applicatif, cette phase consiste également à explorer l'ingénierie documentaire (typographique notamment) au travers du prisme homogène de ce langage.
3. Enfin, la phase d'« intégration » consiste à recueillir les retombées de l'activité exploratoire, c'est-à-dire contribuer à rebours à la communauté Lisp via des propositions d'extensions ou de modifications de sa spécification, le maintien et la documentation de son écosystème et son rayonnement académique et industriel. Il s'agit aussi de propager les résultats obtenus à d'autres communautés en généralisant les concepts et en explorant leur applicabilité à d'autres langages dynamiques, ou bien en contribuant à des domaines théoriques par le fruit de la recherche faite dans ce langage. Enfin, du point de vue applicatif, il s'agit de concrétiser la recherche en typographie numérique par des réalisations concrètes, ou bien d'attaquer d'autres domaines applicatifs que l'ingénierie documentaire.

3.2 Résultats

Note : cette section n'est pas exhaustive. Elle décrit les résultats les plus notables en lien avec la trajectoire et la stratégie de recherche décrite précédemment.

3.2.1 Consolidation

En ce qui concerne le renforcement de l'assise académique de Lisp, on retiendra essentiellement trois points.

1. Il a été démontré expérimentalement que contrairement à une croyance très répandue, la très haute dynamicité du langage n'entraîne pas nécessairement un coût en performance de nature à rendre Lisp inutilisable en production (Verna, 2006a, 2009a, nº 42, nº 32). La première publication mentionnée a obtenu le prix du meilleur article de son workshop (European Lisp Workshop 2006) et a débouché sur deux conférences invitées : l'une au LaBRI de Bordeaux (Verna, 2006b, nº 52) et l'autre à la Vrije University de Brucelles (Verna, 2010b, nº 51).
2. La grande expressivité du langage Lisp (génie logiciel multi-paradigme, orthogonalité) a été démontrées académiquement à plusieurs reprises (Verna, 2008a, 2010a, nº 4, nº 5), en particulier en ce qui concerne l'application de la méta-programmation au mélange des paradigmes objet et fonctionnel. Ce travail a par la suite débouché sur une invitation à écrire un chapitre de livre traitant de l'application de ces technologies aux DSL (Verna, 2012a, nº 1).

2. https://hal.science/hal-02988180v1

3. Les points noirs correspondent à la densité de publication. La première zone creuse (2015 –2018) correspond à une période très chargée au niveau enseignement (reprise en main, refonte, et conception de nouveaux cours). La deuxième (2020 – 2021) est la période qui suit immédiatement la préparation et la soutenance de mon habilitation.

3. L'écosystème de ce langage (en particulier le lien entre mondes académiques et industriels) a été considérablement renforcée par la création en 2008 d'ELS (European Lisp Symposium[4]), dont je suis co-fondateur et président du comité de pilotage. Cette conférence s'est progressivement imposée comme premier événement international de la communauté (elle est également ouverte aux autres dialectes de Lisp). Aujourd'hui, elle n'a plus d'« européen » que le nom puisqu'elle rassemble chaque année entre 90 et 120 participants venant non seulement d'Europe, mais d'aussi loin que les États-Unis ou le Japon (sans compter les participants en distanciel depuis son hybridation en 2020). ELS a obtenu le statut « In-Cooperation-With » d'ACM (Association for Computing Machinery) pendant 8 ans (2016–2023), statut que nous avons décidé d'abandonner afin de poursuivre un référencement Scopus et un éventuel classement dans le Core (en cours).

Enfin, en termes d'ingénierie documentaire et typographique, la communauté TEX a été sensibilisée aux avantages que présente Lisp pour une éventuelle modernisation du système de Donald Knuth (Verna, 2012b, 2013a, nº 25, nº 27).

3.2.2 Exploration

En termes exploratoires, les trois pistes principales de recherche fondamentale sont les suivantes : l'application innovante de paradigmes existant, l'extension ou l'amélioration de paradigmes instables ou embryonnaires, et le développement de nouveaux paradigmes.

