République Tunisienne
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université de Carthage

Institut Supérieur des Technologies de l'Information et de la Communication

Rapport de Stage de Fin d'Année

Système embarqué et IoT

Parcours : ingénierie des systèmes d'information et logiciels

Par

Elaa Henchir

Détecter un composant dans une carte électronique

Réalisé du 15 Juin 2021 au 15 juillet 2021 au sein de CORAIL TECHNOLOGIE

Encadrant Professionnel :

Malek Marzouki

Anné Universitaire : 2021-2022

Dédicaces

A mes chers parents,

Aucun mot, aucune dédicace ne pourrait exprimer mon respect, ma considération, et mon amour pour les sacrifices que vous avez consentis pour mon instruction et mon bien-être. Vous représentez pour moi le symbole de la bonté par excellence, la source de tendresse et l'exemple du dévouement qui n'a pas cessé de m'encourager et de prier pour moi.
Que Dieu vous accorde la santé, le bonheur, la prospérité et une longue vie.

A mon frère et mes sœurs,

Je ne peux exprimer à travers ces lignes tous mes sentiments d'amour et de tendresse envers vous. Je vous souhaite la réussite dans votre vie, avec tout le bonheur qu'il faut pour vous combler.
Que Dieu vous apporte la sérénité, l'aide à réaliser vos vœux et vous offre un avenir plein de succès.

A mes amis ,

Je ne peux trouver les mots justes et sincères pour vous exprimer mon affection et mes pensées, vous êtes pour moi ma deuxième famille. En témoignage de l'amitié qui nous unit et des souvenirs de tous les moments que nous avons passés ensemble, je vous dédie ce travail et je vous souhaite une vie pleine de santé et de bonheur.

Elaa Henchir

Remerciements

Je tiens en premier lieu de remercier la société CORAIL TECHNOLOGIE qui m'a offert l'opportunité de développer mes connaissances et elle m'a mis dans le chemin de la vie pratique et plus spécialement mon encadrante Mme. MALEK MAZOUK pour ses conseils et son encouragement.

Je n'oublie pas de remercier mes parents qui m'ont fait des sacrifices énormes pour qu'je puisse arriver là où je suis.

Ainsi que toute personne qui m'a aidé de loin ou de près à la réussite de ce projet. Qu'ils reçoivent l'expression de mes sincères reconnaissances et gratitudes.

Enfin, je tiens à apprécier l'honneur des membres du jury pour avoir accepté de m'accorder leur attention et évalué de mon travail.

Table des matières

Table des figures

Introduction Générale

Plus de 75 % des lancements de produits se soldent par un échec dès la première année, souvent parce que les marques mettent un produit sur le marché sans avoir préalablement testé ce dernier. C'est pour cela que le test de produit représente un format gagnant pour les marques qui peuvent en faire un levier d'influence efficace.

Dans un objectif purement professionnel et ce pour pallier aux défis du monde du travail, pour découvrir son fonctionnement, ses activités et ses objectifs, j'ai décidé de passer un stage qui m'offre la possibilité d'élargir mes connaissances et d'acquérir une méthode de travail en passant de la théorie à la pratique.
Mon choix s'est orienté vers la société « Corail technologie » qui est spécialisée en fabrication d'outillages et logiciels de test. C'est dans ce cadre que se situe l'élaboration du sujet de fin d'année au sein de corail qui consiste à tester la présence d'un composant spécifique dans plusieurs cartes électroniques.

Dans ce rapport, on va présenter en premier lieu l'entreprise et le contexte de stage, ainsi que les concepts théoriques utiles pour l'élaboration de notre projet. on finit par détailler les taches effectuées en vue de réaliser cette application.

Chapitre 1

Cadre général

1.1 Introduction

Ce chapitre présente l'organisme d'accueil, le contexte du projet, la problématique et un diagnostic technique de la solution developpée.

1.2 Présentation de l'entreprise

1.2.1 Organisme d'accueil

« Corail technologie » est une Société Tunisienne crée en 2007 et située dans le Technopole de Borj Cedria. C'est une société de service et d'ingénierie électronique, développement de logiciel et de test.
Le rôle principal de « Corail technologie » est la fabrication des machines spéciales ainsi que l'assistance à l'industrie.
Cette société développe des compétences dans les domaines de la conception, la réalisation des cartes électroniques, le câblage industriel, le développement software et embarqué et la fabrication mécanique.

FIGURE 1.1 – Logo de l'entreprise Corail

1.2.2 Les domaines d'activités

Le domaine d'activité de la Société est partagé en deux secteurs :

◇ Le secteur de L'Ingénierie Test :
Industrialisation des moyens de test pour les produits électroniques
Industrialisation des machines spéciales
Conception et réalisation des cartes électroniques
Développement des applications de test
Prestation d'ingénierie

◇ La réalisation des cartes électroniques :
La Société produit deux types de circuits imprimés :
– Le type professionnel fabriqué par le partenaire français Inter systèmes, ayant un nombre de couches allant jusqu'à 12 pour les projets finis et la fabrication en grande quantité.
– Des circuits artisanaux simples et rapides dont le nombre de couches va du simple à la double face essentiellement pour la réalisation de prototypes.

1.2.3 Les clients

Les principaux clients de Corail Technologie sont de grands fabricants de produits électroniques et de développement embarqué tels que : ZODIAC AEROSPACE, SAGEM, SOMEF, etc ... La figure 1.2 présente les clients de corail.

FIGURE 1.2 – Principaux clients de Corail Technologie

1.3 Contexte du projet

Ce projet consiste à tester la présence d'un composant dans une carte électronique à travers une caméra. On a décidé d'utiliser LabVIEW et vision builder pour réaliser ce projet et répondre à nos besoins.

1.4 Problématique

L'entreprise Corail Technologie est responsable de la fabrication de nombreux produits de test dont la plupart nécessitent l'intégration d'un système permettant la détection de présence des composants sur les cartes électroniques.

En effet, une bonne identification d'un composant nécessite « GigE Basler camera » avec un support fixe non disponible chez la société à cause du problème de financement car cette solution qui paraît fiable et parfaite, coûte environ 1400 DT (400 Euro).

1.5 Solution proposée

En attendant que la caméra « GigE » sera disponible chez corail, nous envisageons de réaliser le test sur les cartes électroniques en chargeant des images claires de ces cartes à partir de notre PC, c'est-à-dire on va tester et visualiser des images des cartes.

1.6 Conclusion

Dans le premier chapitre, on a présenté l'entreprise et le contexte du projet. Dans le chapitre qui suit, on va faire l'analyse et la conception.

Chapitre 2

Analyse et conception

2.1 Introduction

Ce chapitre consiste à analyser les besoins des utilisateurs en respectant un cahier des charges, présnter les diagrammes pour modéliser l'objet d'étude et exposer l'architecture système.

2.2 Analyse des besoins

2.2.1 Les besoins fonctionnels

Contrôle des cartes : Permet de visualiser les cartes en cours de test à travers une interface simple.
Etat des cartes : Sert à donner l'état de carte en cours de test.
Calcul de temps de test : Sert à donner la durée pendant laquelle la carte est testée.
Nombre des cartes : Permet de donner le nombre total des cartes, le nombre des cartes « passed » et le nombre des cartes « failed ».

