LOG430 – Architecture logicielle

# Équipe 02

Scénarios de qualité, tactiques et justifications des attributs produits parles équipes en classe

## <Disponibilité> – Justification 1

Le manque de billets dans le guichet automatique est important à la fois pour le client et pour la banque. La satisfaction du client sera affectée car le retrait ne pourra pas être fait. De plus, la banque perdra de l’argent puisqu’elle ne pourra pas charger de frais de transaction.

## <Disponibilité> – Justification 2

Il est important de traiter les erreurs de communication avec les systèmes bancaires car si on assume que tout ira bien, des transactions pourraient être perdues et le solde des comptes pourrait ne pas être exact. Advenant une erreur qui aurait des répercussions sur le solde du compte d’un usager de la banque, le client pourrait être très mécontent et la banque verrait des répercussions directes sur sa réputation.

## <Disponibilité> – Scénario 1

**Source** : Transaction de retrait

**Stimuli** : Manque de Billets

**Artefact** : Processus de retrait d’argent du logiciel du guichet.

**Environnement** : Opération normale

**Réponse** :

***Détection de la faute:***

●  Informer le client du manque de billets

●  Avertir l'établissement bancaire

***Récupération****:*

●  Retourner au menu principal du guichet en mode dégradé (retrait non disponible)

**Mesure de la réponse :**

●  Système disponible 99% du temps

●  Temps de détection du problème

­●  Temps de réparation du problème

●  Temps où le système est en mode dégradé

## <Disponibilité> – Scénario 2

**Source** :

● Infrastructure réseau bancaire

○ Routeur

○ Fil réseau

**Stimuli** : Perte de connexion réseau

**Artefact** : Réseau de communication

**Environnement** : Opération normale

**Réponse** :

Détection de la faute :

●  Informer d’une erreur réseau via l’interface utilisateur du guichet

●  Informer les administrateurs à partir du serveur

●  Garder une trace de l’erreur dans les journaux

  Récupération de la faute

●  Rendre le système indisponible

●  Annuler toutes les transactions en cours

●  Réessayer périodiquement d’établir une connexion au réseau bancaire

**Mesure de la réponse :**

●  Système disponible 99% du temps

●  Temps pris pour détecter la faute

●  Temps pris pour réparer la faute

## <Disponibilité> – Tactique 1

**Description** : Condition Monitoring – Faire une vérification du nombre de billets disponibles dans la machine avant de débiter le compte de l’utilisateur.

**Justification** : En faisant une vérification du nombre de billets disponibles, le système s’assure que la machine pourra couvrir la demande. De plus, le système de retrait peut vérifier si le montant désiré par le client est disponible. Finalement, une alerte peut être envoyée a l'établissement bancaire avant qu’il n’y ait plus de fonds disponible

## <Disponibilité> – Tactique 2

**Description** : Heartbeat – Le client envoie régulièrement un message au serveur pour lui indiquer qu’il est encore connecté.

**Justification** : On utilise le heartbeat pour que le serveur puisse détecter dans une fenêtre de 10 secondes si un guichet est hors-ligne. Cela nous permet d'aviser les personnes responsables. Vue la nature incrémentale du heartbeat, on peut vérifier si le problème est très sérieux (plusieurs heartbeats manqués veut dire un problème plus sérieux qu'un seul heartbeat manqué).

## <Disponibilité> – Tactique 3

**Description** : Transactions – Des transactions sont utilisées pour toutes les opérations bancaires.

**Justification** : On utilise des transactions pour éviter d'enregistrer des données partielles. Par exemple, on ne donne pas de billets de 20$ à l'usager à moins que tout ait été complété au niveau bancaire.

## <Disponibilité> – Tactique 4

**Description** : Removal from service – On bloque le fonctionnement de la machine tant que le réseau est indisponible. On affiche un message pour indiquer que le service est interrompu.

