Übung 3 – User Stories

Methoden des Software Engineering

Christian Kroiß

Outline

I. Writing User stories

II. Non-functional requirements as user stories

Writing user stories

- Guidelines for writing good user stories: INVEST [Wake2003]

Independent

Negotiable

Valuable to users or customers

Estimatable

Small

Testable

INVEST: Independent

Independent

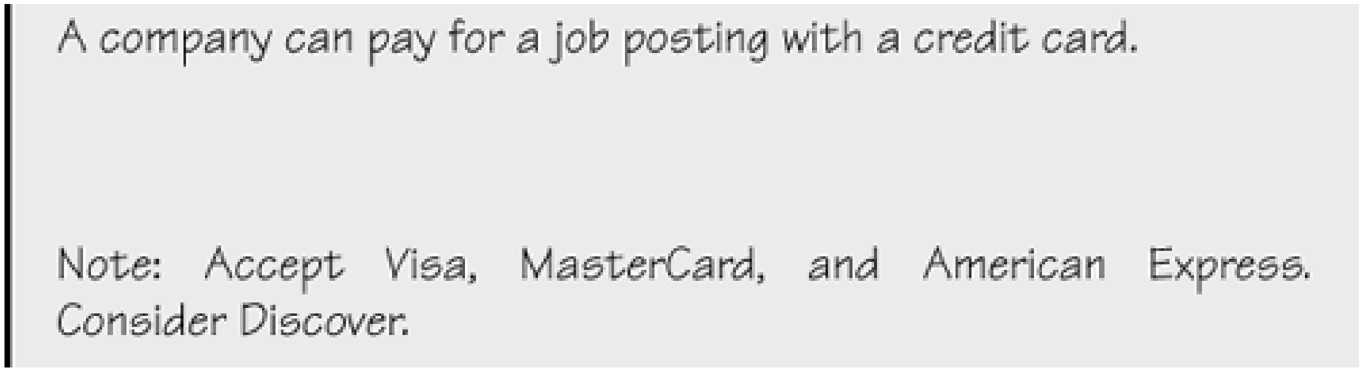
- No overlap in concept
- We'd like to be able to schedule and implement them in any order
- Not always achievable, e.g.
  - 2 days for supporting one type of credit card
  - 1 day for each other type

LUDWIG-
MAXIMILIANS-
UNIVERSITÄT
MÜNCHEN

INVEST: Negotiable

Negotiable

- Not an explicit contract for features
- Captures the essence, not the details
- Over time: add notes, test ideas, etc.



A company can pay for a job posting with a credit card.

Note: Accept Visa, MasterCard, and American Express. Consider Discover.

[Cohn2004]

INVEST: Valuable to users or customers

Valuable to users or customers

- Consider interests of direct users but also of purchaser
- Best way to ensure: have the customer write the stories

INVEST: Valuable to users or customers

Example:

Use

”Up to fifty users should be able to use the application with a five-user database license.”

Instead of

“All connections to the database are through a connection pool.“

INVEST: Estimatable

Estimatable

- There are three common reasons why a story may not be estimatable:
  1. Developers lack domain knowledge.
  2. Developers lack technical knowledge.
  3. The story is too big.
- Possible Solutions
  - 1. & 2.: let developers do time-boxed experiment (called *spike* in Extreme Programming)
  - 3.: Split stories

INVEST: Small

Small

- Oversized stories (epics) not usable in planning
- But: too small means too much overhead
- Solutions
  - Split
  - Combine

