GLO-4008/GLO-7008 - Travail Pratique 2

Rapport

Equipe 4

Ludovic HU - 537 174 920

Bappi RAHAMAN - 537 046 96

Allan TARCY - 537 169 500

1. Décomposition manuelle

A. Quelle que soit votre approche, décrivez les détails de chaque étape:

Pour effectuer notre décomposition, nous avons commencé par repérer le fonctionnement global de l'application (observation de chaque classe présentée dans le fichier `petclinic_monolith.json`). D'après l'observation de l'api, et des models, nous avons extrait les contextes délimités en repérant les liens entre les classes (e.g le lien fort entre `Owner` et `Pets`).

Les concepts retrouvés sont (propriétaire, animal, vétérinaires, visites).

Chaque service est responsable de ses données et contient donc toutes les implémentations en base de données. Cependant, beaucoup de classes héritent des mêmes entités de base. Nous avons donc décidé de positionner ces entités au sein des services qui les appellent le plus. Elles seront certainement dupliquées, une fois le développement entamé.

Enfin, le service superviseur contient les classes qui supervisent le fonctionnement de l'application (initialisation de l'application, gestion des erreurs et exceptions).

2. Mono2micro

B. Le développeur peut-il influencer la qualité de la sortie de Mono2micro par le biais de ses entrées ? Si oui, décrivez ces entrées et comment vous les avez modifiées:

En utilisant Mono2micro, le développeur peut choisir le nombre maximum de partitions à générer. De plus, il possède le contrôle total des traces à générer.

Nous avons essayé de modifier le nombre maximum de microservices, d'abord en suivant les recommandations dans le fichier `config.ini`, puis selon selon nos propres décisions. Au niveau des traces, nous avons essayé de simuler les principaux 'use cases' des applications en fonction de notre compréhension.

3. MSExtractor

A. En utilisant les images Docker des outils d'analyse statique fournies sur le code source de PetClinic et Plants, générez les données requises pour MSExtractor. Afficher les fichiers générés à la fin de cette étape:

Liste des fichiers générés
hyperparams.json logbook.json logs.log decompositions.json

C. Le développeur peut-il influencer la qualité de la sortie de MSExtractor par le biais de ses entrées ? Si oui, décrivez ces entrées et comment vous les avez modifiées:

Oui, nous pouvons le faire via un grand nombre de paramètres. Comme par exemple, nous pouvons déterminer le nombre de micro-services maximal que nous souhaitons obtenir ou encore le nombre maximal de microservice. Mais les paramètres les plus intéressants sont probablement la '-G' qui représente le nombre de nombre de génération que nous souhaitons avoir étant donné que nous utilisons un algorithme génétique. C'est aussi pour cette raison que nous avons un grand contrôle sur la mutation via '-s' (probabilité de mutation). Ceux-ci sont de bon ch

4. TopicDecomp

B. Le développeur peut-il influencer la qualité de la sortie de TopicDecomp par le biais de ses entrées ? Si oui, décrivez ces entrées et comment vous les avez modifiées:

TopicDecomp permet d'ajuster le nombre de concepts ainsi que la "tolérance" pour déterminer l'appartenance d'un élément à un concept. Nous avons choisi de conserver la résolution initiale, qui parcourt l'ensemble des solutions possibles.

5. Évaluation

A. Lancez la plateforme et chargez une décomposition. Quels sont les services qui composent cette plateforme ?

En chargeant une décomposition réalisée avec mono2micro sur l'application petclinic, nous avons obtenu 6 services. Un service pour les vétérinaires, un pour les visites, un autre pour les propriétaires et leurs animaux, un pour la clinique et deux autres pour le reste des classes (tests et classes utilitaires)

B. Les métriques d'évaluation définies représentent différents aspects de la décomposition. À votre avis, quelle métrique devrait être utilisée lors de la sélection de la décomposition ?

A ce stade, la métrique la plus importante est certainement l'ICP qui mesure les dépendances entre chaque microservice. C'est un des enjeux principaux. On voudrait éviter d'avoir des services trop liés.

D. Sur quelle approche de décomposition parmi manuelle, Mono2micro, MSExtractor et TopicDecomp avez-vous basé vos résultats ? Pourquoi avez-vous choisi cette décomposition de base par rapport aux autres ?

Nos résultats sont basés sur notre décomposition manuelle. Il semblait plus simple de modifier nous même les microservices compte tenu de la taille de l'application et de notre familiarité avec les concepts étudiés. Certaines métriques semblent inadaptées au problème étudié et les approches de décompositions ne prennent pas vraiment en compte l'architecture des éléments de l'application pour former les microservices (ex: la relation entre `Owner` et `Pet`)

E. Avez-vous modifié la décomposition pendant la phase d'évaluation ? Si oui, quelles modifications avez-vous apportées et pourquoi ? Sinon, pourquoi la décomposition de base était-elle suffisante ?

La décomposition de base n'obtient pas de très bon score sur toutes les métriques, mais certains aspects de dépendances sont ignorés par la plateforme d'évaluation. La solution de base reste tout de même intéressante, car elle assure un nouveau de couplage relativement bas entre les microservices.