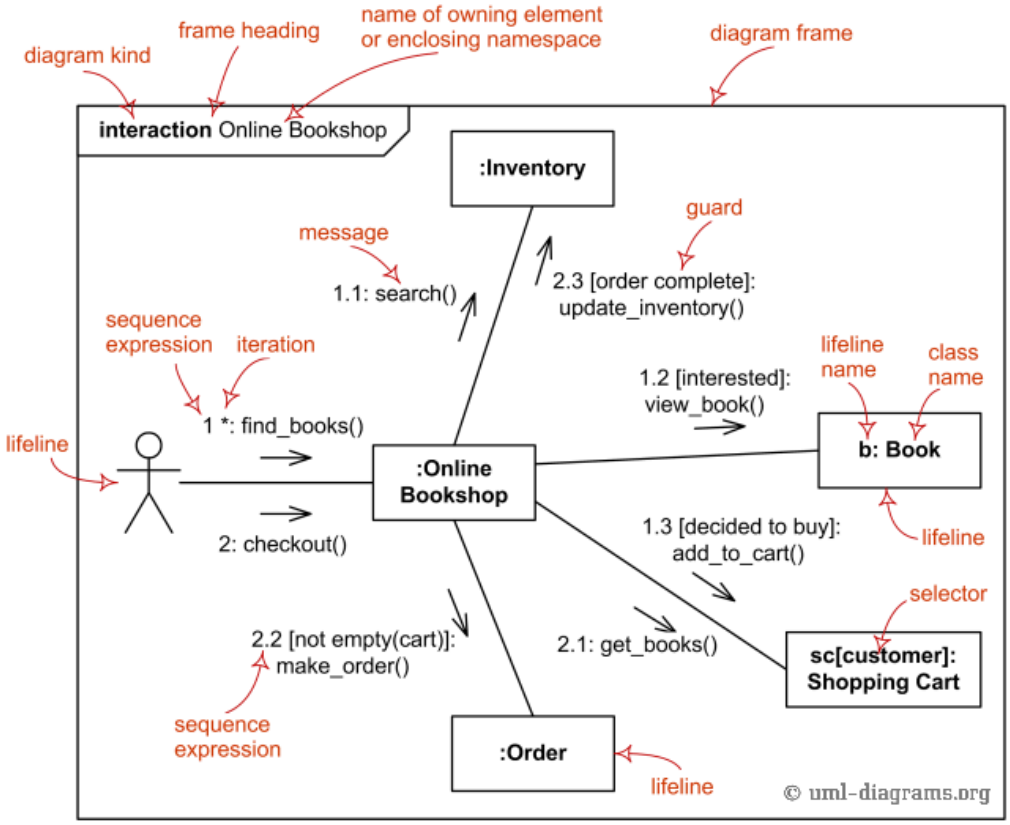
Collaboration diagram adalah diagram yang mengelompokkan pesan pada kumpulan diagram sekuen menjadi sebuah diagram. Dalam diagram tersebut terdapat method yang dijalankan antara objek yang satu dan objek lainnya. Di diagram kolaborasi ini, objek harus melakukan sinkronisasi pesan dengan serangkaian pesan-pesan lainnya.

Collaboration Diagram juga dikenal sebagai **Communication Diagram** atau **Interaction Diagram** , merupakan ilustrasi dari relasi dan interaksi antara objek software pada Unified Modeling Language (UML).

Collaboration Diagram mengelompokan message pada kumpulan diagram sekuen menjadi sebuah diagram. Dalam diagram kolaborasi yang dituliskan adalaha operasi atau metode yang dijalankan antara objek yang satu dengan objek yang lainnya secara keseluruhan. Oleh karena itu dapat diambil dari jalannya interaksi pada semua diagram sekuen. Untuk menggambarkan objek dari sebuah diagram kolaborasi, dapat menggunakan pilihan Object. Dan untuk menghubungkan antar objek yang satu dengan objek yang lain digunakan Link.

