MICHIGAN STATE UNIVERSITY

2. Technical Specifications

CSE 498, Collaborative Design

Wayne Dyksen
Brian Loomis

Department of Computer Science and Engineering
Michigan State University

Spring 2006

Overview

- What is architecture?
  - How do I get started on the project?
- Case study driven discussion
- Thinking like an architect
  - How to break it down

2-2

What is programming?

- Programming is both the act of applying a logical approach to solving a problem and writing that logic down in a form that the computer understands
  - Programs solve problems and are tools, expressing our imagination about the problem
- Writing programs is more like writing mathematical proofs than building a house
  - Each developer has a toolbox of constructs
  - Apply the constructs iteratively in steps to create a solution to the problem
  - Each program is a unique solution but not necessarily the only one

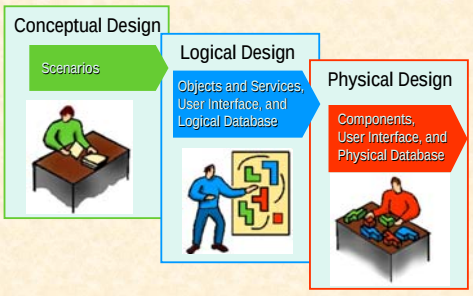
2-3

In the early part of a project…

- Requirements gathering and understanding
  - State the problem unambiguously
- Architecture and design
  - Determine your approach to the problem
- First working prototype
  - Test your hypothesis
- Feature complete build
  - Formalize the proof
- Ship it!
  - Turn it in!

2-4

Design Process Overview



2-5

Starting with something like…

Channel Vantage: Web-Based Data Entry Tool for General Motors

- General Motors is dedicated to developing and maintaining strategies for each dealership located in the United States. Such strategies contain information concerning opening, closing, relocating or even realigning dealerships. To help facilitate General Motors with the development of these plans, for over 7,000 different dealer sites, ChannelVantage developed a Microsoft Access application for data entry. This application allowed General Motors personnel to quickly develop plans using pre-defined and legally approved text. One problem with this approach is that much of the current interface is cluttered with buttons and controls that are used to develop standardized text. When General Motors personnel have finalized strategies they wish to publish, there is a multi-step manual process that must be performed by ChannelVantage to get those plans uploaded into a GM mainframe.

2-6

Understanding requirements

- Capture the end-user or project sponsor's intent
- Starting with…
  - Often no formal docs, maybe only a shared vision
  - Maybe an incomplete problem statement
- Approach…
  - Talk with your customer a lot!
  - Identify business entities (objects) and relationships
  - Remove ambiguity and logical/business inconsistencies
  - Validate business rules and assumptions
- Ending with…
  - Formal phrasing of requirements in a document
    - Defined scope of what is in and not in the solution (boundary conditions and features not included)
    - User scenarios or experiences
  - Validated initial schedule, cost, and risk (things not yet known)

2-7

Possibly a vision document

Content		Purpose
Problem statement	→	Why you want to do it
Vision statement	→	What you want the product to be
Solution concept	→	What you will do
User profiles	→	Who will use the product
Business goals	→	What you want to accomplish
Design goals	→	How you plan to accomplish it

2-8

Another example: TechSmith

- "We would like to create a website that allows users to index and "edit" videos. Users of the website should be able to tag and bookmark specific points or segments of a video. They should also be able to create a video out of multiple video segments in a non-destructive manner."

- The scope of this project might include *what*?
- What might be *nice* to do?
- What is definitely *out of scope*?