2.2.2 Les besoins non fonctionnels

La rapidité de traitement : Compte tenu du grand nombre de transactions par jour, le temps de traitement doit être le plus proche possible du temps réel.
La performance : Nous utilisons les performances pour spécifier la durée pendant laquelle le système répond aux demandes d'entrée. Ce terme fait référence à la vitesse à laquelle le système effectue le traitement.
La disponibilité : L'application devrait être opérationnelle d'une façon continue, car l'utilisateur peut faire des réservations à tout moment. Le système doit être en permanence à la disposition de ses utilisateurs.
Ergonomie et Simplicité : Cette fonctionnalité permet à l'utilisateur d'être à l'aise lors de l'utilisation ou de la consultation du site.
L'extensibilité : Cela nous donne la possibilité d'ajouter, de modifier ou de supprimer des fonctionnalités.
Fiabilité : Notre application doit être bien testée avant de l'héberger aux clients afin d'éviter les éventuels des bugs.

2.3 Cahier des charges

2.3.1 Enoncé des besoins

Titre : Identifier la présence d'un composant dans une carte à travers une caméra.
En se basant sur les librairies du LabVIEW et vision Builder, on va réaliser un projet permettant de :

— Détecter un composant de la carte électronique.
— Donner le résultat du test.
— Calculer le temps nécessaire pour tester chaque carte.

2.3.2 Principe de fonctionnement

Pour pouvoir réussir ce projet, deux logiciels sont utilisés à savoir un logiciel d'application « Vision Builder » et un logiciel d'ingénierie des systèmes « LabVIEW ».

◇ A travers Vision Builder, la configuration de la carte est assurée comme suit :

- Sélectionner l'emplacement des images des cartes à tester
- Préciser dans quelle partie de la carte se trouve le composant.
- Communiquer avec LabVIEW afin de faire l'acquisition de l'image.

◇ A travers LabVIEW, 3VI seront réalisés :

- La première VI sert à établir la connexion entre Vision Builder et LabVIEW ; LABVIEW comporte un module qui se n'appelle «NI VISION BUILDER » qui offre plusieurs outils de vision. En se basant sur la palette **Vision Builder AI** , on va lancer vision builder. Ensuite, établir la connexion entre notre programme dans vision builder et LabVIEW et finalement on va faire l'inspection de l'image en cours de traitement.
- La deuxième VI sert à prendre l'image en cours de traitement pour donner le résultat d'inspection.
- La troisième VI sert à relier entre les deux VI afin d'afficher la carte, afficher l'état de test et calculer le nombre total des cartes.

2.4 Modélisation

Pour pouvoir analyser et clarifier un projet, il faut utiliser un outil de modélisation tel que le Diagramme « Fast ».

2.4.1 Définition

C'est une technique permettant de représenter sous forme de diagramme les relations logiques existant entre les fonctions d'un sujet en répondant aux questions « comment » et « pourquoi ».
La méthode FAST nous permet de déterminer les relations logiques entre les fonctions.

2.4.2 Le diagramme Fast du projet

Le diagramme « Fast » permet de développer une compréhension commune du projet, définir, simplifier et clarifier le problème ainsi que d'organiser et comprendre les relations entre les fonctions. La figure 2.1 représente le diagramme fast de notre projet.

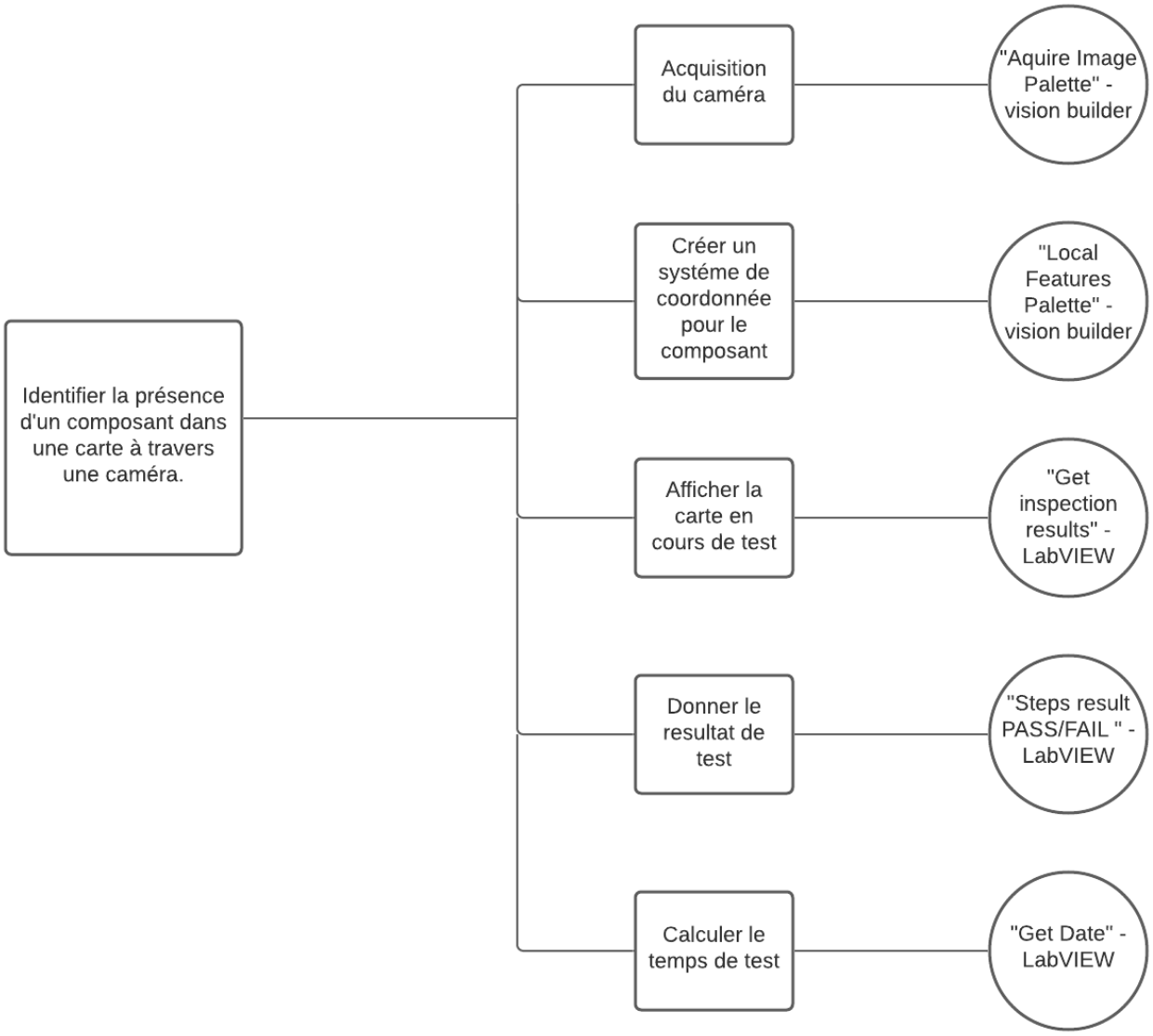

Figure 2.1 – Diagramme FAST

2.5 Le diagramme flowchart

2.5.1 Définition

« Flowchart » est un type de diagramme qui représente un flux de travail ou un processus. Un organigramme peut également être défini comme une représentation schématique d'un algorithme, une approche étape par étape pour résoudre une tâche.

L'organigramme montre les étapes sous forme de cases de différentes sortes, et leurs ordre en reliant les cases avec des flèches. Cette représentation schématique illustre un modèle de solution à un problème donné. Les organigrammes sont utilisés pour analyser, concevoir, documenter ou gérer un processus ou un programme dans divers domaines.

2.5.2 Le diagramme flowchart du projet

Dans cette sous-section, on expose le diagramme flowchart qui permet de représenter l'algorithme à suivre pour réaliser cette application. La figure 2.2 décrit le diagramme « Flowchart » du projet.

