BLADE : Un outil d'aide à la décision automatique pour guider le choix de technologie Blockchain

Nicolas Six, Nicolas Herbaut, Camille Salinesi

BLADE : Un outil d'aide à la décision automatique pour guider le choix de technologie Blockchain

BLADE : An automated decision support tool to guide the choice of a Blockchain technology

Nicolas Six, Nicolas Herbaut, Camille Salinesi[1]

[1] CRI, Université Paris 1 Panthéon-Sorbonne, France, {nom}.{prenom}@univ-paris1.fr

RÉSUMÉ. Les entreprises souhaitant déployer des solutions basées sur la blockchain sont confrontées à une pléthore de technologies concurrentes ayant chacun un grand nombre de paramètres propres devant être ajustés par un expert. Les études existantes proposant des modèles de décision pour blockchain ne proposent pas de solution pour le choix des paramètres assurant la mise en œuvre automatique des exigences non-fonctionnelles. Cet article, version étendue de précédents travaux, présente BLADE (BLockchain Automated DEcision Engine), un outil d'aide à la décision pour blockchain permettant de mieux prendre en compte les exigences de haut niveau et préférences. Tout d'abord, une base de connaissances de solutions blockchain est construite à partir de documentations, de livres blancs et de travaux de recherche académiques. Celle-ci permet à BLADE d'exécuter un processus de décision multicritère automatisé donnant la solution la plus pertinente à partir d'exigences et préférences, extraites de la norme de qualité logicielle ISO 25010. L'implémentation de cet outil est ensuite réalisée au sein d'une plateforme web permettant la saisie aisée des exigences et des préférences de l'utilisateur. Enfin, l'approche proposée est validée grâce à un cas d'étude de gestion de chaîne logistique. Cette étude est une première étape afin de concevoir une solution permettant la conception et l'implémentation d'applications blockchain de bout-en-bout.

ABSTRACT. Companies trying to build new solutions using blockchain are confronted with a plethora of available concurrent technologies that have many control knobs which require fine-tuning by experts. Exiting studies that build decision models for blockchain adoption or selection lack an automated way to use non-functional requirements to provide recommendations. This article, extended from previous works, introduces BLADE (BLockchain Automated DEcision Engine), a decision support tool for blockchain to better take into account high level requirements and preferences for recommendations. From documentations, white papers and academic papers, a knowledge base of blockchain solutions is constructed. This allows BLADE to execute an automated multi-criteria decision process giving the most relevant solution based on requirements and preferences, extracted from the ISO 25010 software quality standard. An implementation of this tool is performed within a web platform allowing the easy capture of user requirements and preferences for recommendation. Finally, the proposed approach is validated on a supply chain management case study. This study is a first step in order to design a solution allowing the design and implementation of end-to-end blockchain applications. While still limited in scope, BLADE will include more blockchain alternatives and more flexible requirements inputs in future work.

MOTS-CLÉS. Blockchain, Ingénierie des exigences, Aide à la décision multicritère

KEYWORDS. Blockchain, Requirements engineering, Multi-criteria decision analysis

1. Introduction

La blockchain (ou chaîne de blocs) est un registre distribué maintenu à jour par un ensemble de nœuds distribués. Les utilisateurs peuvent interagir avec les nœuds afin d'envoyer à la blockchain des transactions. Un registre blockchain prend la forme d'un ensemble de blocs contenant les transactions soumises par les utilisateurs, ainsi que des métadonnées sur lui-même ou le registre. Chaque bloc est relié au précédent par une empreinte (en anglais, *hash*) de celui-ci. Dans ce sens, comme la modification d'un bloc altérerait cette valeur ainsi que toutes les autres valeurs de hachage des blocs suivants, il est théoriquement impossible d'altérer le contenu d'un bloc. Les nouveaux blocs sont formés par un sous-ensemble de nœuds chargés de regrouper les transactions dans un bloc et de le valider en mettant en œuvre différents processus cryptographiques en fonction de la blockchain utilisée, afin de garantir sa validité lors de son ajout à la blockchain.

La blockchain est apparue lors de la création de Bitcoin [Nak08], afin de permettre aux utilisateurs d'échanger la cryptomonnaie de même nom. Par la suite, de nombreuses blockchains ont pu voir le jour. Ethereum, la plus connue, est plébiscitée [Woo14] pour sa capacité à déployer et à interagir avec des contrats intelligents (en anglais, smart contracts), logés dans la blockchain [Sza97]. Les contrats intelligents pour la blockchain permettent non seulement d'exécuter des fonctions directement au sein de celle-ci, mais aussi de stocker des états. Ils bénéficient donc directement des propriétés particulières de la blockchain, qui sont intégrité, décentralisation, non-répudiation des transactions et transparence [Wus18]. Cela donne à la blockchain un statut de tiers de confiance artificiel, où il est possible de faire confiance au code et à la puissance du réseau contrairement aux tiers de confiance conventionnels tels que les banques ou les gouvernements, qui n'assurent la validité des transactions que de part leur statut.

Ces caractéristiques ont attiré l'attention des industriels et universitaires, qui voient en la blockchain un moyen de révolutionner la manière d'échanger de la valeur entre individus ainsi que de garantir la véracité et l'intégrité des données stockées dans celle-ci. En effet, la blockchain serait "un support numérique natif pour la valeur, par lequel nous pourrions gérer, stocker et échanger de multiples biens [...] de pair à pair et de manière sécurisée" [Tap16]. De ce fait, on trouve dans la littérature de nombreux cas d'utilisations pertinents de la blockchain dans différents secteurs d'activités, tels que la gestion de chaîne logistique [Abe16], la finance [Hyv17], le contrôle du réseau [Her17], l'identité décentralisée numérique [Tak18] ou encore la santé [Ekb16].

Cependant, malgré son potentiel, la blockchain fait face à de nombreux freins à l'adoption. Ces freins peuvent être catégorisés en trois catégories : organisationnels, légaux, et techniques [Pre20]. Par exemple, une étude réalisée par l'entreprise PwC en 2018 [1] décrit les problèmes les plus récurrents rencontrés par les entreprises : une régulation incertaine sur le sujet, un manque de confiance envers les autres acteurs lors de leur participation à un projet utilisant la blockchain, ainsi que la difficile gestion de la propriété intellectuelle des données et biens qui y sont enregistrés. Ces problèmes, majoritairement organisationnels et légaux, sont résolus petit à petit grâce à la collaboration des acteurs de l'écosystème blockchain et des instances juridiques et gouvernementales compétentes. Néanmoins, les entreprises sont encore confrontées à un frein technologique, et ce pour plusieurs raisons :

1. Elles peuvent aussi avoir du mal à intégrer la blockchain à leurs systèmes d'information et processus métiers existants, car il n'existe pas encore de bonnes pratiques identifiées et éprouvées en entreprise par les architectes logiciels. Afin de pallier ce point, des études ont été menées afin d'assister l'intégration de la blockchain dans des architectures logicielles. Dans ce sens, une étude propose une collection de modèles architecturaux pour blockchain et smart-contracts, ainsi que les différents cas où ces modèles sont applicables [Xu18].
2. Elles peinent à concevoir et implémenter des applications blockchain, en raison du décalage entre technologies blockchain et autres solutions "traditionnelles".
3. Elles peuvent rencontrer des difficultés à recruter des collaborateurs spécialisés en blockchain, la technologie étant encore jeune.

Les développeurs se posent plusieurs questions. Quelle blockchain utiliser dans un contexte donné, sachant qu'il existe de nombreuses technologies concurrentes avec, pour chacune, des propriétés et carac-

1. https ://www.pwc.com/gx/en/issues/blockchain/blockchain-in-business.html

téristiques qui leur sont propres ? N'est-il pas finalement plus raisonnable d'utiliser une solution "éprouvée" au lieu d'une blockchain (base de données, microservices ...) ? Comment configurer les différents paramètres de la blockchain, qui ont un impact important sur la satisfaction des exigences (performances, résilience, sécurité ...) tels que l'algorithme de consensus ou l'intervalle inter-blocs, nécessitant souvent l'intervention d'experts dans le domaine pour aboutir à un résultat satisfaisant les exigences ?

Plusieurs travaux ont été menées pour répondre aux deux premières questions. Dans [Wus18, Koe18], des modèles de décision facilitent le choix de la solution blockchain. [Bel19] présente un *vadémécum* contenant toutes les informations nécessaires à la compréhension de la blockchain d'un point de vue technique, ainsi qu'un modèle de décision pour la blockchain appliqué à plusieurs scénarios d'exemple. Les systèmes de recommandation proposés s'appuient en général sur une série de questions abstraites, relatives à des problématiques telles que "ai-je besoin d'une blockchain ?" ou "quel type de blockchain adopter ?". En revanche, elles ne fournissent pas de recommandations précises, et n'entrent pas dans les détails de conception ou les choix des paramètres et propriétés blockchain. Les utilisateurs souhaitant obtenir une recommandation plus précise restent donc dépendants d'experts, ce type d'étude étant difficile pour les personnes non initiées. De plus, ces études se concentrent principalement sur les exigences orientées solution blockchain, alors que les utilisateurs ont des exigences liées à la qualité logicielle (performance, sécurité, fiabilité ...). Les liens qui associent les attributs blockchain aux qualités logicielles telles que définies en ingénierie, sont souvent peu explicites et il est difficile de quantifier l'impact d'un paramètre blockchain sur les qualités logicielles de la solution finale. Enfin, lorsque le nombre d'attributs techniques considérés devient important, il est extrêmement compliqué de réaliser un choix les prenant tous en compte, la complexité des calculs rendant vite hors de portée tout choix rationnel.

