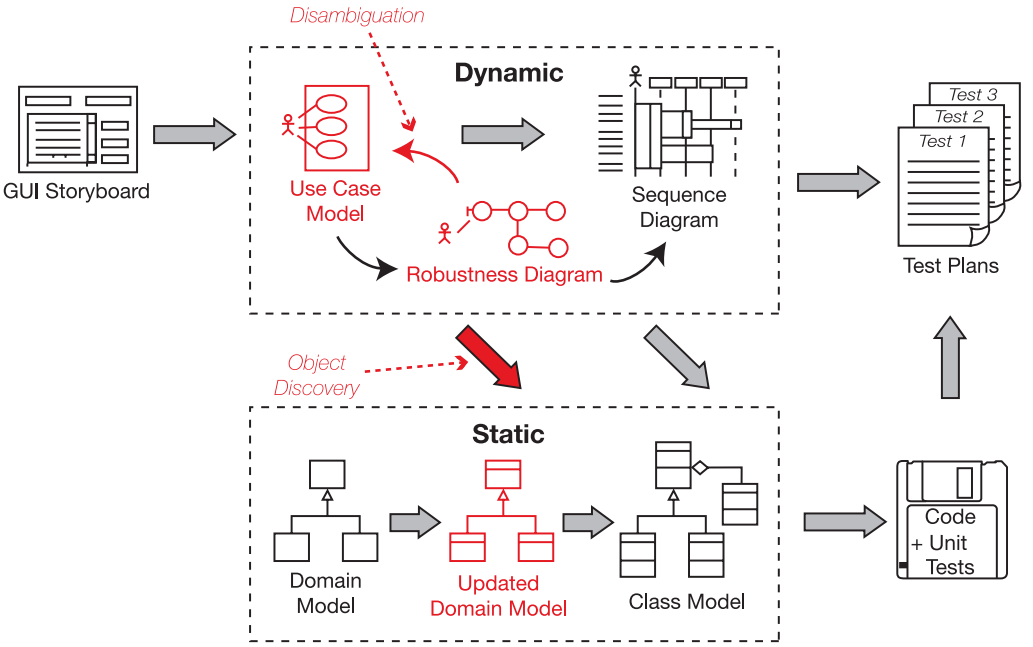
BAB 5

Analisis Kekokohan



To beralih dari kasus penggunaan ke desain terperinci (dan kemudian ke kode), Anda perlu menautkan kasus penggunaan Anda ke objek. Teknik yang kami jelaskan dalam bab ini, analisis ketahanan, membantu Anda menjembatani kesenjangan dari analisis ke desain dengan melakukan persis seperti itu. Singkatnya, ini adalah cara menganalisis teks use case Anda dan mengidentifikasi kumpulan objek tebakan pertama untuk setiap use case. Ini diklasifikasikan ke dalam objek batas, objek entitas, dan pengontrol (yang seringkali lebih mirip fungsi daripada objek).

Pemandangan 10.000 Kaki

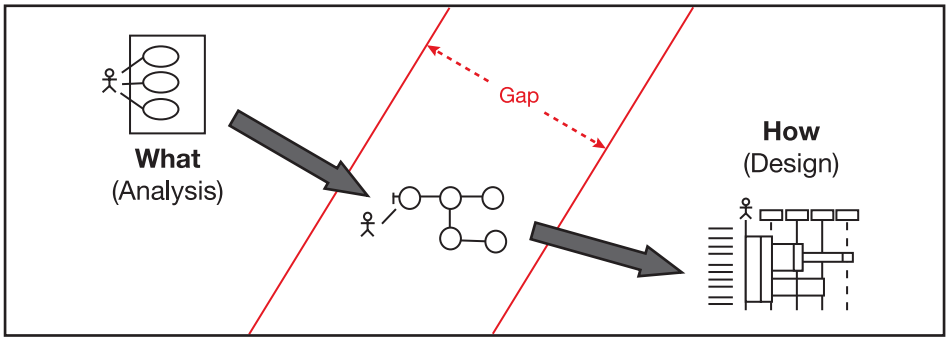
Sebuah diagram ketahanan adalah gambar objek dari sebuah use case. Diagram kekokohan dan teks use case harus sama persis, sehingga diagram kekokohan memaksa Anda untuk mengikat teks use case ke objek. Ini memungkinkan Anda untuk mendorong desain berorientasi objek maju dari kasus penggunaan, dan ini benar-benar "keajaiban" dari analisis ketahanan.

Menggambar diagram ketahanan memastikan bahwa use case adalah**ditulis dalam konteks model domain**—yaitu, semua istilah (kata benda dan frasa kata benda) yang masuk ke model domain juga harus digunakan secara langsung dalam teks kasus penggunaan Anda.

Di mana Analisis Kekokohan Sesuai dengan Proses?

Melihat Gambar 5-1, semacam analisis ketahanan terjadi di jalan tengah yang suram antara analisis dan desain. Jika Anda menganggap analisis (yaitu, kasus penggunaan) sebagai "apa" dan desain sebagai "bagaimana", maka analisis ketahanan benar-benar desain awal. Selama fase ini, Anda mulai membuat beberapa asumsi awal tentang desain Anda, dan Anda mulai berpikir tentang arsitektur teknis (juga lihat Bab 7) dan memikirkan berbagai kemungkinan strategi desain. Jadi itu analisis bagian dan desain bagian.

Ini juga merupakan teknik penting untuk menghilangkan ambiguitas dari (disambiguasi) teks kasus penggunaan Anda.



Gambar 5-1.*Menjembatani kesenjangan antara "apa" dan "bagaimana"*

Seperti Belajar Naik Sepeda

Mempelajari teknik ini memiliki sedikit kesamaan dengan belajar mengendarai sepeda. Sampai Anda “mengerti”, analisis ketangguhan bisa tampak sangat sulit, tetapi begitu Anda mendapatkannya, itu benar-benar sangat sederhana. Untuk mempercepat pemahaman Anda, kita akan membahas banyak contoh dalam bab ini. Pengalaman telah menunjukkan kepada kami bahwa Anda biasanya perlu menggambar enam atau lebih diagram ketahanan sebelum sen jatuh dan Anda tiba-tiba mendapatkannya. Ingatlah,**diagram ketahanan adalah gambar objek dari use case**.

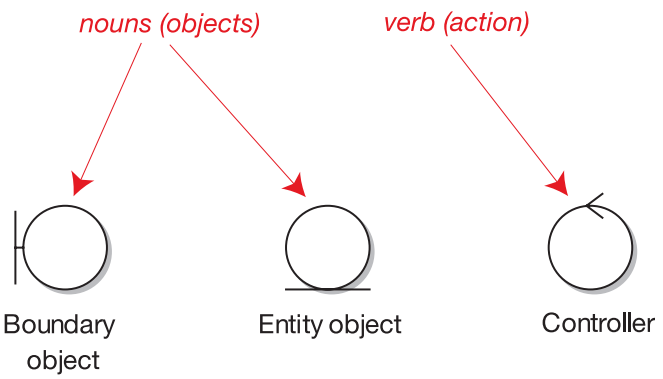
Setelah Anda memahaminya, Anda akan dapat membuat diagram ketahanan dalam waktu sekitar sepuluh menit (atau kurang) untuk setiap kasus penggunaan. Sebenarnya, seperti yang akan Anda lihat,**triknya adalah menulis use case Anda dengan benar**. Jika diagram ketahanan membutuhkan lebih dari sepuluh menit untuk menggambar, Anda dapat bertaruh bahwa Anda menghabiskan sebagian besar waktu itu untuk menulis ulang teks kasus penggunaan Anda.

– **Tip**Menggunakan alat CASE dapat membuat hidup Anda lebih mudah, tetapi diagram ketahanan adalah diagram yang sangat cepat dan sederhana yang dapat Anda tulis di selembar kertas atau papan tulis. Seringkali sangat membantu untukbuat sketsa diagram Anda di atas kertas sebelum mencoba menggambarnya di komputer(terutama ketika Anda baru pertama kali mempelajari tekniknya).

Diagram Anatomi Robustness

Sebuah diagram ketahanan agak dari hibrida antara diagram kelas dan diagram aktivitas. Ini adalah representasi bergambar dari perilaku yang dijelaskan oleh kasus penggunaan, menunjukkan kelas yang berpartisipasi dan perilaku perangkat lunak, meskipun sengaja menghindari menunjukkan kelas mana yang bertanggung jawab atas bit perilaku mana. Setiap kelas diwakili oleh ikon stereotip grafis (lihat Gambar 5-2). Namun, diagram ketahanan lebih mirip dengan**diagram aktivitas** (atau diagram alur), dalam arti bahwa satu objek "berbicara" dengan objek berikutnya. Alur tindakan ini diwakili oleh garis antara dua objek yang berbicara satu sama lain.

Ada korelasi langsung 1:1 antara alur tindakan dalam diagram ketahanan dan langkah-langkah yang dijelaskan dalam teks kasus penggunaan.