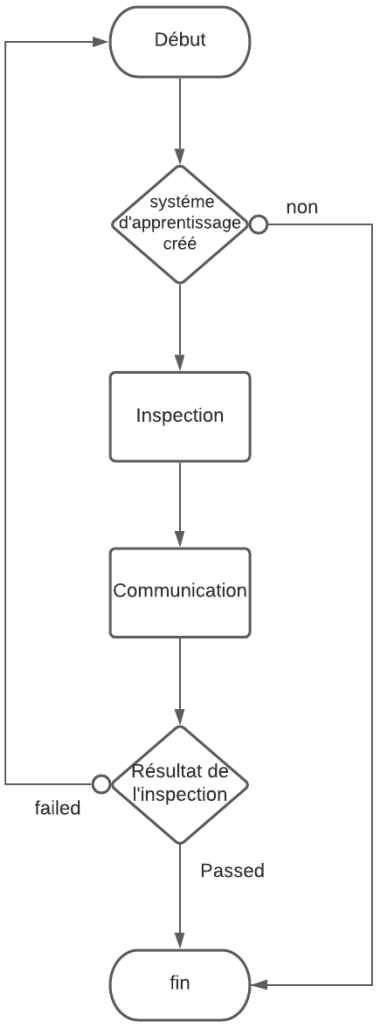

FIGURE 2.2 – Diagramme Flowchart

2.6 Architecture système

L'architecture d'un système est un modèle conceptuel qui décrit ses propriétés externes et internes. Aussi la manière dont elles se projettent dans ses éléments ainsi que leurs relations, les principes de conception et d'évolution du système.

Notre système est basé sur une communication hardware avec la caméra de vision et une communication software entre LabVIEW et Vision Builder. Les données d'inspection issues de la caméra seront traitées par Vision Builder. Ce dernier communique avec LabVIEW pour afficher ces données.
La figure 2.3 montre les principaux éléments du système.

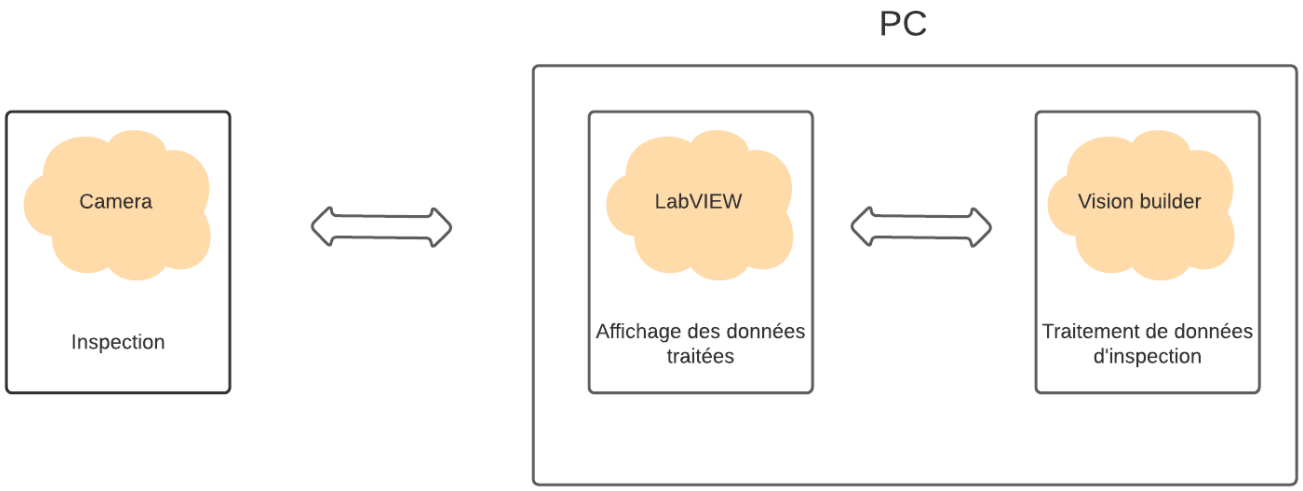

FIGURE 2.3 – Schéma synoptique

2.7 Conclusion

Dans ce chapitre, on a identifié les besoins fonctionnels et non fonctionnels et présenté le cahier des charges. Par la suite, on a réalisé une conception pour notre projet. Dans le chapitre suivant, on passe à la partie la plus détaillée qui concerne la réalisation du projet.

Chapitre 3

Réalisation du projet

3.1 Introduction

Ce chapitre est consacré pour l'environnement matériel et logiciel. L'environnement logiciel fournit le développement software et l'interface graphique de notre application. Des exemples de cas de test des cartes éléctroniques sont présentés à la fin de ce chapitre.

3.2 Environnement matériel

3.2.1 Réseau de transfert d'images

Les réseaux spécialisés et dédiés au traitement d'images proposent des structures réseau performantes et des interfaces d'échange de données adaptées aux applications de vision industrielle.

Les systèmes d'acquisition et de traitement d'images sont "gourmands" en terme de bande passante et nécessitent d'échanger des volumes importants de données (les images) tout en garantissant des temps de réponses parfois très courts.

Lorsque le système de traitement est embarqué dans un capteur de vision, la principale contrainte est le volume de données échangées, selon le nombre d'images que l'utilisateur souhaite visualiser et à quel rythme, selon la fréquence d'archivages souhaitée, et selon le nombre de caméras en réseau.

Lorsque le traitement des images est externe au système d'acquisition, il est de plus nécessaire de transférer toutes les trames des images acquises vers le système de traitement dans un temps raisonnable pour l'application, de façon à garantir le traitement avec un temps de réaction du système de vision vers les équipements à commander.

3.2.2 GigE vision

« GigE VISION » (Gigabit Ethernet Vision) est un système de communication spécialisé de haute performance dédié aux systèmes de vision et plus généralement dédié au transfert d'images, dans l'intention d'interfacer des caméras en provenance de différents fabricants avec n'importe quel autre équipement et avec n'importe quelle application utilisateur.

C'est l'interface la plus courante pour connecter une caméra de vision dans les applications de vision industrielle.

Les réseaux « GigE Vision » fonctionnent en utilisant le protocole UDP (Protocole de Datagramme Utilisateur) plutôt qu'en utilisant le protocole TCP (Protocole de Contrôle du Transport).

3.2.3 Camera GigE

Les caméras Gigabit Ethernet sont utilisées dans les usines pour les inspections de qualité des produits et l'automatisation des processus.

La caméra Gigabit Ethernet fournit une image RAW via un câble réseau à l'ordinateur. Ensuite, le logiciel de traitement d'images sur l'ordinateur analyse ces images.

Cette caméra se base sur l'alimentation en PoE (Power over Ethernet). Le principe de base d'une alimentation en POE (Power over Ethernet) est de pouvoir coupler avec un seul et même câble une entrée de courant et une entrée de données réseaux.

FIGURE 3.1 – Industrial camera GigE

Vision builder offre une communication hardware pour faire l'acquisition des caméras à partir de plusieurs bus tel que « GigE ». C'est pour cela qu'on a proposé ce type de caméra qui est le plus compatible avec notre système de vision.

3.3 Environnement logiciel

3.3.1 LabVIEW

LabVIEW est un logiciel d'ingénierie système pour les applications qui nécessitent des tests, des mesures et des contrôles avec un accès rapide au matériel et aux informations sur les données.

En effet, il offre de larges possibilités de communication entre l'ordinateur et le monde physique ainsi que d'importantes bibliothèques mathématiques permettant de réaliser des multiples traitements sur les signaux mesurés.

Le langage de programmation utilisé dans LabVIEW, nommé G, fonctionne par flux de données. L'exécution d'un code est déterminée par un schéma graphique, le diagramme, qui est le code source. Le programmeur connecte différentes fonctions sous forme d'icônes dans le diagramme par l'intermédiaire de fils dessinés entre les terminaisons des blocs d'icônes. La figure 3.2 représente un tableau contenant les avantages et les inconvénients du LabVIEW.