Pour pallier à ces difficultés, cet article étendu, fruit de précédents travaux [Six20b], introduit un outil de décision automatisé intitulé BLADE (BLockchain Automated DEcision Engine) qui détermine la blockchain la plus pertinente pour une série d'exigences et de préférences, relatives à la qualité logicielle. Celles-ci sont comparées aux caractéristiques des alternatives considérées par une méthode d'aide à la décision multicritère. Ces caractéristiques sont établies sous la forme d'une base de connaissance construites à partir de la littérature existante (expériences, revues de littérature ...), les livres blancs des blockchains considérées ainsi que les résultats d'expériences (benchmarks). L'article présente également une application de BLADE à un cas d'utilisation pertinent dans le domaine de la gestion de chaîne logistique. Cette partie sera l'occasion de valider les résultats de BLADE, par le biais d'expériences manuelles confirmant les décisions prises par l'outil.

Dans cette version étendue, l'implémentation de BLADE est décrite. Trois parties sont présentées : (1) la base de connaissances, et ses différentes tables de données, (2) l'API, point d'entrée de l'outil permettant de calculer les scores via l'implémentation du processus de décision au sein d'un solveur, et (3) la plateforme web, visible par l'utilisateur et permettant la saisie aisée des exigences par une interface guidant l'utilisateur dans ce but. Dans cette implémentation, BLADE se voit également doté d'un modèle de dépendances, permettant d'avertir l'utilisateur lors de la saisie d'exigences conflictuelles.

La section 2. de cette étude est consacrée aux principes et aux caractéristiques du processus de décision composant BLADE. La section 3. présente l'implémentation faite de BLADE, et de sa mise à disposition par le biais d'une plateforme web permettant la saisie facile des exigences. BLADE est ensuite utilisé dans la section 4. dans le contexte d'un cas d'étude sur la gestion de chaîne logistique. Les travaux connexes à cette étude sont présentés dans la section 5., puis une discussion est faite quant à l'approche

de cette étude et les résultats obtenus dans la section 6.. Enfin, la section 7. conclut cette étude et présente les travaux futurs envisagés.

2. Construction du processus de décision

Cette section présente le processus de décision contenu dans BLADE afin de déterminer quelle blockchain utiliser à partir des exigences. Une présentation des alternatives blockchains ainsi que de leurs attributs liés est faite. Six blockchains sont considérées, chacune présentant 14 attributs. Les entrées nécessaires au fonctionnement du processus de décision sont ensuite détaillées. Elles sont constituées d'exigences et de préférences relatives aux attributs des blockchains composant la base de connaissances. Enfin, la logique interne du processus de décision est présentée. Afin d'émettre une recommandation, l'algorithme d'aide à la décision multicritère TOPSIS est utilisé [Lai94]. Celui-ci permet de classer des alternatives ayant des attributs de différents types et échelles, à partir d'une matrice de pondération (pour chacun des attributs) donnée en entrée. Une présentation détaillée de cet algorithme est faite dans la section 2.2..

2.1. *Entrées*

La précision d'un algorithme d'aide à la décision multicritère dépend majoritairement des données saisies en entrée. Cette sous-section présente une approche pour construire une base de connaissance fiable et adaptée, ainsi qu'une méthode pour éliciter les poids qui seront appliqués à chacun des critères pour l'exécution du processus de décision.

Alternatives et attributs

Pour alimenter le processus d'aide à la décision, une première version de base de connaissance a été construite, contenant un ensemble d'alternatives de blockchains a_m et de leurs attributs respectifs c_n. La Table **2.2** montre l'ensemble de ces alternatives et de ces attributs. Ce panel spécifique de blockchains a été choisir car (hors Bitcoin) elles sont considérées comme les blockchains les plus utilisées par les fournisseurs de service blockchain en entreprise [2]. Cependant, la blockchain Bitcoin a tout de même été incluse dans la base de connaissance, car elle la plus connue du grand public, mais aussi la plus ancienne. De plus, il existe d'autres applications à Bitcoin qui ne sont pas liées à son utilisation en tant que cryptomonnaie, notamment par l'usage du champ *OP_RETURN* des transactions Bitcoin [Bar17].

L'objectif de ce travail étant d'aider les entreprises à prendre des décisions sur la blockchain à utiliser sans avoir d'expertise particulière quant à la configuration de celle-ci, un ensemble de critères a été choisi, catégorisables par les différents macro-caractéristiques [3] relatives à la qualité logicielle proposés par la norme ISO 25010 [4], un standard définissant les différents attributs qualité à considérer afin de garantir la qualité d'un système ou d'un logiciel lors de son implémentation. Les attributs ont été choisis pour leur

2. https ://www.hfsresearch.com/pointsofview/whos-winning-the-battle-of-enterprise-blockchain-platforms
3. Ici, le terme macro-caractéristiques réfère aux 7 catégories de qualité logicielle (Sécurité, Fiabilité, ...) sous lesquelles sont introduits les attributs de qualité logicielle.
4. https ://iso25000.com/index.php/en/iso-25000-standards/iso-25010

pertinence envers les considérations à avoir lors du choix d'une blockchain, mais aussi pour la possibilité à les retranscrire sous format numérique. Par conséquent, ces critères ne sont pas seulement spécifiques à la technologie blockchain, mais relatifs à la qualité système.

La fig. **1** présente un diagramme indiquant les attributs choisis dans le modèle de décision, et leurs connexions entre les macro-caractéristiques relatives à la qualité logicielle, ainsi que les caractéristiques blockchain influant sur eux (par exemple, l'intervalle inter-blocs a un impact sur la latence et le débit de transaction par seconde de la blockchain).

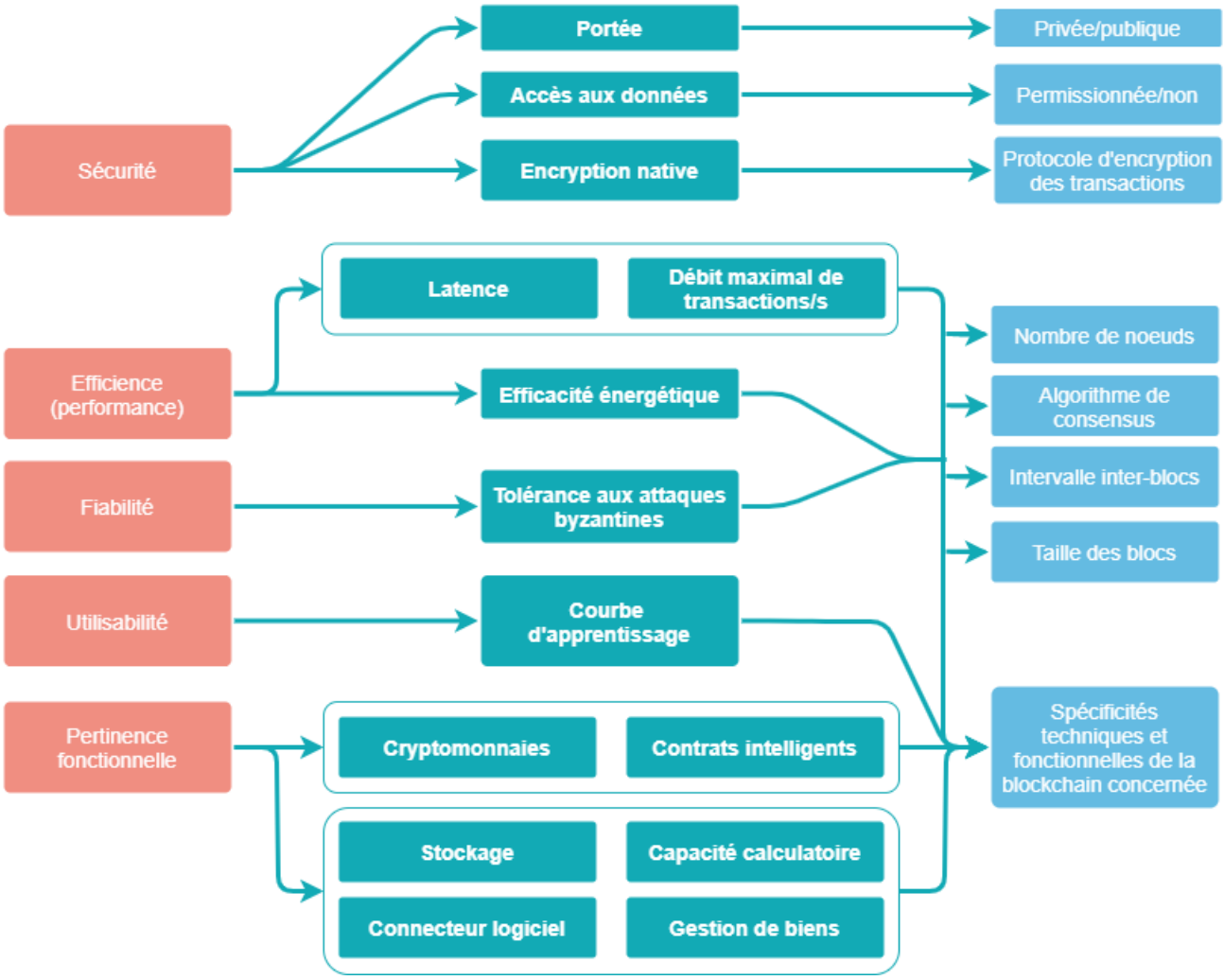

Figure 1.: *Attributs choisis (milieu) reliés aux qualités système (à gauche) et blockchain (à droite).*

Les valeurs saisies pour chacun des attributs d'alternatives de la base de connaissance proviennent de différentes sources : études ([Bel19]), livres blancs ([Bro16], [Nak08], [Woo14]), documentations techniques et littérature scientifique ([And18]).