Gambar 5-2.*Simbol diagram ketahanan*

Tiga stereotip kelas yang ditunjukkan pada Gambar 5-2 adalah sebagai berikut:

- **objek batas**: “Antarmuka” antara sistem dan dunia luar (pikirkan kembali Gambar 3-2). Objek batas biasanya layar atau halaman web (yaitu,**lapisan presentasi**yang berinteraksi dengan aktor).
- **Objek entitas**: Kelas dari model domain (lihat Bab 2).
- **Pengendali**: The "lem" antara objek batas dan entitas.

Ini berguna untuk memikirkan objek batas dan objek entitas sebagai kata benda, dan pengontrol sebagai kata kerja. Ingatlah aturan berikut saat menggambar diagram ketahanan Anda:

• Kata benda dapat berbicara dengan kata kerja (dan sebaliknya).

• Kata benda tidak dapat berbicara dengan kata benda lain.

• Kata kerja dapat berbicara dengan kata kerja lain.

Kami akan meninjau kembali aturan ini nanti dalam bab ini (lihat Gambar 5-8 dan 5-9).

- **Latihan**Dua dari berikut ini adalah konstruksi hukum, tetapi dua yang mana?

sebuah. Batas - Pengontrol - Entitas

b.Entitas - Entitas

c.Pengontrol - Pengontrol

d.Batas - Batas - Pengontrol

Aturan ini**membantu untuk menegakkan pola kata benda-kata kerja-kata benda dalam teks kasus penggunaan Anda**. Jika teks kasus penggunaan Anda mengikuti pola ini, diagram kekokohan sangat mudah untuk digambar; jika tidak, diagramnya bisa sangat sulit untuk digambar.

Anggap ini sebagai sinyal peringatan dini: jika Anda tidak dapat menggambar diagram ketahanan sederhana dari use case, bagaimana Anda akan membuat desain detail darinya?**Diagram urutan sepenuhnya bersifat kata benda-kata kerja-kata benda**: objek adalah kata benda, dan pesan yang ada di antara mereka adalah kata kerja. Jadi dengan mendapatkan teks Anda dalam format kata benda-kata kerja-kata benda sekarang, Anda membuat tugas desain terperinci jauh lebih mudah daripada yang seharusnya.

Analisis ketahanan memberikan pemeriksaan kewarasan untuk kasus penggunaan Anda.

Analisis Robustness dalam Teori

Di bagian ini, kami menjelaskan teori di balik analisis ketahanan, diselingi dengan contoh-contoh dari proyek Toko Buku Internet. Kami akan mulai dengan 10 pedoman analisis ketahanan teratas kami.

10 Pedoman Analisis Kekokohan Teratas

Prinsip-prinsip yang dibahas dalam bab ini dapat diringkas sebagai daftar pedoman. Daftar 10 teratas kami mengikuti.

10. Tempel teks kasus penggunaan langsung ke diagram ketahanan Anda.

9.Ambil kelas entitas Anda dari model domain, dan tambahkan yang hilang.

8.Berharap untuk menulis ulang (disambiguasi) kasus penggunaan Anda saat menggambar diagram ketahanan.

7.Buat objek batas untuk setiap layar, dan beri nama layar Anda dengan jelas.

6.Ingatlah bahwa pengontrol hanya sesekali**objek kontrol nyata**; mereka lebih biasanya**fungsi perangkat lunak logis**.

5.Jangan khawatir tentang arah panah pada diagram ketahanan.

4.Tidak apa-apa untuk menyeret use case ke diagram ketahanan jika dipanggil dari use case induk.

3.Diagram kekokohan mewakili desain konseptual awal dari use case, bukan desain detail literal.

2.Kelas batas dan entitas pada diagram ketahanan umumnya akan menjadi instance objek pada diagram urutan, sedangkan pengontrol akan menjadi pesan.

1.Ingatlah bahwa diagram kekokohan adalah "gambar objek" dari use case, yang tujuannya adalah untuk memaksa penyempurnaan teks use case dan model objek.

Mari kita telusuri item-item dalam daftar ini secara lebih rinci.

10. Tempel Teks Use Case Langsung ke Diagram Kekokohan Anda

Melakukan ini sangat membantu untuk memperkuat fakta bahwa Anda sedang menggambar objek gambar dari peristiwa yang dijelaskan dalam use case. Selain itu, Anda akan**bekerja melalui use case kalimat pada suatu waktu**saat Anda menggambar diagram, jadi akan lebih mudah jika teks berada di dekat Anda.

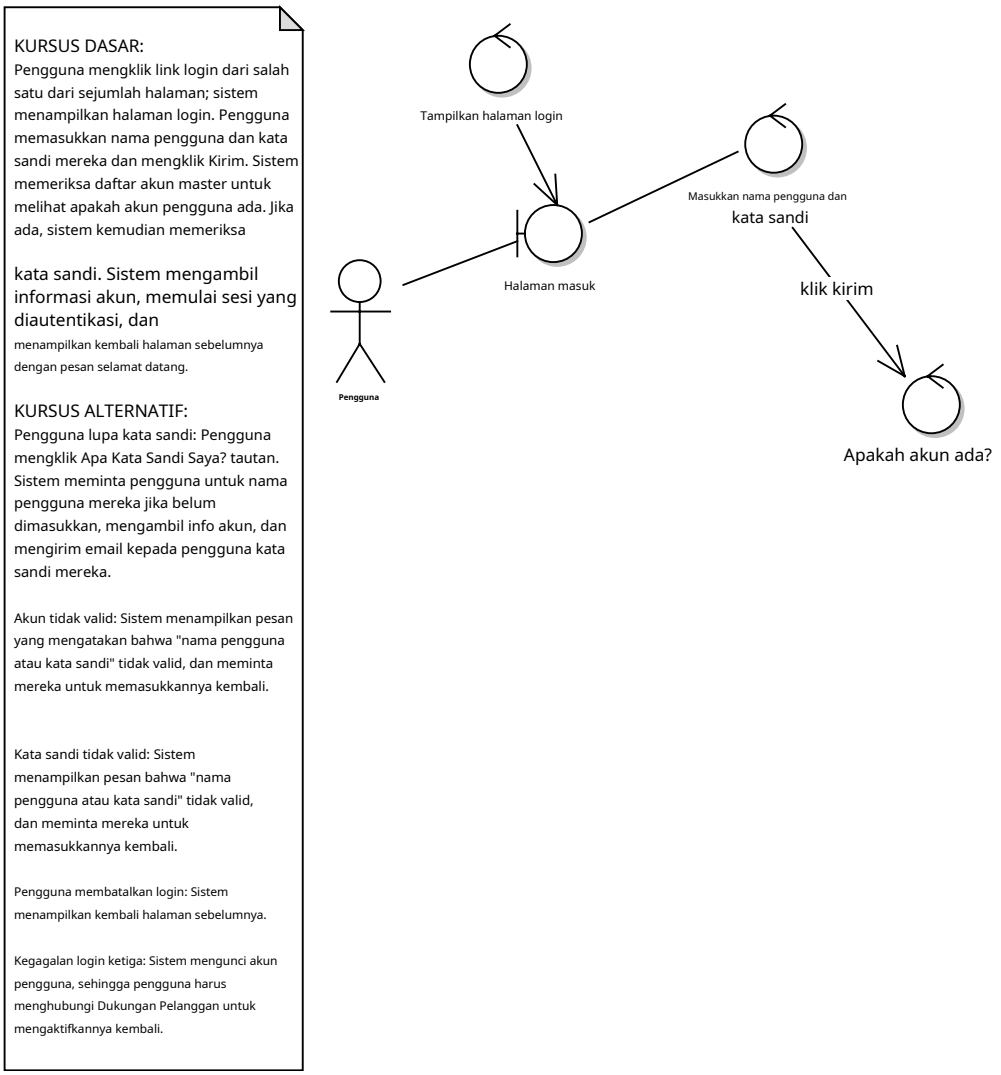
Gambar 5-3 menunjukkan contoh diagram ketahanan dalam proses untuk Toko Buku Internet, untuk *Gabung*kasus penggunaan. Ini adalah snapshot dari diagram pada tahap awal. Sejauh ini, hanya beberapa kalimat pertama dari use case yang telah ditarik ke dalam diagram.

Pada Gambar 5-3, teks use case telah disisipkan langsung ke catatan pada diagram. Karena diagram kekokohan pada dasarnya adalah representasi bergambar dari kasus penggunaan, ada baiknya untuk memiliki teks di sana pada diagram:**mereka adalah dua pandangan yang berbeda dari hal yang sama**, jadi Anda harus dapat menelusuri teks dan menelusurinya pada diagram (dan sebaliknya).

- **Tip**Menggunakan alat CASE seperti EA, dimungkinkan untuk membuat tautan langsung antara use case dan catatan pada diagram ketahanan, sehingga jika use case diperbarui, catatan pada diagram diperbarui secara otomatis.

