If software is simply for automation, what would a washing machine be like?

RE Process

Lawrence Chung
Department of Computer Science
The University of Texas at Dallas

RE Process:
What is a Process?

- Given input, transforms it into output
- Consist of a set of activities
- Process descriptions are also **specifications**
  - Often produced by Requirements Engineers
  - Should be as complete, consistent and clear

RE Process: Why?

Quality of product ⟸ **Quality of Process**

- Garbage in garbage out,
  so get the right requirements

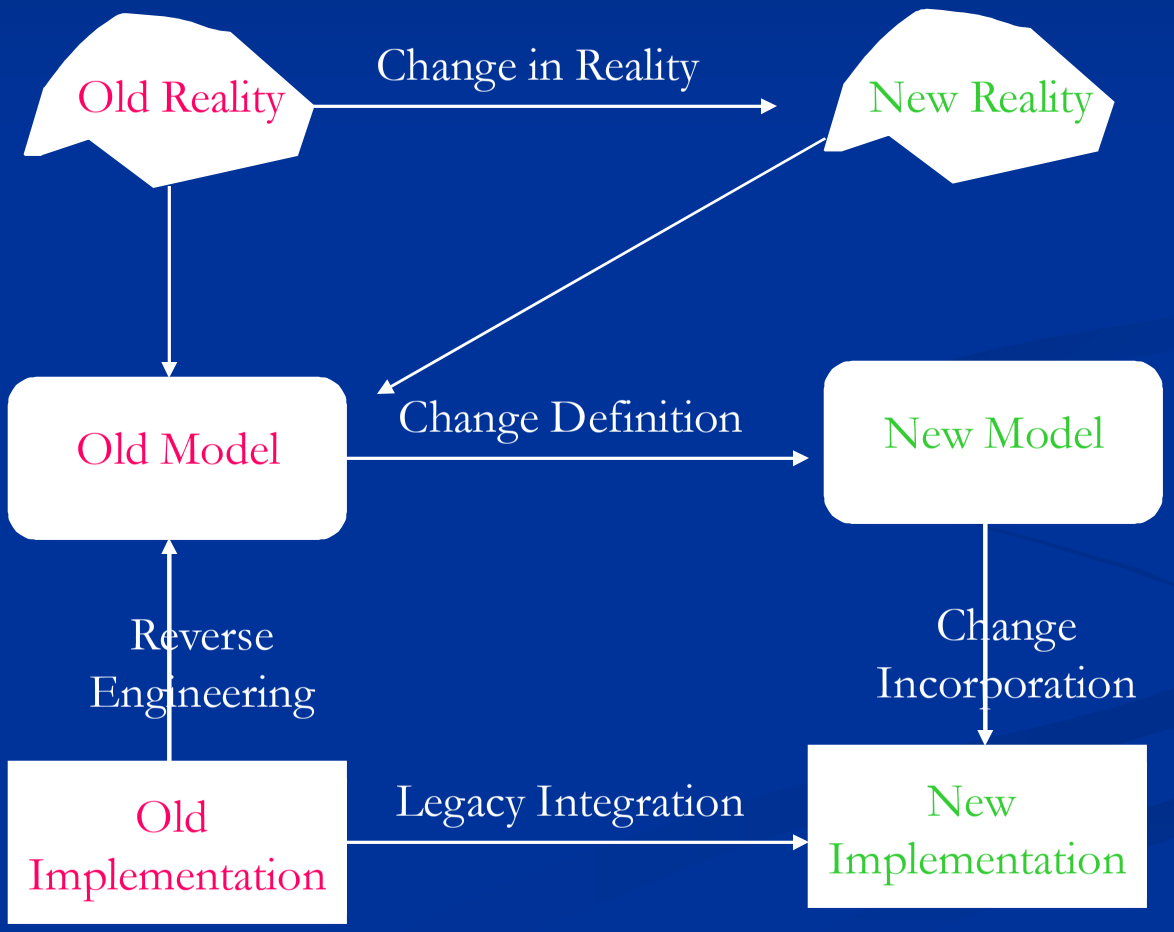
Product

cf. washing machine

It is more important to understand the problem than the solution. [Albert Einstein]

