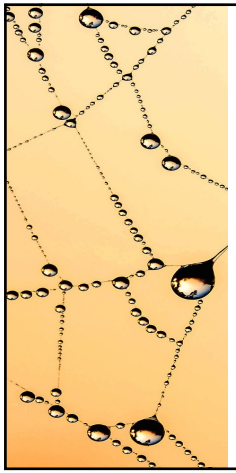


Développement informatique agile et conception éducative continuée dans l'usage :

comment articuler cette dépendance réciproque dans la conception d'une plateforme éducative ?

Marcel Grosjean, Nicolas Perrin, David Piot et Alexandre Fetelian

AUPTIC 2022

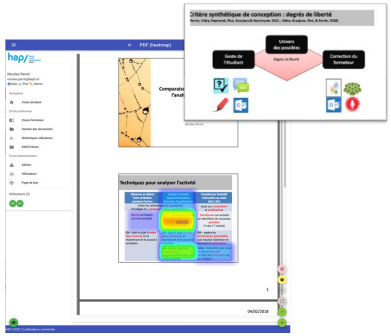
1

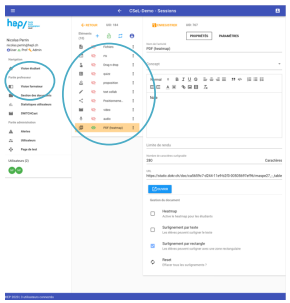
Relation entre numérique et pédagogique : une hypothèse

- **Symposium :**
  - "Développer l'accès au numérique sans perdre de vue les finalités pédagogiques communes. Possible ?"
  - "On dit parfois que la fin justifie les moyens, mais dans le cadre de l'intégration du numérique, nous vivons parfois l'inverse."
- **Notre hypothèse :**
  - Il y a une définition intrinsèque entre les fins et les moyens (Dewey, 1967)
  - Le pédagogique et le numérique sont dans une relation de co-définition
  - Cela se traduit potentiellement par des méthodes agiles/itératives... Mais cela ne va pas de soi !
- **Notre démarche : dialogue entre design informatique et "pédagogique"**

2

DOKR : évolution de la conception

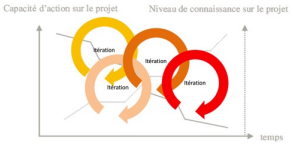




3

Démarche de conception dans une «entrée activité»
(Béguin, 2013 ; Hatchuel, 1996 ; Midler, 1996 ; Sanchez, 2015)

- **Trois perspectives : cristallisation (anticipation + prescription), plasticité (laisser une liberté résiduelle), développement (de l'outil et de l'activité)**
- **Hypothèse (concepteur) – apprentissage/réponse (opérateur) – apprentissage (concepteur) = apprentissage croisé et réduction progressive des marges de manœuvre**
- **Lors des itérations, ce n'est pas uniquement le design qui est modifié mais aussi les hypothèses de conception qui sont (in)validées, précisées...**
- **Des démarches d'ingénierie concourante et la production d'objets intermédiaires permet de contrôler l'hétérogénéité des contraintes**



4

Méthode agile
(Johnson & Ekstedt, 2016 ; Venters et al., 2018)

- **En informatique, le développeur définit l'architecture du programme en définissant l'unité de base du programme.**
  - L'architecture du programme définit la communication entre les parties de celui-ci.
  - Elle permet de poser des fondations stables permettant une maintenance et une évolution efficiente durant tout le cycle de vie du projet.
- **Dans le cas contraire, cela se traduit par une complexité accidentelle mettant à l'épreuve la disponibilité, la résilience et la durabilité (Venters et al., 2018).**
  - Les différentes briques logicielles ajoutées au fur et à mesure des développements donnent lieu à de nombreuses dettes techniques
  - Les décisions d'architectures sauvages aboutissent à des problèmes de forts couplages entre composants et d'une cohésion questionnable.
- **La démarche consiste à favoriser une co-définition en amont du cahier des charges**

