M2 COMASIC – Examen
Sûreté de Fonctionnement
28/01/2019

Durée de l'examen 2h.

Consignes particulières : les documents sont interdits. Les téléphones portables doivent être rangés. Le présent examen est noté sur 20 pts. Le barème est indiqué devant chaque question à titre indicatif.

I Principes de la sûreté de fonctionnement (3 pts)

Question 1) (1 pt) Donnez un exemple d'exigence non-fonctionnelle de sûreté de fonctionnement

Question 2) (1 pt) Définissez l'élimination des fautes et la tolérance aux fautes, expliquez en quoi ces deux pratiques sont complémentaires.

Question 3) (1 pt) Définissez brièvement ce qu'est une zone de confinement d'erreur dans une description architecturale d'un système.

II Architecture redondantes et modèles de fautes (8 pts)

Question 4) Supposons que l'on dispose d'une bibliothèque codée dans le langage de votre choix (C ou Java). Dans cette bibliothèque, on suppose que l'on dispose d'une fonction/méthode implémentant le calcul de la racine carrée d'un nombre à virgule flottante, le paramètre d'entrée a pour type double, le type de la valeur renvoyée est aussi double.

a) (1 pt) Expliquez le principe du mécanisme de l'enveloppe de fonction en général pour une fonction pouvant défaillir soit à cause de fautes activées lors de l'exécution du code, soit à cause de fautes d'interaction.

b) (1 pt) Illustrez ce principe pour la fonction racine carrée en précisant en quoi pourrait consister l'enveloppe (si l'on suppose que les fautes activées localement engendre des défaillances entraînant la production d'un résultat négatif).

Question 5) Nous supposons que l'on souhaite améliorer la fiabilité d'une fonction de calcul dont le temps d'exécution est inférieur à T si l'exécution est correcte.

a) (1,5 pt) Expliquer le principe de fonctionnement de la stratégie de réplication active pour N répliques. (architecture, comportement et hypothèse(s) permettant de justifier l'intérêt de l'architecture)

b) (0,5 pt) En faisant l'hypothèse que toute faute s'activant dans une réplique déclenche une défaillance byzantine, indiquer combien d'activations de fautes peuvent être tolérées dans le pire scénario pour une architecture à 9 répliques (dans le cas où les défaillances sont permanentes et où il n'y a pas de réparation).

Question 6) Supposons maintenant que chaque réplique d'une architecture de réplication active à 5 répliques possède 2 modes de défaillance différents : crash et byzantin. La différence est que même dans le pire cas, une défaillance par crash se traduit par l'absence de résultat. On souhaite modifier l'architecture pour s'adapter à l'hypothèse suivante : « le nombre de défaillance byzantine est toujours strictement inférieur au nombre de répliques fonctionnant correctement ». On notera en revanche que le nombre de défaillances par crash n'est pas contraint.

a) (0,5) Sous l'hypothèse énoncée ci-dessus quel(s) est(sont) les mode(s) de défaillance(s) possible(s) de l'architecture complète (à 5 répliques)?

b) (1,5 pts) Le composant en charge de produire le résultat final d'une telle architecture est appelé souvent « voteur », son comportement est d'attendre N valeurs puis de produire la valeur majoritaire. Expliquez comment il faudrait modifier / adapter ce comportement pour 1) ne pas attendre indéfiniment les répliques crashées, 2) pouvoir produire une réponse le plus tôt possible sans avoir à attendre la fin d'exécution de répliques défaillantes byzantines.

c) (2 pts) Identifiez l'ensemble des combinaisons de défaillances qui n'empêchent pas l'architecture de produire une réponse correcte (si l'on suppose le comportement décrit au début de la question 6 et le voteur décrit en 6b)), une formule peut suffir si elle est correctement justifiée

III Mécanismes de tolérance aux fautes (5 pts)

Question 7) (1 pt) Rappelez le principe du mécanisme de recouvrement avant.

Question 8) (2 pts) Indiquez en justifiant si la recouvrement arrière peut être appliqué de manière raisonnable pour une application en fonction des fautes suivantes (on supposera que la détection est parfaite … i.e. elle a lieu dès activation)

1. Pointeur (adresse de variable) non initialisé à la déclaration mais utilisé la ligne suivante (déréférencé).
2. Connexion réseau temporairement perdue entraînant la défaillance d'un appel système cherchant à exploiter les communications
3. Interférences physiques temporaires avec le matériel affectant uniquement les données de l'application
4. Interférences physiques temporaires avec le matériel affectant le code et les données de l'application.

Question 9) (2 pts) Tolérance aux fautes pour les données et entrelacement
On suppose que l'on dispose d'une manière de stockée une information codée sur 45 bits sur des blocs de 64 bit de telle sort que l'on sait tolérer 9 bits corrompus (car la distance de hamming sur ce code est de 19=2*9+1). **En supposant que l'on a à stocker N blocks en mémoire mais qu'une faute peut altérer jusqu'à 4 octets contigus, proposer une manière d'utiliser le codage par bloc décrit précédemment pour**

tolérer ces « bursts » de bits erronés en le couplant à une stratégie d'entrelacement du contenu stocké (i.e. décrivez votre solution sur le nombre de blocs suffisant pour l'illustrer).

IV Raisonnement autour de l'analyse quantitative (4 pts)

Il est dit qu'une architecture de réplication passive permet d'augmenter la disponibilité d'une fonction de calcul si les défaillances correspondent à des défaillances par crash. Nous allons supposer que l'on dispose de 4 répliques et que le crash est du à une défaillance matérielle qui se produit indépendamment des calculs réalisé :que le système soit la réplique primaire (celle qui réalise les calcul) ou une réplique secondaire. Les deux types de répliques sont soumis aux mêmes conditions d'occurrence des défaillances.

Question 9) (1 pt) Supposons que la probabilité pour une réplique de défaillir avant 100h de fonctionnement est de 10^{-2}. Que dire de la probabilité que l'architecture répliquée complète ait défailli avant 100h de fonctionnement ? (on supposera que les événements de défaillance sont des événements aléatoire indépendants).

Question 10) (0,5 pts) Nous souhaitons maintenant modéliser le fait que plus le temps de traitement dure plus la probabilité de crash augmente : quel modèle est le plus adapté entre une chaîne de Markov à temps continu et une chaîne de Markov à temps discret ?

Question 11) (2,5 pts) Proposez un modèle à base d'une ou plusieurs chaînes de Markov permettant

a) de capturer le fait que plus le temps s'écoule, plus la probabilité de défaillance augmente (indépendamment de l'application exécutée).

b) de tenir compte de l'indépendance de l'occurrence des défaillances de répliques

c) de tenir compte du fait qu'en moyenne une réplique défaille au bout de T unité de temps.

d) de capturer pour chaque réplique le fait que la détection du crash prend en moyenne T_{det} unités de temps (indépendamment sur chaque réplique) et qu'après cette détection la réparation du crash prend en moyenne T_{rep} unités de temps.

La définition d'une telle chaîne devra contenir une description du sens pratique de ses différents états.