**Résumé article MSR19\_paper\_12-final :**

Il existe déjà de la génération de code a partir de certains modèles a base de grammaire. Mais pas a partir d’un modèle de plus haut niveaux (plus abstrait) tel que des modèles UML. D’une part, les générateurs de code des éditeurs UML produisent typiquement des squelettes où le gros du développement reste à faire pour laisser le choix a l’utilisateur de concevoir le logiciel comme il veut. Donc ils préconisent une approche par transformation de modèles dans laquelle les transformations restent simples. La complexité se trouve donc alors dans le processus de transformation qui se veut adaptable et configurable.

Lors de l’étude de cas concernant à une implémentation d’une porte de garage, il ont constater que pour un même logiciel, il pouvait avoir une infinité de choix de conception tel qu’utiliser un pattern state au lieu des types énumérés. Ce qui confirme que la complexités d’une transformation de modèle se trouve dans le processus de transformation, de choix de transformation. Donc tout au long de la transformations de modèles, le concepteur devra interagir avec le système pour énoncer ces choix pour les différents composant du système.

Ils en ont conclue aussi qu’il faut enrichir les modèles, formaliser les processus de raffinement, rendre modulaire et personnaliser le développement pour s’appuyer sur des outils de transformation. Tout cela pour facilité transformation de modèles suivant les choix du concepteur.

**Résumé article « Refining Automation System Control with MDE »**

Le but de cet article est l’assistance pour la transition par étapes du modèle vers le code dans le but de réduire le temps entre le modèle d’analyse et l’implémentation.

3 alternatives pour développer une application à partir d’un modèle logique :

- développement manuel :

- processus de transformation du modèle :

- génération automatique de code :