RE Process:
The Basic RE Evolutionary Process

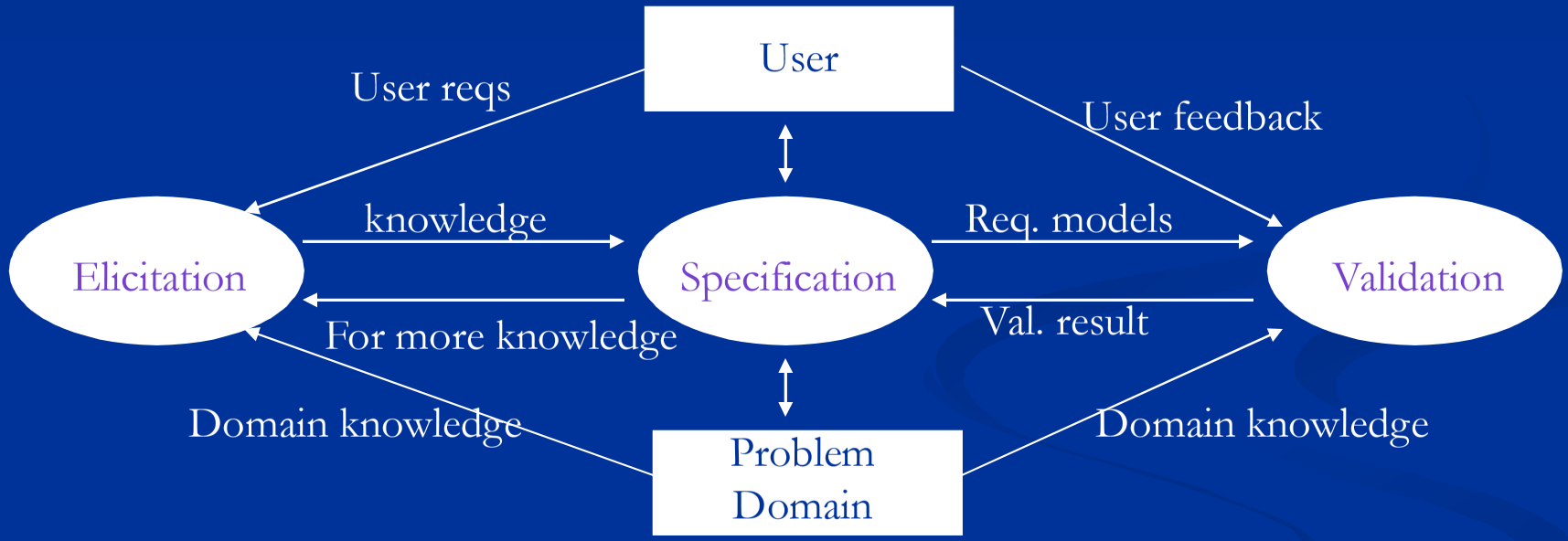
Evolution is inevitable – *traceability* is more than a virtue



RE Process: A Basic Framework [Loucopolos]