Diagram kolaborasi sangat cocok untuk penggambaran interaksi sederhana antara jumlah dari benda-benda yang relatif kecil . Tetapi ada beberapa vendor menawarkan beberapa software untuk membuat dan mengedit diagram kolaborasi sehingga dapat menggambarkan interkasi jumlah objek dan pesan yang relatif besar .

Diagram kolaborasi menunjukkan Informsi yang sama persis dengan diagram sekuensial, tetapi dalam bentuk dan tujuan yang berbeda. Sebagaimana diagram sekuensial, diagram kolaborasi digunakan untuk menampilkan aliran skenario tertentu di dalam use case. Jika diagram sekuensial disusun berdasrkan urutan waktu, diagram kolaborsi lebih berkonsentrasi pada hubungan antar obyek-obyek .



Komponen dan Simbol Collaboration Diagram

Berikut adalah komponen dan simbol dari collaboration diagram yang harus kalian ketahui.

Sombol	Deskripsi
Object	Object ialah yang melakukan interaksi pesan.
Link	Link ialah yang menghubungkan object satu dengan yang lainnya.
Arah	Arah pesan yang terjadi
Pesan	Pesan atau message ialah sesuatu yang dikirim.

Komponen collaboration diagram.

Simbol dalam *collaboration* akan lebih mudah dihafal sebab lebih sedikit, hal ini berbeda dengan di *flowchart* yang memiliki lebih banyak simbol.

- *Object* digambarkan denga simbol persegi panajang.
- *Link* digambarkan dengan simbol garis lurus.
- Stimulus atau arah pesan digambarkan dengan simbol arah panah.
- Pesan hanya berbentuk sebuah tulisan saja.

Tahap Membuat *Collaboration Diagram*

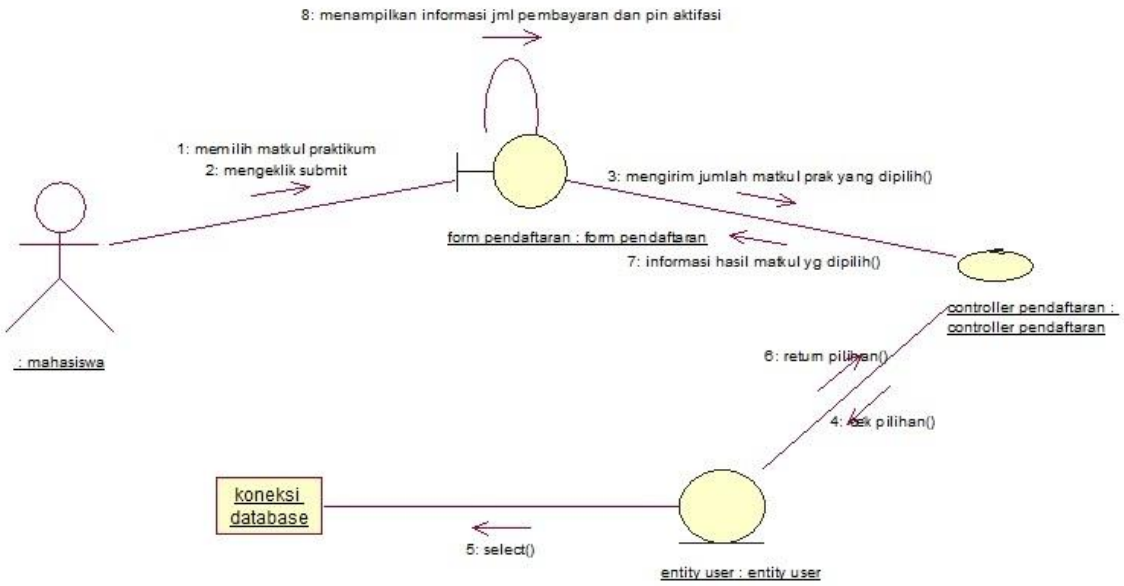
Berikut adalah beberapa tahapan yang dapat dilakukan dalam membuat collaboration diagram.