- **Latihan**Kami menunjukkan versi lengkap dari diagram ini nanti, pada Gambar 5-5. Tetapi sebelum Anda melihatnya, coba selesaikan diagramnya, dengan mengikuti contoh pengontrol dan panah pesan yang telah kita tambahkan sejauh ini pada Gambar 5-3. Cukup ikuti teks use case, dan gambarkan interpretasi literal ke dalam diagram (tanpa mencoba memikirkan detail desain. Ingat, Anda belum melakukan desain OO yang sebenarnya. ***cukup desain awal untuk memvalidasi bahwa Anda memahami kasus penggunaan***).

(Petunjuk: Ini masih awal, karena kami baru saja memperkenalkan konsep dasar, jadi kemungkinan besar akan membuat beberapa kesalahan. Tapi jangan berkecil hati; maksudnya di sini hanya untuk mencobanya, dan kemudian memikirkan diagramnya. yang Anda gambar saat Anda membaca beberapa halaman berikutnya.)



Gambar 5-3.*Diagram ketahanan yang telah selesai sebagian dengan teks kasus penggunaan yang ditempelkan*

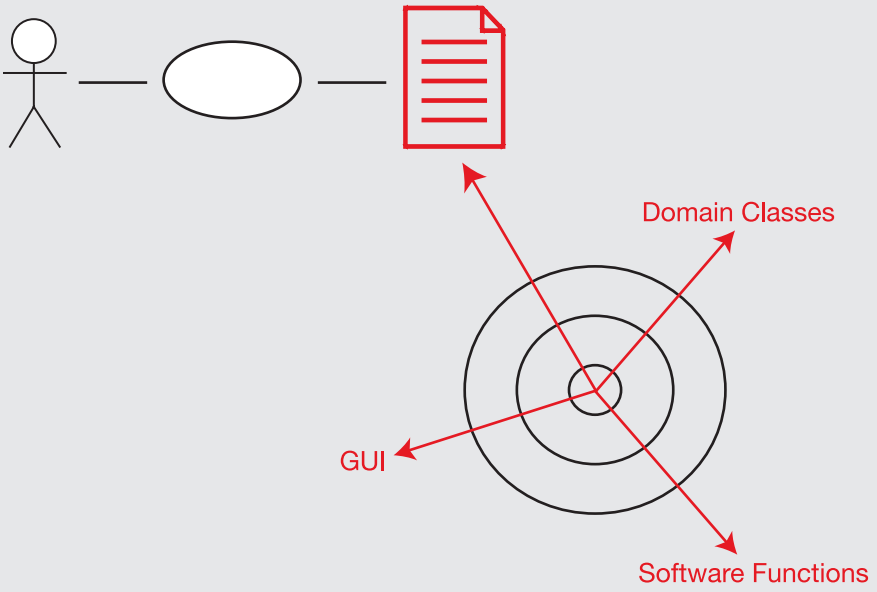
9. Ambil Kelas Entitas Anda dari Model Domain, dan Tambahkan Yang Tidak Ada

Sebagian besar entitas pada diagram ketahanan Anda akan berasal dari model domain Anda. Namun, karena Anda mengatur waktu upaya pemodelan domain awal Anda dalam beberapa jam, wajar jika Anda berharap bahwa Anda mungkin kehilangan beberapa kelas domain. Saat Anda menggambar diagram ketahanan dan ini terjadi, pastikan Anda**tambahkan kelas yang hilang ke dalam model domain**.

Proses ICONIX**mengasumsikan bahwa model domain awal Anda tidak akan lengkap**dan berharap bahwa objek yang hilang akan ditemukan selama analisis ketahanan. Dalam buku ini, kami menyebut proses ini sebagai ***penemuan objek***.

MENGIKAT KASUS PENGGUNAAN ANDA KE DESAIN

Diagram ketahanan mengikat tiga elemen ke kasus penggunaan Anda: GUI, kelas domain, dan daftar fungsi perangkat lunak yang diinginkan (lihat Gambar 5-4).



Gambar 5-4. *Diagram kekokohan mengikat tiga elemen ke kasus penggunaan Anda.*

Seperti yang ditunjukkan Gambar 5-4, persyaratan perilaku yang ditentukan dalam kasus penggunaan Anda perlu menyentuh beberapa aspek berbeda dari sistem, termasuk bagaimana pengguna berinteraksi dengan GUI dan memanipulasi objek inti dari domain masalah. Di antara GUI dan objek domain adalah tempat di mana fungsi perangkat lunak hidup.

Pada diagram ketahanan, Anda menggambarkan elemen GUI menggunakan objek batas, fungsi perangkat lunak menggunakan pengontrol, dan objek domain menggunakan entitas. Perhatikan bahwa ini secara substansial berbeda dari diagram kolaborasi, yang terkadang membingungkan dengan diagram ketahanan. (Diagram kolaborasi hanya menunjukkan interaksi objek.)

8. Berharap untuk Menulis Ulang Kasus Penggunaan Anda Saat Menggambar Diagram Kekokohan

Pengalaman telah menunjukkan bahwa**kasus penggunaan draf pertama**cenderung menunjukkan ciri-ciri berikut: mereka**biasanya tidak jelas, ambigu, tidak lengkap, dan tidak benar**. Tidak mengherankan bahwa begitu banyak proyek telah berjuang dengan kasus penggunaan tanpa adanya a**teknik disambiguasi**seperti analisis ketahanan. Menghilangkan ambiguitas dari use case adalah salah satu tujuan utama dari teknik ini.

"Keajaiban" dari teknik ini pada kenyataannya adalah kerja keras:**menggambar diagram ketahanan memaksa Anda untuk mengerjakan kasus penggunaan satu kalimat pada satu waktu**. Tindakan sederhana ini hampir

selalu memunculkan kesalahan dalam teks use case draf pertama, jadi penting untuk menulis ulang use case secara paralel dengan menggambar diagram ketahanan.

7. Buat Objek Batas untuk Setiap Layar

Menggambar diagram kekokohan dapat menegakkan penamaan yang tidak ambigu (atau, seperti yang ingin kami katakan,*nomenklatur yang jelas*dari objek batas Anda[1]). Jika Anda melihat objek batas berlabel "halaman web" pada diagram ketahanan, berhenti, cari tahu nama halaman, dan gunakan nama asli.

6. Ingat bahwa Kontroler Biasanya Fungsi Perangkat Lunak Logis

Tentu saja mungkin untuk memiliki kelas intensif kontrol dalam desain Anda (misalnya, kelas manajer), dan Anda pasti dapat merepresentasikannya sebagai pengontrol. Namun, jangan berasumsi bahwa setiap pengontrol pada diagram ketahanan akan mewakili kelas kontrol yang sebenarnya. Dalam banyak kasus, pengontrol pada diagram ketahanan hanya digunakan sebagai pengganti fungsi perangkat lunak. Terlalu sering menggunakan kelas pengontrol (misalnya, satu pengontrol kasus penggunaan per kasus penggunaan) dalam sebuah desain dapat membawa kita kembali ke dekomposisi fungsional, jadi kelas pengontrol harus digunakan dengan hemat.[2] Menampilkan campuran objek dan fungsi adalah salah satu cara lain di mana diagram ketahanan secara substansial berbeda dari diagram kolaborasi.

Jika Anda melihat sekelompok pengontrol pada diagram ketahanan yang semuanya berkomunikasi satu sama lain, maka itu adalah kandidat yang baik untuk kelas manajer (terutama jika perilaku status terbatas tidak sepele). Jika Anda merasa perlu menggambar diagram status untuk kasus penggunaan, Anda mungkin juga memerlukan kelas pengontrol, tetapi sebagian besar kasus penggunaan Anda umumnya tidak intensif-negara (bahkan dalam beberapa sistem waktu nyata).

5. Jangan Khawatir Tentang Arah Panah pada Diagram Kekokohan

Ingatlah bahwa diagram ketahanan Anda memiliki dua misi utama dalam hidup:

- Untuk memaksa Anda untuk**disambiguasi teks kasus penggunaan Anda**
- Untuk membantu Anda untuk**temukan benda-benda yang hilang**dalam model domain Anda

Arah mana yang ditunjuk panah pada diagram kekokohan tidak melakukan apa pun untuk memajukan salah satu dari tujuan ini. Sebagai konsekuensi,arah panah saja. . . tidak. . .***urusan***. Dengan serius. Percayalah pada kami. Ini benar-benar tidak. Oh, dan satu hal lagi: itu tidak penting.

Secara formal, panah pada diagram ketahanan mewakili*asosiasi komunikasi*. Anda dapat menunjukkan aliran data atau aliran kontrol dan, jika kami tidak menyebutkannya sebelumnya, arah panahnya adalah**bukan**penting.

---

1. Untuk kepentingan XPers yang mungkin membaca buku ini (!), itulah "DisambiguatedNomenclatureOfYourBoundaryObjects."

2. Seperti yang akan Anda lihat nanti, tren saat ini tampaknya menjadi penggunaan kelas pengontrol yang lebih berlebihan, di mana setiap fungsi perangkat lunak sebenarnya memiliki kelas pengontrol. Tampaknya bagi kita bahwa industri mungkin telah mengambil langkah mundur yang besar dengan pemikiran semacam ini.

4. Tunjukkan Kasus Penggunaan yang Dipanggil pada Diagram Kekokohan Anda

Tidak apa-apa untuk menyeret use case ke diagram ketahanan jika dipanggil dari use case induk.