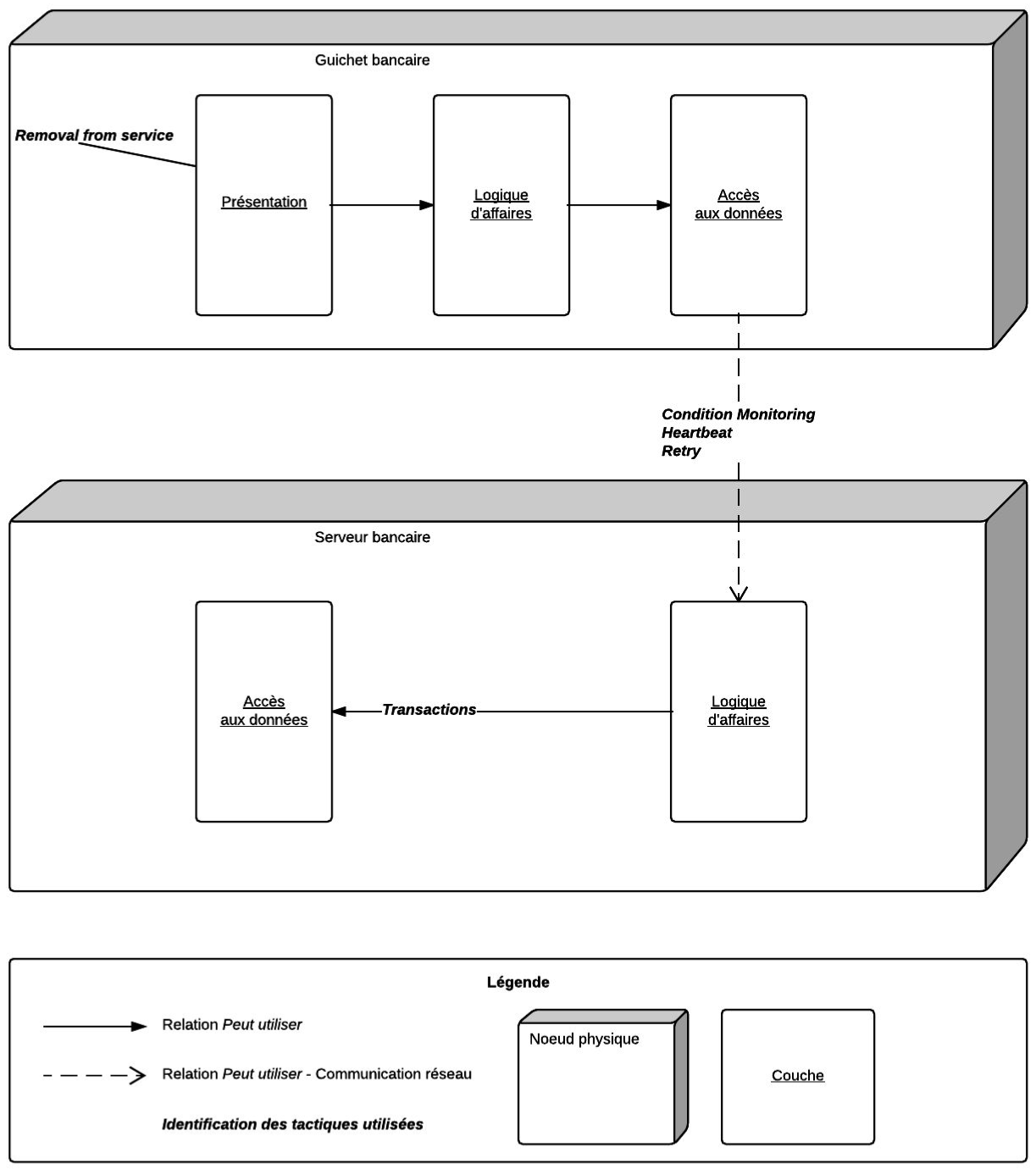
**Justification** : En bloquant tout de suite l’interface en cas de problème et en indiquant l’erreur sur la machine, on évite que l’usager pense que sa transaction a été réalisée avec succès.

## <Disponibilité> – Tactique 5

**Description** : Retry – Le guichet réessaye de se connecter en cas de perte de connexion .

**Justification** : On réessaie de rétablir la connexion périodiquement pour faciliter la reprise des opérations et ainsi éviter d'avoir à mobiliser un technicien et réduire le temps moyen de récupération (MTTR) .

## <Disponibilité> – Vue architecturale



Le diagramme de vue architecturale pour la disponibilité montre une vue d’ensemble de l’architecture du guichet automatique bancaire et intègre l’utilisation des tactiques architecturales pour la disponibilité.

Éléments du diagramme

|  |  |
| --- | --- |
| Élément | Description |
| Guichet bancaire | Le noeud physique correspondant à la machine de guichet automatique bancaire et le logiciel qui y est exécuté. |
| Guichet bancaire / Présentation | La couche Présentation du guichet bancaire comporte les éléments d’interface graphique du guichet. Cette couche est responsable pour la tactique de Removal From Service pour rendre le GAB inaccessible en cas de panne. |
| Guichet bancaire / Logique d’affaires | La couche Logique d’affaires du guichet bancaire comporte la logique interne du guichet bancaire, soit une petite partie de la logique bancaire qui peut être faite directement sur le guichet. |
| Guichet bancaire / Accès aux données | La couche d’Accès aux données sur le guichet bancaire sert essentiellement à accéder aux données du serveur bancaire distant. Elle est aussi responsable d’intégrer les tactiques  Condition Monitoring , Heartbeat et Retry lors de sa communication avec le serveur bancaire. |
| Serveur bancaire | Représente le serveur bancaire qui exécute un logiciel pour traiter les données bancaires ainsi qu’une base de données transactionnelle. |
| Serveur bancaire / Logique d’affaires | La couche de Logique d’affaires du serveur bancaire permet de faire les traitements demandés par le guichet (transactions entre comptes, débits, crédits, etc.). Elle utilise la tactique T ransactions pour communiquer avec la couche d’Accès aux données. |
| Serveur bancaire / Accès aux données | La couche d’Accès aux données du serveur bancaire permet de gérer les données bancaires de manière transactionnelle. |

