11 Aplicativo móvel Teste

A tecnologia dos computadores muda rapidamente. Em um piscar de olhos o computador passou do desktop para o laptop e agora para o dispositivo móvel portátil. Essa migração mudou a maneira como conduzimos nossas vidas, negócios e governos. Também afetou significativamente a maneira como os desenvolvedores e testadores de software fazem seu trabalho.

A maioria dos profissionais de teste de software considera o teste de aplicativos móveis muito desafiador – mais do que quase qualquer outro tipo ou plataforma de software. Na verdade, são os dispositivos e o ambiente móvel mais do que o ''aplicativo'' que impõem o desafio. Esses dois componentes adicionam muitas variáveis e complexidades que podem distorcer ou mascarar problemas em seu aplicativo, o que dificulta o projeto de um plano de teste robusto. Resumidamente, você precisa considerar o desempenho e a confiabilidade da rede, interfaces de usuário consistentes, influências do transcodificador, diversidade de dispositivos e plataformas de recursos limitados.

Neste capítulo, apresentamos uma área relativamente nova de teste de software: testar aplicativos para dispositivos móveis e smartphones. Começamos descrevendo o ambiente de aplicativo móvel, que difere daquele de um aplicativo autônomo em desktops, laptops e servidores. Em seguida, enumeramos os desafios de testar aplicativos móveis – alguns dos quais abordamos anteriormente neste livro. Por fim, abordamos algumas abordagens de teste e considerações de casos de teste para ajudar a diminuir sua curva de aprendizado neste novo território.
Depois de ler este capítulo, você deve entender melhor os desafios e obstáculos de testar aplicativos móveis.

Ambiente móvel

Com a ampla implantação de hotspots sem fio, a linha entre a computação móvel e as atividades baseadas em rede sem fio "tradicional" ficou turva.
Assim, para começar aqui precisamos definir os termos dispositivo móvel e aplicativos móveis, com relação ao conteúdo deste capítulo. Nesse sentido, nos referimos a um dispositivo móvel como aquele que tem a capacidade de executar aplicativos baseados em rede por meio de um link de dados de celular ou satélite. Isso abrange a maioria dos smartphones, tablets e PDAs. Dito isso, não cometa o erro de identificar dispositivos móveis apenas pela aparência. Os laptops modernos podem aceitar placas de celular ou satélite plug-in, e alguns laptops têm esse acesso integrado. Com base nessa definição de dispositivo móvel, um aplicativo móvel é um programa baseado em rede que é executado em um dispositivo móvel.

Essa distinção é importante. Sim, é verdade que a maioria dos dispositivos móveis pode usar hotspots e pontos de acesso sem fio sem problemas. No entanto, essas conexões oferecem maior confiabilidade e velocidades mais altas do que as redes celulares, mesmo com a adoção das tecnologias 3G e 4G. Assim, você projeta seu aplicativo móvel com a expectativa de que ele usará links de dados relativamente lentos e comparativamente não confiáveis. Você também pode desenvolver aplicativos autônomos, como jogos, que rodam em um dispositivo móvel sem a necessidade de usar a rede da operadora. Mas, para os propósitos deste capítulo, não consideramos aplicativos independentes como aplicativos móveis. Nosso foco está nos desafios associados a aplicativos executados em redes de dados celu

A chave para criar planos de teste bem-sucedidos para seus aplicativos móveis é entender o ambiente de computação móvel. A Tabela 11.1 identifica várias áreas cruciais que você deve investigar ao projetar planos de teste.
Primeiro, você deve entender os problemas de conectividade do dispositivo e as velocidades de rede, disponibilidade regional e latência. Tenha em mente a filosofia subjacente deste livro: Seus testes não devem provar que seu aplicativo funciona, mas que seu aplicativo não funciona para os casos de uso. Por exemplo, se você tiver um serviço baseado em localização ou aplicativo de e-mail, seus testes deverão identificar problemas de software quando a rede da operadora estiver lenta ou indisponível.

