



Mémoire en vue de l'obtention du diplôme
De Master.

- Filière -
Télécommunications

- Spécialité -
Système des Télécommunications et Réseaux

- Thème -

Revue des Outils de Simulation et des Métriques de Performance pour les Architectures de Villes Intelligentes

- Réalisé par -

**BOUDJAMMALA Abdessalam
SARI Mohammed Islam**

Membres du jury :

Prénom et NOM	Etablissement	Qualité
Mme. Akila SID	ENSTA	Président
M. Abdeldjallil MOUZAOU	CERIST	Promoteur
Mme. Seloua CHOUAF	ENSTA	Co-Promoteur
M. Tarek CHERIFI	ENSTA	Examineur
Mme. Imane CHIALI	ENSTA	Examineur

Revue des Outils de Simulation et des Métriques de Performance pour les Architectures de Villes Intelligentes

Mohammed Islam SARI¹, Abdessalam BOUDJAMMALA¹, Seloua CHOUAF¹, Abdeldjalil MOUZAOU²

¹Génie Electrique et Informatique Industrielle

Ecole Nationale Supérieure des Technologies Avancées, Bordj El Kiffane, Algérie

{m_sari, a_boudjammala, seloua.chouaf}@ensta.edu.dz

²Division Réseau

CEntre de Recherche sur l'Information Scientifique et Technologie, Ben Aknoun, Algérie

mabdeldjalil@cerist.dz

Abstract—La croissance rapide des zones urbaines et la complexité des infrastructures ont conduit à l'émergence du concept de ville intelligente. Afin de concevoir et de valider ces architectures avancées, l'introduction des outils de simulation ainsi que l'utilisation des métriques de performance sont devenues essentielles. Cette recherche propose une revue des outils de simulation les plus couramment utilisés dans ce domaine, ainsi qu'une analyse des métriques permettant d'évaluer la performance des solutions mises en œuvre.

Index Terms—Ville intelligente, outils de simulation, performance, validation, architecture urbaine

I. INTRODUCTION

L'urbanisation rapide à l'échelle mondiale impose aux villes de relever des défis majeurs exprimés en termes de gestion de la mobilité, des ressources, de l'énergie, de la sécurité et de l'environnement. Pour y faire face, le concept de *ville intelligente* (smart city) s'est imposé, en intégrant les technologies numériques au cœur de la planification et de la gestion urbaine.

Dans ce contexte, les outils de simulation jouent un rôle fondamental. Ils permettent de modéliser des systèmes complexes, de tester différents scénarios et d'évaluer les performances des solutions envisagées. Ces simulations fournissent des résultats très proches de la réalité, facilitant ainsi la prise de décision, tout en réduisant les coûts et les risques liés à l'expérimentation sur le terrain.

Cependant, une ville intelligente repose sur plusieurs piliers technologiques, parmi lesquels certains prennent une importance croissante : le *Big Data*, les réseaux de communication (*networking*), la *blockchain* et l'informatique en nuage (*cloud computing*). Ces technologies permettent de collecter, d'échanger, de sécuriser et de traiter d'immenses volumes de données, tout en assurant l'interopérabilité et la résilience des systèmes urbains[1][2].

Notre plan de revue s'articulera autour des points suivants:

- Une revue de la littérature sur le concept de ville intelligente, ses enjeux et ses caractéristiques.
- Une revue des outils de simulation utilisés dans le contexte des villes intelligentes.

- Une étude des métriques de validation permettant d'évaluer les performances des systèmes simulés.

II. REVUE DE LA LITTÉRATURE SUR LES VILLES INTELLIGENTES.

Le développement urbain au XXI^{ème} siècle a engendré un profond changement de paradigme. Face à l'urbanisation galopante et à la complexité croissante de la vie urbaine, la recherche sur les villes intelligentes est devenue une priorité stratégique, mobilisant chercheurs, acteurs industriels, décideurs politiques et collectivités locales. Bien que les technologies de l'information et de la communication aient connu une progression fulgurante, rendant possible l'émergence de villes dites intelligentes, ce concept reste encore en constante évolution et soulève de nombreux défis techniques, sociaux et environnementaux [3].

Selon les estimations des Nations Unies, la population mondiale passera de 7,2 milliards en 2015 à 9,7 milliards en 2050, soit une augmentation de 32 %, tandis que la population urbaine augmentera de 63 %, atteignant environ 6,3 milliards d'habitants. D'ici 2030, plus de 60 % des habitants de la planète vivront dans des zones urbaines, avec une croissance particulièrement marquée en Afrique, en Asie et en Amérique latine [4].

Cette urbanisation accélérée s'explique, d'une part, par la migration des populations rurales vers les centres urbains à la recherche de meilleures conditions de vie (emploi, éducation, soins de santé, accès à la culture, etc.) et, d'autre part, par les flux migratoires en provenance de pays touchés par la pauvreté ou des conflits sociaux et militaires, vers les pays industrialisés [4][5].

Dans ce contexte, il devient essentiel de repenser les modes de gestion des villes. Pour répondre aux enjeux tels que la surpopulation, la consommation énergétique, la gestion des ressources ou encore la protection de l'environnement, les villes doivent devenir plus intelligentes. Cela implique le recours à des approches innovantes et à des technologies

avancées permettant une gouvernance plus efficace, durable et résiliente des espaces urbains.

III. REVUE DES OUTILS UTILISÉS POUR LA SIMULATION DES VILLES INTELLIGENTES

Les villes intelligentes s'appuient sur un écosystème technologique dense et interconnecté, où l'expérimentation virtuelle (via simulation ou émulation) permet d'anticiper les comportements des systèmes avant leur déploiement réel.

Voici cinq piliers technologiques clés identifiés dans ce contexte :

1) **Émulateurs et simulateurs de réseaux**: Les simulateurs permettent de tester des architectures de communication complexes dans des environnements réalistes sans coûts matériels. Ils sont utilisés pour évaluer la qualité de service, analyser les performances des protocoles IoT, 5G, VANETs, etc., ou tester la résilience face aux cyberattaques [6].

Parmi les simulateurs et les émulateurs, on trouve :

- **Mininet** : est un outil d'émulation de réseau qui utilise la virtualisation légère pour simuler des réseaux complets sur une seule machine. Il permet de créer des réseaux virtuels réalistes en exécutant du code réel de noyau, de commutateur et d'application, ce qui facilite le développement, le test et le déploiement d'applications SDN [7][8].
 - **QualNet** : est un simulateur de réseaux haute performance, optimisé pour la simulation de réseaux filaires, sans fil et hybrides. Il offre une modélisation détaillée des protocoles, des couches physiques et des dispositifs réseau, permettant une analyse précise des performances. Grâce à sa technologie de simulation parallèle, QualNet assure une exécution rapide des scénarios complexes. Il est largement utilisé dans la recherche académique et industrielle pour la conception et l'évaluation de réseaux de communication [9].
 - **SUMO** : (Simulation of Urban MObility) est un simulateur de trafic open source, microscopique et multimodal, développé par le Centre aérospatial allemand (DLR). Il permet de modéliser et de simuler le comportement individuel des véhicules, des piétons et des transports publics dans des réseaux routiers complexes. Grâce à sa flexibilité et à sa précision, SUMO est largement utilisé dans la recherche sur la mobilité urbaine, l'évaluation des politiques de transport et le développement de systèmes de transport intelligents[10].
 - **COOJA** : est un simulateur de réseaux pour Contiki OS, permettant de modéliser et d'analyser des réseaux de capteurs sans fil et des applications IoT. Il offre une simulation multi-niveaux, allant de la modélisation abstraite à l'émulation précise du matériel. Grâce à son interface graphique, les utilisateurs peuvent configurer des topologies réseau et surveiller les communications entre nœuds. Cooja est un outil essentiel pour la recherche et le développement dans le domaine des réseaux embarqués et de l'IoT [11][12].
- **OMNeT++** : est un cadre de simulation à événements discrets, modulaire et orienté objet, développé en C++. Il est conçu pour la modélisation de réseaux de communication et de systèmes distribués. Son architecture permet la création de modèles à partir de composants réutilisables appelés modules. Des frameworks tels qu'INET étendent ses fonctionnalités en offrant des modèles pour divers protocoles et technologies réseau [13].
 - **NS-3** : est un simulateur de réseaux à événements discrets, open source, conçu pour la recherche et l'enseignement. Il permet la modélisation précise de protocoles et de systèmes de communication complexes. Grâce à son architecture modulaire et à sa prise en charge de divers protocoles, il offre une grande flexibilité pour la simulation de réseaux filaires et sans fil. NS-3 est largement adopté par la communauté scientifique pour l'évaluation des performances réseau [14].
 - **CORE** : (Common Open Research Emulator) est un émulateur de réseaux en temps réel et open-source, développé par le Naval Research Laboratory des États-Unis. Il permet de simuler des réseaux complexes en utilisant des conteneurs Linux (LXC) pour virtualiser les nœuds, offrant ainsi une émulation réaliste des protocoles réseau. CORE est particulièrement adapté pour tester des applications réseau, des protocoles de routage et des systèmes distribués dans des environnements contrôlés. Il est largement utilisé dans la recherche militaire et académique pour l'évaluation des performances réseau et la validation des configurations avant le déploiement réel [15].
 - **NetSim** : est un simulateur de réseaux à événements discrets développé par Tetcos, permettant la modélisation et l'analyse de divers types de réseaux. Il offre une interface graphique pour la création de scénarios réseau et prend en charge de nombreuses technologies, telles que TCP/IP, Wi-Fi, LTE et 6LoWPAN. NetSim permet également l'intégration avec des outils externes comme MATLAB et SUMO. Il est largement utilisé dans la recherche pour l'évaluation des performances réseau et la validation de protocoles [14].
 - **SimGrid** : est un cadre de simulation open source pour les applications distribuées, permettant de modéliser et d'analyser des systèmes complexes. Il prend en charge divers domaines tels que le cloud computing, le HPC et les systèmes P2P. Grâce à ses modèles précis et à ses interfaces en C++, Python et Java, il facilite le prototypage et l'évaluation des performances. SimGrid est largement utilisé dans la recherche pour la simulation de plateformes distribuées [16].

