Scénarios, tactiques, vues architecturales

Disponibilité

Table des matières

[**1 Scénario : Un disque dur tombe en panne**](#_30j0zll) **5**

[Tactique 1 : Monitor](#_lbzf0uqa3wv) 5

[Tactique 2 : Active redundancy](#_y97wzd4hz65e) 5

[Tactique 3 : Spare](#_sys1io47ut9b) 5

[Tactique 4 : State resynchronisation](#_1fob9te) 5

[Vue architecturale](#_8khp2a34jll2) 6

[Diagramme](#_3znysh7) 6

[Spare](#_2et92p0) 7

[Description du diagramme](#_tyjcwt) 7

[Table d’éléments](#_3dy6vkm) 7

[Liens entre les vues et les tactiques](#_1t3h5sf) 7

[**2 Scénario : Le participant de la bourse veut effectuer une action, mais le serveur n’est pas disponible**](#_4d34og8) **8**

[Tactique 1 : Hearthbeat](#_j7lm6yk3iygl) 8

[Tactique 2 : Non-stop forwarding](#_2e006190yo71) 8

[Tactique 3 : Monitor](#_io76n6ay2q2t) 8

[Vue architecturale (à l’extérieur de la classe)](#_ry5focvac76v) 8

[Diagramme](#_2s8eyo1) 8

[Légende](#_i7ead7ml534a) 8

[Texte de description du diagramme](#_zg8q4wzh60z4) 8

[Table de description des éléments du diagramme](#_uap55dcfulrk) 9

[Texte décrivant la relation entre les éléments et les tactiques](#_6b6fa9lrcn76) 9

[**3 Scénario : Panne de service dû à une surabondance de requêtes**](#_17dp8vu) **11**

[Tactiques](#_lnxbz9) 12

[Tactique 1 : Détection - Battement de coeur (Heartbeat)](#_2rwqzfqqqys4) 12

[Tactique 2 : Récupération - Rechange (Spare)](#_8c61jbn6141h) 12

[Tactique 3 : Prévention - Mise hors d’usage (Out of use)](#_pj4kkg4bcagq) 12

[Vue architecturale](#_1ksv4uv) 13

[Diagramme](#_44sinio) 13

[Légende](#_z337ya) 13

[Texte de description du diagramme](#_4i7ojhp) 14

[Table de description des éléments du diagramme](#_2xcytpi) 14

[Texte décrivant la relation entre les éléments et les tactiques](#_1ci93xb) 14

[**4 Scénario : Denial of Service : le serveur DNS du système est surchargé et arrête de fonctionner normalement**](#_3whwml4) **15**

[Tactique 1 : Heartbeat](#_si9k89d5emga) 15

[Tactique 2 : Ignore Faulty Behavior](#_8coulak6iv5u) 15

[Tactique 3 : Predictive Model](#_r44fl58bf5nn) 15

[Vue architecturale](#_jrrwox6u5wnh) 16

[Diagramme](#_3az3dnevmazl) 16

[Texte de description du diagramme](#_6q96qow1ugp7) 16

[Table de description des éléments du diagramme](#_6hvj0rye5n6o) 16

[Texte décrivant la relation entre les éléments et les tactiques](#_aa5i6b1sen94) 17

[**5 Scénario : Trop de requêtes à l’externe via FIX et SAIL dans une très petite période pouvant ainsi créer un ‘system time-out’.**](#_3as4poj) **18**

[Tactique 1 : Ségréguer les services FIX et SAIL](#_szo95rrtllr7) 18

[Tactique 2 : Architecture distribuée](#_hmihwtm7eyvu) 18

[Tactique 3 : Avertissement](#_jdmcbocfdahy) 18

[Vue architecturale (à l’extérieur de la classe)](#_gdqdispmv72i) 19

[Diagramme](#_1pxezwc) 19

[Légende](#_es1557ber1wy) 19

[Texte de description du diagramme](#_fcmt4dwlvac3) 20

[Table de description des éléments du diagramme](#_kl2luhndaqsk) 20

[Texte décrivant la relation entre les éléments et les tactiques](#_4b2lxvehil8y) 20

[**6 Scénario : Le serveur prend feu**](#_49x2ik5) **21**

[Tactique 1 (Detect faults) : Monitoring](#_ums3345zyspk) 21

[Tactique 2 (Recover from faults) : Active redundancy (hot spare)](#_2eqest2312nd) 21

[Tactique 3 (Prevent faults) : Exception Prevention](#_2tdzfpufkljo) 21

[Vue architecturale (à l’extérieur de la classe)](#_6ilqaahafwn4) 22

[Diagramme](#_2p2csry) 22

[Légende](#_h50sv4wqd932) 22

[Texte de description du diagramme](#_n884yyjb74n9) 22

[Table de description des éléments du diagramme](#_80yzq7hj1vzn) 22

[Texte décrivant la relation entre les éléments et les tactiques](#_nrh2iaded1yo) 22

[**7 Scénario : Beaucoup trop d’ordres de transactions pour le système via SAIL et FIX causent des timeouts**](#_147n2zr) **23**

[Tactique](#_3o7alnk) 23

[Tactique 1 : Modèle prédictif](#_23ckvvd) 23

[Tactique 2 : Séparer les services FIX et SAIL](#_nfxbwgix0qe4) 23

[Tactique 3 : Redondance Active](#_9uyde7ba51lq) 23

[Diagramme vue architecturale](#_3irezcpbo2i2) 24

[Légende](#_ihv636) 24

[Texte de description du diagramme](#_32hioqz) 24

[Table de description des éléments du diagramme](#_1hmsyys) 24

[Texte décrivant la relation entre les éléments et les tactiques](#_41mghml) 25

[**8 Scénario : Assurer une disponibilité de 99.98%**](#_2grqrue) **26**

[Tactique 1 : Ping/Echo](#_1v1yuxt) 27

[Description : Connaître le temps de réponse et la disponibilité d’un système](#_4f1mdlm) 27

[Tactique 2 : Retry](#_2u6wntf) 27

[Description: On redémarre la transaction qui a échoué.](#_19c6y18) 27

[Tactique 3 : Predictive Model](#_3tbugp1) 27

[Description: On cumule plusieurs données statistiques par rapport aux équipements et à leur utilisation.](#_28h4qwu) 27

[Vue architecturale (à l’extérieur de la classe)](#_1mrcu09) 28

[Diagramme](#_46r0co2) 28

[Légende](#_111kx3o) 28

[Texte de description du diagramme](#_3l18frh) 28

[Table de description des éléments du diagramme](#_206ipza) 28

[Texte décrivant la relation entre les éléments et les tactiques](#_4k668n3) 28

[**9 Scénario : L’interface du Market Surveillance n’a plus de courant.**](#_2zbgiuw) **30**

[Tactique 1 : Détection -TimeStamp](#_1egqt2p) 30

[Tactique 2 : Détection - Heart Beat](#_o6d7i34kqty9) 30

[Tactique 3 : Récupération - Redondance externe](#_mugribjj00uu) 30

[Vue architecturale](#_xm357kaz6zif) 31

[Diagramme](#_3ygebqi) 31

[Légende](#_k4s51mnyltok) 31

[Texte de description du diagramme](#_met97d40d9eh) 31

[Table de description des éléments du diagramme](#_t6tmmb4o8jo0) 31

[Texte décrivant la relation entre les éléments et les tactiques](#_jc6amg8tkupc) 32

[**10 Scénario (problématique) : Perte de communication avec l’interface SAIL pour recevoir les transactions.**](#_2dlolyb) **33**

[Tactique 1 : PING/ECHO](#_lvg9enj8l9nw) 33

[Tactique 2 : Heartbeat](#_61se448lut2b) 33

[Tactique 3 : Active Redundancy / Loadsharing](#_8sj080atoo1y) 33

[Vue architecturale (à l’extérieur de la classe)](#_e81qaxuwk1z3) 35

[Diagramme](#_ygv40ljd8k95) 35

[Légende](#_6y3lzm3rhogx) 35

[Texte de description du diagramme](#_3zndr75mjux8) 35

[Table de description des éléments du diagramme](#_i0wveoa2v7rp) 35

[Texte décrivant la relation entre les éléments et les tactiques](#_3pt6xtsxro78) 36

[**11 Scénario : Devrait avoir une disponibilité de 99.98**](#_sqyw64) **37**

[Tactique 1 : Battement cardiaque (Heartbeat)](#_uxc32yyupcm2) 37

[Tactique 2 : Redondance active (Active redundancy)](#_cg8agiru5pij) 37

[Tactique 3 : Removal from service](#_enq27ute7uyg) 37

[Vue architecturale (à l’extérieur de la classe)](#_q96d6rhkvrzv) 38

[Diagramme](#_q08f5asv4vei) 38

[Légende](#_138w9royo506) 38

[Texte de description du diagramme](#_73yzf2akfhv4) 38

[Table de description des éléments du diagramme](#_g3kklur31530) 38

[Texte décrivant la relation entre les éléments et les tactiques](#_8b9hvc6w3sur) 39