Tidak hanya boleh melakukan ini, tetapi juga cara paling sederhana untuk menunjukkan satu kasus dipanggil oleh kasus lain pada diagram ketahanan. Faktanya, itu satu-satunya cara masuk akal yang kami temukan. Cobalah—ini bekerja dengan sangat baik.

3. Diagram Robustness Mewakili Desain Konseptual Awal dari Use Case

Berikut adalah beberapa kebenaran mendasar tentang pengembangan sistem:

- Merupakan ide yang baik untuk memahami sepenuhnya persyaratan sebelum melakukan desain.
- Sering**tidak mungkin untuk sepenuhnya memahami persyaratan tanpa melakukan beberapa desain eksplorasi**.

Kedua pernyataan ini mungkin tampak bertentangan, tetapi solusinya cukup sederhana: Anda dapat melakukan desain konseptual untuk tujuan**memvalidasi persyaratan perilaku*sebelum melakukan desain yang sebenarnya***, dari mana Anda akan membuat kode. Diagram kekokohan mewakili desain konseptual, sedangkan desain sebenarnya ditunjukkan pada diagram urutan.

Pemrogram sering mengalami masalah dengan diagram ketahanan karena mereka terbiasa berpikir dalam kerangka desain detail yang konkret, dan mereka perlu mengambil langkah mundur dari berpikiran literal dan belajar berpikir pada tingkat konseptual yang sedikit lebih abstrak. Ini bisa rumit, karena pemrograman cenderung menjadi keterampilan yang berpikiran sangat literal. Namun, setelah Anda menguasai keterampilan**memanipulasi desain pada tingkat konseptual**abstraksi, Anda akan menemukan bahwa sejumlah manfaat dihasilkan, terutama kemampuan untuk menulis kasus penggunaan yang tepat dan tidak ambigu.

Kabar baiknya adalah, menguasai keterampilan ini tidak mengharuskan Anda pensiun ke biara Tibet untuk bermeditasi dan berlatih selama beberapa tahun; hanya membutuhkan beberapa jam untuk menggambar diagram.

2. Objek pada Diagram Robustness Anda Akan "Morph" menjadi Detail Desain

Kelas batas dan entitas pada diagram ketahanan umumnya akan menjadi instance objek pada diagram urutan, sedangkan pengontrol akan menjadi pesan. Juga disarankan untuk membuat kasus uji untuk pengontrol.

Ingatlah bahwa objek batas dan objek entitas adalah kata benda, dan pengontrol adalah kata kerja (yaitu, tindakan yang dilakukan pada objek). Dengan demikian, masuk akal bahwa pengontrol (tindakan) akan menjadi metode pada kelas batas dan entitas.

1. Ingat Bahwa Robustness Diagram Adalah “Gambar Obyek” dari Use Case

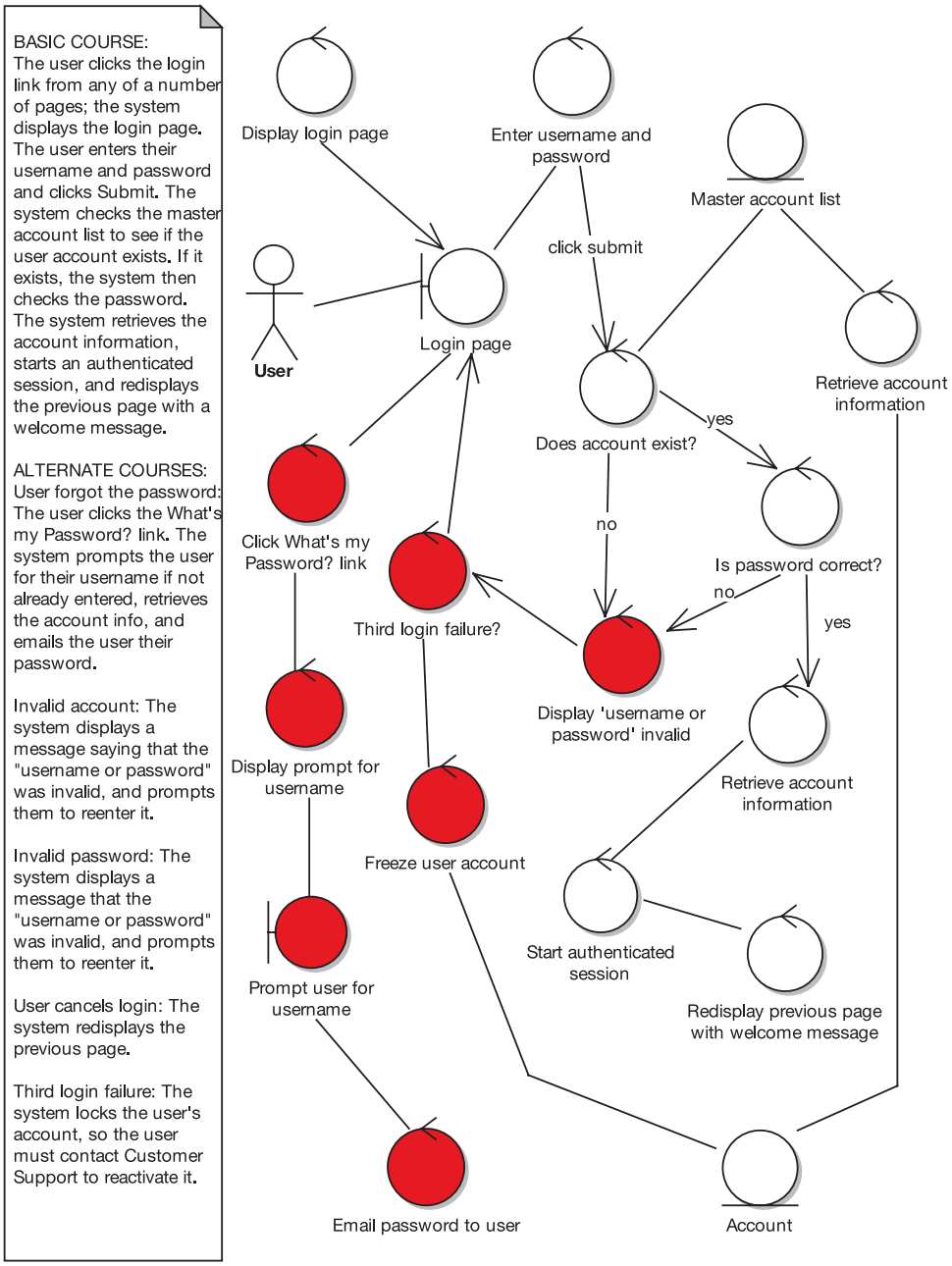
Diagram kekokohan adalah "gambar objek" dari use case, yang tujuannya adalah untuk memaksa penyempurnaan teks use case dan model objek.***Diagram ketangguhan mengikat kasus penggunaan ke objek (dan ke GUI).***

Diagram ketahanan tidak sama dengan diagram kolaborasi UML. Anda menunjukkan komunikasi objek-ke-objek pada diagram kolaborasi, tetapi diagram ketahanan secara harfiah adalah gambaran objek dari use case. Karena diagram kekokohan dan teks use case harus sama persis, diagram kekokohan memaksa Anda untuk mengikat teks use case ke objek, sehingga memungkinkan Anda untuk mendorong desain berorientasi objek maju dari use case.

Implikasi penting dari semua ini adalah, karena diagram ketahanan harus menunjukkan semua use case,***itu harus menunjukkan tidak hanya kursus dasar, tetapi semua kursus alternatif juga***

(semua pada diagram yang sama). Ini adalah alasan bagus mengapa kasus penggunaan Anda harus mengikuti aturan dua paragraf (lihat Bab 3).

Gambar 5-5 menunjukkan diagram lengkap untuk *Gabung* use case (yang kita mulai sebelumnya dalam bab ini), menunjukkan kursus dasar dan kursus alternatif.



Gambar 5-5. *Contoh diagram ketahanan*

- **Tip**Pada Gambar 5-5, beberapa objek diarsir merah. Ini adalah objek (terutama pengontrol) untuk kursus alternatif. Meskipun tidak penting, akan sangat membantu untuk menunjukkan kursus alternatif dalam warna yang berbeda dari kursus dasar. Efek yang sama dapat dicapai (dan memiliki manfaat tambahan sebagai bentuk tinjauan) dengan mencetak diagram dan menggunakan spidol berwarna berbeda untuk melacak kursus dasar dan kursus alternatif.

APAKAH SAYA BENAR-BENAR MEMBUTUHKAN SEMUA KONTOL TAMPILAN #$%^ itu?

Masalah umum yang menjadi perhatian sebagian orang saat menggambar diagram ketahanan adalah bahwa diagram mereka terkadang memiliki sejumlah pengontrol Tampilan.