La table I propose une présentation comparative des principaux outils de simulation et d'émulation réseau.

TABLE I: Comparaison des outils de simulation et d'émulation réseau

Outil	Type	Domaine	Open Source	Language
SimGrid	Simulateur	Calcul distrib., Cloud, HPC	Oui	C++, Java, Py
QualNet	Simulateur	Mobile, Mil., IoT	Non	C++
NetSim	Simulateur	5G, IoT, Réseaux entreprise	Non	C/C++, .NET
CORE	Émulateur	Réseaux IP, SDN, Test temps réel	Oui	Python
NS-3	Simulateur	Protocoles, Recherche réseau	Oui	C++, Python
OMNeT++	Simulateur	Filaires, Sans fil, IoT	Oui	C++
COOJA	Simulateur	Capteurs, IoT	Oui	Java, C
SUMO	Simulateur	Trafic, Mobilité urbaine	Oui	C++, Python
Mininet	Émulateur	SDN, OpenFlow	Oui	Python
INET	Frame work	Simulation OMNeT++	Oui	C++

- **INET** : est un plateforme open source pour OMNeT++ permettant la simulation de réseaux filaires, sans fil et mobiles. Il offre des modèles détaillés pour les protocoles réseau, la mobilité et les applications. Sa modularité facilite la conception et l'évaluation de nouveaux protocoles et scénarios réseau. INET est largement utilisé dans la recherche pour simuler des architectures de communication complexes [17].

La figure 1 illustre, sous forme de représentation sectorielle, une répartition de l'utilisation des principaux outils de simulation et d'émulation réseau par la communauté scientifique et technique.

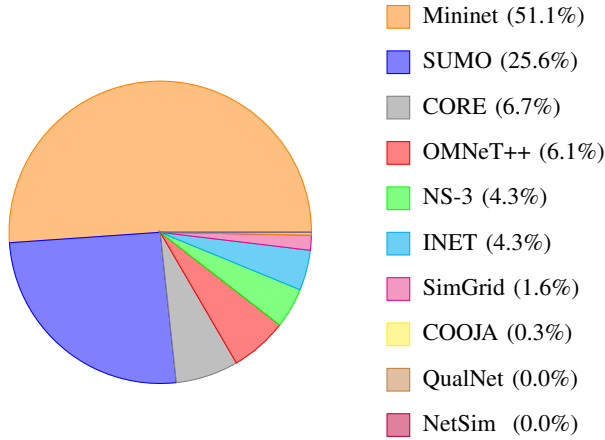


Fig. 1: Répartition de l'utilisation des outils de simulation et d'émulation réseau (basée sur le nombre d'étoiles Github)

2) Informatique en brouillard (Fog) et en nuage (Cloud):

Le Fog rapproche les ressources de calcul des capteurs IoT, réduisant la latence et la congestion, tandis que le Cloud offre élasticité et capacité de traitement massive. Des architectures hybrides ont été proposées pour combiner les avantages des deux paradigmes. [18] [19] [20].

- **iFogSim** : est un simulateur Java développé pour modéliser et simuler des environnements de calcul en brouillard (Fog) et en nuage (Cloud). Il permet d'évaluer

la performance des stratégies de placement de services, de gestion des ressources et de latence dans des architectures distribuées. iFogSim facilite la simulation de scénarios de villes intelligentes et d'IoT en reproduisant des topologies complexes et reste extensible pour divers cas d'étude [21].

- **Amazon EC2** :est un service d'infrastructure à la demande proposé par AWS, qui permet de créer, configurer et gérer des machines virtuelles dans le cloud. Il offre une grande scalabilité et flexibilité pour le déploiement d'applications, avec des options de tarification adaptées à différents besoins. EC2 est couramment utilisé pour héberger des services cloud dans les architectures IoT et villes intelligentes. Il prend en charge divers systèmes d'exploitation, types d'instances et zones géographiques [22].
- **AWS CloudWatch** :est un service de surveillance et d'observabilité proposé par Amazon Web Services, conçu pour collecter et suivre des métriques, des logs et des événements en temps réel. Il permet aux utilisateurs de visualiser les performances, de détecter les anomalies et d'automatiser les réponses aux changements dans leur infrastructure cloud. CloudWatch est essentiel pour assurer la résilience et l'efficacité des applications distribuées, notamment dans les environnements IoT et de villes intelligentes. Il s'intègre facilement avec d'autres services AWS comme EC2, Lambda ou S3 [23].
- **Google Cloud Platform (GCP)** :est une suite de services de cloud computing proposée par Google, offrant des ressources d'infrastructure, de stockage, d'analyse de données, d'IA et de machine learning. Elle permet aux développeurs et aux entreprises de déployer et de gérer des applications évolutives dans le cloud. GCP est largement utilisée pour les projets de villes intelligentes, notamment pour le traitement en temps réel des données IoT. Sa flexibilité et son intégration avec les outils open source en font une solution puissante dans les environnements distribués [24].

- **Cisco Fog Computing** :est une architecture développée par Cisco pour étendre les capacités du cloud jusqu'à la périphérie du réseau. Elle permet le traitement, le stockage et l'analyse des données près des sources (capteurs, dispositifs IoT), réduisant ainsi la latence et la charge du réseau. Ce paradigme est essentiel dans les systèmes temps réel, tels que les villes intelligentes, les transports intelligents et les soins de santé connectés. Cisco Fog Computing est à la base du consortium OpenFog, visant à standardiser cette approche [25].
- **Microsoft Azure IoT Edge** :est une plateforme qui étend les services de cloud computing d'Azure aux appareils en brouillard, permettant l'exécution locale de charges de travail telles que l'IA, l'analyse de données et la gestion d'appareils. Elle permet de traiter les données près de leur source, réduisant la latence et la dépendance au cloud. Azure IoT Edge est particulièrement adaptée aux scénarios de villes intelligentes nécessitant des décisions en temps réel. Elle s'intègre nativement avec l'écosystème Azure, facilitant la gestion à grande échelle [26].

La figure 2 illustre, sous forme de représentation sectorielle, la répartition de l'utilisation des principales plateformes et outils liés au cloud, au fog computing et à l'edge computing par la communauté scientifique et technique.

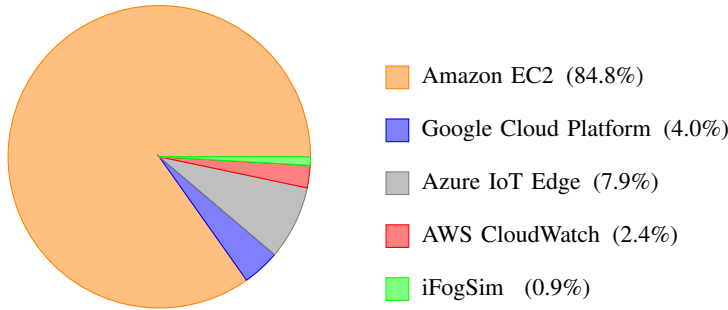


Fig. 2: Répartition de l'utilisation des plateformes cloud, fog et edge (basée sur le nombre d'étoiles GitHub)

La table 2 propose une vue comparative d'un ensemble d'outils liés à la simulation et à l'infrastructure du cloud et du fog computing.