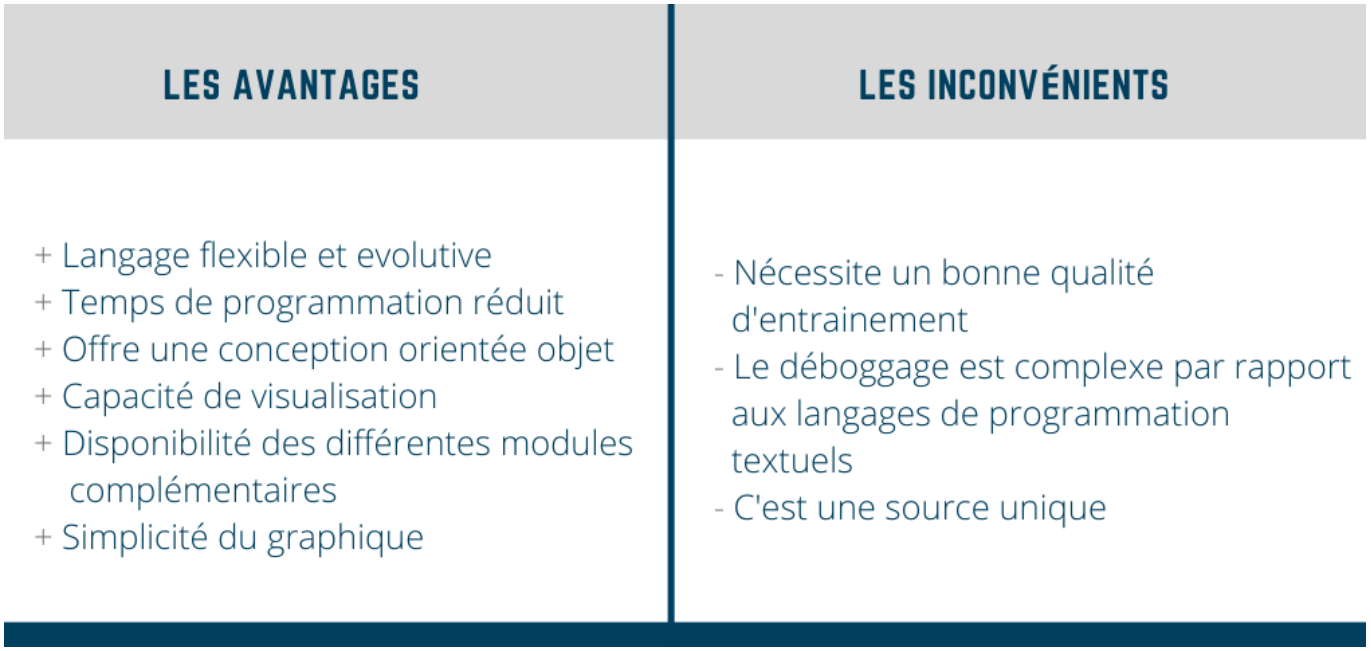

FIGURE 3.2 – Les avantages et les inconvénients du LabVIEW

Alternatives à LabVIEW

En ce qui concerne le contrôle, le test et la mesure, on peut développer avec des logiciels tels que :

— LabWindows CVI , de National Instruments, qui est un environnement de développement pour le langage C et qui offre les mêmes bibliothèques logicielles de fonctions.
— Measurement Studio , de National Instruments, qui est un ensemble de bibliothèques de fonctions et de contrôles graphiques pour Visual Studio, permettant ainsi de la programmation en C++, Visual Basic ou C sharp tout en profitant de fonctionnalités destinées au contrôle, test et mesure.
— TestStand et VeriStand , de National Instruments (plutôt complément qu'alternative : séquenceur de tâches).

LabVIEW VIs

Avec LabVIEW on construit graphiquement des modules logiciels appelés des « VI » (Virtual Instruments) au lieu d'écrire du code dans un langage informatique textuel.

Les VIs sont des modules de code individuels qui constituent une application complète.

Le rôle d'un VI est d'acquérir des données issues par exemple de fichiers, du clavier ou encore de cartes électroniques d'Entrée/Sorties, de les analyser, et de les présenter au travers d'interfaces hommes-machines graphiques.

Dans un aspect, l'instrument virtuel LabVIEW pourrait être comparé à un sous-programme utilisé dans certains langages de programmation.

Le VI LabVIEW se compose de deux éléments principaux :

— Front panel : c'est la conception de l'interface utilisateur où l'on dessine et place tous les éléments visuels.
C'est-à-dire on contrôle l'entrée de l'utilisateur : bouton, interrupteur, potentiomètre, curseur, zone de saisie de valeur/listes ...
On contrôle aussi la sortie du programme : voyant, graphe, thermomètre, zone de texte ...

— Block diagram : contient le code d'exécution.

3.3.2 Vision builder

Vision Builder AI est un logiciel d'application qui nous aide à configurer des caméras, personnaliser le traitement d'images à partir de centaines d'algorithmes et d'étapes d'inspection, interagir avec le matériel d'automatisation et générer des résultats d'inspection grâce aux outils de développement pilotés par menu.

A travers « Vision Development Module » on peut communiquer avec vision builder depuis LabVIEW. Ce module nous permet de :

— Déployer des modèles d'apprentissage en profondeur sur des systèmes de vision industrielle.
— Inspecter et mesurer les produits.
— Effectuer un traitement d'image à grande vitesse sur des FPGA.

3.4 Développement software du programme

3.4.1 Configuration de la carte

Vision Builder AI utilise un diagramme d'état pour définir des inspections avec des états et transitions qui régissent le flux d'exécution de l'inspection. Des inspections simples peuvent être définies à l'aide de l'inspection par défaut à un seul état.
La figure 3.3 montre le diagramme d'état d'inspection de notre programme.

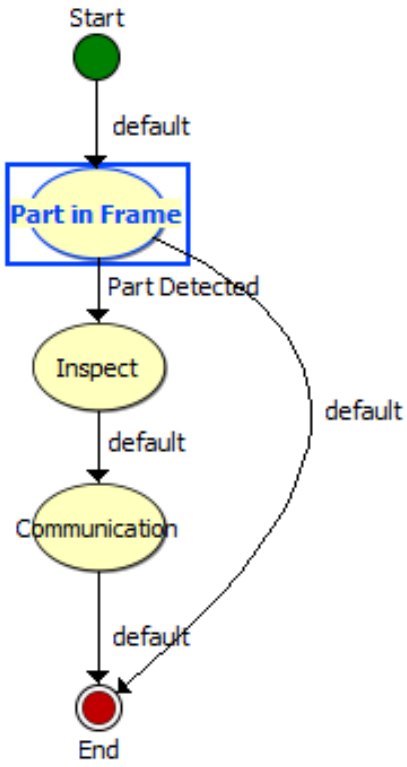

FIGURE 3.3 – Diagramme d'état d'inspection

« Part in Frame » est la plus importante étape dans ce diagramme. Elle est exécutée pour donner finalement l'état d'inspection. La figure 3.4 représente la conception de cette étape.

