lecture 8:
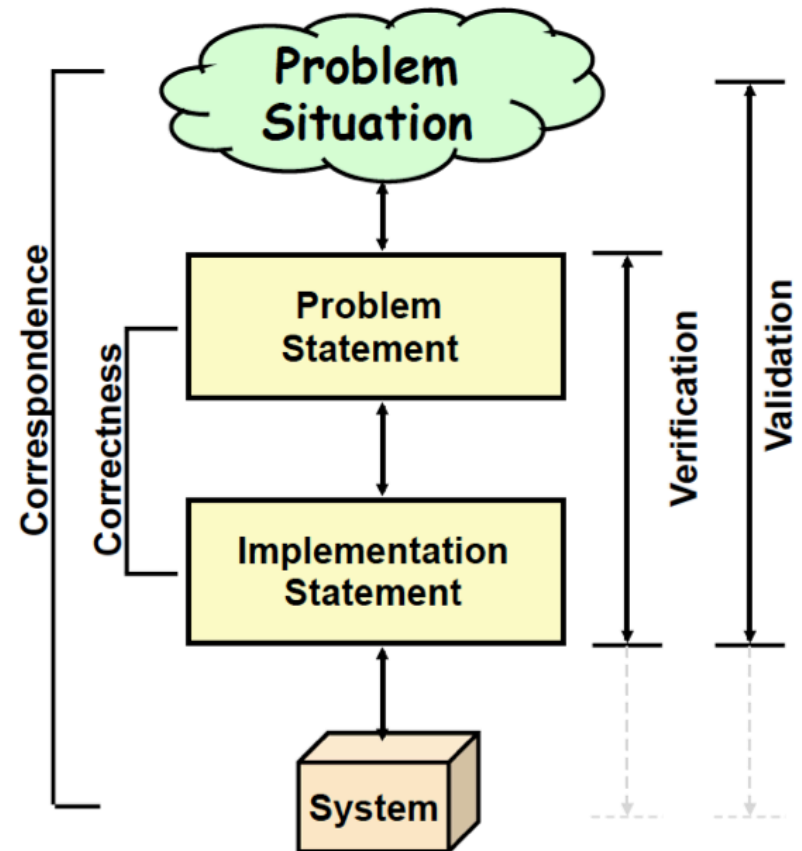# *requirements to design*

csc302h

winter 2014

- requirements analysis
  - It's all about (correctly) identifying the purpose

what problem are we trying to solve?

  - answer this wrong and you'll have a quality fail (and all it's associated nastiness)

- will talk more about v&v later but...
  - verification answers; does the solution solve the problem?
  - validation answers; did solving the identified problem satisfy the goals of the stakeholders?

- from requirements to production

Application Domain

D - domain properties

R - requirements

s - specification

Machine Domain

C - computers

P - programs

- D: consists of assumptions, or, truths
- R: are the wants, the things that solve the problem
- S: is the bridge, what must the system do to satisfy R
- P: is the program that satisfies S
- C: is easy, buy it, rent it, or it lives in the cloud, or…

- in general: S (given D) $\Rightarrow$ R

Computer Science
UNIVERSITY OF TORONTO

Application Domain

D - domain properties

R - requirements

s - specification

Machine Domain

C - computers

P - programs

- verification: does the system (C & P) satisfy the specification?

- validation: was the knowledge of the domain [D] and requirements [R] complete & correct (to the best of our ability)

- we're not done…actually, we're never done!
  - requirements change over time
  - implemented system changes the world it's in
    - recall Lehman's e-type programs?
  - requirements incomplete, despite best efforts
  - the last person screwed up!

- so, therefore, requirements analysis is usually an iterative process

- requirements analysis can be iterative for other reasons too:
  - ex. trying to capture all requirements may not be:
    - possible (can't predict the future)
    - cost effective
  - ex. facebook, web browsers

Computer Science
UNIVERSITY OF TORONTO

- so how do we solicit requirements? where do they come from?
  - identify the stakeholders
  - identify the goals of the stakeholders
  - then think very hard about:

  what problem are

  we trying to solve?