3) **Plateformes blockchain**: La blockchain garantit la sécurité, la transparence et la traçabilité des échanges dans un environnement distribué. Elle est utilisée pour la protection des données, l'authentification, ou encore l'intégration de l'apprentissage fédéré pour la cybersécurité [27] [28] [29] [30].

– Plateformes multi-chaînes :

- **Polkadot** : est une plateforme blockchain multi-chaîne conçue pour permettre l'interopérabilité entre blockchains hétérogènes via une architecture partagée. Elle se compose d'une chaîne centrale (Relay Chain) qui coordonne des chaînes secondaires (Parachains) avec sécurité mutualisée. Polkadot favorise la scalabilité horizontale et la spécialisation fonctionnelle des chaînes. Ce modèle répond aux besoins des systèmes décentralisés complexes [31].
- **Cosmos** : est une plateforme blockchain conçue pour permettre l'interopérabilité entre chaînes indépendantes via un protocole standardisé appelé IBC (Inter-Blockchain Communication). Elle repose sur une architecture modulaire composée de hubs et de zones, coordonnée par le Cosmos Hub. En s'appuyant sur le moteur de consensus Tendermint, Cosmos offre scalabilité, souveraineté des chaînes et finalité rapide des transactions. Cette approche favorise l'émergence d'un écosystème décentralisé de blockchains interopérables [32].

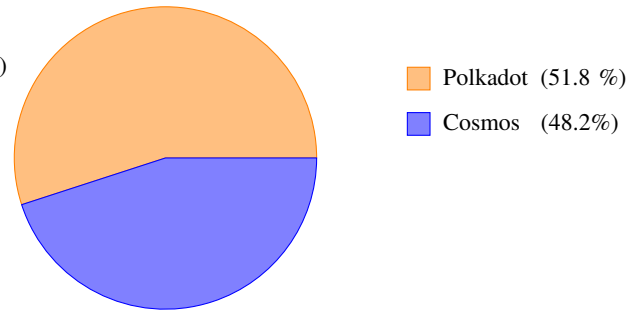


Fig. 3: Répartition de l'utilisation des plateformes Polkadot vs Cosmos (basée sur le nombre d'étoiles Github)

TABLE II: Comparaison des outils d'informatique en brouillard et en nuage

Outil	Type	Domaine	Open Source	Langage	Cloud
iFogSim	Simulateur	Simulation Fog	Oui	Java	Non
Amazon EC2	Service Cloud	Infrastructure as a Service (IaaS)	Non	Multi-API	Oui
AWS CloudWatch	Service Cloud	Supervision	Non	Multi-API	Oui
Google Cloud Platform (GCP)	Service Cloud	IaaS & PaaS	Non	Multi-API	Oui
Cisco Fog	Architecture (conceptuelle)	Fog Arch.	Non	–	Partiel
Azure IoT Edge	Émulateur / Runtime	Edge IoT	Partiel	.NET, C, Python	Oui

La figure 3 illustre, sous forme de représentation sectorielle, la répartition estimée de l'utilisation des principales plateformes et outils liés aux plateformes multi-chaînes par la communauté scientifique et technique.

– Plateformes de contrats intelligents :

- **Cardano** : est une plateforme blockchain de troisième génération conçue pour offrir une infrastructure sécurisée, évolutive et durable pour les contrats intelligents. Elle utilise un consensus Proof-of-Stake appelé Ouroboros, visant à optimiser la consommation énergétique tout en assurant une décentralisation robuste. Cardano se distingue par une approche académique rigoureuse, intégrant des méthodes formelles pour garantir la fiabilité et la sécurité des applications décentralisées. Cette plateforme supporte le développement d'applications complexes avec une architecture modulaire et interopérable [33].
- **Tezos** : est une plateforme blockchain auto-évolutive conçue pour supporter des contrats intelligents et des applications décentralisées avec un haut niveau de sécurité. Elle repose sur un mécanisme de consensus Proof-of-Stake et intègre un système de gouvernance on-chain permettant aux utilisateurs de proposer et voter des mises à jour du protocole. Tezos utilise le langage formel Michelson, qui favorise la vérification mathématique des contrats intelligents. Cette approche vise à améliorer la fiabilité des applications critiques dans des domaines comme la finance et l'industrie [34].
- **EOS** : est une plateforme blockchain conçue pour exécuter des contrats intelligents à grande échelle, avec un objectif de performance proche des systèmes centralisés. Elle utilise un consensus Delegated Proof-of-Stake (DPoS) permettant un débit élevé de transactions et une faible latence. EOS met l'accent sur la scalabilité, la gratuité des transactions pour les utilisateurs, et la flexibilité des mises à jour. Sa structure favorise le développement rapide d'applications décentralisées tout en assurant une gouvernance semi-centralisée via des producteurs de blocs élus [35].
- **NEO** : est une plateforme blockchain conçue pour le développement d'une économie intelligente, combinant actifs numériques, identité numérique et contrats intelligents. Elle repose sur le consensus Delegated Byzantine Fault Tolerance (dBFT), qui permet une finalité rapide des transactions et une gouvernance efficace. NEO se distingue par son support natif de plusieurs langages de programmation comme C#, Java et Python, facilitant l'accès aux développeurs. L'architecture de NEO vise à assurer l'interopérabilité, la conformité réglementaire et des performances adaptées aux usages à grande échelle [36].

La figure 4 illustre, sous forme de représentation sectorielle, la répartition estimée de l'utilisation des principales plateformes et outils liés aux plateformes de contrats intelligents par la communauté scientifique et technique.

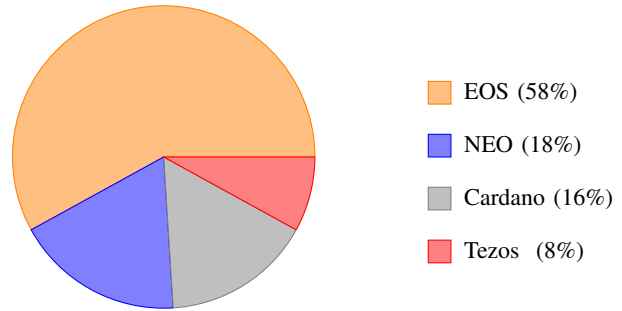


Fig. 4: Répartition de l'utilisation des principales plateformes de contrats intelligents (basée sur le nombre d'étoiles Github)

– Oracles et intégration de données :

- **Chainlink** : est un réseau décentralisé d'oracles permettant aux smart contracts d'accéder à des données externes de manière fiable et sécurisée. Les oracles agissent comme des ponts entre les blockchains et le monde réel, facilitant l'intégration de données hors-chaîne dans les contrats intelligents. Cette intégration de données est essentielle pour exécuter des logiques conditionnelles sur la blockchain, comme des paiements automatisés basés sur des événements du monde réel. Chainlink renforce cette intégration par un système de validation décentralisée, réduisant les risques de falsification des données [37].
- **IOTA** : est un protocole de registre distribué conçu spécifiquement pour l'Internet des objets (IoT), reposant sur une structure appelée Tangle plutôt qu'une blockchain classique. Contrairement aux blockchains traditionnelles, IOTA permet des transactions sans frais, une scalabilité élevée et une vérification parallèle des transactions. Sa conception vise à surmonter les limitations de la blockchain dans les environnements à ressources limitées, facilitant ainsi une intégration fluide des données IoT [38].

La figure 5 illustre, sous forme de représentation sectorielle, la répartition de l'utilisation des principales plateformes et outils liés aux oracles et à l'intégration de données par la communauté scientifique et technique.