1. **Exemple d'application innovante : optimisation orientée contexte.** La programmation orientée contexte est une extension de l'approche objet permettant de représenter dynamiquement et de façon abstraite des variations comportementales dépendantes du contexte d'exécution. C'est une approche visant un très haut niveau de généricité, mais qui induit notoirement un coût important en performance. Nous avons montré que paradoxalement, il est possible d'utiliser cette approche à des fins d'optimisation, prenant ainsi le contre-pied des approches classiques utilisant des techniques de généricité statique (Senta et al., 2012 ; Verna and cois Ripault, 2015, nº 26, nº 41). En d'autres termes, il devient possible d'être à la fois générique et performant dans une approche objet traditionnelle, sans pour autant surcharger le modèle ou rater des optimisation transversales.
2. **Exemple d'amélioration de paradigme : combinateurs de méthode.** Tandis que dans l'approche objet traditionnelle, le mécanisme de sélection des méthodes polymorphes est câblé, celui de Lisp est programmable au travers des « combinaisons de méthodes ». Cependant, ce concept n'était pas arrivé à maturité au moment de la standardisation du langage : il est par conséquent partiel, mal défini, et parfois contradictoire. Nous avons proposé une clarification du concept, une implémentation plus propre au travers d'un protocole méta-objet, ainsi qu'une extension appelé « combinateurs alternatifs » permettant à une fonction générique de changer sa propre politique de dispatch au cours du temps (Verna, 2018b, 2023a, nº 11 [publication choisie], nº 21).
3. **Exemple de conception de paradigme : expressions rationnelles de type.** Les langages dynamiques permettent d'exprimer des séquences de valeurs hétérogènes sur leur type, tout en sachant optimiser les cas homogènes. Cependant, la question de l'optimisation quand une séquence hétérogène reste régulière se pose. Nous avons proposé un nouveau paradigme nommé RTE (Regular Type Expressions) utilisant des expressions rationnelles appliquées aux types de données. Ce paradigme permet de spécifier de façon déclarative des régularités dans les séquences de types, débouchant soit sur une plus grande facilité de vérification, soit sur des stratégies d'optimisation quand la séquence est connue (Newton et al., 2016, 2017 ; Newton and Verna, 2018a,b, nº 20, nº 22, nº 23, nº 40).

En parallèle à ces aspects fondamentaux, l'exploration en ingénierie documentaire et typographique a donné lieu à la création d'une plateforme d'expérimentation libre et nommé Etap (Experimental Typesetting Algorithms Platform[5], Verna 2022, 2023b, nº 10, nº 14) ainsi qu'à des premiers résultats en matière de typographie haute qualité (Verna, 2024b, nº 6 [publication choisie]).

3.2.3 Intégration

— Le travail sur l'optimisation orientée contexte avait été initialement appliqué au traitement d'images. Il s'en est suivi une thèse (maintenant soutenue) sur le mélange des paradigmes statiques et dynamique dans ce domaine (Esteban et al., 2022a, nº 12 [publication choisie]).
— Des discussions sont actuellement en cours sur l'incorporation des combinateurs de méthodes dans SBCL, l'une des principales implémentations libres de Lisp. Les mainteneurs d'ABCL (une implémentation de Lisp au dessus de la JVM) ont également exprimé leur intérêt. Plus généralement, je suis l'un des éditeurs du Common Document Repository[6] qui est le dépôt central de recensement de toutes les propositions d'extension, de clarification, ou de modification du standard Lisp (l'équivalent des SRFI de Scheme). Je suis également l'auteur de trois de ces propositions (Verna, 2012c, 2011a,b, nº 55, nº 56, nº 57).
— Le travail de recherche sur les expressions rationnelles de type a fait l'objet d'une implémentation libre[7] en Lisp, ainsi que d'un portage vers d'autres langages comme Python[8] ou Clojure[9]. Par ailleurs, l'implémentation du moteur de RTE

4. https://european-lisp-symposium.org
5. https://github.com/didierverna/etap
6. https://cdr.common-lisp.dev/
7. https://github.com/jimka2001/cl-rte
8. https://gitlab.lre.epita.fr/jnewton/python-rte
9. https://gitlab.lre.epita.fr/jnewton/clojure-rte

a débouché sur une étude théorique des diagrammes de décision binaires et des automates finis (Newton and Verna, 2019a,b, nº 2 [publication choisie], nº 16).
— L'exploration des aspects réflexifs de Lisp a débouché sur deux bibliothèques de génération automatique et non intrusive de documentation : Declt [10] et Quickref [11]. Ces outils sont aujourd'hui utilisés pour la création et la maintenance automatiques de manuels de référence de plus de 2200 bibliothèques, accessibles au travers d'un site web officiel [12]. Ce site constitue à l'heure actuelle la plus grosse base documentaire de l'écosystème Lisp.
— Ce projet a récemment été augmenté d'une cohorte disponible au télé chargement sur le même site. Cette cohorte est constituée de plus d'un demi million d'entités programmatiques (fonctions, macros, classes, variables, *etc.*) et de leurs relations (hiérarchies de classes, spécialiseurs de méthodes, *etc.*). Cette cohorte est encore en développement, mais elle a déjà fait l'objet d'une étude préliminaire (Verna, 2024c, nº 8) offrant de précieuses informations sur la morphologie du code Lisp tel qu'il est développé aujourd'hui.
— La recherche en typographie numérique de haute qualité a abouti cette année à la conception d'une extension au « Knuth-Plass », l'algorithme de justification de paragraphe de TEX (Verna, 2024b, nº 6 [publication choisie]). Ce travail est actuellement en phase d'intégration dans LuaMetaTEX, le successeur de LuaTeX qui est maintenu par les mêmes personnes. Par ailleurs, je viens d'être invité à faire partie du comité de programme de DocEng [13], la conférence où le travail a été présenté.