 - it may not be what you were told (rare if it is)

Computer Science
UNIVERSITY OF TORONTO

goal analysis

Goal Elaboration:

"Why" questions explore higher goals (context)
"How" questions explore lower goals (operations)
"How else" questions explore alternatives

Relationships between goals:

One goal helps achieve another (+)
One goal hurts achievement of another (-)
One goal makes another (++)
   Achievement of goal A guarantees achievement of goal B
One goal breaks another (--)
   Achievement of goal A prevents achievement of goal B
Precedence ordering – if goals must be achieved in a particular order
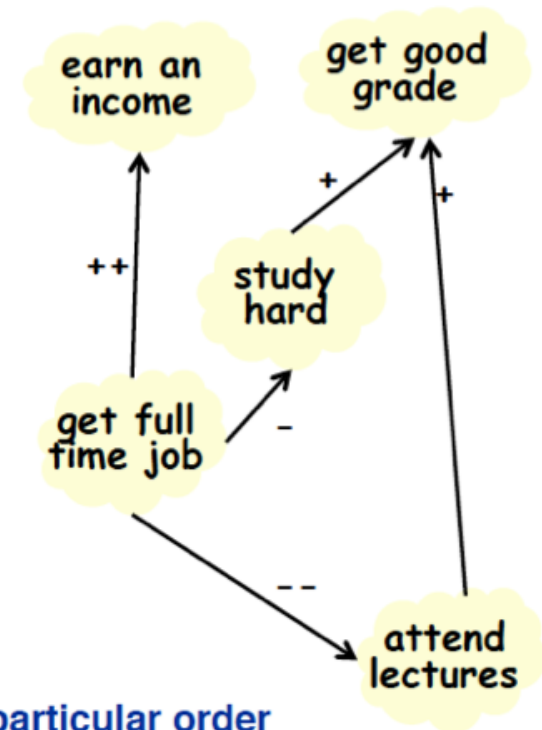
Obstacle Analysis:

Can this goal be obstructed, if so how?
What are the consequences of obstructing it?

earn an income

get good grade

study hard

get full time job

attend lectures

++

+

+

-

--

requirements to design

From Requirements to Design

Identifying Actors

Building a Domain Model

Goal modeling

Obstacle Analysis

Scoping

Use Cases

Starting Point

Given:

a vague request for a new feature from users of your software

1. Identify the problem ← *stakeholder goal modeling & domain models*

What is the goal of the project?
What is the "vision" of those who are pushing for it?

2. Scope the problem

Given the vision, how much do we tackle?
What new functionality will be needed?

use cases

3. Identify solution scenarios ←

Given the problem, how will users interact with the software to solve it?

4. Map onto the Architecture ← *robustness analysis*

How will the needed functionality be met?
What new modules / classes will be needed?

What do Requirements Analysts do?

Given a "problem"…

Some notion that a "problem" needs solving

  e.g. dissatisfaction with the current system

  e.g. a new business opportunity

  e.g. a potential saving of cost, time, resource usage, etc.

A Requirements Analyst is an agent of change

… the requirements analyst must:

identify the "problem" / "opportunity"

  Which problem needs to be solved? (identify problem **Boundaries**)

  Where is the problem? (understand the **Context**/Problem Domain)

  Whose problem is it? (identify **Stakeholders**)

  Why does it need solving? (identify the stakeholders' **Goals**)

  When does it need solving? (identify Development **Constraints**)

  What might prevent us solving it? (identify **Feasibility** and **Risk**)

  How might a software system help? (collect some **Scenarios / Use Cases**)

Refesher

Application Domain

Machine Domain

D - domain properties

s - specification

C - computers

R - requirements

P - programs

Domain Properties (assumptions):

things in the **application domain** that are true whether or not we ever build the proposed system

(System) Requirements:

things in the **application domain** that we wish to be made true by delivering the proposed system

Many of which will involve phenomena the machine has no access to

A (Software) Specification:

is a description of the behaviours that **the program** must have in order to meet the **requirements**

Can only be written in terms of shared phenomena!

Identifying Actors

Ask the following questions:

Who will be a primary user of the system? (primary actor)

Who will need support from the system to do her daily tasks?

Who or what has an interest in the results that the system produces ?

Who will maintain, administrate, keep the system working? (secondary actor)

Which hardware devices does the system need?

With which other systems does the system need to interact with?