A seguir, há três áreas relacionadas a dispositivos — diversidade, restrições e métodos de entrada — que abordaremos em detalhes mais adiante neste capítulo. Para criar planos de teste bem-sucedidos, você e sua equipe de teste devem considerar os vários dispositivos no mercado, os recursos variados de cada um e como o usuário interage com os dispositivos.

TABELA 11.1 Considerações de projeto de teste de ambiente móvel

Área	Comente
Conectividade	Provisionamento de dispositivos
	Velocidade da rede
	Latência da rede
	Disponibilidade de rede em áreas remotas
	Confiabilidade do serviço
Dispositivos de diversidade	Vários navegadores da web para testar
	Várias versões de ambientes de execução para Java ou outras linguagens
Restrições do dispositivo	Memória ou processador limitado
	Tamanho de tela pequeno
	Vários sistemas operacionais
	Recursos multitarefa
	Tamanhos de cache de dados
Dispositivos de entrada	Telas sensíveis ao toque
	Caneta
	Rato
	Botões
	Rolos
Instalação e Manutenção	Instalando e desinstalando
	Aplicação de patches
	Atualizando

Por último, você precisa determinar como instalar e manter seu aplicativo. Alguns fornecedores, como a Apple, mantêm lojas online onde o usuário compra o aplicativo, mas somente após a Apple certificar seu aplicativo para sua plataforma. Isso facilita um pouco a instalação e a manutenção, pois você tem um sistema de distribuição único e certificado.

Desafios de teste

Como afirmado, o teste de aplicativos móveis está repleto de desafios. Para ajudar a alcançá-los, podemos categorizar a maioria em quatro categorias: diversidade de dispositivos, infraestrutura de rede da operadora, scripts e usabilidade. Você precisa pensar cuidadosamente em cada um ao projetar casos de teste. A combinação de cisalhamento de tipos de dispositivos, sistemas operacionais, métodos de entrada do usuário e preocupações de rede significam que as compensações devem ser equilibradas com tempo, finanças e r

recursos para chegar a um plano de teste econômico que detecte a maioria dos bugs em um prazo razoável. A construção de uma estratégia de teste que combine os métodos discutidos nos capítulos anteriores ajudará.

No restante desta seção, discutimos essas categorias e oferecemos conselhos sobre como lidar com cada um.

Diversidade de dispositivos móveis

A diversidade cada vez maior de dispositivos apresenta um desafio de teste significativo e muitas vezes subestimado para alguém novo no teste de aplicativos móveis. Às vezes parece que os fabricantes introduzem novos dispositivos diariamente, tornando quase impossível acompanhar os ciclos de lançamento.
Pior, mais dispositivos significam mais itens a serem considerados em seus testes. Aqui está um exemplo simples para ilustrar apenas alguns itens que você precisa avaliar quando um novo dispositivo é lançado:

> Suponha que a Motorola desenvolva um novo método de entrada de texto via tela sensível ao toque para seus telefones baseados em Android. Você pode projetar um teste para determinar se o novo método de entrada do dispositivo interrompe seu aplicativo? Em caso afirmativo, você pode corrigir seu aplicativo sem interromper o suporte para outros dispositivos baseados em Android, como tablets? Você pode até obter um dispositivo para testar? Você tem acesso a uma rede de operadoras suportadas?

Quase por definição, junto com a diversidade de dispositivos vem a diversidade de sistemas operacionais, navegadores, ambientes de tempo de execução de aplicativos, resoluções de tela, interfaces de usuário, ergonomia, tamanho de tela e muito mais. Você deve estar ciente de todos esses fatores ao criar testes. A diversidade de dispositivos também força o teste de usabilidade na frente e no centro, o que em algum momento exige que os testadores avaliem seu aplicativo nos dispositivos de destino. Usar emuladores é uma ótima maneira de começar, mas no final você precisará testar dispositivos reais em redes de operadoras reais.