3.3 Transversalité

En marge de l'activité de recherche décrite dans les sections précédentes, j'éprouve également un très vif intérêt pour la transversalité, c'est-à-dire la mise en évidence de liens qui unissent des disciplines a priori décorrélées, les disciplines en question pouvant même ne pas faire partie du monde scientifique. Cette activité reste cependant marginale et par conséquent occasionnelle. Jusqu'ici, elle s'est exprimée de deux façons.

En 2010, j'ai proposé une nouvelle vision des aspects évolutionnistes du monde logiciel en établissant une analogie comportementale entre l'écosystème LATEX et la virologie. En considérant qu'une classe LATEX constitue le code génétique d'un document, et que les styles ajoutés sont des virus, on peut expliquer de façon tout à fait fascinante les interactions entre classes, styles, et documents en des termes virologiques. Tandis que les biologistes ont tendance à considérer la nature comme du « bricolage » et l'ingénierie comme quelque chose de plus « propre », ce travail a montré que la réciproque était toute aussi vraie. Il s'en est suivi deux publications (Verna, 2010c, 2011c, nº 28, nº 31) ainsi qu'une invitation à être « keynote speaker » sur le sujet (Verna, 2014, nº 50). Ce travail m'a également valu d'être cité par Antoine Danchin [14], biologiste français porteur de la vision qu'une cellule biologique est assimilable à un ordinateur vivant.

En 2018, j'ai souhaité développer une intuition alors vieille de quelques années : la recherche des liens qui unissent mes trois passions : le langage Lisp, le Jazz (activité qui est la raison de mon temps partiel), et l'Aïkido, un art martial que je pratique depuis très longtemps et que j'enseigne également. L'idée de ce travail est de considérer chacune de ces disciplines comme une langue (rationnelle pour Lisp, émotionnelle pour le Jazz, et corporelle pour l'Aïkido), de montrer que leur fonctionnement est identique, et qu'elles correspondent chacune à la même tournure d'esprit (en particulier sur le plan esthétique). Au delà de l'intérêt purement philosophique, ce travail a également le mérite d'expliquer en quoi nous pouvons entretenir un rapport intime et émotionnel avec les langages de programmation, et par conséquent pourquoi il y a tant de guerres de religion à leur sujet. Ce travail a été publié sous la forme d'un essai (Verna, 2018a, nº 3 [publication choisie]) qui a été primé. Plutôt que d'en faire une présentation classique, j'en ai par ailleurs tiré une sorte de court métrage disponible sur YouTube [15].

4 Activités de Recherche

4.1 Comités de Pilotage de Conférences Internationales

— Onward ! (ACM SIGPLAN Symposium on New Ideas in Programming and Reflections on Software), 2020–2023.
— ELS (European Lisp Symposium), co-fondateur et président depuis 2008.
— ELW (European Lisp Workshop), 2007–2010.

4.2 Comités de Lecture de Revues Internationales

— Art, Science, and Engineering of Programming Journal, 2021.

En Bref

Co-fondateur et président du comité de pilotage d'ELS (European Lisp Symposium)

- Comités de pilotage de conférences internationales : 3 / cumul 22 ans
- Comités de lecture de revues internationales : 2 / cumul 5 ans
- Chaires de conférences internationales : 4 / cumul 7 ans
- Comités de programme de conférences internationales : 11 / cumul 30 ans

10. https://github.com/didierverna/declt
11. https://gitlab.common-lisp.net/quickref/quickref
12. https://quickref.common-lisp.net
13. https://doceng.org/
14. https://doi.org/10.1111/1462-2920.12270
15. https://www.youtube.com/watch?v=CbciCrQCpkI

— Journal of Universal Computer Science, 2007 – 2010.

4.3 Chaires de Conférences Internationales

— Onward ! (ACM SIGPLAN Symposium on New Ideas in Programming and Reflections on Software) Essays, 2020.
— ILC (International Lisp Conference), 2014.
— ELS (European Lisp Symposium), 2011.
— ELW (European Lisp Workshop), 2007–2010.

4.4 Comités de programme de conférences internationales

— DocEng (ACM SIGWEB Symposium on Document Engineering), 2025.
— Onward ! (ACM SIGPLAN Symposium on New Ideas in Programming and Reflections on Software) Essays, 2024.
— ICCQ (International Conference on Code Quality), 2023–2025.
— ILC (International Lisp Conference), 2012, 2014.
— DLS (Dynamic Languages Symposium), 2011, 2013, 2015.
— SAC (ACM SIGAPP Symposium on Applied Computing), 2012, 2013, 2015.
— ELS (European Lisp Symposium), 2011—2016.
— COP (Context-Oriented Programming Workshop), 2010, 2013, 2016, 2018.
— FARM (ACM SIGPLAN Functional Art, Music, Modelling, and Design Workshop), 2018.
— ELW (European Lisp Workshop), 2007—2010.
— DyLa (Dynamic Languages and Applications Workshop), 2013, 2014.

4.5 Autres comités

— Most Notable Onward ! 2010 Paper Award (comité de sélection 2020).
— Fondateur et premier organisateur du séminaire Performance et Généricité du Lre en 2008.
Une à deux sessions par mois.

