De l'expérience développeur

Compte-rendu d'une première exploration

Antoine Aubé

Stack Labs

16 janvier 2025

Avant-propos : point lexical

Expérience développeur

Developer Experience

DX

Avant-propos (bis) : d'où je parle ?

1. Début 2024 : découverte dans un audit
2. Été 2024 : documentation pendant l'intermission
3. Décembre 2024 : colloque au sujet de la *DX* (...)

Avant-propos (ter) : version longue

https://arjca.fr/compilations/dx.gmi

Les corrections, commentaires et compléments sont les bienvenus !

Plan de la présentation

1. Qu'est-ce que la *DX* ?
2. Pourquoi se préoccuper de la *DX* ?
3. Comment améliorer la *DX* ?
4. Conclusion : et maintenant.. ?

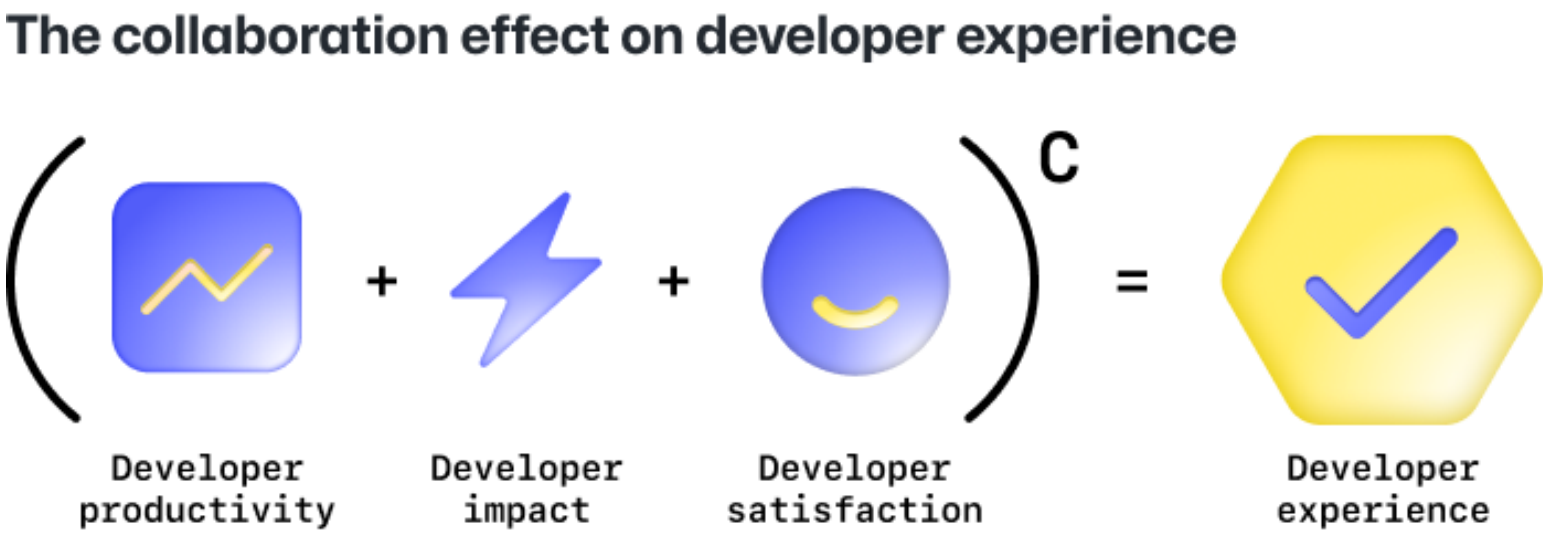
Plan de la présentation

1. Qu'est-ce que la *DX* ?
2. Pourquoi se préoccuper de la *DX* ?
3. Comment améliorer la *DX* ?
4. Conclusion : et maintenant.. ?