Isso levanta outra faceta do teste de aplicativos móveis: testar em dispositivos reais versus emuladores. Do ponto de vista econômico, você deve fazer o máximo de testes possível com emuladores. Pode ser inviável financeiramente, mesmo que você consiga obter um dispositivo e acessar a rede sem fio, para testar na plataforma real. Dito isso, os emuladores apenas emulam; eles não são os dispositivos reais. Portanto, é provável que você observe diferenças entre os testes

com um emulador e o dispositivo real. Por exemplo, as cores e formas de botões e caixas de entrada podem passar nos testes de aceitação em um emulador, mas falhar no dispositivo de destino devido às diferenças de resolução de tela e profundidade de cor entre o dispositivo e um emulador baseado em PC.

Em suma, você precisa perceber que pode haver centenas de dispositivos móveis com potencial para acessar seu aplicativo. Portanto, durante as fases de coleta de requisitos e redação de especificações, você será solicitado a tomar algumas decisões difíceis e escolher um subconjunto razoável de dispositivos para suporte e teste. Esteja ciente de que todos os dispositivos que você não testar podem não funcionar com seu aplicativo; portanto, você pode perder não apenas um cliente, mas também uma base de clientes.

Infraestrutura de rede da operadora

Testar seu aplicativo em uma rede de operadora configura outro desafio.
Isso é especialmente verdadeiro se você deseja oferecer suporte a várias operadoras. Dois dos maiores obstáculos a serem superados são: entender e adaptar-se à infraestrutura da operadora e superar obstáculos baseados em localização.

Compreender a infraestrutura de uma operadora é fundamental para desenvolver um bom plano de teste. Inicialmente, você pensaria que seu aplicativo móvel usa a rede de uma operadora como um ponto de acesso sem fio IP. Não tão. A Figura 11.1 ilustra a infraestrutura “típica” da maioria das operadoras sem fio. A primeira diferença é que o protocolo não é baseado em IP; geralmente é um protocolo baseado em RF

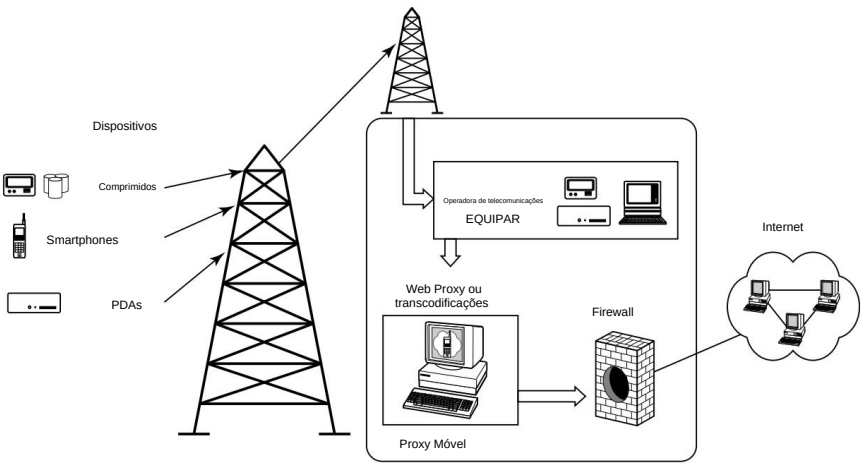

FIGURA 11.1 Rede de dados de operadora sem fio genérica.

como acesso múltiplo por divisão de código (CDMA), acesso múltiplo por divisão de tempo (TDMA) ou sistema global para celular (GSM). Os protocolos baseados em RF tratam os protocolos baseados em IP como uma "carga útil" e os entregam ao dispositivo móvel, que então decodifica a carga útil e a apresenta ao aplicativo.