[**12 Scénario: Il est impossible de se connecter au système de bourse, car le serveur n’est pas disponible**](#_1rvwp1q) **40**

[Vue architecturale](#_4bvk7pj) 41

[Diagramme et légende](#_2r0uhxc) 41

[Texte de description du diagramme](#_3q5sasy) 41

[Table de description des éléments du diagramme](#_25b2l0r) 41

[Texte décrivant la relation entre les éléments et les tactiques](#_kgcv8k) 42

# 1 Scénario : Un disque dur tombe en panne

|  |  |
| --- | --- |
| **Objectifs d'affaires** | L’entreprise veut offrir des services fiables à leurs clients |
| **Source** | Disque dur |
| **Stimulus** | Panne |
| **Artéfact** | Stockage persistant |
| **Environnement** | Opération normale |
| **Réponse** | Détection de la faute :   * Enregistrer la faute * Notifier la personne appropriée   Rétablissement :   * Masquer la faute * Mettre le disque fautif hors service * Changer le disque par un backup |
| **Mesure de la réponse** | * Temps de détection * Temps de réparation ou d'accès à l’alternative |
| **Questions** | 1. Quel est l’importance des données sur le disque dur? 2. Y a-t-il qu’un data-center? 3. Quel niveau de RAID? 4. Il y a-t-il eu une perte de données ? 5. À quel niveau les clients sont affectés ? |

## 1.1 Tactique 1 : Monitor

**Description**: Services de supervision des disques durs.

**Justification**: Afin de vérifier l’état et la santé des disques durs, il est nécessaire pour l’entreprise d’acquérir un outil de supervision de ces derniers. En supervisant les disques durs, on est en mesure de réagir rapidement en changeant le disque dur défaillant.

## 1.2 Tactique 2 : Active redundancy

**Description**: Utilisation de plusieurs disques prêt et configurés à recevoir et répondre des transactions en parallèle

**Justification**: Le client va prendre la première réponse reçue parmis les multiples disques qui ont répondu. L’échec d’un disque à fournir une réponse sera presque invisible puisque la même réponse provenant des autres disques sera disponible. La différence de temps de réponse sera seulement de quelques millisecondes (dans le cas où la réponse du disque fautif aurait atteint sa destination en premier).

## 1.3 Tactique 3 : Spare

**Description**: Une plateforme composé de harddrive près à remplacer des composantes défectueuse. Cette méthode est principalement employé pour la réparation de la source du problème (le bris du harddrive) plutôt que d’offrir une alternative immédiate à l’usagé.

**Justification**: Permet de remplacer le disque dur endommagé par un neuf de façon transparente, rapide et sans l’intervention humaine.

## 1.4 Tactique 4 : State resynchronisation

**Description**: Synchronisation d’un nouveau disque dur

**Justification**: Lorsqu’un disque dur tombe en panne, il faudra l’échanger avec un nouveau. Il est donc important de synchroniser ce dernier avec les autres disques durs grâce à la configuration en RAID.

## 1.5 Vue architecturale

### Diagramme

### 

### Spare

Afin d’assurer une disponibilité constante des disques durs, il est nécessaire pour l’entreprise doit tenir plusieurs disques durs dans son inventaire. Lorsqu’un disque dur tombe en panne, il sera donc possible de le remplacer par l’un des disques durs en inventaire. De plus, il faudra garder un certain minimum de disque dur afin d’éviter d’être à court de disques durs.

### Description du diagramme

Dans le diagramme ci-dessus, on décrit 3 tactiques permettant de détecter et réparer une défaillance au niveau des disques durs. Tout d’abord, la tactique du monitoring nous permet d’analyser et détecter lorsqu’un disque dur devient défaillant. Ensuite, nous avons le “active redundancy”, une tactique permettant d’avoir une très grande disponibilité d’accès à l’information. Pour terminer, la tactique du “state synchronisation” offre une stabilité au niveau d’intégralité des données sur les disques durs.

### Table d’éléments

|  |  |
| --- | --- |
| Processus | Programme en cours d’exécution sur une machine (serveur) |
| Décision | Logique booléenne. Deux sorties possibles: l’une lorsque la condition est vrai, l’autre lorsque la condition est fausse. |
| Association | Sens du chemin logique à suivre dans le graphe |

### Liens entre les vues et les tactiques

Pour ce qui est du monitoring, la vue ci-dessus présente un système de requête effectué par l’outil de monitoring. C’est une requête de test qui permet de savoir si le disque dur est fonctionnel et contient l’intégralité des données. Ce système de monitoring doit être le plus réaliste possible, donc en temps réel. Ensuite, pour le “active redundancy”, on remarque que le ou les serveurs connectés avec les disques durs ont une communication, car le serveur envoient une requête vers les disques durs et celui qui répondra le plus rapidement enverra la réponse au serveur. Les autres réponses seront ignorées par les serveurs. Pour terminer, nous avons un diagramme d’état pour la tactique de “state synchronisation” qui démontre les décisions que le système prendra en fonction du contenu des disques durs. Si le disque dur ne contient pas l’intégralité des données, il devra être synchronisé avec l’état le plus à jour des disques durs (en redondance).

# 2 Scénario : Le participant de la bourse veut effectuer un action, mais le serveur n’est pas disponible

|  |  |
| --- | --- |
| **objectifs d'affaires** | Permettre au participant de la bourse d’effectuer des actions de ventes ou d’achat boursiers |
| **Source** | Le serveur |
| **Stimulus** | Le serveur ne répond pas |
| **Artéfact** | Processus, connexion réseau |
| **Environnement** | Opération normale |
| **Réponse** | Masquer la faute => rediriger le trafic vers un autre serveur  Notifier les entitées appropriées (pour effectuer la maintenance nécessaire) |
| **Mesure de la réponse** | 1 ms pour 98% du temps en toutes circonstances |
| **Questions** | 1. Est ce que tout le réseau ne répond pas? 2. A-t-il trop de trafic qui circule sur le réseau? 3. Est-ce une attaque envers les serveur (Denial of service)? 4. Y a-t-il d’autre serveur qui ne répondent pas? 5. Est-ce une erreur temporaire dans le réseau? |

## 2.1 Tactique 1 : Heartbeat

**Description**: C’est un mécanisme de détection de défaut qui emploie un échange de messages périodique entre un moniteur de système et un processus surveillé.

**Justification**: Permet de détecter les défauts rapidement lorsqu’un service n’est pas disponible. Cette tactique permettra de réduire le transport et le traitement avec les messages périodique pour un système en constante évolution.

## 2.2 Tactique 2 : Non-stop forwarding

**Description**: La supervision qui gère la connectivité et les informations de routage et le plan de données qui fait le travail réel de routage des paquets de l’expéditeur au receveur.

**Justification**: Permet de router les paquets avec un chemin différent que celui utilisé en cas de panne d’un routeur. Cette tactique permet de garder le service en marche jusqu’à que la panne soit réparée.

## 2.3 Tactique 3 : Monitor

**Description**: Un moniteur permet de vérifier l’état des composants d’un système.

**Justification**: Le moniteur pourra détecter une défaillance ou une congestion dans le système. Ce qui permettra de vérifier si les composants utilisées sont toujours aussi performants.

## 2.4 Vue architecturale (à l’extérieur de la classe)

### Diagramme

### Légende

### Texte de description du diagramme

L’application est toujours en communication avec le serveur de la bourse. Il y a plusieurs serveurs de l’application de la bourse, afin de prévenir des défaillances qui coupe complètement le service. Les différentes tactiques utilisées sont illustrées dans le diagramme.