- *Use case diagram* telah dibuat secara benar dan tapat.
- Menentukan *object* (*object* adalah bagian dari *class*).
- Mencari dan menetukan *actor*.
- Tambahkan pesan dalam setiap diagram.
- Satu *use case* hanya memiliki satu diagram.
- Membuat *sequence diagram* terlebih dahulu, agar lebih mudah dalam membuat *collaboration diagram*.

Contoh Collaboration Diagram

Berikut adalah contoh *collaboration diagram* yang dapat kalian pelajari.

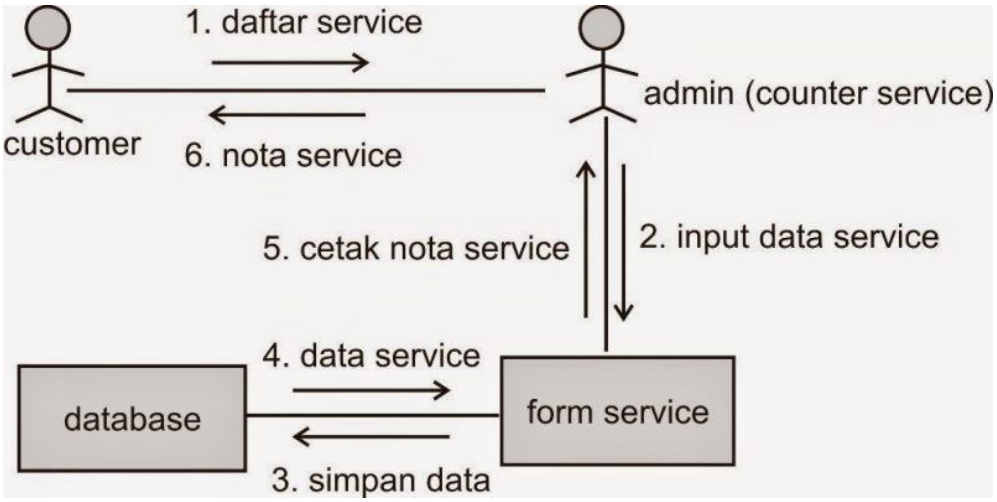
1. Contoh Collaboration Diagram Pendaftaran Praktikum



Coba kalian perhatikan peoses yang terdapat pada sequence diagram di atas, setiddaknya terdapat langkah-langkah sebagai berikut.

1. Mahasiswa memilih matakuliah praktikum.
2. Kemudian klik *submit*.
3. Mengirim jumlah matakuliah yang diambil.
4. Cek pilihan.
5. Masuk ke *database*.
6. Kembali ke pilihan.
7. Sistem menampilkan informasi mata kuliah yang dipilih.
8. Menampilkan jumlah informasi yang harus dibayar.

2. *Collaboration Diagram Service*



Berikut penjelasan langkah-langkah dari collaboration diagram di atas.

1. Melakukan daftar *service*.
2. Admin melakukan *input data service.*
3. Form secara otomatis akan menyimpan data.
4. Memanggil *data service.*
5. Cetak nota tagihan.
6. Memberikan nota *service* kepada customer.

Beberapa hal yang dapat disimpulkan dari pembahasan mengenai collaboration diagram ini ialah.

- *Collaboration diagram* digunakan untuk menggambarkan hubungan antar objek.
- *Collaboration diagra*m merupakan bentuk luas dari *sequence diagram.*
- *Collaboration* dibuat setelah *use case, sequence* dan *component diagram*.

Class diagram

Class diagram atau diagram kelas adalah salah satu jenis diagram struktur pada UML yang menggambarkan dengan jelas struktur serta deskripsi *class*, atribut, metode, dan hubungan dari setiap objek. Ia bersifat statis, dalam artian diagram kelas bukan menjelaskan apa yang terjadi jika kelas-kelasnya berhubungan, melainkan menjelaskan hubungan apa yang terjadi.