4.6 Relecteur Technique de Livre

— GNU Autoconf, automake, libtool. Gary V. Vaughan, Ben Elliston, Tom Tromey and Ian Lance Taylor. *New Riders, October 2000.* Isbn 9781578701902.

5 Activités d'Encadrement

En Bref

• Thèses en cours	2
• Thèses soutenues	2
• Stages de M2	3
• Stages de M1	1
• Étudiants-Chercheurs	9
• Jurys de thèses / suivi	3

5.1 Thèses

5.1.1 En Cours

— **Directeur. Co-encadrement 50%.**
Maya Mouhammad. Prédiction des Intrusions Réseau par Apprentissage Fédéré et Interprétable des Anomalies dans un Environnement Distribué. Epita/Lre, Sorbonne Université/Upmc, APL Data Center, Edite de Paris.
— **Directeur. Co-encadrement 34%.**
Khaoula Sghaier. Sécurisation des Échanges de Communication V2X des Unités de Contrôle Télématiques. Epita/Lre, Sorbonne Université/Upmc, Telecom SudParis/SCN, VEDECOM, Edite de Paris.

5.1.2 Soutenues

— **Directeur. Co-encadrement 34%.**
Baptiste Esteban. Implémentations Modernes et Performantes de Techniques Morphologiques pour le Traitement d'Images et l'Estimation de Bruit. Epita/Lre, Sorbonne Université/Upmc, Edite de Paris. Soutenue le 21 Décembre 2023.
— **Co-encadrement 90%.**
Jim E. Newton. Representing and Computing with Types in Dynamically Typed Languages. Epita/Lre, Sorbonne Université/Upmc, Edite de Paris. Soutenue le 20 Novembre 2018.

5.2 Stages de Master

— Clément Bonnefoy. Interface et programmation graphique pour une bibliothèque de traitement d'images. *Master STL, Upmc, 2016.*

— Krista Druskhu. Étude comparative de langages pour la conception de DSL. *Master STL, Upmc, 2015.*
— Vuong Ha Minh et Tung Nguyen Duc. Pérénisation d'une plateforme de benchmarking pour Common Lisp. *Master Génie Logiciel, Université de Bordeaux I, 2010.*

5.3 Étudiants-Chercheurs

Le Lre intègre des étudiants pour la durée de leur cursus ingénieur. Nous leur dispensons une formation par la recherche en les faisant participer à nos projets. Cela s'effectue en parallèle du cursus classique pour le tronc commun et cela devient formalisé par une majeure « double compétence orientée recherche » en fin d'études. Ils ont donc un ou plusieurs mini-sujets de recherche à mener et à valider sous notre supervision durant leur passage au Lre. Ces étudiants peuvent véritablement être considérés comme de « mini-thésards ». Pour s'en convaincre, dans la liste ci-dessous, les étoiles (⋆) indiquent ceux qui ont été publiant avant leur sortie d'Epita (donc, avant même leur 3e cycle). On retrouve ainsi leurs noms dans la section Publications. Enfin, le cas échéant, ceux dont le parcours scolaire a continué sur un DEA (pour les plus anciens), un Master (plus les plus jeunes) puis une thèse, sont également indiqués.

- Aline Vongsavanh
⋆ Antoine Hacquard
⋆ Léo Valais
⋆ François Ripault
⋆ Laurent Senta
⋆ Christopher Chedeau
- Simon Odou, Master et thèse (Lri / Paris-Sud XI)
⋆ Guillaume Pitel, DEA SC et thèse (Limsi, Orsay)
⋆ Yoann Fabre, DEA Iarfa et thèse (Upmc)

5.4 Jurys de thèse et comités de suivi

— **Jury de thèse / rapporteur.** Marco Heisig. Design and Implementation of the Parallel Programming Language Petalisp. *Fau Erlangen, Allemagne. Juillet 2025. Directeur : Harald Köstler.*
— **Jury de thèse / rapporteur.** Jorge Vallejos. Modularising Context Dependency and Group Behaviour in Ambient-Oriented Programming Languages. *Software Languages Lab, Vrije Universiteit Brussel, Juillet 2011. Direction : Theo D'Hondt, Wolfgang De Meuter, Pascal Costanza.*
— **Comité de suivi.** Amaury Curiel. Arbres de Décision Binaires et Graphes Dirigés Acycliques. *Sorbonne Université/Upmc/LIP6/APR.*
— **Comité de suivi.** Bertrand Petit-Heidelein. Informatique Diffuse et Arts de la Scène : Musique Massivement Interactive. *Inria/Cirm, Stic (école doctorale). Direction : Manuel Serrano, François Paris.*

6 Publications Choisies

Cette section présente la liste des 5 publications choisies pour être jointes au dossier et justifie les choix qui ont été fait. La sélection a été établie sur des publications récentes, mais de façon a couvrir malgré tout un large éventail du travail, en lien avec la stratégie de recherche décrite en Section 3 page 5. La sélection a par ailleurs été pensée de manière à illustrer la variété des cibles impactés par le travail. Chaque article joint est précédé par une traduction de son « abstract » en français.

