SWEN30006
Software Design and Modelling

Case Study:
Chauffeured Cars

"I've yet to see any problem, however complicated, which when you looked at it the right way didn't become still more complicated."

—Poul Anderson

Introduction

- A new case study: Chauffeured Car Company
- Provide illustrative examples for architecture analysis and improvement

Client

Client: Chauffeured Car Company which has about 30 drivers

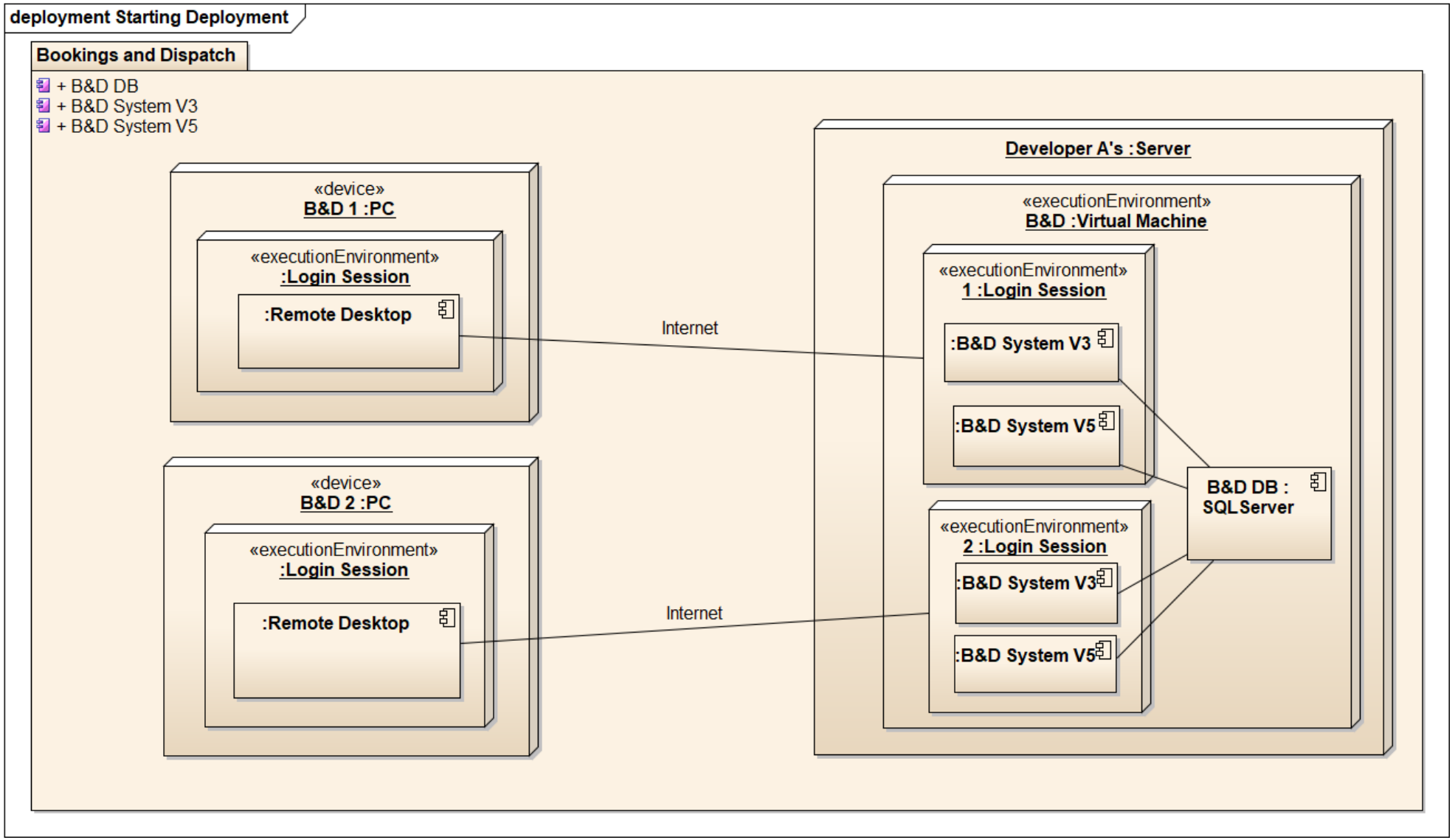
- Bookings and Dispatch System
  - Business Critical
  - Bookings phone and email
  - 1 – 4 Operators concurrently
- Uber-style operations encroaching on business
  - Different business model
  - Need to highlight differences to Customers

