**Plan de Test**

**Développement Collaboratif**

**Acteurs :**

Nicolas Buffon, Marc Plano-Lesay, Jean-Christophe Ricard, Adrien Vetillart.

**Versions :**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Auteur | Date | Description |
| 1.0 | Jean-Christophe Ricard | 16/01/2012 |  |

**Introduction**

Ce document a pour objectif de définir les cas de tests à effectuer durant le développement de l’application Fire Disaster Simulator. Ici, nous allons décrire les tests nécessaires, les scénarios associés, les résultats attendus, leurs objectifs et les moyens utilisés pour y parvenir. Au vu du temps qui nous est imparti afin de réaliser ce projet, nous nous concentrerons sur les cas d’utilisation critiques de celui-ci.

I. Éléments ciblés 5

II. Vue d’ensemble 6

A. Tests inclus dans le développement 6

B. Tests exclus du développement 7

III. Types et techniques 7

A. Test d’intégrité 7

B. Test fonctionnels 7

C. Test de l’interface graphique 7

D. Test de performance 7

E. Test d’installation 7

# Éléments ciblés

Les éléments ciblés par les tests développés pour Fire Disaster Simulator sont les cas d’utilisation décris ci-dessous. Il est important de noter que nous compléterons ce document avec les résultats des tests effectués sur ces cas d’utilisations.

**Réaliser la simulation** – L’objectif principal de l’application est de pouvoir exécuter une simulation correctement. Il est donc primordial de vérifier son bon fonctionnement.

**Charger une carte** – Ce cas d’utilisation permettant la personnalisation du terrain est un des plus importants, car il constitue le cœur du logiciel.

**Lancer la simulation** – Composant le cas d’utilisation **Réaliser la simulation**, il est primordial de tester que celle-ci se lance correctement.

**Positionner un incendie** – Ce cas d’utilisation offre à l’utilisateur la possibilité de personnaliser l’intervention à effectuer, et donc de pouvoir simuler des cas très précis. L’interaction et le paramétrage de la simulation sont les piliers de Fire Disaster Simulator.

**Enlever un incendie** – Ce cas découle directement du précédent. Il est donc associé à celui-ci, et pourra être testé dans le même temps.

**Positionner un robot** – Il est un élément important dans le paramétrage de la simulation, au même titre et pour les même raisons que **Positionner un incendie**.

**Enlever un robot** – Celui-ci découle du cas d’utilisation précédent, au même titre qu’**Enlever un incendie**.

# Vue d’ensemble

## Tests inclus dans le développement

Durant ce développement, nous allons donc inclure les scénarios d’utilisation critiques. Ci-dessous, nous allons décrire ceux-ci, ainsi que les résultats attendus.

**Réaliser la simulation** :

L’utilisateur doit être capable de lancer une simulation sans avoir de paramètres à modifier. Il faudra donc qu’une carte soit pré-chargée, que quelques robots et un ou plusieurs incendies soit positionnés sur celle-ci. Ainsi, la simulation doit pouvoir se dérouler.

Le résultat attendu sera donc que la simulation, directement après le lancement de Fire Disaster Simulator, puisse être lancée en cliquant uniquement sur le bouton correspondant. On inclus également qu’elle doit se dérouler correctement, jusqu’à sa fin.

**Charger une carte**:

L’utilisateur doit pouvoir choisir une carte et la charger dans le logiciel. Cela inclus donc qu’il puisse parcourir ses fichiers depuis l’application.

On attend donc du logiciel qu’il puisse charger un fichier de carte vierge, afin de réaliser une simulation sur celle-ci.

**Lancer la simulation**:

L’utilisateur doit avoir la possibilité de lancer l’exécution d’une simulation, après avoir réglé les paramètres de son choix. Ceux-ci devront donc être effectifs durant la simulation.

Le résultat obtenu devra donc être que l’utilisateur puisse lancer l’exécution d’une simulation.

**Positionner et enlever un incendie** :

L’application doit pouvoir fournir à l’utilisateur la possibilité de placer un ou plusieurs incendies sur la carte chargée. L’utilisateur doit également pouvoir les supprimer, dans le cas d’une erreur de placement par exemple.

Le résultat attenu est donc de pouvoir placer et supprimer des incendies afin de paramétrer la simulation.

**Positionner et enlever un robot**:

Enfin, l’application se devra de proposer à l’utilisateur la possibilité de placer un ou plusieurs robots sur la carte chargée. L’utilisateur doit également pouvoir les supprimer, dans le cas d’une erreur de placement par exemple.

Le résultat attenu est donc de pouvoir placer et supprimer des robots afin de paramétrer la simulation.

## Tests exclus du développement

Nous excluons de ce développement les cas d’utilisation moins importants (cf. document des cas d’utilisation), jugés de priorités moyenne ou basse. Nous exclurons également les tests de charge, dédiés à des application d’architecture client-serveur. Les tests de sécurité d’accès n’ont également pas lieu d’être, étant donné que c’est une application locale. Enfin, nous n’effectuerons pas de tests de configuration, car Fire Disaster Simulator ne nécessite pas de configuration préalable à son exécution.

# Types et techniques

## Test fonctionnels

L’objectif de ces tests est de vérifier chacune des fonctionnalités du logiciel décrites dans les plans de test ci-dessus. En déroulant chacun des scénarios, et en vérifiant chacun des retours du logiciel, on est ainsi en mesure de vérifier leur bon fonctionnement.

Pour que l’ensemble du test soit positif, il suffit que chacune des étapes du scénario se déroule comme prévu. De plus nous n’avons aucun besoin particulier afin de réaliser cette opération. En effet, seul un utilisateur de l’application est nécessaire.

## Test de l’interface graphique

Le test de l’interface graphique permet de vérifier que celle-ci soit claire, aisée à comprendre et adaptée à l’utilisateur. Il est possible d’effectuer ce test de la même façon qu’un test fonctionnel. C’est-à-dire en déroulant un scénario avec un utilisateur lambda de l’application, sans lui donner aucune consigne sur la façon de réaliser un scénario. Il est important de noter chacune de ses remarques, et de voir ses réactions en fonction des éléments graphique.

Afin de vérifier le résultat du test, nous devons choisir les points sur lesquels nous devrons porter notre attention sur ce test et poser des questions à l’utilisateur. Celui ne devras avoir aucun lien avec le développement du projet, mais faire partie du panel d’utilisateurs, pour que le test soit le plus efficace possible.

## Test de performance

Le test de performance, dans notre cas, permet de vérifier que l’application est capable de fonctionner sur le plus grand nombre de machines. C’est-à-dire selon la puissance de celles-ci. Afin de vérifier les résultats de ce test, nous aurons besoin de plusieurs machines. Par exemple, d’un ordinateur de bureau puissant, d’un netbook et d’un ordinateur portable de travail, un peu plus puissant qu’un netbook. Un simple panel de 3 ou 4 machines peut suffire à notre échelle. Durant ce test, nous déroulerons quelques scénarios simples (réaliser une simulation, charger une carte, etc..) afin de vérifier si notre application ne met pas un temps trop élever à traiter les informations requises.

## Test d’installation

Enfin, le test d’installation, quant à lui, consiste à vérifier que l’application Fire Disaster Simulator s’installe correctement sur les systèmes d’exploitation compatibles avec Java. Dans ce cas là, nous testerons les trois principaux : Windows, Mac OS et Linux. Les pré-requis pour ce test sont uniquement des machines avec ces systèmes d’exploitation installés, ainsi que Java. Il nous incombera de vérifier également qu’elle se lance correctement.