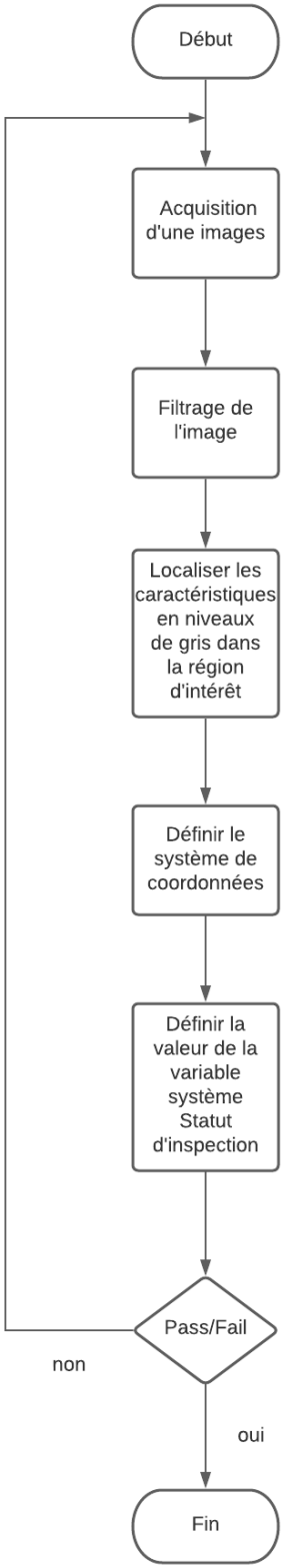

FIGURE 3.4 – Conception part in frame

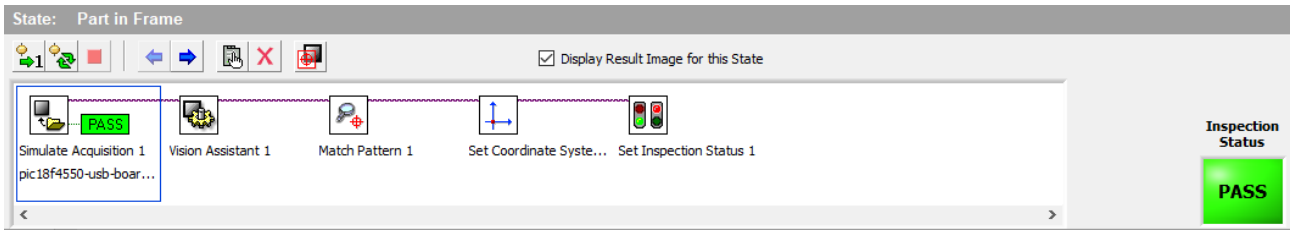

FIGURE 3.5 – Etape d'inspection

— Simulate acquisition : Permet de simuler une acquisition en direct en chargeant des images à partir d'un fichier. Dans notre cas, on a mis 4 images des cartes électroniques claires dans un dossier et on a chargé ces images à partir de ce fichier.

— Vision assistant : Sert à faire le choix d'une région d'intérêt et de les séparer de l'arrière-plan. Il est possible de choisir toute l'image ou de limiter cette région en traçant un carré qui détermine la région d'intérêt. La région d'intérêt a été limitée à un carré, car cela permet de minimiser le temps que prend cette étape à trouver l'image qu'elle recherche.

— Match Patten : Permet de rechercher le composant sélectionné. La détermination de ce composant se fait en cliquant sur le bouton « new template » dans la table « template » qui sert à encadrer le composant à rechercher. Il est important que ce composant soit retrouvée parce que le système de coordonnées sera basé sur ce composant dans la prochaine étape.

— Set coordinate system : Sert à créer un système de coordonnées à partir des points définis dans les étapes précédentes, qui peut varier d'une image à l'autre. Le système de coordonnées pour repositionner les régions d'intérêt dans les étapes suivantes pour correspondre à la position de l'objet à inspecter.

— Set Inspection status : Permet de mettre à jour la valeur de la variable système statut d'inspection.

3.4.2 VI Connexion

Notre projet se compose essentiellement de 3VI. La première VI est dédiée à lancer Vision Builder à travers LabVIEW. On va spécifier le chemin pour accéder à notre projet dans Vision Builder « carte-corail-program.vbi » grâce à la fonction **build path** qui se trouve dans la palette File I/O.

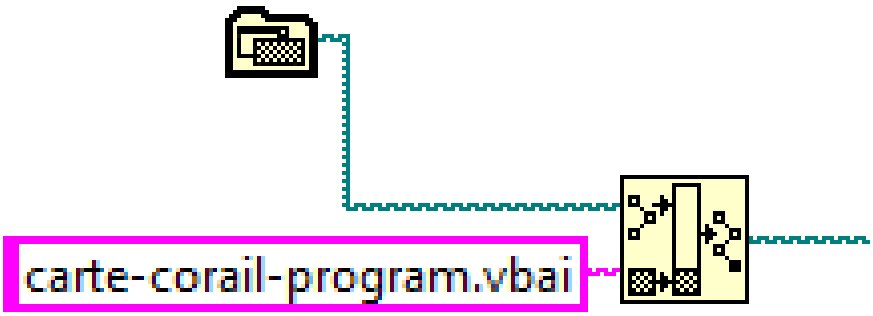

FIGURE 3.6 – Build path

Ce chemin sera l'entrée de la fonction **Open inspection**. Avant d'ouvrir une inspection, on doit tout d'abord lancer vision builder pour exécuter cette inspection.

Vision Builder sera lancé à travers la fonction **Launch local VBAI**. Cette fonction permet de spécifier la version de vision builder, donc on doit choisir une version compatible avec celle du LabVIEW faute de quoi la connexion ne sera pas établie.

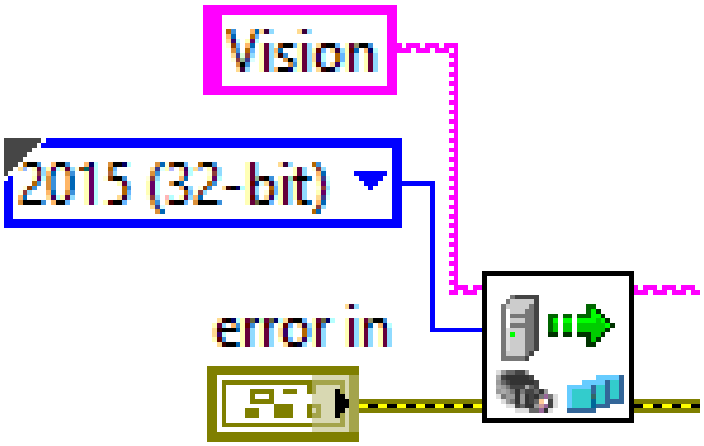

FIGURE 3.7 – Launch local VBAI

Après on doit lancer la connexion, puis relier la sortie de la fonction **open connection** et la sortie de la fonction build path aux entrés de la fonction **Open inspection.**
La figure 3.8 représente la programmation de la VI connexion à travers LabVIEW.

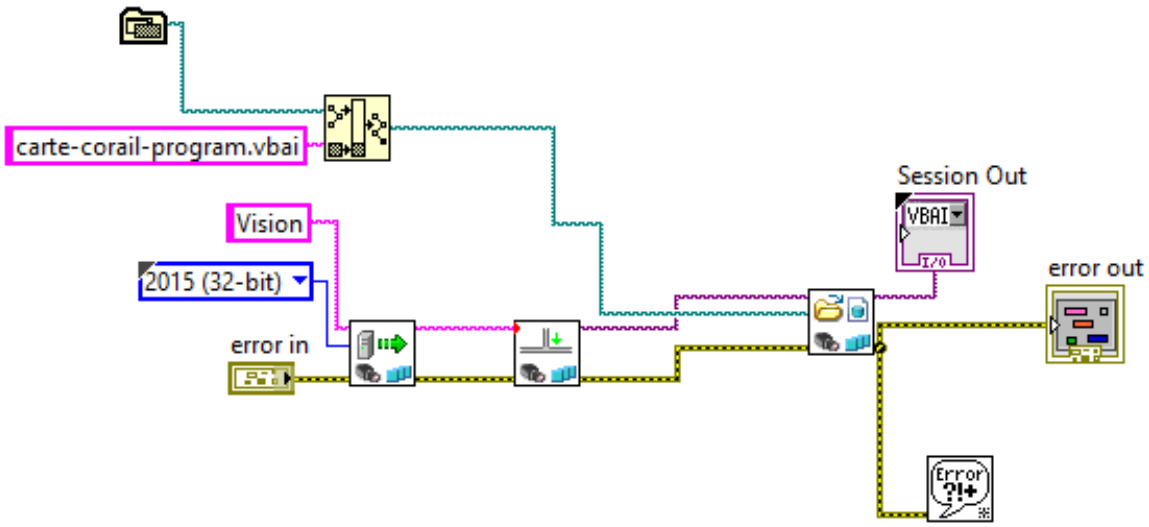

FIGURE 3.8 – Programmation LabVIEW de la VI connexion

3.4.3 VI Image

La première fonction à utiliser dans cette VI est **Run inspection** qui permet d'exécuter une itération de l'inspection actuellement chargée par Vision Builder.