5

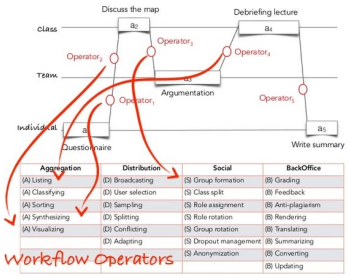
Un processus de conception mal maîtrisé

- **Constat : deux stratégies différentes de développement itératif**
  - Informatique : les méthode agiles ne modifient pas les hypothèses de conception ; elles restreignent les transformations à la marge
  - Formation : le développement continué dans l'usage consiste à définir progressivement les hypothèses de conception ; les transformations sont structurelles
- **Analogie : une brique LEGO™ traditionnelle ne permet pas des assemblages à 45°. Ajouter cette contrainte implique de redesigner la brique élémentaire et l'architecture de la construction**

6

Une relation de co-définition est pourtant nécessaire...

- **Stabiliser une l'échange d'information entre les briques (autour du heatmap)**
- **Laisser ouvert le processus itératif à l'intérieur des briques = fonctionnalités particulières (heatmap)**
- **S'inspirer des graphes d'orchestration (Dillenbourg, 2015)**
  - La décomposition nous aide à expliciter (et pas forcément à "procéduraliser")
  - Des principes de conception qui articulent matériel et numérique



7

Références

Béguin, P. (2013). La conception des instruments comme processus dialogique d'apprentissages mutuels. In *Ergonomie constructive* (pp. 147-160). Presses Universitaires de France.

Dewey, J. (1967). *Logique : la théorie de l'enquête.* PUF.

Dillenbourg, P. (2019). *Instructional Design with Orchestration Graphs.* EPFL. https://tube.switch.ch/channels/fd7f2f13

Dillenbourg, P. (2015). *Orchestration graphs. Modeling Scalable Education.* EPFL press.

Hatchuel, A. (1996). Coopération et conception collective. Variété et crises des rapports de prescription. In G. de Terssac & E. Friedberg (Eds.), *Coopération et conception* (pp. 101-121). Octarès.

Johnson, P., & Ekstedt, M. (2016). The Tarpit–A general theory of software engineering. *Information and Software Technology, 70*, 181-203.

Midler, C. (1996). Modèles gestionnaires et régulation économiques de la conception. In G. de Terssac & E. Friedberg (Eds.), *Coopération et conception* (pp. 63-85). Octarès.

Perrin, N., Uldry, S., Raymond, C., Piot, D., Grosjean, M., & Deschryver, N. (2021, 17-19 novembre). Transformer les formats pédagogiques instrumentés : pertinence de recourir aux « degrés de liberté » comme critère de conception ? AUPTIC.education 2021, Sierre.

Sanchez, É., & Monod-Ansaldi, R. (2015). Recherche collaborative orientée par la conception. Un paradigme méthodologique pour prendre en compte la complexité des situations d'enseignement-apprentissage. *Éducation et didactique, 9*(2), 73-94.

Uldry, S., Grosjean, M., Piot, D., & Perrin, N. (2020, 11-13 novembre). Travailler et réguler la compréhension en grand groupe : conception d'une plateforme éducative en s'appuyant sur le concept de degrés de liberté. AUPTIC .education 2020, Louvain.

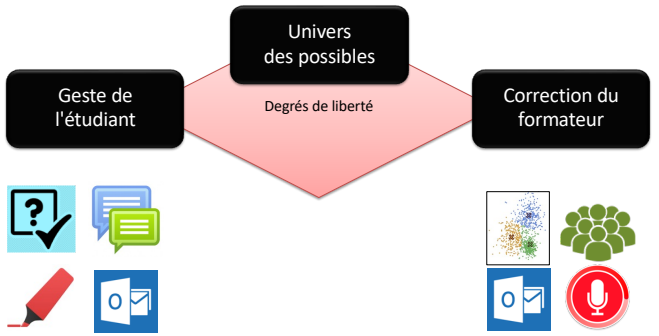
Venters, C. C., Capilla, R., Betz, S., Penzenstadler, B., Crick, T., Crouch, S., Nakagawa, E. Y., Becker, C., & Carrillo, C. (2018). Software sustainability: Research and practice from a software architecture viewpoint. *Journal of Systems and Software, 138*, 174-188.