### Table de description des éléments du diagramme

|  |  |
| --- | --- |
| **Tactiques/Élement** | **Description** |
| Heartbeat | Tactique effectué pour vérifier l’état de la connection au serveur |
| Non stop forwarding | Tactique effectuée pour empêcher la perte de paquet |
| Monitor | Tactique Monitor effectué sur le serveur, pour regarder les états des composantes |
| Achat/Vente | Une transaction courante effectué par un participant |
| Application de bourse | Une instance de l’application qui peut effectuer des transaction |
| ServeurBourse | Une instance du serveur qui gardent les données des transactions |

### Texte décrivant la relation entre les éléments et les tactiques

La tactique de non-stop forwarding envoie les paquets entrant vers une autre instance de serveur de l’application dans le cas d’une panne. Le monitoring permet de faire une vérification des composantes du serveur afin de prévenir une panne. Le heartbeat permet de savoir si la connectivité au serveur est saine. Toutes ces tactiques viennent de l’application de la bourse. Dans le schéma, une transaction régulière est aussi illustrée

# 3 Scénario : Panne de service dû à une surabondance de requêtes

|  |  |
| --- | --- |
| **Objectifs d'affaires** | Avoir une disponibilité de 99.98% durant les heures d’ouverture afin de favoriser l’accès au marché boursier de manière fiable et constant. |
| **Source** | Infrastructure Linux et Windows. |
| **Stimulus** | Une panne de service, ~~surabondance de requêtes.~~ |
| **Artéfact** | Les connexions réseaux. |
| **Environnement** | Environnement normal. |
| **Réponse** | 1. Garder une copie des détails du *downtime;* 2. Notifier l’équipe de la maintenance; 3. Désactiver l’accès au marché boursier pour cesser le flux des erreurs. |
| **Mesure de la réponse** | Délai entre le début de la défaillance et sa résolution (downtime ou MTTR) |
| **Questions** | 1. Quelles sont les opérations possibles pour réparer le crash? 2. Comment informer les utilisateurs et les administrateurs rapidement de la panne? 3. Comment les transactions suspendues seront stockées? 4. Quelle est la capacité des serveurs? |

# 

# 

## 3.1 Tactiques

## 3.2 Tactique 1 : Détection - Battement de coeur (Heartbeat)

Description : Une émission constante de message est établie. Ces messages ou requêtes sont envoyés de manière périodique d’une composante à une autre afin d’assurer qu’il n’y a pas de faute. Lorsqu’une faute est détectée, celle-ci est passée à la méthode de recouvrement de faute.

Justification : C’est une bonne méthode d’analyse des composantes puisqu’elle permet d’avoir l’état constant de celles-ci. On note ainsi un haut niveau de détection puisque le système est constamment sous surveillance au rythme de la détection via les battements de coeur.

## 3.3 Tactique 2 : Récupération - Rechange (Spare)

Description : On conserve une deuxième plateforme comportant les divers composants essentiels pour faire fonctionner le système et pour prévenir les fautes occasionnelles. À la base, cette seconde plateforme n’est pas nécessairement active, mais conserve les informations dans un point de contrôle. Elle est activée lorsque la plateforme principale tombe en panne et continuer les tâches sans embûche.

Justification : Cette tactique permettrait de toujours avoir un logiciel fonctionnel en parallèle lorsqu’une des deux instances n’est plus en mesure de fonctionner. En cas de panne, le système boursier pourrait fonctionner sur une autre plateforme. Ainsi, on diminue les interruptions de service du côté de l’utilisateur.

## 3.4 Tactique 3 : Prévention - Mise hors d’usage (Out of use)

Description : Les capacités des logiciels et des serveurs sont testées fréquemment pour vérifier leur efficacité et leur fonctionnement. Les logiciels et les serveurs pourront être mis hors d’usage pour un redémarrage, une modification physique ou logicielle. Les logiciels informent les clients et les administrateurs des fautes.

Justification : Possibilité de faire des statistiques et déterminer les moments où les serveurs sont surchargés. Ainsi, il sera possible d’attribuer de nouveaux serveurs pour satisfaire à la demande (redirection avec le logiciel). Redémarrage des serveurs pour libérer la mémoire quand les serveurs sont moins en demande. Lorsque les serveurs ou les logiciels ne sont plus fonctionnels, les administrateurs et les utilisateurs seront informés et pourront agir en conséquence, par exemple, si une pièce physique brise, le logiciel informera les administrateurs et celle-ci pourra être changée.

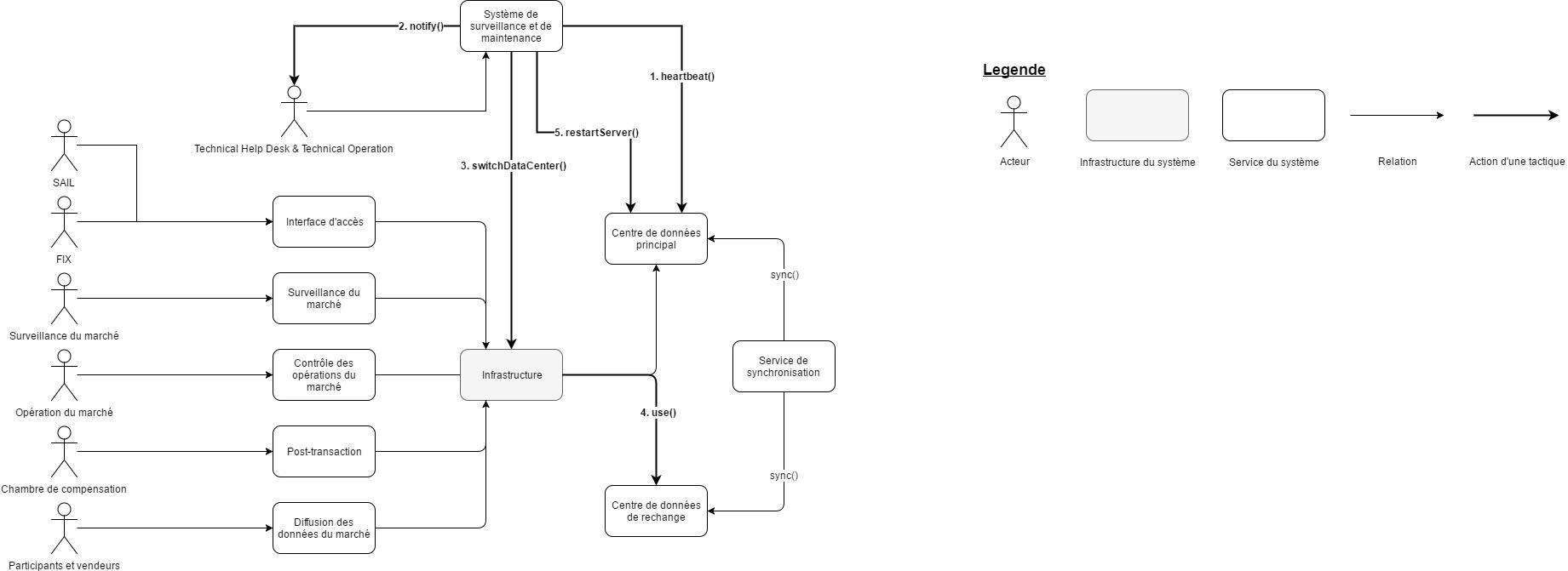
# 

## 3.5 Vue architecturale

## 3.6 Diagramme

## 

## 3.7 Légende



## 

## 

## 3.8 Texte de description du diagramme

Le diagramme est composé des différents services pouvant être utilisés par les acteurs ainsi que le système permettant de maintenir l’infrastructure et les deux centres de données. La section des acteurs montre les différentes utilisations du service du système, telles que l’interface d’accès, la surveillance du marché et la diffusion des données du marché. La section sur la maintenance montre la surveillance du système et l’accès de celle-ci par un technicien des opérations. Finalement, la section des deux centres de données montre la relation entre ces deux centres et le service de synchronisation. Le service de synchronisation permet de garder les deux centres de données pleinement synchronisés afin que le deuxième centre prenne la relève dans le cas où le premier soit en défaillance.

## 3.9 Table de description des éléments du diagramme

|  |  |
| --- | --- |
| **Tactiques** | **Descriptions** |
| Battement de coeur (*Heartbeat)* | Obtenir des alertes en provenance du système de surveillance et de maintenance et le centre de données principal. |
| Rechange (*Spare*) | Permet un “back-up” en cas de panne du centre de données primaire. |
| Mise hors d’usage (*Out of use*) | Redémarrer les serveurs pour effectuer des changements physiques ou logiciels. |

## 3.10 Texte décrivant la relation entre les éléments et les tactiques

La première tactique, soit le battement de coeur, est représentée selon une fonction qui ferait l’envoi de messages entre le système de surveillance et de maintenance et le centre de données principal. Le système de surveillance agit comme un observateur dans la totalité du système et est notifié lorsque le battement de coeur n’est pas retourné. Lorsque le battement n’est pas retourné, c’est un signe que le centre de données principal est proie à une faute. Ainsi, lorsque le système est hors d’usage, on passe à la seconde tactique qui implique la récupération du système, la rechange, qui effectuera le traitement de la faute. La deuxième tactique, soit la rechange, est représentée comme une procédure (switchDataCenter()) entre le système de maintenance et l’infrastructure principale. Lorsque notifiée, l’infrastructure autorisera un changement du centre de données principal vers un centre de données secondaire qui prendra alors en charge les requêtes des utilisateurs. La troisième tactique, soit la mise hors usage, permet de relier le système de surveillance avec l’infrastructure. Lorsque l’infrastructure est en besoin logiciel, soit un besoin en mémoire, le logiciel peut alors vérifier s’il peut redémarrer les serveurs (le centre de données principal). Aussi, le système de surveillance capte les données d’utilisation au niveau de l’infrastructure pour examiner quand il est possible de faire des modifications physiques. Il serait alors possible d’arrêter les serveurs pour faire des changements, par exemple, ajouter plus de mémoire ou changer de processeur.