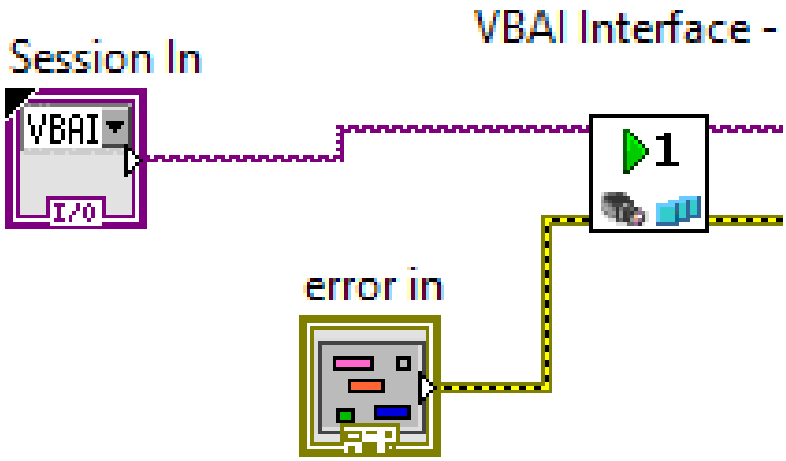

FIGURE 3.9 – Run inspection

Avant de prendre l'inspection de l'image on doit rendre les mesures de toutes les étapes de l'inspection accessibles à travers la fonction **Enable Inspection Measurements.**

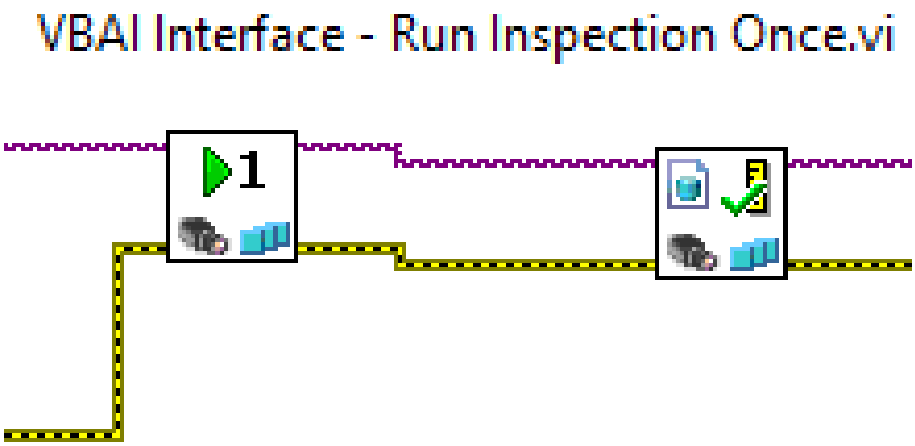

FIGURE 3.10 – Enable inspection

La fonction **Get Inspection Image** nous donne comme résultat l'image de l'inspection chargée par vision builder qui va être la sortie de cette VI.

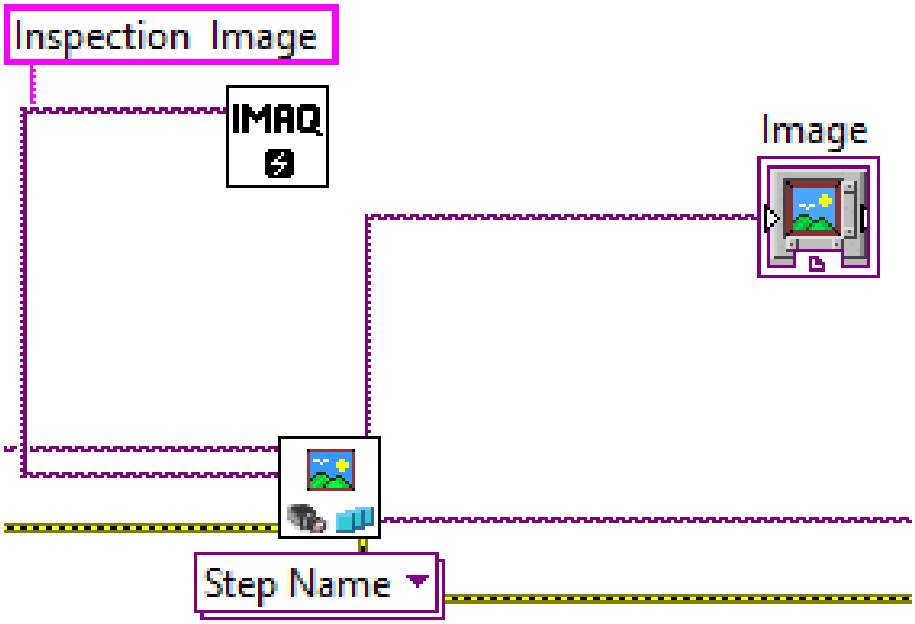

FIGURE 3.11 – Get inspection image

Aussi cette VI contient deux autres sorties donnés par la fonction **Get Inspection results :**

- **Error out :** En cas d'erreur lors de la transmission des informations.
- **Inspection status :** Inspection Status renvoie un booléen qui indique si l'itération d'inspection a réussi ou échoué. TRUE indique la réussite et FALSE indique l'échec. La figure 3.12 représente la programmation de la VI image à travers LabVIEW.

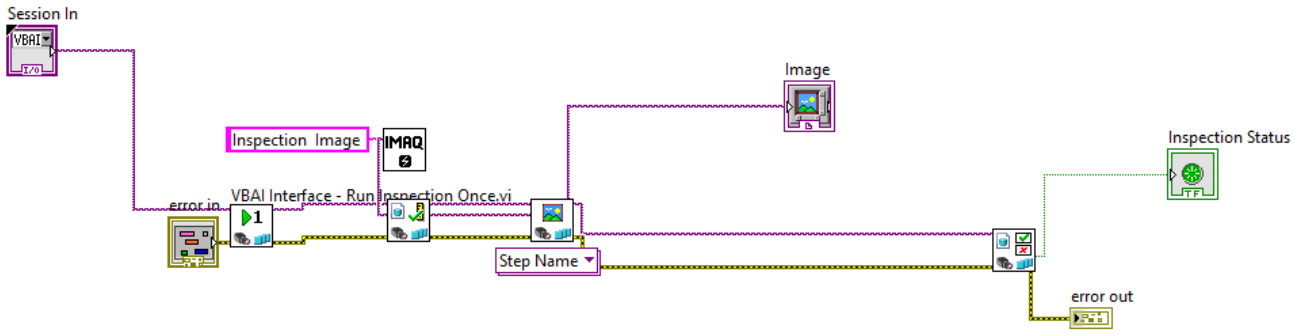

FIGURE 3.12 – Programmation LabVIEW de la VI image

3.4.4 VI main

Du au traitement parallèle dans LabVIEW, on a décidé d'utiliser **Flat sequence stucture** qui se trouve dans la palette structure pour assurer qu'un sous-diagramme s'exécute avant ou après un autre sous-diagramme.