En quête d'une définition dans l'industrie

Constat : *Business-driven definition*, emphase sur la productivité

Exemple (abrégé) de GitHub [Dav23] :



Métriques-clés proposées sont ceux de *DORA* pour *DevOps* : fréquence de déploiement, temps de déploiement d'un changement, temps de récupération suite à une panne, taux de pannes [FHK18]

En quête d'une définition académique

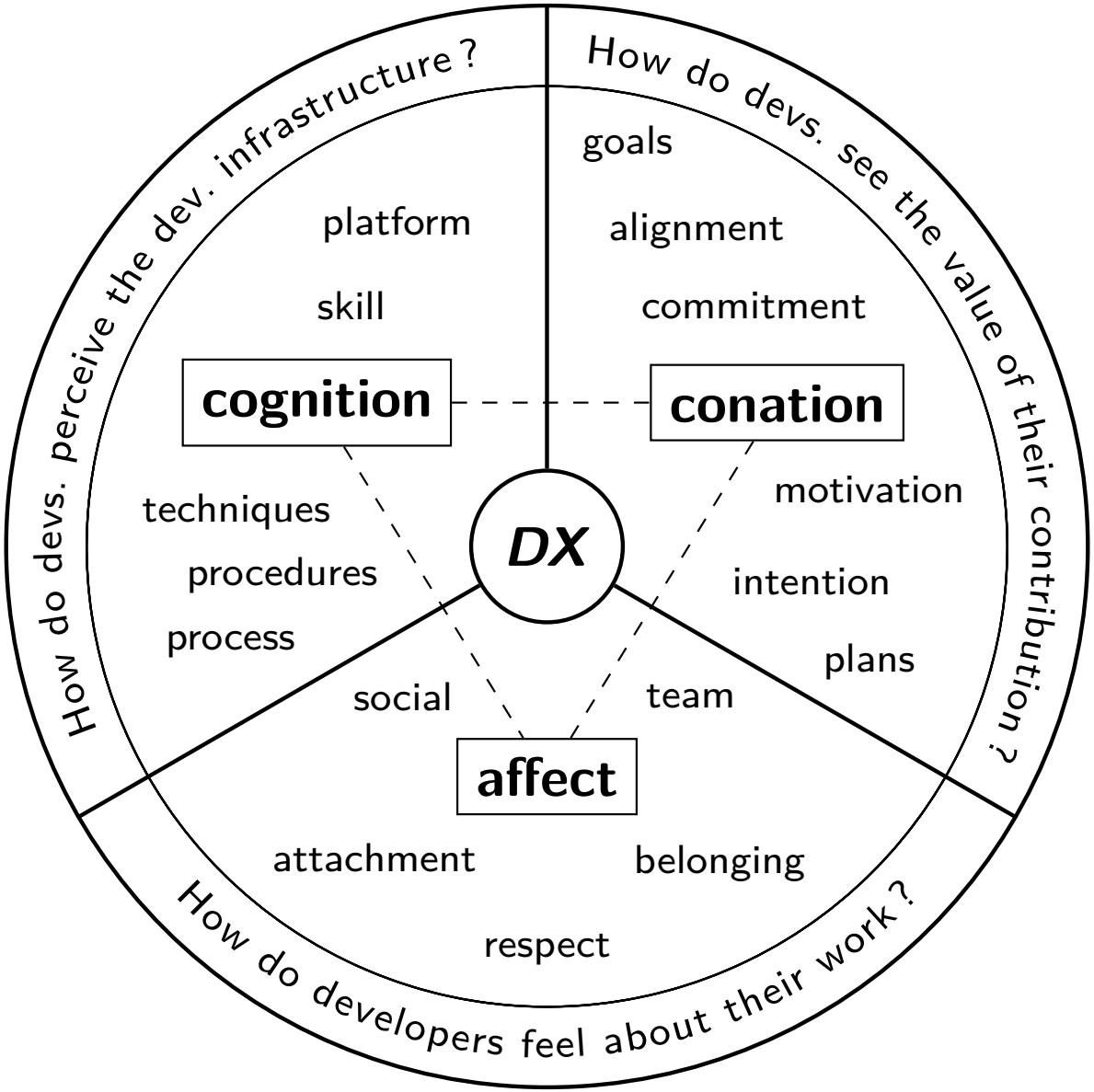
Nombreuses références au travail de Fagerholm et Münch [Nyl20]

Définition (Expérience développeur [FM12])

L'expérience développeur est l'ensemble des expériences en lien avec les activités et les artefacts qu'un développeur rencontre lors de son implication dans le développement d'un logiciel.