Certaines de ces valeurs sont floues (marquées par le symbole $\mp$), car soumises à variation de la topologie et la configuration du réseau blockchain ainsi que des caractéristiques techniques des nœuds le composant (CPU, mémoire vive ...). Leur valeur est donc construite à partir d'attributs connus, comme l'algorithme de consensus supporté (un algorithme tolérant les fautes byzantines comme l'algorithme PoW de Bitcoin aura un débit de transaction plus faible qu'un algorithme tolérant aux pannes, comme Raft utilisé par Hyperledger Fabric). Néanmoins, ces valeurs pourront être fixées lorsque les paramètres blockchain sont connus. BLADE devant prendre en compte des actifs déjà présents dans l'entreprise

(comme l'infrastructure technique ou les modèles de processus métiers), un objectif futur sera la réalisation de tests de performance futurs afin de pouvoir donner une valeur fixe aux attributs variables en fonction du contexte donné. Cette base de connaissance sera également variable dans le temps. Les valeurs des attributs des différentes blockchains choisies seront modifiées si nécessaire (mise à jour d'un des éléments d'une blockchain). Ces variations pouvant avoir un impact sur le choix de la meilleure alternative par BLADE, il sera nécessaire d'évaluer l'ancienneté de la base de connaissance afin de déterminer si la recommandation est pertinente à un instant donné.

Poids et conditions définis par l'utilisateur

Afin d'obtenir une préconisation de blockchain conforme aux attentes de l'utilisateur, le processus de décision automatisé doit prendre en compte les exigences et préférences de celui-ci. Lorsque l'utilisateur est invité à saisir ses choix, il peut marquer un critère comme *Requis* ou *Indésirable*. Lors de la prise de décision, une alternative dont l'attribut ne respecterait pas l'une de ces deux exigences serait automatiquement disqualifiée des alternatives possibles, indépendamment de son score obtenu par l'exécution de l'algorithme d'aide à la décision multicritère.

L'utilisateur peut également indiquer ses préférences quant aux attributs, par le biais d'une échelle de notation prenant la forme d'une suite de labels, chacun des labels étant lié à une valeur numérique (définis dans la table **2.1**). Le choix d'un label permet donc d'obtenir une valeur de préférence $p_n \in \ltimes$ pour chacun des critères c_n. Afin d'obtenir les poids de chacun des critères ω_n de telle façon à ce que la somme de ces poids soit égale à 1, il faut diviser chacune des préférences p_n pour un critère par la somme des préférences.

Variable linguistique	**Valeur de préférence** p_n
Extrêmement désirable	4
Tout à fait désirable	3
Désirable	2
Faiblement désirable	1
Indifférent	0

Tableau 2.1.: *Échelle de notation associant labels et valeurs de préférence.*

2.2. *Logique interne*

Tout d'abord, le processus de décision effectue un premier filtrage des alternatives en fonction des exigences de l'utilisateur. Si un critère marqué comme *Requis* ou *Indésirable* n'est pas respecté par l'une des alternatives, elle est automatiquement éliminée, peu importe le score qu'elle aurait pu obtenir à l'aide de l'algorithme de décision qui suit. Pour un critère *Requis* qui n'est pas un booléen, l'utilisateur spécifie une valeur extremum. Par exemple, si un certain nombre de transactions par seconde est requis, les alternatives qui n'atteignent pas la valeur seuil seront disqualifiées.

Le processus de décision automatisé sur les alternatives restantes repose sur l'utilisation d'un algorithme d'aide à la décision multicritère appelé TOPSIS (Technique for Order of Preference by Similarity to Ideal Solution) [Lai94]. L'algorithme TOPSIS est basé sur le fait que l'alternative a_m la plus pertinente

pour un ensemble de choix donné doit être la plus proche possible de la solution idéale positive A^+ et la plus éloignée de la solution idéale négative A^-.

Attributs/Alternatives	**Bitcoin**	**Ethereum**	**Ethereum**	**Hyperledger Fabric**	**Corda**	**Tezos**
Algorithme de consensus	PoW [5]	PoW	PoA [6]	Raft	PBFT [7]	PoS [8]
Ouvert publiquement	Oui	Oui	Non	Non	Non	Oui
Permissions	Non	Non	Non	Oui	Oui	Non
Encryption native	Non	Non	Non	Oui	Oui	Non
Débit (tx/s)	3,8	15	$\mp$100	$\mp$1000	$\mp$1000	30
Latence (s)	3600	180	$\mp$10	<1	<1	60
Efficient en énergie	Non	Non	Oui	Oui	Oui	Oui
Tolérant aux fautes byzantines	50,00%	50,00%	33,30%	0,00 %	33,30%	33,30%
Contrats intelligents	Non	Oui	Oui	Oui	Oui	Oui
Cryptomonnaies	Oui	Oui	Oui	Non	Non	Oui
Element de stockage	Basique	Avancé	Avancé	Avancé	Avancé	Avancé
Elément de calcul	Non	Avancé	Avancé	Avancé	Avancé	Avancé
Elément gestionnaire de biens	Basique	Avancé	Avancé	Avancé	Avancé	Avancé
Connecteur logiciel	Non	Avancé	Avancé	Avancé	Avancé	Avancé
Courbe d'apprentissage	Faible	Moyenne	Moyenne	Très élevé	Très élevé	Très élevé

Tableau 2.2.: *Alternatives et attributs retenus.*

Le choix de cet algorithme a été guidé grâce à une étude présentant un état de l'art des études portant sur le choix d'une méthode d'aide à la décision multicritère [Kor07]. Celle-ci propose un cadre de décision incluant différentes propriétés sur lesquelles porter attention lors du choix d'une méthode d'aide à la décision multicritère. Nous avons jugé que la méthode TOPSIS était adaptée au processus de décision, notamment car elle supporte l'analyse multicritère d'attributs nombreux et variés (ce qui est le cas lors de la comparaison de deux blockchains) tout en étant simple d'implémentation et précise dans la décision. Elle permet également de prendre en compte des poids définis par un utilisateur, ce qui est requis étant donné le mode opératoire du processus de décision. Ce n'est par exemple pas le cas de la méthode de Condorcet, ou Borda [Zwi16], ou seuls les attributs auraient joué un rôle dans la sélection de la meilleure alternative. Aussi, un autre candidat potentiel à cet outil est AHP [Pod09]. Possédant un grand nombre de similarités avec TOPSIS, cette dernière méthode a néanmoins été retenue pour sa plus grande facilité à saisir les poids. En effet, AHP requiert de comparer les attributs des deux à deux pour exprimer l'importance qu'a un attribut par rapport à un autre pour l'utilisateur.

Plusieurs étapes sont nécessaires à l'exécution de la méthode TOPSIS, détaillées dans les sous-parties

5. Proof-of-work (PoW), preuve de travail
6. Proof-of-Authority (PoA), preuve d'autorité
7. Practical Byzantine Fault Tolerance
8. Proof-of-Stake (PoS)

suivantes.

Construction de la matrice - Soit m alternatives a et n attributs c pour chacune d'entre elles. Le regroupement de ces alternatives donne une matrice $X = \{x_{ij}\}$ pour $\{i \in \mathbb{N} \mid 1 \leq i \leq m\}$ et $\{j \in \mathbb{N} \mid 1 \leq j \leq n\}$.

Normalisation de la matrice et application des poids - Normaliser les critères ayant des unités et échelles différentes entre eux est nécessaire afin de pouvoir les comparer entre eux. C'est également à cette étape que sont appliqués les poids provenant des préférences de l'utilisateur.

$$v_{ij} = r_{ij} * \omega_j = \frac{x_{ij}}{\sqrt{\sum_{i=1}^{m} x_{ij}^2}} * \omega_j \tag{1}$$

Calcul des solutions idéales positive et négative puis mesure de l'écart avec chacune des alternatives - En sélectionnant les meilleures et les pires performances de chacun des critères de la matrice de décision normalisée pondérée, on peut déterminer les solutions idéales positive A^+ et négative A^- afin de mesurer l'écart de chacune des alternatives avec ces deux solutions que l'on notera S^+ et S^-.

$$Pour A^+ = (v_1^+, ..., v_j^+), \tag{2}$$

$$Si^+ \triangleq \sqrt{\sum_{j=1}^{m} (v_{ij} - vj^+)^2} \tag{3}$$

$$Pour A^- = (v_1^-, ..., v_j^-), \tag{4}$$

$$Si^- \triangleq \sqrt{\sum_{j=1}^{m} (v_{ij} - vj^-)^2} \tag{5}$$

Calcul de la distance relative C_i avec la solution idéale - Cette dernière étape permet de donner un score à chaque alternative, qui représente sa distance avec la solution idéale. L'ordonnancement de ces scores permet de définir la meilleure alternative possible par rapport aux alternatives données ainsi que des préférences de l'utilisateur.

$$C_i = \frac{Si^-}{Si^+ + Si^-}$$

3. Implémentation

Cette section détaille l'implémentation de BLADE, dans un outil déployé en ligne [9] et disponible en open-source sur Github [10]. L'implémentation de chacune des parties est présentée : (1) la base de connaissances, (2) l'API contenant les différents solveurs composant BLADE, et (3) la plateforme web

9. https ://recommender.blade-blockchain.eu/
10. https ://github.com/nicoSix/blade-project

permettant la saisie aisée des exigences. Les solveurs sont également présentés plus en détail dans leurs sous-sections respectives.

3.1. *Architecture et implémentation de l'outil*

Pour l'implémentation de cet outil, le modèle architectural trois-tiers est utilisé, basé sur la séparation entre client, serveur, et données [Bas03]. Ici, la partie *données* sera la base de connaissances. Seulement accessible via la partie *serveur*, elle permet de stocker l'ensemble des alternatives et de leurs attributs respectifs. Pour cela, plusieurs tables existent :

— La première contient l'ensemble des alternatives. Chaque alternative est définie par son nom, et son algorithme de consensus. En effet, ces deux attributs sont suffisants pour identifier une blockchain parmis celles existantes. Les alternatives contiennent également un tableau qui définit les 14 attributs utilisés dans l'aide à la décision, présentés dans la figure fig. **1**.

— Une seconde table contient l'ensemble des attributs disponibles pour décision. Un attribut y est défini par son label (eg. latence), son coût (une variable prenant pour valeur 0 ou 1 et permettant au solveur de savoir s'il faut maximiser ou minimiser l'atttibut) ainsi que son type (eg. numérique, booléen, ...). L'utilisation d'une telle structure rend plus facile la maintainabilité de la base de connaissances dans le temps. En effet, les alternatives introduites dans la base de connaissances doivent respecter ce plan.