Além disso, a maioria das operadoras usa algum tipo de transcodificador ou proxy da Web entre a Internet e o dispositivo. Esses dispositivos podem executar uma variedade de funções. E às vezes é difícil determinar exatamente o que ocorre, a menos que você trabalhe diretamente com as operadoras. Muitas vezes, eles não revelam essas informações para fins competitivos. A seguir, uma pequena lista do que pode ocorrer no proxy ou transcodificador da Web de uma operadora:

- Transforme ou transcodifique conteúdo em WAP ou HTTP.
- Compacte os dados para um melhor rendimento.
- Criptografe o tráfego para privacidade e segurança.
- Bloqueie o acesso a determinados sites de alta largura de banda.
- Retire os cabeçalhos HTML e outros metadados das páginas da Web que seu aplicativo pode usar.

A transcodificação pode causar inconsistências na interface do usuário em vários dispositivos. Alguns dispositivos suportam WAP (Wireless Application Protocol), enquanto outros suportam HTTP. WAP usa Wireless Markup Language (WML) para entrega de conteúdo. WAP e WML foram planejados para serem o "padrão" para entrega de conteúdo sem fio, mas nunca ganharam uma base sólida. No entanto, vários dispositivos o implementam, então você pode encontrá-lo durante seus testes. No entanto, a maioria dos smartphones e tablets suportam HTML e, portanto, dependem de HTTP para fornecer conteúdo. Se você tiver problemas de interface do usuário entre dispositivos e operadoras, verifique com cada um para determinar se WAP/WML ou HTTP/HTML está sendo usado.

Embora a compactação de dados se destine a melhorar a taxa de transferência, muitas vezes durante os períodos de alta atividade a taxa de transferência pode diminuir devido à sobrecarga da compactação. O mesmo vale para a segurança: firewalls e camadas semelhantes podem diminuir a taxa de transferência durante horas de alto volume.

Finalmente, você deve superar os obstáculos baseados em localização. Obviamente, para testar na rede de uma operadora, você precisa ter acesso a ela. Por exemplo, e se você tiver um aplicativo de viagem para um smartphone: como você testa redes de operadoras em outras partes do país ou em outros países? Resposta: Você deve viajar para lá ou contratar alguém para testá-lo para você. Ambos aumentam o custo do teste.

Script

Uma área frequentemente negligenciada de teste de aplicativos móveis é a criação e execução de scripts de teste. Dispositivos reais não permitem que você carregue scripts automatizados e repetíveis no dispositivo; a equipe de teste executa manualmente todos os scripts. Ou seja, alguém percorre um script de teste escrito projetado para encontrar erros em um caso de teste no dispositivo de destino. Observe que dissemos "dispositivo de destino". Existem muitos destinos no ambiente móvel.

Como apontamos nos capítulos anteriores, o teste manual é propenso a erros. Infelizmente, isso é inevitável ao testar aplicativos móveis em dispositivos reais. Como mencionado, a maioria dos emuladores possui uma rica funcionalidade de script e pode realizar a maior parte dos testes de regressão e do sistema. No entanto, no final, você ainda precisa ter alguém para trabalhar com o dispositivo. (Mais adiante neste capítulo, explicaremos como criar um script de teste manual genérico para suportar vários dispositivos.)

A notícia refrescante é que os dispositivos móveis estão se tornando muito mais sofisticados e poderosos. Dada a competitividade do mercado, é razoável esperar que um produto de script automatizado apareça. O iOS da Apple, o sistema operacional Windows Mobile e o sistema operacional Android estão amadurecendo rapidamente, portanto, é provável que esse problema não seja um problema em versões futuras.

Usabilidade

O teste de usabilidade apresenta desafios semelhantes aos dos scripts de teste. Lembre-se dos capítulos anteriores de que o teste de usabilidade é principalmente uma abordagem de caixa branca. Assim como testar aplicativos de desktop independentes, uma equipe de teste deve tentar manualmente encontrar bugs na interface do usuário e nas camadas de interação do usuário do seu aplicativo.