- **Développeur** : n'importe qui impliqué dans l'activité de développement d'un logiciel (programmeur, testeur, architecte, ...)
- **Expérience** : dans le sens de « *faire l'expérience de* »

Modèle tridimensionnel de Fag. et Münch [FM12]



DX et Productivité ?

En tout cas, pas dans sa définition...

Plan de la présentation

1. Qu'est-ce que la *DX* ?
2. Pourquoi se préoccuper de la *DX* ?
3. Comment améliorer la *DX* ?
4. Conclusion : et maintenant.. ?

Une très bonne question

Pour les développeurs :

Une meilleure $DX \approx$ de meilleures conditions de travail.

Pour la direction :

Une meilleure $DX =$ un coût.

Mais pour quel gain ?

Une (vraiment) très bonne question

Formulé autrement :

Que gagne-t-on à avoir des développeurs heureux, motivés et en maîtrise de leur environnement technique ?

Une question bien documentée

De nombreux travaux répondent déjà à cette question.
Par exemple :

- Murphy-Hill et coll. [MJS+21] : étude des facteurs prédisant une haute productivité. Principaux résultats catégorisables : affects positifs, motivation, un environnement technique adéquat
- Khan et coll. [KBH11] : corrélation entre l'humeur et la capacité à déboguer des programmes
- Barros et coll. [BTV24] : étude des facteurs humains critiques pour le succès des projets Agile. La sécurité psychologique est un facteur indirect de succès significatif
- ...

Graziotin et coll. : que font les dév. malheureux ?

À contrepied : **que perd-t-on à avoir une mauvaise *DX* ?**

- Une étude auprès de plus de 1300 développeurs
- Objectif de l'étude : déterminer pourquoi les développeurs malheureux le sont, et quelles en sont les conséquences sur leur travail

Quelques résultats [GFWA18] :

Conséquences internes	**Conséquences externes**
Mauvaises performances cognitives	Mauvaise productivité
Anxiété et désordres mentaux	Délais accrus
Motivation détériorée	Procédures moins bien suivies
Mise en retrait du travail	Interruptions fréquentes de l'état de *flow*
	Mauvaise qualité de code
	Destruction de code

Plan de la présentation

1. Qu'est-ce que la *DX* ?
2. Pourquoi se préoccuper de la *DX* ?
3. Comment améliorer la *DX* ?
4. Conclusion : et maintenant.. ?