Look for:

the users who directly use the system
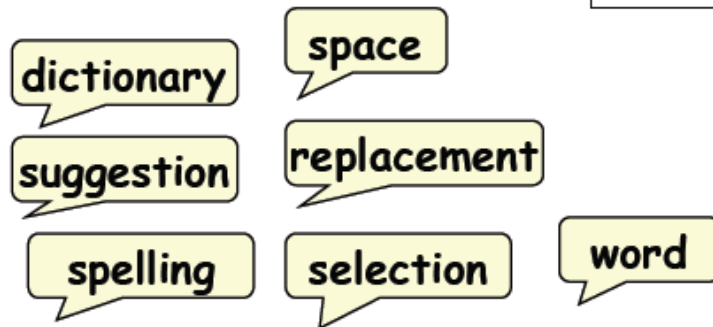
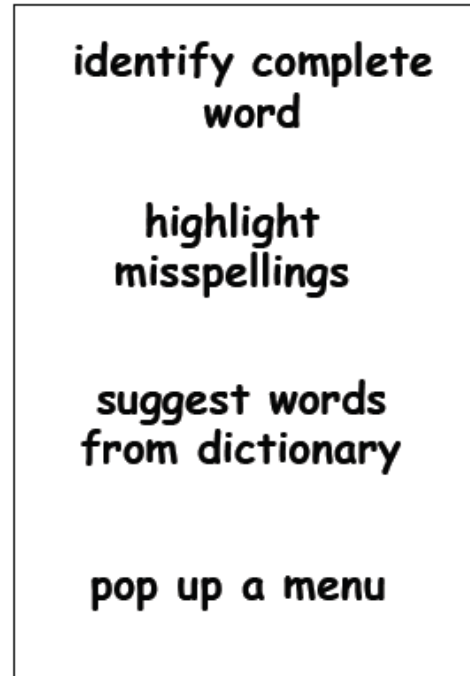also others who need services from the system
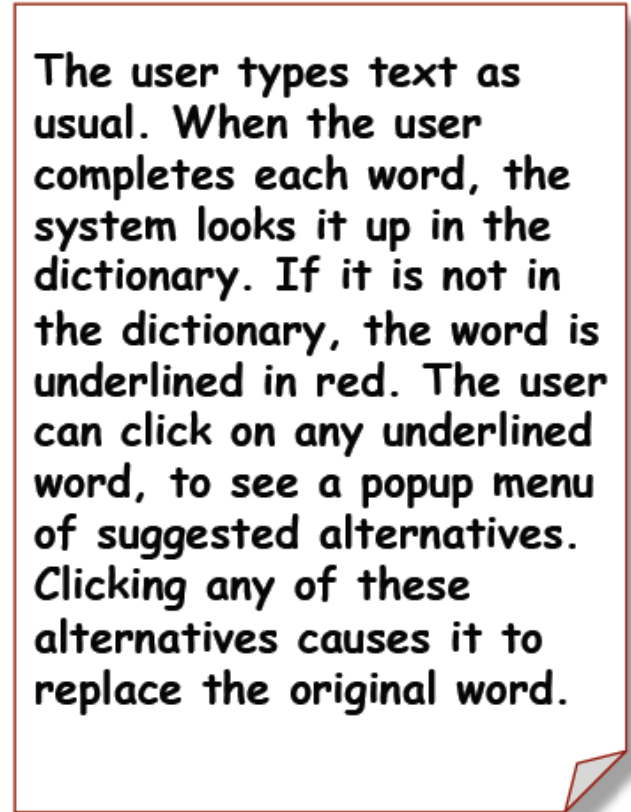
Key distinctions

A requirement (goal)

> reduce the number of spelling mistakes

Domain Concepts

dictionary

space

suggestion

replacement

spelling

selection

word

Functions

identify complete word

highlight misspellings

suggest words from dictionary

pop up a menu

A Use Case

The user types text as usual. When the user completes each word, the system looks it up in the dictionary. If it is not in the dictionary, the word is underlined in red. The user can click on any underlined word, to see a popup menu of suggested alternatives. Clicking any of these alternatives causes it to replace the original word.