Tel qu’illustré sur le diagramme, les tactiques sont implémentées dans des éléments précis du système à des endroits stratégiques et naturels.

Les diverses tactiques implémentées dans le système illustré permettent de régler les deux problématiques établies dans Justification 1 et Justification 2.

# Équipe 04

## <Disponibilité> – Justification (en classe)

• Assurer au client la gestion de son compte bancaire en tout temps.

• Assurer que la machine ATM est un montant suffisant pour compléter la transaction.

## <Disponibilité> – Scénario 1 (en classe)

**Source**: Le client désire retirer un montant que la machine ATM ne  détient pas.

**Stimuli**: Le système va répondre au client qu’il y a un problème X et qu’il peut seulement exécuter un retrait d’un montant X inférieur.

**Artéfact**: Processus de détection du montant dans ATM

**Environnement**: Système fonctionnel, manque d’argent

**Réponse**: Le système affiche à l’écran qu’il y a un manque de fonds pour répondre à sa requête. Possibilité d’alerte au mainteneur de la machine ATM.

**Mesure de la réponse**: Le temps attribué à ce que la machine ATM soit fonctionnelle.

## <Disponibilité> – Scénario 2 (en classe)

**Source**: Communication avec la base de données

**Stimuli**: Disfonctionnalité au niveau de la communication avec la base de données.

**Artéfact**: Processus de communication avec la base de données

**Environnement**: Système est fonctionnel

**Réponse**: Couper le service, afficher un message au client d’aller au comptoir, alerter la maintenance de cette erreur.

**Mesure** **de la réponse**: Le temps qu’un technicien répare le problème de communication avec la base de données.

## <Disponibilité> – Tactique 1 (en classe)

**Description**: Predictive model + Monitor +  Exception handling

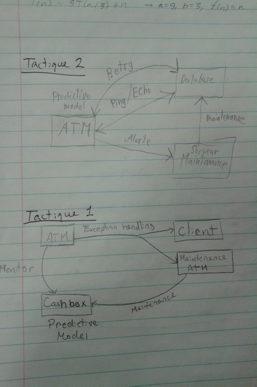
**Justification**: Predictive model : Afin de prévenir la possibilité dans laquel le client voudrait retirer un montant supérieur à ce que la machine contient. Ensuite, en cas ou le client désire retirer un montant plus élevé que le montant dans la machine ATM, celle-ci lui affiche à l’écran.

## <Disponibilité> – Tactique 2 (en classe)

**Description**: Predictive model + Retry + Ping Echo

**Justification**: Pour commencer, il est nécessaire de vérifier l’accessibilité de la base de données, ensuite, le \*predictive model\* s’assure qu’il peut y avoir un échange d’information avec le serveur qui contient la base de données. Pour finir, la tactique Retry est engagé lorsqu’il y a une coupure de communication en pleins milieu d’une transaction.

## <Disponibilité> – Vue Architecturale (en classe)



•  Diagramme

•  Légende

•  Texte de description du diagramme

•  Table de description des éléments du diagramme

•  Texte décrivant la relation entre les éléments et la tactique

•  texte décrivant comment la tactique répond a chacun des scénarios de qualité.

# Équipe 06

## <Disponibilité> – Justification (en classe)

On risque de perdre des clients

Interruption de transactions en cours

Besoin de se rendre au comptoir

Peut-être fermé à cette heure

Demande plus de temps et de personnel

Le guichet qui est en panne peut ne pas rendre la carte (assistance technique nécessaire)

Moins de profits pour la banque

## <Disponibilité> – Scénario 1 (en classe)

**Source**: Client qui utilise le guichet

**Stimuli**: Introduire un objet dans la fente qui accepte des enveloppes de dépôts qui bloque le mécanisme.

**Artéfact**: Module de réception des enveloppes de dépôts

**Environnement**: Le guichet est en opération

**Réponse**: Fonctionner en mode dégradé, en permettant les retraits mais pas les dépôts. Envoyer une requête de service à un technicien.