Contexte de la *DX*

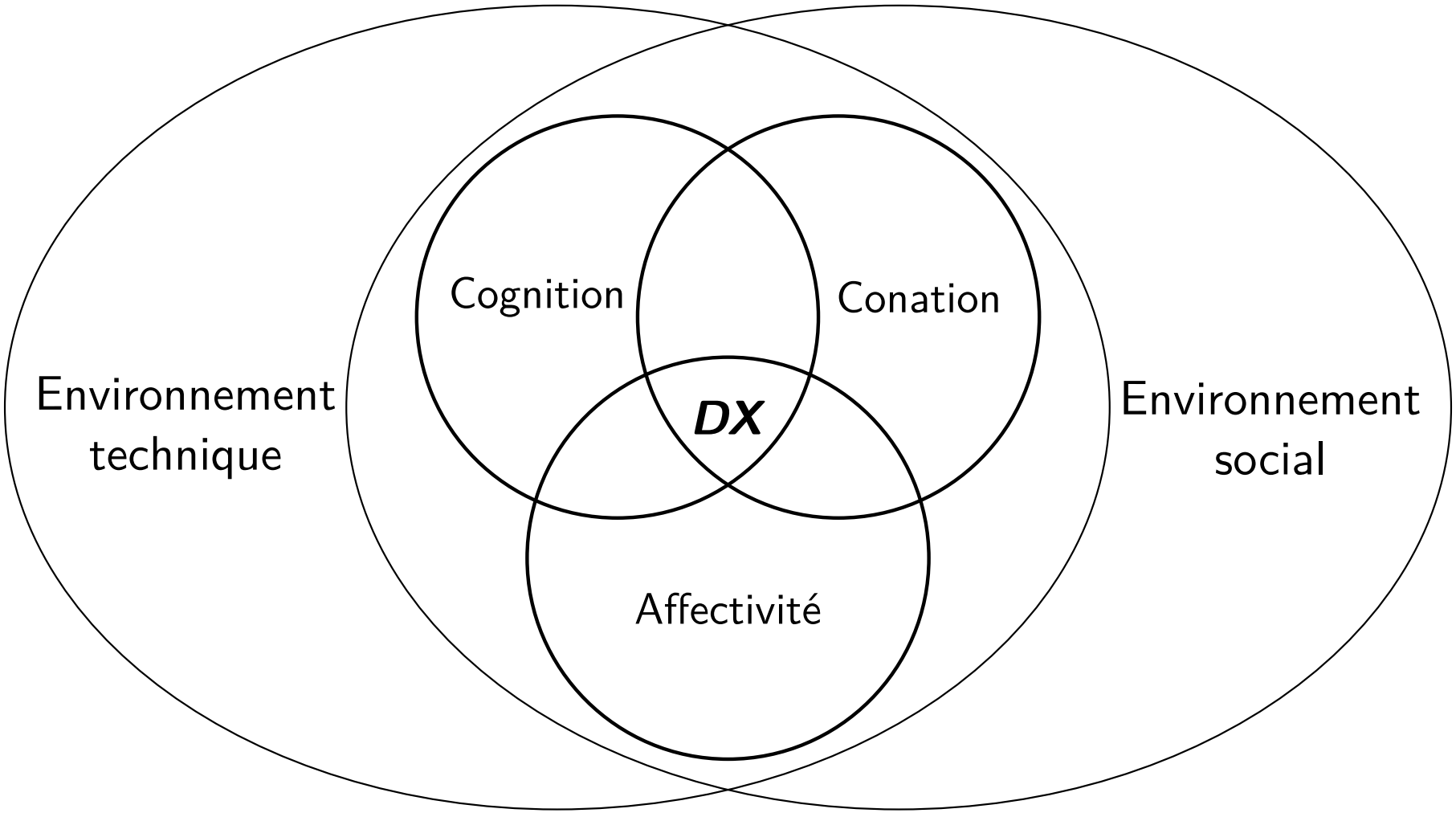
