Institut National de Recherche et de Sécurité

Guide pour la mise en oeuvre d'une méthode de validation, par tierce partie, des parties de circuits de commande de presses traitées par une sytème de commande programmable

NOTE SCIENTIFIQUE & TECHNIQUE

Juin 2005

NST N° 252

Guide pour la mise en œuvre d'une méthode de validation, par tierce partie, des parties de circuits de commande de presses traitées par un système de commande programmable

J. BAUDOIN, J.P. BELLO, J.C. BLAISE

Sommaire

1. Introduction

L'évolution croissante de la technologie conduit de plus en plus les constructeurs et les rénovateurs de machines à utiliser des Automates Programmables Industriels dédiés à la Sécurité (APIdS) pour assurer à la fois la commande de ces machines et les fonctions de sécurité des opérateurs. De plus, depuis 1998, la position du ministère chargé du travail a évolué en acceptant sous conditions l'utilisation de certains Automates Programmables Industriels dédiés à la Sécurité (APIdS) pour gérer des fonctions de sécurité sur les machines [1].

La fiabilité, la souplesse d'utilisation et le niveau élevé de sécurité apportés par ces composants dédiés à la sécurité permettent de les intégrer dans les circuits de commande de machines dites "dangereuses" en lieu et place des circuits de commande traditionnels conçus à base de composants électromécaniques.

Par contre, une mauvaise intégration d'un APIdS, tant du point de vue du matériel que du point de vue du logiciel, est susceptible de mettre en cause la sécurité des opérateurs, par exemple, en laissant subsister des risques de démarrages intempestifs d'éléments mobiles dangereux ou en neutralisant les dispositifs de protection des opérateurs.

Les méthodes de conception et de validation utilisées pour les circuits de commande doivent évoluer pour s'adapter aux spécificités des APIdS.

Les automaticiens chargés de mettre en œuvre de tels composants doivent travailler avec des procédures de développement bien formalisées, afin de limiter les risques d'erreur dès la conception.

De leur côté, l'INRS, dans le cadre de ses activités de certification CE des presses travaillant les métaux à froid, ou les organismes de contrôles en général ont besoin d'une méthode les aidant à valider le circuit de commande de machines intégrant des APIdS, dès leur conception et avant leur mise en service.

L'INRS a donc réalisé une étude concernant l'élaboration d'une méthode de validation des circuits de commande utilisant un système de commande programmable. L'étude s'est appuyée sur le cas des presses mécaniques travaillant les métaux à froid compte tenu de l'expérience de l'INRS sur ce type de machines, et sur le cas d'un APIdS représentatif du marché.

La méthode de validation par tierce partie, élaborée lors de cette étude, est décrite dans la suite du document.

2. Principe de la méthode de validation par tierce partie

2.1. Présentation

But et champ d'application de la méthode

Cette méthode a été conçue pour le cas des presses mécaniques travaillant les métaux à froid, mais dans le niveau de détail proposé elle peut également s'appliquer dans le cadre plus général d'un autre type de machine. L'analyse fonctionnelle et la complexité de celle-ci doit toutefois se rapprocher de celles d'une presse intégrant un APIdS ou un dispositif de commande à logique programmée ou paramétrée offrant des similitudes avec le dispositif étudié (architecture, utilisation de « blocs fonctionnels constructeur » validés, séparation des fonctions de sécurité de la partie fonctionnelle,…).

Le principe de la méthode peut quant à lui s'appliquer plus largement à d'autres types de systèmes automatisés de production.

Public

La méthode de validation s'adresse à une tierce partie, tel qu'un organisme de contrôle ayant en charge la validation d'équipements de travail. Cette tierce partie sera appelée "vérificateur" dans le reste du document.

Principe de la méthode

Il n'est plus possible de valider un système de commande complexe lorsqu'il est achevé. Le processus de développement lui-même constituant un des maillons essentiels de la validation [2] [3] [4], le vérificateur doit être impliqué le plus en amont possible. Pour traiter correctement la sécurité, il est nécessaire d'avoir une vue des différentes étapes de développement d'une application.

La méthode de validation propose d'analyser chaque étape du cycle de développement d'un circuit de commande, sans procéder à un examen précis des équations logiques (lignes de code) du logiciel applicatif. Elle s'appuie sur des documents, des informations et des résultats de tests fournis par le concepteur. Il est donc impératif que la conception soit réalisée avec des **procédures de développement formalisées** [5] [6] [7].

Pour toutes ces raisons, la méthode de validation nécessite une confiance justifiée entre le vérificateur et le concepteur telle que présentée au § 3.

Enfin, bien que la méthode s'appuie sur certains principes ou documents recommandés par les normes habituellement utilisées en génie logiciel [8], elle n'en impose aucun. Elle s'appuie principalement sur les référentiels techniques ou réglementaires cités dans le cahier des charges.

Avertissement : Ce document est un guide ; en ce sens, il n'est pas utilisable en l'état pour un processus de validation. Il fixe dans les chapitres 2

et 3 une approche pour aborder la problématique de validation puis décrit dans les chapitres 4 et 5 les éléments devant être analysés.

Tout vérificateur désirant mettre en œuvre la méthode proposée doit dans un premier temps être en accord avec l'approche présentée, puis, dans un second temps, décliner cette approche en l'adaptant à son contexte. En effet, le contenu de certaines parties doit être adapté au cas précis qu'il doit traiter compte tenu des spécificités de la machine, de l'APIdS et du processus de développement proposé par le concepteur.

Comment la méthode proposée s'inscrit-elle par rapport à la validation globale d'une machine ?

Par rapport au cycle de développement global d'une machine suivi par le concepteur, représenté figure 1, la méthode présentée ici traite de la partie entourée en couleur verte. C'est la seule partie où la méthode de validation est influencée par l'utilisation d'un APIdS. La validation des parties plus en amont (circuit de commande global, machine,…) devant se dérouler suivant les mêmes règles que pour un câblage électromécanique, il n'a pas été jugé utile de la traiter dans ce document.

L'analyse des risques est supposée avoir déjà été effectuée.

La méthode se base sur un cahier des charges de la partie commande finalisé prenant en compte une demande client et un référentiel réglementaire (par exemple, la Directive Machines [9]) ou technique (par exemple une norme harmonisée EN 692 [10], ou les documents ED 782 [11], ED 783 [12] de l'INRS).

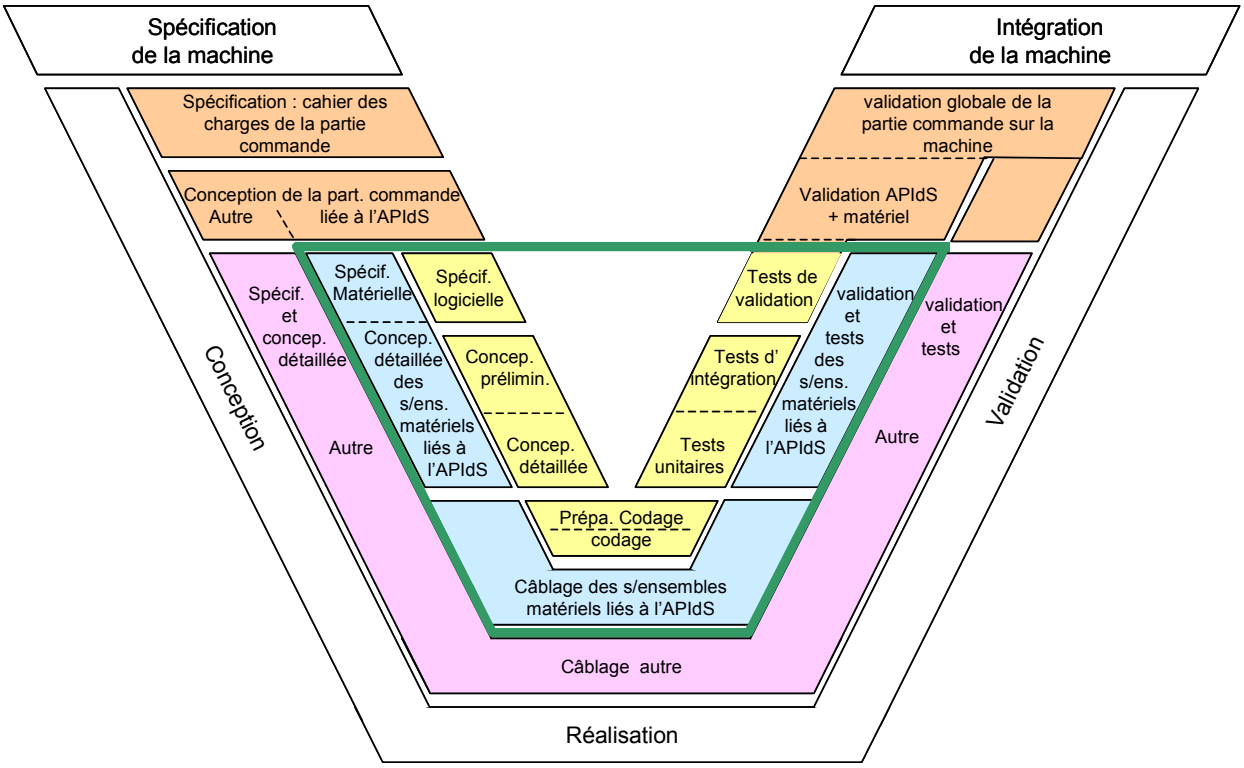

Figure 1 : Cycle de développement en « V » d'une machine d'après [2]

Comment la méthode s'inscrit-elle par rapport à la validation de la partie commande ?