2-9

Architecture and design

- Start translating the requirements into a plan and logical design that can be implemented as a program to solve the problem

- Starting with…
  - Requirements and user scenarios
- Approach…
  - Break a big problem into lots of little problems
  - Identify all the moving parts and interactions
- Ending with…
  - **Technical (or functional) specification**
    - Architecture of solution
      - User interface mockup
      - Interfaces to other systems or data formats
      - Entity/object model for system (pseudo-code for business data rules and functions)
      - Data schema
    - Identification of core feature set for the prototype
  - Test plan, schedule, risk analysis

2-10

Contents of a technical spec

Content		Purpose
Vision summary	→	What you want the product to be, justification for it, and key high-level constraints
Design goals	→	What you want to achieve with the product
Requirements	→	What you require from the product including "non-functional" requirements like reliability, scalability, security, etc.
Usage summary	→	When the product will be used and who will use it
Features	→	What exactly the product does, user interface mockup, event models, object diagrams (and use cases), data schema
Dependencies	→	Other factors the product depends on (external interfaces and compatibility)
Schedule	→	Key dates and deliverables
Issues	→	What risks might impact the project
Appendixes	→	Network topology, deployment plans, dev environment setup

2-11

Examples

- Quantum, Empowernet
- Last semesters…

- Every good spec is different but similar
  - Make sure you are complete
    - All tiers of the architecture – database, business objects, user interface
    - Topology
    - Standards
  - Make sure you use two models at least
    - Network diagram and class descriptions
    - Use cases
  - Ask yourself if you could be the tester on this system

2-12

Architecture constraints

- Communication
  - Speed: Ethernet, GigE, 802.11b/g, or dialup
  - Protocol: TCP/IP, IrDA, POTS
- Topology
  - One machine versus multiple interacting
  - External systems
- CPU speed
  - PDA, Itanium server, mainframe
- Memory availability
- Device-specific parameters
  - PDA display size or ink on TabletPC
- Legacy support or previous versions of the current app

2-13

Architecture tradeoffs

- Complexity
  - Number of technologies in use
  - Design patterns vs. execution speed
  - Number of tiers or subsystems
- Fully-custom, semi-custom, or off-the-shelf
  - Platform (OS, servers, SDKs, ++)
  - Language and compiler choice
  - Project type choice
- Appropriate technology
  - Reusable modules
  - Special-purpose languages
  - Community support
- Tools and process
  - How automated a process do you need?
  - How do you communicate designs? (UML, ORM, etc.)

2-14

A Basic OO Modeling Process

1. Identify logical entities (the “things” and their operations) from the requirements statements
2. Make these into variables with specific data types
3. Identify starting point for problem (what do I want to solve?)
4. Determine what entities and logical operations you need from this starting point and relationships between objects
5. For each logical operation, determine inputs and outputs and types of looping constructs (pseudo code)
6. Analyze entities for non-functional requirements – exceptions, security… per coding standards

2-15

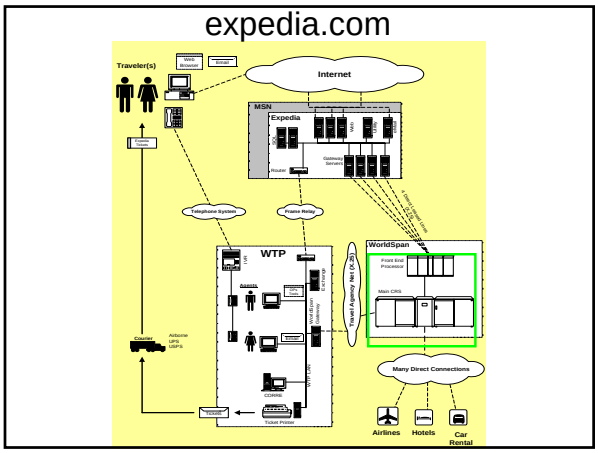
Bigger examples: Google

2-16

Summary

- What we covered
  - Identify what you don’t know
  - Get to the spec quickly and completely
  - Learn what questions to ask early on in your projects
  - Remember, “every system was built by mortals”
- Resources
  - Functional specs samples are online
  - Look at your team’s assignment and figure out what you don’t know
  - Start communication with your customer
  - When is your spec due?

2-18


expedia.com
Traveler(s)
Internet
MSN
Expedia
Telephone System
Frame Relay
WTP
WorldSpan
Travel Agency Net (X.25)
Many Direct Connections
Airlines
Hotels
Car Rental