Atompm ou Xtext :

Les deux approches, graphique et textuelle, se valent au niveau du rendement et la qualité du travail fourni. La préférence de l'une ou l'autre se fera purement subjectivement selon la maîtrise de l'outil et la maîtrise des parties prenantes (clients, développeurs, experts de domaine...) de l'environnement de travail. Et en tant que programmeur, on est sensé maîtriser les deux outils (atompm et eclipse, en utilisant Xtext) parce que, souvent, on est en dernière position par rapport aux autres parties prenantes pour choisir l'outil de travail car, il est de notre devoir par rapport à notre domaine d'à peu près tout maîtriser.
Subjectivement et personnellement, j'ai un penchant pour Xtext. Bien que la description ne se fait et ne se voit que textuellement, tout le travail est dans la définition de la grammaire, le reste est générée donc je trouve que ça fait moins de travail. Et en entreprise, Xtext est utilisée et est plus fameux qu'Atompm donc ce serait plus judicieux de mettre plus d'efforts à le maîtriser.
Toutefois, je ne tarie pas d'éloges sur Atompm, l'outil est superbe et son orientation vers le visuel facilite l'apprentissage et la maîtrise parce que, naturellement, la mémoire visuelle humaine est la plus puissante et c'est naturellement plus facile de maîtriser une chose avec des retours imagés qu'avec du texte seulement. La documentation aussi est bien faite et faite de manière à être le plus facile à comprendre possible. Le retour visuel facilite l'apprentissage dans ma conception des choses, car, comme on dit, une image vaut mille mots. Je reproche toutefois à Atompm ses quelques bugs de plantage pendant qu'on est à fond dans un travail, dû, par exemple, à une perte de connexion au serveur ou ce genre de choses.
Sinon, à part cela, Atompm est juste parfait pour développer des DSL de manière graphique.

En définitive, les deux outils sont intéressants à maîtriser. Chacun a ses atouts à défendre et les maîtriser tous les deux ne nous fait que plus de culture.