# 4 Scénario : Denial of Service : le serveur DNS du système est surchargé et arrête de fonctionner normalement

|  |  |
| --- | --- |
| **Objectif d'affaires** | Le système devrait avoir une disponibilité de 99.98% |
| **Source** | Hacker (externe) |
| **Stimulus** | Attaque DOS (serveur DNS arrête de répondre) |
| **Artéfact** | Table de routing du serveur DNS |
| **Environnement** | Opération système normale |
| **Réponse** | Informer les opérateurs du système (Technical Help Desk & Operation)  « Blacklister » l'adresse IP de l’attaquant si possible  Au besoin, shutdown du système pour le réparer |
| **Mesure de la réponse** | Temps pour détecter l’attaque  Pourcentage de disponibilité du système |
| **Questions** | 1. Comment détecter une attaque DOS (combien de requêtes par intervalle de temps)? 2. Quel processus est en charge de détecter la source (IP) de l’attaquant ? 3. Si le système doit être arrêté complètement, combien de temps avant qu’il redevienne disponible ? 4. Quel système va monitorer les serveurs ? |

## 4.1 Tactique 1 : Heartbeat

**Description**: Le serveur (composante) doit informer le système « Monitor » qu’il est toujours fonctionnel à une intervalle X de temps.

**Justification**: De cette façon on peut avoir le statut du serveur en temps réel.

## 4.2 Tactique 2 : Ignore Faulty Behavior

**Description**: Ignorer la source des attaques DOS.

**Justification**: En ayant un processus qui permet d’isoler et « blacklister » l’adresse IP de l’attaquant, nous pouvons bloquer et ignorer le comportement non-désiré.

## 4.3 Tactique 3 : Predictive Model

**Description**: Dans le message du heartbeat, nous pouvons ajouter le nombre de requêtes faites depuis le dernier heartbeat.

**Justification**: De cette façon, nous pouvons monitorer l’achalandage du serveur est agir en conséquence (prévention)

## 4.4 Vue architecturale

### Diagramme

### Dispo_Dessin - Page 1

### Texte de description du diagramme

Ce diagramme est une représentation banalisée d’une architecture web soumise à une attaque de type « Déni de service ». Le style composant et connecteur a été utilisé pour mieux présenter les sources des attaques DOS. Dans le cas de base, un utilisateur force un terminal à envoyer une grande quantité de requêtes HTTP au **serveur DNS** dans le but de surcharger sa capacité à traiter les requêtes et à rediriger les utilisateurs. Lorsque le **serveur DNS** est placé dans une telle situation, il devient incapable d’envoyer des signaux au **serveur de surveillance** (moniteur) dans un délai raisonnable et ce-dernier va envoyer une alerte aux **opérateurs systèmes** via une alerte SMTP.

### Table de description des éléments du diagramme

|  |  |
| --- | --- |
| Élément du diagramme | Description |
| Attaquant DOS | Machine envoyant des requêtes à un serveur avec une très haute fréquence et des adresses IP différentes dans le but de provoquer un arrêt de la fonctionnalité du **serveur DNS** à rediriger lesdites requêtes. |
| Client légitime | Utilisateur envoyant des requêtes HTTP au serveur DNS dans le but de pouvoir accéder à un serveur applicatif. |
| Serveur DNS | Composant physique contenant une table de routage permettant de rediriger les éléments reçus vers le serveur applicatif approprié, et une liste noire d’adresse à ignorer. |
| Serveur applicatif | Serveur contenant les pages webs accessibles à un client légitime. |
| Moniteur « Heartbeat » | Application utilisée pour recevoir des **signaux ICMP** ainsi que d’autres informations pour mesurer et analyser le système.  Cette application lance des alertes à un intervenant approprié si le seuil des mesures est dépassé. |
| Opérateur système | Administrateur du système ayant accès aux outils nécessaires pour le maintenir en fonction. |

### Texte décrivant la relation entre les éléments et les tactiques

Pour la tactique **Heartbeat**, c’est le **serveur DNS** qui communique directement avec le **moniteur** pour signaler son bon fonctionnement. Cette tactique travaille de concert avec la tactique **Predictive Model**. Cette dernière permet d’évaluer l’achalandage du serveur en envoyant, par exemple, le nombre de requête par seconde au **moniteur** et d’établir un seuil qui correspond au bon comportement du système. Dans le cas d’une attaque DOS, le **serveur DNS** avertira immédiatement le **moniteur** d’un comportement suspect, et le processus de défense contre les DOS sera exécuté rapidement. Viendra ensuite la tactique **Ignore Faulty Behaviour** qui enregistrera l’adresse IP qui envoie un trop grand nombre de requête dans le **serveur DNS**. Ainsi le serveur pourra ignorer les requêtes provenant de cette adresse, et continuer à servir les adresses au comportement régulier.

# 5 Scénario : Trop de requêtes à l’externe via FIX et SAIL dans une très petite période pouvant ainsi créer un ‘system time-out’.

|  |  |
| --- | --- |
| **objectifs d'affaires** | Disponibilité de 99,98% |
| **Source** | ping, Personne externe, les autres bourse, |
| **Stimulus** | Le serveur ne répond pas avant le time out, la réponse est incorrect |
| **Artéfact** | canaux de communications, process |
| **Environnement** | opération normal lors des heures d’ouverture de la bourse |
| **Réponse** | logger, avertir les parties prenantes, |
| **Mesure de la réponse** | L’interval de temps passé en mode dégradation. |
| **Questions** | 1. Quel est le time-out? 2. Quel sont les spécifications du système utilisé? 3. Quel est le nombre de requête par FIX et SAIL? 4. Quel est la taille de chacune des requêtes? |

## 5.1 Tactique 1 : Ségréguer les services FIX et SAIL

**Description**: Nous allons créer des clusters différents: un pour SAIL et, un pour FIX réception de transaction et un pour FIX envoi de transaction.

**Justification**: Ceci permet d’empêcher que la non disponibilité d’un des services impacte les autres.

## 5.2 Tactique 2 : Architecture distribuée

**Description**: Nous allons créer un système distribué pour gérer la charge de travail dynamiquement.

**Justification**: Le load balancer que nous mettrons en place nous permettra de répartir la charge selon la capacité des machines afin de répondre rapidement à la demande des clients. De plus, nous serons en mesure d’ajouter des machines lorsque la demande augmente. L’architecture sera donc “scalable”.

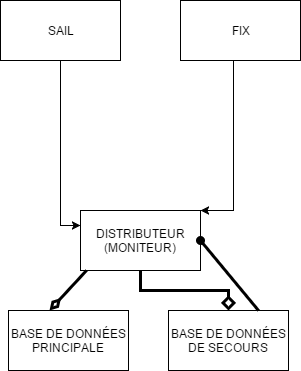
## 5.3 Tactique 3 : Avertissement

**Description** : Si le système est sur le point de tomber dans un état dégradé, on pourrait avoir seulement les informations d’une moitié d’une requête et donc, empêcher cette transaction avant qu’elle ne soit faite.

**Justification** : Une transaction semi-faite pourrait endommager le système et n’est certainement pas fiable. Par conséquent, il est primordial d’avoir une transaction complète ou aucune.

## 5.4 Vue architecturale (à l’extérieur de la classe)

### Diagramme



### Légende

 réception et envoi de données (transfert de paquets de données)

 distribution des tâches

 demande de refus de transaction

### Texte de description du diagramme

Cette vue composantes et connecteurs démontrent comment les nouveaux systèmes misent en place augmente la disponibilité du système.

### Table de description des éléments du diagramme

* SAIL : Interface propriétaire (binaire) de la bourse de Montréal)
* FIX: Interface standard de message pour système boursier
* Distributeur (moniteur) : moniteur qui surveille la charge de travail de la BD et commence à transmettre à la base de données de secours au nécessaire tout en avertissant l’administrateur de l’erreur.
* Base de données de secours : base de données en cas de débordement de tâche de la base de donnée principale.
* Base de données : base de données traitant toutes les transactions bancaires

### Texte décrivant la relation entre les éléments et les tactiques

Afin de prévenir le manque de disponibilité de notre système, nous avons séparé les deux différents ‘Front-end’. SAIL et FIX devraient être accédés et gérés séparément. Par conséquent, dans le diagramme ci-dessus, le tout a été ségretter. Ainsi, la gestion des réceptions et envoies de chacun des deux modules ce fonds indépendamment. De plus, le distributeur moniteur la base de données principale et si elle advient à voir qu’elle est surchargée, elle envoie une notification à l’administrateur et envoie les prochains paquets sur le serveur de secours. Finalement, si le serveur de secours arrive à sa limite aussi avant qu’un intervenant n’aille rétablit le système, le refus de transaction est commencé, et ce, avant la perte de données d’un paquet.