Diagram kelas ini sesuai jika diimplementasikan ke proyek yang menggunakan konsep object-oriented karena gambaran dari *class diagram* cukup mudah untuk digunakan.

Desain model dari diagram kelas ini sendiri dibagi menjadi dua bagian. Bagian pertama merupakan penjabaran dari database. Bagian kedua merupakan bagian dari modul MVC, yang memiliki *class interface*, *class control*, dan *class entity*.

Fungsi *class diagram*

Diagram kelas ini memiliki beberapa fungsi, fungsi utamanya yaitu menggambarkan struktur dari sebuah sistem. Berikut ini adalah fungsi-fungsi lainnya:

- Menunjukan struktur dari suatu sistem dengan jelas.
- Meningkatkan pemahaman tentang gambaran umum atau skema dari suatu program.
- Dapat digunakan untuk analisis bisnis dan digunakan untuk membuat model sistem dari sisi bisnis.
- Dapat memberikan gambaran mengenai sistem atau perangkat lunak serta relasi-relasi yang terkandung di dalamnya.

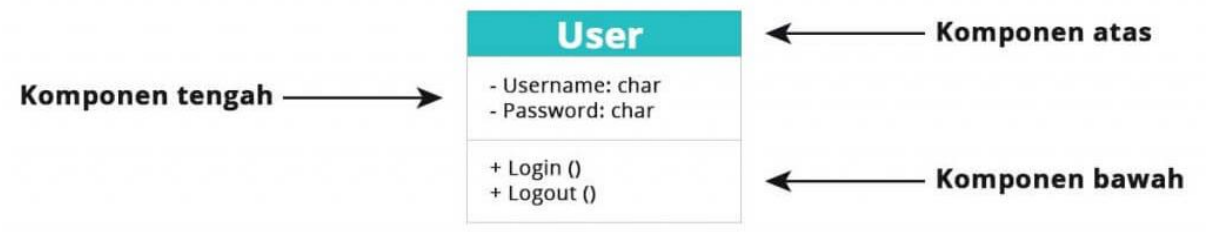
Keunggulan

Menggunakan diagram kelas memberikan banyak keunggulan bagi proses pengembangan perangkat lunak dan dalam bisnis. Berikut ini adalah keunggulan dari diagram kelas:

- Diagram kelas berfungsi untuk menjelaskan suatu model data untuk sebuah program, baik model data sederhana maupun kompleks.
- Memberikan gambaran umum tentang skema aplikasi dengan jelas dan lebih baik.
- Membantu kamu untuk menyampaikan kebutuhan dari suatu sistem.

Komponen penyusun *class diagram*

Diagram kelas memiliki tiga komponen penyusun. Berikut ini adalah komponen-komponennya:



- **Komponen atas**
  Komponen ini berisikan nama *class*. Setiap class pasti memiliki nama yang berbeda-beda, sebutan lain untuk nama ini adalah simple name (nama sederhana).

- **Komponen tengah**
  Komponen ini berisikan atribut dari *class*, komponen ini digunakan untuk menjelaskan kualitas dari suatu kelas. Atribut ini dapat menjelaskan dapat ditulis lebih detail, dengan cara memasukan tipe nilai.

- **Komponen bawah**
  Komponen ini menyertakan operasi yang ditampilkan dalam bentuk daftar. Operasi ini dapat menggambarkan bagaimana suatu *class* dapat berinteraksi dengan data.