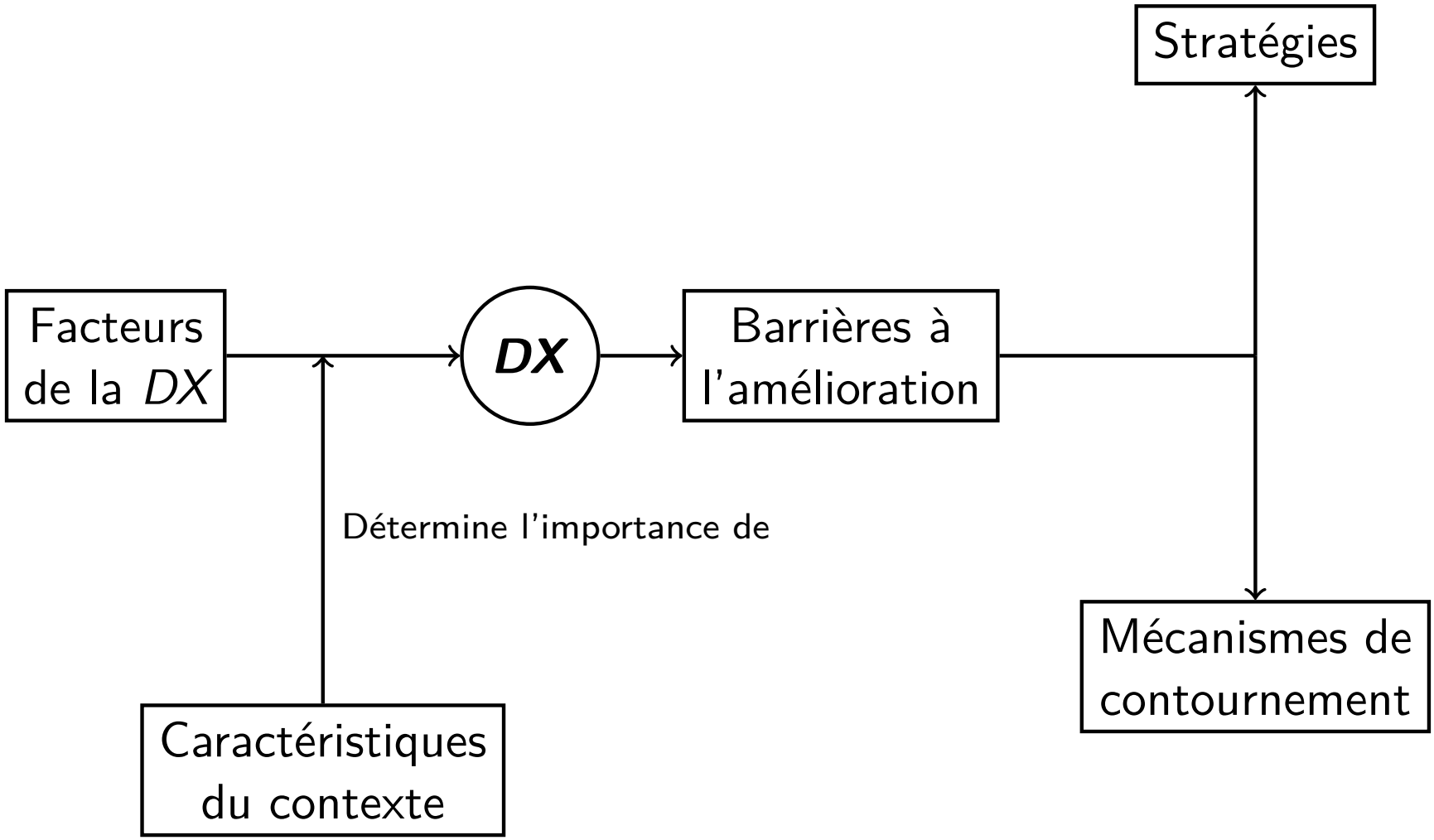
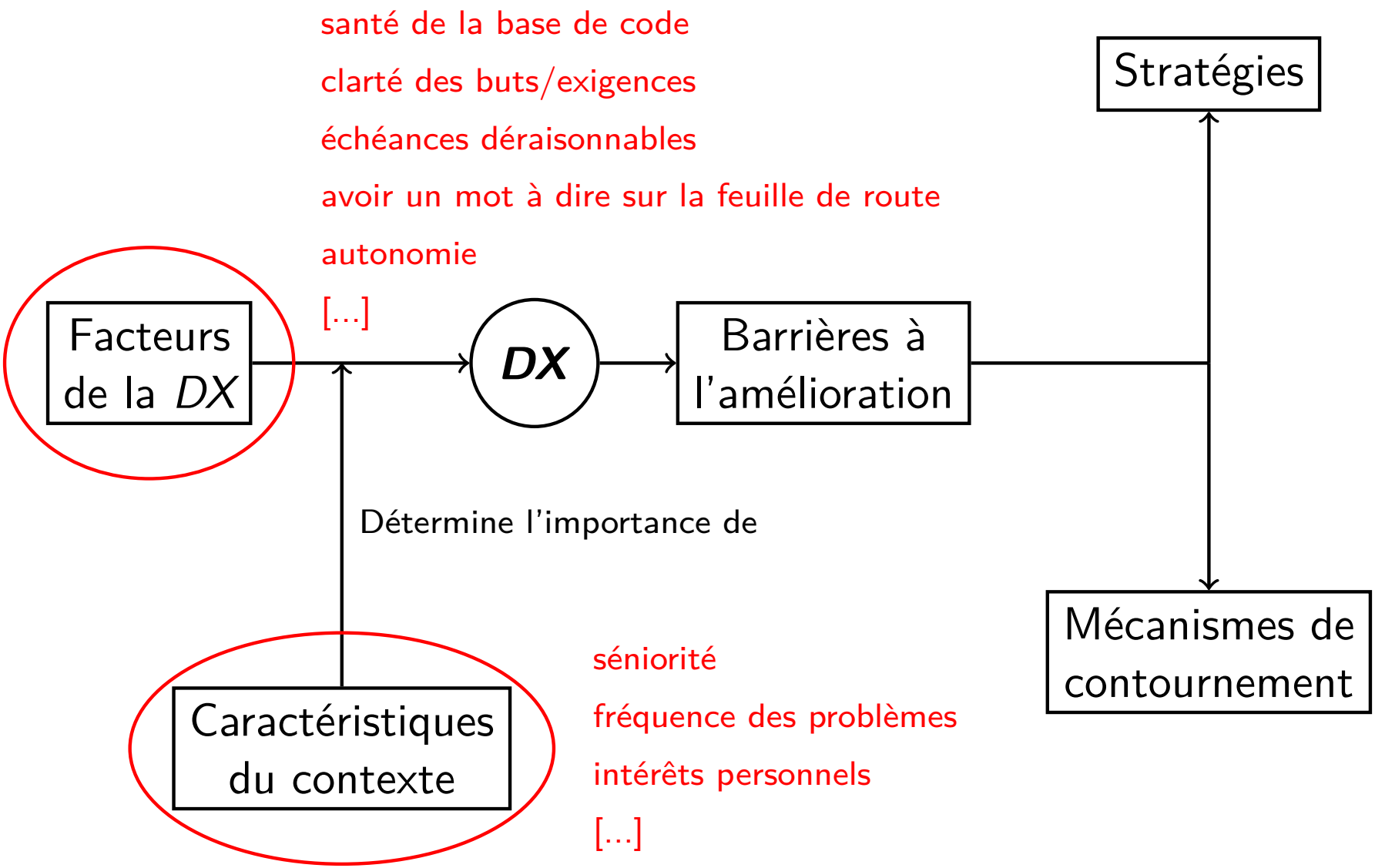