# 6 Scénario : Le serveur prend feu

|  |  |
| --- | --- |
| **Objectifs d'affaires** | L’interface SAIL doit avoir une disponibilité de 99.98%. |
| **Source** | Un client externe utilise l’interface SAIL pour voir la valeur de ses actions |
| **Stimulus** | (Omission) Aucune réponse car le serveur a pris feu. |
| **Artéfact** | Serveur d’application |
| **Environnement** | Normal operation |
| **Réponse** | Cataloguer les erreurs/incident qui se produisent  Notifier les personnes concernées  Rediriger les requêtes vers un autre serveur |
| **Mesure de la réponse** | Pendant combien de temps l’interface SAIL n’est pas disponible |
| **Questions** | 1. Ça prend combien de temps pour détecter que l’interface SAIL n’est pas disponible ? |

## 6.1 Tactique 1 (Detect faults) : Monitoring

**Description**: Le serveur de monitoring va vérifier aussi fréquemment que possible si le serveur de l’interface SAIL est bien en ligne.

**Justification**: Puisqu’on a seulement 5 secondes (environ) de downtime permis par jour, il faut que la détection d’un problème soit extrêmement rapide. C’est pour ca que le monitor devra fonctionner le plus rapidement qu’il peut.

## 6.2 Tactique 2 (Recover from faults) : Active redundancy (hot spare)

**Description**: Les requêtes sont redirigées vers un autre serveur lorsque le premier n’est pas disponible.

**Justification**: On peut imaginer plusieurs serveurs qui attendent de recevoir des requêtes. Lorsque le moniteur détecte qu’un serveur est hors-service, il le retire de sa liste des serveurs available.

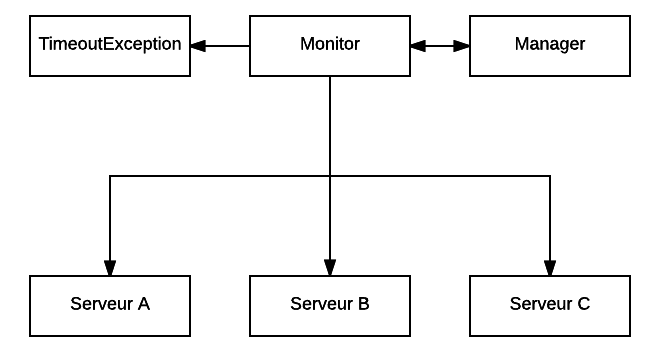
## 6.3 Tactique 3 (Prevent faults) : Exception Prevention

**Description**: Utilisation de classes pour définir le comportement d’exception.

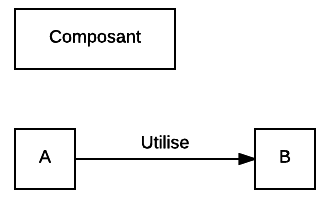
**Justification**: Permet de récupérer le système de manière transparente

## 6.4 Vue architecturale (à l’extérieur de la classe)

### Diagramme



### Légende



### Texte de description du diagramme

Dans le diagramme précédent, nous voyons les entités qui servent à offrir un service très fiable. Le concept est simple: Un moniteur (Monitor) s’occupe de surveiller les trois serveurs périodiquement pour s’assurer qu’ils sont disponible. Quand un des serveurs ne répond pas au moniteur, ce dernier lancera une exception et procédera au redémarrage du serveur. Les trois serveurs font de la redondance active, donc si un serveur ne répond pas ou n’arrive pas à traiter une requête, cette requête sera transférée vers un autre serveur. Le manager est l’entité qui reçoit les requêtes et qui communique avec le moniteur pour s’assurer qu’elles soient traité le plus rapidement possible.

### Table de description des éléments du diagramme

Serveur d’application: Serveur A, B et C

Serveur de balancement: Monitor et Manager

### Texte décrivant la relation entre les éléments et les tactiques

Monitor et les serveurs proviennent de la tactique Monitoring.

Monitor, Manager et les serveurs proviennent de la tactique de redondance active.

Monitor + TimeoutException proviennent de la tactique Exception Prevention.

# 7 Scénario : Beaucoup trop d’ordres de transactions pour le système via SAIL et FIX causent des timeouts

**Objectif d'affaire**

Disponibilité de 99,98%

**Source**

Les bourses dans le monde entier, les clients et les autres machines qui font des appels aux APIs.

**Stimulus**

Les requêtes retournent un timeout (faute).

**Artéfact**

Processus, processeurs, canaux de communication.

**Environnement**

Opérations normales durant les heures d’ouverture prescrites.

**Réponse**

Faire un log de l’incident avec le plus d’informations possibles pour le débogage à l’avenir.

Avertir tous les membres concernés par la panne.

**Mesure de la réponse**

La totalité du temps pendant lequel le système était en mode dégradé.

## 7.1 Tactique

## 7.2 Tactique 1 : Modèle prédictif

**Description**: Utiliser les données historiques pour anticiper les pics d’activité.

**Justification**: En créant un modèle prédictif de la charge sur les serveurs de la bourse, nous pouvons anticiper l’utilisation des serveurs avant même que le pic n’arrive. Ce faisant, nous pouvons avoir la matériel et les ressources nécessaires prêtes lorsque le pic survient.

## 7.3 Tactique 2 : Séparer les services FIX et SAIL

**Description**: Séparer les services FIX et SAIL, sur des machines différentes.

**Justification**: En séparant les services, une panne ou surcharge d’un des deux systèmes n’affectera pas l’autre ou le fera minimalement. Cela ne règle pas le problème mais permet plutôt de mitiger les effets.

## 7.4 Tactique 3 : Redondance Active

**Description**: Nous allons créer un système de redondance active pour gérer la charge de travail dynamiquement.

**Justification**: Nous allons avoir un cluster de machines sur lesquelles la charge pourra être répandue avec un répartiteur de charge. Se faisant, nous allons prévenir les timeouts en ne surchargeant pas les machines et en ajoutant dynamiquement d’autres machines lorsque nécessaire.

## 7.5 Diagramme vue architecturale

### 

## 7.6 Légende

Les hexagones représentent des requêtes suivant un protocole donné provenant de l’extérieur du serveur. Ces requêtes sont typiquement émises par les autres bourses ou des clients de la bourse.

Le rectangle représente une façade externe à laquelle les entités externes effectuent leurs requêtes.

Les flèches représentent la direction d’une requête.

Les rectangles arrondis représentent des machines qui répondent à une requête pour un protocole donné.

## 7.7 Texte de description du diagramme

Ce diagramme représente la tactique de redondance active que nous voulons adopté. En utilisant un cluster de machines auquel des machines peuvent être ajouté dynamiquement, en plus d’avoir séparé les services sail et fix sur des serveurs spécialisé dans le traitement de telles requêtes, nous empêchons la présence de timeout, puisqu’il nous est possible d'augmenter le nombre de serveurs en temps réel.

## 7.8 Table de description des éléments du diagramme

|  |  |
| --- | --- |
| Requête | Représente une requête pour un protocole donné. Par exemple, SAIL pour les clients de la bourse et FIX pour les communications inter-bourses. |
| Load balancer | Distribue les requêtes sur leur machines en fonction de leur capacité et de leur type. |
| Machine | Traitent la requête qui leur a été envoyée par le load balancer |

## 7.9 Texte décrivant la relation entre les éléments et les tactiques

Le load balancer permet d’implémenter la redondance active en répartissant la charge sur les différentes machines et en ajoutant des machines dynamiquement selon la charge et le modèle prédictif. De plus, les machines sont dédiées à SAIL ou à FIX, qui ont chacun leur propre load balancer, les services sont donc séparés.

# 8 Scénario : Assurer une disponibilité de 99.98%

|  |  |
| --- | --- |
| **Objectifs d'affaires** | Assurer une disponibilité de 99.98% |
| **Source** | Systèmes *SAIL* et *FIX* |
| **Stimulus** | Le système crash pendant la transaction |
| **Artéfact** | Réseau |
| **Environnement** | Overload du trafic entrant |
| **Réponse** | * Notifier l’administrateur réseau, ainsi que la personne effectuant la transaction. |
| **Mesure de la \_ réponse** | * Le temps de disponibilité |
| **Questions** | 1. Aurait-il été possible d’éviter le crash?  2. Combien de transactions ont été touchés par ce crash?  3. Pendant combien de temps le système sera affecté? |

## 

## 

## 8.1 Tactique 1 : Ping/Echo

## 8.2 Description : Connaître le temps de réponse et la disponibilité d’un système

**Justification :** Cette tactique permet de s’assurer que l’on respecte les spécifications en prenant une mesure de temps de réponse en plus de s’assurer que le système est disponible

## 8.3 Tactique 2 : Retry

## 8.4 Description: On redémarre la transaction qui a échoué.

**Justification:** En ré-essayant la transaction, nous pouvons surmonter le problème d’indisponibilité du système. Si le système n’a pas été disponible pour une fraction de secondes, cette tactique sera rapide et efficace.

