Lecture 11
Communicating the Requirements
Data and State Models

Com S/SE 4090/5090

Robyn Lutz

rlutz@iastate.edu

Office hours:

Robyn: 11 Tues, 230 Atanasoff; 3 Fri (https://iastate.zoom.us/my/rlutz), & by appointment

Carter: cwunsch@iastate.edu,12 Tues (0112 Pearson); 12 Thurs (https://iastate.webex.com/meet/cwunsch), & by appointment

Arushi: arushi17@iastate.edu, 1 Wed (0112 Pearson) & by appointment

HW 2 due Thurs, Oct. 3

Exam 1 Thurs, Oct. 10; Review 10/8

Reading

Chap. 10 – “Data, Your Secret Weapon”

Chap. 17 – “Define the Business Data”

"Data: your secret weapon" (Chap. 10)

Lecture also based on [Weigers & Beatty, Software Requirements (Developer Best Practices), 2013]

We specify software functionality to create, modify, delete, display, process & use data

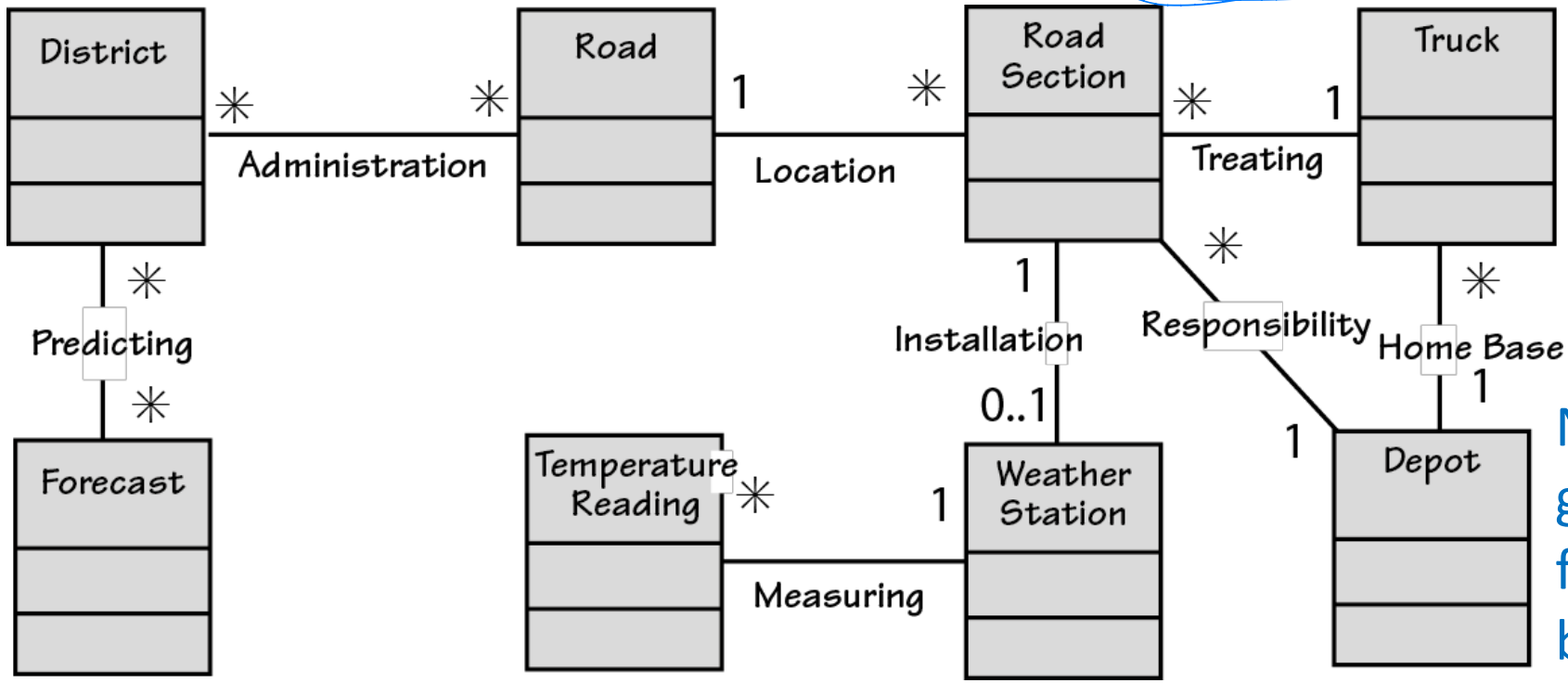
Data model: "A model showing classes of data and the associations between them." (Glossary in textbook)