Many variations and extensions

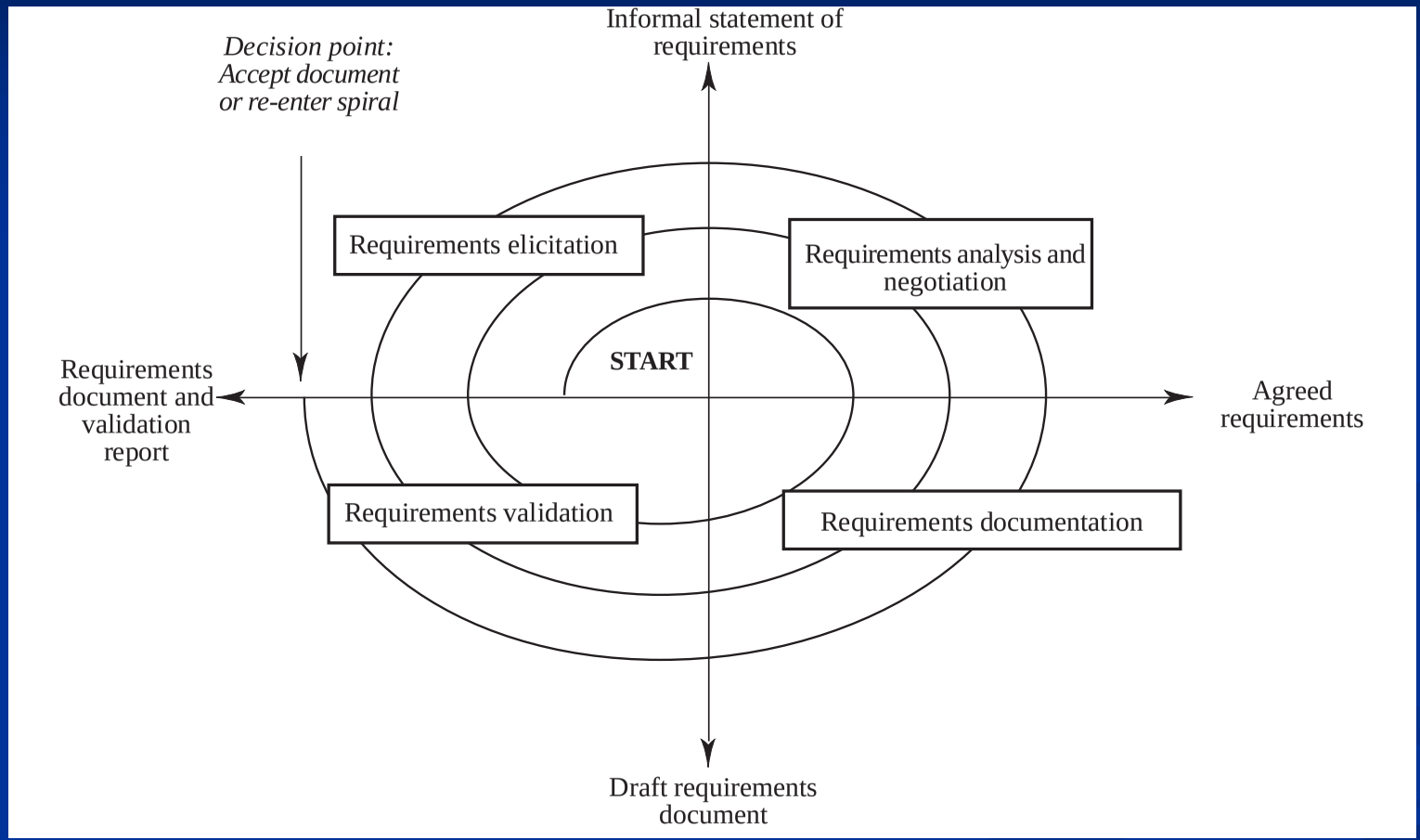
- **3 fundamental activities:**
  understand, (formally) describe, attain an agreement on, the problem



- **Elicitation: determine what's really needed, why needed, whom to talk to**
- **Specification: produce a (formal) RS model: translate "vague" into "concrete", etc. make various decisions on what & how**
- **Validation: assure that the RS model satisfies the users' needs**

RE Process: Spiral Model [KotonyaSummerville98]