## 8.5 Tactique 3 : Predictive Model

## 8.6 Description: On cumule plusieurs données statistiques par rapport aux équipements et à leur utilisation.

**Justification:** Avec une analyse du trafic sur les systèmes, nous serons en mesure de modéliser les tendances d’accès au service en fonction de l’heure. Ainsi, nous pourrons augmenter la puissance de calcul ou voir à ce que le trafic soit dirigé adéquatement. De plus, tenir en compte de la durée de vie de l’équipement réseau et le storage permettra de remplacer au moment juste les équipements avant que bris n’arrive.

## 

## 

## 8.7 Vue architecturale (à l’extérieur de la classe)

### Diagramme

### 

### Légende

* Une boîte équivaut à une composante.
* Le lien équivaut à la communication entre les composantes.

### Texte de description du diagramme

Dans le diagramme nous pouvons voir que le moniteur envois des demande d’échos à deux composantes. Ces composantes retournent une réponse permettant d’envoyer une alerte à la composante “Alert” si la réponse ne se fait pas ou que le délais est dépassé.

### Table de description des éléments du diagramme

* SAIL : Interface propriétaire (binaire) de la bourse de Montréal
* FIX: Interface standard de message pour système boursier
* Market Surveillance: Affiche toutes les commandes en direct et les historiques par entreprise ou instrument. Permets à la Surveillance du marché annuler les commandes / devis pour une entreprise ou un commerçant donné.
* Market operations : Gestion des alertes par instrument ou firme

### Texte décrivant la relation entre les éléments et les tactiques

La composante *Market Surveillance* est responsable de vérifier l’état des deux services critiques, soit *SAIL* et *FIX*. Cette composante effectue la vérification à l’aide d’une tactique **ping/echo**. Si un problème doit être communiqué, *Market Operations* sera contacté.

Suite à un problème de communication, un hôte pourra réessayer la communication avec l’un de ces systèmes. Cela démontre alors l’utilisation de la tactique **retry.**

Enfin, dans le but de prévenir toute problématique causée par un nombre trop élevé de communications, une **analyse prédictive** sera faite afin de modéliser le comportement des utilisateurs et ainsi pouvoir ajuster les ressources, au besoin.

# 9 Scénario : L’interface du Market Surveillance n’a plus de courant.

|  |  |
| --- | --- |
| **objectifs d'affaires** | Je veux que mon système soit disponible 99.98% du temps. |
| **Source** | Heartbeat monitor |
| **Stimulus** | Panne d’électricité |
| **Artéfact** | Serveur, le guichet, systèmes de communication |
| **Environnement** | Opération normale |
| **Réponse** | Alerte les techniciens du système |
| **Mesure de la réponse** | * % du temps ou le système est non-opérationnel * Temps de détection du système en panne * Temps de réparation du système en panne |
| **Questions** | 1. Par quel moyen est-ce que le système alerte les techniciens? 2. Quel est le temps d’intervention moyen? 3. Quelle est la stratégie de réparation du système? 4. Est-ce que le système de Market Surveillance possède un générateur? 5. Faut-il un autre centre de données en réplication continue? |

## 9.1 Tactique 1 : Détection -TimeStamp

**Description**: Ajout de la date et de l’heure sur toutes les opérations.

**Justification**: Cela permet de détecter les espaces dans le temps entre les opérations. Nous pouvons ainsi détecter une anomalie et de reprendre au bon moment lors de la récupération.

## 9.2 Tactique 2 : Détection - Heart Beat

**Description**: Le système envoie périodiquement un signal pour donner l’état du système.

**Justification**: Cela permet de détecter, à un délai fixe, lorsque le système serait en panne ou non disponible.

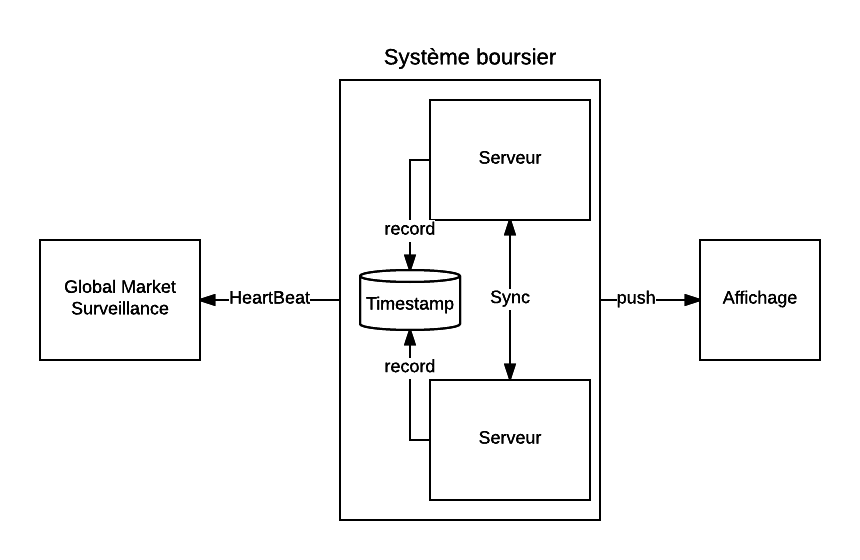
## 9.3 Tactique 3 : Récupération - Redondance externe

**Description**: Tenter de communiquer avec le serveur redondant afin de permettre la continuité de la surveillance du marché.

**Justification**: Cela permet de continuer à afficher les commandes et les historiques en temps réel et de pouvoir annuler les commandes ou devis pendant que le premier serveur est remis en marche.

## 9.4 Vue architecturale

### Diagramme



### Légende

 : Système

: Lien

 : Base de données

### Texte de description du diagramme

On a représenté notre vue architecturale à l’aide d’un diagramme de la famille «allocation» et on pourrait même dire que le style déploiement a été utilisé. Les éléments graphiques utilisés diffèrent un peu de ce que l’on retrouve souvent dans la famille «allocation», mais une légende vient préciser les composants. Le diagramme vient donc présenter les systèmes qui interagissent entre eux afin de démontrer les 3 tactiques que nous avons proposées à notre scénario. On y retrouve donc autant les tactiques de détection de panne que la tactique de récupération par redondance avec les 2 serveurs.

### Table de description des éléments du diagramme

**Serveur**: Module contenant le matériel serveur et son logiciel. Il peut se synchroniser avec d’autres.

**Global market Surveillance**: API externe fournissant les données provenant du marché mondial.

**Timestamp**: Base de données enregistrant les dates associées aux informations reçues.

**Affichage**: Module gérant l’affichage des informations provenant du marché. Il contient les écrans ainsi que les ordinateurs connectés fournissant l’image.

### Texte décrivant la relation entre les éléments et les tactiques

Pour représenter les tactiques expliquées plus tôt, nous avons représenté dans notre vue les différents systèmes nécessitant une interaction entre eux. Donc, dans le cas de la tactique de détection avec Heartbeat, on voit que le système boursier envoie un signalement comme quoi il est toujours actif au système de *Global Market Surveillance*. Dès que ce dernier n’a plus de nouvelles de notre système boursier, cela signifie qu’il est en panne (coupure d’électricité dans notre cas). Ensuite, on voit que pour mettre en place la tactique 2 de détection avec Timestamp, on a une base de données qui garde de façon persistante les informations de temps de chacune des opérations se déroulant sur le système boursier. Lors de la détection d’une longue période de temps sans aucune mise à jour de la base de données des opérations, on peut en conclure une panne du système encore dans ce cas. Finalement, pour présenter la tactique 3 de récupération avec redondance, nous avons placé 2 serveurs en parallèle qui sont synchronisés. Dans le cas d’une panne d’un des serveurs causée par une panne de courant, le 2e serveur pourra prendre le relai et ainsi garder le système boursier disponible. Notons que les 3 serveurs ne sont pas sur la même source de courant.

https://docs.google.com/drawings/d/srEYNtHJ9PAvzsoMRYQyWVw/image?w=421&h=36&rev=1&ac=1

# 

# 10 Scénario (problématique) : Perte de communication avec l’interface SAIL pour recevoir les transactions.

|  |  |
| --- | --- |
| **objectifs d'affaires** | Maintenir une disponibilité de 99,98% |
| **Source** | Interface SAIL |
| **Stimulus** | Omission |
| **Artéfact** | Communication channel or process |
| **Environnement** | Normal operation |
| **Réponse** | notify entities (detect), Prevent the fault from becoming a failure (recover) |
| **Mesure de la réponse** | Mesurer la disponibilité réseau et du service de l’interface SAIL |
| **Questions** | 1. Qui prend action lors de la réception d’une détection de faute? 2. L’interface SAIL est-elle sous forme de système distribué? 3. Qui gère le canal de communication avec l’interface SAIL? 4. Quel est le niveau de service (SLA) du canal de communication? |

## 10.1 Tactique 1 : PING/ECHO

**Description**: Le logiciel de traitement des ordres doit connaître en permanence l’état de communication réseau avec l’interface SAIL (ou son interface réseau). Le système de traitement des ordres envoie un message *PING* vers le système SAIL qui lui répond avec un *ECHO*, confirmant ainsi l’état de la connectivité. En cas de faute, le « helpdesk » peut être averti et/ou des mécanismes de mitigation peuvent être enclenchés rapidement.

