Discussion Slides

Deeper

- List a few different types of descriptive models?
  - *For the most part, all UML behavioral and structural models are descriptive.*
- What is the difference between a descriptive model for a user and a descriptive model for a developer?
  - *User models describe the system's external appearance and function. Developer models describe internal structure and operation.*

Deeper

- Give examples of requirements questions that could be answered by analytical modeling?
  - *Task structure and user interface questions are often best answered by prototype usability questions.*
  - *Estimates of required performance can often be modeled by simple mathematical modeling.*

Deeper

- What is the difference between a product mock-up and an engineering proof of concept?
  - *The purpose of a product mock-up is to show potential supporters/customers what the product would look like.*
  - *The purpose of an engineering proof-of-concept is to demonstrate that we have a viable solution to what has here-to-fore been considered to be a hard problem.*

Deeper

- Give examples of a consumer product or service mock-up.
  - *A hand-made prototype of a new personal music player.*
  - *Simulated content and interactions with a new web service.*
  - *A Tivo-like product implemented on a work-station rather than in a custom appliance.*

Deeper

- What is the wisdom behind modeling with a purpose?
  - *If you are very clear about your goals:*
    - *it is more likely that they have been well refined.*
    - *you are likely to chart a more direct path to them.*
    - *you are less likely to be distracted by tangential issues.*
    - *you will be more likely to recognize whether or not you have achieved them.*
  - *If you're modeling without clear goals, you may spend lots of time building the wrong things.*

Deeper

- Examples of descriptive model purpose?
  - *Management … which pieces need to be built*
  - *Development … what each component does*
  - *Support … how components will be deployed*
  - *Training … how to use the system*
- Examples of analytical model purpose?
  - *Management … to size a task*
  - *Development … to validate an approach*
  - *Support … how system must be configured*

Deeper

- What is the wisdom behind <u>traveling light</u>?
  - *Ongoing maintenance is a killer, and models are no exception. It takes work to keep models up-to-date with changing requirements and design.*
  - *The fewer things you decide to maintain, the less time you will spend maintaining them.*
  - *Discarding a model after it has served its purpose is a good thing.*
  - *Relatively few models are worth keeping.*

Deeper

- Traveling light for descriptive models?
  - *Figure out which design models will be the most useful, and then build only those.*
  - *Understand which models will be superceded by the code, and which (few) will be kept for training purposes.*
- Traveling light for analytical models?
  - *Precisely understand the key question, and model only what is necessary to answer it.*
  - *Understand which few models will have ongoing value.*

Deeper

- What is the wisdom behind using <u>multiple models</u>?
  - *This is where "one size fits all" meets "divide and conquer".*
  - *Smaller and simpler models that attempt to directly address particular problems will generally do so more effectively.*
  - *Moreover, it will often be the case that N smaller models can be built much more quickly and easily than a single super-model to address the same issues.*

Deeper

- Interpret <u>multiple</u> for <u>descriptive</u> models?
  - *Use multiple types of models (behavioral, structural) to illustrate different aspects of the system.*
  - *Separate models for independent subsystems will be simpler and easier to understand.*
- Interpret <u>multiple</u> for <u>analytical</u> models?
  - *Don't try to get one model to answer all your questions. Build different kinds of models to answer different kinds of questions.*

Deeper

- What is the wisdom behind putting <u>content</u> over <u>representation</u>?
  - *Standard representations are seldom requirements to be imposed on projects. They are tools that have been developed to help solve particular problems.*
  - *If a standard representation doesn't quite meet your needs, or if it seems to heavy for your problem, you are usually free to improvise a more appropriate representation.*

Deeper

- What does content over form mean for descriptive models?
  - *Don't worry about strict adherence to a particular modeling language. Represent your system in any way that works.*
- What does content over form mean for analytical models?
  - *Don't feel obligated to use a more formal or powerful modeling technique if a simpler one will (reliably) give you the answer you need.*

Deeper

- Why is a graphical design language interesting?
  - *Any language can be evaluated on the basis of range, ease of expression and precision of expression.*
  - *While graphical languages are (in general) less precise and powerful than textual ones, the human mind can often absorb structures and relationships much more easily and quickly from a visual representation than a textual one.*

Deeper

- How is UML different from other graphical flow-charting & design systems of the last 50 years?
  - *It supports precise and complete high level descriptions of objects and their relationships.*
  - *Its wide range of sub-languages permit meaningful description of many more aspects of system behavior and structure.*
  - *The sub-languages are related in ways that enable easy integration of multiple models.*

Deeper

- Why is it useful to standardize a graphical design language?
  - *People can more quickly understand a language they already know.*
  - *Designs captured in a standardized language can be more easily shared and reused.*
  - *A standardized language can express much more nuanced concepts (compare AMSLAN with charades).*
  - *A large population will attract tool developers.*

Deeper

- What distinguishes a sketching language from a specification language?
  - *A sketching language can quickly and easily describe general concepts.*
  - *A specification language can precisely describe objects and their behavior.*
- What distinguishes a specification language from a programming language?
  - *A programming language can completely specify data types and flow control.*

Deeper

- How does a textual representation enable the development of CAD tools?
  - *A standard textual representation permits a design to be easily passed between tools.*
  - *A graphical design tool can output the textual representation for the created program.*
  - *Browsers, type-checkers, interpreters and other tools can parse the textual representation as a means of importing the model on which they will.*

Deeper

- If the average service rate (μ) is faster than the average request arrival rate (λ), why would there ever be a queue?

  Requests can arrive sooner or later than the average arrival rate, and service times can be shorter or longer than the average service time. During a burst in requests, or a time of requests that require more time to service, a queue should be expected to form.

Deeper

- What can we do if a waiting time is predicted to be unreasonably long?
  1. *add more servers to the queue.*
  2. *make the servers faster.*
  3. *reduce the number of requests*

Deeper

- What can we do if a Markov model tells us we will spend too much of our time in an unavailable state?
  1. *speed up the repair time for that state.*
  2. *reduce the number of incidents that lead to that state.*
  3. *change the design so that the system can continue to provide service while in that state.*

Deeper

- Why might we prefer to test a component like a scheduler in a simulator (vs. in a real system)?

  A real system might not be available (yet).

  A simulator can force decisions through the scheduler <u>many</u> orders of magnitude faster than a real system would, enabling us to collect data against a wider range of traffic patterns in a shorter period of time.

Deeper

- Why might random traffic be better than sampled traffic?

  One sample may not be representative of all likely usage scenarios. Random traffic may provide fuller exercise.

- Why might captured traffic be better than randomly generated traffic?

  Real traffic isn't random. There are patterns to requests, and better algorithms often exploit knowledge of these patterns.

Deeper

- How do you decide what your "proof of concept" prototype should do?

  What doubts that need to be assuaged?

  What demonstration would clearly assuage them?