Un circuit de commande de machine utilisant un APIdS comprend souvent également une partie traitée en logique câblée telle que par exemple celle représentée figure 2. La présente méthode ne s'intéresse qu'à la partie traitée par l'APIdS (cf. Annexe B). Il n'a pas été jugé utile de traiter dans ce document de l'éventuelle partie traitée en logique câblée (électromécanique).

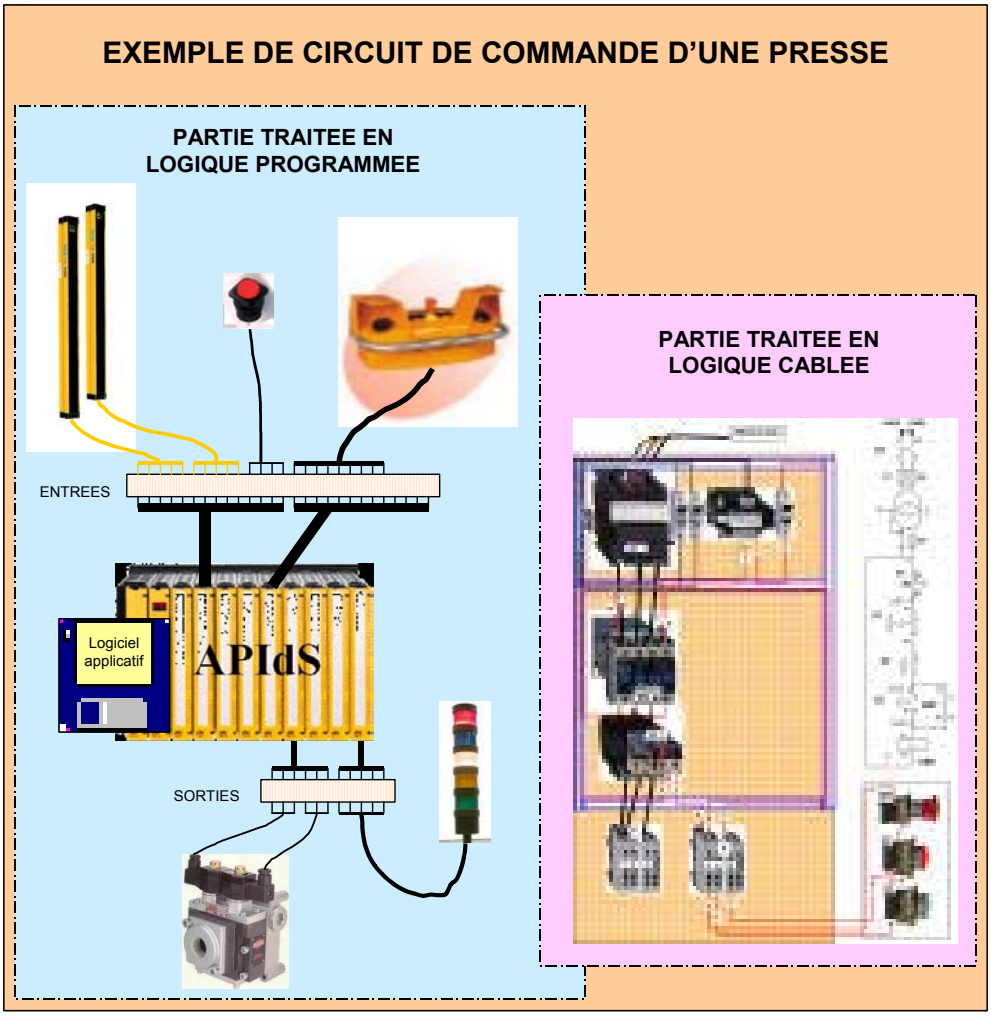

Figure 2 : Exemple d'illustration d'un circuit de commande d'une presse

La méthode proposée est prévue pour valider le logiciel applicatif (programme principal de l'APIdS), la partie matérielle liée à l'APIdS, depuis le choix jusqu'à la mise en oeuvre (type d'APIdS, raccordement des entrées sorties,…) ainsi que la documentation.

Pour le logiciel, elle traite :

- de la partie relative à la sécurité des opérateurs,
- éventuellement, d'une partie logicielle ne traitant pas de sécurité des opérateurs, dans le cas où celle-ci pourrait influencer la précédente.

2.2. Représentation schématique de la méthode

La méthode de validation proposée s'inscrit dans le cadre d'un suivi du cycle de développement utilisé par le concepteur. Dans l'exemple de la figure 3 il est schématisé par un cycle de développement logiciel en "V" représenté en couleur jaune qui se compose de deux branches :

- La branche de gauche, descendante, représente les étapes d'études et de réalisation (appelées communément étapes de conception). La traçabilité est assurée par la réalisation d'un dossier propre à chaque étape ;
- La branche de droite, montante, représente les étapes de tests des opérations de conception.

Une relation constante s'établit entre les étapes de conception et de tests. Les étapes de tests consistent à vérifier la bonne réalisation des étapes de conception sur la base des plans de tests réalisés lors de chacune de ces étapes.

La méthode proposée consiste à suivre pas à pas ce cycle en associant à chaque étape de développement (activité du concepteur), une étape de vérification (activité du vérificateur) tel que représenté en couleur verte sur la Figure 3.

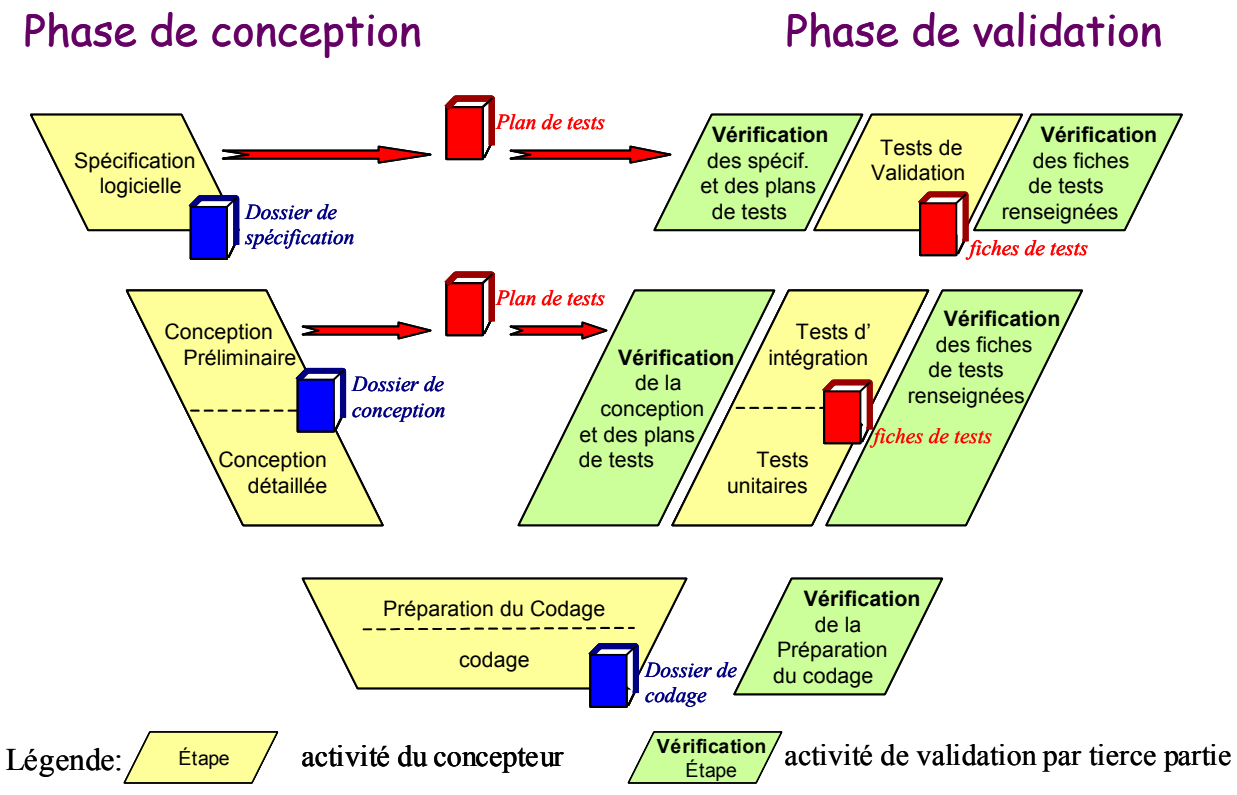

Figure 3 : Représentation schématique du cadre de la méthode de validation proposée

Cette méthode permet :

- de s'assurer que toutes les étapes d'un cycle de développement en V sont traitées par le concepteur,

- éventuellement de corriger des erreurs ou de modifier le développement avant qu'il ne soit trop tard,
- d'acquérir un niveau de confiance justifié par la procédure de développement et le sérieux mis en œuvre par le concepteur.