Figure tirée de la thèse de F. Fagerholm [Fag15].

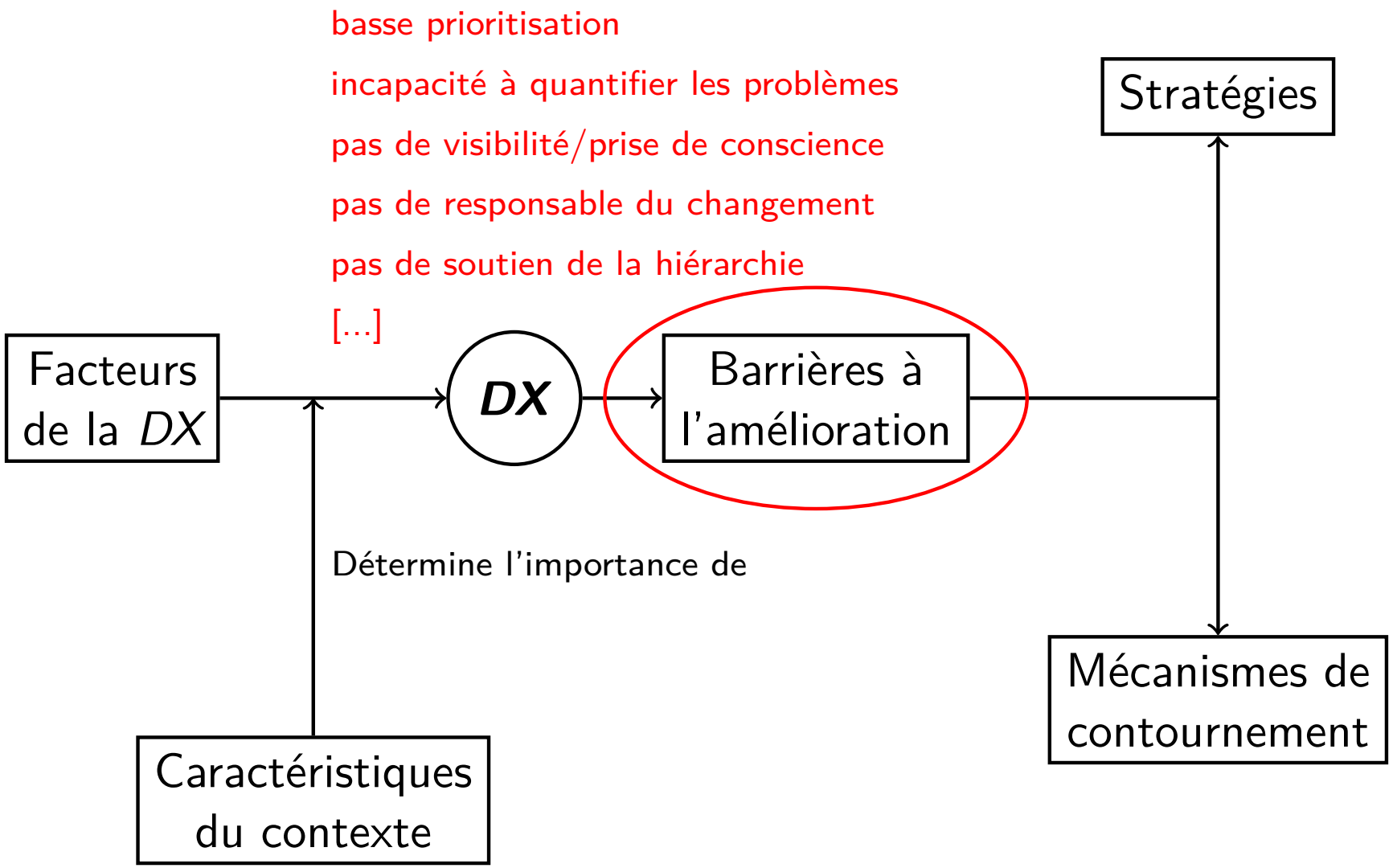
Comprendre pour améliorer la *DX* [GSN22]