Article n° 6 Similarity problems in paragraph justification : an extension to the Knuth-Plass algorithm. Didier Verna. In *Proceedings of the ACM SIGWEB Symposium on Document Engineering 2024*, DocEng'24, pages 127–130, New York, NY, USA, August 2024. Association for Computing Machinery. Isbn 9798400711695. Doi 10.1145/3685650.3685666.

Cet article montre l'un des résultats du travail de recherche en typographie numérique (cf. Section 3.2.2 page 7), et son intégration dans l'écosystème de TEX (cf. Section 3.2.3 page 7).

Article n° 11 A MOP-based implementation for method combinations. Didier Verna. In *ELS'23, 16th European Lisp Symposium*, pages 6–15, Amsterdam, Netherlands, April 2023. Isbn 9782955747476. Doi 10.5281/zenodo.7818680.

Cet article montre l'un des résultats du travail de recherche sur les protocoles méta-objet et la réflexivité (cf. Section 3.2.2 page 7), ainsi que son intégration dans un compilateur Lisp (cf. 3.2.3 page 7).

Article n° 12 The cost of dynamism in static languages for image processing. Baptiste Esteban, Edwin Carlinet, Guillaume Tochon, and Didier Verna. In *Proceedings of the 21st ACM SIGPLAN International Conference on Generative Programming : Concepts & Experiences*, Auckland, New Zealand, December 2022.

Cet article illustre le fruit d'un travail effectué sur les questions de performance et de généricité au travers d'une direction de thèse, et montre son rayonnement à l'extérieur du champ initial d'investigation : les langages statiques C++ et Rust, et leur application au traitement d'images.

Article n° 2 A theoretical and numerical analysis of the worst-case size of reduced ordered binary decision diagrams. Jim E. Newton and Didier Verna. *ACM Transactions on Computational Logic*, 20(1), January 2019. ISSN 15293785. DOI 10.1145/3274279.

Cet article montre comment le travail sur les expressions rationnelles de type (cf. Section 3.2.2 page 7), au départ orienté langage et effectué au travers d'un encadrement de thèse, a débouché sur une étude théorique des diagrammes de décision binaires et des automates finis, dépassant largement le cadre initial d'investigation (cf. 3.2.3 page 7).

Article n° 3 Lisp, Jazz, Aikido. Didier Verna. *The Art, Science and Engineering of Programming Journal*, 2(3), March 2018. DOI 10.22152/programming-journal.org/2018/2/10.

Cet article est une publication un peu hors norme. Elle illustre mon goût pour la transversalité (cf. Section 3.3 page 8) puisqu'elle tisse des liens entre l'informatique, les arts, et les arts martiaux. Bien que plus philosophique que scientifique, elle a trouvé sa place dans un journal d'informatique qui lui a décerné le prix de la sélection du jury.

En Bref

• Keynotes et invitations	5
• Chapitres de livre	1
• Relecture technique de livre	1
• Revues internationales	4
• Conférences internationales	35
• Workshops internationaux	4
• Conférences francophones	4
• Autres	9
• Exposés Oraux	14

7 Publications

7.1 Chapitres de Livres

1. Extensible languages: Blurring the distinction between DSLs and GPLs. Didier Verna. In Marjan Mernik, editor, *Formal and Practical Aspects of Domain-Specific Languages: Recent Developments*, chapter 1. IGI Global, September 2012. ISBN 9781466620926. DOI 10.4018/978-1-4666-2092-6.ch001.

7.2 Articles de Revues

2. A theoretical and numerical analysis of the worst-case size of reduced ordered binary decision diagrams. Jim E. Newton and Didier Verna. *ACM Transactions on Computational Logic*, 20(1), January 2019. ISSN 15293785. DOI 10.1145/3274279.
3. Lisp, Jazz, Aikido. Didier Verna. *The Art, Science and Engineering of Programming Journal*, 2(3), March 2018. DOI 10.22152/programming-journal.org/2018/2/10.
4. Revisiting the visitor: the just do it pattern. Didier Verna. *Journal of Universal Computer Science*, 16(2):246–271, 2010. DOI 10.3217/jucs-016-02-0246.
5. Binary methods programming: the CLOS perspective. Didier Verna. *Journal of Universal Computer Science*, 14(20): 3389–3411, 2008. DOI 10.3217/jucs-014-20-3389.