**Mesure de la réponse**: Détecter dans 99% des temps si le module qui accepte les enveloppes est bloqué ou non. Temps que le système reste en mode dégradé, en attendant la réparation.

## <Disponibilité> – Scénario 2 (en classe)

**Source**: Le client qui utilise le guichet

**Stimuli**: Suite à un retrait, il reste moins de billets que le quart de la capacité maximale

**Artéfact**: Détecteur de la réserve de billets du guichet

**Environnement**: Le guichet est en opération

**Réponse**: Fonctionner en mode dégradé en permettant les dépôts mais pas les retraits. Envoyer un appel de service pour qu’un technicien vienne ravitailler le guichet.

**Mesure de la réponse**: Probabilité que le guichet devienne éventuellement à cours de billets et le temps moyen pendant lequel le guichet est à court de billets.

## <Disponibilité> – Tactique 1 (en classe)

**Description**: Detect fault - Condition monitoring

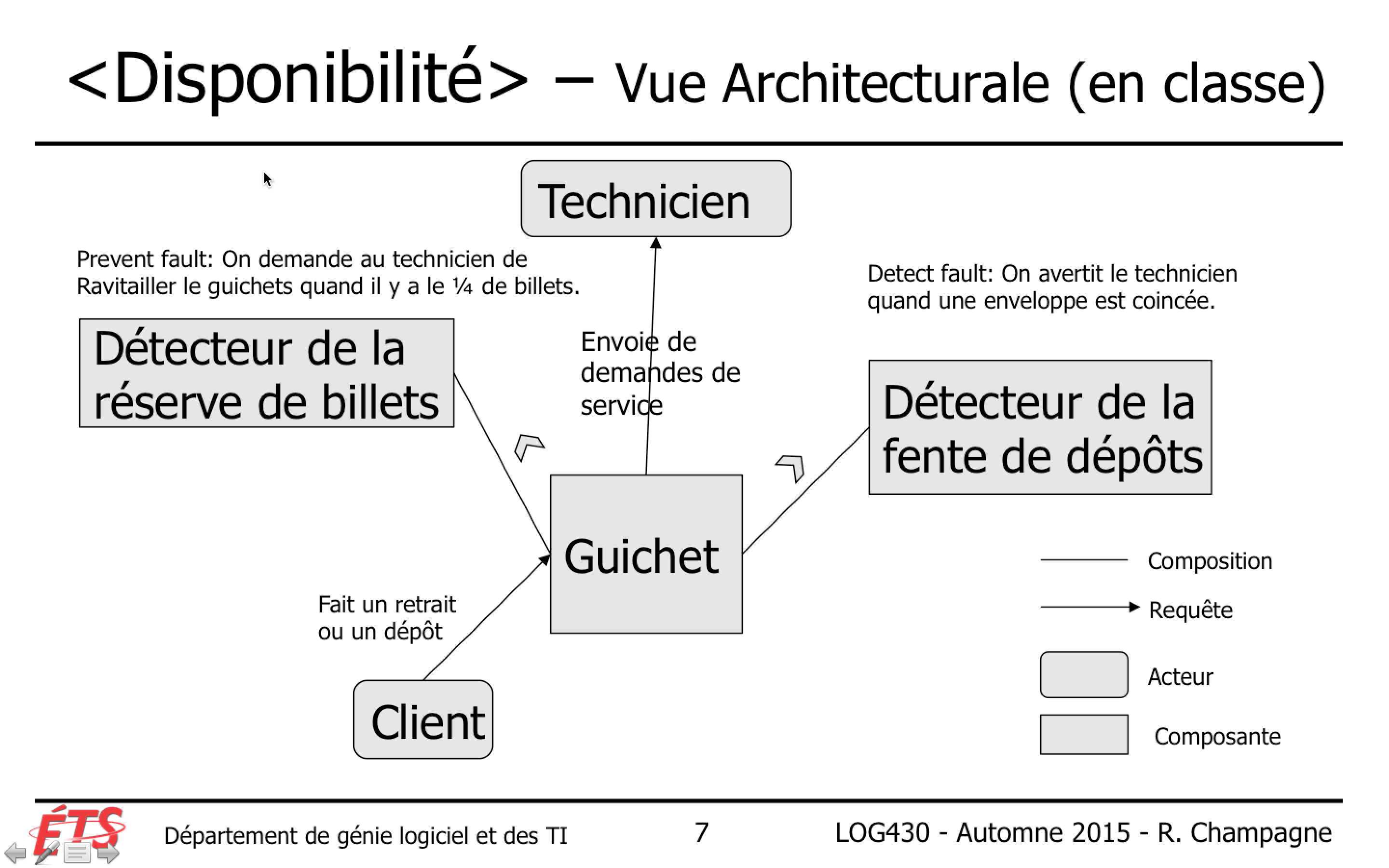
**Justification**: Le guichet vérifie périodiquement à l’aide d’un détecteur optique si un objet reste coincé dans le module qui accepte les enveloppes, ou si le moteur qui permet d’accepter l’envelope n’est plus capable de tourner.

