Vers des programmes d'automates vraiment sûrs ?

A l'heure de l'informatisation galopante des solutions d'automatisation, s'assurer de leur bon fonctionnement passe de plus en plus par la vérification et la validation de programmes. Des alternatives aux méthodes manuelles arrivent, issues du monde de l'informatique pure.

Les automates sont formidables. Ils peuvent tout faire, ou presque ; il suffit pour cela de les connecter aux bonnes entrées/sorties et, surtout, de les alimenter avec le bon programme. Autrement dit, de passer des heures à écrire des centaines de lignes de code, sans garantie que l'automate remplira parfaitement sa tâche en toutes circonstances, ni même que l'on n'y laisse traîner aucune coquille fatale… Et dans ce domaine, les erreurs coûtent cher. *« Dans le forage pétrolier, par exemple, les automates sont souvent responsables de retards dans la recette et la mise au point sur site, qui créent un surcoût de plus de 20 millions de dollars par projet »*, témoignait Thierry Coq, principal consultant system and software reliability chez le spécialiste du management des risques DNV, lors d'une journée du Club Automation intitulée : *« Conception sûre des applications de contrôle-commande »*.

Prendre modèle sur l'informatique classique

La question posée par les spécialistes des automatismes est simple : *« Y a-t-il des moyens de s'assurer que ce que l'on fait est bon ? »*. Hélas, mis à part les recommandations des fournisseurs de matériel, il n'existe pas de règles de programmation des automates industriels. En revanche, certains développent des méthodes leur permettant de vérifier et de valider leurs programmes tout au long de leur conception. Le cabinet DNV, par exemple, a cherché à appliquer le modèle qualité Sqale développé pour l'informatique classique, à des programmes d'automates.

Le système est simple. Il s'agit de soumettre le code source à des sondes mesurant certaines caractéristiques, avant de les analyser et de synthétiser les informations dans des tableaux de bord adaptés à chacun des intervenants dans le projet : client, chef de projet, informaticiens et automaticiens. Parmi les indicateurs clés de cette méthode, l'indice de remédiation mesure le coût nécessaire pour amener le programme sondé au niveau d'un programme sans défaut.

Parce que Sqale est indépendant du langage de programmation, la solution mise au point en s'appuyant sur les outils d'amélioration de programmes automates d'Itris Automation Sqare, convient aux 5 langages de l'IEC (International Electrotechnical Commission). Et chacun y trouve son compte : *« le chef de projet peut suivre la qualité et l'avancement du travail, et faire du benchmarking. Les services méthodes, ont un outil qui leur permet de vérifier si leurs règles sont en adéquation avec la réalité et les standards, et peuvent partager les*

méthodes de façon objective. Pour le client, cela entraîne une simplification de prise de décision car elle se fait sur des données objectives », note Denis Chalon, directeur technique d'Itris Automation Sqare.

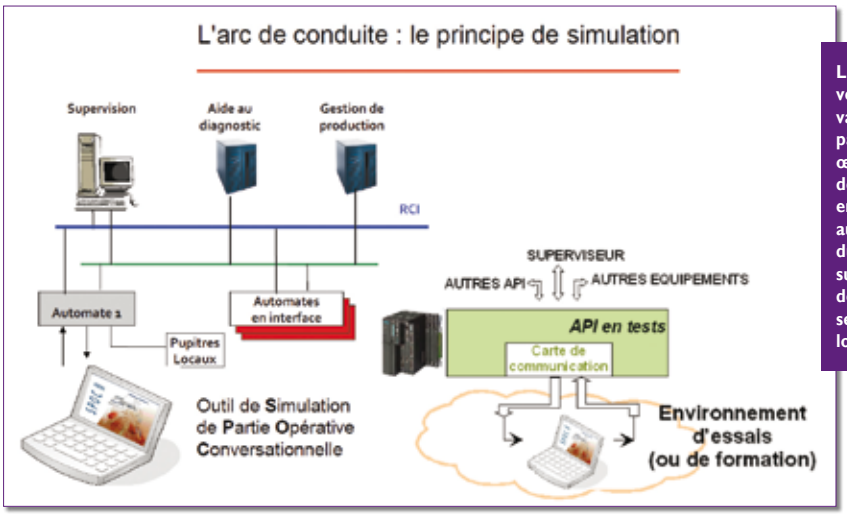
Manifestement, les études menées par DNV et Itris, dont un test lors d'un projet dans l'industrie pétrolière, ont montré qu'il est pertinent d'utiliser les caractéristiques et les seuils de détection employés en informatique classique, quel que soit l'automate concerné. Il reste cependant une difficulté à lever : trouve le moyen de contrôler les langages graphiques.