Client Problem Perspective

Issues with the current system

- Slow/hung, especially when customer on the phone
- When the system is unavailable, operators requiring paper/pen
- Missing functionality, e.g.
  - Invoicing and reporting
  - Customer Portal (Online booking, Profile, …)
- Unhelpful IT provider: Developer A
- Cost: significant overhead for status quo

Current Architecture



Investigation of Issues

Responsiveness

- Remote Desktop over internet to remote server
- Sharing of limited bandwidth: UI freezes

Reliability

- Connection down → no system → paper/pen

Functionality (→ Modifiability)

- Two versions of system: Version 3 (old) is written in Visual Basic 6 and Version 5 (current) is written in C# for the .Net 4 framework.
- Both versions are needed, e.g. *old version is for* bookings; *current version is for* customers management
- System not accessible to modify in current setup

Technical Issues

Technology for version 3 is dated

- need to remove dependency

Database is V3 vintage

- SQL Sever is recent; but the database (schema including types, use of indexing, use of built-in validation, etc) is based on very old version
- Doesn’t use recent features for performance, reliability, …

Design of V3 and V5 elements

- Generally poor
- In particular, highly coupled

Options

Option 1: Use Commercial Off-The-Shelf (COTS) instead

- Pros: Software up & running quickly; low (starting) cost
- Cons: Business differences; control of software

Option 2: Rebuild a whole new system

- Pros: built-for-purpose to meet current needs and business of the client
- Cons: high cost; long delay until available

Option 3: Modify Existing (the current system)

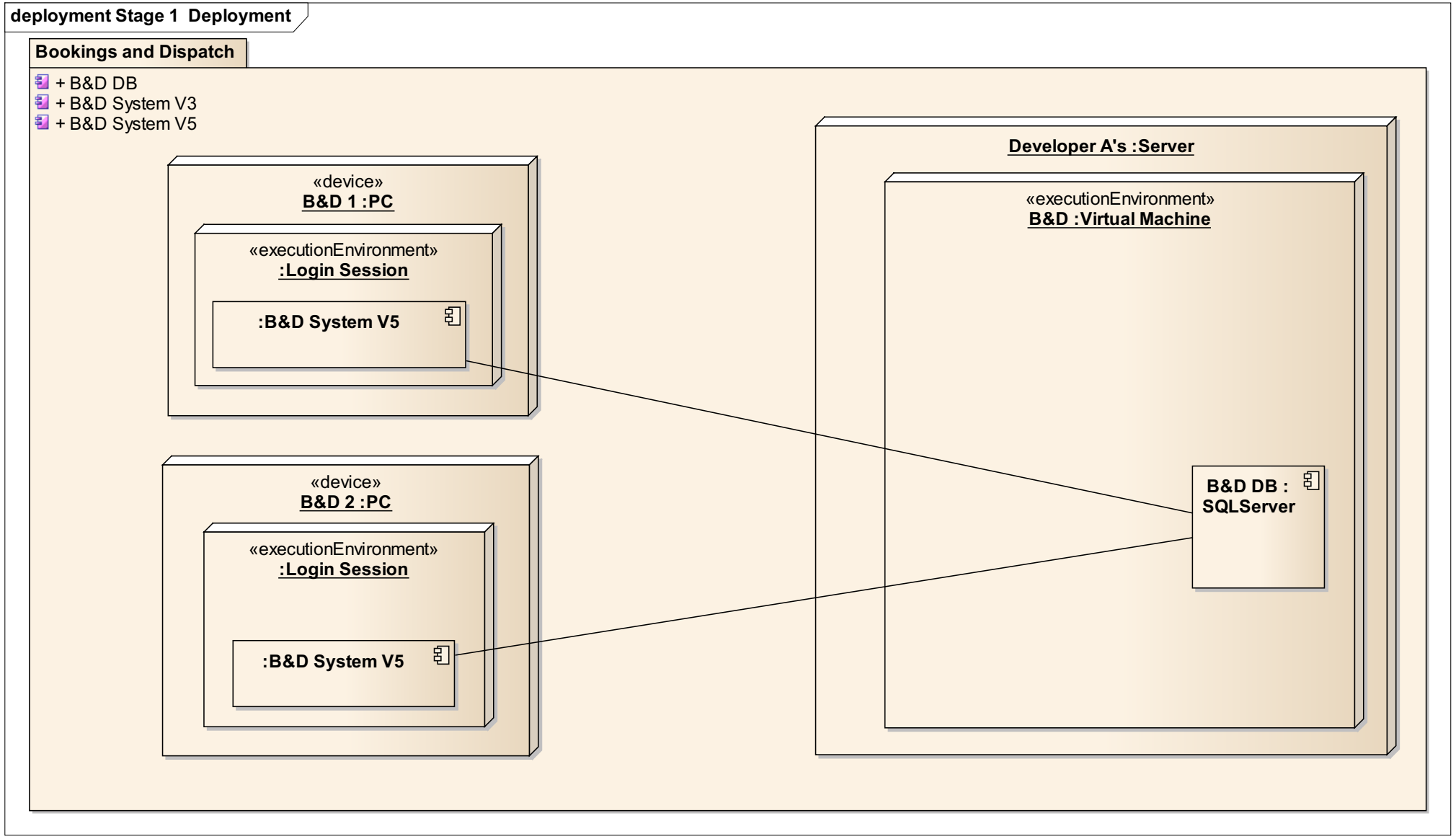
- Pros: business transition easier; control of software
- Cons: Need to find a willing developer; cost?; delay?