Parmi les tâches effectuées par **le concepteur,** la méthode de validation attachera une importance particulière à :

- l'élaboration des dossiers lors de la branche descendante du cycle de développement,
- la préparation des tests lors de la branche descendante du cycle de développement,
- la consignation des résultats de tests lors de la branche montante du cycle de développement.

La méthode précise des vérifications à effectuer, parmi lesquelles :

- l'identification des documents (date, version,…),
- la pertinence des documents par rapport à l'étape concernée,
- la cohérence des informations par rapport à l'étape précédente ,
- la suffisance de l'information,…

3. « Convention » entre le vérificateur et le concepteur

La méthode de validation proposée s'appuie sur une procédure de développement formalisée conduite par le concepteur, elle nécessite une confiance justifiée entre le vérificateur et le concepteur. Il est important de définir au préalable du développement, un cadre d'échange et de coopération appelé ici « convention » entre le vérificateur et le concepteur.

Lors de l'établissement de cette convention, il est important d'aborder les thèmes décrits dans les paragraphes suivants.

3.1. Présentation du projet par le concepteur

Le concepteur doit présenter d'une manière générale l'équipement de travail concerné avec une description sommaire des différents éléments qui le composent ainsi que l'organisation interne de l'équipe en charge de la réalisation de l'équipement de travail.

Cette présentation, notamment de la complexité de l'équipement de travail et du système programmable associé, permet au vérificateur de mesurer l'importance de la tâche à réaliser pour valider le circuit de commande.

3.2. Coordination avec le concepteur

Le concepteur expose sa procédure de développement formalisée des circuits de commande de machine. Le vérificateur expose le cadre de la méthode de validation. Un échange doit s'engager sur la manière de coordonner les deux actions et la manière selon laquelle le concepteur entend répondre à chacune des requêtes du vérificateur.

3.3. Outils et moyens mis en œuvre par le concepteur

Le concepteur doit présenter les moyens de conception et d'essais prévus pour les différentes étapes du cycle de développement du circuit de commande.

3.4. Eléments de conclusion

Les échanges entre le concepteur et le vérificateur doivent notamment permettre de s'assurer :

- que la méthode de développement du concepteur est bien formalisée,
- de la compatibilité entre la méthode de développement et celle de validation et des éventuels aménagements à prévoir,
- qu'un niveau de confiance justifié peut s'établir entre le concepteur et le vérificateur.

4. Analyse technique et documentaire de la partie matérielle

Cette analyse concerne le matériel en lien avec l'APIdS, depuis le choix jusqu'à la mise en œuvre du matériel (type d'APIdS, configuration de l'APIdS et de ses périphériques, raccordement des entrées sorties,…). Le cycle de développement correspondant à cette analyse est représenté en couleur bleue sur la figure 4.

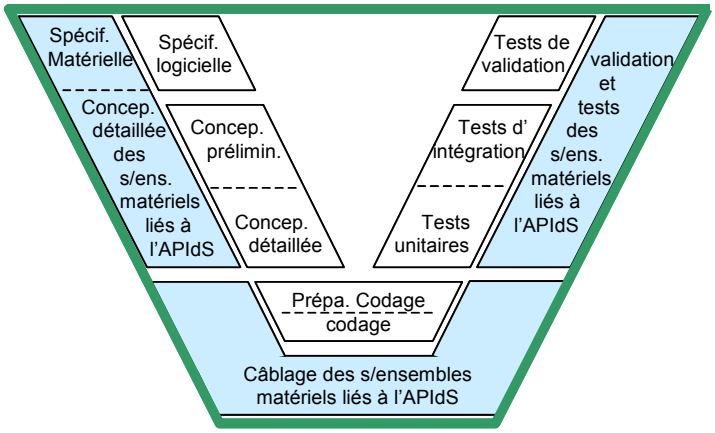

Figure 4 : Mise en évidence du cadre de la méthode dans le cycle en V – partie matérielle

D'une manière générale, les documents fournis doivent être clairement identifiés (nom, projet, version, date, etc.) et le vérificateur s'attachera à contrôler cette exigence.

4.1. Vérification des spécifications et de la conception de la partie matérielle liée à l'APIdS

4.1.1. Livrables à fournir par le concepteur

Les documents fournis doivent comprendre entre autres :

- référence de l'APIds et des cartes d'entrée/sorties (marque, type, notice technique, certificats),
- un document de description de la configuration matérielle de l'APIdS, emplacement des cartes, type de cartes et d'autres informations, …
- tableau regroupant l'ensemble des entrée-sorties de l'automate élaboré lors de la préparation du codage,
- schéma de câblage de l'APIdS et des entrées/sorties,
- nomenclature et caractéristiques des capteurs et pré-actionneurs.

4.1.2. Tâches du vérificateur

a. Caractéristiques de l'APIdS et des cartes d'entrée/sortie

Vérifier que l'APIdS et les cartes d'entrées/sorties retenues sont orientées vers la commande des machines, c'est à dire qu'elles ont été conçues pour interrompre un mouvement dangereux en cas d'apparition d'une défaillance interne.

Les exigences en terme de performance de sécurité de l'APIdS et des cartes d'entrée/sortie doivent être examinées par rapport :

- aux performances de sécurité des fonctions à réaliser, exprimées en termes de SIL (suivant la norme IEC 62061 [13]), de niveau de performance de sécurité (PL suivant le projet de norme ISO 13849-1 [14]) ou de catégories de sécurité (suivant la norme EN 954-1 [15]),
- aux exigences environnementales (telles que CEM) définies suivant les caractéristiques de l'application à réaliser.

b. Configuration matérielle

- vérification de l'adéquation entre les entrées et le type de carte utilisé (standard, sécurité, temps de réponse,…). Il est nécessaire de s'assurer de l'adéquation entre les temps d'acquisition des cartes et la durée des événements à contrôler,
- même vérification pour les sorties (mono canal, redondance ou impulsionnelles).

c. Câblage des alimentations

Vérification des règles élémentaires de câblage (voir la norme EN 60204-1 [16]) et des tensions utilisées.

d. Câblage des entrées/sorties

Vérification sur le schéma de câblage, pour chaque entrée et sortie définie lors de la préparation du codage :

- de sa prise en compte,
- de la nature des contacts utilisés (O, F) pour les entrées,
- de son identification, en utilisant la même que celle utilisée lors des étapes de conception du logiciel,
- du respect des règles élémentaires de câblage (EN 60204-1 [16]),
- du respect des règles de câblage imposées par des fonctions spécifiques (ex : blocs fonctionnels constructeur), par les niveaux de sécurité associés à ces fonctions (ED 905 [17]),…
- des potentiels utilisés,
- de la bonne utilisation des sorties impulsionnelles (ne pas alimenter d'entrées contiguës avec la même sortie impulsionnelle,…).

Pour les entrées (par exemple capteur, bouton poussoir) utilisant des signaux impulsionnels, si des borniers intermédiaires sont utilisés, les bornes de raccordement de ces entrées ne doivent jamais être placées de manière contiguës (exemple : pas de bornes contiguës pour un contact à manœuvre positif d'ouverture d'un capteur de position d'un protecteur utilisé seul).

e. Caractéristiques des capteurs et pré-actionneurs

Vérification :

- de l'adéquation des caractéristiques des capteurs et pré-actionneurs par rapport aux spécifications,
- des éventuelles règles particulières de raccordement des capteurs et pré-actionneurs (cas de certains barrages immatériels,…).