At requirements phase, often represented by UML class diagrams

- high-level here (Robertsons specify the class names & associations, omit attributes and behaviors that an object of the class can perform)
- visual representation
- logical links among classes & cardinalities of those links

Example 1

IceBreaker class diagram of stored data



1 truck treats many road sections

Name on link gives reason for association between classes

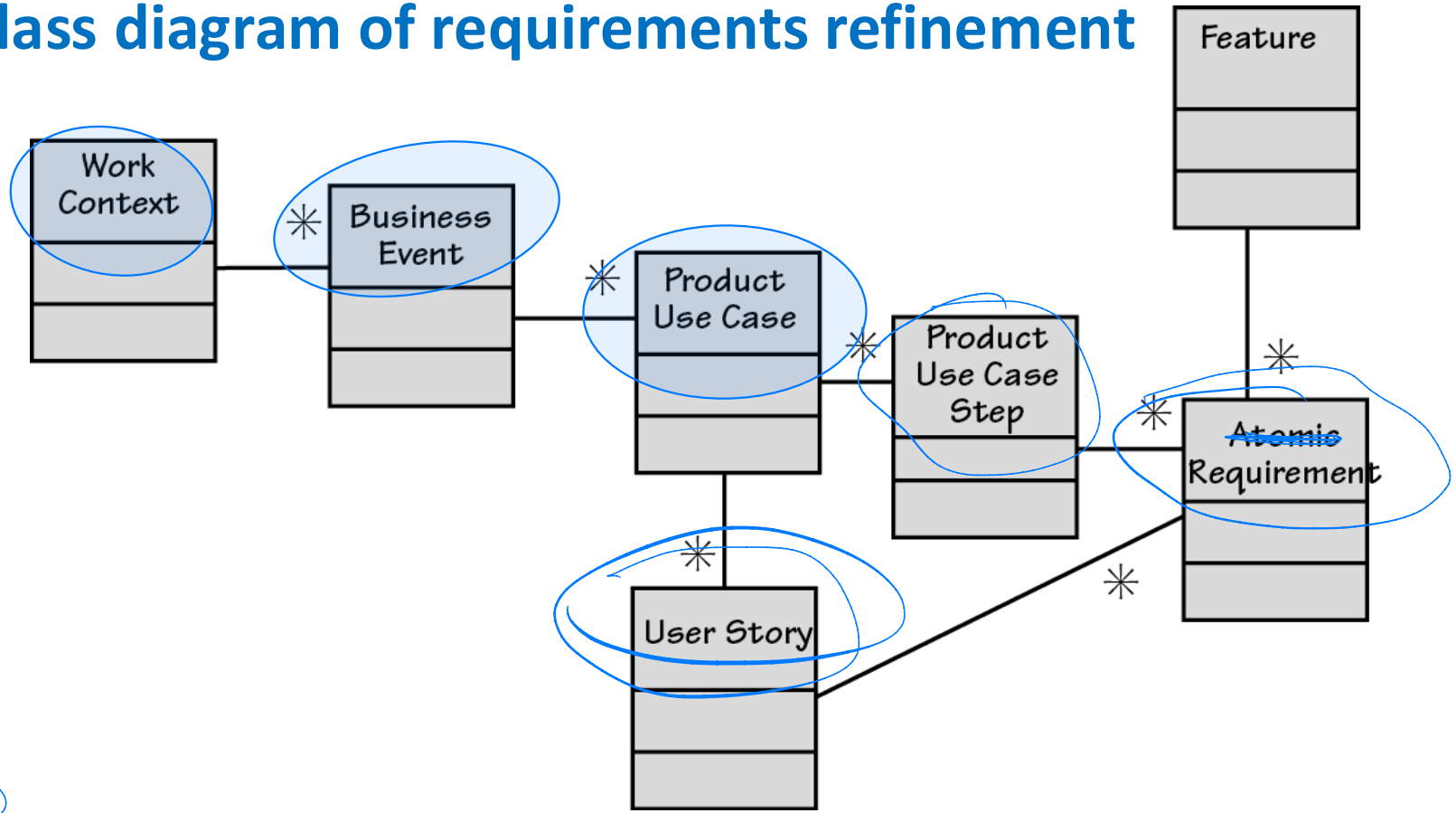
Figure 10.4

A class diagram showing the stored data that the business needs to predict the formation of ice on roads. A truck is dispatched to treat a road when the readings from the weather station and the forecast indicate the road section is about to freeze.

(Blue notes are Robyn's)