**Justification**: Cette tactique permet de détecter plusieurs fautes sur le canal de communication. La non-réception d’un *ECHO* permet d’identifier des pertes de communication. Les métriques telles que la latence (délai entre l’envoi d’un *PING* et la réception de l’*ECHO*) permettent d’identifier des problèmes de performance. Cette tactique permet aussi de définir une certaine sensibilité quant à la définition d’une faute (ex. : 5 non-réception de *ECHO*, latence de plus de 2000ms, etc.) avant de notifier les intervenants ou mettre en action les mesures de mitigation.

## 10.2 Tactique 2 : Heartbeat

**Description**: Le service de l’interface SAIL envoie au système de traitement des ordres un message à un certain intervalle de temps pour confirmer son bon fonctionnement. En cas de faute, le « helpdesk » peut être averti et/ou des mécanismes de mitigation peuvent être enclenchés rapidement.

**Justification**: Au même titre que le *PING/ECHO*, la non-réception de ce message permet de notifier les intervenants et/ou mettre en action des mesures de mitigation. Cela permet non seulement de détecter les fautes du canal de communication, mais aussi les fautes du service SAIL.

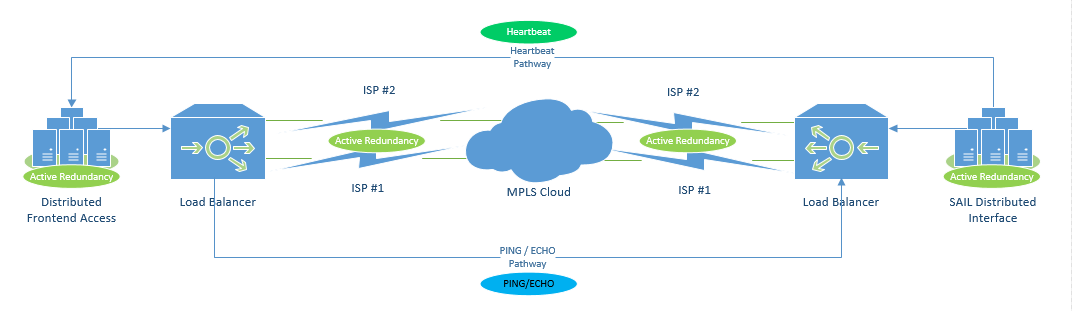
## 10.3 Tactique 3 : Active Redundancy / Loadsharing

**Description**: L’interface SAIL est conçue en système distribué, distribuant le service sur plusieurs plateformes identiques. Aussi, le canal de communication est distribué sur des médiums différents afin de garantir un chemin fonctionnel en cas de faute sur l’un de ceux-ci. Plusieurs approches sont possibles du côté de l’interface SAIL; soit un serveur primaire et plusieurs secondaires prêts à prendre la relève ou plusieurs serveurs qui se distribuent la tâche selon un algorithme quelconque (*round robin, élection, etc.)*. Cette même approche peut aussi être mise en pratique du côté du système de traitement des ordres.

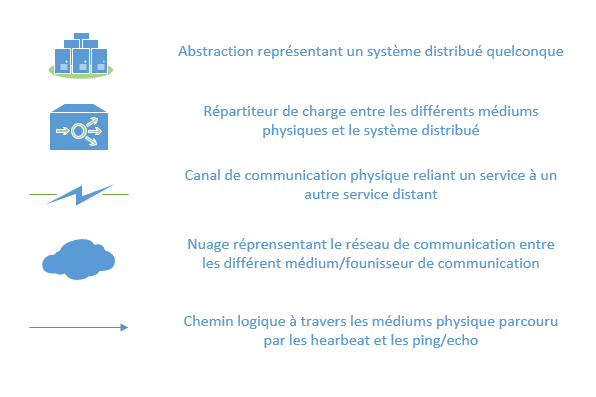
**Justification**: Cette tactique permet de virtuellement garantir une disponibilité de 100% entre l’interface SAIL et le système de traitement des ordres. Plus les serveurs de services sont répartis sur des lieux physiques différents et les canaux de communication répartis sur des médiums différents avec chemins distincts, plus la disponibilité augmente.

## 10.4 Vue architecturale (à l’extérieur de la classe)

### Diagramme



### Légende



### Texte de description du diagramme

Le diagramme illustre les différentes abstractions des composantes reliant un système à un autre ainsi que les chemins logiques parcourus par les messages de contrôle. Il est à noter que les messages logiques parcourent les médiums physiques illustrés, mais sont annotés de façon séparée pour faciliter la lisibilité. La flèche indique le sens initial de la requête et la réponse parcourt le chemin inverse.

### Table de description des éléments du diagramme

|  |  |
| --- | --- |
| Élément | Responsabilité |
| Système distribué | Le système distribué doit gérer les différentes instances du même service et assurer la réponse à une requête parmi l’une des instances selon un algorithme défini. Assure une redondance active entre les instances du même service. |
| Répartiteur de charge | Permets d’abstraire les différents canaux de communications pour le système distribués et possiblement répartir la charge entre ceux-ci (facultatif). |
| Canal de communication | Assure la transmission des données entre deux points via un médium physique.  Chaque canal doit idéalement être de médium physique distinct et de fournisseurs différents. Assure une redondance active entre les différents canaux de communication. |
| Nuage | Représente l’abstraction de l’infrastructure permettant la communication (routage, commutation, etc.) entre différents fournisseurs de service du point A au point B. |
| Heartbeat | Ensemble des requêtes/réponses entre les instances d’un service. Facultatif si le système est entièrement distribué, mais essentiel si le service est composé d’une instance primaire et d’instances secondaires (afin de savoir quand basculer au secondaire). |
| Ping/Echo | Ensemble des requêtes/réponses entre les répartiteurs de charge, pour chaque canal de communication. Cela permet au répartiteur de charge de détecter les fautes sur un médium de communication et basculer la transmission vers un autre, le cas échéant. |

### Texte décrivant la relation entre les éléments et les tactiques

Les répartiteurs de charge communiquent entre eux sur chaque canal de communication à l’aide de *PING/ECHO* afin de détecter les fautes et basculer au besoin. Il s’agit en même temps d’un système de redondance active.

Le service SAIL courant envoie des *heartbeat* avec le système de traitement des ordres afin de communiquer sa disponibilité. Cela permet au système de traitement des ordres de basculer à un service SAIL secondaire en cas de défaillance ou de prendre toute autre action nécessaire.

Les différents systèmes distribués agissent en tant que redondance active en ayant plusieurs instances du service disponible à la fois. Un seul service peut répondre ou bien un algorithme peut distribuer cette charge parmi les différentes instances. Dans ce dernier cas, le *heartbeat* indiquerait la faute d’un service sans avoir à basculer vers un autre service. De plus, la duplication des canaux de communications peut agir en tant que redondance active si tous sont utilisés en même temps, sinon il s’agit d’une redondance passive.

# 11 Scénario : Devrait avoir une disponibilité de 99.98

|  |  |
| --- | --- |
| **Objectifs d'affaires** | Devrait avoir une disponibilité de 99.98% |
| **Source** | Interne: interface qui sert à utiliser FIX |
| **Stimulus** | Omission/crash du service |
| **Artéfact** | Processus |
| **Environnement** | En mode d'opération normale |
| **Réponse** | Prévenir une faute de devenir une panne  Détecter la faute   * Enregistrer les détails de la faute * Ouvrir un billet sur la faute * Identifier les composant(s) fautifs   Reprise après la faute   * Faire une analyse d’impact * Corriger l’anomalie * Effectuer des tests |
| **Mesure de la réponse** | Mesurer le temps de non disponibilité du module affecté par l’erreur  Fréquence d’interruption du service pour un module dû à certaines erreurs  Mesure des impacts de la faute |
| **Questions** | 1. D’où provient la demande à *Checkout* et où est envoyé la traduction de *Checkout*? 2. Veut-on que *Checkout* communique directement le résultat de la traduction au destinataire, ou doit-il passer par *Manager*? |

## 11.1 Tactique 1 : Battement cardiaque (Heartbeat)

**Description** : Cette tactique consiste à vérifier qu'un module demeure "en vie" en écoutant son "battement cardiaque" à partir d'un surveillant de système. Le surveillant envoie périodiquement un message au module surveillé, et ce dernier confirme qu'il est toujours en fonction. Quand le temps de réponse dépasse un certain temps, le module est considéré comme hors fonction.

