Systems Engineering for a Restaurant Integrated Data System

Brian Bacon

ENPM643

Fall 2007

Concept

- Think Applebee's
- Integrated Data System to track dining orders
- How can technology enhance restaurant information operations?

Assumptions:

- This as an R&D product without any particular customer signed up yet.

Main Stakeholder Needs

- Restaurant Owner: Increase Profit
  - Increase Restaurant Revenue
    - Orders per restaurant hour
    - Orders per labor hour
  - Decrease Restaurant Expenses
    - Labor
    - Material (waste)
    - RDS Upkeep
  - Track data trends
- RDS* producer: Increase Profit
  - Increase Revenue
  - Decrease Expenses
    - Make RDS marketable to many restaurants

* RDS = **R**estaurant **D**ata **S**ystem

Other Stakeholders

- Dining Customer
  - Do not interfere with a pleasant dining experience (linked to "Increase restaurant revenue.")
  - Timeliness: Do not make me wait for my food. (linked to "Increase restaurant revenue.")
- Preparer/Busser/Server
  - Timeliness: Let me do my job faster (linked to "Decrease Restaurant Expenses")
- Host(ess)
  - Track the data I am responsible for
- Installer
  - Allow me to do my job correctly, quickly. (linked to "Decrease Producer Expenses")

Use Cases

- Customer Transaction
  - Seating
  - Ordering
  - Preparing
  - Serving
  - Paying
  - Cleaning Up
- Edit Internal Data
- Access Tracked Data
- Installation/Expansion

Actors

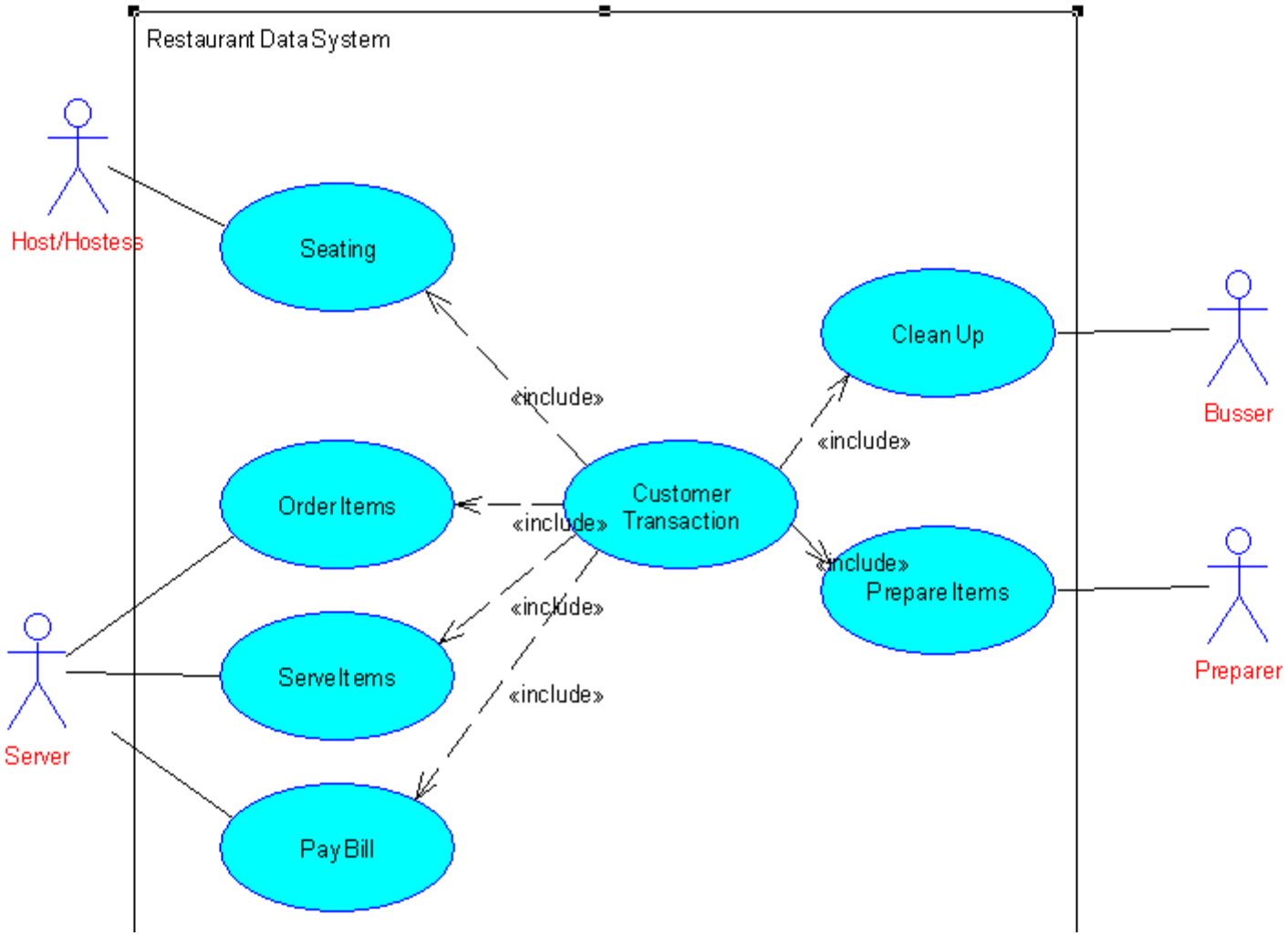
- Owner / General Manager
- Manager / Supervisor
- Preparer (Bartender or Cook)
- Server (Bartender or Waitstaff)
- Host(ess)
- Busser
- Installer