7.3 Articles de Conférences Internationales

6. Towards more homogeneous paragraphs. Didier Verna. In *Proceedings of the ACM Symposium on Document Engineering 2025*, DocEng'25, New York, NY, USA, September 2025. Association for Computer Machinery. ISBN 9798400711695. DOI 10.1145/3704268.3742696.
7. Similarity problems in paragraph justification: an extension to the Knuth-Plass algorithm. Didier Verna. In *Proceedings of the ACM SIGWEB Symposium on Document Engineering 2024*, DocEng'24, pages 127–130, New York, NY, USA, August 2024. Association for Computing Machinery. ISBN 9798400711695. DOI 10.1145/3685650.3685666.
8. A large scale format compliance checker for TEX font metrics. Didier Verna. In Barbara Beeton and Karl Berry, editors, *TUG'24, 42nd TEX Users Group Conference*, volume 45, pages 221–226. TEX Users Group, TEX Users Group, September 2024. DOI 10.47397/tb/45-2/tb140verna-tfm.
9. The Quickref cohort. Didier Verna. In *ELS'24, 17th European Lisp Symposium*, Vienna, Austria, May 2024. ISBN 9782955747483. DOI 10.5281/zenodo.10947962.
10. Structural analysis of the additive noise impact on the α-tree. Baptiste Esteban, Guillaume Tochon, Edwin Carlinet, and Didier Verna. In *Proceedings of the 20th International Conference on Computer Analysis of Images and Patterns (CAIP)*, volume 14185 of *Lecture Notes in Computer Science Series*, Limassol, Cyprus, September 2023. Springer.
11. Interactive and real-time typesetting for demonstration and experimentation: ETAP. Didier Verna. In Barbara Beeton and Karl Berry, editors, *TUG'23, 41st TEX Users Group Conference*, volume 44, pages 242–248. TEX Users Group, TEX Users Group, 2023. DOI 10.47397/tb/44-2/tb137verna-realtime.
12. A MOP-based implementation for method combinations. Didier Verna. In *ELS'23, 16th European Lisp Symposium*, pages 6–15, Amsterdam, Netherlands, April 2023. ISBN 9782955747476. DOI 10.5281/zenodo.7818680.
13. The cost of dynamism in static languages for image processing. Baptiste Esteban, Edwin Carlinet, Guillaume Tochon, and Didier Verna. In *Proceedings of the 21st ACM SIGPLAN International Conference on Generative Programming: Concepts & Experiences*, Auckland, New Zealand, December 2022.