— Enfin, une troisième table permet de stocker des valeurs littérales et leur équivalence numérique. En effet, certains attributs des alternatives peuvent se voir attribuer une valeur littérale (eg. *très faible, moyen, ...*) au lieu d'une valeur numérique. Ces valeurs littérales sont stockées dans la base de connaissances, associées à une valeur numérique. Cela permet également de faciliter la mise à jour de la base de connaissances, en permettant de modifier la valeur numérique d'un attribut exprimé en valeur littérale pour toutes les alternatives et en une fois.

La partie *serveur* est une API (Application Programming Interface), développée en langage Python et utilisant le framework Flask [11]. Le principal avantage obtenu par le découpage de l'application en tiers, et donc par l'indépendance du serveur vis-à-vis du client, est la possibilité de réutiliser l'API dans d'autres applications. Ainsi, l'API peut être intégrée dans de futurs travaux de recherche, présentés dans la section 7.. Cette API permet d'exécuter des requêtes utilisant le solveur de score, qui est le processus de décision, ainsi qu'un autre solveur permettant d'identifier les dépendances d'exclusion lors de la saisie des exigences (présentés dans la sous-section 3.2. et la sous-section 3.3.).

Enfin, le client est une application Web développée en Javascript, et est basée sur le framework React [12]. Ce framework facilite la conception et le développement d'applications mono-pages, au travers de composants et d'états associés. Il a été choisi pour sa compatibilité avec l'approche trois-tiers, ainsi que sa capacité à proposer une application performance et réactive aux choix utilisateurs, des attributs nécessaires pour une implémentation réussie de l'outil. La fig. **2** montre un aperçu de l'interface proposée à l'utilisateur pour qu'il puisse saisir ses exigences et préférences.

La sélection s'opère dans la section de gauche, intitulée *Requirement selection*. Comme présenté dans la section 2., les attributs sont regroupés par macro-caractéristiques de qualité logicielle provenant de

11. https ://flask.palletsprojects.com/en/1.1.x/
12. https ://fr.reactjs.org/

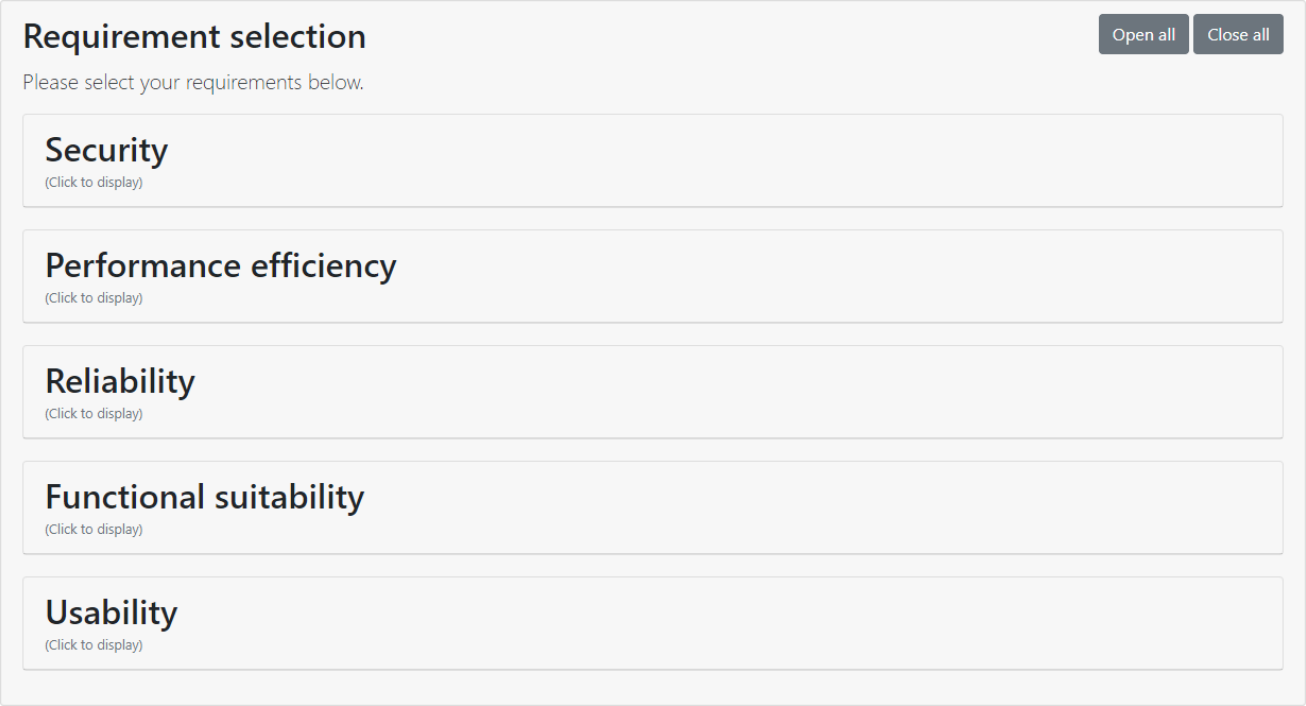

Figure 2.: *Capture d'écran de l'interface de sélection des exigences dans l'outil BLADE.*

la norme ISO 25010. L'utilisateur peut déployer les différents panneaux correspondant à ces macro-caractéristiques afin d'afficher les menus de sélection pour chacun des attributs disponibles. Il peut ensuite saisir son niveau de préférence, si l'attribut est *exigé*, ainsi que sa valeur désirée. L'affichage des résultats se fait en temps réel, dans la partie droite de l'interface. Un tableau est affiché, contenant les alternatives classées par score. Un historique des sélections faites par l'utilisateur est également affiché, pour résumer les attributs entrant en compte dans les recommandations faites par l'outil. Cela lui permet également de supprimer des préférences et exigences pour altérer les recommandations, si nécessaire.

3.2. *Solveur pour génération des scores*

Le solveur pour génération des scores est la partie la plus importante du processus de décision. Par un fichier contenant les exigences et les préférences de l'utilisateur en entrée, il est chargé de calculer un score pour chacune des alternatives en utilisant leurs caractéristiques définies en base de connaissances. Lorsqu'une décision doit être prise par le processus d'aide à la décision, une requête est faite via l'API au serveur, qui la transmet au solveur. Les scores étant transmis sous format JSON, il sera chargé de changer le format de la structure de données dans le format attendu par l'outil. Ensuite, le solveur est chargé de calculer le score de chacune des alternatives et de disqualifier celles qui ne satisfont pas les exigences en entrée. Le fonctionnement de cette partie est défini en détail dans la section 2.. Celui-ci retourne les résultats au serveur, lui-même retourne les résultats au client. Cela permet la mise à jour du tableau contenant les scores pour chacune des alternatives dans l'outil.

3.3. *Solveur pour génération du modèle de dépendances*

Lors de la prise de décision, l'utilisateur peut saisir des exigences conflictuelles. Dans l'outil, une exigence est conflictuelle avec une autre lorsque la sélection des deux exigences disqualifie toutes les

alternatives dans le processus de décision. Permettre à l'utilisateur de saisir des exigences conflictuelles n'est pas quelque chose de souhaitable, car l'objectif de cet outil est de proposer une aide à la décision depuis la sélection des exigences jusqu'à l'affichage des résultats. Afin de palier à ce problème, un solveur a été implémenté coté serveur, permettant de générer un modèle de dépendances pour l'application. Ici, une dépendance est dite *d'exclusion*, c'est-à-dire que le modèle indique pour une exigence quelles sont celles directement en conflit (qui empêchent donc l'obtention d'au moins une alternative valide si toutes deux sélectionnées).

A partir de la base de connaissances, le solveur est chargé de récupérer l'ensemble des attributs disponibles pour les alternatives, et de générer les paires correspondantes. Pour chaque attribut existant, le solveur va récupérer toutes les valeurs qu'il peut prendre, en itérant sur les alternatives disponibles. Il va ensuite créer des paires à partir de toutes ces valeurs, de façon à ce que pour une valeur d'attribut donnée, il existe un nombre de paires égal au nombre de valeurs existantes. Ensuite, un algorithme de retour sur trace (aussi appelé *backtracking algorithm*) est chargé d'appliquer deux contraintes aux paires : (1) une paire ne peut contenir deux attributs identiques et de même valeur, (2) aucune des alternatives contenues dans la base de connaissance ne doit pouvoir satisfaire les deux exigences formulées dans la paire.

D'un point de vue plus formel, étant donnée a une alternative comprise dans A l'ensemble des alternatives, C_n une contrainte du solveur et (r_a, r_b) une paire de valeurs, on pose les contraintes suivantes (eq. (1)) :

$$\begin{aligned} C_1 \text{ satisfaite } &\Leftrightarrow \nexists a \in A, (r_a, r_b) \in a, \\ C_2 \text{ satisfaite } &\Leftrightarrow r_a \neq r_b. \end{aligned} \tag{1}$$

La non-satisfaction de l'une des contraintes entraîne automatiquement la suppression d'une paire de la liste initiale des paires. A la fin de l'exécution de l'algorithme, le résultat est un ensemble de paires représentant l'ensemble des exigences en conflit dans la base de connaissances.

Ce modèle de dépendances est stocké en base de données puis réutilisé à chaque sélection d'exigence par l'utilisateur. En effet, lors du choix d'une exigence, le client va parcourir ce modèle de dépendances et automatiquement griser les différentes valeurs dans le formulaire correspondant aux choix incompatibles (car rendant impossible le choix d'au moins une alternative). Une indication quant aux valeurs conflictuelles sera également donnée à l'utilisateur pour qu'il puisse adapter ses exigences si besoin.