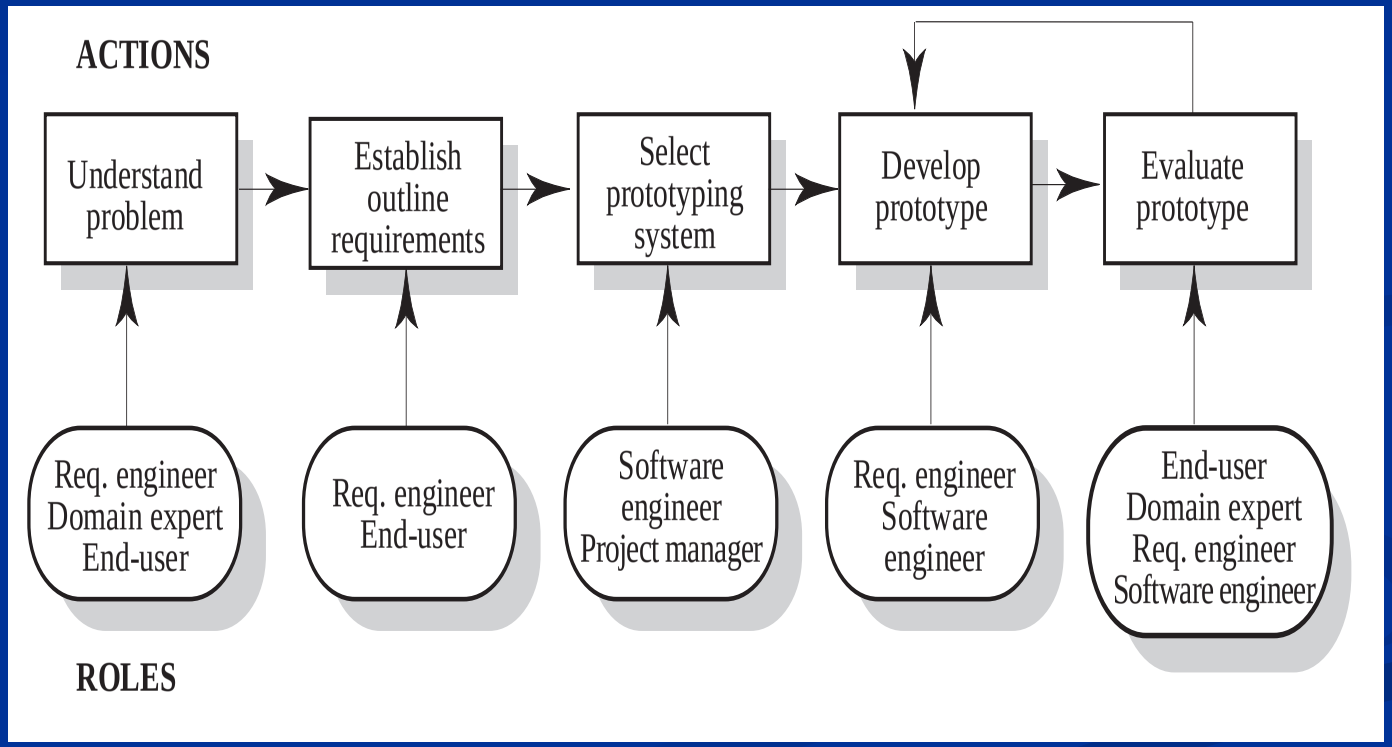
How many cycles? When to analyze and negotiate? Risk analysis?



- Requirements elicitation: Requirements discovered through consultation with stakeholders
- Requirements analysis and negotiation: Requirements are analysed and conflicts resolved through negotiation
- Requirements documentation: A requirements document is produced
- Requirements validation: The requirements document is checked for consistency and completeness

RE Processes:
RAD (Role Actor Diagram)