Example 2

Class diagram of requirements refinement



A feature is a unit of functionality that provides some service to users

Figure 10.6

A hierarchy of requirements. The work context is the highest-level statement of requirements; it is decomposed into the next level, the business events.

The level below the business events comprises the product use cases, each of which is decomposed into a number of product use case steps.

The lowest level includes the atomic requirements, each of which can be traced back up the hierarchy.

Activity diagrams and user stories are also used to group atomic requirements.

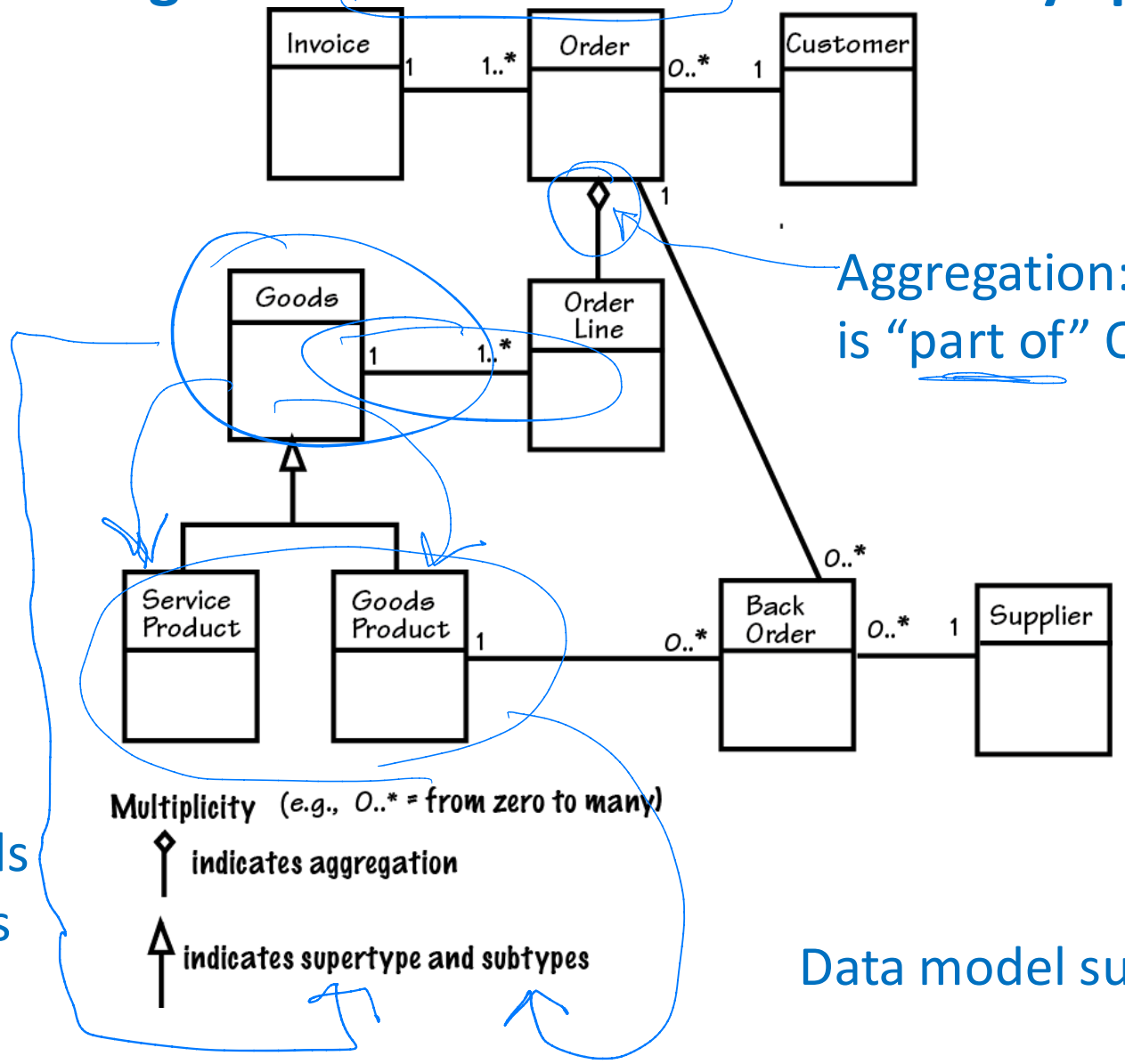
Features are a grouping often used by stakeholders from marketing or product version planning.