## <Disponibilité> – Tactique 2 (en classe)

**Description**: Prevent fault – Predictive model

**Justification**: On prédit que le temps necessaire au technicien pour venir ravitailler le guichet est inférieur de manière probable au temps qu’il faudra pour que les clients retirent tous les billets restants, de manière à normalement ne pas en manquer.

## <Disponibilité> – Vue Architecturale (en classe)



# Équipe 07

## <Disponibilité> – Justification (en classe)

La disponibilité d'un GAB est très importante d'une part pour le client, qui doit être en mesure de consulter ses comptes et de retirer de l'argent (service qu'il paye par ailleurs), et pour la banque, car un GAB en panne nuit sévèrement à son image.

## <Disponibilité> – Scénario 1 (en classe)

**Source**: Logiciel de fonctionnement du GAB (retrait effectué par un client) – A la fin d'une transaction le compteur d'argent chargé est décrémenté

**Stimuli**: Compteur d'argent chargé dans le GAB passe sous le seuil minimal d'argent chargé (Response : le compteur est trop bas)

**Artéfact**: Processus de contrôle de l'argent chargé dans le GAB

**Environnement**: Environnement normal

**Réponse**: Informer l'opérateur qu'il manque des billets, passer en mode « dégradé » en limitant le montant maximal des retraits, informer le client que le montant maximal des retraits est limité

**Mesure de la réponse**: Mesure du temps pendant lequel le compteur est en dessous du seuil (temps entre lequel l'alerte est envoyé à l'opérateur et l'opérateur remplit le GAB)

## <Disponibilité> – Scénario 2 (en classe)

**Source**: Processus de communication du GAB

**Stimuli**: Omission, le GAB n'arrive plus à recevoir de messages de la part du serveur central de la banque

**Artéfact**: Processus de communication du GAB

**Environnement**: Environnement normal

**Réponse**: Insérer l'erreur dans les logs du GAB, passage en mode dégradé : désactivation de toutes les fonctionnalités sauf le dépôt d'argent

**Mesure de la réponse**: Mesure du temps pendant lequel le GAB est en mode dégradé

## <Disponibilité> – Tactique 1 (en classe)

**Description**: Monitoring – Degradation

**Justification**: le processus de contrôle de l'argent chargé dans le GAB dispose d'un compteur correspondant à l'argent chargé dans le GAB, après chaque retrait le processus de transaction du GAB ordonne au processus de contrôle de l'argent de décrémenter de X dollars son compteur. Si le compteur passe en dessous d'un certain seuil, le système passe en mode dégradé, on limite les retraits à un montant maximal de Y dollars. De plus, le processus de contrôle de l'argent demande au processus de communication de transmettre une alerte demandant à un opérateur de la compagnie bancaire de remplir le GAB. Le processus d'affichage affichera alors systématiquement un message correspondant à ce problème sur l'écran du GAB.

## <Disponibilité> – Tactique 2 (en classe)

**Description**: Exception detection (timeout) – Retry - Reconfiguration – Degradation

**Justification**: Le processus de transaction doit consulter le compte du client, il demande au processus de communication de prendre contact avec le serveur central de la banque. Le processus de communication tente de joindre le serveur central de la banque, après X secondes il n'y a toujours pas de réponse, une exception est levée. Le processus de communication réessaye Y fois de joindre le serveur central. Si au bout des Y fois il n'a pas réussi à établir une communication avec le serveur central, il se reconfigure et change de serveur (on suppose que le processus de communication dispose d'une liste de serveurs à tester en cas d'echec). Si le processus de communication arrive au terme de sa liste de serveurs, il fait passer le GAB en mode dégradé. Ce mode autorise uniquement les opérations de dépôt d'argent. On affiche grâce au processus d'affichage les fonctionnalités indisponibles.

## <Disponibilité> – Vue Architecturale (en classe)

## Équipe 09

## <Disponibilité> – Justification (en classe)

Pour le client, il est important de tenir compte de la disponibilité puisqu’on ne veut pas que le système plante lors d’une transaction. On veut que le client retrouve sa carte, son argent, son paiement, son transfert et autre.

## <Disponibilité> – Scénario 1 (en classe)

**Source**: Logiciel de contrôle de transaction à l’intérieur du guichet automatique.

**Stimuli**: Aucune réponse du système lors d’un retrait de 20$.

**Artéfact**: Processus de transaction.

**Environnement**: Opération normale.

**Réponse**: Un message d’erreur s’affiche, on informe le client du problème et lui demande d’effectuer l’opération à nouveau. On redonne la carte et redémarre le système.

