Programmez vos IHM avec Interacto: une démonstration

Arnaud Blouin, Jean-Marc Jézéquel

Programmez vos IHM avec *Interacto*: une démonstration

Arnaud Blouin and Jean-Marc Jézéquel

Univ Rennes, INSA Rennes, Inria, CNRS, IRISA, Rennes
{prénom.nom}@irisa.fr

1 Introduction

Cette démonstration présente *Interacto* [1], une approche pour programmer la partie interactive des interfaces humain-machine (IHM). Cette partie interactive des IHM consiste principalement à traiter les interactions réalisées par les utilisateurs pour les transformer en commandes qui vont modifier ou contrôler le logiciel sous-jacent. Actuellement, les développeurs utilisent principalement une technique proposée avec SmallTalk et le modèle MVC (*Modèle-Vue-Contrôleur*) dans les années 80 : le modèle de traitement des événements IHM. Ce modèle considère les événements IHM de bas niveau (clic souris, *etc.*) comme le concept de première classe que les développeurs peuvent utiliser pour coder et utiliser des interactions utilisateur de plus en plus sophistiquées. Pour les interactions complexes, telles que le drag-lock (un type spécifique de glisser-déposer), le modèle actuel de traitement des événements présente des défauts critiques d'ingénierie logicielle qui entravent la réutilisation et la testabilité du code, et impactent négativement la séparation des préoccupations ainsi que la complexité du code.

Dans [1], nous avons proposé la contribution suivante en matière de génie logiciel : un modèle de traitement des interactions utilisateur appelé *Interacto* qui permet de résoudre les défauts du modèle classique mentionnés ci-dessus.

Cette démonstration a pour but d'expliquer comment *Interacto* fonctionne. Nous utiliserons son implémentation Web (TypeScript/Angular) pour dérouler les différents scénarios utilisés lors de l'étude empirique avec des étudiants dans [1].

2 Présentation d'*Interacto*

Interacto réifie les interactions utilisateur et les commandes IHM en tant qu'objets de première classe et fournit des algorithmes dédiés, des propriétés orientées objet, des optimisations d'exécution et des moyens de test pour permettre aux développeurs de rester concentrés sur les tâches principales de codage des IHM : (i) sélectionner les interactions utilisateur qu'ils doivent utiliser ; (ii) coder comment transformer ces interactions utilisateur en commandes IHM annulables ; (iii) réutiliser les interactions et les commandes ; (iv) écrire des tests IHM. Dans ce modèle, les événements IHM sont désormais considérés comme des concepts d'implémentation de bas niveau rarement utilisés par les développeurs.

Interacto prend la forme d'une API fluente via laquelle un *binder* permet de configurer un *binding Interacto*. Un *binding Interacto* transforme un exécution d'une interaction en une éventuelle commande, comme l'illustre le Listing 1.1.

```
binder()
  .using(() => new DragLock()) // Utilisation d'une interaction drag-lock
  .on(node) // sur l'objet graphique 'node'
  // Pour produire une commande Translate (pour déplacer 'node').
  // La commande Translate est une classe définie par le développeur
  .toProduce(i => new Translate(i.src.clientX, i.src.clientY))
  // Uniquement si le bouton principal de la souris est utilisé
  .when(i => i.button === 1)
  // Pendant l'exécution de l'interaction, mettre à jour la translation
  .then((i, c) => c.setCoord(
     c.shape.x + i.tgt.clientX-i.src.clientX,
     c.shape.y + i.tgt.clientY-i.src.clientY))
  .bind(); // Termine la configuration pour créer un binding Interacto
```

Listing 1.1. Configuration d'un *binding Interacto* pour déplacer un objet graphique

3 Description de la Démonstration

La démonstration impliquera trois scénarios de développement couvrant différentes interactions plus ou moins sophistiquées. Nous nous en servirons pour montrer les problèmes de génie logiciel qui affectent actuellement le modèle de traitement d'évènements IHM.

- Le premier exemple utilise un triple clic sur un objet HTML pour changer sa couleur (stockée dans un service). Nous monterons ensuite comment rendre cette commande annulable (*undo*/*redo*).
- le second exemple se focalise sur une interaction plus sophistiquée, largement utilisée dans les éditeurs de texte mais non fournie sur étagère par les librairies IHM. Cette interaction clavier consiste à attendre un laps de temps d'inactivité clavier (*e.g.* 2 secondes) avant de produire une commande.
- Le troisième exemple consiste à utiliser un glisser-déposer ou un drag-lock pour déplacer un objet HTML dont les coordonnées sont stockées dans un service. Nous monterons ensuite comment rendre cela annulable.

La version TypeScript/Angular d'*Interacto* est disponible librement sur les dépôts NPM[1]. La démonstration utilisera un dépôt Github contenant une application Angular dédiée que tout intervenant pourra télécharger[2].

References

1. Arnaud Blouin and Jean-Marc Jézéquel. Interacto: A Modern User Interaction Processing Model. *IEEE Transactions on Software Engineering*, pages 1–20, 2021. URL: `https://hal.inria.fr/hal-03231669`.

[1] `https://www.npmjs.com/package/interacto`
[2] `https://github.com/interacto/demo-session`