An RE Process is dominated by human, social and organisational factors

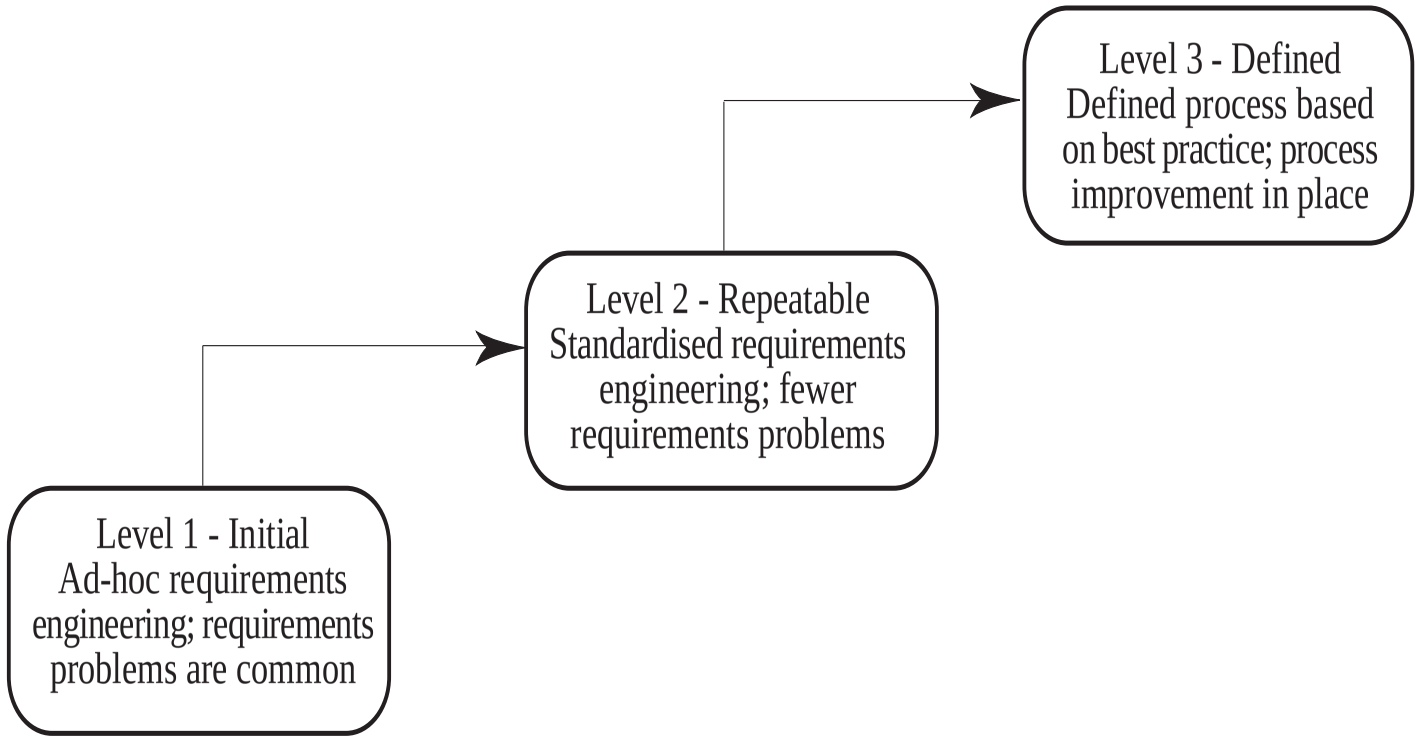


Stakeholders/
Actors/
Agents

for prototyping [Kotonya&Sommerville98]