External System Interfaces

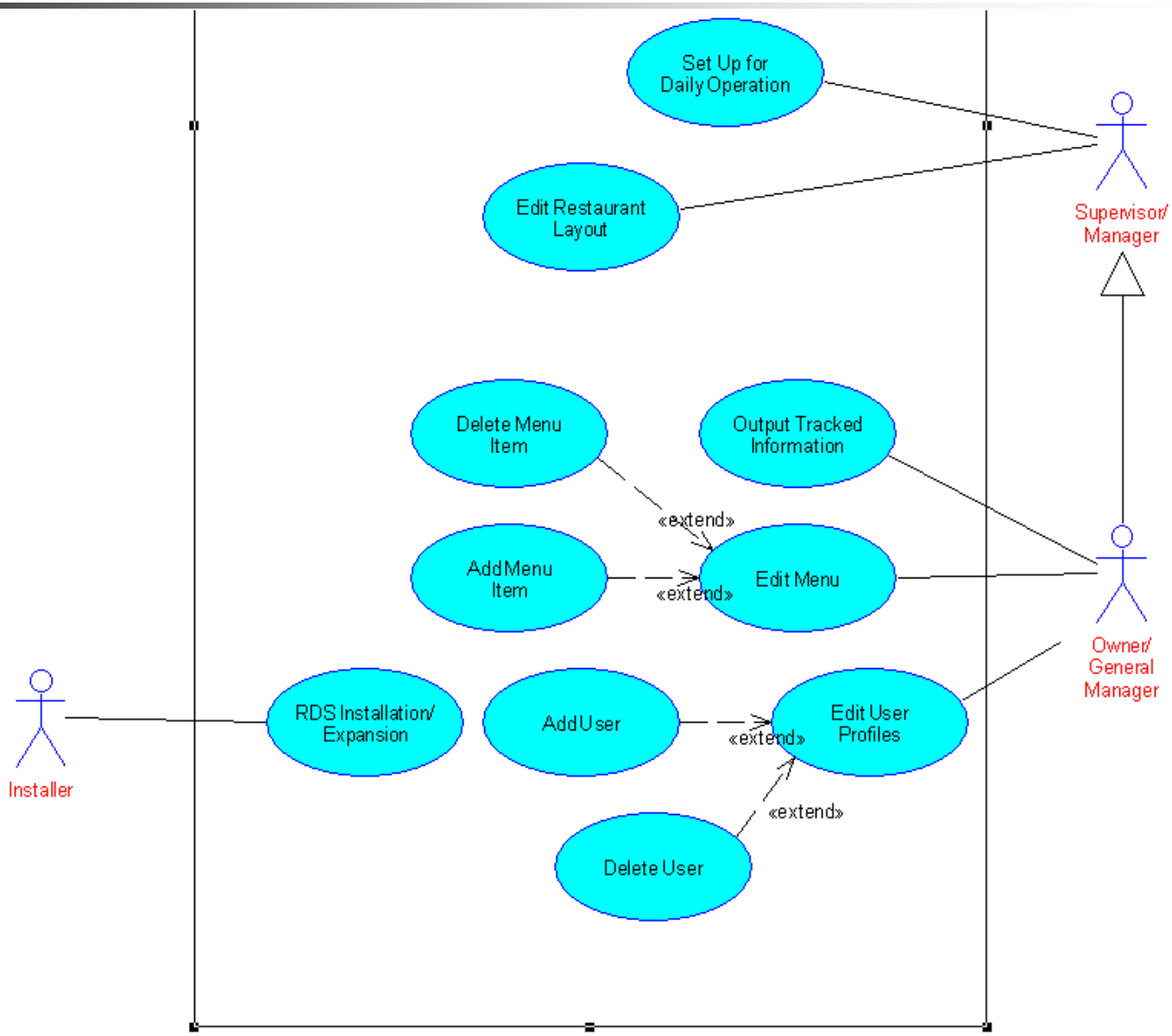
- Payment system (Credit Cards)
- Waiting area paging system
- Inventory ordering system

All can independently be manual or automated; both have benefits and drawbacks.