14. Estimation of the noise level function for color images using mathematical morphology and non-parametric statistics. Baptiste Esteban, Guillaume Tochon, Edwin Carlinet, and Didier Verna. In *Proceedings of the 26th International Conference on Pattern Recognition*, Montréal, Québec, August 2022.
15. Etap: Experimental typesetting algorithms platform. Didier Verna. In *ELS'22, 15th European Lisp Symposium*, pages 48–52, Porto, Portugal, March 2022. Isbn 9782955747469. Doi 10.5281/zenodo.6334248.
16. A corpus processing and analysis pipeline for quickref. Antoine Hacquard and Didier Verna. In *14th European Lisp Symposium*, pages 27–35, Online, May 2021. Isbn 9782955747452. Doi 10.5281/zenodo.4714443.
17. Finite automata theory based optimization of conditional variable binding. Jim E. Newton and Didier Verna. In *ELS'19, 12th European Lisp Symposium*, pages 26–33, Genova, Italy, April 2019. Isbn 9782955747438. Doi 10.5281/zenodo.2635402.
18. Implementing baker's `subtypep` decision procedure. Léo Valais, Jim E. Newton, and Didier Verna. In *ELS'19, 12th European Lisp Symposium*, pages 12–19, Genova, Italy, April 2019. Isbn 9782955747438. Doi 10.5281/zenodo.2646982.
19. Parallelizing Quickref. Didier Verna. In *ELS'19, 12th European Lisp Symposium*, pages 89–96, Genova, Italy, April 2019. Isbn 9782955747438. Doi 10.5281/zenodo.2632534.
20. Quickref: Common Lisp reference documentation as a stress test for Texinfo. Didier Verna. In Barbara Beeton and Karl Berry, editors, *TUG'19, 40th TEX Users Group Conference*, volume 40, pages 119–125. TEX Users Group, TEX Users Group, September 2019.
21. Strategies for typecase optimization. Jim E. Newton and Didier Verna. In *ELS'18, 11th European Lisp Symposium*, pages 23–31, Marbella, Spain, April 2018. Isbn 9782955747421. Doi 10.5281/zenodo.3405191.
22. Method combinators. Didier Verna. In *ELS'18, 11th European Lisp Symposium*, pages 32–41, Marbella, Spain, April 2018. Isbn 9782955747421. Doi 10.5281/zenodo.3247610.
23. Programmatic manipulation of Common Lisp type specifiers. Jim E. Newton, Didier Verna, and Maximilien Colange. In *ELS'17, 10th European Lisp Symposium*, pages 28–35, Vrije Universiteit Brussel, Belgium, April 2017. Isbn 9782955747414. Doi 10.5281/zenodo.3405363.
24. Type-checking of heterogeneous sequences in Common Lisp. Jim E. Newton, Akim Demaille, and Didier Verna. In *ELS'16, 9th European Lisp Symposium*, pages 13–20, AGH University of Science and Technology, Krakow, Poland, April 2016. Isbn 9782955747407. Doi 10.5281/zenodo.3405173.
25. The incredible tale of the author who didn't want to do the publisher's job. Didier Verna. In Barbara Beeton and Karl Berry, editors, *TUG'13, 34th TEX Users Group Conference*, volume 34. TEX Users Group, 2013.
26. TiCL: the prototype (Star TEX: the next generation, season 2). Didier Verna. In Barbara Beeton and Karl Berry, editors, *TUG'13, 34th TEX Users Group Conference*, volume 34. TEX Users Group, 2013.
27. Generic image processing with Climb. Laurent Senta, Christopher Chedeau, and Didier Verna. In *ELS'12, 5th European Lisp Symposium*, Zadar, Croatia, May 2012. Doi 10.5281/zenodo.3248934.
28. Star TEX: the next generation. Didier Verna. In Barbara Beeton and Karl Berry, editors, *TUG'12, 33rd TEX Users Group Conference*, volume 33. TEX Users Group, 2012.
29. Biological realms in computer science. Didier Verna. In *Onward!'11: the ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software Proceedings*, pages 167–176. ACM, October 2011. Isbn 9781450309417. Doi 10.1145/2089131.2089140.
30. Towards LATEX coding standards. Didier Verna. In Barbara Beeton and Karl Berry, editors, *TUG'11, 32nd TEX Users Group Conference*, volume 32, pages 309–328. TEX Users Group, 2011.
31. CLoX: Common lisp objects for XEmacs. Didier Verna. In *ELS'10, 3rd European Lisp Symposium*, Lisbon, Portugal, May 2010. Doi 10.5281/zenodo.3248958.
32. Classes, styles, conflicts: the biological realm of LATEX. Didier Verna. In Barbara Beeton and Karl Berry, editors, *TUG'10, 31st TEX Users Group Conference*, volume 31, pages 162–172. TEX Users Group, 2010.
33. CLOS efficiency: Instantiation. Didier Verna. In *ILC'09 International Lisp Conference*, pages 76–90, MIT, Cambridge, Massachusetts, USA, March 2009. ALU (Association of Lisp Users). Doi 10.5281/zenodo.3386206.
34. Binary methods programming: the CLOS perspective. Didier Verna. In *ELS'08, 1st European Lisp Symposium*, pages 91–105, Bordeaux, France, May 2008. Doi 10.5281/zenodo.3248977.
35. Action recognition: How intelligent virtual environments can ease human-machine interaction. Didier Verna. In Hal Thwaites and Scott Thrane Refsland, editors, *VSMM'00, Sixth International Conference on Virtual Systems and MultiMedia*, pages 703–713, Gifu Research and Development Foundation, Gifu, Japan, October 2000. International Society on Virtual Systems and MultiMedia, Ohmsha Press.
36. *Urbi et Orbi*: Unusual design and implementation choices for distributed virtual realities. Didier Verna, Yoann Fabre, and Guillaume Pitel. In Hal Thwaites and Scott Thrane Refsland, editors, *VSMM'00, Sixth International Conference on Virtual Systems and MultiMedia*, pages 714–724, Gifu Research and Development Foundation, Gifu, Japan, October 2000. International Society on Virtual Systems and MultiMedia, Ohmsha Press.

37. The multicast support in XEmacs. Didier Verna. In *m17n'99, 3nd International Symposium on Multilingual Environements*, Tsukuba, Japan, 1999.
38. Ergonomics and human-machine interaction concerns in Mule. Didier Verna. In *m17n'99, 3nd International Symposium on Multilingual Environements*, Tsukuba, Japan, 1999.
39. Can we define virtual reality? the MrIC model. Didier Verna and Alain Grumbach. In Jean-Claude Heudin, editor, *Virtual Worlds 98*, Lecture Notes in Artificial Intelligence, pages 29–41. Springer-Verlag, 1998.
40. Télé-opération et réalité virtuelle: Assistance à l'opérateur par modélisation cognitive de ses intentions. Didier Verna. In *IHM'97*, pages 205–206. Cépaduès-Éditions, 1997.

7.4 Articles de Workshops Internationaux

41. Recognizing hetergeneous sequences by rational type expression. Jim E. Newton and Didier Verna. In *Meta'18, Meta-Programming Techniques and Reflection Workshop*, Boston, MA, USA, November 2018.
42. Context-oriented image processing. Didier Verna and François Ripault. In *COP'15, Context-Oriented Programming Workshop*, 2015. Isbn 9781450336543. Doi 10.1145/2786545.2786547.
43. Beating C in scientific computing applications. Didier Verna. In *ELW'06, 3rd European Lisp Workshop*, Nantes, France, July 2006.
44. Augmented reality, the other way around. Didier Verna and Alain Grumbach. In M. Gervautz, A. Hildebrand, and D. Schmalstieg, editors, *EGVE, 5th Eurographics Workshop on Virtual Environments*, pages 147–156. Springer, 1999.