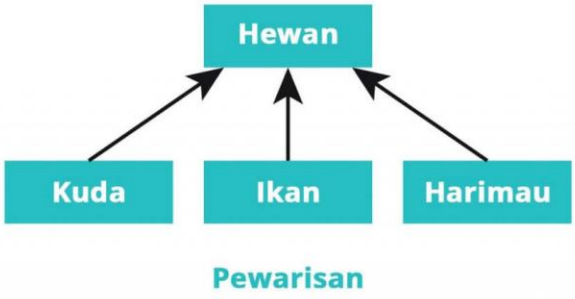
Hubungan antar kelas

Setelah kita mengetahui penjelasan tentang diagram kelas, sekarang kita akan membahas hubungan antar kelasnya. Ada tiga hubungan dalam diagram kelas. Berikut ini adalah penjelasannya:

- **Asosiasi**
  Pertama ada asosiasi. Asosiasi dapat diartikan sebagai hubungan antara dua *class* yang bersifat statis. Biasanya asosiasi menjelaskan *class* yang memiliki atribut tambahan seperti class lain.

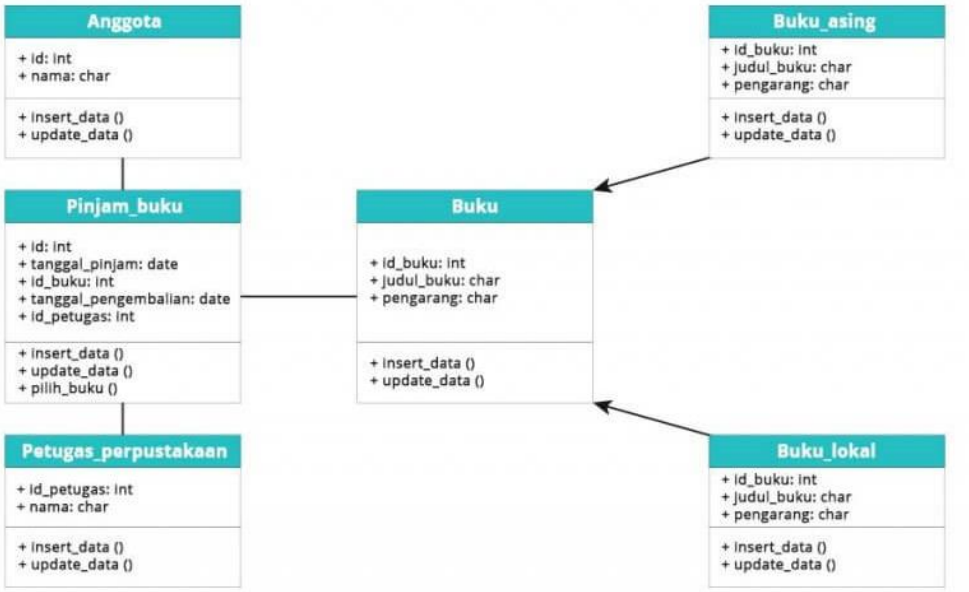
- **Agregasi**
  Agregasi adalah hubungan antara dua *class* di mana salah satu *class* merupakan bagian dari *class* lain, tetapi dua *class* ini dapat berdiri masing-masing.

- **Pewarisan**
  Pewarisan atau *inheritance* dapat disebut juga *generalization* dalam class diagram adalah suatu kemampuan untuk mewarisi seluruh atribut dan metode dari *class* asalnya (*superclass*) ke *class* lain (*subclass*).



Contoh penerapannya

Berikut ini adalah contoh dari diagram kelas sistem perpustakaan.



Component Diagram

Diagram ini bila dikombinasikan dengan diagram penyebaran dapat digunakan untuk menggambarkan distribusi fisik dari modul perangkat lunak melalui jaringan. Misalnya, ketika merancang sistem client-server, hal ini berguna untuk menunjukkan mana kelas atau paket kelas akan berada pada node klien dan mana yang akan berada di server.