**Mesure de la réponse**: 99.9%, 5 secondes pour détecter la faute, l’erreur ou la défaillance et une minute pour réparer le problème.

## <Disponibilité> – Scénario 2 (en classe)

**Source**: Imprimante, hardware

**Stimuli**: Il n’y a plus d’encre dans l’imprimante.

**Artéfact**: Processus d’impression.

**Environnement**: Opération normale.

**Réponse**: Avant d’effectuer sa transaction, l’utilisateur est informé du manque d’encre. Il est possible d’effectuer la transaction sans recevoir de reçu. Il peut aussi aviser le personnel pour avoir un reçu au comptoir.

**Mesure de la réponse**: 99%, 5 secondes pour envoyer un message au personnel technique et afficher un message à l’utilisateur et une demi-journée pour réparer le problème.

## <Disponibilité> – Tactique 1 (en classe)

**Description**: Monitor

**Justification**: Permet de détecter les fautes dans les processus de transaction du guichet bancaire.

## <Disponibilité> – Tactique 2 (en classe)

**Description**: Heartbeat

**Justification**: Le heartbeat envoie le pourcentage d’encre de la cartouche à chaque 6 heures pour prévenir le changement de cartouche.

## <Disponibilité> – Diagramme



<Disponibilité> – Table de description des éléments du diagramme

****

Ce diagramme représente la communication entre le guichet automatique bancaire et le serveur de la banque.

La première relation représente l’imprimante du guichet qui envoie de l’information (le niveau d’encre de l’imprimante) par la classe Heartbeat à la classe RéceptionHeartbeat du serveur.

La deuxième relation représente le serveur qui monitore le guichet automatique bancaire. Ceci se réalise à l’aide de la classe MoniteurDeTransaction du serveur qui monitore la classe GestionDeTransaction du guichet automatique.

**<Disponibilité> – Texte décrivant comment la tactique répond a chacun des scénarios de qualité**

**Monitor**: Le serveur pourra observer les erreurs reliées au guichet automatique bancaire et pourra en aviser le personnel en support ou bien effectuer les différentes modalités reliées avec la gravure du problème.

Le Heartbeat envoie à un moment configuré (aux 6 heures) de l'information ( le niveau d'encre dans l'imprimante) au serveur. Cette tactique est parfaite pour être certain de ne jamais manquer d'encre dans les guichets automatiques.

# Équipe 10

## <Disponibilité> – Justification (en classe)

Le client doit avoir accès à son argent 24h sur 24h, sinon il pourrait changer de fournisseur.

## <Disponibilité> – Scénario 1 (en classe)

**Source**: logicel ATM

**Stimuli**: un memory leak

**Artéfact**: processus de la gestion de la mémoire

**Environnement**: opération normale

**Réponse**: log l’origine, informe le client et la maintenance. Redémarre l’ATM ou continue le système si possible.

**Mesure de la réponse**: Le temps de disponibilité en %, le temps de récupération et le temps de détection du crash.

## <Disponibilité> – Scénario 2 (en classe)

**Source**: heartbeat monitor

**Stimuli**: une panne de réseaux pour ATM

**Artéfact**: processus de monitoring détecte un problème

**Environnement**: opération normal

**Réponse**: log l’origine du problème et informe l’opérateur.

**Mesure de la réponse**: Le temps de disponibilité en %, le temps de récupération et le temps de détection du crash.

## <Disponibilité> – Scénario 3 (en classe)

**Source**: Transaction ATM

**Stimuli**: une panne de réseaux pour ATM

**Artéfact**: processus de transaction

**Environnement**: opération normal

**Réponse**: log l’origine du problème, informe le client et essaie de se reconnecter au serveur bancquaire.

**Mesure de la réponse**: Le temps de disponibilité en %, le temps de récupération et le temps de détection du crash.

## <Disponibilité> – Tactique 1 (en classe)

**Description**: Récupération de la faute (Exception handling)

**Justification**: L’atm doit écrire la raison du crash. L’atm doit aussi retourné à un état normale le plus vite possible. Il doit donc nettoyer mémoire manquante avec un redémarage de ATM.