La méthode développée par DNV et Itris Automation Sqare apporte aux différents intervenants du projet des indicateurs sur les grandes caractéristiques du code étudié.

Appliquer une méthode

Le groupe d'Ingénierie et de conseil en innovation Assystem a, lui aussi, mis au point une méthodologie pour valider et qualifier les systèmes de contrôle commande, baptisée Arc de Conduite. *« Auparavant, chaque développeur et équipementier faisait ses tests unitaires et fonctionnels de son côté. A la connexion de ces systèmes, les erreurs apparaissent : problèmes d'interface, erreurs d'adressage, incohérences logiques et retards de planning. Cela faisait exploser les temps d'essais et de mise en service sur site*, justifie Frédéric Legoubey, Responsable essais Informatique Industrielle chez Assystem. *Pour éviter ces problèmes, nous utilisons une méthode structurée mettant en œuvre une plateforme d'intégration permettant de récupérer, au fur et à mesure de la programmation des différents composants, les constituants (API 1, API2…, supervision etc.) et, pour s'assurer d'être bien calibré quel que soit le fournisseur, de venir avec les mêmes critères de test et de qualification pour tout le monde »*. Selon le responsable essais, cette méthode permet d'anticiper les tests fonctionnels chez les fournisseurs, de réceptionner partiellement des programmes en plateforme, de confirmer la réalisation de l'intégralité de la fourniture, de qualifier les interfaces entre les différents composants et de maximiser le nombre de tests sur plateforme plutôt que sur site.

La méthode d'Assystem passe par une phase de définition des stratégies d'essais (en indiquant l'organisation des tests et les outils qui seront utilisés), puis de rédaction des cahiers de tests et d'un plan de validation associé au cahier des charges de l'installation. Après l'exécution intervient une phase de compte-rendu et d'analyse de plus en plus globale et approchant de plus en plus l'exploitation. La validation passe naturellement par la simulation des parties opératives. Il convient alors de définir ce qu'il faut simuler et quelles conditions de conduite il faudra recréer en fonction des exigences du projet et du secteur. *« Nous essayons actuellement d'automatiser tout cela. Il faut que les fiches de test se génèrent et s'exécutent automatiquement »*, déclare Frédéric Legoubey. Avec un leitmotiv : *« Dans le simulateur, nous essayons de garder un maximum des composants de l'axe à valider, par exemple en utilisant les fonctions de virtualisation des automates »*, note le responsable essais.



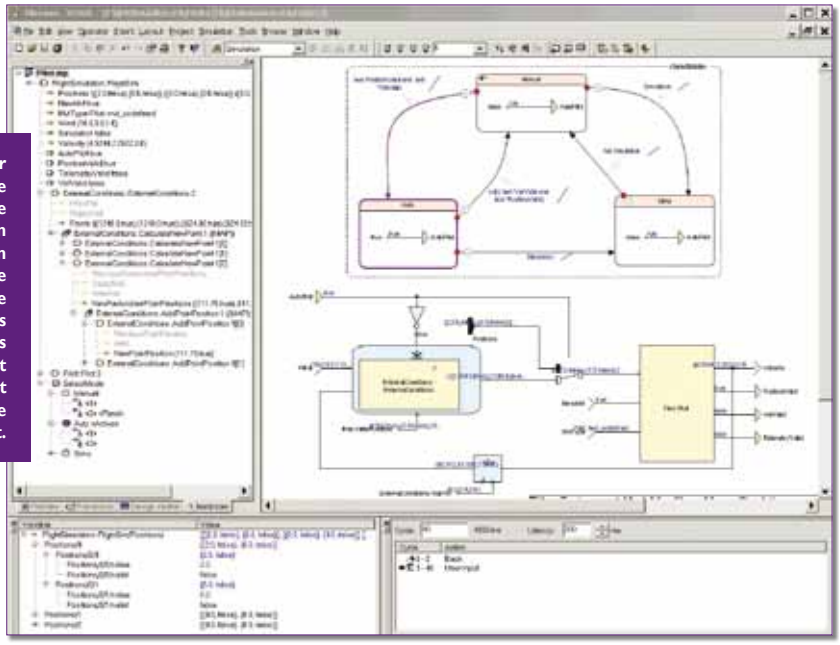
La méthode de vérification et de validation utilisée par Assystem met en œuvre une simulation des parties opératives en lien avec les autres constituants du système (API, supervision, gestion de production), au sein d'une plateforme logicielle commune.