Planned Approach: “Take Control”

Selected Option: Modify existing → Developer identified

Issues in the current system	Solution Approach
Responsiveness • Remote Desktop over internet to remote server • Sharing of limited bandwidth: UI freezes	• Reduce the communication over the internet • Host the system locally instead
Reliability • Connection down → no system → paper/pen	• Update networking setup
Modifiability • Two versions of system: V3 and V5 and V3 uses an old technology (VB 6), but V5 is highly dependent.	• Eliminate V3 by implementing (V3-V5) in V5 • Make V5 accessible for updates
Functionality • Missing functionality, e.g., Invoicing and reporting Customer Portal (Online booking, Profile, …)	• Add high priority low complexity functionality (e.g., invoicing & reporting)

Improved System: Stage 1



Stage 1 Summary

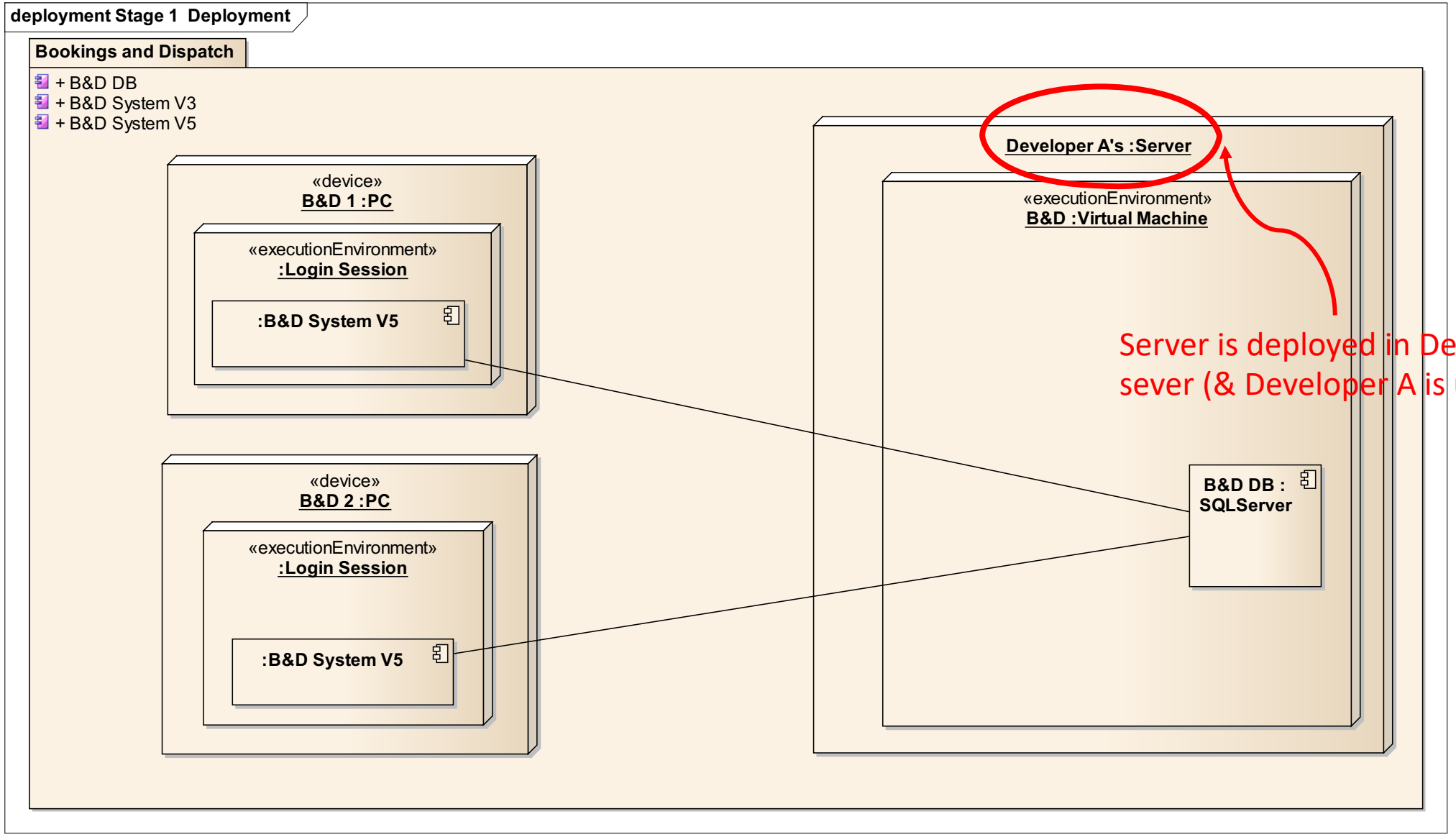
Stage 1:

- Add the functionalities that were only in V3 to V5
- Modify the local environment to support running the V5 Thick Client (the system runs locally at the client side)

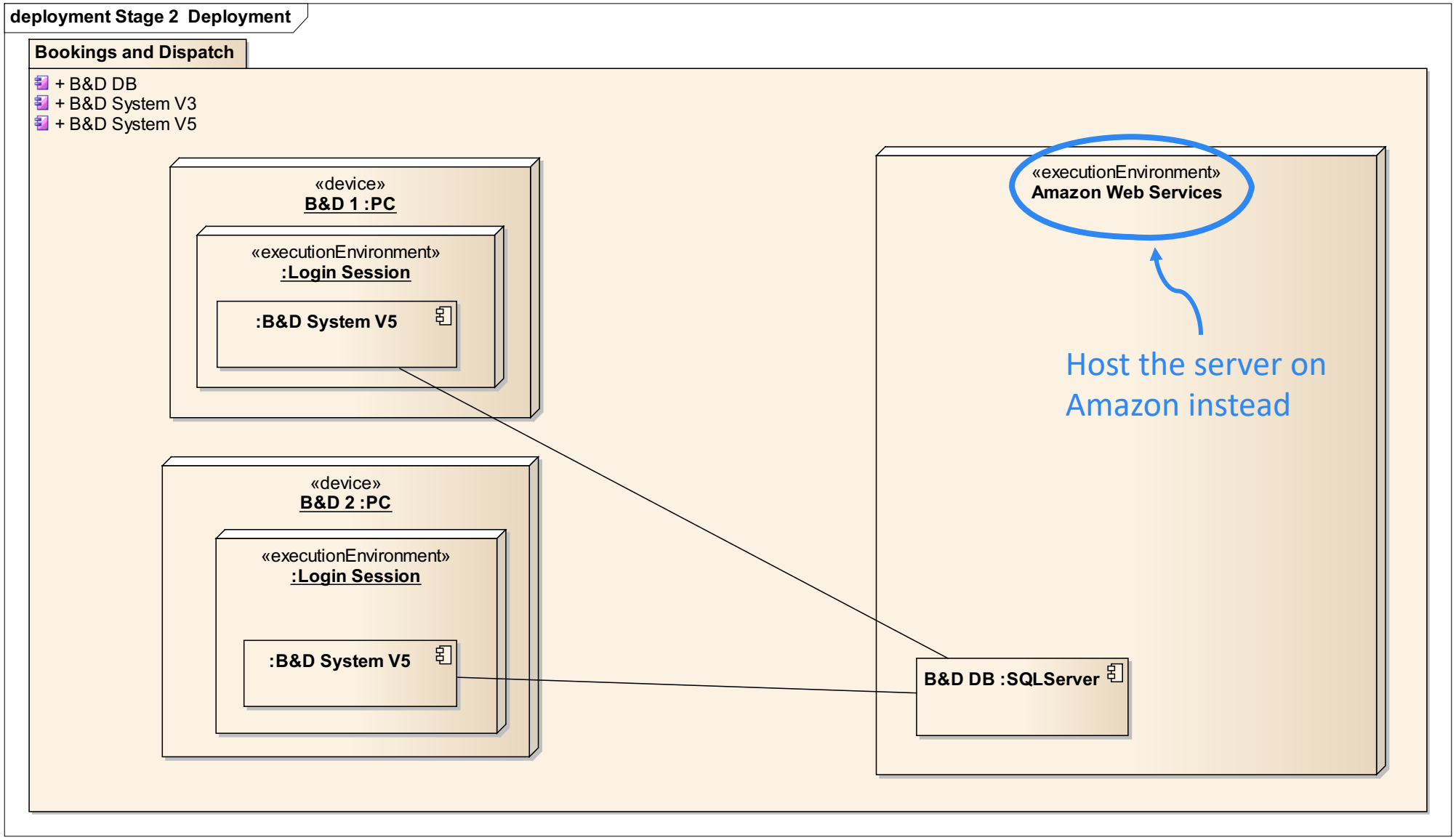
Benefits:

- No more Visual Basic (used by V3)
- No more running two apps (Reduce coupling, increasing maintainability)
- Running locally eliminates Client lag

Improved System: Stage 1 (Remaining issue)



Improved System: Stage 2



Stage 2 Summary

Stage 2:

- Migrate SQL Server from Developer A's server to Amazon

Benefits:

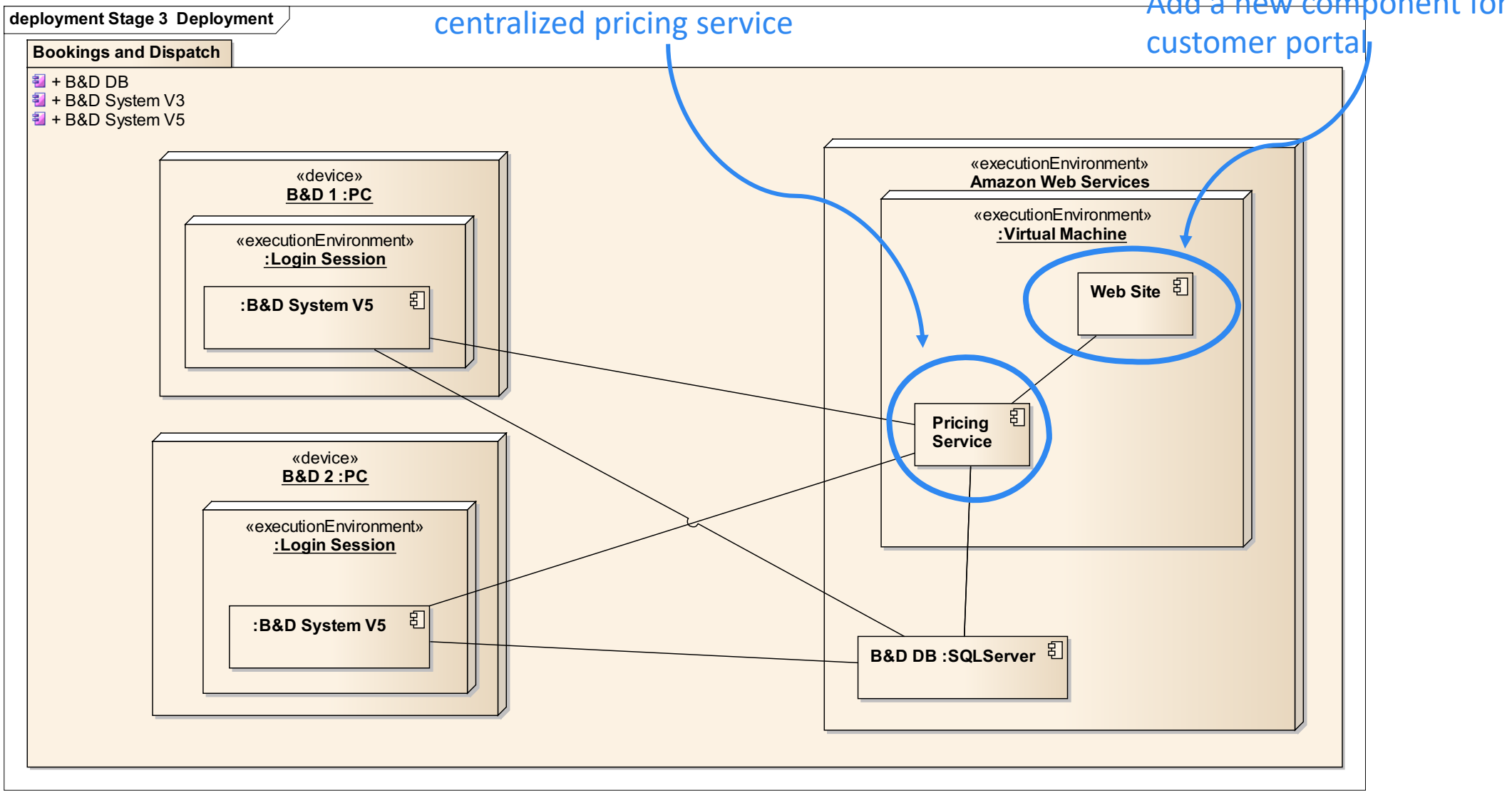
- No further dependence on Developer A
- Increased reliability
- Easy options for increasing backend performance
- Options for increasing redundancy
- Reduced maintenance cost