Comprendre pour améliorer la *DX* [GSN22]



Comprendre pour améliorer la *DX* [GSN22]



Piste n°1 : Mise à jour culturelle

Objectifs :

- **sécurité psychologique**
- **autonomie**
- **sentiment de soutien, cohésion dans l'équipe**
- **collaboration**

Suggestions [Sch, Pla22, Cir22] :

- collecte active des problèmes et transparence sur l'avancement de leur résolution
- inclure les développeurs aux discussions sur la feuille de route
- encourager les comportements non-toxiques [SKK+18]

Piste n°2 : maintien du flow

Il faut protéger le flow [Vil23] :

$$\begin{aligned} \text{flow} &= \text{concentration intense} \\ &= \text{très productif} \end{aligned}$$

De plus :

interrompre le flow = **frustration**

Suggestions [Sch] :

- communication asynchrone
- sanctuariser des morceaux de la semaine
- moins de réunions
- réunions moins longues

Piste n°3 : *Platform Engineering*

Ça consiste à [Rob24] :

- Mettre en place une *IDP*
- Développer l'*IDP* comme un produit interne
- Séparer les équipes Produit de l'équipe Plateforme

Objectifs :

- Standardiser les pratiques en lien avec le *cloud*
- **Réduire la charge cognitive des développeurs**

Plein de solutions techniques : Humanitec, Qovery, OpsLevel, Coherence, Mia Platform, Portainer, Appvia, Argonaut, Nullstone, Mogenius, Port, ...

Piste n°4 : l'inénarrable intelligence artificielle

Divers services rendus pour **réduire la charge cognitive** [DM23] :

- autocomplétion
- aide à la compréhension
- génération automatique : tests, code, documentation, commentaires
- revues de code
- ...

À nouveau, plein de solutions techniques : Copilot, Adrenaline, Qodo, Tabnine, Sourcegraph Cody, Mintlify, Readable, Stepsize, Codeball, Planar, WhatTheDiff, ...

Plan de la présentation

1. Qu'est-ce que la *DX* ?
2. Pourquoi se préoccuper de la *DX* ?
3. Comment améliorer la *DX* ?
4. Conclusion : et maintenant.. ?

Récapitulons...

- La *DX* est l'ensemble des pensées qui nous viennent lorsque nous développons : affects, valeurs, cognition
- Plein d'avantages à avoir une bonne *DX*, plein d'inconvénients à en avoir une mauvaise
- Il y a des travaux pour identifier ce qui influe sur la *DX*
- Il y a plein de solutions **techniques** et **organisationnelles**

Quelques réserves :

- Beaucoup de plateformes propriétaires : quelle longévité ?
- Comment mesurer la *DX* ?
- Comment savoir si les solutions existantes fonctionnent ?