Ao contrário do teste de aplicativos de desktop independentes, o teste de dispositivos móveis envolve mais de uma plataforma para testar. Por exemplo, você desejará pesquisar problemas de consistência da interface do usuário entre os produtos da Apple e as plataformas baseadas em Android. Embora você esteja testando aplicativos móveis, muitas das discussões do Capítulo 7 se aplicam.

Abordagens de teste

Algumas áreas de teste de dispositivos móveis são semelhantes ao teste de aplicativos da Internet, especialmente ao avaliar as infraestruturas de back-end. O prefeito

a diferença está em como você aborda o teste do próprio dispositivo. Com Internet testando, você tem apenas um punhado de navegadores para avaliar; com dispositivos móveis, você tem exponencialmente mais.

Naturalmente, ao testar componentes de back-end, você deve empregar técnicas semelhantes e avaliar considerações semelhantes às discutidas em Capítulo 10, "Teste de aplicativos da Internet". Voltando à Figura 10.1, os níveis 2 e 3 devem ter aproximadamente a mesma configuração que um aplicativo de Internet normal. Como uma revisão rápida, você deve testar as especificações de desempenho, rotinas de validação de dados e componentes de processamento de transações da camada 2. Testar a camada 3 também é o mesmo que com a Internet formulários; testar tempos de resposta, integridade de dados, tolerância a falhas e capacidade de recuperação nesta camada. Se possível, teste os componentes de nível 2 e 3 separadamente do dispositivo para garantir que eles atendam às especificações do seu projeto usando testes de função.

A camada de teste 1, o ambiente do usuário, difere da Internet tradicional teste. Os conceitos apresentados ao testar seu conteúdo e site arquitetura ainda se aplicam. No entanto, o teste do ambiente do usuário equivale a teste do dispositivo.

Devemos observar a importância dos casos de uso ao desenvolver planos de teste para seus dispositivos. Saber quem usará seu aplicativo e como e quando, é imperativo, pois os aplicativos móveis têm vários pontos de falha. A Tabela 11.2 lista os itens que você geralmente não considera ao projetar casos de teste para aplicativos padrão, sejam autônomos ou baseados na Web. Por Por exemplo, testar seu aplicativo na rede da operadora é extremamente importante. Você deseja encontrar problemas relacionados à cobertura irregular ou perda de conectividade. Se sua aplicação envolve transferências de dados, procure problemas com cache de dados e sincronização incompleta com back-end armazenamentos de dados. O que acontece quando a cobertura é restaurada repentinamente após uma interrupção durante o download de um aplicativo? A compra ocorre duas vezes? Verifique se há bugs relacionados ao tratamento de reinicialização de sessão e corrupção de dados. Alguns desses problemas se aplicam a aplicativos baseados na Web executados em um Navegador baseado em PC. No entanto, as LANs/WANs são muito mais estáveis. Quando lidar com redes celulares, você deve esperar perder a conectividade.

Um caso de teste específico para testes móveis é como seu aplicativo lida chamadas de voz e mensagens de texto recebidas. As chances são de que os usuários finais vão querer para suspender seu aplicativo ou executá-lo em segundo plano, enquanto atendem o telefone ou lêem a mensagem de texto. Tente construir casos de teste em que chamadas e mensagens futuras causem problemas em seu aplicativo.