Stage 2: Remaining Issues

Client Problem Perspective

- Slow/hung, especially when customer on the phone *Addressed in Stage 1*
- When the system is unavailable, operators requiring paper/pen *Addressed in Stage 1*
- Missing functionality, e.g.
  - Invoicing and reporting *Addressed in Stage 1*
  - Customer Portal (Online booking, Profile, …)
- Unhelpful IT provider: Developer A *Addressed in Stage 2*
- Cost: significant overhead for status quo *Addressed in Stage 2*

Stage 3


Add a new component for centralized pricing service
Add a new component for customer portal
deployment Stage 3 Deployment
Bookings and Dispatch
+ B&D DB
+ B&D System V3
+ B&D System V5
«device»
B&D 1 :PC
«executionEnvironment»
:Login Session
:B&D System V5
«device»
B&D 2 :PC
«executionEnvironment»
:Login Session
:B&D System V5
«executionEnvironment»
Amazon Web Services
«executionEnvironment»
:Virtual Machine
Web Site
Pricing Service
B&D DB :SQLServer

Stage 3 Summary

Stage 3:

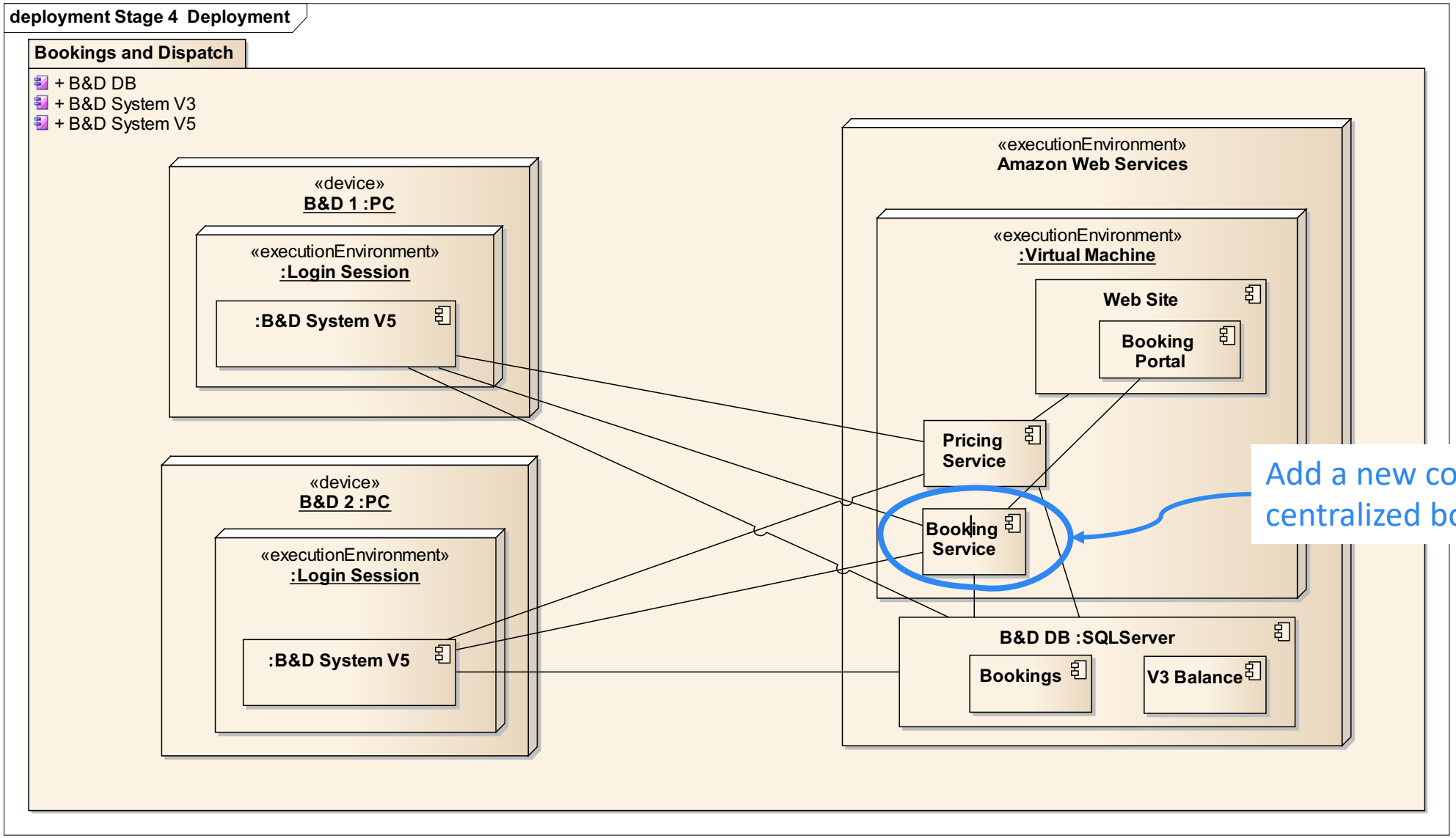
- Introduce pricing service
- Replace pricing in existing system with service
- Make pricing available to customers via web site

Benefits:

- Introduction of a service-based architecture
- Shared single location for services
- Improved transparency for customers
- Opportunities for server optimisation

Remaining issue:
Missing online booking functionality

Stage 4


deployment Stage 4 Deployment
Bookings and Dispatch
+ B&D DB
