Data flow : Un flux de données est “une séquence infinie d’éléments générés de façon continue à un rythme rapide” selon la définition de Baptiste Csernel, Fabrice Clérot et Georges Hébrail. C'est donc un ensemble d'informations circulant d'un point à un autre dans un but précis.

L’**inférence** est un mouvement de la pensée allant des principes à la conclusion. C'est une opération qui permet de passer d'une ou plusieurs assertions, des énoncés ou propositions affirmés comme vrais, appelés prémisses, à une nouvelle assertion qui en est la conclusion.

L'**inférence de types** est un mécanisme qui permet à un compilateur ou un interpréteur de rechercher automatiquement les types associés à des expressions, sans qu'ils soient indiqués explicitement dans le code source.

Horn clause is clause (a disjunction of literals) with at most one positive, i.e. unnegated, literal. A clause with at most one positive (unnegated) literal is called a Horn Clause.

Liquid Types enrich existing type systems with logical predicates and let you specify and automatically verify semantic properties of your code.

For example, the Int type, can be refined with logical predicates to describe integer values different than 0: **{v:Int | v /= 0}**

A type for the divisor operator states that given two integers, you will get back an integer.

div :: Int -> Int -> Int

Using Liquid Types we can refine the above type to describe semantic properties of the operator, specifically that the divisor should never be 0.

div :: Int -> {v:Int | v /= 0} -> Int

L'interprétation abstraite est une théorie d'approximation de la sémantique de programmes informatiques fondée sur les fonctions monotones pour ensembles ordonnés, en particulier les treillis (en anglais : lattice). Elle peut être définie comme une exécution partielle d'un programme pour obtenir des informations sur sa sémantique (par exemple, sa structure de contrôle, son flot de données) sans avoir à en faire le traitement complet.