Le résultat de la méthode ? Son application aurait permis de passer de 6 mois sans simulation de partie opérative à seulement 2 mois (trois semaines de modélisation, 2 semaines d'essais purs en plateforme et 1 mois de mise en service sur le site) pour la mise en place d'une installation chez un

de ses clients de l'industrie pharmaceutique. Et outre ce temps de développement divisé par trois, le démarrage sur site a été réalisé avec zéro bug API…

Vers la génération automatique de code

Pourquoi ne pas partir d'un langage « générique » et réaliser la génération de code en automatique ? C'est ce que propose Esterel technologies en appliquant aux automatismes industriels la méthode Scade, mise au point initialement pour les systèmes embarqués dans le secteur ferroviaire. La logique est simple : *« Si on est capable de générer automatiquement un code à partir d'un modèle vérifié formellement, cela permet de détecter les erreurs le plus tôt possible et ajoute de l'efficacité au logiciel développé »*, déclare Luc Coyette, directeur technique chez Esterel Technologies.



Développé pour l'embarqué dans le ferroviaire, Scade est à la fois un environnement et un langage graphique qui permet de construire des modèles complexes des applications et de les valider avant de générer du code automatiquement.

« Scade est à la fois un langage et un environnement, explique le directeur technique. C'est un langage formel et modulaire, qui permet de définir de manière non ambiguë un modèle d'application sur lequel on applique des outils de vérification pour s'assurer que le modèle est correct. Quand on sait que le modèle est correct, il ne reste plus qu'à créer le code avec un générateur ». Autre avantage, le langage Scade est graphique et inclut les flux de données et les machines à états. Il peut donc modéliser des applications complexes tout en étant compris de la même manière par tous les acteurs d'un projet. Aucune ambiguïté ni effet de bord possible. Et son caractère synchrone (qui calcule les états présents selon les états précédents) associé à la vérification systématique de l'existence d'une valeur initiale évite toutes les difficultés liées à des terminaisons de boucle et les cycles de causalité, parfois difficilement détectables avec d'autres méthodes de programmation.

Ce langage de base est complété d'outils de conception, de vérification (sémantique et syntaxique), de simulation et de qualification (dont un outil de mesure de couverture) et, bien sûr, de génération automatique de code C ou ada, *« avec des adaptateurs pour les PLC »*, annonce le directeur technique d'Esterel Technologies. A noter, le code généré est indépendant de la cible et du système d'exploitation et répond au standard C Ansi (que personne n'utilise, d'ailleurs…).

Scade est certifié pour l'embarqué en aéronautique et dans le ferroviaire. Pour les automates industriels, *« pour l'instant, nous générons du code C et une « glue » faite pour fonctionner sur un certain type d'automates. Cela ne peut donc fonctionner que pour les automates capables d'avaler du C »*, déclare Luc Coyette. L'outil, développé avec KW Software, est disponible depuis décembre. Attention, il est probable que cette solution ne soit économiquement viable que pour des applications d'une certaine importance…

Encore des efforts à faire

Les méthodes de vérification et de validation employées dans l'informatique semblent applicables au monde des automates. D'ailleurs, *« si les analyses statiques de code sont intégrées aux outils utilisés par les automaticiens, ces informations vont lui apporter beaucoup de valeur »*, assure Thierry Coq, de DNV. Cependant, des difficultés liées à l'invasion du logiciel dans les installations demeurent. A commencer par la pérennité des systèmes. Car si un logiciel ne s'use pas comme la mécanique, sa durée de vie n'est pas franchement garantie. Ainsi est-il plus prudent de conserver un mélange de matériel et de logiciel dans les systèmes, afin de s'assurer de pouvoir les utiliser sur le long terme.

Autre difficulté : le turnover important dans le milieu de la programmation des automates. *« Les jeunes qui programmaient les automates il y a cinq ans ne sont plus là. L'expertise en électromécanique commence à manquer et l'expérience informatique est réelle mais très jeune »*, regrette Thierry Coq. Les outils de développement et les automates devront donc à l'avenir s'adapter pour s'adresser à des gens qui ne sont pas vraiment des automaticiens et pas vraiment expérimentés…

Et les logiciels libres ? Une piste intéressante, d'autant que selon le consultant de DNV, *« il semblerait qu'ils respectent plus les standards et les normes que les systèmes des fournisseurs »*. A vérifier dans une prochaine journée du Club automation ? ▮