## <Disponibilité> – Tactique 2 (en classe)

**Description**: détection de la faute (Heartbeat)

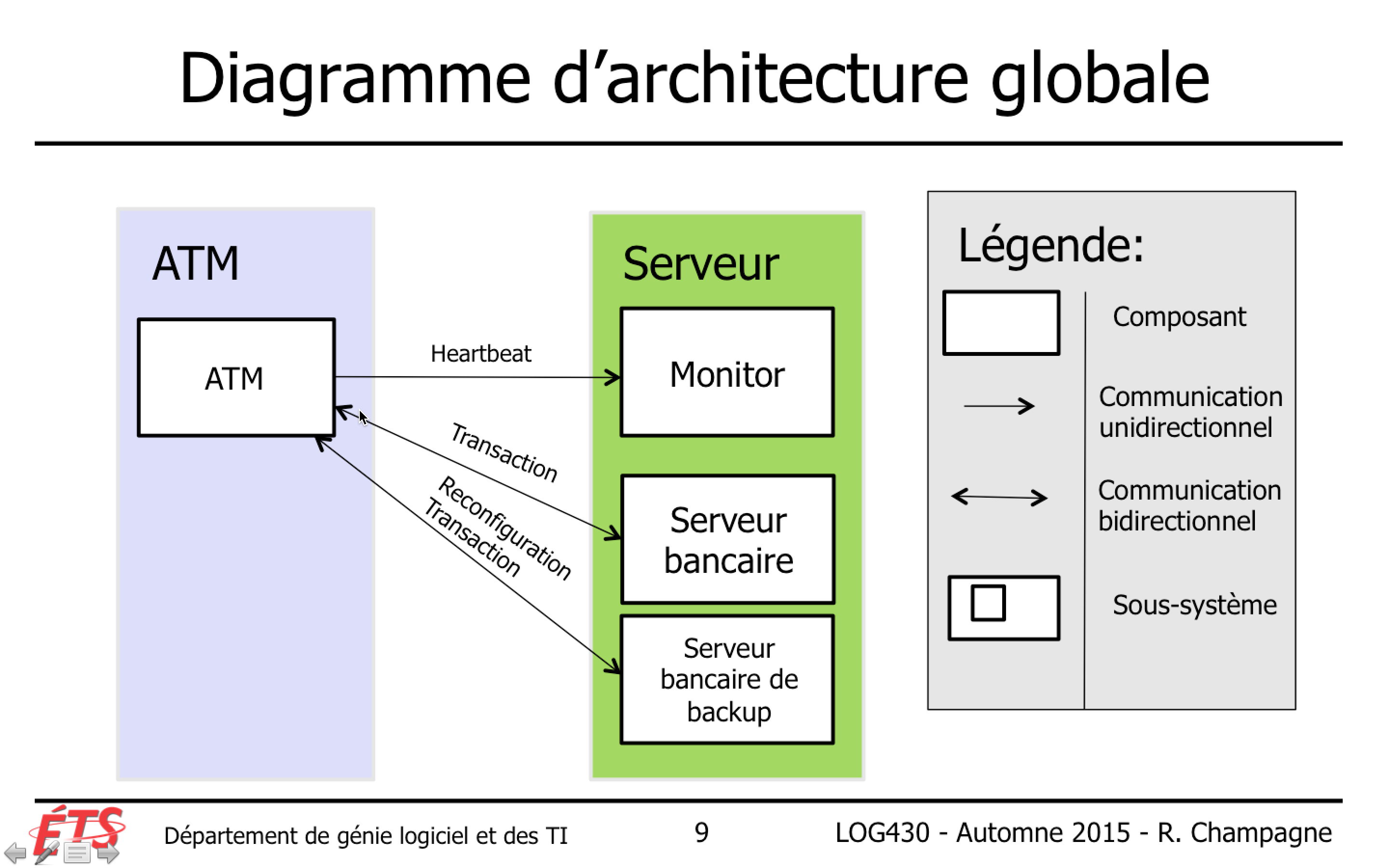
**Justification**: Si l’ATM n’envois pas un état au serveur de monitoring, après un delai définie, un opérateur doit être avertie le plus tôt possible.

## <Disponibilité> – Tactique 3 (en classe)

**Description**: Récupération de la faute (retry/reconfiguration)

**Justification**: L’ATM essaie de se reconnecter au réseau avec des intervale de temps de plus en plus grand, pour finir sa transaction. L’ATM essai aussi différent serveur de transaction.

## Diagramme d’architecture globale



### Description du diagramme globale

Le diagramme d’architecture globale représente la vue globale des composantes des systèmes et la communication entre eux.