INVEST: Testable

Testable

- Especially for non-functional requirements

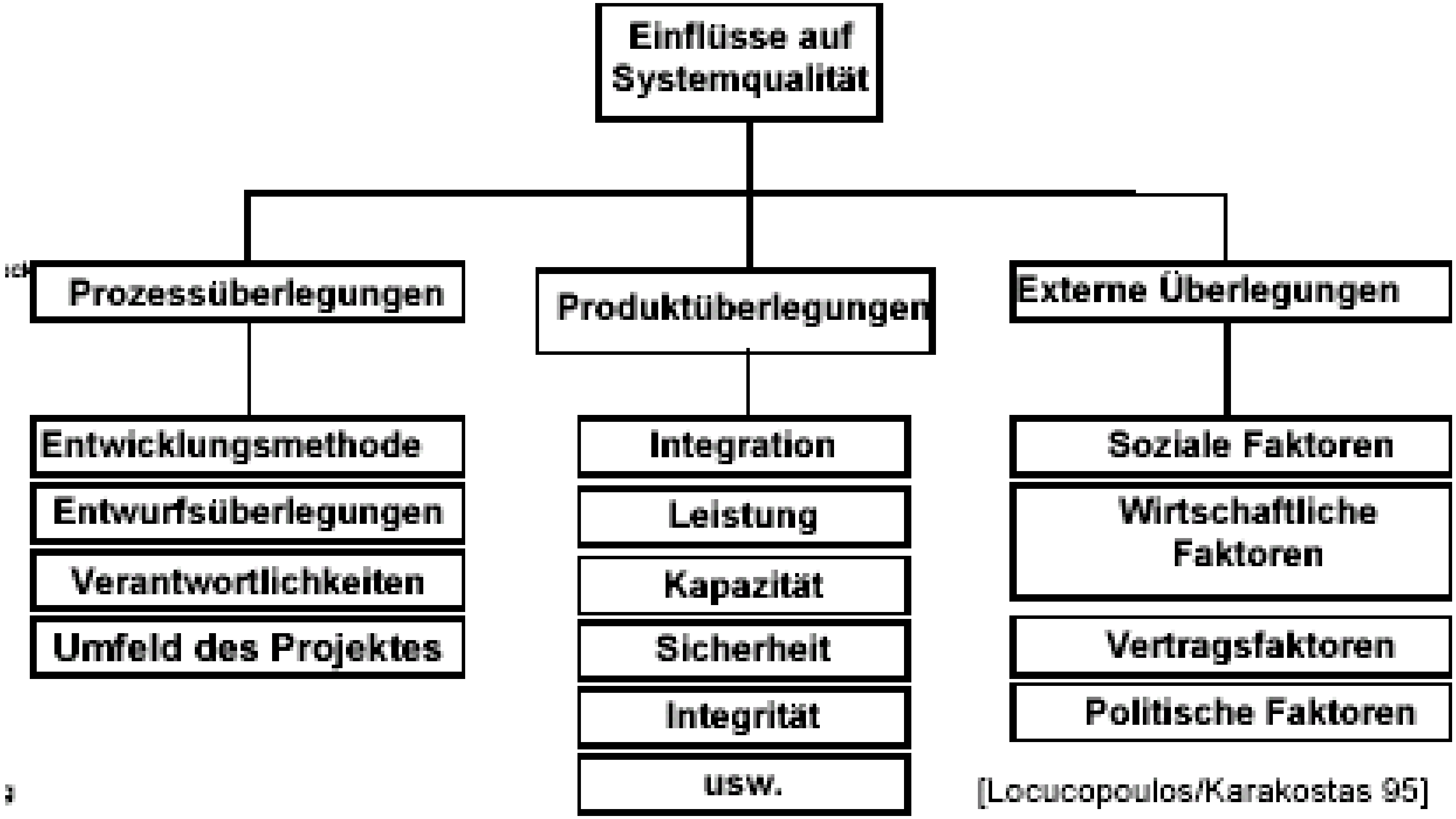
Examples of **untestable** stories:

- A user must find the software <u>easy to use</u>.
- A user must <u>never</u> have to wait long for any screen to appear.

Handling non-functional requirements

- Use “constraint” label to distinguish story card
- Constraints are not scheduled like other user stories
- Make them testable!
- If possible, pin them to a wall!
- Make them testable!

Types of non-functional requirements



[Locucopoulos/Karakostas 95]

Sample constraints

Area	Sample Constraint
Performance	80% of database searches will return results to the screen in less than two seconds.
Accuracy	The software will correctly predict the winner of a football game at least 55% of the time.
Portability	The system shall not make use of any technology that would make it difficult to port to Linux.
Reusability	The database and database access code will be reusable in future applications.

Sample constraints

Area	Sample Constraint
Maintainability	Automated unit tests must exist for all components. Automated unit tests must be run in their entirety at least once every 24 hours.
Interoperability	The system shall be written in Java. All configuration data shall be stored in XML files. Data shall be stored in MySQL.
Capacity	The database will be capable of storing 20 million members on the specified hardware while still meeting performance objectives.

Literatur

[Cohn2004] Mike Cohn. User Stories Applied: For Agile Software Development . Addison-Wesley. 2004.

[Wake2003] William C. Wake. INVEST in Good Stories, and SMART Tasks. 2003. http://xp123.com/xplor/xp0308/