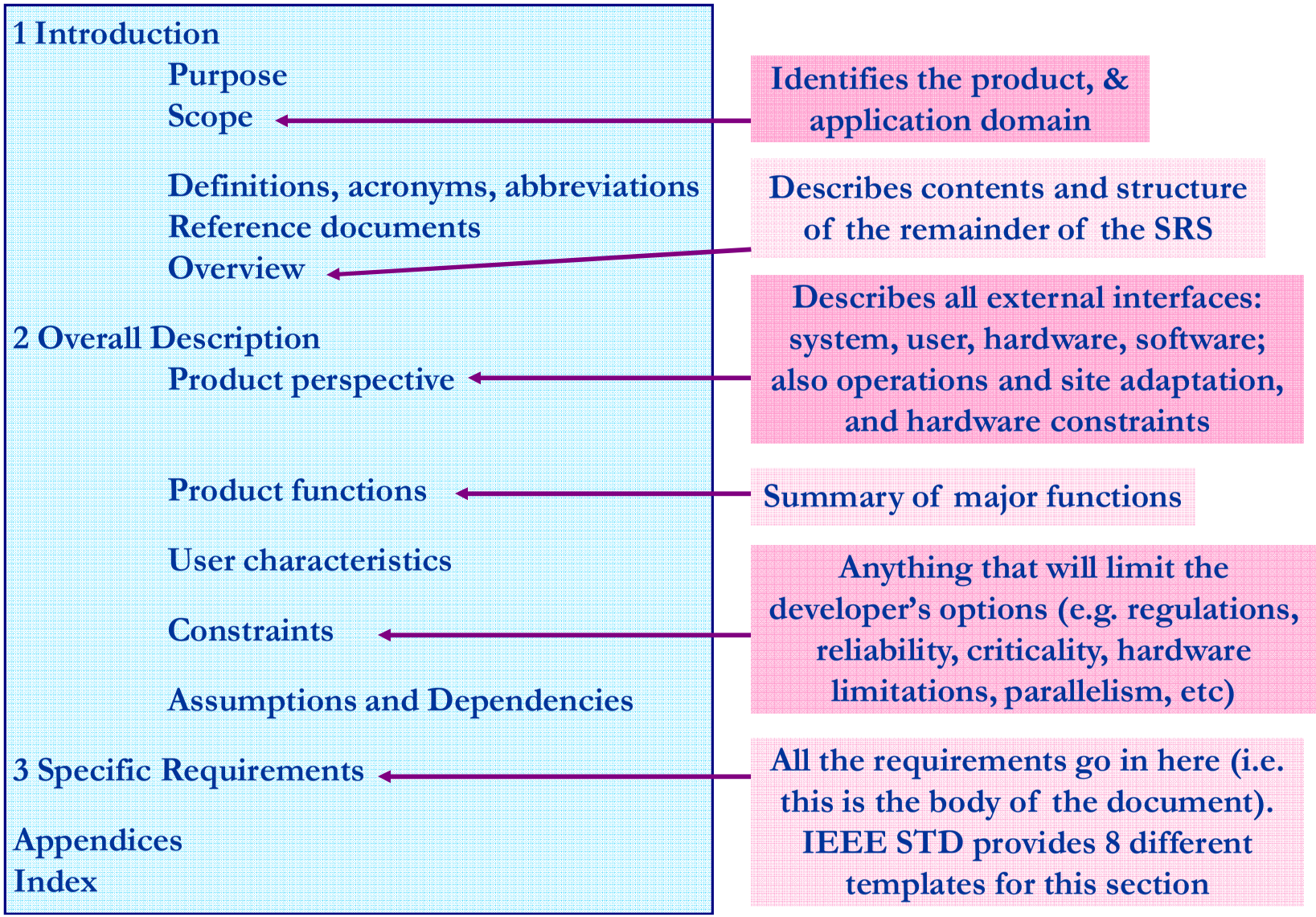
RE Process:
A RE Process Maturity Model

Based on CMM



IEEE Standard for SRS

[IEEE-STD-830-1993] [Blum 1992, p160]



IEEE Standard Section 3

[IEEE-STD-830-1993.] [Blum 1992, p160]

- 3.1 External Interface Requirements
  - 3.1.1 User Interfaces
  - 3.1.2 Hardware Interfaces
  - 3.1.3 Software Interfaces
  - 3.1.4 Communication Interfaces
- 3.2 Functional Requirements
  - *this section organized by mode, user class, feature, etc.*
  - *For example:*
    - 3.2.1 Mode 1
      - 3.2.1.1 Functional Requirement 1.1
      - …
    - 3.2.2 Mode 2
      - 3.2.1.1 Functional Requirement 1.1
      - …
    - …
    - 3.2.n Mode n
      - …
- 3.3 Performance Requirements
  - *Remember to state this in measurable terms!*
- 3.4 Design Constraints
  - 3.4.1 Standards compliance
  - 3.4.2 Hardware limitations
  - etc.
- 3.5 Software System Attributes
  - 3.5.1 Reliability
  - 3.5.2 Availability
  - 3.5.3 Security
  - 3.5.4 Maintainability
  - 3.5.5 Portability
- 3.6 Other Requirements