4. Application à un cas d'étude de gestion de chaîne logistique

Afin de tester puis valider l'outil proposé, cette section montre l'utilisation de BLADE pour obtenir une recommandation dans le cadre d'une étude de cas blockchain [Lon19]. Ici, ce cas d'étude propose d'introduire un système blockchain à une chaîne logistique afin de permettre le partage de données entre les différents acteurs. D'abord, le scénario proposé par l'étude citée est détaillé, puis les différents attributs requis pour la blockchain à implémenter qui découlent de ce sujet afin d'exécuter BLADE. Enfin, les résultats sont validés en utilisant un outil de test de performance blockchain implémenté dans ce but.

4.1. *Scénario "Big-Box"*

La chaîne logistique modélisée dans cette étude est constituée d'un réseau de détaillants de la chaîne de magasins Big-Box, ainsi que de trois grossistes qui alimentent leurs magasins. Les détaillants Big-Box étant regroupés dans une même organisation, l'étude considère qu'il y a un partage de données en temps réel, transparent et fiable entre les magasins. Cependant, les détaillants sont tout de même en compétition, car ils opèrent dans une même région géographique et proposent tous les mêmes gammes de produits. Les clients arrivent au magasin et sélectionnent des produits ainsi que leurs quantités respectives. Si le stock du magasin permet de satisfaire la demande, le produit concerné est réservé dans la quantité demandée ; sinon, une réservation partielle est proposée ; la demande non satisfaite est utilisée pour calculer les réapprovisionnements. L'inventaire est fait avant l'ouverture des magasins ; si une commande est nécessaire alors le détaillant peut choisir l'un des grossistes pour s'approvisionner, en prenant en compte le délai d'approvisionnement, la demande actuelle et la quantité instantanément disponible pour les produits souhaités. Si la quantité d'un produit possédée par un grossiste devait ne pas être suffisante pour tous les détaillants, alors celui-ci est partagé équitablement.

Dans ce contexte, partager la demande globale des différents détaillants entre les grossistes pourrait permettre de prédire plus facilement le stock à constituer pour répondre aux demandes des détaillants. Cependant, les acteurs de ce système restent en concurrence et ne se font donc pas confiance mutuellement. L'étude propose donc la mise en place d'une blockchain permettant d'y enregistrer des données liées à la chaîne d'approvisionnement (notamment la demande du marché) sous la forme de valeur de hachage, ainsi que les différents tiers ayant accès à ces données (s'ils sont autorisés par la blockchain, ils peuvent faire directement une requête pour obtenir ces données auprès du tiers qui les a enregistrées). La sauvegarde de cette valeur permettant d'attester de la véracité des données transmises entre tiers, ils peuvent dorénavant se faire confiance entre eux.

4.2. *Exigences du client Big-Box*

Pour pouvoir préconiser la blockchain à l'aide du processus de décision, il convient d'identifier les attributs de qualité ainsi que les exigences et préférences quant à ces attributs (Section 2.). Pour cela, un tableau d'exigences textuelles est dressé à partir du contenu de l'étude de cas, accompagné d'un résumé des acteurs et systèmes présentés dans cette étude de cas ou déduits de celle-ci. Cela permet ensuite de déterminer en partie les préférences et exigences que l'utilisateur aurait choisi pour saisie dans BLADE.

Exigences textuelles

Avant toute chose, il faut d'abord extraire les exigences textuelles du système blockchain à concevoir. Cependant, ces exigences seront ambigues sans une définition claire des acteurs du système et du système lui-même. Pour lever cette ambiguité, la liste suivante décrit les différents acteurs du système :

— BigBox : entreprise pilotant le réseau de détaillants ainsi que le projet blockchain.
— Grossiste : achète des biens à des fabricants (non spécifiés) et les revends à des détaillants de l'entreprise BigBox.
— Détaillant : gère un ou plusieurs magasins du réseau de l'entreprise BigBox.

— Consortium : un groupe de grossistes et détaillants permettant l'utilisation et assurant le bon fonctionnement de la blockchain.
— Membre du consortium : un grossiste ou détaillant ayant le droit d'inscrire des données sur la blockchain et voter pour l'acceptation de nouveaux membres.
— Candidat : un grossiste ou détaillant ayant postulé pour rejoindre le consortium.

Ces acteurs vont interagir avec le système, qui est composé de deux parties principales : (1) la blockchain, un réseau composé de noeuds, qui stocke le smart-contract nécessaire au bon fonctionnement de l'application proposée par l'étude de cas, et (2) l'application "off-chain", permettant aux différents acteurs d'interagir avec la blockchain et de stocker des informations de stock.

Une fois ces acteurs et systèmes définis, il est possible d'analyser les différentes exigences relatives à l'étude de cas. La table **4.3** détaille ces exigences relevées par catégorie, et affiche leurs dépendances entre elles.

Exigences et préférences pour BLADE

A partir des exigences textuelles, il est possible de formaliser les préférences et exigences qui seront soumises à BLADE. Cette section reprend chacune des macro-caractéristiques de BLADE et explicite les choix faits dans celles-ci par rapport aux exigences textuelles.

Sécurité - Les données stockées dans la blockchain étant simplement des métadonnées ne contenant pas d'information personnelle, elles ne sont pas considérées comme sensibles, pas plus que l'identité des tiers qui est masquée par leur adresse. Il est donc possible d'utiliser une blockchain publique (ce qui est d'ailleurs le choix initial de l'étude), sans chiffrement des données. Les permissions étant par ailleurs gérées à l'échelle du contrat intelligent, il n'est pas nécessaire d'avoir une blockchain supportant la gestion de permissions. Par déduction, ces propriétés n'étant pas importantes dans ce contexte, elles sont toutes marquées comme étant *Indifférent* dans le tableau d'entrées.

Efficience (performance) - Le système blockchain n'a pas besoin de supporter un débit minimal de transactions par seconde (que l'on différencie du nombre de transactions par seconde supportable soumises en entrée) ainsi qu'une latence particulière. Néanmoins, une latence faible pouvant être profitable à l'expérience utilisateur, nous avons tout de même choisi de la fixer à *Faiblement désirable*. Pour ce qui est de l'efficience énergétique, c'est une propriété particulièrement intéressante dans une optique de réduction de coûts, un but détaillé dans l'étude de cas. L'utilisation de blockchains publiques à algorithme de consensus lourds (tel que PoW) est très coûteux en énergie. Nous avons donc choisi la préférence *Tout à fait désirable* pour cette propriété.

Fiabilité - Les acteurs ne se faisant pas confiance entre eux, il est indispensable d'avoir un pourcentage de tolérance aux fautes byzantines, ce qui indique que le système est capable de fonctionner correctement pour un certain nombre de nœuds pouvant avoir un comportement adverse. Nous avons choisi un pourcentage d'au moins 33,3%, ce qui permet de garantir la bonne continuité du réseau blockchain pour un nombre de nœuds fautifs $f + 1 < \frac{n}{3}$, n étant le nombre de nœuds totaux constituant le réseau.

Pertinence fonctionnelle - Pour répondre aux objectifs du sujet défini, la blockchain doit être capable de prendre la forme d'un élément de stockage pour contenir les données des détaillants ainsi que de supporter l'administration de celles-ci, de facto par le biais de contrats intelligents. Ces deux attributs

Catégorie	ID	Exigence	Reliée à
Gestion des membres du consortium	1.1	Quand une candidature pour ajout au consortium est en cours, l'application off-chain doit inscrire sur la blockchain le vote d'un membre du consortium pour cette session si ce membre n'a pas encore voté et si il est authentifié via sa clé privée.	
	1.2	La blockchain doit accepter uniquement les candidatures des grossistes et détaillants affiliés à l'entreprise BigBox.	[1.1]
	1.3	Lorsqu'une majorité absolue de votes en faveur de l'acceptation du candidat est obtenue, la blockchain doit ajouter le candidat accepté à la liste des membres du consortium.	
Publication des données	2.1	Quand un détaillant membre du consortium en fait la requête, l'application off-chain doit inscrire dans la blockchain les métadonnées associées à l'état du stock à l'instant T, si les données ont été enregistrées en base de données au préalable.	[2.2]
	2.2	Quand un détaillant membre du consortium en fait la requête, l'application off-chain doit enregistrer en base de données les données associées à l'état du stock à l'instant T.	
	2.3	Chaque jour, l'application off-chain doit publier les métadonnées relatives aux informations sur les stocks de chaque détaillant faisant partie du réseau.	
Récupération des données	3.1	Quand un membre du consortium en fait la requête, l'application off-chain doit récupérer dans la blockchain les métadonnées associées à l'état d'un stock pour un détaillant à une date donnée.	
	3.2	Quand un détaillant membre du consortium en fait la requête, l'application off-chain doit récupérer les données associées à l'état d'un stock pour un détaillant à une date donnée, en utilisant les métadonnées récupérées au préalable.	[3.1]
Propriétés blockchain	4.1	Si moins de 1/3 des nœuds composant la blockchain sont défectueux, la blockchain doit traiter au moins 20 transactions simultanées sans perte de performance de plus de 5%.	
	4.2	La blockchain doit mettre en oeuvre les exigences fonctionnelles liées au moyen de la technologie des smart-contracts dits "Turing-complets".	[1.1] [1.3] [2.1] [2.3]

Tableau 4.3.: *Exigences du système blockchain souhaitées pour le cas d'étude.*

sont donc définis respectivement comme *Avancé* ainsi que *Requis*. Les autres fonctionnalités n'étant pas nécessaires, elles sont marquées *Indifférent*.

Utilisabilité - Enfin, le dernier attribut choisi est la courbe d'apprentissage : dans un contexte où la blockchain doit permettre d'économiser des coûts associés à la chaîne d'approvisionnement ainsi que de supporter une application de faible complexité, utiliser une technologie dont il est facile d'en apprendre les mécaniques peut être un atout. Nous avons choisi de le marquer *Désirable*.

Compilation des valeurs choisies

La compilation des valeurs de ces qualités système aboutit à la Table **4.4**, permettant d'exécuter le processus de décision automatisé. Ces valeurs permettent également de satisfaire les contraintes définies dans la section 3.3., la saisie dans BLADE de celles-ci est donc possible.