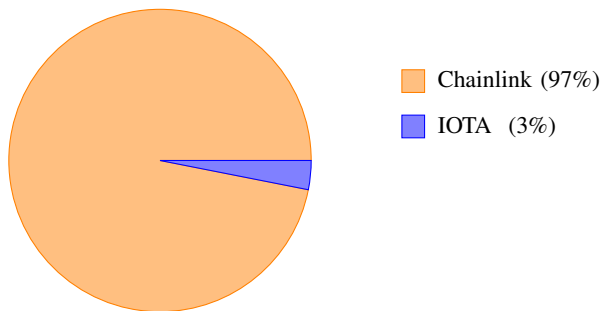


Fig. 5: Répartition de l'utilisation des plateformes et outils liés aux oracles et à l'intégration de données (basée sur le nombre d'étoiles Github)

– Blockchains d'entreprise et privées :

- **Hyperledger Sawtooth** : est une plateforme blockchain modulaire orientée entreprise, conçue pour la flexibilité et la scalabilité. Elle utilise le consensus Proof of Elapsed Time (PoET) pour optimiser la performance et réduire la consommation énergétique. L'étude met en avant son architecture évolutive et son intégration avec des outils de mesure comme Hyperledger Caliper. Elle convient aux cas d'usage nécessitant modularité, permissionnement et performances contrôlées[39].
- **Quorum** : est une blockchain d'entreprise dérivée d'Ethereum, conçue pour les environnements privés et réglementés. Elle offre des fonctionnalités de confidentialité des contrats et de transactions, avec des algorithmes de consensus tels que Raft et IBFT. L'évaluation expérimentale montre une bonne performance sous charges variées, adaptée aux besoins des institutions financières. Elle cible la transparence contrôlée et la conformité réglementaire[40].
- **R3 Corda** : est une plateforme DLT (Distributed Ledger Technology) orientée vers les entreprises, en particulier dans les secteurs financiers. Contrairement aux blockchains traditionnelles, Corda n'utilise pas de blocs mais un registre partagé et contrôlé, assurant la confidentialité par un échange point-à-point. Sa conception contractuelle permet des interactions légales automatisées entre parties de confiance[41].
- **Hyperledger Fabric** : est une plateforme blockchain permissionnée orientée entreprise, développée par la Linux Foundation. Fabric offre une architecture modulaire avec des mécanismes de consensus configurables, comme le Practical Byzantine Fault Tolerance (PBFT). Elle supporte divers langages de programmation pour les contrats intelligents (notamment Go et Java) et se distingue par sa capacité à garantir la confidentialité et

la sécurité nécessaires aux applications industrielles, financières et de santé[42].

La figure 6 illustre, sous forme de représentation sectorielle, la répartition estimée de l'utilisation des principales plateformes blockchain d'entreprise et privées par la communauté scientifique et technique.

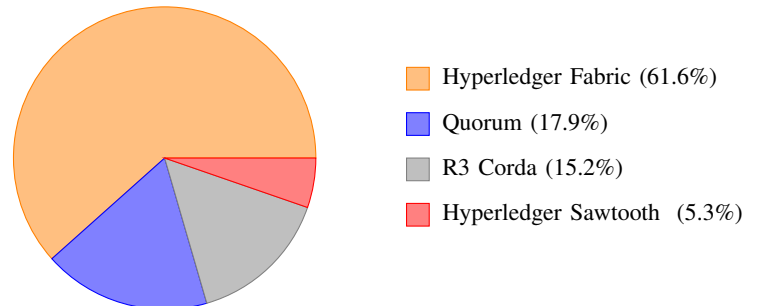


Fig. 6: Répartition de l'utilisation des plateformes blockchain d'entreprise et privées (basée sur le nombre d'étoiles Github)

– Plateformes de paiement et financières :

- **Stellar** : est une plate-forme blockchain conçue pour les paiements internationaux rapides, sécurisés et à faible coût. Elle repose sur le Stellar Consensus Protocol (SCP), un algorithme fédéré efficace sans preuve de travail. Stellar permet l'émission et l'échange d'actifs numériques interopérables. Elle cible l'inclusion financière, notamment via les corridors transfrontaliers[43].
- **Elrond** : est une blockchain performante orientée vers la finance décentralisée (DeFi), offrant des transactions rapides grâce à son protocole de sharding adaptatif. Elle permet la création d'applications décentralisées (dApps) et de plateformes de paiement comme les DEX (échanges décentralisés), tout en maintenant des coûts bas et une grande scalabilité[44].

La figure 7 illustre, sous forme de représentation sectorielle, la répartition estimée de l'utilisation des principales plateformes et outils liés aux plateformes de paiement et financières par la communauté scientifique et technique.

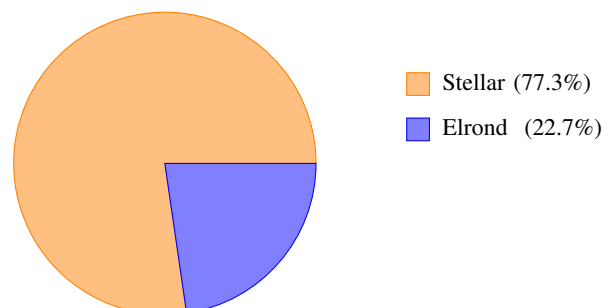


Fig. 7: Répartition de l'utilisation des plateformes de paiement et financières (basée sur le nombre d'étoiles Github)

– Chaîne d’approvisionnement et logistique :

- **VeChain** : est une plateforme blockchain conçue pour améliorer la gestion de la chaîne d’approvisionnement en offrant une transparence, une traçabilité et une efficacité accrues. Fondée en 2015, VeChain utilise une architecture à double jeton (VET et VTHO) et repose sur un mécanisme de consensus par preuve d’autorité (PoA). Elle permet aux entreprises de suivre les produits tout au long de leur cycle de vie, de la production à la livraison, en intégrant des technologies telles que les capteurs IoT et les puces RFID. Cette approche renforce la confiance entre les parties prenantes et réduit les risques liés à la contrefaçon et aux erreurs logistiques [45].

– Plateformes haute performance et évolutives :

- **Avalanche** : est une plateforme blockchain haute performance conçue pour offrir une scalabilité optimale grâce à son consensus innovant basé sur un mécanisme probabiliste et l’utilisation de sous-réseaux (subnets). Cette architecture permet des transactions rapides, une haute évolutivité et une flexibilité dans la création d’applications décentralisées (DeFi, NFT, entreprise). Avalanche est également compatible avec l’Ethereum Virtual Machine (EVM), facilitant ainsi le déploiement d’applications existantes basées sur Ethereum[46].
- **Zilliqa** : est une blockchain publique utilisant le sharding pour permettre une évolutivité linéaire en augmentant le nombre de transactions traitées en parallèle. Zilliqa intègre un consensus hybride combinant une preuve de travail initiale avec une tolérance pratique aux pannes byzantines (pBFT). Elle propose Scilla, un langage de contrats intelligents spécifiquement conçu pour la sécurité et la vérification formelle, réduisant les risques liés aux vulnérabilités fréquentes des contrats intelligents[47].

La figure 8 illustre sous forme de représentation sectorielle la répartition de l’utilisation des principales plateformes et outils liés aux plateformes de haute performance et évolutives par la communauté scientifique et technique.

– Autres plateformes généralistes :

- **Ethereum** : est une plateforme blockchain généraliste open-source permettant le développement de contrats intelligents et d’applications décentralisées (dApps). Ethereum utilise la machine virtuelle Ethereum (EVM) pour exécuter des contrats principalement écrits en Solidity. Suite à sa transition récente vers un consensus Proof-of-Stake (PoS), Ethereum a considérablement réduit sa consommation énergétique, renforçant son attractivité pour les applications écologiques et économiques. Ethereum est largement adopté dans les domaines financiers, des NFT et des applications blockchain variées[48].

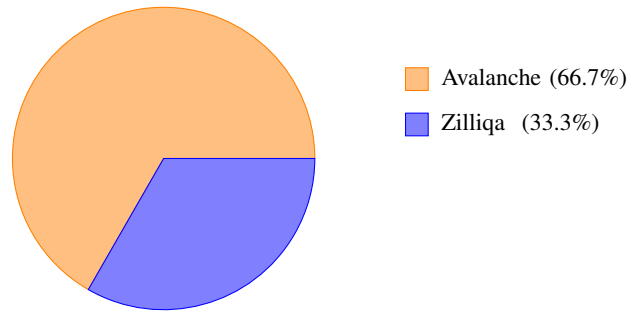


Fig. 8: Répartition de l’utilisation des principales plateformes haute performance et évolutives (basée sur le nombre d’étoiles Github)

La table III propose une vue comparative des principaux frameworks et plateformes blockchain.

4) **Plateformes Big Data**: Les plateformes Big Data permettent l’analyse de grandes quantités de données urbaines. Elles sont essentielles pour la prise de décision en temps réel, la détection d’anomalies ou la planification urbaine [49] [50].

