# ÉCRITURE ET EXÉCUTION RÉPARTIE DE SCÉNARIOS INTERACTIFS

Auteur 1
Organisme
Adresse électronique

Auteur 2
Organisme
Adresse électronique

Auteur 3
Organisme
Adresse électronique

Résumé

Le résumé doit être placé en haut de la colonne gauche et doit contenir entre 150 et 200 mots.

## 1. INTRODUCTION

### 1.1. Contexte

#### 1.1.1. Horloges

#### 1.1.2. Synchronisation de médias

### 1.2. État de l'art

#### 1.2.1. Rappel du modèle d'i-score

#### 1.2.2. Répartition à l'édition

#### 1.2.3. Répartition à l'exécution

#### 1.2.4. Lien entre la répartition à l'édition et la répartition à l'exécution

## 2. TAXONOMIE ET ÉTUDE DE CAS

-> installation en son réparti ?

- Problème de la latence entre i-score et raspberry, + utilisation de bande passante.
- Cas des applis de téléphone : un objet qui s'exécute sur plusieurs machines en parallèle dont on veut aggréger les résultats
- Approche alternative avec application mobile.

## 3. APPROCHE

- Groupe, client
- Maître-esclave

### 3.1. Édition

- Partage de file undo - redo
- Un seul "document" logique sur lequel tout le monde travail. Comme google docs.

### 3.2. Exécution

### 3.3. Introduction de primitives réparties dans i-score

- On souhaite modifier le moins possible le modèle pour les utilisateurs
- Notion de groupe
- Problème des devices

### 3.4. Méchanismes de synchronisation

- Cas possibles :

C1 ——t—— C3
C2 ———|——— C4

* Mode free ou synchronisé : pour chaque client ?

Cas 1 :

C1 ——t—— C3

C1 libre, C3 libre : dès que t est vrai, un message est envoyé au master qui envoie à C3 le trigger. C1 s'est déjà arrêté. Voire, ils peuvent se trig eux même en asynchrone.

C1 synchro, C3 synchro : dès que t est vrai, C1 calcule la date minimale à laquelle C3 peut être notifié, envoie le message et fixe ce temps de son côté.

Cas 2 :

C1 ——t—— C3
C2 ———|

* Synchronisation de la fin
* Synchronisation du début de la suite

En pratique, on n'implémente que la possibilité de synchro / désynchro toute une time node ; dire qu'une granularité plus fine est possible mais que l'intérêt n'apparaît pas en pratique dans les applications (et complexifierait l'UI pour rien).

Question principale : pour un time-node, comment choisit-on sur quelle machine une condition doit être vérifiée ? Possibilité de conditions groupées : faut-il que ce soit interne au formalisme (i.e. une case à cocher) ou externe (on dit explicitement machine1 :/truc && machin

Possibilité : synchronisation via démon externe (PTP, NTP...), mais pas toujours possible (on ne peut pas supposer que l'utilisateur a les droits pour changer l'horloge sur sa machine).

Synchronous ethernet

Avoir une horloge propre à i-score ? Mais du coup maintenant il faut la synchroniser à l'horloge système.

Ce qu'on fait : ping régulier vers chaque client (toutes les 100 millisecondes)

Quand quelque chose doit se synchroniser, on dit à chaque machine à quel instant il est supposé arriver par rapport à son horloge système.

Extension via système de composants

Quand un client reçoit un ordre pour un timenode à t, il l'applique dès que t <= local(t) (modulo un tick ?)

On ne synchronise qu'à chaque point d'entrée / de sortie d'un groupe de contraintes / time nodes

### 3.5. Extension des possibilités d'écriture

* Paramètres partagés
* Pattern matching sur addresses

## 4. PERFORMANCE

- Comparaison de l'algorithme "simple" et de l'algorithme avec retard

## 5. CITATIONS

Toutes les références bibliographiques des citations devront être listées dans la section "References", numérotées et en ordre alphabétique. Toutes les références listées devront être citées dans le texte. Quand vous vous référez au document dans le texte, précisez son numéro [1].

## 6. REFERENCES


[1] Author, E. "Titre du papier", Proceedings of the International Symposium on Music Information Retrieval, Plymouth, USA, 2000.

[2] Untel, A. Titre du livre. L'Armada, Paris, 2005.