4.2. Validation de la partie matérielle liée à l'APIdS

Cette validation devant se dérouler suivant les mêmes règles que pour un câblage électromécanique, il n'a pas été jugé utile de la traiter dans ce document.

5. Analyse technique et documentaire de la partie logicielle

Le cycle de développement logiciel correspondant à cette analyse est représenté en couleur jaune sur la figure 5. La partie entourée de pointillés verts représente le cadre possible de réalisation des tests de validation du logiciel. Elle est développée au § 5.1.1.c

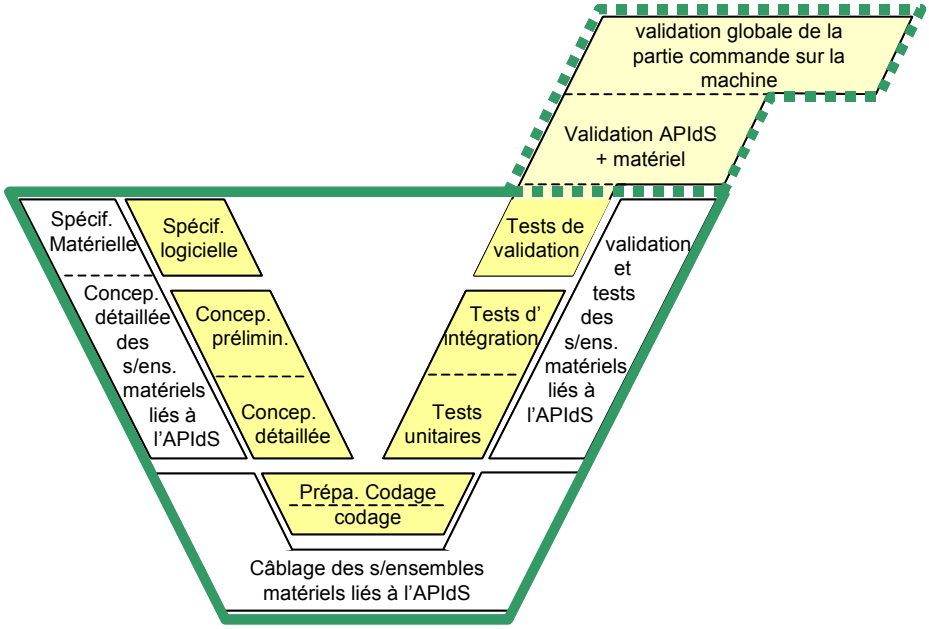

Figure 5 : Mise en évidence du cadre de la méthode dans le cycle en V – partie logicielle

D'une manière générale, les documents fournis doivent être clairement identifiés (nom, projet, version, date, etc.) et le vérificateur s'attachera à contrôler cette exigence.

Note : La partie « *remarque* » fournit des informations complémentaires sur les documents à fournir ou pour faciliter la compréhension des vérifications proposées.

5.1. Vérification des spécifications du logiciel

5.1.1. Livrables à fournir par le concepteur

a. Cahier des charges de la machine

La méthode de validation se base sur un document appelé « cahier des charges » qui doit comprendre au moins une description technique de la machine, des référentiels utilisés, des modes de marche et de protection. Ce document sert de base à l'élaboration des spécifications.

1. Une description technique de la machine

Les caractéristiques principales de la machine sont par exemple :

- catégorie (PM, PH, PPH pour une presse),
- type d'embrayage frein,
- cadence de travail, vitesse variable,
- course variable,
- présentation de la machine (Plan ou photo montrant la machine et les zones dangereuses prises en compte, rappel de la cinématique,…).

2. Une description des référentiels utilisés

Un référentiel réglementaire (par exemple, la Directive Machines [9]) ou technique (par exemple une norme harmonisée EN 692 [10], ou les documents ED 782 [11], ED 783 [12] de l'INRS).

3. Une description des modes de marche et de protection

Celle-ci devra permettre d'expliciter :

- les modes de marche attendus, compte tenu de la sollicitation des organes de commande et des différents capteurs de position,
- les moyens de protection mis en œuvre, notamment ceux traités par la logique de commande, avec leur interaction sur les modes de marche,
- certains choix technologiques effectués a priori, tel que le type d'électrovanne, le nombre de détecteurs de position d'un protecteur, l'utilisation d'un composant de sécurité,…

b. Dossier de spécification logicielle

Les spécifications du logiciel se présenteront sous la forme d'un document texte et de tout support pouvant contribuer à leur précision et à leur compréhension (tableau, grafcet, synoptique, etc.).

Les spécifications doivent être les plus précises possibles, sans ambiguïté et ne pas laisser place à l'improvisation du ou des développeurs du code.

Remarque : Ce document doit faire le lien entre deux catégories professionnelles ayant des connaissances et des langages techniques différents, un concepteur ou un utilisateur de machine et un programmeur. Les termes utilisés doivent donc pouvoir être compris par les deux parties.

Les spécifications du logiciel doivent prendre en compte différents aspects :

- Fonctionnel ; cet aspect découle directement du cahier des charges en décrivant dans le détail le déroulement des modes de fonctionnement,
- Sécurité ; cet aspect doit surtout respecter les exigences en matière de sécurité des référentiels retenus dans le cahier des charges,
- Métier ; la prise en compte de cet aspect implique une très bonne connaissance de la machine, des modes de fonctionnement couramment rencontrés et de la terminologie employée par les utilisateurs afin de penser à décrire des fonctionnalités inhérentes à ce type de machine, sous-entendues mais non exprimées dans le cahier des charges,

- contraintes imposées au logiciel par le matériel liées par exemple à l'architecture de l'APIdS, à l'existence de blocs fonctionnels constructeur,…

c. Plan de tests de validation

Il est composé de trois parties [18] [19]

1. **Les techniques de test**

   Les scénarios prévus sont détaillés en se basant sur les spécifications, par exemple l'état des sorties attendu en fonction de la sollicitation des entrées (test boite noire[1]). Celles-ci doivent être considérées, en état normal, en état dégradé et lorsque la fonction comprend des paramètres numériques, avec des valeurs normales (nominales), aux limites et en dehors.

2. **Les moyens de test**

   Ce sont le mode opératoire et les moyens à mettre en œuvre pour effectuer les tests tel que des outils (simulateurs,…), des environnements de test (configurations différentes du système,…), des jeux d'essais préenregistrés.

3. **Les livrables prévus (fiches de tests)**

   Le type de présentation des résultats est laissé à l'initiative du concepteur (par exemple, fichiers issus d'un atelier logiciel, enregistrement numérique, trace papier,…).

Note :

La réalisation des tests de validation du logiciel nécessite généralement la mise en œuvre de nombreux échanges entre le logiciel applicatif (programme principal) et la partie opérative matérialisée par des entrées (capteurs) et des sorties (actionneurs).

Pour réaliser ces tests, en dehors de leur support matériel (APIdS), le logiciel applicatif devrait avoir été développé avec un atelier de génie logiciel indépendant de l'APIdS, comportant un module de simulation de partie opérative.

Dans la pratique, peu d'industriels disposent de l'équipement nécessaire.

Deux autres voies s'offrent au concepteur pour réaliser les tests de validation du logiciel en les intégrant dans les étapes représentées figure 6 :

- étape de « validation APIdS + matériel », en dehors de la machine, mais en utilisant une partie opérative physique représentative de la machine ou un simulateur informatique de partie opérative,
- étape de « validation globale de la partie commande sur la machine ».

[1] Test boite noire (ou test fonctionnel) : Pas de visibilité du composant sous test, les entrées et résultats attendus sont exprimés en terme de comportement du composant logiciel du point de vue externe.

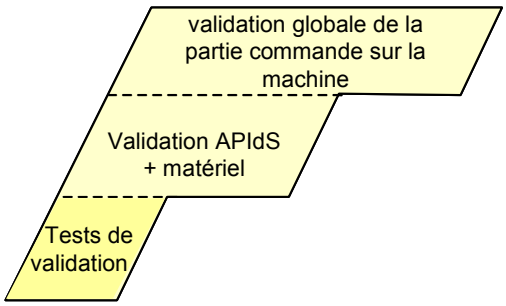

Figure 6 : Cadre possible de réalisation des tests de validation du logiciel

Si les tests de validation du logiciel sont réalisés sur la machine, les conditions de réalisation de ces tests doivent être convenues entre le concepteur et le vérificateur. Cette façon d'opérer présente certains inconvénients (nécessité d'attendre que la machine soit opérationnelle pour effectuer les tests, risques pour la machine ou pour l'opérateur dus à la nature même de certains tests en présence de défaillance,…) que l'on ne rencontre pas en utilisant un outil de simulation de partie opérative (cf. Annexe A).