- **Apache Hadoop** : est un framework open-source conçu pour le stockage et le traitement distribué de grands ensembles de données sur des clusters de machines. Il repose sur le système de fichiers HDFS et le modèle de programmation MapReduce, permettant une scalabilité horizontale et une tolérance aux pannes. Hadoop est largement utilisé pour les applications nécessitant le traitement de volumes massifs de données structurées et non structurées [51].
- **Apache Spark** : est un moteur de traitement de données rapide et généraliste, capable de gérer des charges de travail en batch et en streaming. Il offre des API dans plusieurs langages (Java, Scala, Python, R) et prend en charge des modules pour le SQL, le machine learning et le traitement de graphes. Spark est reconnu pour ses performances élevées grâce au traitement en mémoire [52].
- **Apache Flink** : est une plateforme de traitement de flux de données distribuée, conçue pour des applications nécessitant un traitement en temps réel avec une faible latence. Flink offre des garanties de traitement exactement une fois (exactly-once) et supporte le traitement basé sur le temps d’événement, ce qui le rend adapté aux applications critiques [53].
- **MapReduce** : est un modèle de programmation pour le traitement parallèle de grands ensembles de données. Il divise les tâches en deux phases : la phase Map, qui traite et filtre les données, et la phase Reduce, qui agrège les résultats. MapReduce est la base du traitement de données dans Hadoop [54].

TABLE III: Comparaison des plateformes Blockchain étudiées selon leur type et caractéristiques techniques

Outil	Type	Domaine	Open	Langage
Plateformes multi-chaîne				
Polkadot	Blockchain	Interopérabilité, Parachains	Oui	Rust, Substrate
Cosmos	Blockchain	Écosystème interopérable	Oui	Go, Cosmos SDK
Plateformes de contrats intelligents				
Cardano	Blockchain	Contrats intelligents, DeFi	Oui	Haskell, Plutus
Tezos	Blockchain	Contrats intelligents, Finance	Oui	OCaml, Michelson
EOS	Blockchain	dApps à grande échelle	Oui	C++, WebAssembly
NEO	Blockchain	Économie intelligente	Oui	C#, Java, Python
Oracles et intégration de données				
Chainlink	Oracle décentralisé	Intégration de données hors-chaîne	Oui	Solidity, Go
Band Protocol	Oracle décentralisé	Données temps réel, API	Oui	Rust, Go
Blockchains d'entreprise et privées				
Hyperledger Fabric	Blockchain permissionnée	Réseaux privés entreprise	Oui	Go, Java
Quorum	Blockchain permissionnée	Finance, entreprise	Oui	Solidity, Go
R3 Corda	Blockchain permissionnée	Finance, contrats privés	Partiel	Kotlin, Java
Hyperledger Sawtooth	Blockchain permissionnée	Industrie, traçabilité	Oui	Python, Go, Rust
Plateformes de paiement et financières				
Ripple	Blockchain	Paiements transfrontaliers	Non	Java, C++
Stellar	Blockchain	Paiements, transfert de valeur	Oui	C++, Go
Chaîne d'approvisionnement et logistique				
VeChain	Blockchain	Logistique, suivi IoT	Non	Solidity, VET/VTTHO APIs
Plateformes haute performance et évolutives				
Avalanche	Blockchain	DeFi, NFT, Subnets	Oui	Solidity (EVM compatible)
Zilliqa	Blockchain	Transactions haut débit	Oui	Scilla
Plateformes généralistes				
Ethereum	Blockchain	dApps, DeFi, NFT	Oui	Solidity

La table IV propose une vue comparative des principaux outils et plateformes Big Data.

TABLE IV: Comparaison des plateformes Big Data selon leurs caractéristiques techniques

Outil	Type	Domaine	Open Source	Language
Apache Hadoop	Framework Big Data	Stockage, traitement batch distribué	Oui	Java
Apache Spark	Moteur de traitement	Traitement batch et streaming, in-memory	Oui	Scala, Java, Python, R
Apache Flink	Moteur de flux distribué	Streaming temps réel, faible latence	Oui	Java, Scala
MapReduce	Modèle de programmation	Traitement batch distribué (via Hadoop)	Oui	Java

La figure 9 illustre, sous forme de représentation sectorielle, la répartition estimée de l'utilisation des principales plateformes et outils liés aux plateformes Big Data par la communauté scientifique et technique.

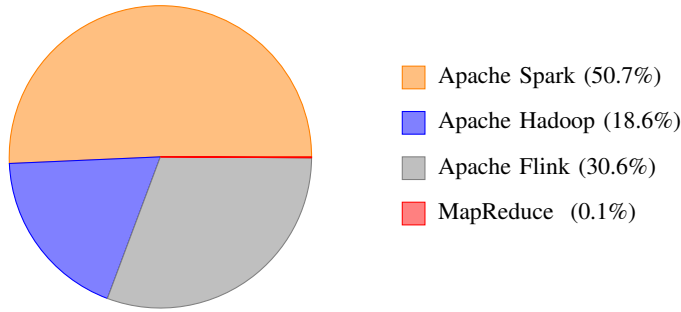


Fig. 9: Répartition de l'utilisation des principales plateformes Big Data (basée sur le nombre d'étoiles Github)

5) **Réseaux à contrôle logiciel (SDN)**: Le SDN permet de contrôler dynamiquement les flux de données dans les infrastructures urbaines. Il joue un rôle crucial dans la gestion de la congestion, la résilience du réseau, et la protection contre les attaques [55] [56] [57] [58].

- **OpenDaylight** : est une plateforme SDN open-source développée pour fournir un contrôle centralisé et programmable des réseaux. Elle supporte divers protocoles comme OpenFlow et NETCONF, et offre une architecture modulaire permettant l'intégration de multiples services réseau [59].
- **ONOS (Open Network Operating System)** : est un système d'exploitation réseau open-source conçu pour les fournisseurs de services. Il offre une haute disponibilité, une scalabilité horizontale et une gestion efficace des ressources réseau, ce qui le rend adapté aux environnements de production à grande échelle [60].
- **Ryu** : est un framework SDN open-source écrit en Python, offrant une API simple pour le développement d'applications réseau. Il supporte plusieurs versions du protocole OpenFlow et est utilisé pour des applications nécessitant une personnalisation rapide et une intégration facile [61].

- **Floodlight** : est un contrôleur SDN open-source basé sur Java, conçu pour être simple à utiliser et extensible. Il supporte OpenFlow et est souvent utilisé dans des environnements de recherche et de développement pour tester des applications SDN [62].

La figure 10 illustre, sous forme de représentation sectorielle, la répartition de l'utilisation des principales plateformes et outils liés aux plateformes Réseaux à contrôle logiciel (SDN) [63].

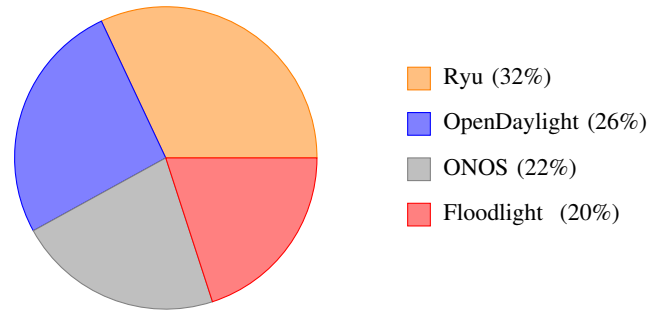


Fig. 10: Répartition de l'utilisation des principaux contrôleurs SDN [63]

La table V propose une vue comparative des principaux contrôleurs et outils SDN.

NB: Il est essentiel de noter que le nombre d'étoiles attribué à un projet de recherche sur GitHub constitue un indicateur clé de sa popularité et de son adoption au sein de la communauté des développeurs et scientifiques. En comparant plusieurs outils de simulation à travers cette métrique, nous avons pu identifier les tendances et préférences du marché, fournissant ainsi des données précieuses pour le tracé des représentations sectorielles présentées dans cette section. Cependant, bien que ce critère reflète l'intérêt porté à chaque outil, il ne traduit pas nécessairement sa performance ni son utilisation effective. Pour affiner l'analyse, cette donnée doit être corrélée avec d'autres paramètres afin d'améliorer la précision des estimations quantitatives et d'obtenir des résultats plus représentatifs de la réalité.

TABLE V: Comparaison des contrôleurs SDN selon leur type et leurs caractéristiques techniques

Outil	Type	Domaine	Open	Language
OpenDaylight	Contrôleur SDN	Contrôle centralisé, intégration protocoles (OpenFlow, NETCONF)	Oui	Java
ONOS	OS réseau SDN	Scalabilité, haute disponibilité, opérateurs télécom	Oui	Java
Ryu	Framework SDN	Dév. rapide, applications personnalisées, OpenFlow	Oui	Python
Floodlight	Contrôleur SDN	Recherche, prototypage, extensibilité	Oui	Java

- Critères de Choix

Le choix d'un outil de simulation adapté à une architecture de ville intelligente repose sur plusieurs critères essentiels qui garantissent son efficacité, sa pérennité et son intégration dans des projets complexes.