Computer Science
UNIVERSITY OF TORONTO

```java
tv.addTextChangedListener(new TextWatcher() {
    /** Characters that define a word boundary. */
    private static final String WORD_BOUNDARY = " .?!,;:)]}\n\t";

    @Override
    public void onTextChanged(CharSequence s, int start, int before, int count) {
        if (((CheckBox) findViewById(R.id.kb_speak_while_typing)).isChecked()) {
            // Check that a single new char was appended, that's all I care about
            if (s != null && s.length() > 0 && before == 0 && count == 1 && start == s.length()-1) {
                if (WORD_BOUNDARY.indexOf(s.charAt(s.length() - 1)) != -1) { // finished a word, speak it!
                    String word = s.toString().substring(0, s.length()-1);

                    // find start of the word
                    int startOfWord = -1;
                    for (int i = word.length()-1; i >= 0; i--) {
                        if (WORD_BOUNDARY.indexOf(word.charAt(i)) != -1) {
                            startOfWord = i+1; break;
                        }
                    }
                    if (startOfWord == -1) startOfWord = 0; // first word

                    // isolate the word to speak
                    word = word.substring(startOfWord, word.length());
                    if (word.length() > 0) MyVoiceApp.speak(word);
                }
            }
        }
    }
    @Override public void afterTextChanged(Editable s) { }
    @Override public void beforeTextChanged(CharSequence s, int start, int count, int after) {}
});
```
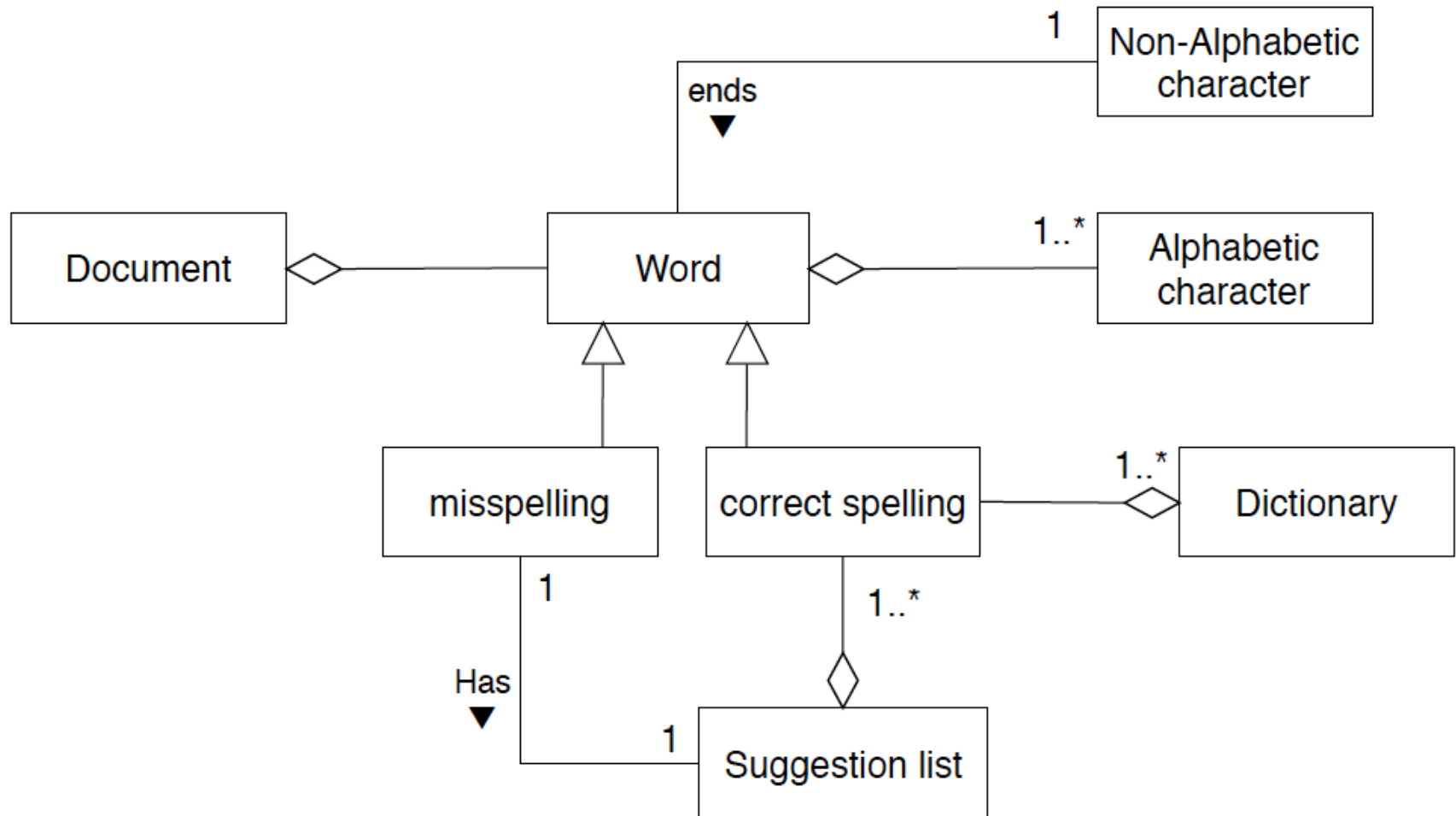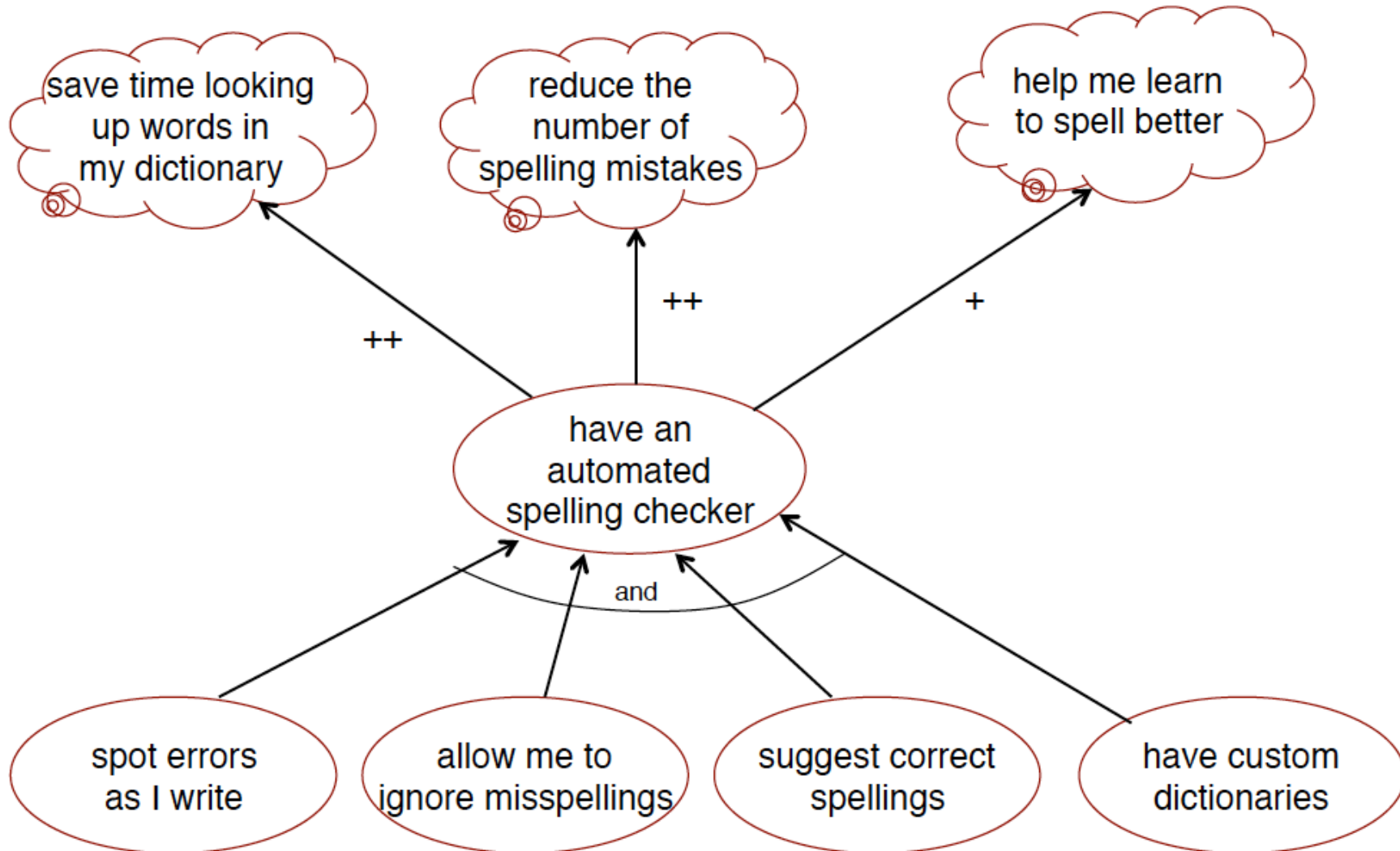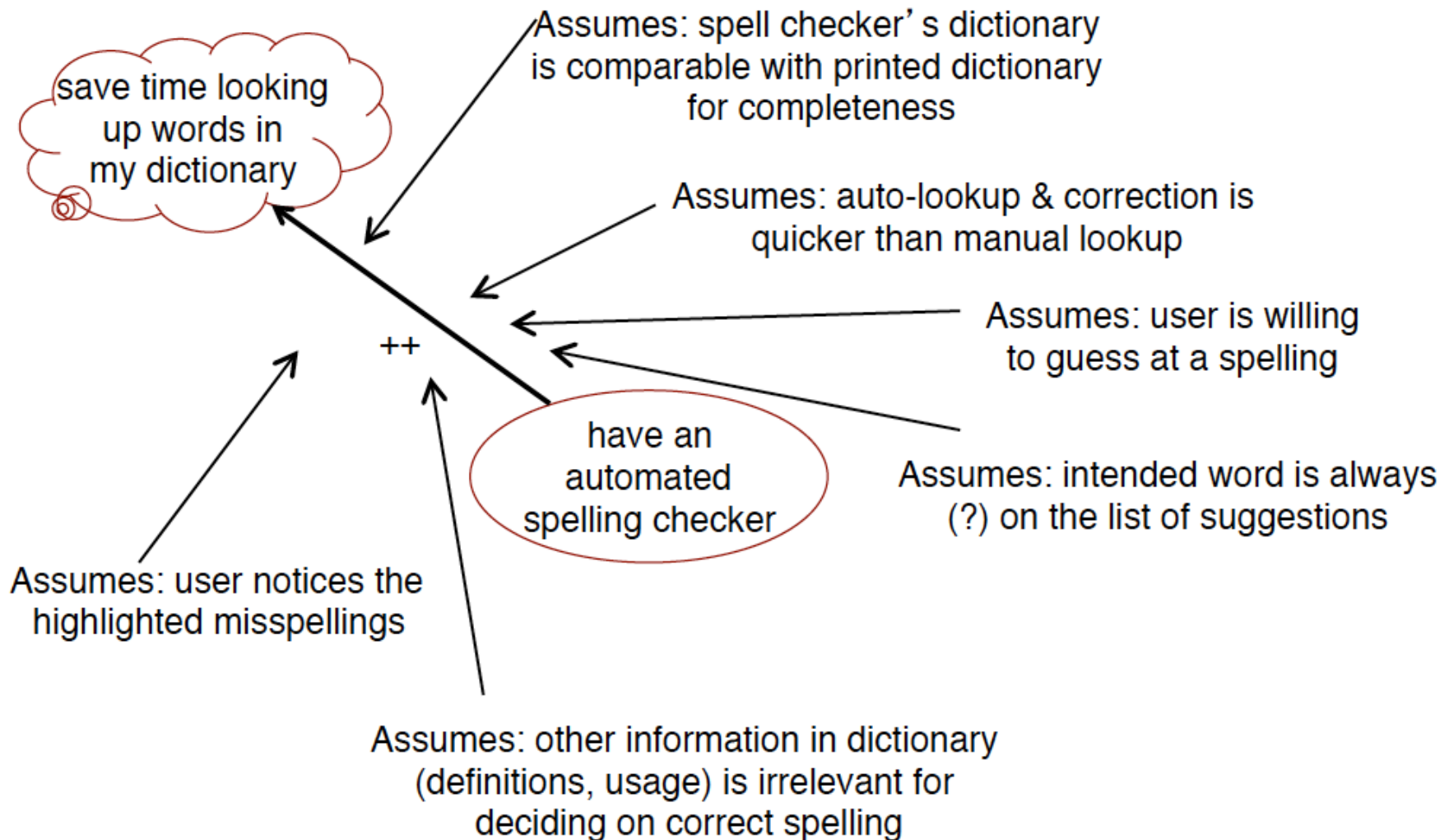
Domain Model