5.1.2. Tâches du vérificateur

a. Examen du dossier de spécification logicielle

- juger le niveau de précision et la suffisance de la description des modes de marche (exemple : organe de commande, impulsion ou action maintenue, phase de descente et de montée, etc.),
- vérifier que tous les aspects "sécurité" (suivant le référentiel de conception choisi, directive, norme,…) relatifs à ce type de machine ont bien été décrits, dans les différentes phases de fonctionnement (descente, remontée automatique, inhibition, etc.) et pour les différents protecteurs et/ou dispositifs de protection (protecteur avant, arrière, barrage immatériel,…),
- s'assurer que les spécifications ne contiennent pas de termes nécessitant une connaissance "métier" de la personne en charge du développement tel que des termes propres à la logique de commande des presses (remontée automatique, anti-répétition, etc.).

Pour des modes de marche couramment utilisés, les spécifications peuvent être comparées avec un canevas préétabli, mais pour une grande partie l'appréciation des spécifications reste basée sur l'expérience du "vérificateur".

b. Examen du plan de tests de validation

- juger du niveau de détail et de la suffisance des scénarios prévus,
- juger de la pertinence du mode opératoire et des moyens à mettre en œuvre pour effectuer les tests,
- apprécier le niveau d'information et de traçabilité pouvant être procuré par la présentation prévue des résultats.

Le vérificateur peut, en cas de besoin, demander des précisions ou des scénarios de tests complémentaires nécessaires pour la phase de validation.

5.2. Vérification de la conception préliminaire (architecture du logiciel)

5.2.1. Livrables à fournir par le concepteur :

a. Dossier de conception préliminaire

Le concepteur doit fournir au moins les documents suivants :

Structure

Un document présentant la structure générale du futur logiciel applicatif (partie traitant de la sécurité et partie purement fonctionnelle). Il devra être fourni sous la forme d'une arborescence des différentes "tâches" et "unités d'organisation de programme" POUs[2] (programmes, blocs fonctionnels,…) utilisées avec une description sommaire du contenu de chacun de ces éléments.

Un document explicitant les mesures mises en œuvre pour que les parties fonctionnelles n'influent pas sur la partie traitant de la sécurité (pas de traitement de parties fonctionnelles dans le programme principal traitant des fonctions de sécurité) et le cas échéant comment dialoguent entre elles ces deux parties.

Réutilisabilité du programme/des blocs fonctionnels

Dans le cas de conception d'une série de machines différentes (prévision d'équipements optionnels, nombre de capteurs/actionneurs différent, …) un document décrivant les mesures mises en œuvre pour assurer la réutilisabilité[2] de tout ou partie du programme principal (par exemple, choix de paramétrage et/ou d'adressage indirect).

Remarque : Un logiciel applicatif de presse est rarement utilisé pour une seule machine, il est souvent réutilisé pour une application voisine nécessitant quelques modifications et la configuration matérielle de l'automate peut être très différente d'une application à l'autre. Dans ce cas l'aspect de portabilité[2] du programme principal doit être pris en considération.

b. Plan de tests d'intégration

Il est composé de trois parties

1. **Les techniques de test :**
   - Les scénarios prévus sont détaillés en se basant sur les spécifications, par exemple l'état des sorties attendu en fonction de la sollicitation des entrées (test boite noire sur un ensemble de fonctions liées en elles),

[2] Program Organisation Unit – voir CEI 61131-8 [19]

- dans le cas d'un test boite blanche[3] le plan de tests doit permettre de :
    - rechercher des erreurs latentes (dues à l'activation d'une mémoire, d'un compteur, d'une temporisation,…) n'ayant pas été détectées lors des tests boite noire et qui pourraient avoir des conséquences ultérieures indésirables,
    - vérifier les liens et l'enchaînement correct des programmes, des blocs fonctionnels, des différentes branches parcourues ainsi que le séquencement temporel des traitements.

2. **Les moyens de test**

   Ce sont le mode opératoire et les moyens à mettre en œuvre pour effectuer les tests tel que des outils (simulateurs,…), des environnements de test (configurations différentes du système,…), des jeux d'essais préenregistrés.

3. **Les livrables prévus (fiches de tests)**

   Le type de présentation des résultats est laissé à l'initiative du concepteur (par exemple, fichiers issus d'un atelier logiciel, enregistrement numérique, trace papier,…).

5.2.2. Tâche du vérificateur

a. Examen des documents de la conception préliminaire

Structure du futur logiciel

- vérifier la cohérence de l'architecture proposée avec les spécifications,
- vérifier que le programme traitant de la sécurité est séparé de celui traitant de la partie purement fonctionnelle (par exemple deux parties distinctes de l'APIdS).
- juger les mesures mises en œuvre pour que les parties fonctionnelles n'influent pas sur la partie traitant de la sécurité et le cas échéant comment dialoguent entre elles les parties traitant de la sécurité et celles qui sont purement fonctionnelles.

  Remarque : Dans le cas de certains APIdS, c'est une séparation physique qui est proposée et qui doit être utilisée. Cette option offre plusieurs avantages : si l'utilisateur de la machine doit modifier certains paramètres d'exploitation ou modifier une séquence de fonctionnement, indépendante de la sécurité des opérateurs, il peut le faire sans risquer de remettre en cause la sécurité de la machine. D'autre part le verrouillage par mot de passe de la partie sécurité offre une garantie d'intégrité supplémentaire pour cette partie.
- juger du découpage du programme principal (granulométrie), retrouver au minimum les fonctions identifiées dans les spécifications logicielles. Par exemple, les blocs fonctionnels traitant :
    - o des dispositifs de protection, par exemple barrage immatériel,

[3] Test boite blanche (ou test structurel) : Visibilité du composant sous test, les jeux de test sont produits en analysant le code source.

- des protecteurs, suivant les zones de la machine,
- des modes de marche,
- …

doivent être facilement identifiables. L'interaction entre ces différents blocs fonctionnels doit également apparaître.

Remarque : Ce découpage en fonctions « élémentaires » permet leur identification précise, facilite leur mise au point et leur validation, permet leur réutilisation à l'intérieur du même logiciel ou pour d'autres applications futures.

- vérifier dans le cas d'utilisation de blocs fonctionnels « constructeur » que leur comportement correspond aux spécifications attendues (fonctionnement, limite d'utilisation, niveau de sécurité, etc.),
- vérifier que les blocs n'ont pas été détournés de leur fonction initiale.

Remarque : Lorsque les spécifications techniques de ces blocs ne correspondent pas exactement à la fonction attendue, ou que les spécifications de certains blocs ne sont pas suffisantes ou sont mal assimilées, il est préférable de développer un nouveau bloc fonctionnel ou un programme correspondant à cette fonction plutôt que de mal l'adapter ou de le détourner de sa fonction première.

Réutilisabilité du programme principal (le cas échéant)

- juger les mesures mises en œuvre pour faciliter la réutilisation du programme.

Remarque : Vérifier que le concepteur a mis en œuvre les mesures suivantes ou des solutions équivalentes :

1. ***Paramétrage** - Lorsque des paramètres du logiciel affectant des fonctions de sécurité sont susceptibles d'être modifiés en fonction de paramètres de la machine (cadence, discordance, etc.), il est souhaitable de procéder à un paramétrage indirect permettant de regrouper tous les paramètres machine dans un même bloc, de filtrer ces valeurs (bornage) afin que les modifications autorisées ne puissent pas dégrader le niveau de sécurité et permettant des commentaires explicites.*
2. ***Adressage indirect** - Si le logiciel applicatif est destiné à être réutilisé pour tout ou partie avec des configurations matérielles plus ou moins différentes, il est intéressant d'utiliser indirectement les entrées/sorties par l'intermédiaires de bits internes. Cet adressage se fera dans un bloc unique évitant ainsi d'avoir à intervenir dans différents endroits du programme principal.*

b. Examen du plan de tests d'intégration

- juger du niveau de détail et de la suffisance des scénarios prévus (test d'intégration),
- juger de la pertinence du mode opératoire et des moyens à mettre en œuvre pour effectuer les tests,

- apprécier le niveau de l'information et de traçabilité pouvant être procuré par la présentation prévue des résultats.

Le vérificateur peut, en cas de besoin, demander des précisions ou des scénarios de tests complémentaires nécessaires pour la phase de validation.

5.3. Vérification de la conception détaillée

5.3.1. Documents à fournir par le concepteur

a. Dossier de conception détaillée

Chacun des blocs fonctionnels utilisés sera accompagné :

- d'une identification précise (nom, fonction, et pour les blocs fonctionnels réutilisables, date, version),
- d'une spécification détaillée de la fonction attendue,
- d'une interface de raccordement (entrées, sorties) aux autres éléments du programme.