- **La flexibilité** permet au simulateur de modéliser une variété de scénarios. Par exemple, dans le travail de [64], la plateforme est adaptée à différents contextes urbains et à des flux de données hétérogènes.
- **L'extensibilité** est également cruciale. L'étude [65] montre comment un environnement peut être enrichi par des modules de détection d'anomalies intégrés à des réseaux sécurisés.
- **L'interopérabilité** permet aux outils de simulation de se connecter à d'autres systèmes. Dans [66], l'approche met en avant des enjeux de connectivité entre services urbains.
- **La scalabilité** est mise en évidence dans plusieurs études comme celle de [67], qui évalue explicitement la latence et la performance à grande échelle.
- **L'ouverture du code et le modèle économique** (open source et gratuité) influencent la flexibilité, le coût de déploiement et l'adoption à long terme. La disponibilité sous licence libre (MIT, Apache, GPL) est un atout, tout comme l'accès gratuit ou freemium aux outils utilisés [68].
- **La documentation et le support technique** incluent la richesse des guides, tutoriels et FAQ disponibles, ainsi que la réactivité sur les canaux communautaires (GitHub, forums). Ces éléments sont essentiels pour assurer la maintenabilité et la montée en compétence rapide des utilisateurs [67].
- **La communauté d'utilisateurs et de développeurs** se mesure à travers le nombre de contributeurs actifs, la fréquence des mises à jour et le suivi des problèmes. Un projet soutenu par une communauté dynamique est souvent plus pérenne et fiable [69].
- **La compatibilité et le respect des standards** désignent la capacité d'un outil à fonctionner sur différentes plateformes (Windows, Linux, macOS, Docker) et à s'interfacer facilement grâce à des formats ouverts (JSON, XML, OpenAPI, etc.) [70].

IV. MÉTRIQUES DE PERFORMANCE

Pour s'assurer qu'un système répond bien aux attentes, il est essentiel de le jauger à l'aide de quelques indicateurs clés. Ces métriques nous permettent de mesurer, de manière concrète et fiable, la rapidité, la capacité de traitement, la consommation et la robustesse de nos services. En surveillant ces chiffres, on peut à la fois repérer rapidement les points faibles et valider les améliorations apportées. Vous trouverez ci-dessous les principales définitions utilisées pour évaluer et optimiser les performances de vos applications.

- **Latence / Délai** Intervalle de temps écoulé entre l'émission d'une requête et la réception de la réponse finale. Il englobe le temps de propagation, de traitement et de transmission, et se mesure généralement en millisecondes. Une latence faible est cruciale pour les applications en temps réel et les services critiques où chaque milliseconde compte [26].
- **Débit (Throughput)** Volume de données ou nombre d'événements traités par unité de temps (p.ex., Mbps pour le trafic réseau ou transactions/s pour les requêtes applicatives).
Il reflète la capacité du système à supporter des charges importantes et à maintenir un flux continu sans goulot d'étranglement [71].
- **Scalabilité** Capacité du système à conserver ses performances (latence et débit) lorsque la charge augmente, que ce soit en ajoutant des nœuds, en augmentant le nombre d'utilisateurs ou le volume de données.
Elle se traduit par des variations minimales des indicateurs clés malgré la montée en charge [25].
- **Précision / Exactitude (Accuracy)** Proportion de prédictions ou de classifications correctes par rapport au total des décisions émises par un modèle ou un algorithme.
Calcul : $(\text{vrais positifs} + \text{vrais négatifs}) \div (\text{total des cas})$.
Utile pour évaluer la fiabilité des systèmes de détection d'anomalies ou de reconnaissance d'images [72].
- **Consommation énergétique** Quantité d'énergie dépensée pour exécuter un traitement ou maintenir un service, mesurée en joules ou en wattheures.
Elle inclut la consommation des CPU, GPU, dispositifs IoT et équipements réseau.
Optimiser cette métrique est essentiel pour réduire les coûts opérationnels et l'empreinte environnementale [73].
- **Utilisation CPU / Ressources** Pourcentage d'occupation du processeur, de la mémoire ou d'autres ressources (I/O, stockage) par l'application ou le service. Permet de détecter les goulots d'étranglement, d'ajuster le dimensionnement de l'infrastructure et de planifier la montée en charge [74].
- **Surcharge mémoire / Stockage** Espace additionnel requis pour le fonctionnement ou la simulation (caches, logs, snapshots, tables de hachage). Exprimée en méga- ou gigaoctets, cette métrique aide à anticiper les besoins en capacité et à éviter les saturations [75].
- **Fiabilité** probabilité que le service reste opérationnel et sans erreur sur une période donnée (taux de disponibilité).
- **Overhead** : coût supplémentaire (temps de calcul,

mémoire, bande passante) induit par les mécanismes de sécurité, de réplication ou de chiffrement.

L'équilibre entre fiabilité et overhead caractérise la robustesse du système [72].

- **Taux de détection** Pour les systèmes de sécurité ou de surveillance, proportion des incidents réels détectés correctement (vrais positifs) par rapport au total des incidents.

Associé aux taux de faux positifs et faux négatifs, il se combine souvent dans un F_1 -score pour un bilan global [72].

- **Temps d'exécution** Durée totale nécessaire pour accomplir un traitement, une transaction ou un job complet, mesurée du lancement à l'achèvement. Utile pour comparer l'efficacité de différentes implémentations ou configurations et optimiser les workflows [76].

- **Temps d'entraînement** Temps requis pour entraîner un modèle de machine learning ou deep learning sur un ensemble de données.

Ce délai dépend de la complexité du modèle, de la taille des données et des ressources matérielles disponibles [77].

- **Coût de transaction / Frais** Montant monétaire ou quantité de "gas" facturée pour chaque opération ou transaction dans les architectures blockchain ou distribuées. Il impacte directement la viabilité économique des applications décentralisées [78].

- **Bande passante** : Capacité maximale de transfert de données d'une liaison ou d'un segment réseau, exprimée en Mbps ou Gbps.

Elle conditionne la qualité de service, notamment pour le streaming, la vidéoconférence ou la transmission de gros volumes de données [25].

- **Taux de hachage** : nombre d'opérations de hachage réalisées par seconde par le réseau ou un mineur.

- **Difficulté** : paramètre ajusté périodiquement pour maintenir un temps de génération de bloc constant.

Ces deux derniers indicateurs évaluent la sécurité et la performance des blockchains proof-of-work [78].

V. CONCLUSION

L'essor des zones urbaines et la complexité croissante de leurs infrastructures imposent aujourd'hui de recourir à des approches virtuelles pour concevoir et valider des architectures de villes intelligentes. Notre revue a montré que la simulation s'impose comme un vecteur indispensable de réduction des coûts et des risques, en offrant la possibilité de tester des scénarios variés avant leur déploiement réel.

Nous avons passé en revue un riche écosystème d'outils — de Mininet à SimGrid en passant par iFogSim et les plateformes cloud natives — qui, chacun à sa manière, permet de modéliser les réseaux de communication, les flux de mobilité, les ressources de calcul et les interactions IoT. Ce panorama met en lumière la nécessité d'adapter le choix de l'outil au contexte d'étude, selon des critères de flexibilité, d'extensibilité, d'interopérabilité et de scalabilité.

L'analyse des métriques de performance (latence, débit, consommation énergétique, fiabilité, etc.) a, quant à elle, permis de dégager un cadre d'évaluation unifié, facilitant la comparaison des architectures et l'optimisation des paramètres clés. Ces indicateurs constituent le socle de toute démarche de validation rigoureuse et permettent de garantir la robustesse et la durabilité des solutions.

Enfin, pour poursuivre le développement de villes toujours plus résilientes et efficaces, plusieurs pistes de recherche se dessinent :

- Explorer la synergie inter-outils pour des simulations multi-domaines (réseaux, mobilité, énergie).
- Affiner ou proposer de nouvelles métriques adaptées aux nouveaux usages (IA embarquée, blockchain).
- Mettre en place des bancs d'essai hybrides combinant simulation et prototype edge/fog.
- Développer des référentiels ouverts favorisant la reproductibilité et la comparabilité des études.

Ces perspectives ouvrent la voie à une ingénierie urbaine plus agile, prête à relever les défis futurs des villes intelligentes.

REFERENCES

- [1] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, 2014.
- [2] M. Batty, K. W. Axhausen, F. Giannotti, *et al.*, "Smart cities of the future," *The European Physical Journal Special Topics*, vol. 214, no. 1, pp. 481–518, 2012.
- [3] M. Matyja, "Smart city as a concept of intelligent urban development," *Studia Ekonomiczne. Zeszyty Naukowe Uniwersytetu Ekonomicznego w Katowicach*, no. 330, pp. 75–84, 2017. [Online]. Available: <https://bazekon.uek.krakow.pl/en/rekord/171505527>.