TABELA 11.2 Categorias de teste para teste de aplicativos móveis

Categoria de teste	Descrição
Instalar/Desinstalar	Certifique-se de que o usuário possa instalar corretamente seu aplicativo.
	Certifique-se de que o usuário possa desinstalar completamente seu aplicativo.
Rede	Verifique se o aplicativo responde adequadamente à perda de rede.
A infraestrutura	
	Verifique se o aplicativo responde adequadamente à restauração da rede.
	Verifique se o aplicativo responde adequadamente a sinais fracos.
Chamada recebida/	Teste se o usuário pode aceitar chamadas/mensagens de texto enquanto o
Mensagem	aplicativo está em execução.
Manipulação	Teste se o usuário pode retomar o aplicativo ao finalizar chamadas/mensagens de texto.
	Teste se o usuário pode rejeitar chamadas/mensagens de texto sem interromper o aplicativo.
	Teste se o usuário pode iniciar uma chamada/mensagem de texto sem interromper o aplicativo.
Memória baixa	Certifique-se de que o aplicativo permaneça estável quando o dispositivo encontrar uma situação de pouca memória.
Mapeamentos de teclas	Teste se todos os mapeamentos de teclas funcionam conforme especificado.
Comentários	Certifique-se de que o feedback do usuário ao pressionar a tecla ocorra dentro das especificações de design do aplicativo.
Saindo	Verifique se o aplicativo sai normalmente quando iniciado pressionando as teclas, fechando a tampa ou usando o controle deslizante.
	Confirme se o aplicativo atende às especificações de design quando o usuário inicia o desligamento do dispositivo.
Carregamento	Certifique-se de que o aplicativo funcione conforme projetado ao entrar no modo de carregamento.
	Certifique-se de que o aplicativo funcione conforme projetado no modo de carregamento.
	Certifique-se de que o aplicativo funcione conforme projetado ao sair do modo de carregamento.
Bateria	Teste como o aplicativo se comporta com bateria fraca.
Condições	Meça a rapidez com que o aplicativo drena a bateria.
	Certifique-se de que o aplicativo responda de acordo com a especificação quando a bateria for removida enquanto o dispositivo estiver ligado.
Interação do dispositivo	Certifique-se de que o aplicativo não sobrecarregue a CPU.
	Certifique-se de que o aplicativo não consuma muita memória.

TABELA 11.3 Dispositivos versus Emuladores

Teste	Desvantagens da abordagem	Vantagens
Real Dispositivos	Caro, especialmente se você segmentar uma ampla base de dispositivos móveis	Capacidade de testar a capacidade de resposta do aplicativo
	Incapacidade de instalar ferramentas de desenvolvimento de medição ou diagnóstico	Inspeção visual do aplicativo no dispositivo real para verificar a consistência da interface do usuário
	Não é possível instalar em scripts de teste de execução	Rede de operadoras de teste capacidade de resposta
	Disponibilidade de rede	Identifique bugs específicos do dispositivo
Emuladores	Incapacidade de identificar bugs relacionados ao dispositivo	Custo-benefício
	O hardware subjacente pode distorcer o desempenho no dispositivo real	Fácil de gerenciar; suporte a vários dispositivos com um único emulador

No restante do capítulo, abordaremos algumas abordagens para teste de dispositivos nas quais você basicamente tem duas opções: testar em dispositivos reais ou usar emuladores de dispositivos. A Tabela 11.3 oferece algumas vantagens e desvantagens de cada abordagem.

Testes com dispositivos reais

O teste manual com dispositivos reais é inevitável. Apesar de caro, tem algumas vantagens. Somente testando com o dispositivo você pode experimentar suas nuances e ter uma noção real da experiência do usuário. Além disso, você só pode testar determinados casos com dispositivos reais. Testar a confiabilidade da rede de uma operadora e determinar o efeito de uma chamada ou mensagem de texto recebidas são exemplos óbvios. Em um dispositivo real você também pode avaliar como seu aplicativo se comporta. Ele carrega rápido e roda a uma velocidade aceitável? Parece bom?
A interface do usuário é consistente em seus dispositivos de destino? Por último, mas não menos importante, você pode determinar bugs específicos do dispositivo. Isso é quase impossível com um emulador. Se você encontrar um bug específico do dispositivo, o desafio é corrigi-lo sem quebrar a compatibilidade com outros dispositivos.