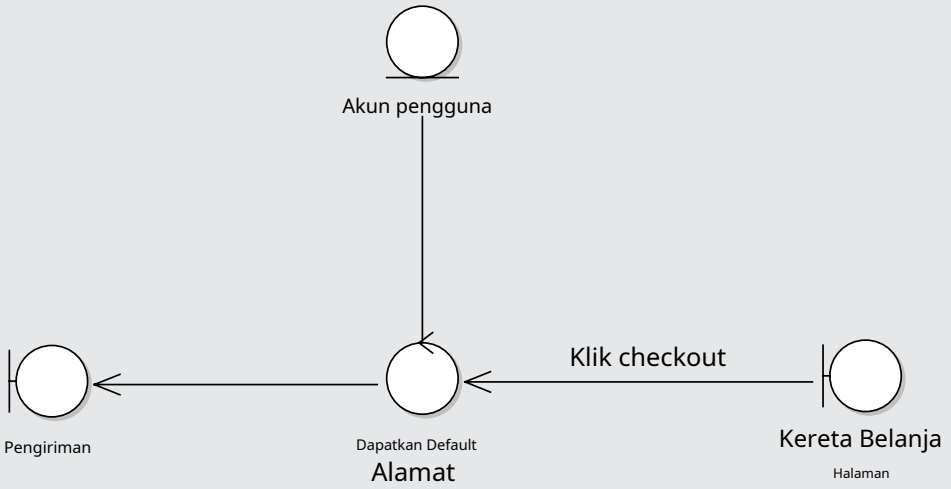
Umumnya, jika pengontrol berbicara dengan objek batas (seperti pada Gambar 5-6), maka tidak akan melanggar aturan kata benda-kata-kata benda diagram untuk mengabaikan pengontrol Tampilan tambahan. Fakta bahwa halaman akan ditampilkan sudah ditunjukkan oleh panah dari pengontrol Dapatkan Alamat yang Diminta ke objek batas Halaman Alamat Pengiriman.

Namun, jika kamu**anggap Tampilan sebagai "Inisialisasi halaman,"**masuk akal untuk menempatkan pengontrol Tampilan pada diagram di mana pun diperlukan (lihat Gambar 5-7). Bahkan, jika itu membantu, sebut saja "Inisialisasi halaman" alih-alih "Tampilan." Pada Gambar 5-7, Anda dapat melihat bahwa sistem mendapatkan alamat pengiriman default dan kemudian menginisialisasi halaman dengan pengaturan default (melalui pengontrol Tampilan).

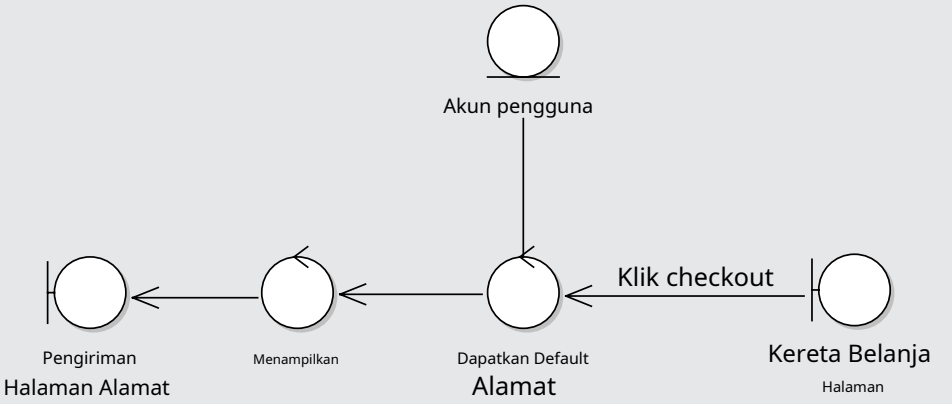
Kode inisialisasi tampilan cenderung tidak sepele, sehingga membantu menempatkan pengontrol Tampilan secara eksplisit pada diagram ketahanan. Jika Anda mulai mengabaikan pengontrol Tampilan, maka Anda**sebenarnya melewatkan banyak perilaku inisialisasi dalam kasus penggunaan**. Untuk sebagian besar, ini bukan hal yang ingin Anda lupakan. Saat Anda menggambar pengontrol Tampilan, tanyakan pada diri Anda, “Apa yang ditampilkan di layar ini? Apakah saya harus mengambilnya dari database?” (dll.)

Faktanya, ketika Anda mulai menggambar diagram urutan dari teks use case (lihat Bab 8), pengontrol Tampilan biasanya menjadi operasi pada kelas batas, tetapi itu adalah keputusan desain (mengalokasikan operasi ke kelas), dan Anda tidak boleh khawatir tentang detail desain saat menggambar diagram ketahanan.

Seperti yang akan Anda lihat di Bab 12, Anda dapat membuat kasus uji langsung dari pengontrol pada diagram ketahanan Anda, jadi ini adalah dorongan lain untuk menambahkan pengontrol Tampilan itu!



Gambar 5-6.*Pengontrol berbicara langsung ke objek batas*



Gambar 5-7.*Pengontrol berbicara dengan objek batas melalui pengontrol Tampilan*

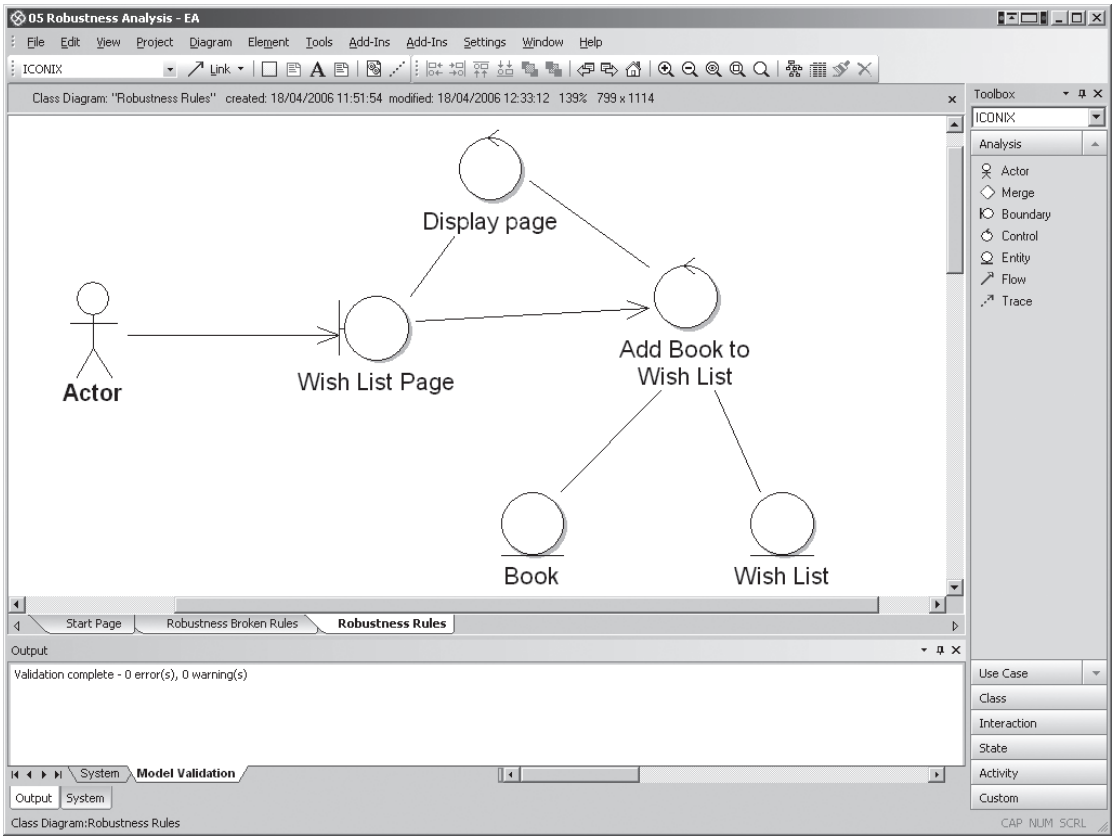
Lebih Lanjut Tentang Aturan Diagram Kekokohan

Aturan kata benda-kata kerja-kata benda diagram ketahanan mungkin tampak tidak perlu membatasi pada awalnya, tetapi pada kenyataannya mereka membantu Anda untuk mempersiapkan teks kasus penggunaan Anda untuk aturan yang jauh lebih ketat (jika kadang-kadang implisit) yang perlu Anda terapkan saat Anda membuat desain dari kasus penggunaan Anda. Aturan diagram ketahanan mudah dipelajari, tetapi sekarang ada dukungan alat yang muncul untuk menangkap pelanggaran aturan. Gambar 5-8 dan 5-9 menunjukkan alat pemodelan favorit kami, EA, yang memvalidasi beberapa diagram ketahanan untuk kesalahan model. Sejauh yang kami tahu, orang-orang di Sparx Systems (www.sparxsystems.com)adalah satu-satunya yang telah menerapkan pemeriksaan aturan untuk diagram ini, pada saat penulisan ini.