- [4] M. Eremia, L. Toma, and M. Sanduleac, "The smart city concept in the 21st century," *Procedia Engineering*, vol. 181, pp. 12–19, 2017. DOI: 10.1016/j.proeng.2017.02.357.
- [5] D. Stawasz and D. Sikora-Fernandez, "The concept of smart city in the theory and practice of urban development," *Financial Internet Quarterly "e-Finanse"*, vol. 11, no. 4, pp. 117–132, 2015. [Online]. Available: https://doi.org/10.14636/1734-039X_11_4_009.
- [6] N. Moustafa, "A new distributed architecture for evaluating ai-based security systems at the edge: Network TON_IoT datasets," *Sustainable Cities and Society*, vol. 72, p. 103041, 2021, ISSN: 2210-6707. DOI: 10.1016/j.scs.2021.103041. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2210670721002808>.
- [7] M. Project, *Mininet: An instant virtual network on your laptop*, <https://mininet.org>, Accessed: 2025-05-17, 2024.
- [8] D. V. Khoa and T. N. N. Khanh, "Emulation of software-defined network using mininet," *Dalat University Journal of Science*, vol. 11, no. 1, pp. 80–92, 2021. DOI: 10.37569/DalatUniversity.11.1.657(2021). [Online]. Available: <https://tckh.dlu.edu.vn/index.php/tckhdhdl/article/view/657>.
- [9] S. Munot, R. C. Jaiswal, H. Bhoskar, *et al.*, "Wi-fi inter-network simulation and performance analysis using qualnet," in *Smart Trends in Computing and Communications (SmartCom 2024)*, ser. Lecture Notes in Networks and Systems, vol. 948, Springer, Singapore, 2024, pp. 187–198. DOI: 10.1007/978-981-97-1329-5_15. [Online]. Available: https://doi.org/10.1007/978-981-97-1329-5_15.
- [10] D. A. Guastella, E. Montero-Porras, A. Morales-Hernández, and G. Bontempi, "Traffic modeling with sumo: A tutorial," *arXiv preprint arXiv:2304.05982*, 2023. [Online]. Available: <https://arxiv.org/abs/2304.05982>.
- [11] A. Shahrokhi and M. Ahmadi, "Power evaluation of iot application layer protocols," *arXiv preprint arXiv:2405.07326*, 2024. [Online]. Available: <https://arxiv.org/abs/2405.07326>.
- [12] V. Rege, "Design and implementation of an antenna model for the cooja simulator," *arXiv preprint arXiv:1610.06129*, 2016. [Online]. Available: <https://arxiv.org/abs/1610.06129>.
- [13] A. Varga, "A practical introduction to the omnet++ simulation framework," in *Recent Advances in Network Simulation*, ser. EAI/Springer Innovations in Communication and Computing, A. Virdis and M. Kirsche, Eds., Springer, Cham, 2019, pp. 3–51. DOI: 10.1007/978-3-030-12842-5_1. [Online]. Available: https://doi.org/10.1007/978-3-030-12842-5_1.
- [14] F. Raimondi and M. Pagano, "Computer network simulation with ns-3: A systematic literature review," *Electronics*, vol. 9, no. 2, p. 272, 2020. DOI: 10.3390/electronics9020272. [Online]. Available: <https://doi.org/10.3390/electronics9020272>.
- [15] J. Ahrenholz, C. Danilov, T. R. Henderson, and J. H. Kim, "Core: A real-time network emulator," in *2008 IEEE Military Communications Conference*, IEEE, 2008, pp. 1–7. DOI: 10.1109/MILCOM.2008.4754076. [Online]. Available: <https://doi.org/10.1109/MILCOM.2008.4754076>.
- [16] H. Casanova, A. Giersch, A. Legrand, M. Quinson, and F. Suter, "Versatile, scalable, and accurate simulation of distributed applications and platforms," *Journal of Parallel and Distributed Computing*, vol. 74, no. 10, pp. 2899–2917, 2014. DOI: 10.1016/j.jpdc.2014.06.008. [Online]. Available: <https://doi.org/10.1016/j.jpdc.2014.06.008>.
- [17] L. Mészáros, A. Varga, and M. Kirsche, "Inet framework," in *Recent Advances in Network Simulation*, ser. EAI/Springer Innovations in Communication and Computing, Springer, 2019, pp. 55–106. DOI: 10.1007/978-3-030-12842-5_2. [Online]. Available: https://doi.org/10.1007/978-3-030-12842-5_2.
- [18] "Focan: A fog-supported smart city network architecture for management of applications in the internet of everything environments," *Journal of Parallel and Distributed Computing*, 2020.
- [19] "Computational viability of fog methodologies in iot enabled smart city architectures-a smart grid case study," *EAI Endorsed Transactions on Smart Cities*, 2020.
- [20] "Dewbcity: Blockchain network-based dew-cloud modeling for distributed and decentralized smart cities," *The Journal of Supercomputing*, 2021.
- [21] H. Gupta, A. V. Dastjerdi, S. K. Ghosh, and R. Buyya, "Ifogsim: A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments," *Software: Practice and Experience*, vol. 47, no. 9, pp. 1275–1296, 2017. DOI: 10.1002/spe.2509.
- [22] M. Armbrust, A. Fox, R. Griffith, *et al.*, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010. DOI: 10.1145/1721654.1721672.
- [23] U. V. Chavan, "A study of aws cloudwatch challenges and its emerging trends on latest technologies," *International Journal of Scientific Research in Engineering and Management (IJSREM)*, vol. 7, no. 6, pp. 1–5, 2023. [Online]. Available: <https://ijsrem.com/download/a-study-of-aws-cloudwatch-challenges-and-its-emerging-trends-on-latest-technologies/>.
- [24] S. Challita, F. Zalila, C. Gourdin, and P. Merle, "A precise model for google cloud platform," in *2018 IEEE International Conference on Cloud Engineering (IC2E)*, 2018, pp. 177–183. DOI: 10.1109/IC2E.2018.00041.
- [25] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing: A platform for internet of things and analytics," *Big data and internet of things: A roadmap for*