Exploring Goals

Obstacle Analysis

save time looking
up words in
my dictionary

Assumes: spell checker's dictionary
is comparable with printed dictionary
for completeness

Assumes: auto-lookup & correction is
quicker than manual lookup

Assumes: user is willing
to guess at a spelling

++

have an
automated
spelling checker

Assumes: intended word is always
(?) on the list of suggestions

Assumes: user notices the
highlighted misspellings

Assumes: other information in dictionary
(definitions, usage) is irrelevant for
deciding on correct spelling

Some Requirements emerge…

"Functional Requirements"

- (?) User can see definitions for suggested spellings
- User shall be able to add custom dictionaries
- User shall be able to add new words to a custom dictionary
- User shall be able to declare certain words be ignored for spell checking for the current document
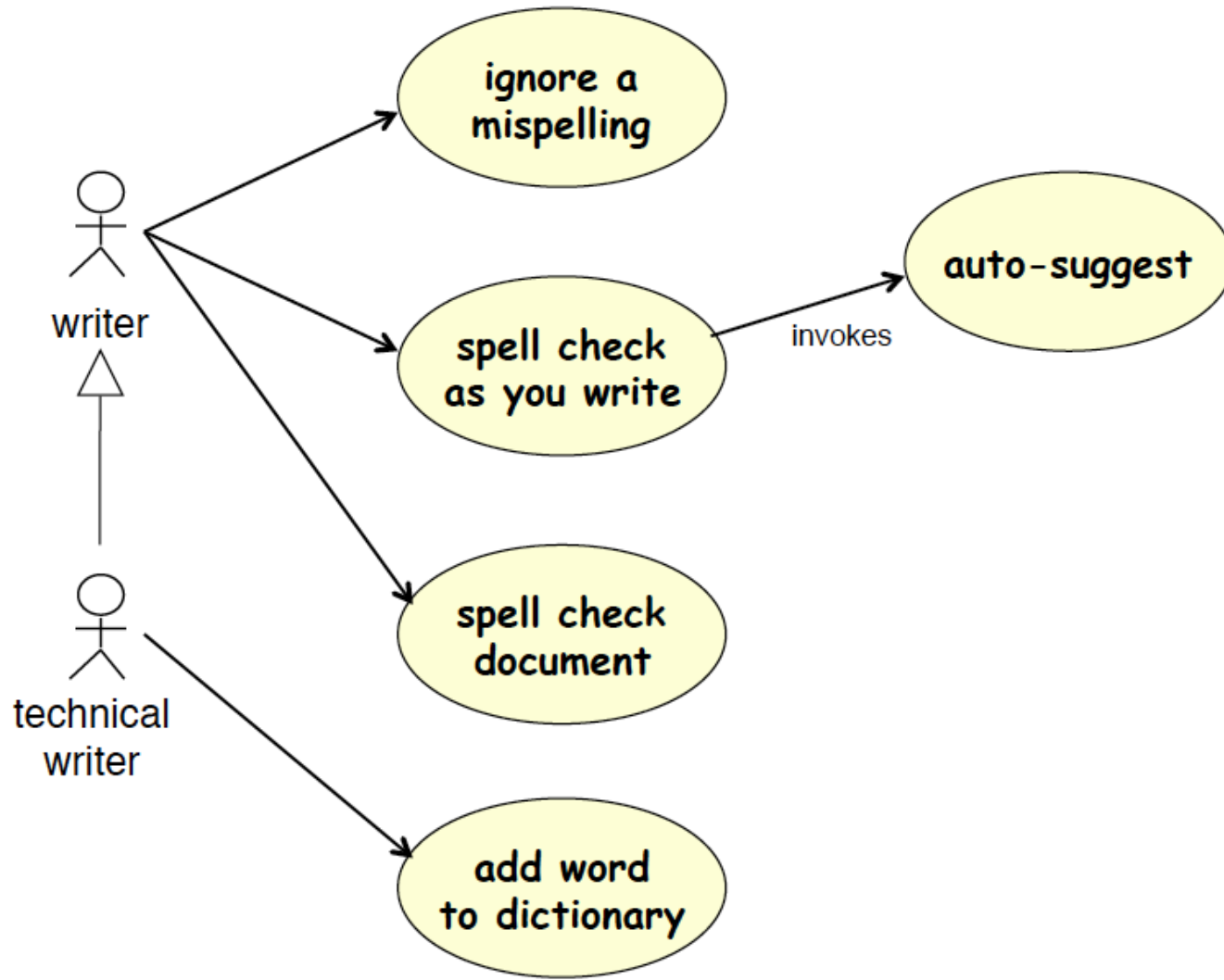
"Quality Requirements"

- Dictionary should be as comprehensive as printed dictionaries
- Checking and suggesting should be fast
- Highlighted misspellings must be clearly visible

Use Case Diagram

Finding Use Cases

For each actor, ask the following questions:

Which functions does the actor require from the system?

What does the actor need to do ?

Does the actor need to read, create, destroy, modify, or store some kinds of information in the system ?

Does the actor have to be notified about events in the system?

Does the actor need to notify the system about something?

What do those events require in terms of system functionality?

Could the actor's daily work be simplified or made more efficient through new functions provided by the system?