Dalam Gambar

sendiri tidak

jabatan adalah



Gambar 5-8.*Semua mungkin**sah**hubungan diagram ketahanan*

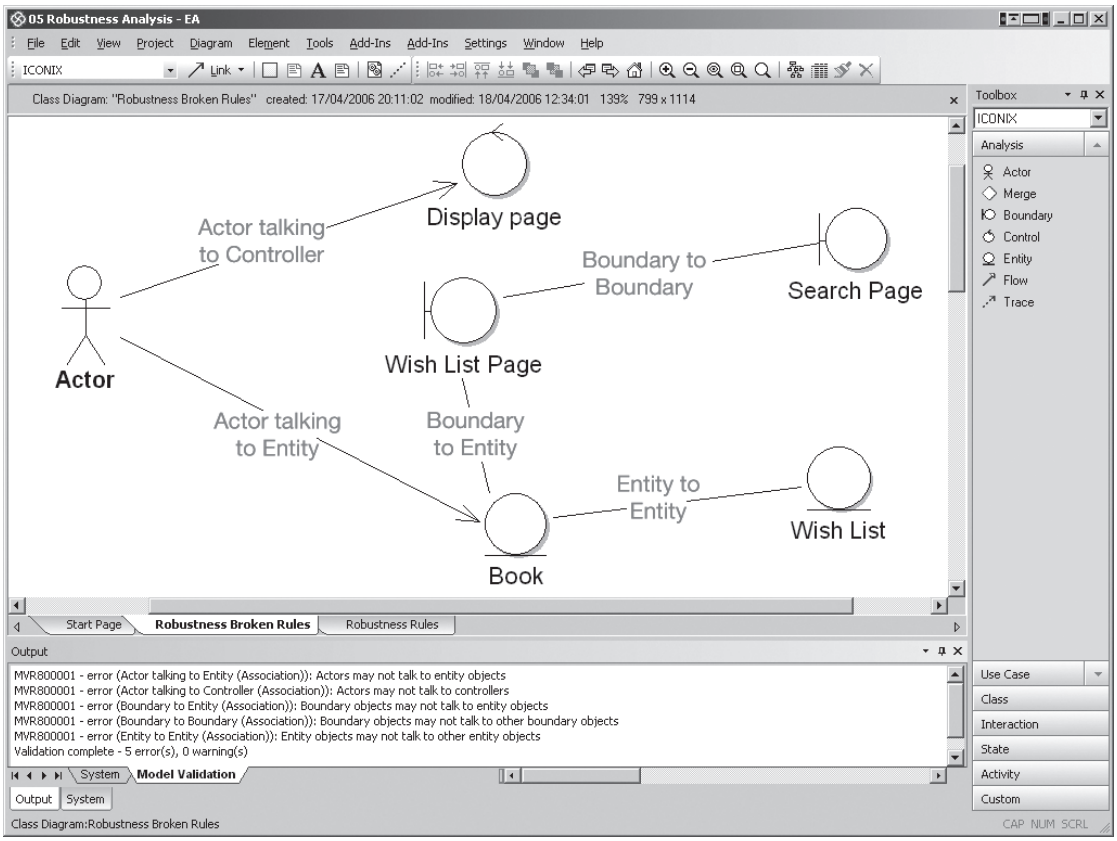
Hubungan yang ditunjukkan pada Gambar 5-8 diperbolehkan karena

• Seorang Aktor dapat berbicara dengan Objek Batas.

• Objek Batas dan Pengendali dapat berbicara satu sama lain(Kata Benda <-> Kata Kerja).

• Pengontrol dapat berbicara dengan Pengontrol lain(Kata Kerja <-> Kata Kerja).

• Pengendali dan Objek Entitas dapat berbicara satu sama lain(Kata Kerja <-> Kata Benda).



Gambar 5-9.*Pemeriksa aturan diagram ketahanan di EA menunjukkan semua kemungkinan**tidak sah**hubungan*

Hubungan yang ditunjukkan pada Gambar 5-9 adalah**tidak diperbolehkan**karena

- Aktor tidak dapat berbicara langsung dengan Pengendali atau Entitas(harus berbicara dengan Objek Batas).
- Objek Batas dan Objek Entitas tidak dapat berbicara langsung satu sama lain(harus melalui Controller).
- Entitas tidak dapat berbicara langsung dengan Entitas lain(harus melalui Controller).
- Objek Batas tidak dapat berbicara langsung dengan Objek Batas lainnya(harus melalui Controller).

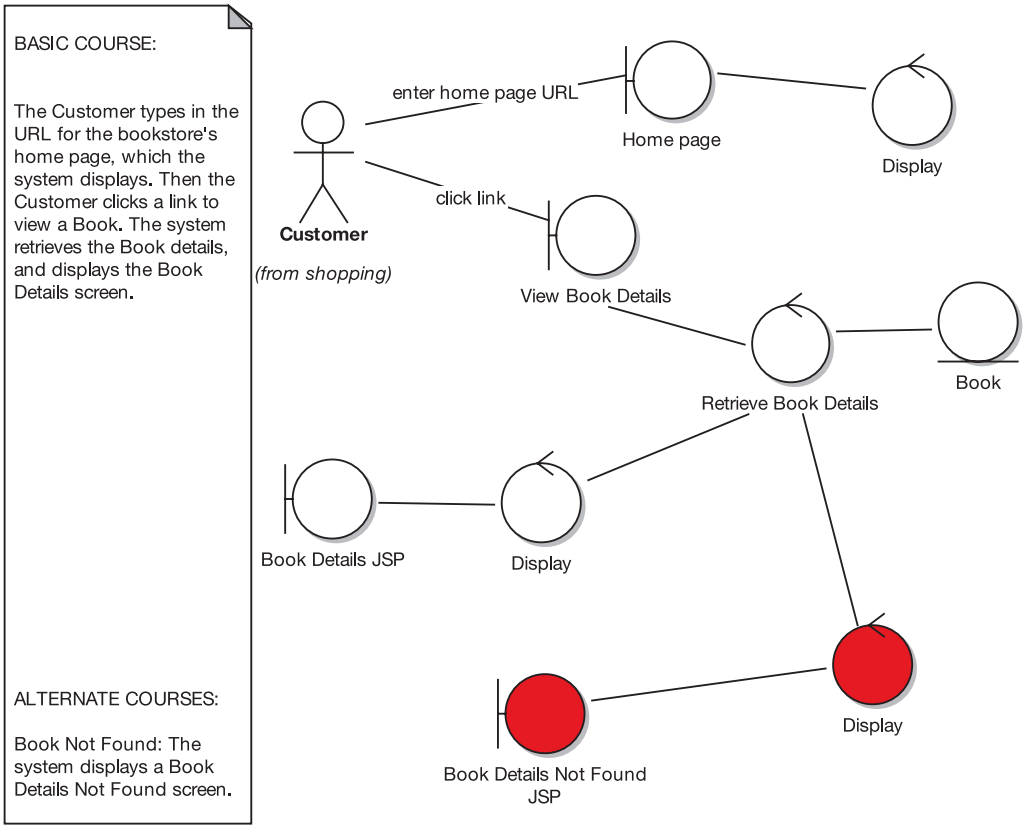
Bagaimana Anda Melakukan Analisis Kekokohan?

Anda melakukan analisis ketahanan untuk kasus penggunaan dengan**mengerjakan teks use case, satu kalimat pada satu waktu**, dan menggambar aktor, objek dan pengontrol batas dan entitas yang sesuai, dan hubungan di antara berbagai elemen diagram. Anda harus dapat menyesuaikan kursus dasar dan semua kursus alternatif dalam satu diagram.

Sekarang untuk beberapa contoh (keduanya dari Toko Buku Internet). Kita akan mulai dengan diagram kekokohan yang lengkap, dan kemudian kita akan menelusuri diagram lain selangkah demi selangkah.

Diagram Kekokohan untuk Kasus Penggunaan "Tampilkan Detail Buku"

Gambar 5-10 menunjukkan upaya pertama pada diagram ketahanan untuk*Tampilkan Detail Buku*use case, yang kami perkenalkan di Bab 4.