8

9

Critère synthétique de conception : degrés de liberté

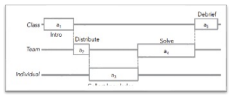
(Perrin, Uldry, Raymond, Piot, Grosjean & Deschryver, 2021 ; Uldry, Grosjean, Piot, & Perrin, 2020)



10

Le graphe d'orchestration

(Dillenbourg, 2015, 2019)



- **■ Temps ■ niveaux (individuel, équipe, classe) ■ activités ■ tests ■ connecteurs (dépendance entre a1 et a2 = rôle pédagogique, poids, élasticité…) ■ compétences (plus ou moins décomposées) ■ les interfaces tangibles ■ les contrôleurs (conditions, répétition, parallélisme, les rôles)**
  - **Connecteurs (= justification pédagogique)** : ■ rôle (prérequis, zpd, motivation, info…) ■ groupement (agrégation tels que chunks, décomposé, sélectionné f() critères, étendu à x cas, contrasté…) ■ translation (procéduralisation, formalisation, modification du format, point de vue différent, identifier les erreurs, enseigner) ■ généralisation (induction, déduction, réflexion sur la base expérience, analogie, transfert forme/fond, exceptions, synthèse via sélection ou relation
  - **Opérateurs (= utiliser données créées par ETU de a_1 à a_2)** : ■ agrégation des traces (lister, classifier, ordonner, synthétiser, visualiser yc sa position +/- anonyme) ■ distribuer des datas (tâche/data ; partager ; confronter, adapter au style d'apprentissage…) ■ social (taille du groupe, homogénéité, spliting, assigner un rôle, changer de rôle, rotation des groupes, groupes de besoin) ■ différentiation (niveau, types de connaissance, background, opinion, géographie et temporel ; anonymiser) ■ back-office (attribuer un grade, donner feed-back, vérifier plagiat, traduire, maj, convertir cm/inch, résumé automatique, traiter une image…) ■ pattern = créer des conflits (exemple : jigsaw)

11

Connecteurs et opérateurs

Preparation	Set	Translation	Generalization
(P) Prerequisite	(S+) Aggregation	(T) Proceduralization	(G+) Induction
(P) ZPD	(S+) Expansion	(T) Elicitation	(G+) Deduction
(P) Adv. organizer	(S–) Decomposition	(T) Alternate	(G+) Extraction
(P) Motivation	(S–) Selection	(T) Reframe	(G+) Synthesis
(P) Anticipation	(S=) Juxtaposition	(T) Reverse	(G=) Analogy
(P) Logistics	(S=) Contrast	(T) Repair	(G=) Transfer
(P) Data collection	(S=) Identity	(T) Teach	(G–) Restriction

Connecteurs
(= justification pédagogique)

Aggregation	Distribution	Social	BackOffice
(A) Listing	(D) Broadcasting	(S) Group formation	(B) Grading
(A) Classifying	(D) User selection	(S) Class split	(B) Feedback
(A) Sorting	(D) Sampling	(S) Role assignment	(B) Anti-plagiarism
(A) Synthesizing	(D) Splitting	(S) Role rotation	(B) Rendering
(A) Visualizing	(D) Conflicting	(S) Group rotation	(B) Translating
	(D) Adapting	(S) Dropout management	(B) Summarizing
		(S) Anonymization	(B) Converting
			(B) Updating

Opérateurs
(= utiliser données créées par ETU de a_1 à a_2)

12