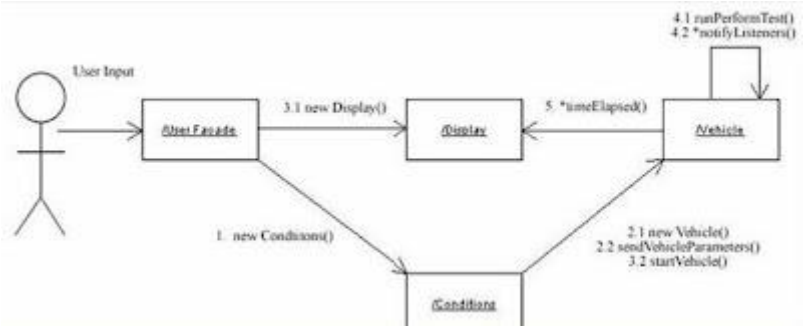
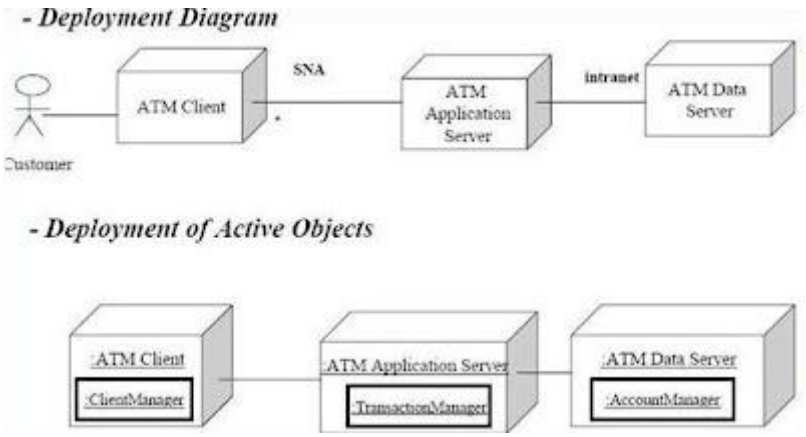


Diagram komponen juga dapat berguna dalam merancang dan mengembangkan sistem berbasis komponen. Karena berfokus pada analisis sistem berorientasi objek dan desain.

Deployment Diagram

Deployment diagram menggambarkan detail bagaimana komponen di deploy dalam infrastruktur system, dimana komponen akan terletak (pada mesin, server atau piranti keras), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi server, dan hal-hal lain yang bersifat fisikal. Hubungan antar node (misalnya TCP/IP) dan requirement dapat juga didefinisikan dalam diagram ini.



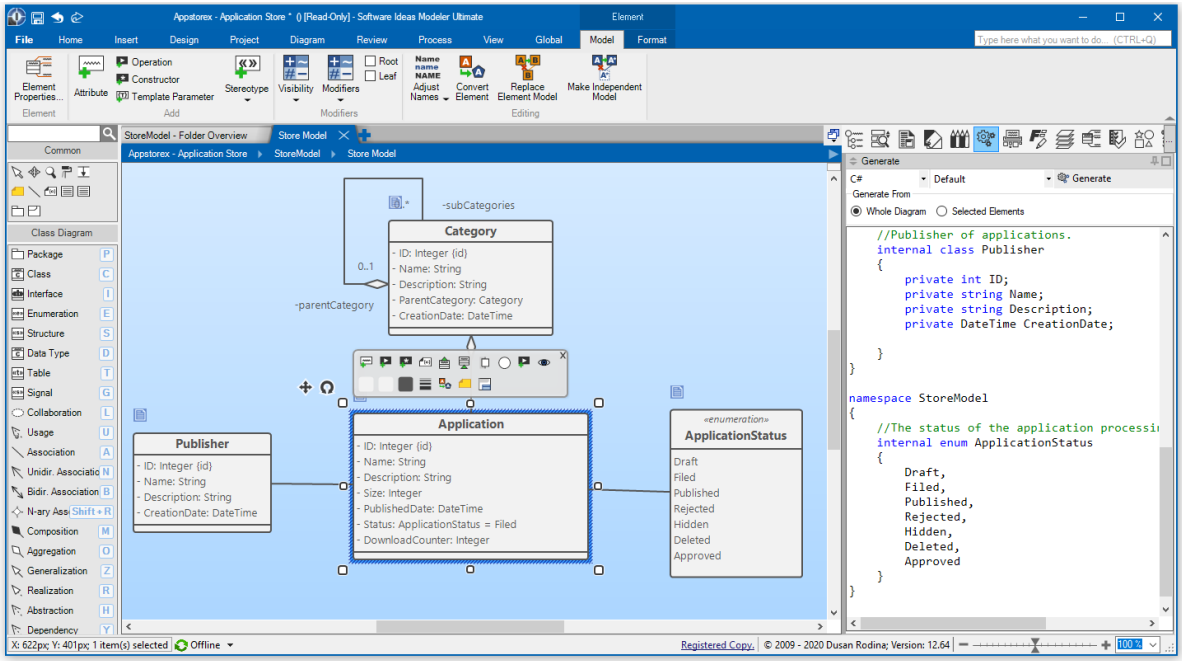
Ubah UML menjadi kode. Generator kode kami memungkinkan Anda mengonversi diagram menjadi kode sumber dalam berbagai bahasa pemrograman. Sebagian besar, Anda perlu membuat kode dan skrip untuk diagram kelas UML atau diagram hubungan entitas. Namun, Anda juga dapat membuat kode dari diagram mesin status UML atau jenis diagram lainnya dengan templat yang ditentukan secara khusus.