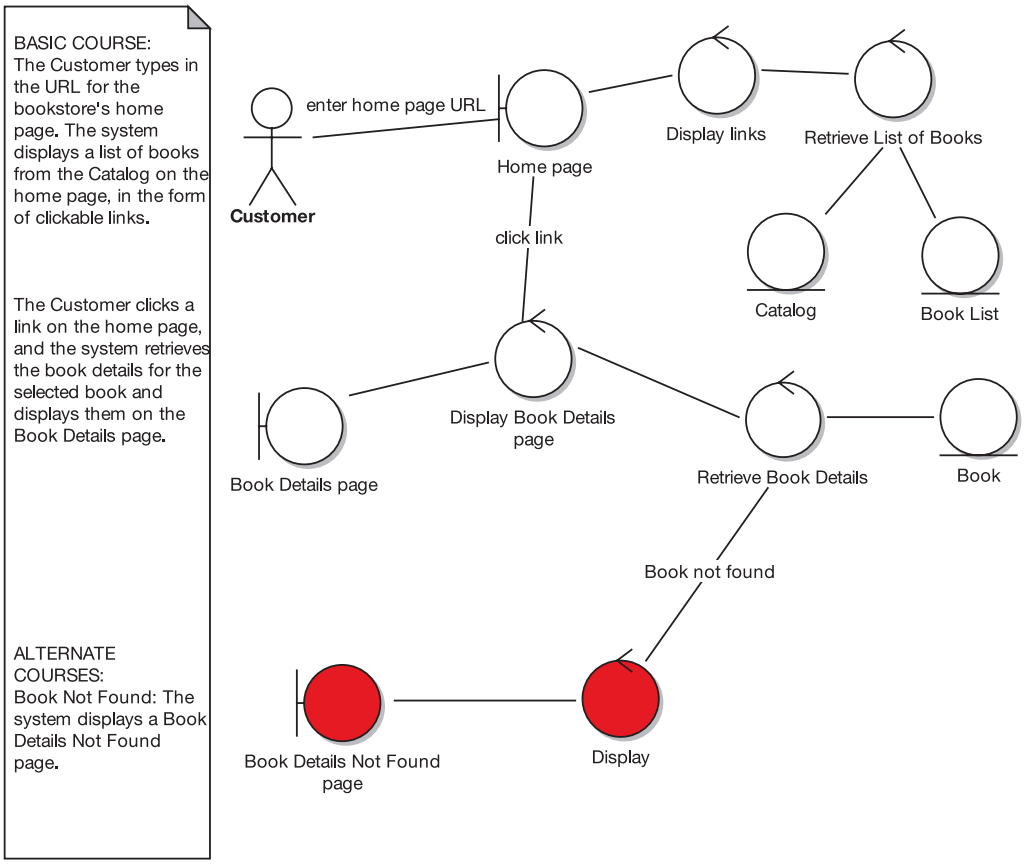
Gambar 5-10.*Upaya pertama pada diagram ketahanan untuk*Tampilkan Detail Buku*kasus penggunaan*

Seperti yang Anda lihat, Anda dapat menelusuri diagram kekokohan dengan membaca teks use case dan mengikuti diagram itu sendiri. Jika Anda pernah menemukan ada sesuatu dalam teks use case yang tidak ada dalam diagram, maka diagram tersebut tidak lengkap (dan sebaliknya).

Ada beberapa masalah dengan diagram ini yang perlu diselesaikan:

- "Link klik" harus berasal dari objek batas halaman beranda dan menuju ke pengontrol yang mengarahkan tampilan halaman detail buku (mungkin ini adalah "halaman detail buku" dan bukan "halaman detail tampilan buku"). Selalu tunjukkan tindakan pengguna yang akan datang *mati* objek batas dan dengan demikian menautkan layar/halaman sebelumnya ke layar/halaman berikutnya melalui pengontrol.
- Saat ini ada dua kelas batas: "lihat halaman detail buku" dan "detail buku JSP". Apakah benar-benar ada dua batasan pada tingkat desain konseptual? Atau apakah keduanya mewakili halaman detail buku? Apakah batas "lihat halaman detail buku" sebenarnya seharusnya menjadi pengontrol yang disebut "tampilkan halaman detail buku"?
- Jangan menyebut hal-hal JSP (atau ASP, atau apa pun) pada diagram ketahanan, karena itu terlalu spesifik teknologi untuk desain konseptual. Lebih baik menyebutnya layar dan halaman (ini sudah dilakukan dalam teks kasus penggunaan).
- Cobalah untuk tidak mencampur kata benda dan kata kerja seperti dalam "lihat halaman detail buku"—Anda akan membingungkan diri sendiri! **Nama halaman harus berupa kata benda.**

Gambar 5-11 menunjukkan versi yang dikoreksi.



Gambar 5-11. *Diagram ketahanan yang dikoreksi untuk* Tampilkan Detail Buku *kasus penggunaan*

Kami akan kembali ke *Tampilkan Detail Buku* gunakan kasus secara berkala di seluruh buku, dan kami akan membawanya ke kode sumber.

Tentu saja, mudah bagi kami untuk menunjukkan kepada Anda diagram yang sudah jadi dan berkata, "Nah, begitulah cara Anda melakukannya!" Jadi di bagian selanjutnya, kita akan berjalan melalui proses menggambar diagram ketahanan dari awal.

Diagram Kekokohan untuk Kasus Penggunaan “Tulis Ulasan Pelanggan”

Sekarang kita akan berjalan melalui proses analisis ketahanan langkah demi langkah untuk *Tulis Ulasan Pelanggan* kasus penggunaan.

Langkah pertama ditunjukkan pada Gambar 5-12. Kami membuat diagram kekokohan kosong yang baru.

- **Tip** Jadikan diagram ketahanan sebagai diagram anak dari kasus penggunaan yang Anda modelkan. (Hal yang sama berlaku untuk diagram urutan, yang akan Anda tambahkan nanti.) Sarangkan diagram ini "di dalam" use case di browser proyek.

Selanjutnya, tempel teks use case langsung ke diagram.
Tahap selanjutnya adalah membaca kalimat pertama dari kursus dasar dalam teks use case:

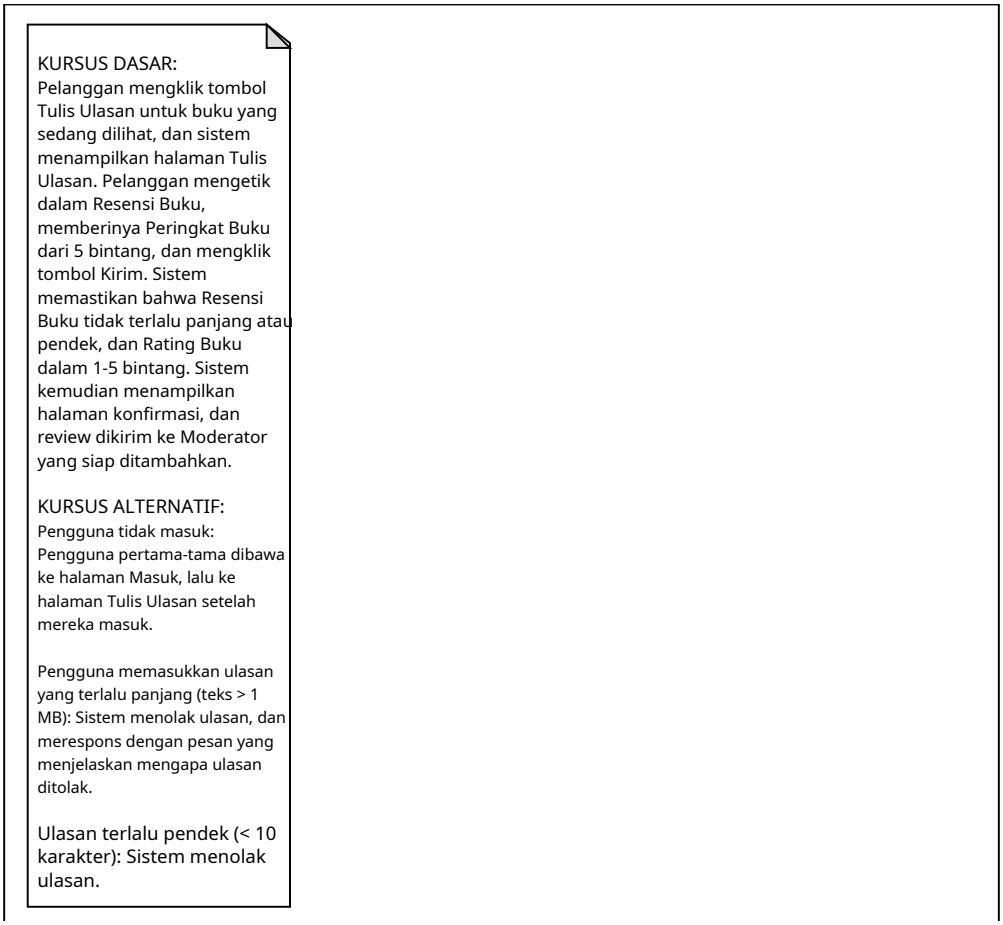
Pelanggan mengklik tombol Tulis Ulasan untuk buku yang sedang dilihat, dan sistem menampilkan layar Tulis Ulasan.

Hal pertama yang dirujuk adalah Pelanggan, jadi kita perlu menempatkan aktor Pelanggan ke dalam diagram.

- **Tip** Anda dapat menyeret aktor langsung dari tampilan hierarki (browser proyek).

- **Tip** Jika Anda menatap layar bertanya-tanya bagaimana memulainya. . . baik, Anda tidak sendirian. Saat Anda mempelajari cara menggambar diagram ketahanan, memulai diagram baru biasanya merupakan bagian yang paling sulit.

Cara termudah untuk memulai adalah dengan **mulai dari kalimat pertama teks use case dan gambar apa yang Anda baca**. Jika itu tidak akan diterjemahkan dengan mudah ke dalam diagram, maka mungkin saja kasus penggunaan dimulai pada titik yang salah (misalnya, jika menggambarkan tindakan yang mengarah ke tindakan pertama pengguna, maka itu mungkin menggambarkan bagian dari penggunaan yang berbeda kasus dan harus ditulis ulang).