Attributs	Exigences	Valeur exigée	Préférences
Ouvert publiquement	Aucune		Indifférent
Permissions	Aucune		Indifférent
Encryption native des données	Aucune		Indifférent
Débit (tx/s)	Aucune		Indifférent
Latence (s)	Aucune		Faiblement désirable
Efficient en énergie	Aucune		Tout à fait désirable
Tolérant aux fautes byzantines	Requis	$\geq$33,33 %	Désirable
Contrats intelligents	Requis	Oui	Indifférent
Cryptomonnaies	Aucune		Indifférent
Élément de stockage	Requis	Avancé	Indifférent
Élément de calcul	Aucune		Indifférent
Élément gestionnaire de biens	Aucune		Indifférent
Connecteur logiciel	Aucune		Indifférent
Courbe d'apprentissage	Aucune		Désirable

Tableau 4.4.: *Exigences et préférences soumises à BLADE.*

4.3. *Résultats*

L'exécution du processus automatisé élimine l'alternative Bitcoin, car elle ne permet pas le support de contrats intelligents, ainsi que l'alternative Hyperledger Fabric, car elle ne tolère pas les fautes byzantines. Deux matrices sont obtenues, l'une contenant les poids et l'autre les alternatives possibles (resp. Ethereum-PoW, Ethereum-PoA, Corda et Tezos). Sachant qu'un poids à 0 pour un attribut donné rend celui-ci insignifiant dans le calcul du score de chaque alternative, il est possible de simplifier ces matrices pour les valeurs définies dans les eq. (1) et eq. (2).

$$W = \begin{pmatrix} 0.25 \\ 0.75 \\ 0.5 \\ 0.5 \end{pmatrix} \qquad (1) \qquad A = \begin{pmatrix} 180 & 10 & 1 & 60 \\ 0 & 1 & 1 & 1 \\ 0.5 & 0.33 & 0.33 & 0.33 \\ 0.4 & 0.4 & 0.8 & 0.8 \end{pmatrix} \qquad (2)$$

S'en suit l'exécution de l'algorithme d'aide à la décision, qui propose les résultats suivants (Table **4.5**). L'algorithme de décision considère donc l'alternative Ethereum-PoA comme étant la meilleure. En effet, son score obtenu est le plus proche de 1 (solution idéale positive) des trois alternatives.

Alternative	Score
Ethereum, PoA	0.98054669
Corda, PBFT	0.78586689
Tezos, PoS	0.22030198
Ethereum, PoW	0.21413310
Hyperledger Fabric, Raft	Disqualifiée
Bitcoin, PoW	Disqualifiée

Tableau 4.5.: *Résultat de l'exécution du processus de décision.*

4.4. *Validation de la solution proposée*

La sous-section précédente a montré via BLADE que la solution la plus adaptée pour le problème posé est Ethereum-PoA. Pour confirmer la pertinence de la solution, cette sous-section vise à évaluer la robustesse et la performance du réseau Ethereum par le biais d'un outil permettant de tester ses performances, développé dans ce sens. Cet outil, accessible en open-source [13], permet l'exécution semi-automatique de benchmarks sur une blockchain Ethereum déployée dans ce but. Afin de réaliser ce test de performance, l'outil utilise des machines du réseau Grid'5000, un banc d'essai flexible, de grande taille et configurable à souhait pour le support d'expériences de large échelle. Aussi, les machines sont dédiées à la tâche pour laquelle elles sont allouées, non partagées avec d'autres processus. Utiliser Grid'5000 permet donc une reproductibilité aisée de l'expérience proposée dans cette sous-section.

Pour la réalisation de ce test de performance, un contrat intelligent pour Ethereum a également été implémenté. Lorsque déployé sur la blockchain, il permet de faire les opérations définies dans le scénario de chaîne logistique (sauvegarde de données hachées, administration des tiers autorisés à utiliser l'application). L'outil est ensuite utilisé pour mettre en place l'infrastructure du test de performance, représentée par la fig. **3**.

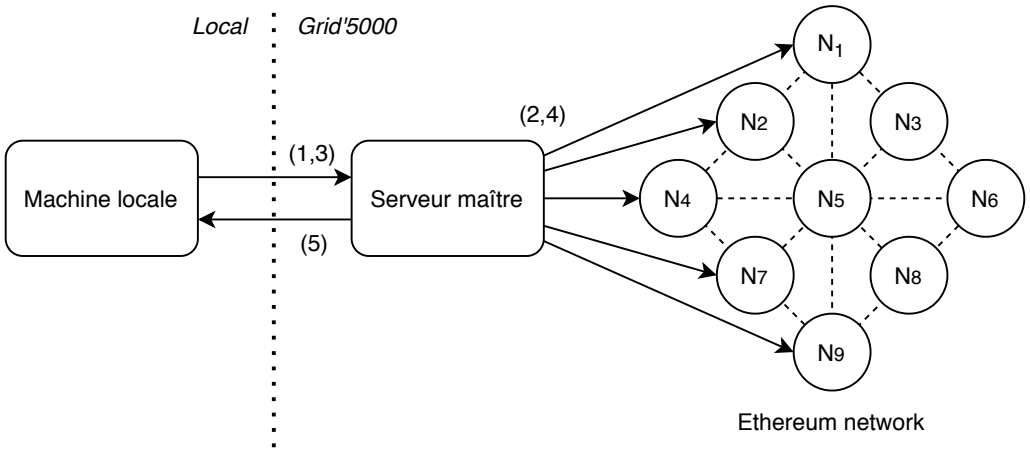

Figure 3.: *Typologie de l'infrastructure déployée pour l'exécution du test de performance.*

Trois types de machines y sont à distinguer :

— Machine locale : cette machine sert à lancer l'outil, qui va allouer les machines pour l'expérience ainsi que récupérer les résultats du test.

13. https ://github.com/nicoSix/sc-archi-gen

— Serveur maître : initialisée par l'outil, cette machine va démarrer les nœuds Ethereum-PoA en utilisant les paramètres fournis par l'outil, puis exécuter le test de performance.
— Nœud Ethereum : ces machines composent le réseau blockchain et sont à l'écoute de transactions provenant du serveur maître.

La séquence d'exécution d'un benchmark est la suivante : l'outil demande l'allocation des machines pour l'exécution des benchmarks sur le réseau Grid'5000 (1). Une fois les machines obtenues, il envoie au serveur maître la configuration du réseau blockchain. Cette configuration contient notamment l'intervalle inter-bloc, la taille des blocs produits, les informations sur les machines qui feront office de nœud, le type de benchmark (ici, l'envoi de transactions à un unique smart-contract), et le nombre de transactions envoyées par seconde à la blockchain. Le serveur maître va initialiser les machines comme défini par la configuration (2), en utilisant Salt [14], un outil permettant de configurer rapidement plusieurs clients depuis un serveur. Il va ensuite attendre le signal de la machine locale pour lancer le benchmark (3). Le serveur maître va démarrer le benchmark (4), en envoyant au réseau blockchain un grand nombre de transactions chaque seconde, ce nombre étant défini dans la configuration. Ces transactions sont envoyées à chaque nœud de manière équitable : chacun d'entre eux reçoit le même nombre de transactions chaque seconde, la somme d'entre elles étant le volume de transactions par seconde attendu pour le benchmark. Enfin, une fois le benchmark terminé, la machine locale reçoit les résultats de celui-ci (5). Elle peut aussi rester connecté en permanence au serveur maître pour voir en temps réel l'état des nœuds pendant le benchmark.

Pour ce test de performance, neuf nœuds composent le réseau Ethereum-PoA. Les nœuds possèdent chacun un processeur Intel Xeon Gold 5220 (18 cores), 96 GiB de mémoire vive, deux SSD de 480GB et 960GB respectivement, et une bande passante de 2x25 Gbps. Le serveur chargé de piloter l'expérience par l'envoi de transaction possède les mêmes caractéristiques techniques. Chacun des nœuds utilise le client d'Ethereum Geth, configuré avec l'algorithme PoA Clique [15], un intervalle de génération de bloc laissé à la valeur par défaut de 5 secondes, et une taille de bloc non limitée. Le test de performance est réalisé en plusieurs benchmarks, chaque benchmark pouvant envoyer un différent volume de transactions par seconde en entrée des nœuds. Ce nombre est compris entre 380 et 470, avec un intervalle de 10 par point de mesure. 10 benchmarks sont exécutés pour chaque point de mesure. La valeur attendue pour un point de mesure est le ratio entre le nombre de transactions ayant été acquittées, et le nombre total de transactions en entrée. Il est mesuré après l'exécution d'un benchmark, qui dure 215 secondes : 200 secondes avec envoi de transactions, puis 15 secondes de pause pour permettre l'acquittement des dernières transactions envoyées.

La fig. **4** présente les résultats de cette expérience, par le biais d'un diagramme en boites, Celui-ci est constitué de 10 boites pour chacun des volumes de transactions par seconde appliqués en entrée de l'outil de benchmark. Sur cette figure, une boite représente une série de 10 benchmarks d'un même volume de transaction par seconde appliqué en entrée. La barre rouge représente la valeur médiane de cette série.