7.5 Articles de Conférences Francophones

45. Analyse structurelle de l'influence du bruit sur l'arbre alpha. Baptiste Esteban, Guillaume Tochon, Edwin Carlinet, and Didier Verna. In *29e Colloque sur le traitement du signal et des images*, Grenoble, France, August 2023. GRETSI - Groupe de Recherche en Traitement du Signal et des Images.
46. Estimation de la fonction de niveau de bruit pour des images couleurs en utilisant la morphologie mathématique. Baptiste Esteban, Guillaume Tochon, Edwin Carlinet, and Didier Verna. In *Proceedings of the 28st Symposium on Signal and Image Processing (GRETSI)*, Nancy, France, September 2022.
47. Généricité dynamique pour des algorithmes morphologiques. Baptiste Esteban, Edwin Carlinet, Guillaume Tochon, and Didier Verna. In *Proceedings of the 28st Symposium on Signal and Image Processing (GRETSI)*, Nancy, France, September 2022.
48. Définir le virtuel: une vision cognitive. Didier Verna. In *ReViCo'99, Réalité Virtuelle et Cognition*, Paris, France, December 1999.

7.6 Keynotes et Exposés sur Invitation

49. Traitement des similarités dans la justification de paragraphe. Didier Verna. Exposé GUTenberg, September 2025.
50. Justification de paragraphe: le knuth-plass. Didier Verna. Exposé GUTenberg, January 2024.
51. Biological realms in computer science. Didier Verna. Keynote at ACCU'14, apr 2014.
52. CLOS efficiency: Instantiation. Didier Verna. Invited Talk at the Vrije University of Brussels, 2010.
53. Scientific computing in lisp: Beyond the performance of C. Didier Verna. Invited Talk at LaBRI, Université de Bordeaux I, France, 2006.

7.7 Rapports et publications diverses

54. *(Dynamic (Programming Paradigms)) ;; Performance and Expressivity.* Didier Verna. PhD thesis, EDITE de Paris, Sorbone-Université, EPITA/LRDE, July 2020. Habilitation à Diriger les Recherches.
55. JSPP: Morphing C++ into JavaScript. Christopher Chedeau and Didier Verna. Technical Report 201201-TR, LRDE (EPITA Research and Development Laboratory), January 2012.
56. Standard output streams default behavior in terminal sessions. Didier Verna. Common Document Repository #11, 2012. Doi 10.5281/zenodo.3414042.
57. Clarification proposal for CLHS 22.3. Didier Verna. Common Document Repository #7, 2011. Doi 10.5281/zenodo.3413913.
58. File-local variables. Didier Verna. Common Document Repository #9, 2011. Doi 10.5281/zenodo.3414042.
59. LaTeX curricula vitae with the CurVe class. Didier Verna. *The PracTeX Journal*, (3), August 2006.
60. CV formatting with CurVe. Didier Verna. *TUGBoat, Communications of the TeX Users Group*, 22(4):361–364, December 2001. Issn 0896320.

61. *Télé-opération et Réalité Virtuelle: Assistance à l'Opérateur par Modélisation Cognitive de ses Intentions.* Didier Verna. PhD thesis, ENST (École Nationale Supérieure des Télécommunications de Paris), Paris, France, February 2000. ENST 00 E007.
62. Comment définir le virtuel ? le modèle MrIC. Didier Verna. Technical Report 97 D 008, ENST (École Nationale Supérieure des Télécommunications de Paris), 46 rue Barrault, 75013 Paris, France, 1997.

7.8 Exposés Oraux

63. A taste of Julia. Didier Verna. ACCU'16, April 2016.
64. A taste of Julia. Didier Verna. Séminaire Performance et Généricité du LRDE, April 2016.
65. La musique des programmes. Didier Verna. Soirée Thématique de l'EPITA: l'Esthétique en Informatique, 2016.
66. Referential transparency is overrated. Didier Verna. ACCU'15, April 2015.
67. The bright side of exceptions. Didier Verna. ACCU'13, April 2013.
68. Extensibility for DSL design and implementation: a case study in Lisp. Didier Verna. DSLDI'13, DSL Design and Implementation Workshop, April 2013.
69. DSLs from the perspective of extensible languages. Didier Verna. ACCU'12, April 2012.
70. Lisp extensibility: Impact on DSL design and implementation. Didier Verna. Tutorial at ILC'12, the International Lisp Conference, 2012.
71. Meta-circularity, and vice-versa. Didier Verna. ACCU'11, April 2011.
72. Clon, the command-line options nuker. Didier Verna. ILC'10, the International Lisp Conference, 2010.
73. Revisiting the visitor: the just do it pattern. Didier Verna. ACCU'09, April 2009.
74. Performance and genericity: the forgotten power of Lisp. Didier Verna. ACCU'08, April 2008.
75. Sémantique et localisation de l'assistance en réalité virtuelle. Didier Verna and Alain Grumbach. In *Journées Nationales du Groupe de Travail sur la Réalité Virtuelle*, pages 105–112, 1998.
76. Assistance cognitive à la télé-opération en monde virtuel. Alain Grumbach and Didier Verna. In *Journées Nationales du Groupe de Travail sur la Réalité Virtuelle*, pages 38–46, 1996.