Dans le cas des blocs fonctionnels « constructeur » pré-programmés, la notice d'instructions du constructeur devra remplir cette fonction.

b. Plan de tests unitaires

Il est composé de trois parties :

1. **Les techniques de test :**
   - Les scénarios prévus sont détaillés en se basant sur les spécifications détaillées, par exemple l'état attendu des sorties du bloc fonctionnel par rapport à la sollicitation des entrées (test boite noire). Celles-ci doivent être considérées, en état normal, en état dégradé et lorsque la fonction comprend des paramètres numériques avec des valeurs normales (nominales), aux limites et en dehors.
2. **Les moyens de test**

   Ce sont le mode opératoire et les moyens à mettre en œuvre pour effectuer les tests tel que des outils (simulateurs,…), des environnements de test (configurations différentes du système,…), des jeux d'essais préenregistrés.
3. **Les livrables prévus**

   Le type de présentation des résultats est laissé à l'initiative du concepteur (par exemple, fichiers issus d'un atelier logiciel, enregistrement numérique, trace papier,…).

Dans le cas d'utilisation de blocs fonctionnels « constructeur » déjà validés, la notice du constructeur devra préciser les conditions de cette validation (en fournissant par exemple une copie d'un certificat).

5.3.2. Tâche du vérificateur

a. Examen des documents de la conception détaillée

Pour les blocs fonctionnels « constructeur » :

- s'assurer que les spécifications du constructeur (ou comportement du bloc) correspondent aux spécifications du système à valider (fonctionnement, limite d'utilisation, niveau de sécurité, etc.).
- vérifier que les blocs n'ont pas été détournés de leur fonction initiale.

  Remarque : Lorsque les spécifications techniques de ces blocs ne correspondent pas exactement à la fonction attendue, ou que les spécifications de certains blocs ne sont pas suffisantes ou sont mal assimilées, il est préférable de développer un nouveau bloc fonctionnel ou un programme correspondant à cette fonction plutôt que de mal l'adapter ou de détourner un bloc de sa fonction première.

Pour les blocs fonctionnels à développer par le concepteur, la tâche consistera à :

- juger le niveau de précision et la suffisance de la description de la fonction (identification fournissant le nom du bloc fonctionnel, un descriptif fonctionnel, un numéro de version du bloc),
- juger la pertinence de l'architecture du bloc fonctionnel (variables internes, externes) et éventuellement de sa capacité à être réutilisé.

  Remarque (éléments d'appréciation) :

  - *Aspect « limitation du couplage » : l'intégration et la suppression d'un bloc fonctionnel sont facilitées lorsque celui-ci est conçu de manière à limiter ses échanges avec le reste du programme.*
  - *Aspect « cohérence » : les traitements effectués dans un bloc fonctionnel doivent tous contribuer à la réalisation d'une même fonctionnalité.*
  - *Aspect « limite de décomposition » : la décomposition des fonctions principales en fonctions élémentaires permet leur réutilisation dans un plus grand nombre d'applications. Une trop grande décomposition augmente les temps d'intégration, une faible décomposition diminue les possibilités de réutilisation.*
- Pour les fonctions couramment utilisées (arrêt d'urgence, contrôle protecteur, etc.) les spécifications en conception détaillée peuvent être comparées avec un canevas préétabli, mais pour une grande partie l'appréciation des spécifications reste basée sur l'expérience du "vérificateur".

b. Examen du plan de tests unitaires

- juger du niveau de détail et de la suffisance des scénarios prévus,
- juger de la pertinence du mode opératoire et des moyens à mettre en œuvre pour effectuer les tests,
- apprécier le niveau de l'information et de traçabilité pouvant être procuré par la présentation prévue des résultats,

- dans le cas d'utilisation de blocs fonctionnels « constructeur » déjà validés, la notice du fabricant et les certificats correspondants seront examinés à la place du plan de tests correspondant.

Le vérificateur peut, en cas de besoin, demander des précisions ou des scénarios de tests complémentaires nécessaires pour la phase de validation.

5.4. vérification de la préparation du codage

5.4.1. Dossier de préparation du codage

Certains des documents existants, préparés lors des étapes précédentes, éventuellement complétés ou amendés, peuvent être utilisés pour constituer ce dossier.

Structure

Un document présentant la structure générale du futur logiciel applicatif (parties traitant de la sécurité et partie purement fonctionnelle) devra être fourni sous la forme d'une arborescence des différentes tâches et unités d'organisation du programme utilisées avec les appellations propres à l'APIdS utilisé (par exemple blocs OB, FB, SB) renseignées (par exemple OB101, FB1).

Zone mémoire

Un document devra présenter le découpage de la zone mémoire qu'il est envisagé d'utiliser.

Entrées/sorties

Fournir un tableau regroupant l'ensemble des entrées-sorties de l'automate.

Pour les entrées, le tableau fera apparaître :

- le numéro de la variable (par exemple I 0.01),
- le nom précis de la variable (par exemple mnémonique),
- la désignation de la variable,
- le type de contact utilisé pour activer cette variable (O ou F),
- l'alimentation de la variable (simple : 24vDC ou impulsionnelle).

Pour les sorties, le tableau fera apparaître :

- le numéro de la variable (par exemple O 1.01),
- le nom précis de la variable,
- la désignation de la variable,
- le type de sortie (mono-canal, redondante ou impulsionnelle).

Remarque : Le nom et la désignation des variables utilisés dans ce tableau doivent être comparables à ceux qui seront utilisés lors du codage et qui apparaîtront sur le listing des entrées/sorties.

Configuration logicielle de l'APIdS

Fournir un document récapitulant les paramètres généraux de l'APIdS. Par exemple, il s'agit des valeurs retenues pour certains temps de réponse à renseigner compte tenu de la technologie des composants externes à l'APIdS, de la déclaration des entrées/sorties impulsionnelles,…

5.4.2. Tâche du vérificateur

Structure

- Vérifier que les règles de structuration et d'organisation du programme principal proposées par le fabricant de l'automate sont respectées.
- Vérifier que le traitement des entrées, de la gestion du processus et des sorties est matérialisé par des zones de programme distinctes (dans la mesure du possible).

Zone mémoire

- Vérifier que :
  - les différentes variables ne se chevauchent pas afin de ne pas risquer des erreurs,
  - la zone mémoire autorisée a été découpée par le concepteur en fonction des variables utiles, zone bit, zone octet et zone mot (la zone octet étant généralement très peu utilisée).
- Vérifier que les zones mémoires, potentiellement réservées pour être utilisées par les blocs fonctionnels constructeurs (BFC), ne sont pas utilisés dans le programme principal.

  Remarque : Dans certains automates, ces zones mémoires sont accessibles en écriture avec le risque d'écraser des données et de perturber le fonctionnement de ces blocs.

  La réservation d'une zone pour les BFC permet d'ajouter postérieurement des BFC qui n'étaient pas prévus sans nécessité de remettre en cause tout le programme.
- Pour les variables internes des blocs fonctionnels, développés par le concepteur du logiciel applicatif, prévues pour être utilisées plusieurs fois (variables qui résultent des équations et calculs mais ne nécessitent pas d'exploitation extérieure à un bloc) :
  - lorsqu'elles sont indexées automatiquement, vérifier qu'une zone mémoire est réservée pour ces variables internes,
  - lorsqu'elles ne sont pas indexées automatiquement, vérifier qu'elles sont différentes pour chacun des blocs.

Entrées/sorties

- Vérifier l'utilisation de toutes les entrées-sorties définies dans les étapes de conception précédentes.
- Vérifier la règle de base concernant la gestion des ordres d'arrêt et de marche qui doivent être associés respectivement à des contacts d'entrée O et F.
- Vérifier l'utilisation de toutes les entrées impulsionnelles définies dans les étapes de conception précédentes.

Configuration logicielle de l'APIdS

- Comparer la déclaration des entrées/sorties impulsionnelles par rapport au tableau regroupant l'ensemble des entrée-sorties de l'automate.

5.5. Vérification du codage

5.5.1. Documents à fournir par le concepteur

Fournir les documents suivants :

- le listing du programme principal,
- la table des variables (avec mnémonique et commentaires),
- le tableau des références croisées,
- la structure du programme principal (arborescence des blocs utilisés)

pour l'édition de ces documents, privilégier l'utilisation des outils fournis par l'atelier de programmation lorsqu'ils existent.

- une liste :
    - des calculs qui sont effectués, avec localisation dans le programme principal,
    - des sauts de programme qui sont effectués, avec localisation dans le programme principal,
    - des variables mémorisées (SET-RESET) qui sont utilisées, avec localisation dans le programme principal.