Pembuat Kode UML

Software Ideas Modeler memungkinkan Anda **menghasilkan kode** dalam berbagai bahasa pemrograman dan skrip. **Generator kode UML** mendukung bahasa pemrograman berikut dengan template generasi yang telah ditentukan sebelumnya:

- Skrip Aksi
- C++
- C#
- Delphi
- Jawa
- Javascript
- PHP
- Penyangga Protokol
- Python
- Rubi
- SQL DDL
- TypeScript
- VB6

- VB.NET
- XSD



Editor Diagram dengan Bilah Sisi Pembuatan Kode

UML ke Kode

Software Ideas Modeler dapat **menghasilkan kode dari diagram UML** . Ada templat yang telah ditentukan sebelumnya yang memungkinkan Anda untuk mengubah Diagram Kelas UML Anda dengan semua kelas, antarmuka, enumerasi, dan berbagai hubungan ke kode sumber di Java , C# , Python , C++, atau bahasa lain.

Anda dapat menentukan tipe logis (UML) untuk atribut dan operasi dan generator akan mengubahnya menjadi tipe yang benar dalam bahasa target. Jika ada kebutuhan untuk tipe khusus bahasa yang tidak dapat diselesaikan dari tipe logika umum, Anda dapat menggunakan kumpulan tipe untuk bahasa spesifik secara langsung dalam diagram - maka tipe yang tepat akan digunakan.

Bagaimana Cara Menghasilkan Kode untuk Diagram?