**Justification** : Le module servant à transtyper les messages entre les interfaces SAIL et FIX est vital au transactions entre la bourse de Montréal et les différentes institutions financières. Il est donc primordial de vérifier que ce module soit accessible en tout temps.

## 11.2 Tactique 2 : Redondance active (Active redundancy)

**Description** : Plusieurs modules identiques font les mêmes opérations en parallèle , ce qui ne compromet pas la qualité du service en cas de panne.

**Justification** : Afin d’éviter qu'une faute du module compromette la disponibilité de l'entièreté du système, nous avons opté pour un mécanisme de redondance active. Plusieur serveurs font la gestion des version des messages FIX et SAIL en même temps. Ainsi, lorsqu'un serveur est défaillant, un autre serveur peut immédiatement prendre le relais. Ce type de redondance est plus coûteux qu'une redondance passive, mais la rapidité étant prioritaire dans le marché boursier, il est important de ne perdre aucun temps dans la mise en fonction d'un nouveau serveur.

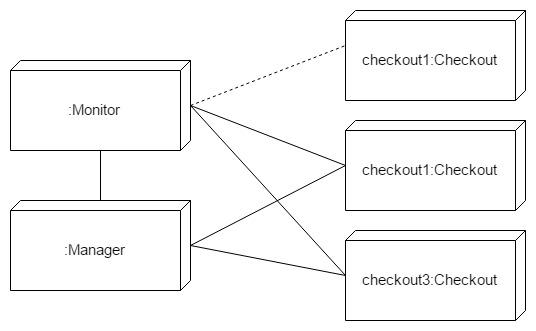
## 11.3 Tactique 3 : Removal from service

**Description** : Lorsqu’il y a une faute sur un module, celui-ci est retiré de ses fonctions le temps que la faute soit corrigée.

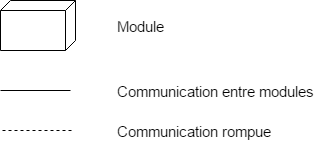
**Justification** : L’utilisation de la tactique de redondance active permet aisément de retirer un module puisque les autres assurent la même qualité de service.

## 11.4 Vue architecturale (à l’extérieur de la classe)

### Diagramme



### Légende



### Texte de description du diagramme

Dans ce diagramme, le module *checkout1* ne donne plus de *Heartbeat* à *Monitor*; il considère donc qu'il est inactif. Les deux autre modules permettent de continuer d’effectuer les opérations normales. *Monitor* informe *Manager* que seuls *checkout2* et *checkout3* sont encore actifs, suite à quoi *Manager* communique avec ces derniers.

### Table de description des éléments du diagramme

|  |  |
| --- | --- |
| **Élément** | **Description** |
| Monitor | Module responsable de vérifier l'état des instances de Checkout. Il envoi un message à ces derniers selon un interval régulier et une réponse des modules surveillés (un "heartbeat") lui confirme qu'ils sont toujours en fonction. Le Monitor sait donc quels modules sont actifs. |
| checkout1, checkout2 et checkout3 | Module redondant de Checkout. Puisqu'il s'agit d'une redondance active, tous les modules en fonction reçoivent les messages à traduire et communiquent leur résultat au Manager. |
| Manager | Module responsable de communiquer avec Monitor pour envoyer les messages à traduire à tous les modules Checkout en fonction. Il utilise ensuite le résultat du premier module en fonction. C'est donc Manager qui se trouve à mettre hors service un module qui est déclaré comme non fonctionnel par Monitor puisque toute communication avec le module inactif est rompue. Pour finir, c'est Manager qui agit en tant qu'interface de traduction pour le reste du système. |
| Lien entre Monitor et les modules Checkout | Monitor envoie un message à tous les modules de Checkout. Ces derniers répondent en confirmant qu'ils sont encore actifs. |
| Lien entre Monitor et Manager | Manager demande à Monitor quels modules de Checkout sont actifs. Monitor répond avec la liste de ces modules. |
| Lien entre Manager et les modules Checkout | Manager envoie chaque message à traduire à tous les modules de Checkout actifs selon Monitor. Les modules de Checkout répondent avec le message traduit dans le format désiré. |

### Texte décrivant la relation entre les éléments et les tactiques

Les trois modules de Checkout permettent la redondance active du système (dans le diagramme, le premier est hors service mais les deux autres continuent de fonctionner normalement sans interrompre le service). ~~Le module de Monitor permet d’appliquer la tactique de~~ *~~Heartbeat~~* ~~en surveillant les différents modules de Checkout. Le Manager permet d’appliquer la tactique de~~ *~~Removal from service~~*. Le Manager communique avec le Monitor pour savoir quelle instance de checkout est *inactive* pour la retirer par la suite.

# 12 Scénario: Il est impossible de se connecter au système de bourse, car le serveur n’est pas disponible

|  |  |
| --- | --- |
| **Objectifs d'affaires** | Disponibilité du système de bourse à 99.98% durant les heures d’ouverture |
| **Source** | Serveur |
| **Stimulus** | Aucune communication du serveur (ne réponds pas) |
| **Artéfact** | Canal de communication (les connexions réseau) |
| **Environnement** | Opération normale |
| **Réponse** | * Informer le client de continuer d’opérer |
| **Mesure de la réponse** | * Temps pour détecter la faute: inférieur à 1 min. * Temps pour effectuer la réparation: moins de 1 min |
| **Questions** | 1. Combien de fois faut-il réessayer la connexion? 2. Est-ce que le problème est encore présent? 3. Combien de temps le système prend pour démarrer un serveur en backup? |

**Tactique 1 "Detect Faults" : Ping/Echo**

**Description**: Le monitor envoie des ping au serveur du système de bourse pour s’assurer que celui-ci est accessible.

**Justification**: L’utilisation de cette tactique permet de détecter rapidement un problème lié à l’accessibilité du système de bourse et donc, une réparation plus rapide et un retour de la disponibilité du système.

**Tactique 2 "Recover from faults" : Retry**

**Description**: Le système réessaye de se connecter au serveur.

**Justification**: Il est possible que la connexion au système de bourse échoue dû à une anomalie. D’autres tentatives pourraient réussir. Le nombre de tentatives doit être limité afin de détecter un problème majeur.

**Tactique 3 "Recover from faults" : Redondance passive**

**Description**: Un serveur en backup recevra des mises à jour du serveur principal. Lorsque le serveur principal présente des problèmes ou ne répond pas, le serveur en backup s’active et prend la relève du serveur principale.

Dans le cas où le ping n’atteint pas sa destination de manière permanente, le dispatcher va lancer le ou les serveurs redondants afin de rétablir le système.

**Justification**: La redondance passive permet de préserver un haut niveau de disponibilité, puisque le système sera hors ligne pour quelques minutes, le temps que les serveurs redondants s’activent.

## 12.1 Vue architecturale

### Diagramme et légende

### 

### Texte de description du diagramme

Lorsque le “monitor” ne reçoit plus de signal avec les serveurs actifs, celui-ci va avertir le dispatcher pour qu’il active le ou les serveurs redondants.

### Table de description des éléments du diagramme

**Client**: Les clients externes qui utilisent le système de bourse.

**Serveur prime**: Le serveur actif du système de bourse pour accomplir les transactions.

**Serveur spare**: Les serveurs inactifs redondants du système de bourse s’activeront dans le cas ou le serveur prime ne répond pas.

**Monitor** : Le système qui envoie des pings à chaque intervalle pour avoir la condition des serveurs actifs.

Dispatche**r**: Le système qui a la responsabilité de créer la connexion de l’extérieur du système au serveur interne correspondant.

### Texte décrivant la relation entre les éléments et les tactiques

**Tactique Ping/Écho:**

Le monitor envoie des signaux à chaque intervalle au serveur actif du système de bourse afin de connaître si le serveur est toujours disponible pour prendre des requêtes.

**Tactique Retry:**

Si le serveur actif n’est pas capable de répondre après un nombre d’écho déterminer le dispatcher va activer des serveurs.

**Tactique Redondance Passive:**

Les serveurs redondants ne sont pas actifs tant et aussi longtemps que le serveur prime est fonctionnel et optimal.