Class diagram of stored data for “user buys product”

Example 3

Figure 15.6

This class diagram shows the classes and associations between them that are part of the pattern Customer Wants to Buy Goods. Consider the business rules communicated by this diagram. A Customer may make zero or many Orders, each of which is invoiced. The Order is for a collection of Order Lines. An Order Line is for Goods, which might be a Service Product or a Goods Product. Only Goods Products can have a Back Order. Now consider how many situations in which these business rules, data, and processes might be reused.



Aggregation: Order Line is “part of” Order

Supertype/Subtype:
Service Product “is-a” Goods
Goods Product “is-a” Goods

Data model supports reuse

Class model of stored data

Class: “a collection of elementary data items (attributes) for something that is important to the business” (needed by the work)

Classes are things used/stored by the business: nouns

Ex: customer, employee

- object you sell (cars, flights, etc.)
- service you provide (mortages, contracts)
- account
- invoice
- credit card
- adjacent system (from the context diagram)

Class [glossary]: “physical or abstract entity within a context of study that has one or more attributes of stored data”

CRUD check finds missing requirements

avoid incomplete requirements

Each data class must be:

- Created [by a use case]
- Referenced [by a use case, after being created]
  - Referenced but not created -> requirement is missing
    - Ex: class Depot

Some data classes must also be:

- Updated
- Deleted

Trace CRUD actions to software requirements

Ex: CRUD check for missing requirements [Robertsons]

Table 17.2. The CRUD Table

Each cell shows the identifier of the business event that creates, references, updates, or deletes the entity. Gaps in the table indicate missing events.

Class	Create	Reference	Update	Delete
Depot		7		
District		2, 8, 10		
Forecast	2	8, 10		
Road	3	4, 8, 9, 10	3	
Road Section	3	4, 8, 10, 11	3, 9	
Temperature Reading	1	6, 8, 10		
Truck	7	8, 9, 11	7, 10	
Weather Station	4	1, 6	5	

Data modeling

use the client's vocabulary

Converge on a common terminology: Glossary

Start with input/output flows on context diagram

often created in the design phase

Data dictionary: "specification (to the elemental level) of the terms used in the requirements specification."