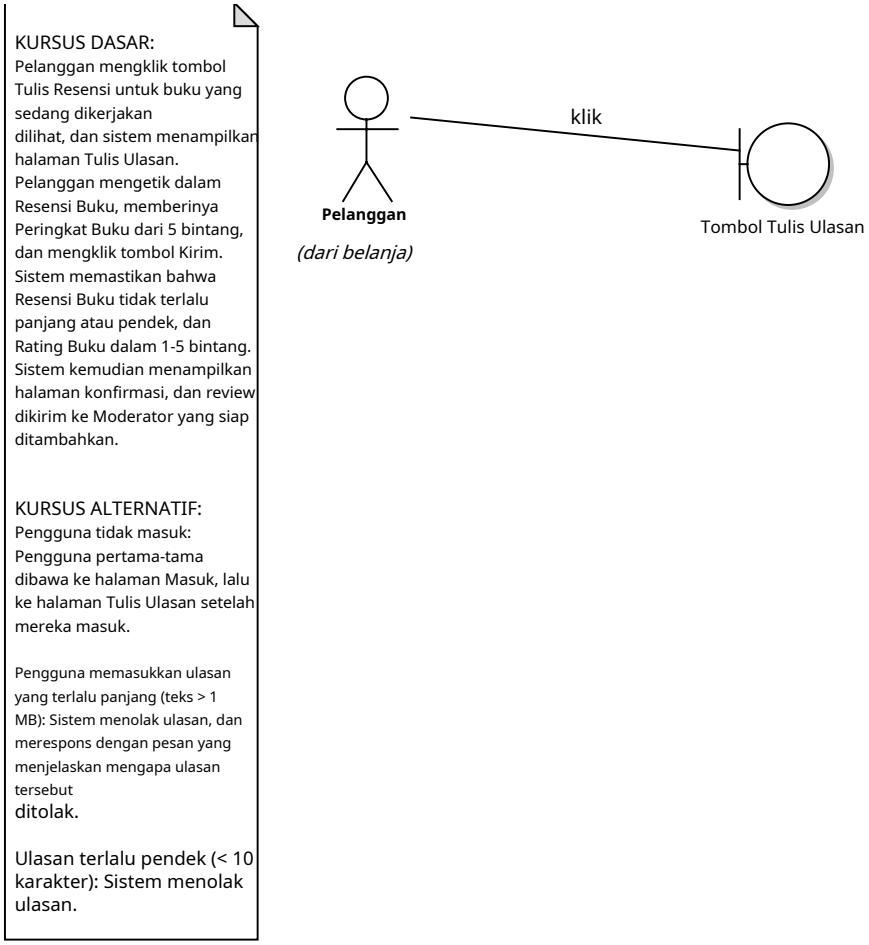


Gambar 5-12. *Langkah 1: Buat yang baru, kosong* Tulis Ulasan Pelanggan *diagram ketahanan*

Selanjutnya, itu *tampaknya* seperti hal yang jelas untuk dilakukan adalah menampilkan tombol Tulis Ulasan sebagai objek batas dan menunjukkan Pelanggan berinteraksi dengannya (lihat Gambar 5-13).

Anda mungkin bertanya-tanya apakah boleh menempatkan widget GUI seperti tombol pada diagram ketahanan kami. Dalam praktiknya, kami menemukan bahwa hal itu membuka kotak Pandora. Jika Anda menyertakan satu widget GUI, maka Anda mulai berpikir bahwa Anda harus menyertakan semuanya, yang berarti . . . Anda akan berada di sana sepanjang malam menggambar pengontrol dan objek batas untuk semua bidang teks, kotak daftar, tombol, label, dan seterusnya untuk setiap layar. Astaga. Lebih baik untuk menghindari jatuh ke dalam perangkap itu, dan ***hindari menggambar widget GUI individual (di bawah layar/halaman/tingkat bingkai) pada diagram ketahanan***.

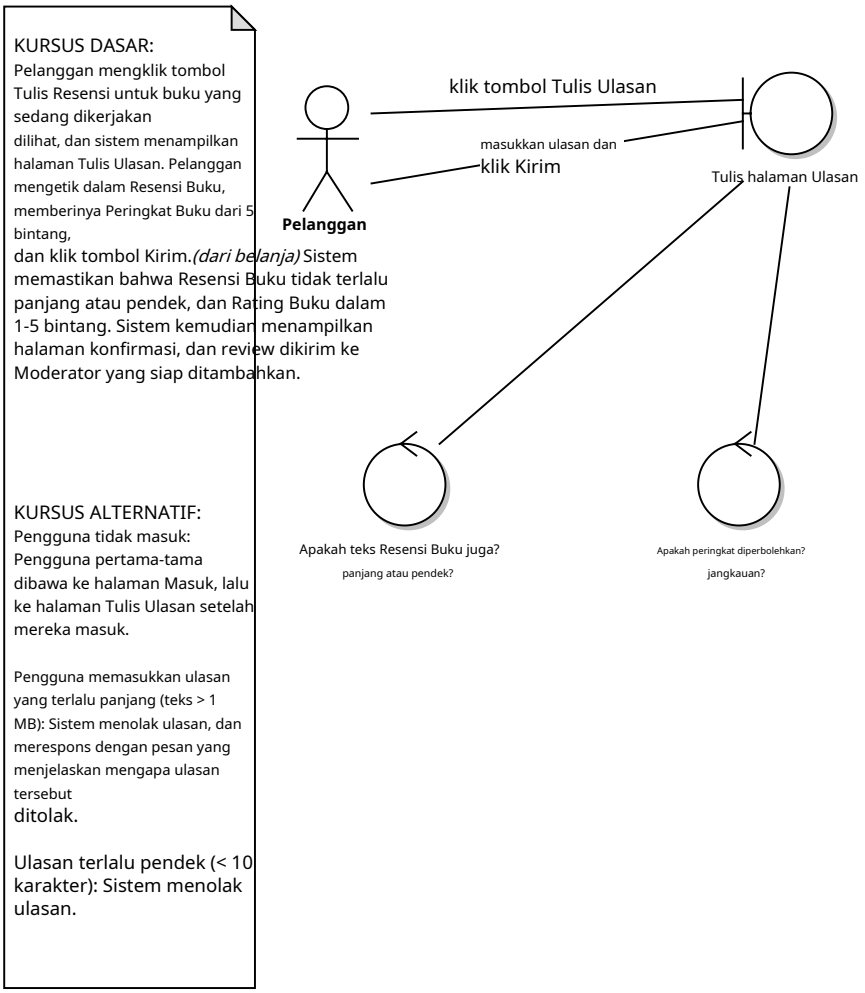
Seperti yang Anda lihat pada Gambar 5-14, kami telah menghapus objek batas Tombol Tulis Ulasan dan menurunkannya ke pesan antara Pelanggan dan objek Layar Ulasan Tulis. Jika elemen UI mutlak harus disebutkan secara eksplisit (misalnya, jika Anda merasa itu membuat diagram lebih jelas), maka itu bisa dimasukkan sebagai label pesan, seperti yang telah kita lakukan pada Gambar 5-15. Ini tidak penting, meskipun—bahkan, diagram mungkin akan sedikit lebih jelas tanpanya.



Gambar 5-13. *Langkah 2: Temukan kesalahan yang disengaja.*

Pada Gambar 5-14, kami juga telah menambahkan beberapa pengontrol untuk mewakili validasi yang dijelaskan dalam teks use case. Teks yang direpresentasikan dalam diagram (sejauh ini) adalah sebagai berikut:

> *Pelanggan mengetik dalam Resensi Buku, memberinya Peringkat Buku dari lima bintang, dan mengklik tombol Kirim. Sistem memastikan bahwa Resensi Buku tidak terlalu panjang atau pendek, dan Peringkat Buku berada dalam satu dan lima bintang.*



Gambar 5-14.*Langkah 3: Kami sekarang telah memperbaiki kesalahan dan menambahkan beberapa pengontrol validasi.*

Ada beberapa masalah dengan diagram ini sejauh ini. Dua pesan antara Pelanggan dan objek batas Layar Tulis Ulasan cukup kikuk. Selain itu, pendekatan ini menimbulkan beberapa ambiguitas ke dalam diagram, karena tidak jelas pengontrol mana yang dipanggil saat pengguna mengklik tombol Tulis Ulasan versus saat pengguna memasukkan ulasan dan mengklik Kirim. Ternyata, kalimat pertama dari teks use case (“Pelanggan mengklik tombol Tulis Ulasan untuk buku yang sedang dilihat”) mengisyaratkan layar Detail Buku, yang tidak ditampilkan baik dalam teks maupun diagram . Jika kita menambahkan layar itu, maka kita dapat memiliki panah dari Pelanggan ke layar Detail Buku, dan kita juga dapat menambahkan pengontrol Tampilan untuk menampilkan layar Tulis Ulasan yang sedang ditampilkan.

Masalah lainnya adalah nama pengontrol "Apakah teks Resensi Buku terlalu panjang atau pendek?" agak panjang, jadi kita bisa mempersingkatnya menjadi "Apakah panjang Resensi Buku OK?"