- smart environments*, pp. 169–186, 2014. DOI: 10.1007/978-3-319-05029-4_7.
- [26] O. Ali and M. K. Ishak, “Bringing intelligence to iot edge: Machine learning based smart city image classification using microsoft azure iot and custom vision,” *Journal of Physics: Conference Series*, vol. 1529, no. 4, p. 042076, 2020. DOI: 10.1088/1742-6596/1529/4/042076.
 - [27] “Blockchain-empowered cloud architecture based on secret sharing for smart city,” *Journal of Information Security and Applications*, 2021.
 - [28] “Trusted data infrastructure for smart cities: A blockchain perspective,” *Building Research & Information*, 2021.
 - [29] “Securing critical iot infrastructures with blockchain-supported federated learning,” *IEEE Internet of Things Journal*, vol. 9, no. 4, 2022.
 - [30] “Blockchain based hybrid network architecture for the smart city,” *Future Generation Computer Systems*, 2022.
 - [31] J. Burdges, A. Cevallos, P. Czaban, *et al.*, “Overview of polkadot and its design considerations,” *arXiv preprint arXiv:2005.13456*, 2020.
 - [32] J. Kwon and E. Buchman, “Cosmos whitepaper,” *A Network of Distributed Ledgers*, vol. 27, pp. 1–32, 2019.
 - [33] S. Barac, I. Botički, G. Perković, V. Radošević, and I. Terzić, “Cardano-what is it and how to start working with it,” in *2023 46th MIPRO ICT and Electronics Convention (MIPRO)*, IEEE, 2023, pp. 1727–1732.
 - [34] V. Allombert, M. Bourgoïn, and J. Tesson, “Introduction to the tezos blockchain,” in *2019 International Conference on High Performance Computing & Simulation (HPCS)*, IEEE, 2019, pp. 1–10.
 - [35] W. Song, W. Zhang, L. Zhai, *et al.*, “Eos. io blockchain data analysis,” *The Journal of Supercomputing*, pp. 1–32, 2022.
 - [36] E. Elrom and E. Elrom, “Neo blockchain and smart contracts,” *The Blockchain Developer: A Practical Guide for Designing, Implementing, Publishing, Testing, and Securing Distributed Blockchain-based Projects*, pp. 257–298, 2019.
 - [37] S. K. Ezzat, Y. N. Saleh, and A. A. Abdel-Hamid, “Blockchain oracles: State-of-the-art and research directions,” *IEEE Access*, vol. 10, pp. 67 551–67 572, 2022.
 - [38] L. T. Khrais, “Comparison study of blockchain technology and iota technology,” in *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, IEEE, 2020, pp. 42–47.
 - [39] B. Ampel, M. Patton, and H. Chen, “Performance modeling of hyperledger sawtooth blockchain,” in *2019 IEEE International Conference on Intelligence and Security Informatics (ISI)*, 2019, pp. 59–61. DOI: 10.1109/ISI.2019.8823238.
 - [40] A. Baliga, I. Subhod, P. Kamat, and S. Chatterjee, “Performance evaluation of the quorum blockchain platform,” *arXiv preprint arXiv:1809.03421*, 2018.
 - [41] J. Polge, J. Robert, and Y. Le Traon, “Permissioned blockchain frameworks in the industry: A comparison,” *Ict Express*, vol. 7, no. 2, pp. 229–233, 2021.
 - [42] E. Androulaki, A. Barger, V. Bortnikov, *et al.*, “Hyperledger fabric: A distributed operating system for permissioned blockchains,” in *Proceedings of the Thirteenth EuroSys Conference*, 2018, pp. 1–15.
 - [43] M. Lokhava, G. Losa, D. Mazières, *et al.*, “Fast and secure global payments with stellar,” in *Proceedings of the 27th ACM Symposium on Operating Systems Principles*, 2019, pp. 80–96.
 - [44] S. González Rubio, “Testdex: Un dex simple en la xarxa elrond,” 2022.
 - [45] Z. She, “Vechain: A renovation of supply chain management,” in *Proceedings of the International Conference on Education, Information Engineering and Intelligent Systems (ICEIEIS)*, Francis Academic Press, 2022, pp. –. [Online]. Available: https://webofproceedings.org/proceedings_series/ESSP/ICEIEIS%202022/G68.pdf.
 - [46] T. Rocket, M. Yin, K. Sekniqi, R. van Renesse, and E. G. Sirer, “Scalable and probabilistic leaderless bft consensus through metastability,” *arXiv preprint arXiv:1906.08936*, 2019.
 - [47] X. Wang, B. C. Ooi, K.-L. Tan, *et al.*, “Zilliqa: A scalable and secure blockchain platform,” in *Proceedings of the 2019 ACM SIGMOD International Conference on Management of Data*, ACM, 2019, pp. 1393–1405.
 - [48] G. Wood, “Ethereum: A secure decentralised generalised transaction ledger,” *Ethereum project yellow paper*, vol. 151, pp. 1–32, 2014.
 - [49] “Urban planning and smart city decision management empowered by real-time data processing using big data analytics,” *Sensors*, 2020.
 - [50] “Privacy-preserving anomaly detection in the cloud for smart cities,” *Journal of Parallel and Distributed Computing*, 2021.
 - [51] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, “The hadoop distributed file system,” *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, pp. 1–10, 2010.
 - [52] M. Zaharia, A. Chen, M. Davidson, *et al.*, “Apache spark: A unified engine for big data processing,” *Communications of the ACM*, vol. 59, no. 11, pp. 56–65, 2016.
 - [53] P. Carbone, A. Katsifodimos, S. Ewen, V. Markl, S. Haridi, and K. Tzoumas, “Apache flink: Stream and batch processing in a single engine,” in *IEEE Data Engineering Bulletin*, vol. 38, 2015, pp. 28–38.
 - [54] J. Dean and S. Ghemawat, “Mapreduce: Simplified data processing on large clusters,” *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.
 - [55] “Seal: Sdn based secure and agile framework for protecting smart city applications from ddos attacks,” *Journal of Network and Computer Applications*, 2020.

- [56] "A context-aware privacy-preserving method for iot-based smart city using software defined networking," *Computers & Security*, 2021.
- [57] "Reliability-aware flow distribution algorithm in sdn-enabled fog computing for smart cities," *IEEE Transactions on Vehicular Technology*, 2022.
- [58] "Time-constrained big data transfer for sdn-enabled smart city," *IEEE Communications Magazine*, 2017.
- [59] J. Medved, R. Varga, A. Tkacik, and K. Gray, "Open-daylight: Towards a model-driven sdn controller architecture," in *2014 15th International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS)*, IEEE, 2014, pp. 1–6.
- [60] P. Berde, M. Gerola, J. Hart, *et al.*, "Onos: Towards an open, distributed sdn os," in *Proceedings of the third workshop on Hot topics in software defined networking*, ACM, 2014, pp. 1–6.
- [61] Ryu Development Team, *Ryu sdn framework*, [urlhttps://osrg.github.io/ryu/](https://osrg.github.io/ryu/), Accessed: 2024-05-21.
- [62] Floodlight Project, *Floodlight openflow controller*. [Online]. Available: <https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/overview>.
- [63] S. Rejeb, M. Othman, and M. A. Mohamed, "A comprehensive review on software defined networking controllers: Classification and performance evaluation," *Journal of Network and Computer Applications*, vol. 204, p. 103399, 2022. DOI: 10.1016/j.jnca.2022.103399. [Online]. Available: <https://doi.org/10.1016/j.jnca.2022.103399>.
- [64] S. Benedict, "Shared mobility intelligence using permissioned blockchains for smart cities," *New Generation Computing*, vol. 40, pp. 1009–1027, 2022. DOI: 10.1007/s00354-021-00147-x. [Online]. Available: <https://doi.org/10.1007/s00354-021-00147-x>.
- [65] S. Otoum, I. A. Ridhawi, and H. Mouftah, "Securing critical iot infrastructures with blockchain-supported federated learning," *IEEE Internet of Things Journal*, vol. 9, no. 4, pp. 2592–2601, 2022.
- [66] Q. H. Cao, I. Khan, R. Farahbakhsh, G. Madhusudan, G. M. Lee, and N. Crespi, "A trust model for data sharing in smart cities," in *2016 IEEE International Conference on Communications (ICC)*, 2016, pp. 1–7.
- [67] S. K. Singh, Y.-S. Jeong, and J. H. Park, "A deep learning-based iot-oriented infrastructure for secure smart city," *Sustainable Cities and Society*, vol. 60, p. 102252, 2020, ISSN: 2210-6707. DOI: <https://doi.org/10.1016/j.scs.2020.102252>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S221067072030473X>.
- [68] K. Crowston, J. Howison, and H. Annabi, "Information systems success in free and open source software development: Theory and measures," *Software Process: Improvement and Practice*, vol. 11, no. 2, pp. 123–148, 2006.
- [69] A. Capiluppi and M. Michlmayr, "From the cathedral to the bazaar: An empirical study of open source spirit," in *Proceedings of the IFIP International Conference on Open Source Systems*, ser. IFIP International Federation for Information Processing, vol. 234, Springer, 2007.
- [70] ISO/IEC, *Systems and software engineering – systems and software quality requirements and evaluation (square) – system and software quality models*, International Organization for Standardization, 2011.
- [71] S. Munot, R. C. Jaiswal, H. Bhoskar, *et al.*, "Wi-fi internetwork simulation and performance analysis using qualnet," in *Smart Trends in Computing and Communications (SmartCom 2024)*, ser. Lecture Notes in Networks and Systems, vol. 948, 2024, pp. 187–198.
- [72] S. Otoum, I. A. Ridhawi, and H. Mouftah, "Securing critical iot infrastructures with blockchain-supported federated learning," *IEEE Internet of Things Journal*, vol. 9, no. 4, pp. 2592–2601, 2022.
- [73] A. Shahrokhi and M. Ahmadi, "Power evaluation of iot application layer protocols," *arXiv preprint arXiv:2405.07326*, 2024.
- [74] H. Gupta, A. V. Dastjerdi, S. K. Ghosh, and R. Buyya, "Ifogsim: A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments," in *Software: Practice and Experience*, vol. 47, 2017, pp. 1275–1296.
- [75] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The hadoop distributed file system," in *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, 2010, pp. 1–10.
- [76] T. Nam and T. A. Pardo, "Conceptualizing smart city with dimensions of technology, people, and institutions," in *Proceedings of the 12th Annual International Digital Government Research Conference*, 2011, pp. 282–291.
- [77] M. Zaharia, A. Chen, M. Davidson, *et al.*, "Apache spark: A unified engine for big data processing," *Communications of the ACM*, vol. 59, no. 11, pp. 56–65, 2016.
- [78] X. Wang, B. C. Ooi, and K.-L. Tan, "Zilliqa: A scalable and secure blockchain platform," in *Proceedings of the 2019 ACM SIGMOD International Conference on Management of Data*, 2019, pp. 1393–1405.