Flat sequence stucture se compose d'un ou plusieurs sous-diagrammes, ou cadres, qui s'exécutent de manière séquentielle. Les trames de cette structure s'exécutent de gauche à droite. Les données quittent chaque trame à la fin de l'exécution. Cela signifie que l'entrée d'une trame peut dépendre de la sortie d'une autre trame.

Dans notre cas, on a utilisé **Flat sequence** pour initialiser le nombre de carte "pass", le nombre de carte "fail" et le nombre total des cartes à zéro avant de lancer le programme principal.

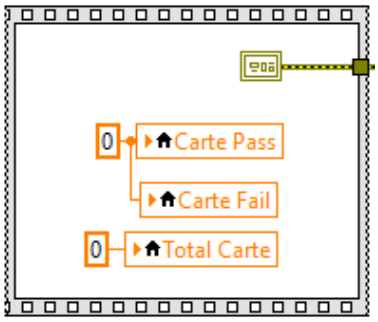

FIGURE 3.13 – Flat sequence

Afin de réduire le nombre des VI dans le bloc diagramme principal, il est préférable de concevoir des sous VI permettant de regrouper les VI par thème. Chaque sous VI comprend des entrés et des sorties permettant de faire passer les données d'une VI à un autre. Cette méthode facilite la lecture du code développé et le rendre plus lisible.

Les étapes du programme se traduisent dans les points suivants :

1. Lancer vision builder en utilisant la VI connection.
2. Utiliser la boucle for pour faire l'inspection de 4 images : la VI image donne comme sortie l'image de l'inspection chargée par vision builder qui va être la sortie de cette VI.
3. Afficher l'image dans « font panel »
4. Afficher le nombre des cartes calculé.
5. Afficher l'état de chaque carte (« PASSED / FAILED »)

Si **Inspection Status** renvoie TRUE alors le nombre de cartes "pass"
sera incrémenter de 1.
Si **Inspection Status** renvoie FAULSE alors le nombre de cartes "fail" sera incrémenter de 1.

Pour le temps de test, on a utilisé la fonction **Get date** in seconds avant l'inspection de l'image et après l'inspection puis on a utilisé **substrat function** pour calculer la différence entre les deux temps afin de donner finalement le temps de test (le temps d'inspection) de chaque carte.
La figure 3.14 représente la programmation de la VI main à travers LabVIEW.

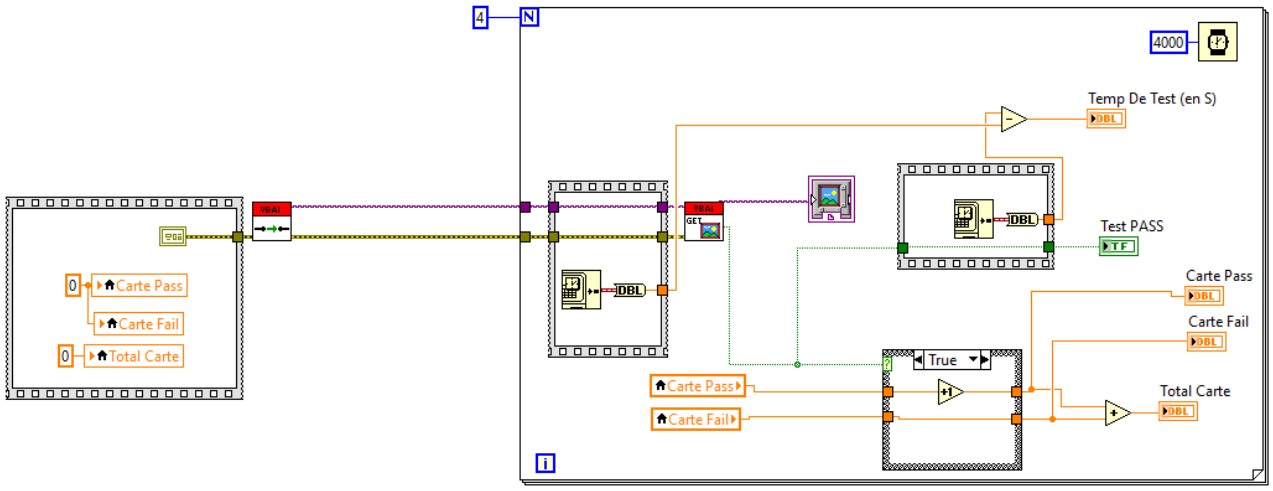

FIGURE 3.14 – Programmation LabVIEW de la VI main

3.5 Interface graphique

Cette interface permet de visualiser la carte en cours de test. De plus, un changement de couleurs va nous aider à mieux connaitre l'état de chaque carte, ce changement est en effet une LED qui s'allume en vert si notre carte contient le composant, et en rouge dans le cas contraire.

Avec le changement du couleur, le texte changera en conséquence avec chaque état ; soit notre test est « failed » soit notre test est « passed ».

Des indicateurs numériques sont utilisés pour afficher plusieurs valeurs tel que : le nombre des cartes « passed », le nombre des cartes « failed », le nombre total des cartes testés, le temps de test...

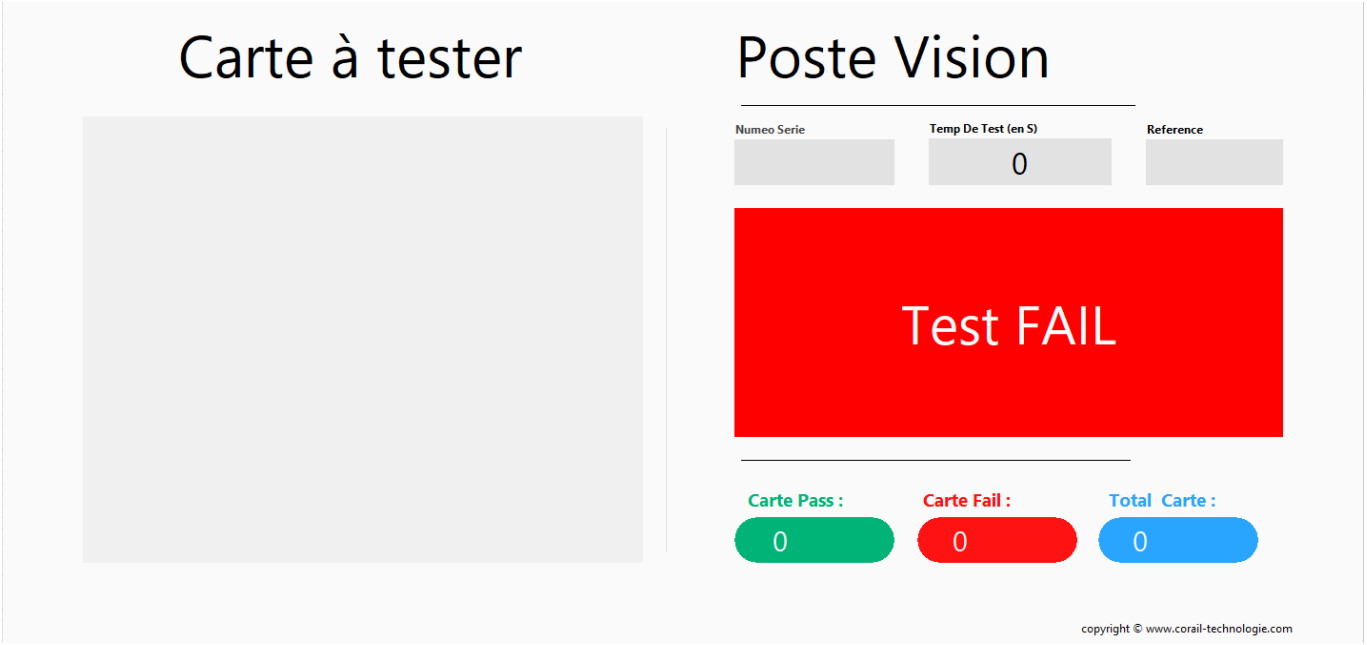

FIGURE 3.15 – L'interface graphique de l'application

3.6 Exemple d'un test « Passed »

Dans cet exemple, on va présenter le cas où notre test est vrai, c'est-à-dire le cas où le composant existe dans la carte et se trouve dans sa bonne position. Dans ce projet, on a décidé de tester un microcontrôleur. La figure 3.16 montre le cas d'un test « Pass ».