5.5.2. Tâche du vérificateur

Sur la base des documents fournis, il doit être possible de vérifier certaines règles de codage tel que :

- l'impossibilité d'avoir une division par zéro (comparaison avant opération) ou que les capacités de calcul ne risquent pas d'être débordées,
- l'utilisation limitée des sauts de programme, lorsqu'un saut est néanmoins utilisé, l'adresse du saut doit se trouver dans le même bloc, sinon, les variables internes des programmes qui ne sont pas scrutées ne sont pas mises à jour ; vérifier leur mise à jour par programmation,
- l'absence de variables mémorisées (SET-RESET). Lorsqu'elles sont malgré tout nécessaires :
    - s'assurer que les conditions de mise à 1 et de mise à 0 sont situées dans le même bloc (module), placées de façon consécutive,
    - s'assurer qu'il n'existe qu'une seule équation de SET et une seule équation de RESET pour une même variable,
    - s'assurer que l'ordre d'écriture des équations n'influe pas sur leur résultat,
    - s'assurer que les commandes de SET et de RESET sont parfaitement complémentaires ou qu'une commande simultanée de SET et RESET est impossible.
- s'assurer de la présence de mnémoniques et de commentaires pour toutes les variables.

Remarque : chaque variable utilisée doit être associée au minimum à un mnémonique et à un descriptif sommaire. L'utilisation des mnémoniques diminue le risque d'erreur dans l'utilisation des variables (mentalement, la variable et son mnémonique sont associés à un élément concret de la

machine) et permet une compréhension plus rapide des équations. Pour être efficace, cette codification doit répondre à des règles faciles à mettre en œuvre qui conduisent à un mnémonique le plus explicite possible et à des règles de cohérence par rapport à la documentation de la machine.

En plus des mnémoniques attachés aux variables du programme (bit, octet, mot) il est nécessaire d'associer à chaque équation ou segment du programme un commentaire sur la fonction réalisée. Ce commentaire sera plus ou moins détaillé en fonction de la complexité de l'équation.

- Vérifier que les sorties de commande d'actionneur sont commandées dans une équation unique et scrutées cycliquement par l'APIdS.
- Vérifier qu'il n'existe aucune instruction d'écriture sur la mémoire image des entrées de l'automate (bit, octet, mot), par exemple avec la table des références croisées.

Notes :

- En fonction de la taille et de la qualité de développement du programme principal, les règles de codage proposées pourront être vérifiées soit de manière exhaustive, soit de façon ponctuelle et aléatoire.
- Certaines parties du code peuvent également être explorées à l'initiative du vérificateur, notamment lors des étapes de vérification des tests unitaires ou d'intégration.

5.6. Vérification des étapes de tests

Les vérifications à effectuer sont identiques pour les trois étapes de tests du logiciel :

- tests unitaires,
- tests d'intégration,
- tests de validation.

Il n'y a que le contenu des tests, défini dans les cahiers de tests lors des étapes de conception, qui diffère.

5.6.1. Documents à fournir par le concepteur

Pour chaque étape de **test** du cycle de développement en V, figure 3 (branche montante), le concepteur doit fournir les résultats des tests sous forme de «fiches de tests» renseignées.

Si lors des tests, des problèmes sont apparus, le concepteur doit fournir les « fiches de tests » renseignées version N identifiant les problèmes. Après résolutions des problèmes, il doit fournir des éventuelles nouvelles versions des documents amont modifiés suite à des actions correctives ainsi que « fiches de tests » renseignées version N+1.

5.6.2. Tâche du vérificateur

Vérifier que toutes les fiches de tests définies dans les plans de tests ont été renseignées.

Si lors des tests, le concepteur a rencontré des problèmes, reprendre les étapes de conception concernées par les modification à apporter (branche descendante du cycle de développement en V). Vérifier les fiches de test version N+1 renseignées.

Le vérificateur peut, en cas de besoin, demander de "rejouer" certains tests pour confirmer leur réalisation.

6. Conclusion

La méthode de validation par tierce partie proposée repose sur une adéquation de sa mise en œuvre avec le processus de développement du concepteur. Il a été postulé que tout processus de développement peut être schématisé par un cycle en V. Il est nécessaire de rappeler que c'est un cycle théorique et donc, en ce sens, son application comporte quelques adaptations mais, globalement, tout cycle de développement peut être représenté schématiquement de la sorte.

L'objectif de la méthode n'est pas d'imposer un processus de développement particulier mais de pouvoir s'y intégrer en tant que tierce partie.

La méthode de validation par tierce partie propose d'analyser chaque étape du cycle de développement d'un circuit de commande, sans procéder à un examen précis des équations logiques (lignes de code) du logiciel applicatif. Elle s'appuie sur des documents, des informations et des résultats de tests fournis par le concepteur. Pour toutes ces raisons, la méthode de validation par tierce partie nécessite une confiance justifiée entre le vérificateur et le concepteur.

Il est primordial de préciser que : la validation par tierce partie ne doit en aucun cas se substituer à la validation par le concepteur!

Nous rappelons enfin que cette méthode de validation s'adresse à une tierce partie compétente techniquement pour le type de machine concernée et maîtrisant les techniques d'intégration d'un APIdS , tel qu'un organisme de contrôle ayant en charge la validation d'équipements de travail dans le cadre d'une évaluation de conformité.Cette évaluation peut avoir lieu pour une machine neuve dans le cadre de la certification par organisme notifié ou de l'auto-certification mais aussi dans le cadre de la rénovation d'une machine en service.

Bibliographie

[1] CT5 – Ministère de l'Emploi et de la Solidarité. *Note relative à l'acceptation de certains automates programmables pour gérer des fonctions de sécurité sur machines,* mai 1998, e23/APIDS/26 mai 1998, 5 p.

[2] P. Lhoste, *Contribution au génie automatique : concepts, modèles, méthodes et outils*. Habilitation à diriger des recherches, Université de Nancy I, 147 p, 1994.

[3] Gaudel et al. *Précis de logiciel*, Enseignement de l'informatique, Ed. Masson, ISBN: 2-225-85189-1, 142 p., 1996.

[4] B.W. Boehm, *Verifying and validating software requirements and design specifications*, IEEE Software, 1984.

[5] E03.65.036.G, *Règles de conception et de réalisation des logiciels d'automatismes*, document CNOMO PSA Peugeot- Citroën, 22 p., 1999.

[6] ND 2172, C. Neugnot, M. Kneppert, *Logiciels applicatifs relatifs à la sécurité. Étude des problèmes liés à leur exploitation*, INRS, 16 p., 2002.

[7] NS 243, P. Lamy, *Développement du logiciel applicatif : hiérarchisation des prescriptions de la norme CDV CEI 62061 et guide explicatif à destination des utilisateurs d'automates programmables*, INRS, 47 p., 2004.

[8] NF ISO/CEI 12207, *Traitement de l'information – Ingénierie du logiciel. Processus du cycle de vie du logiciel*, AFNOR, 66 p., 1995.

[9] Directive 98/37/CE du 22 juin 1998 concernant le rapprochement des législations des états membres relatives aux machines, J.O.C.E. n° L207 du 23 juillet 1998, CE, Bruxelles, 46 p.

[10] NF EN 692, *Presses mécaniques – Sécurité*, 72 p., 1998.

[11] ED 782 , *Presses mécaniques pour le travail à froid des métaux. Amélioration de la sécurité sur les presses en service dans le cadre de leur rénovation. Spécifications techniques à l'usage des préventeurs et des rénovateurs*, INRS, 28 p., 2004

[12] ED 783 , *Presses pour le travail à froid des métaux. Amélioration de la sécurité sur les presses en service dans le cadre de leur rénovation. Guide à l'usage des utilisateurs et des préventeurs*, INRS, 34 p., 2003

[13] FDIS CEI 62061 - *Sécurité des machines - Sécurité fonctionnelle des systèmes de contrôle électriques/électroniques/électroniques programmables*, 90 p., 2003.

[14] prEN ISO 13849-1, *Sécurité des machines - Parties des systèmes de commande relatives à la sécurité - Partie 1 : principes généraux de conception*, 2004.

[15] NF EN 954-1, *Sécurité des machines - Parties des systèmes de commande relatives à la sécurité - Partie 1 : principes généraux de conception*, AFNOR, 39 p., 1997.

[16] NF EN 60204-1, *Sécurité des machines - Équipement électrique des machines - Partie 1 : prescriptions générales*, AFNOR, 104 p., 1998.

[17] ED 905, *Câblage des entrées et sorties d'automates programmables dédiés à la sécurité*, INRS, 64 p., 2003.

[18] STSARCES Project (N° SMT4-CT97-2191), *Guide for the construction of software tests*, WP 1.2/2, Final report, Nancy, INRS, 1999.