Apesar das vantagens, testar com dispositivos reais também apresenta algumas sérias desvantagens. Por exemplo, é caro porque você deve comprar o dispositivo,

bem como pagar pelo tempo de antena da transportadora. Nenhum dos dois é barato e, se você estiver testando vários dispositivos de várias operadoras em várias regiões, as despesas aumentam de acordo. Alguns fabricantes de dispositivos e provedores de serviços têm dispositivos que você pode alugar ou acessar remotamente, o que pode reduzir alguns custos. Se você segmentar uma plataforma individual, como a família Apple iPhone, poderá ser poupado de grande parte dessa despesa. Ainda assim, você precisará de uma quantidade suficiente de cada tipo (iPad, iPhone, iTouch) para testar.

Além disso, testar com dispositivos reais é um processo manual de caixa branca. Alguém deve apertar os botões, tocar nas telas e inserir dados. Como você sabe, o teste manual é propenso a erros, mesmo com as melhores instruções e testadores treinados. Além disso, adiciona outra despesa ao processo. Você deve manter anotações precisas sobre cada script de teste bem documentado e seus resultados.
Em seguida, avalie a eficácia dos scripts e elimine aqueles com pouco ou nenhum valor (ou seja, não encontre bugs).

Como observamos anteriormente, o uso de dispositivos reais elimina uma arma importante no arsenal do testador de software: scripts de teste automatizados. Portanto, você deve usar scripts manuais escritos que especificam ações genéricas, não detalhes sobre como executar a ação em um dispositivo. Scripts de teste detalhados para cada dispositivo seriam um desafio para criar e manter. Em pouco tempo, você teria uma biblioteca de scripts, que pode ficar obsoleta quando o dispositivo for atualizado. Os scripts genéricos permitem testar as especificações do sistema em vários dispositivos.

Por exemplo, iPhones, iPads e dispositivos baseados em Android dependem muito de telas sensíveis ao toque para entrada do usuário. Outros dispositivos, como BlackBerries ou telefones "padrão", possuem teclados ou teclados numéricos para permitir a entrada do usuário. A Tabela 11.4 fornece um script de exemplo para verificar se seu aplicativo, um e-reader, aborta se você receber uma mensagem de texto enquanto lê um e-book. Observe que o script não especifica exatamente como executar nenhuma etapa, apenas executar a etapa usando os recursos de entrada do usuário do dispositivo. Podem ser botões, telas sensíveis ao toque ou comandos de voz. Em nenhum momento você especifica "Pressione OK" ou "Pressione Enviar". Essa abordagem genérica permitirá que você avalie casos de teste em vários dispositivos.

Por último, os fabricantes geralmente "bloqueiam" dispositivos reais, o que significa que você não pode carregar ferramentas para monitorar ou depurar seu aplicativo. Então, quando você atinge um bug, é mais difícil isolar o problema. Por exemplo, se seu aplicativo estiver lento, você não saberá se é a rede da operadora, problemas de transcodificação, seu aplicativo ou uma combinação deles. Somente por tentativa e erro você pode identificar problemas.

TABELA 11.4 Script de teste de dispositivo genérico

1. Inicie o aplicativo e-reader.
2. Abra o e-book.
3. Inicie a mensagem SMS para o dispositivo a partir de outro dispositivo.
4. Verifique se o alerta de mensagem SMS é exibido.
5. Abra a mensagem SMS.
6. Escolha Responder à mensagem SMS.
7. Compor mensagem SMS.
8. Envie uma mensagem SMS.
9. Verifique a notificação de envio de mensagem SMS.
10. Retorne ao e-book.
11. Verifique se o aplicativo de e-book está em execução.
12. Verifique o retorno à mesma página ou marcador.
13. Saia do aplicativo e-reader.