- "maintaining a data dictionary is a serious investment in quality" [Wiegers & Beatty]

Data *flow*: shows data that move from one process to another, usually represented by named arrows. Ex: Fig. 10.9

State modeling (not in textbook)

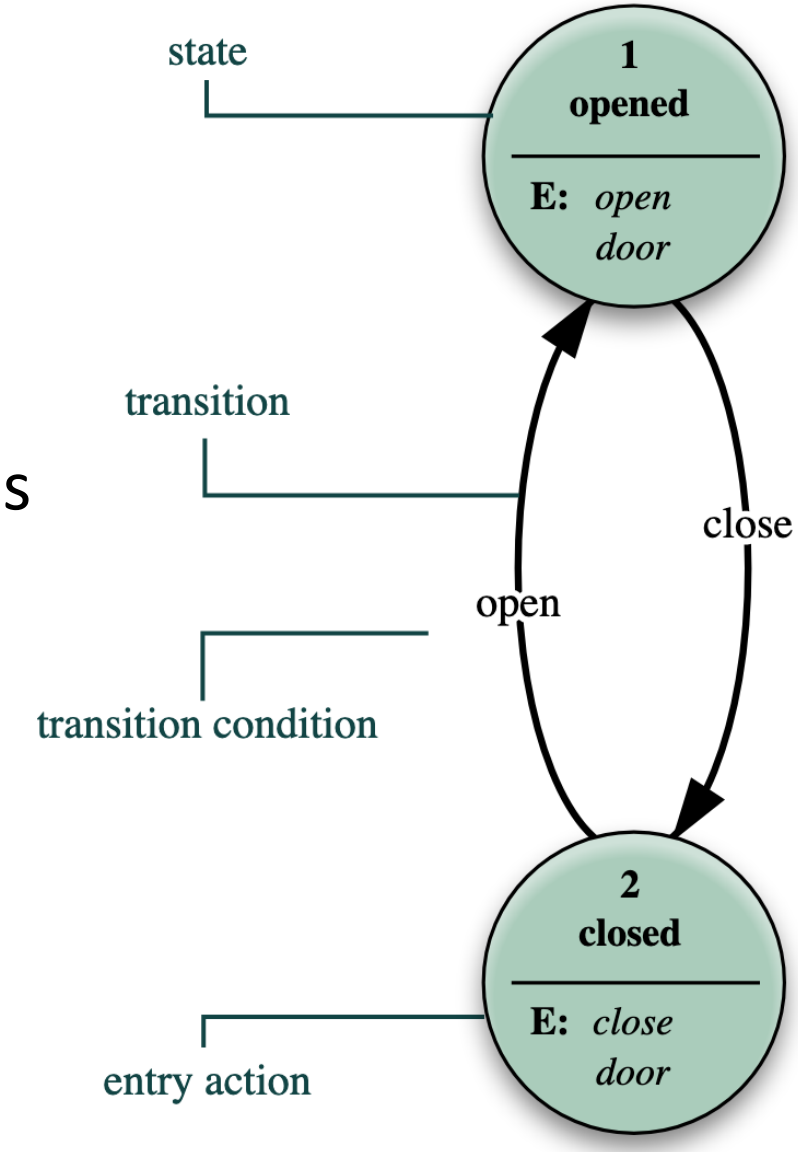
- Widely used for real-time systems where stimuli from outside the system drive its behavior
  - Ex: stimulus from motion sensor may move room_state from “empty” to “occupied”
  - Ex: operator’s command may move a valve from “open” state to “closed” state
  - Ex: completion of a task may move the system from “processing” to “waiting”
- State machine model captures the behavior of system components in response to internal or external events
  - Behavior is a sequence of state transitions
  - Model shows how system (or component) changes its state in response to events
  - Event triggers transition from one state to another

Example 1: State model

State diagram for a door:

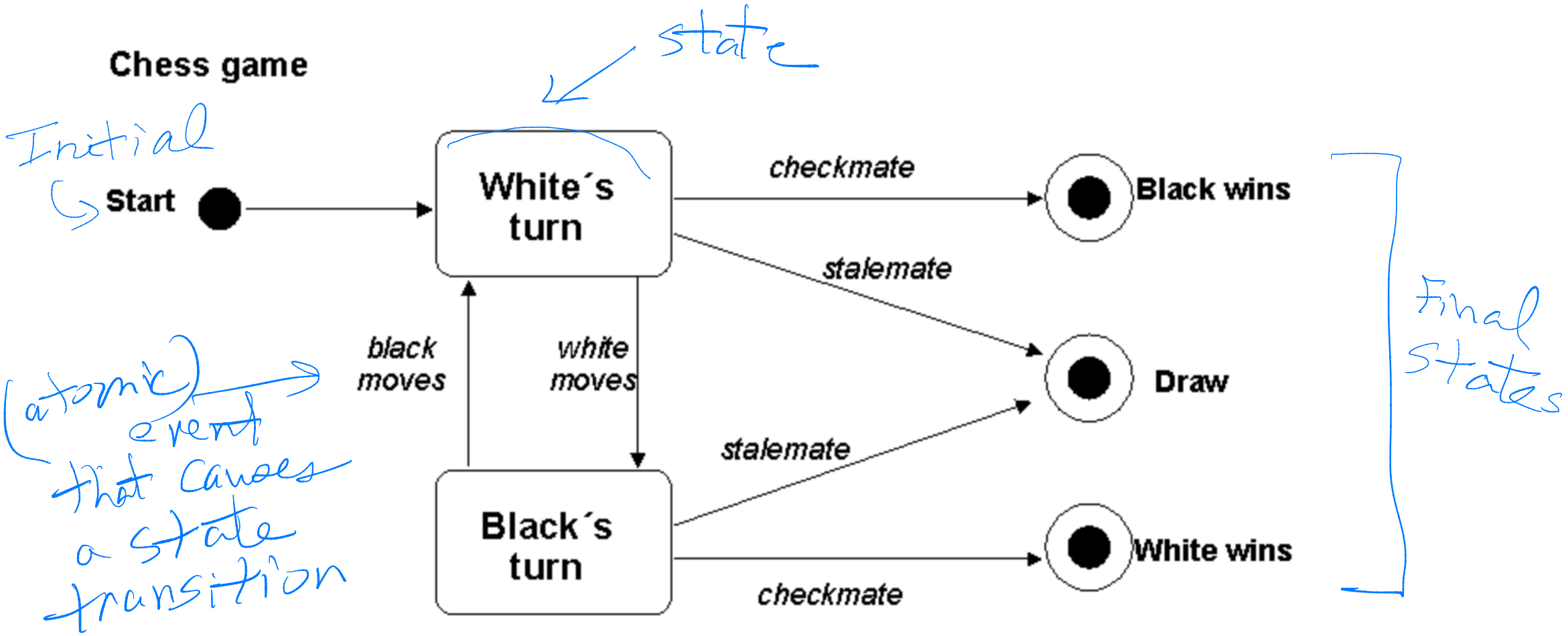
- Nodes are possible states
- Arrows are possible state-changes (transitions)

Visual representation of states & transitions prompts customer to review the requirements & spot gaps/mistakes in the system behavior shown