[19] ND 2140, P. Charpentier, *Comment construire les tests d'un logiciel*, INRS, 14 p., 2000.

[20] IEC/TR 61131-8, *Automates programmables - Partie 8: Lignes directrices pour l'application et la mise en oeuvre des langages de programmation,* rapport technique IEC, version anglaise, 110p., 2003.

[21] J.C. Blaise, J.P. Bello, J. Baudoin, *Validation of the control system of a press using a programmable electronic system*, actes du congrès Sécurité des Systèmes Industriels Automatisés (SIAS 2003), Nancy, pp. 5.91-5.96, 13-15 octobre 2003.

[22] NST 224, D. Dei-svaldi, M. Kneppert *Gestion des fonctions de sécurité par automate programmable dédié à la sécurité (APIdS)*, INRS, 24 p., 2002.

Simulation de partie opérative

Les objectifs de la simulation de partie opérative sont détaillés dans [6]. Nous n'en présentons ici que les principes schématisés figure 7.

Nous avons évoqués lors de différents paragraphes de tests la possibilité du concepteur de recourir à un tel outil. Dans le cadre de l'étude ayant mené à l'élaboration du présent document, des développements de scénarios de tests et de modélisation de partie opérative ont montré une faisabilité, il faudrait toutefois s'assurer des conditions de réalisation. De toute façon, la simulation de partie opérative ne sera pas « l'outil miracle » qui valide à la place du vérificateur. Il conviendrait de définir avec les vérificateurs du cadre de l'utilisation potentielle de cet outil par le concepteur.

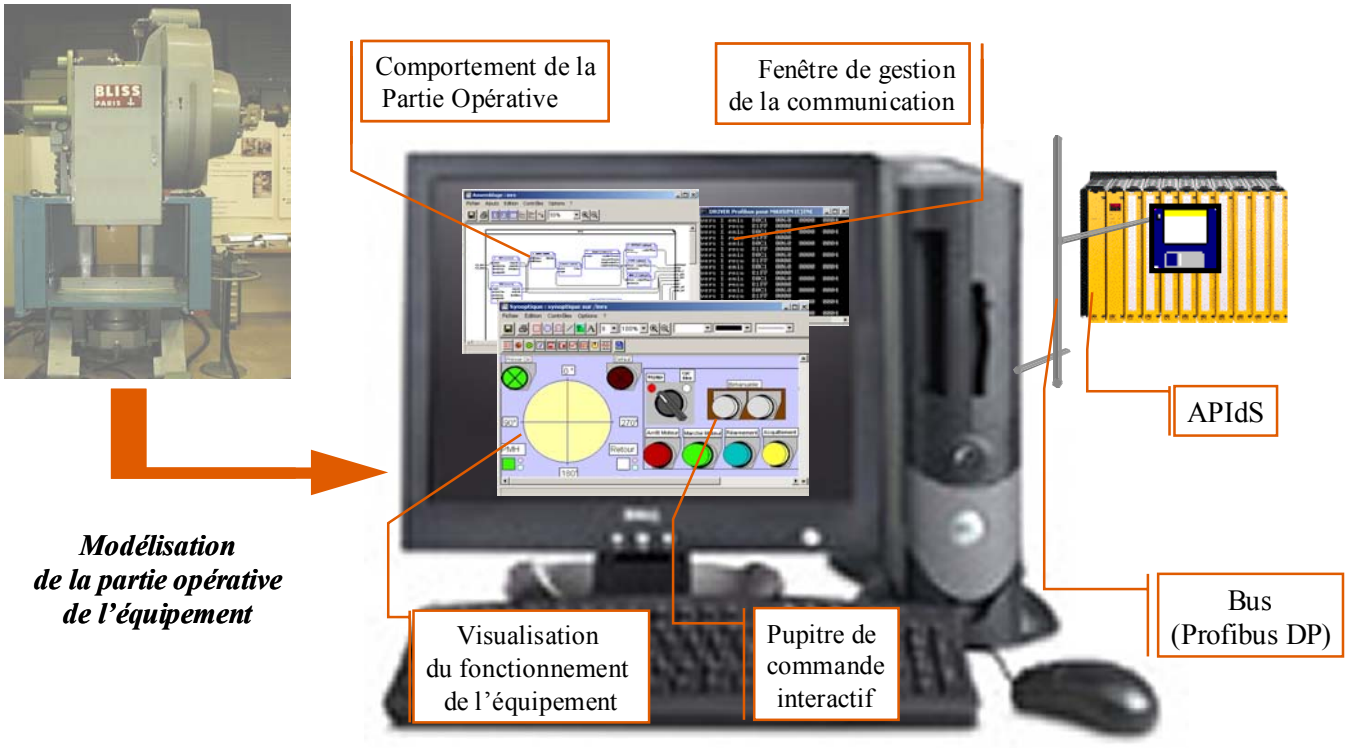

Figure 7 : Principe de la simulation de partie opérative d'après [21]

Description d'un APIdS

Un APIdS est constitué d'une interface d'entrée, d'une unité de traitement - qui va lire les données d'entrées, effectuer les traitements requis et activer les sorties - et d'une interface de sortie [22]. L'intégration d'un APIdS dans une machine requiert :

- le développement d'un logiciel applicatif au sein de l'unité de traitement, qui commande l'acquisition des entrées, leur traitement et l'activation des sorties,
- la connexion de l'APIdS via ses interfaces d'entrée et de sortie, avec les autres éléments de la machine.

Un APIdS est constitué des mêmes modules qu'un API standard, mais son architecture est telle qu'il est potentiellement utilisable pour traiter des fonctions de sécurité.

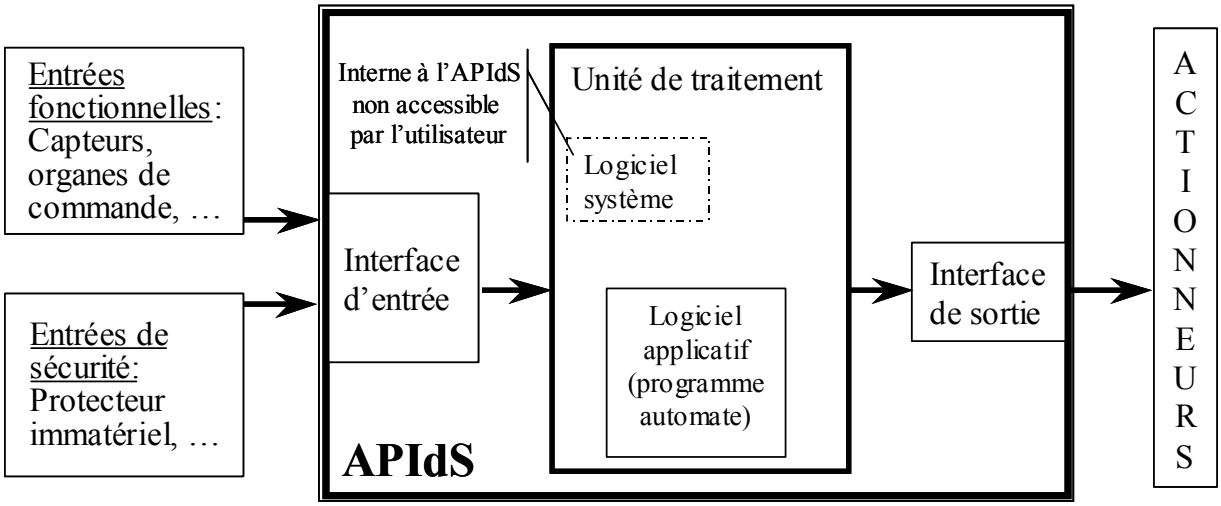

Figure 8 : Représentation schématique d'un APIdS

COLLECTION DES NOTES SCIENTIFIQUES & TECHNIQUES

Les notes scientifiques et techniques présentent des travaux de synthèse élaborés par des experts en hygiène et sécurité du travail, en particulier de l'INRS : résultats d'études, comptes rendus de séminaires et de colloques, etc.

Sur la base de ces documents peuvent être réalisées des publications plus concises dans des revues scientifiques et/ou de prévention.

Institut national de recherche et de sécurité
pour la prévention des accidents du travail et des maladies professionnelles
Siège social : 30, rue Olivier-Noyer 75680 Paris cedex 14 • Tél. 01 40 44 30 00
Fax 01 40 44 30 99 • Internet : www.inrs.fr • e-mail : info@inrs.fr
Centre de Lorraine : avenue de Bourgogne BP 27 54501 Vandœuvre cedex
Tél. 03 83 50 20 00 • Fax 03 83 50 20 97