+ B&D System V3
+ B&D System V5
«device»
B&D 1 :PC
«executionEnvironment»
:Login Session
:B&D System V5
«device»
B&D 2 :PC
«executionEnvironment»
:Login Session
:B&D System V5
«executionEnvironment»
Amazon Web Services
«executionEnvironment»
:Virtual Machine
Web Site
Booking Portal
Pricing Service
Booking Service
B&D DB :SQLServer
Bookings
V3 Balance
Add a new component for centralized boking service

Stage 4 Summary

Stage 4:

- Introduce booking service
- Replace booking in existing system with service
- Make booking portal available to customers via web site
- Partition database to separate out bookings

Benefits:

- Customers can now make their own bookings (*critical*)
- Further develop service based architecture
- Start migration from V3 DB schema and configuration

Case Study Lessons

- Client's perspective is critical, however
  - Analysis is required to get the full picture
- Understanding of business requirements is critical
  - Missing key elements (e.g. workflow, interface requirements) can make the system unusable
- Knowing where you want to get to is the easy part
  - Planning and executing an acceptable path there is often the real challenge
  - Design helps by providing options

Lecture Identification

Lecturer: Patanamon Thongtanunam

Semester 2, 2020

These slides include materials from:

Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development, Third Edition, by Craig Larman, Pearson Education Inc., 2005.