Mon programme : le plein-emploi

Moult thèmes à explorer

- *Onboarding*
- Plateformes de développement (*IDP*, *Platform Engineering*)
- Prise de décision technique
- Gestion de projet
- *IDE*
- Ludification
- *IA* générative (...)
- Contextes spéciaux
- ...

Plein de manières de contribuer

- Recenser l'existant, partager des interrogations et des *REx*
- Publications : lire, synthétiser
- Outils et pratiques : essayer, évaluer, comparer
- Partager via des écrits, des interventions en Stack Day, ...

Ensuite : innovation. Identifier les manques, proposer des solutions, expérimenter.

(Peut-être) bientôt : GT DevX du GDR GPL

(*Groupe de Travail sur l'Expérience Développeur du Groupement De Recherche en Génie de la Programmation et du Logiciel*)

- Initié au sein du groupe de travail dédié au Déboguage
- Journée de lancement le 4 décembre 2024, dédiée à un cadrage des défis du futur groupe de travail
  - Matinée : présentations (essentiellement orientées Déboguage)
  - Après-midi : idéation
- Pour Stack Labs : collaborations potentielles avec des laboratoires pour expérimenter des moyens d'évaluer ou d'améliorer la *DX*, ...

Conditionnel : dossier pour la création du *GT* étudié début 2025.

Bibliographie I


Leonor Barros, Carlos Tam, and João Varajão, *Agile software development projects-unveiling the human-related critical success factors*, Inf. Softw. Technol. **170** (2024), 107432.


Alex Circei, *Top 9 solutions to improve dx – developer experience*, 2022.


Gwen Davis, *Developer experience : What is it and why should you care ?*, 2023.


Ruth Dillon-Mansfield, *4 ways to improve developer experience with ai*, 2023.


Fabian Fagerholm, *Software developer experience : Case studies in lean-agile and open source environments*.


Nicole Forsgren, Jez Humble, and Gene Kim, *Accelerate : The science of lean software and devops : Building and scaling high performing technology organizations*, IT Revolution, 2018.


Fabian Fagerholm and Jürgen Münch, *Developer experience : Concept and definition*, 2012 International Conference on Software and System Process, ICSSP 2012, Zurich, Switzerland, June 2-3, 2012 (D. Ross Jeffery, David Raffo, Ove Armbrust, and LiGuo Huang, eds.), IEEE Computer Society, 2012, pp. 73–77.

Bibliographie II


Daniel Graziotin, Fabian Fagerholm, Xiaofeng Wang, and Pekka Abrahamsson, *What happens when software developers are (un)happy*, Journal of Systems and Software **140** (2018), 32–47 (en).


Michaela Greiler, Margaret-Anne D. Storey, and Abi Noda, *An actionable framework for understanding and improving developer experience*, CoRR **abs/2205.06352** (2022).


Iftikhar Ahmed Khan, Willem-Paul Brinkman, and Robert M Hierons, *Do moods affect programmers' debug performance ?*, Cognition, Technology & Work **13** (2011), 245–258.


Emerson R. Murphy-Hill, Ciera Jaspan, Caitlin Sadowski, David C. Shepherd, Michael Phillips, Collin Winter, Andrea Knight, Edward K. Smith, and Matthew Jorde, *What predicts software developers' productivity ?*, IEEE Trans. Software Eng. **47** (2021), no. 3, 582–594.


Anders Nylund, *A multivocal literature review on developer experience*.


Mia Platform, *Team's developer experience : why it urgently needs improving*, 2022.


Stéphane Robert, *Qu'est-ce que le platform engineering*, 2024.


Carlos Schults, *How to improve developer experience : 7 things to change*.

Bibliographie III

Kurt Schneider, Jil Klünder, Fabian Kortum, Lisa Handke, Julia Straube, and Simone Kauffeld, *Positive affect through interactions in meetings : The role of proactive and supportive statements*, J. Syst. Softw. **143** (2018), 59–70.

Fernando Villalba, *What is developer experience (devex) ?*, 2023.

Avez-vous des **questions** ?

- Contact : `courriel@arjca.fr`
- Diapositives : `https://arjca.fr/diapositives/2025-01-16-dx.pdf`
- Version longue : `https://arjca.fr/gemlog/2025-01-16-dx-revue-web.gmi`