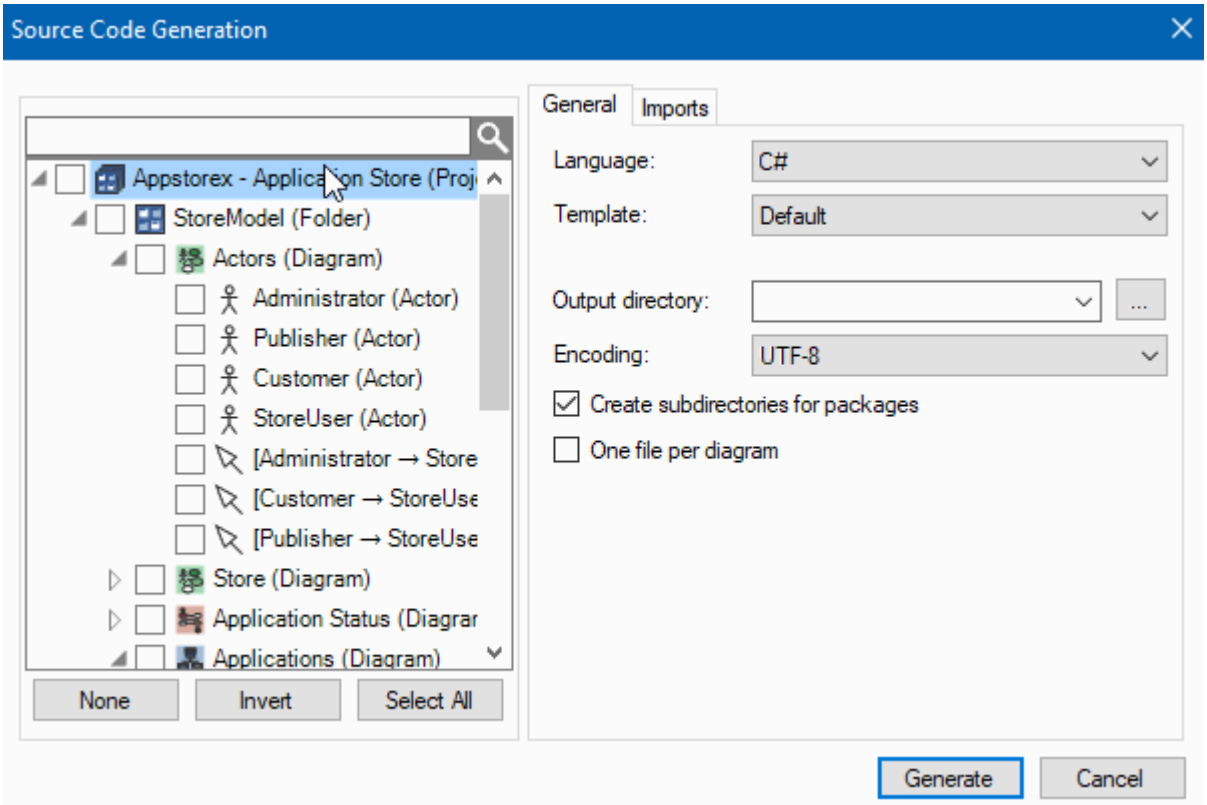
Ada dua cara untuk **menghasilkan kode sumber untuk diagram** - menggunakan bilah sisi Generasi atau dialog Generasi.

Cara tercepat adalah dengan menggunakan sidebar Generasi. (Ini dapat ditampilkan dari Ribbon/View tab/Window group/Sidebars drop-down/Generation). Bilah samping memungkinkan Anda membuat kode untuk diagram aktif atau hanya untuk elemen yang dipilih dalam editor diagram. Jadi Anda dapat dengan mudah membuat cuplikan kode dan menyalinnya dari editor kode di dalam bilah sisi.

Bagaimana Cara Menghasilkan Kode untuk Beberapa Diagram atau Seluruh Proyek?

Anda juga dapat **membuat kode sumber dari UML** menggunakan dialog Pembuatan Kode Sumber. Ini berguna ketika Anda ingin membuat kode untuk keseluruhan proyek atau lebih diagram sekaligus. Anda dapat membukanya dari pita - tab Proses / Grup Generasi / Tombol Kode Sumber.

Anda dapat memilih diagram mana yang ingin Anda buat dalam dialog **Pembuatan Kode Sumber .** Dialog mendukung pencarian di pohon proyek dan menawarkan tombol untuk pemilihan beberapa node.



Bahasa atau Diagram yang Tidak Didukung ke Transformasi Kode

Jika Anda perlu **mengubah diagram menjadi kode** untuk bahasa pemrograman tertentu atau kerangka kerja dalam bahasa yang ditentukan, Anda dapat melakukannya. Software Ideas Modeler menyediakan bahasa templat kode sumber yang memungkinkan Anda menentukan templat apa pun untuk tipe diagram apa pun yang Anda inginkan. Anda bahkan dapat menambahkan bahasa pemrograman baru.