Use Case Diagram (1 of 2)



Use Case Diagram (2 of 2)



Requirements Determination and System Development Plan

- Partner with an existing restaurant
- Begin with observation and interviews to determine some basic requirements.
  - Make basic design choices correctly
- Proceed with rapid spiral development
  - Produce a system based on basic requirements with the most important elements present.
  - Receive feedback on system
  - Incorporate feedback and add more functionality.
  - Iterate until a releasable product is attained.
- Slow the spiral for mature product iteration & support

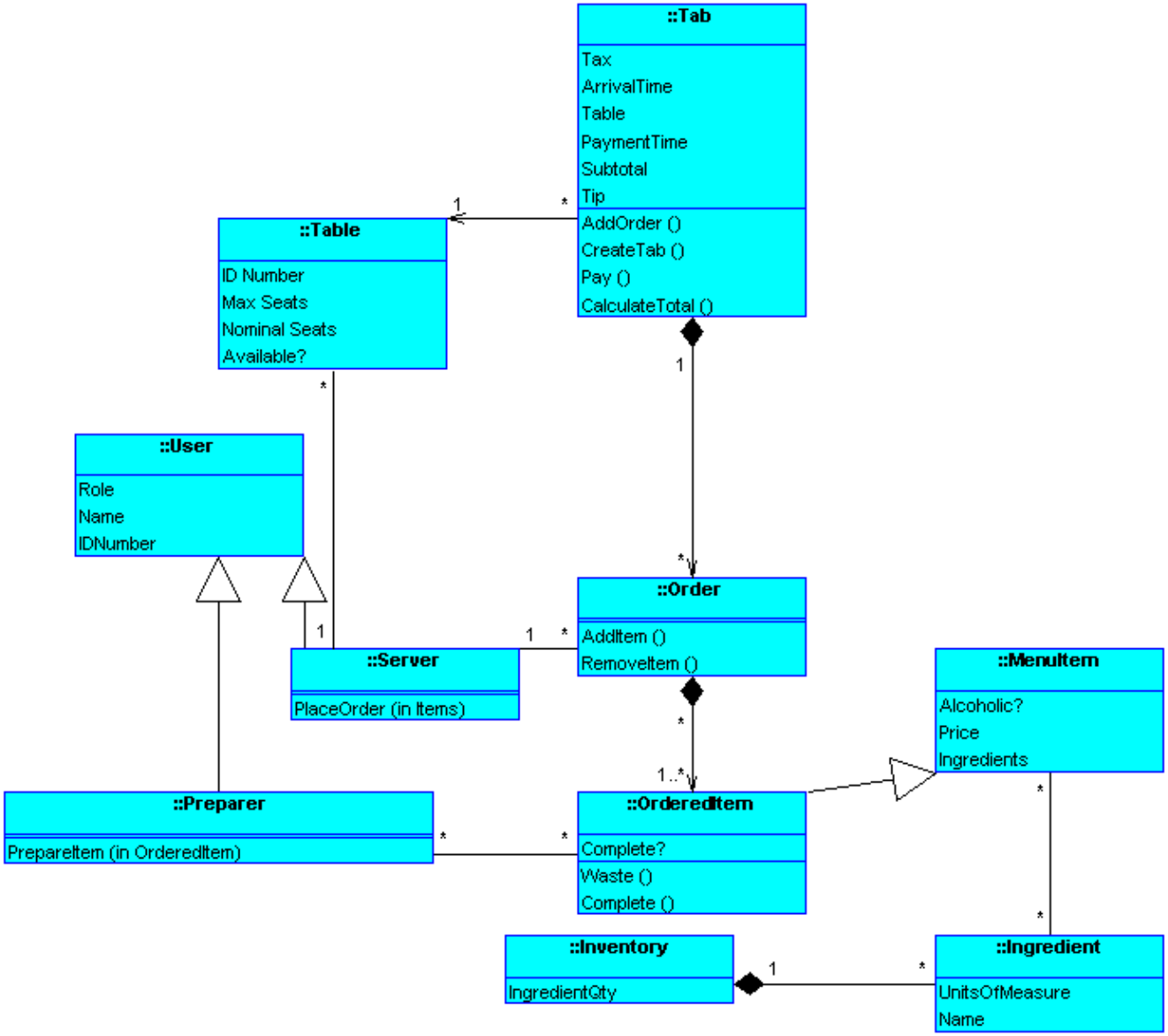
Key Requirements

- Human Factors
  - Interface simplicity
  - Interaction time
- Environmental Withstand
  - Dining Room
  - Kitchen
- Security
- Scalability
- Cost
  - We are late movers in the market – must be cost competitive.

Verification Plan

- Expect many requirements to be verified by demonstration *in situ*
- The system also lends itself to simulation
  - Useful for "shall not" requirements
  - Also good after product maturity

Class Diagram



Trade Studies

Centralized	Distributed
Core components protected from harsh env	Easily scalable & expandable
Interface stations smaller	Self healing
Upgrades easier	Maybe less latency

- Also External Interfaces: Automatic vs. Manual
  - Functionality & Time vs. Simplicity

Path Forward

- Create Verification & Traceability Matrices for requirements
- Enhance current Use Case descriptions & diagram
  - External Systems
- Complete Class Diagram
  - Attributes & Operations
  - Audit classes' relationships
- Expand Distributed vs. Centralized trade study

Questions?