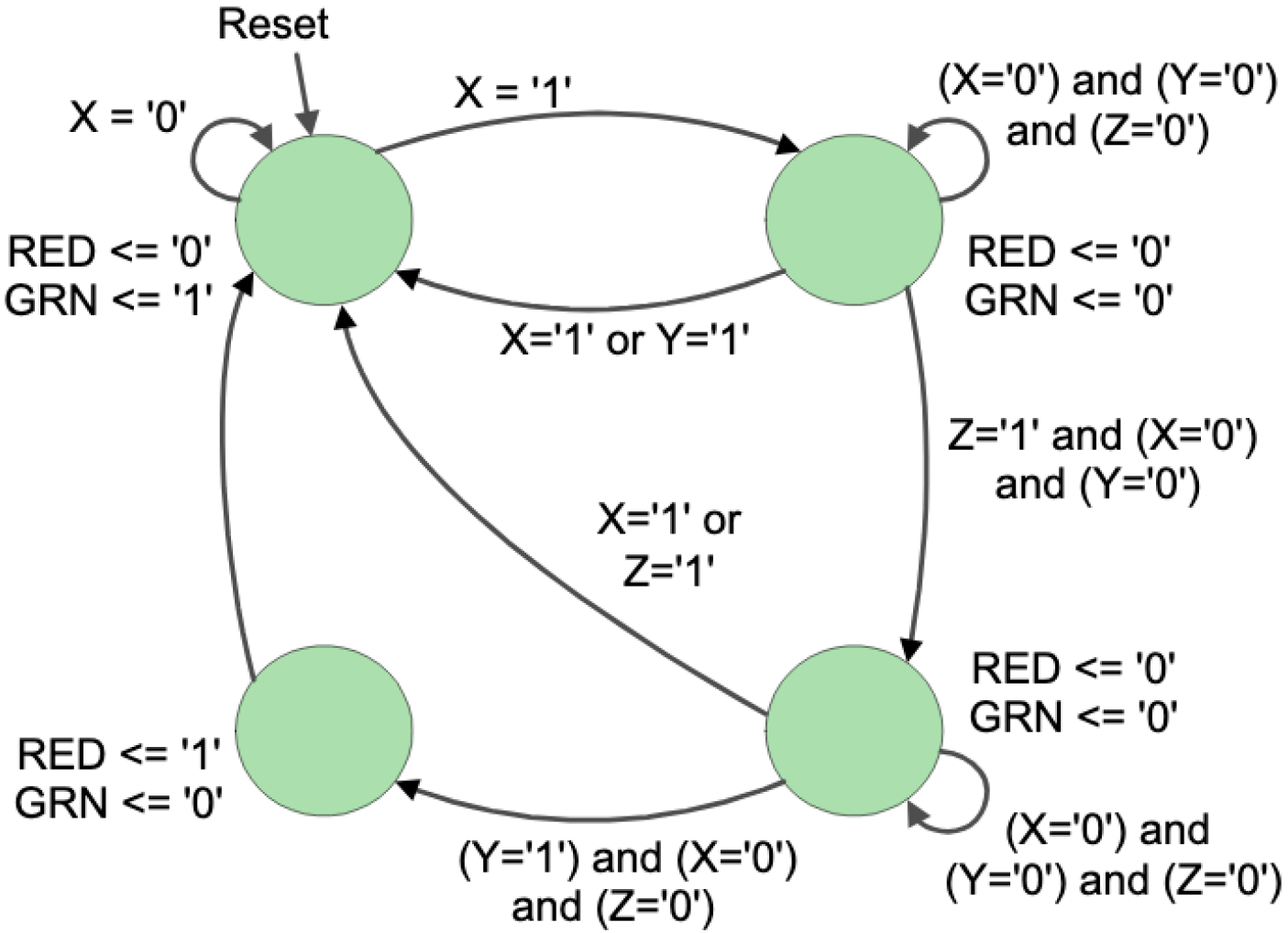
By 1st Macguy314, reworked by Perhelion / German translation by Babakus - Own work based on: File:Finite state machine example with comments.gif, Public Domain, https://commons.wikimedia.org/w/index.php?curid=2614970

Example 2: State model for a process



https://users.csc.calpoly.edu/~jdalbey/SWE/Design/STDexamples.html

Example 3: State model for a system



Receives input from 3 buttons labeled X, Y, and Z
Outputs 2 signals called RED and GRN
RED iff the correct 3-button sequence X-Z-Y
is detected
GRN when a new sequence starts.
(Doesn't show state names here)

https://www.realdigital.org/doc/fc26cf6e35d2a61c6e2871dd9be9e21a