Ces résultats montrent que le réseau blockchain arrive bien à supporter une charge de 380 transactions par seconde. Une telle infrastructure est donc amplement capable de supporter une charge composée de 20 transactions par jour (pour chacun des fournisseurs) ainsi que quelques transactions ponctuelles d'administration du consortium. Le choix d'Ethereum-PoA est donc pertinent pour le cas d'utilisation

14. https ://www.saltstack.com/
15. https ://github.com/ethereum/EIPs/issues/225

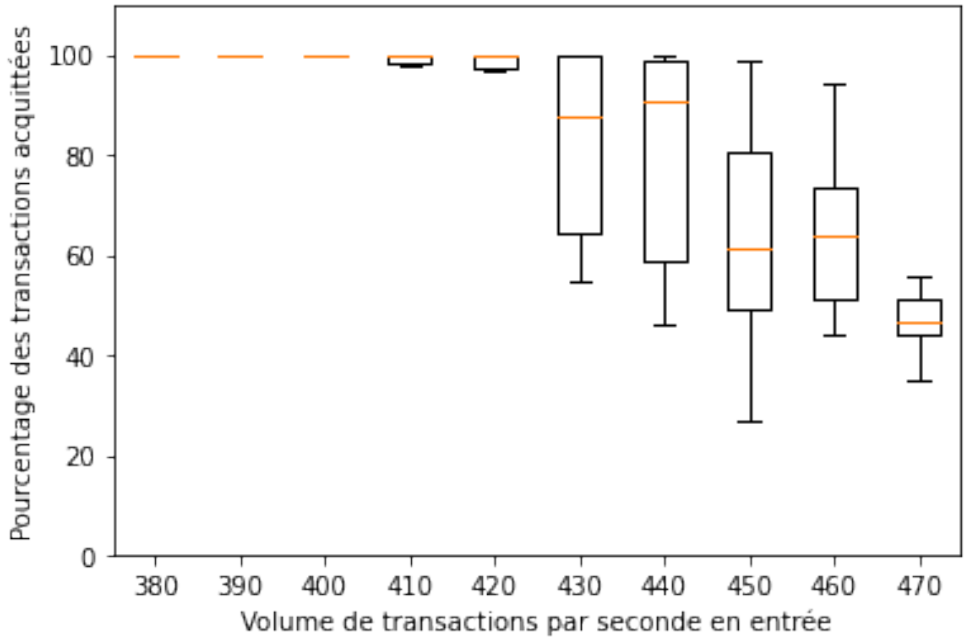

Figure 4.: *Diagramme en boîtes présentant le résultat des tests de performance Ethereum-PoA. Chaque point en abscisse représente une série de 10 benchmarks réalisés pour le nombre associé de transactions par seconde en entrée et dans la configuration d'expérience définie en section* 4.4.

donné d'un point de vue performance.

Cette figure montre également un écart-type faible pour les valeurs les moins élevées de transactions par seconde appliquées en entrée, mais également pour les valeurs les plus élevées. Au contraire, les séries pour les benchmarks réalisés sur les valeurs de 430 à 460 transactions par seconde montrent un écart-type élevé. Cela s'explique par la capacité des nœuds à tenir cette charge. En effet, dans cette tranche, les nœuds atteignent leur limite et peuvent rapidement tomber hors-service, contrairement aux autres valeurs ou les nœuds vont/ne vont forcément pas tomber hors-service.

5. Travaux liés

Cette étude s'inscrit dans la lignée des travaux réalisés pour faciliter l'adoption de la blockchain par le biais d'une aide à la décision entre différents types de blockchains, ou par la décision entre l'utilisation d'une blockchain ou non dans un contexte donné.

[Wus18] liste les principales propriétés de la blockchain (Transparence, intégrité, confiance ...) et proposent un modèle de décision sur l'adoption de la blockchain ou non en fonction de la réponse à certaines questions (telles que : "Y a-t-il plusieurs tiers impliqués ?" ou "Sont-ils de confiance ?") liées au cas d'étude donné. Ils appliquent ensuite leur modèle à plusieurs cas d'usages d'exemples. Bien qu'il y ait une étude des paramètres de la blockchain permettant de définir les questions du modèle de décision, le résultat est d'un niveau d'abstraction très élevé (blockchain publique, privée, permissionnée ou pas de blockchain). Il ne permet donc pas de prendre une décision précise sur la technologie de blockchain à utiliser ainsi que ces paramètres. Dans [Koe18], une revue de littérature est effectuée sur des études relatives aux modèles de décision pour la blockchain afin de construire un nouveau modèle à partir de celles-ci. Les résultats de ce modèle sont un peu plus précis que le précédent, mais ne donnent tou-

jours pas une recommandation précise. [Lab19] présente également un travail de revue de littérature, en utilisant une approche DSR (Design Science Research) afin de construire un nouveau modèle. Celui-ci comporte plusieurs niveaux de décision et prennent en compte des propriétés blockchain, ce qui permet à un utilisateur de faire un choix avec une précision accrue en sortie par rapport aux études précédentes.

De plus, l'étude montre les dépendances entre certains paramètres (par exemple, la confidentialité et la transparence). Cependant, les paramètres en entrée sont majoritairement spécifiques à la blockchain et conditionnent l'utilisation du modèle par un expert. Une autre étude intéressante présente une troisième approche d'aide à la décision en proposant un travail complet de détail des fondamentaux blockchain dans la première partie de leur étude, ainsi qu'un modèle de décision introduisant des critères opposés (tels que performance/coûts), mais également une série de questions pour affiner le choix ("Quand utiliser la blockchain ?", "Quoi utiliser ?", "Comment utiliser cette blockchain ?") [Bel19]. Toutes ces études permettent de guider la prise de décision pour un projet blockchain donné, mais ne permettent pas d'aller plus en détail (paramètres blockchain) à cause des limitations des modèles de décision. Le manque d'automatisation et la résolution manuelle des questions ne permettent pas de prendre en compte un grand nombre d'exigences en entrée.

Certaines études ont été réalisées afin de répondre à cette problématique. À titre d'exemple, [Tan19] propose d'utiliser une méthode d'aide à la décision multicritère appelée TOPSIS, qui est la même que celle utilisée dans cette étude, afin de déterminer la meilleure solution de blockchain publique disponible à partir d'un ensemble de critères en entrée. L'approche est intéressante dans ce contexte, mais ne permet pas de prendre en compte d'autres blockchains (privées, permissionnées). De plus, les critères techniques blockchain sont regroupés sous les critères "basic technology", "applicability" et "transaction per second", le premier étant quantifié via des experts, les recommandations données en résultat peuvent donc manquer de précision si l'on se place du point de vue de l'entreprise souhaitant démarrer son projet.

Dans [Far20], un système de prise de décision pour les technologies blockchain est implémenté, basé sur de précédents travaux pour d'autres technologies. Un sondage a été réalisé auprès d'experts pour déterminer les critères de choix les plus pertinents, puis une base de connaissance contenant les valeurs de ces attributs choisis pour un ensemble large de blockchains (obtenus avec des livres blancs, études, tests de performance ...) est construite afin de donner des recommandations via un moteur d'inférence. L'outil proposé est performance et permet de donner des recommandations précises, mais cette étude a pour but d'aller plus loin en proposant une contribution spécifiquement orientée blockchain (prise en compte de processus métiers et de modèles architecturaux spécifiques) qui soit plus accessible pour des non-experts en blockchain, par le biais d'un modèle qui lie les attributs blockchain et qualité logicielle. Ainsi, l'utilisateur peut saisir des exigences plus courantes que celles spécifiques à la technologie blockchain.

6. Discussion

La prédiction obtenue, qui est d'utiliser Ethereum-PoA, est pertinent pour plusieurs raisons. En effet, toutes les fonctionnalités que nous estimons nécessaires à la bonne implémentation du cas d'étude choisi sont présentes, tout en permettant de garantir un coût optimal de celle-ci (faible difficulté d'apprentissage et économique en énergie). Cependant, certaines limitations du processus de décision sont à prendre en compte lors de l'utilisation de l'outil. Tout d'abord, la méthode demeure sensible aux variations de poids. Si nous avions choisi un poids supérieur concernant le débit de transactions, un résultat différent en sor-

tie aurait pu être obtenu. Des études de sensibilité peuvent permettre d'établir des intervalles, servant à indiquer à quel degré un poids peut varier sans affecter le résultat final. Il existe également des méthodes, comme celle de la détermination de poids par l'entropie, permettant de limiter l'impact des critères ayant une forte entropie en diminuant leur poids [Hua08]. Il faut également prendre en compte la possibilité d'inversion de rang (aussi appelé *rank reversal*) lors de l'utilisation de TOPSIS [GC12]. En effet, même si TOPSIS est adapté aux objectifs de l'outil, qui est l'ajout aisé de nouvelles alternatives et la prise de décisions à partir d'attributs d'unités et d'échelles différentes, l'ajout de nouvelles alternatives peut entraîner un changement important dans le rang de chacune des alternatives après l'exécution du processus de décision. Ensuite, l'échelle de notation choisie pour l'expression des préférences peut entraîner un biais selon la perception des écarts entre les différentes valeurs proposées par l'utilisateur. Afin de rendre le résultat plus fiable, d'autres systèmes de pondération pourraient être considérés, comme par exemple celui de AHP. Aussi, les attributs des alternatives en base étant statiques (par exemple, le nombre de transaction par seconde ne prenant pas en compte les ressources de la machine), cela peut entraîner une incertitude quant à la fiabilité des résultats. De futurs travaux seront à réaliser pour proposer différentes valeurs aux attributs en fonction du contexte de la décision.

Pour la seconde expérience mettant en œuvre un test de performance de la blockchain Ethereum-PoA, celle-ci n'était plus capable de traiter 100% des transactions entrantes à partir de 400 transactions par seconde dans ce contexte expérimental. Le suivi de l'exécution sur chacun des nœuds montre que cette incapacité apparaît lorsque le CPU des nœuds n'est plus capable de supporter la charge de transactions reçues par le client Geth. Il est cependant possible de diminuer l'intervalle inter-blocs afin d'augmenter les performances, mais une valeur trop basse pourrait dégrader la qualité du réseau (difficulté à aboutir à un consensus entre nœuds d'autorité) et augmenter l'espace disque nécessaire (chaque bloc comportant au moins un entête de taille non nulle). Par conséquent, la valeur par défaut a été conservée, mais étudier l'impact d'une baisse sur la stabilité pourrait être profitable.