Testando com emuladores

Testar com emuladores pode não ser a abordagem preferida, mas geralmente é a mais prática e econômica, e ainda tem algumas vantagens.
Primeiro, os emuladores permitem testes funcionais rápidos e baratos de seu aplicativo. Você pode percorrer o aplicativo para encontrar eventos e circunstâncias que não atendem aos requisitos do programa. Identifique esses bugs usando emuladores antes de gastar com testes de dispositivos.

Em segundo lugar, os emuladores são fáceis de gerenciar e, como são executados em PCs, todo testador ou desenvolvedor pode ter um emulador. Os desenvolvedores podem gerenciar o software por conta própria, eliminando a necessidade de administradores de sistema.
Terceiro, a maioria dos pacotes de emuladores suporta vários dispositivos. Para testar um dispositivo diferente, basta carregar um perfil de dispositivo diferente. O melhor de tudo é que você não incorre em custos caros de tempo de antena da transportadora. Quarto, os emuladores são executados em computadores com mais recursos, como CPUs mais rápidas e mais memória. Tempos de resposta rápidos durante os testes permitem que você conclua os testes mais rapidamente.

A última e provavelmente mais significativa vantagem é que a maioria dos emuladores emprega linguagens de script de alto nível, para que você possa criar testes consistentes e automatizados, que são menos propensos a erros e mais rápidos do que os testes manuais.
O script automatizado também permite testes de regressão mais fáceis e rápidos, o que é especialmente importante ao verificar se as alterações feitas no seu

aplicativo para suportar um dispositivo não interrompa o suporte para outro. As linguagens de script em emuladores geralmente são independentes de dispositivo. Referindo-se à Tabela 11.4, quando você executa o script da Etapa 8, "Enviar mensagem SMS", o emulador executará essa função independentemente do dispositivo. Isso permite que scripts sejam usados entre dispositivos.

A desvantagem de usar emuladores para teste é que você não pode identificar as nuances e bugs de cada dispositivo. Como dissemos antes, em algum momento, você deve testar seu aplicativo nos dispositivos de destino. Sem testar em dispositivos reais, você nunca pode ter 100% de certeza de que atende às especificações de compatibilidade e desempenho. No entanto, não descarte o uso de emuladores para a maior parte de seus testes. É uma maneira econômica e eficiente de eliminar a maioria de seus bugs.

Resumo

O teste de aplicativos móveis representa uma nova fronteira no teste de software.
O ambiente móvel adiciona maior complexidade e mais interações não experimentadas ao testar aplicativos autônomos padrão. Dito isso, com uma compreensão dos desafios, você pode melhorar muito suas chances de testar seu aplicativo com sucesso.

Comece tentando obter um controle sobre o universo de dispositivos que você deseja oferecer suporte. Você quer oferecer suporte apenas a smartphones e tablets baseados em Android ou quer dar suporte à maioria dos principais fornecedores de tablets e smartphones? Em seguida, entenda a infraestrutura de rede da operadora. Ele transcodifica, criptografa, compacta ou de alguma forma modifica os dados antes de enviá-los ao dispositivo?

Você também precisa encontrar um equilíbrio entre o emulador e o teste de dispositivo real. Ambos têm seus prós e contras. Devido aos custos, você provavelmente usará mais emuladores e economizará o teste do dispositivo para as fases finais. Use as categorias de teste na Tabela 11.2 como ponto de partida para desenvolver o seu próprio. Consulte as categorias com frequência ao definir seus casos de teste. Além disso, trate qualquer script de teste escrito e resultado como código-fonte; certifique-se de ter backups adequados e alguma forma de controle de alterações nos documentos de teste. Para economizar tempo e dinheiro, revise a eficácia de cada script e elimine aqueles que não agregam valor.

Depois de entender os fundamentos do teste de aplicativos móveis, você não deverá ter problemas para criar planos de teste e casos de uso. Uma coisa é certa, os aplicativos móveis estão aqui e, mais cedo ou mais tarde, você precisará aprender a testar esses aplicativos exclusivos. Por que não começar agora?