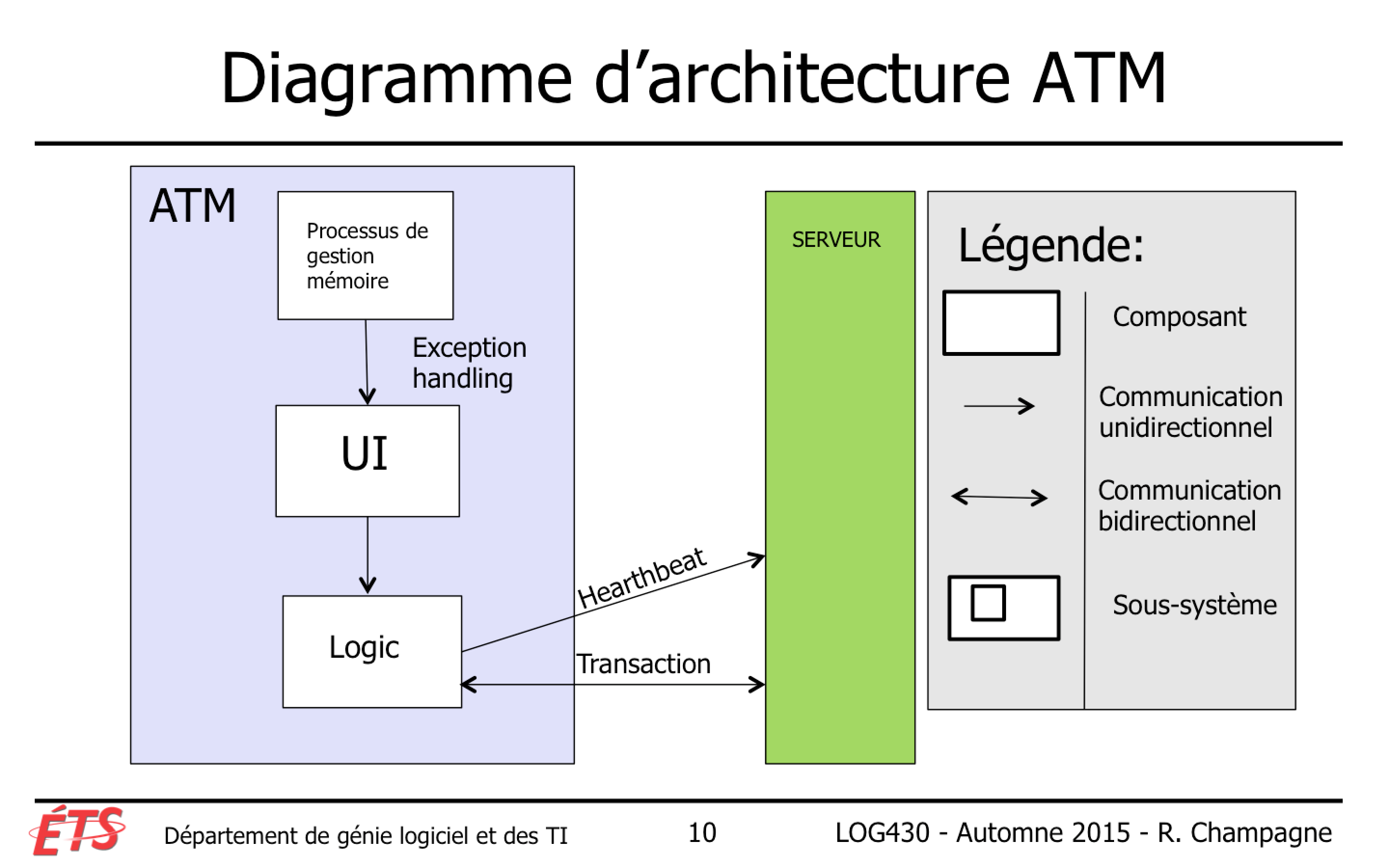
**ATM**: le système principal qui interagit avec le client.

**Monitor**: le serveur qui monitor l’état de ATM.

**Serveur bancaire**: le serveur qui contient les informations nécessaire pour les transactions de l’atm.

**Serveur bancaire de backup**: serveur de backup si le serveur bancaire principal ne fonctionne pas.

### Diagramme d’architecture ATM



**Description du diagramme ATM**

Le diagramme d’architecture ATM représente une ébauche de la vue spécifique des composantes du système ATM et de ces communications.

**ATM**: le système principal qui interagit avec le client.

**UI**: composante de l’ATM qui affiche les information au client.

Processus de gestion de mémoire: Gère la mémoire et agit si il y a un memory leak avec l’aide de execption handling.

**Logic**: logique du logiciel et de la communication.

### Description des Tactique

**Tactique 1**: Pour récupérer d’un memory leak, nous utilisont un exception handling qui redémare le système de ATM et informe l’UI pour le client.

**Tactique 2**: Pour récupérer d’une panne de réseaux, l’ATM envoie un heartbeat au serveur de monitoring et le serveur communique avec un opérateur en cas de problème.

**Tactique 3**: Pour récupérer d’une panne de réseaux, pendant un transaction, le server retry la transaction et reconfig au serveur bancaire de backup

### Résumé Tactique

Les trois tactiques développer permet d’avoir une meilleures disponibilités pour le client.