Documenting Use Cases

For each use case:

prepare a "flow of events" document, written from an actor's point of view.

describe what the system must provide to the actor when the use case is executed.

Typical contents

How the use case starts and ends;

Normal flow of events;

Alternate flow of events;

Exceptional flow of events;

Documentation style:

Choice of how to represent the use case:

English language description

Activity Diagrams - good for business process

Collaboration Diagrams - good for high level design

Sequence Diagrams - good for detailed design

13

Sample Use Case documentation

Brief name
(This is all that appears on the on use case diagram)

Name: **Place Order**

Precondition: A valid user has logged into the system.

Description:

1. The use case starts when the customer selects Place Order.
2. The customer enters his or her name and address.
3. If the customer enters only the zip code, the system will supply the city & state.
4. The customer will enter product codes for the desired products.
5. The system will supply a product description and price for each item.
6. The system will keep a running total of items ordered as they are entered.
7. The customer will enter credit card payment information.
8. The customer will select Submit.
9. The system will verify the information, save the order as pending, and forward payment information to the accounting system.
10. When payment is confirmed, the order is marked Confirmed, an order ID is returned to the customer, and the use case ends.

Exceptions:

In step 9, if any information is incorrect, the system will prompt the customer to correct the information.

Postcondition: The order has been saved in the system and marked confirmed.

Clear start and end

Every step clearly identifies who carries it out

Clear statement of any effects

Scoping decision I

Decide the scope of the problem:

E.g. Bookstore example:

"Textbooks are often not ordered in time for the start of classes"

But that's just a symptom. (So you ask the manager "why?")

"Because we don't receive the booklists from instructors early enough"

Is that just a symptom of some other problem? (...so ask the instructors "why?")

"Because the instructors aren't allocated to courses early enough"

Is that just a symptom of some other problem? (...so ask the UG office "why?")

"Because we never know who's available to teach until the last minute"

Is that just a symptom of some other problem? (...so ask the dept chair "why?")

"Because there's always uncertainty about who gets hired, sabbaticals, etc."

Is that just a symptom of some other problem? (...so ask the dept chair "why?")

"Because instructors we want to hire don't accept our offers early enough"

Is that just a symptom of some other problem? (...so ask the new recruits "why?")

"Because some other universities seem to wait for ages before making offers"

Is that just a symptom of some other problem? (...so ask U of Waterloo, etc, "why?")

"Because it takes our department a long time to reach consensus on hiring"

Is that just a... ...oh wait... ...maybe we can develop a decision support system for faculty hiring at U of Waterloo, and that will help us get our textbooks for the start of class...

How to scope the problem

Difficulty:

Every problem can be seen as as symptom of some other (larger) problem

You can keep on tracing root causes forever if you're not careful

Approach: (…ask yourself these questions…)

Is there a reasonable expectation that this problem can be solved?
(…independently of the larger problem?)

Is there a reasonable expectation that solving this problem will help?
(…without also solving the larger problem?)

Is this a problem that the stakeholders want solved?
(do the "local experts" think this problem is the one that matters?)

Is this a problem that someone will pay you to solve?
(Hint: a feasibility study should quantify the return on investment)

Scoping Decision II

Decide the scope of the solution

Say you decided that *delay in processing booklists from instructors* is the right level of problem to tackle.

"So, let's computerize the submission of textbook forms from instructors"

But while we're at it:

"it would help if we also computerized the submission of orders to the publishers"

…and of course:

"we ought to computerize the management of book inventories too, so we can quickly check stock levels before ordering new books"

…and in that case:

"we might as well computerize the archives of past years booklists so that we can predict demand better"

…and therefore:

"it would also make sense to provide a computerized used book exchange, because that has a big effect on demand for new books"

…and then of course there's … oh, wait, this is going to cost millions!

Bookstore manager: "tell me again how this automated used book exchange will help me order books faster?"

How to scope the solution

Difficulty:

We could keep on throwing more technology at the problem forever

It's hard to decide when to stop adding extra "bells and whistles"

Approach (...select among alternatives carefully...)

Is there a reasonable expectation that this alternative can be implemented?

(...independently of all the other options?)

Is there a reasonable expectation that implementing this alternative will (help to) solve the original problem?

(...without also having to address other aspects of the problem?)

Is this a solution that the stakeholders can live with?

(do the "local experts" think they would use all these functions?)

Is this a solution that someone will pay you to build?

(Hint: a feasibility study should quantify the return on investment for each alternative)