7. Conclusion et travaux futurs

Dans cette étude, un outil nommé BLADE est présenté. BLADE permet d'obtenir une recommandation sur la blockchain à utiliser étant donné un ensemble d'exigences et de préférences utilisateurs. Pour cela, un panel pertinent de blockchains ainsi que de critères relatifs à la qualité d'un système (norme ISO 25010) ont été sélectionnés pour créer une base de connaissance, puis une liste de termes permettant à un utilisateur de soumettre ses préférences et exigences quant aux critères choisis pour la décision a été choisie. Le processus de décision est ensuite présenté en détail, de l'envoi d'entrées dans celui-ci à la recommandation par le biais de TOPSIS. Une implémentation de BLADE incluant ce processus de décision est présentée. Elle permet notamment la saisie aisée des exigences au travers d'une plateforme web. Enfin, le processus de décision est validé à travers cas d'étude de gestion de chaîne logistique et montré que BLADE est capable de recommander une blockchain alignée aux besoins de l'utilisateur. Une implémentation de l'outil est disponible sur Github [16] ainsi qu'en accès direct, en ligne [17].

Cette étude est une première étape pour concevoir un processus de décision automatisé plus étendu,

16. https ://github.com/nicoSix/blade-project
17. https ://recommender.blade-blockchain.eu/

car il pourrait prendre en compte un plus grand nombre d'entrées (topologie d'architecture système, infrastructure, processus métiers...). Cela permettrait également, à l'aide de ces informations, d'exécuter un test de performance personnalisé (tel que celui présenté dans la sous-section 4.4.) pour chaque utilisateur avant même d'exécuter l'algorithme de décision, le but étant de fixer de manière extrêmement précise les valeurs des critères variants (débit de transactions, latence ...). Une autre piste d'amélioration est l'utilisation d'approches basées sur la logique floue ou les modèles bayésiens qui permettrait de tenir compte de l'aspect subjectif des critères de décision. L'un des challenges les plus importants qui devra également être adressé dans le futur, sera la mise à jour de la base de connaissances. En effet, le résultat fourni par BLADE sera pertinent si les attributs restent à jour, au travers de l'évolution des différentes blockchain proposées, des ajouts d'attributs utilisés dans l'outil, et des benchmarks réalisés qui permettront d'affiner certaines valeurs de la base de connaissances. Cela passera par une collaboration étroite avec des entreprises ainsi que des experts blockchain et architectes, mais également par la collection de traces lors de l'utilisation de l'outil.

C'est également une première étape dans la conception d'un outil couvrant une plus grande partie du développement d'applications blockchain, depuis la conception jusqu'à sa mise en production. En effet, l'un des objectifs futurs est de proposer un outil capable de recommander une blockchain ainsi qu'une architecture logicielle, à partir d'exigences, de modèles de processus business et des biens de l'entreprise (topologie d'infrastructure, expérience des collaborateurs, ...), puis de générer automatiquement des fichiers et du code permettant de rapidement mettre en oeuvre l'architecture et la blockchain. Cette approche est décrite plus en détail dans une autre étude [Six20a]. Aussi, ce futur outil pourra également proposer la génération automatique de fonctionnalités blockchain, via une approche ligne de produit pour logiciels.

Par ces travaux, un objectif serait de proposer une suite d'outils et de composants permettant d'assister l'architecte dans toutes les étapes du développement d'applications blockchain, et étendre l'état de l'art actuel sur l'intégration de la blockchain dans le domaine de l'architecture logicielle.

Remerciements

Les expériences présentées dans ce document ont été réalisées à l'aide du banc d'essai Grid'5000, soutenu par un groupe d'intérêt scientifique hébergé par INRIA et comprenant le CNRS, RENATER et plusieurs autres universités et organisations (voir https ://www.grid5000.fr).

Bibliographie

[Abe16] ABEYRATNE, S. A., MONFARED, R. P. : «Blockchain ready manufacturing supply chain using distributed ledger.» *International Journal of Research in Engineering and Technology*, *vol. 5*, n° 9, (2016), p. 1–10.

[And18] ANDROULAKI, E., BARGER, A., BORTNIKOV, V., CACHIN, C., CHRISTIDIS, K., DE CARO, A., ENYEART, D., FERRIS, C., LAVENTMAN, G., MANEVICH, Y., ET COLLAB. : «Hyperledger fabric : a distributed operating system for permissioned blockchains.» dans *Proceedings of the Thirteenth EuroSys Conference*, p. 1–15, 2018.

[Bar17] BARTOLETTI, M., POMPIANU, L. : «An analysis of Bitcoin OP_RETURN metadata.» dans *International Conference on Financial Cryptography and Data Security*, p. 218–230, 2017.

[Bas03] BASS, L., CLEMENTS, P., KAZMAN, R. : *Software architecture in practice*, Addison-Wesley Professional, 2003.

[Bel19] BELOTTI, M., BOZIC, N., PUJOLLE, G., SECCI, S. : «A Vademecum on Blockchain Technologies : When, Which and How.» *IEEE Communications Surveys & Tutorials*, *vol. 21*, n° 4, (2019), p. 3796–3838.

[Bro16] BROWN, R. G., CARLYLE, J., GRIGG, I., HEARN, M. : «Corda : an introduction.» *R3 CEV, August*, *vol. 1*, (2016), p. 1–15.

[Ekb16] EKBLAW, A., AZARIA, A., HALAMKA, J. D., LIPPMAN, A. : «A Case Study for Blockchain in Healthcare :"MedRec" prototype for electronic health records and medical research data.» dans *Proceedings of IEEE Open & Big Data Conference*, vol. 13, p. 1–13, 2016.

[Far20] FARSHIDI, S., JANSEN, S., ESPAÑA, S., VERKLEIJ, J. : «Decision Support for Blockchain Platform Selection : Three Industry Case Studies.» *IEEE Transactions on Engineering Management*, p. 1–20.

[GC12] GARCÍA-CASCALES, M. S., LAMATA, M. T. : «On rank reversal and TOPSIS method.» *Mathematical and Computer Modelling*, *vol. 56*, n° 5-6, (2012), p. 123–132.

[Her17] HERBAUT, N., NEGRU, D. : «A model for collaborative blockchain-based video delivery relying on advanced network services chains.» *IEEE Communications Magazine*, *vol. 55*, n° 9, (2017), p. 70–76.

[Hua08] HUANG, J. : «Combining entropy weight and TOPSIS method for information system selection.» dans *2008 IEEE Conference on Cybernetics and Intelligent Systems*, p. 1281–1284, 2008.

[Hyv17] HYVÄRINEN, H., RISIUS, M., FRIIS, G. : «A blockchain-based approach towards overcoming financial fraud in public sector services.» *Business & Information Systems Engineering*, *vol. 59*, n° 6, (2017), p. 441–456.

[Koe18] KOENS, T., POLL, E. : «What blockchain alternative do you need ?» dans *Data Privacy Management, Cryptocurrencies and Blockchain Technology*, p. 113–129, Springer, 2018.

[Kor07] KORNYSHOVA, E., SALINESI, C. : «MCDM techniques selection approaches : state of the art.» dans *2007 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making*, p. 22–29, 2007.

[Lab19] LABAZOVA, O. : «Towards a Framework for Evaluation of Blockchain Implementations.» *ICIS 2019 Proceedings*, p. 1–10.

[Lai94] LAI, Y.-J., LIU, T.-Y., HWANG, C.-L. : «Topsis for MODM.» *European journal of operational research*, *vol. 76*, n° 3, (1994), p. 486–500.

[Lon19] LONGO, F., NICOLETTI, L., PADOVANO, A., D'ATRI, G., FORTE, M. : «Blockchain-enabled supply chain : An experimental study.» *Computers & Industrial Engineering*, *vol. 136*, (2019), p. 57–69.

[Nak08] NAKAMOTO, S. : «Bitcoin : A peer-to-peer electronic cash system.» 2008.

[Pod09] PODVEZKO, V., ET COLLAB. : «Application of AHP technique.» *Journal of Business Economics and Management*, , n° 2, (2009), p. 181–189.

[Pre20] PREWETT, K. W., PRESCOTT, G. L., PHILLIPS, K. : «Blockchain adoption is inevitable—Barriers and risks remain.» *Journal of Corporate Accounting & Finance*, *vol. 31*, nº 2, (2020), p. 21–28.

[Six20a] SIX, N. : «Decision Process for Blockchain Architectures based on Requirements.» dans *CAiSE (Doctoral Consortium)*, p. 53–61, 2020.

[Six20b] SIX, N., HERBAUT, N., SALINESI, C. : «Quelle Blockchain choisir ? Un outil d'aide à la décision pour guider le choix de technologie Blockchain.» *INFORSID 2020*, p. 135–150.

[Sza97] SZABO, N. : «Formalizing and Securing Relationships on Public Networks.» *First Monday*, *vol. 2*, nº 9.

[Tak18] TAKEMIYA, M., VANIEIEV, B. : «Sora identity : secure, digital identity on the blockchain.» dans *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, vol. 2, p. 582–587, 2018.

[Tan19] TANG, H., SHI, Y., DONG, P. : «Public blockchain evaluation using entropy and TOPSIS.» *Expert Systems with Applications*, *vol. 117*, (2019), p. 204–210.

[Tap16] TAPSCOTT, D. : *Blockchain Revolution : How the Technology Behind Bitcoin Is Changing Money, Business, and the World*, Portfolio, 2016.

[Woo14] WOOD, G., ET COLLAB. : «Ethereum : A secure decentralised generalised transaction ledger.» *Ethereum project yellow paper*, p. 1–32.

[Wus18] WUST, K., GERVAIS, A. : «Do you need a blockchain ?» *Proceedings - 2018 Crypto Valley Conference on Blockchain Technology, CVCBT 2018*, p. 45–54.

[Xu18] XU, X., PAUTASSO, C., ZHU, L., LU, Q., WEBER, I. : «A pattern collection for blockchain-based applications.» dans *Proceedings of the 23rd European Conference on Pattern Languages of Programs*, p. 1–20, 2018.

[Zwi16] ZWICKER, W. S. : «Introduction to the Theory of Voting.» 2016.