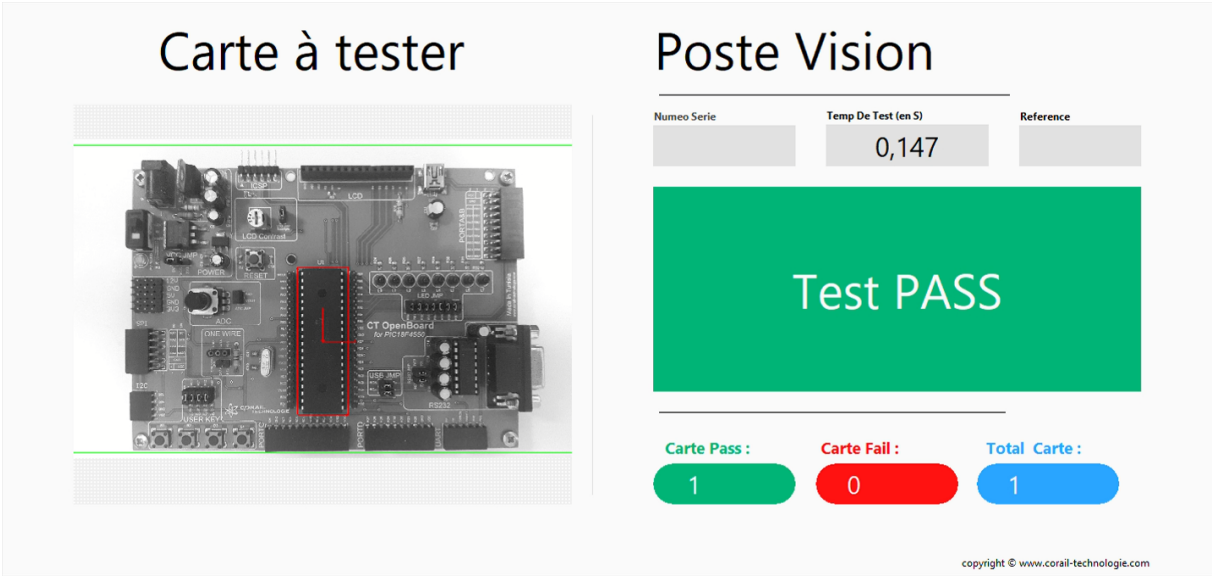

FIGURE 3.16 – Test « Passed »

Le changement du couleur du rouge au vert, qui représente le cas « true », ainsi que l'incrémentation du nombre des cartes « pass » nous indiquent l'existence du microcontrôleur dans la carte électronique. Cette carte nécessite 0,147s pour être testée.

3.7 Exemple d'un test « Failed »

Dans cet exemple, on va présenter le cas où notre test est faux, c'est-à-dire le cas où le composant n'existe pas dans la carte. La figure 3.17 montre le cas d'un test « Fail ».

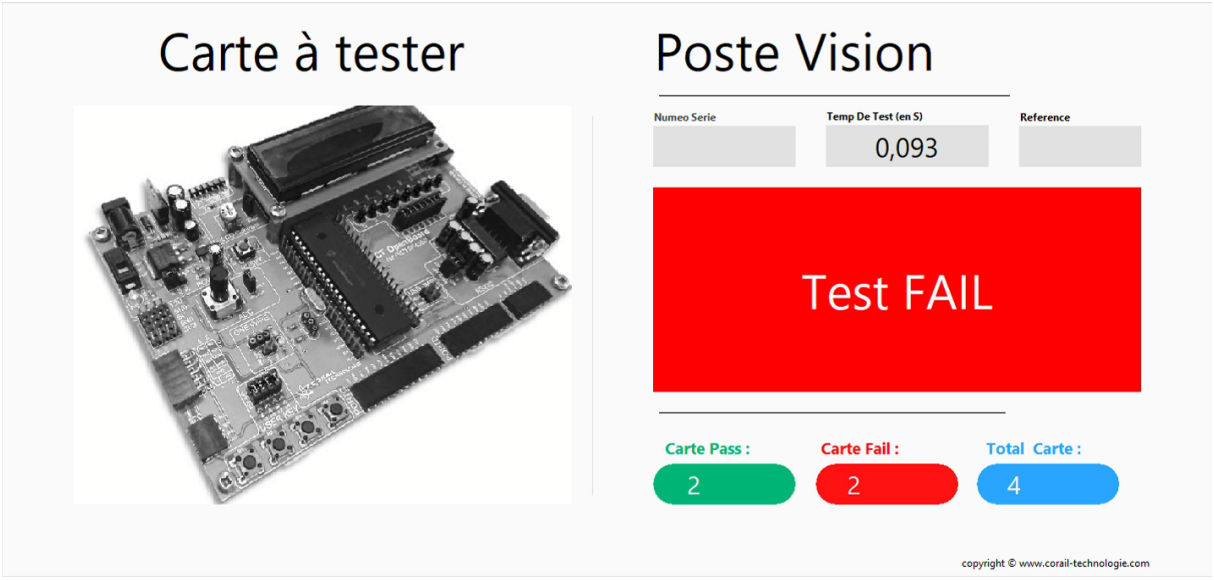

FIGURE 3.17 – Test « Failed »

Le changement du couleur rouge , qui représente le cas « false », ainsi que l'incrémentation du nombre des cartes « fail » nous indiquent que le microcontrôleur est mal positionné dans ce cas. Cette carte nécessite 0,093s pour être testée.

3.8 Conclusion

Au fil de ce chapitre, on a présenté l'environnement matériel et logiciel. Egalement, on a décrit les différents outils des logiciels utilisés pour programmer l'application et nous avons fini par exposer l'interface graphique .

Conclusion Générale

Grace à ce stage, j'ai pu mettre en pratique mes connaissances théoriques acquises durant ma formation à l'ISTIC. Ce projet de fin d'année a été réalisé au sein de la société Corail Technologie. Il consiste à tester la présence d'un composant dans une carte électronique.

Ce projet m'a été d'une part, une opportunité pour découvrir le travail hiérarchique et professionnel au sein de la société et ; apprendre les bonnes pratiques nécessaires à la réalisation d'un produit de qualité d'autre part. Les difficultés inhérentes à ce stage telle que la répartition du temps et des efforts ont été surmontées.

Ce travail a accompli ses objectifs, mais comme toute œuvre humaine il ne prétend pas la perfection et nécessite alors des améliorations. Dans cette optique, Il est possible de réaliser certaines modifications sur ce projet dès que le type du caméra « GigE » sera disponible chez Corail pour pouvoir tester et visualiser les cartes à travers cette camera.