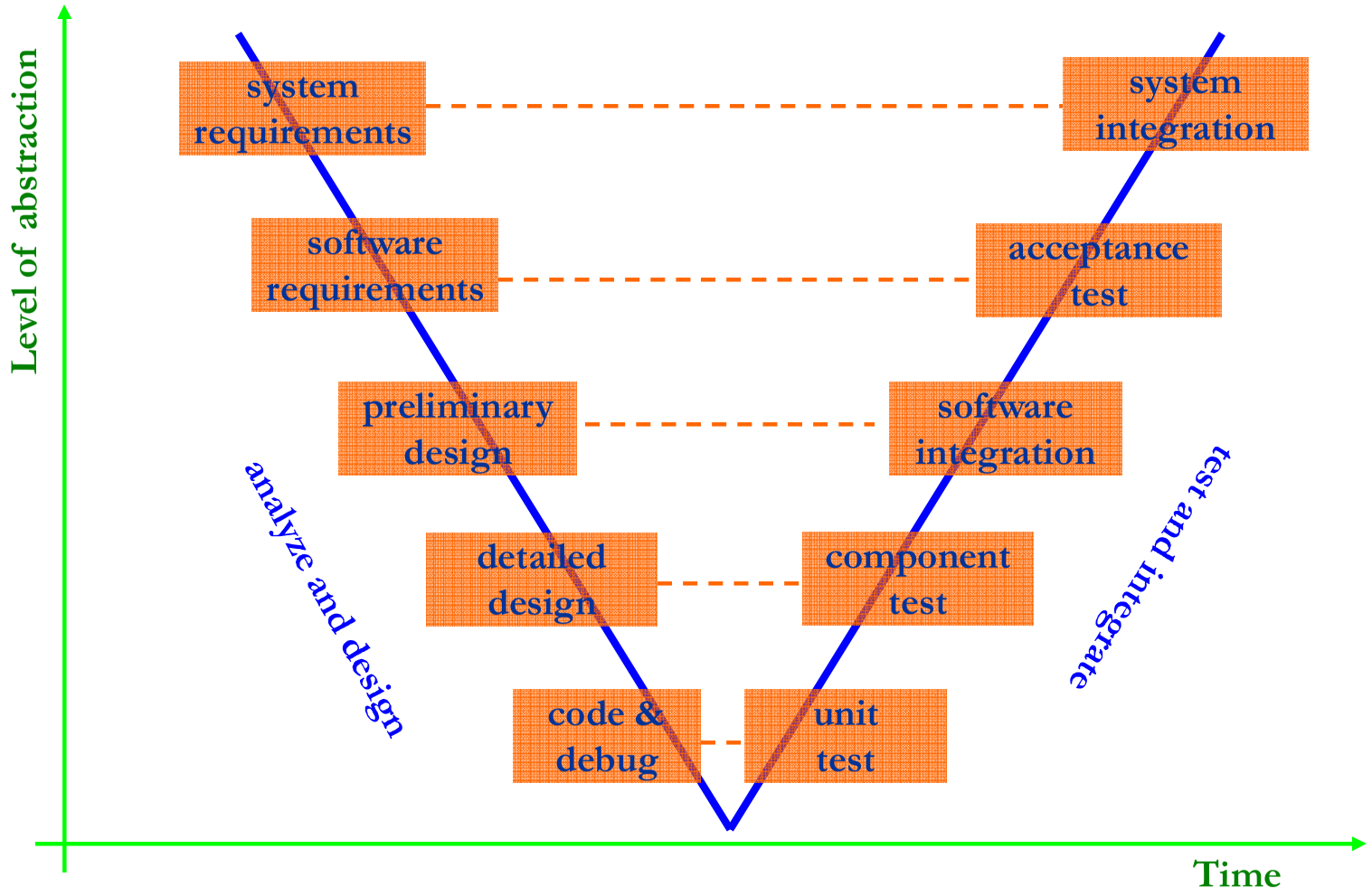
RE in Agile Methods

- Basic Philosophy
  - Reduce communication barriers
    - Programmer interacts with customer
  - Reduce document-heavy approach
    - Documentation is expensive and of limited use
  - Have faith in the people
    - Don't need fancy process models to tell them what to do!
  - Respond to the customer
    - Rather than focussing on the contract
- Weaknesses
  - Relies on programmer's memory
    - Code can be hard to maintain
  - Relies on oral communication
    - Mis-interpretation possible
  - Assumes single customer representative
    - Multiple viewpoints not possible
  - Only short term planning
    - No longer term vision

E.g. Extreme Programming

- **Instead of a requirements spec, use:**
  - **User story cards**
  - **On-site customer representative**
- **Pair Programming**
- **Small releases**
  - **E.g. every three weeks**
- **Planning game**
  - **Select and estimate user story cards at the beginning of each release**
- **Write test cases before code**
- **The program code is the design doc**
  - **Can also use CRC cards (Class-Responsibility-Collaboration)**
- **Continuous Integration**
  - **Integrate and test several times a day**

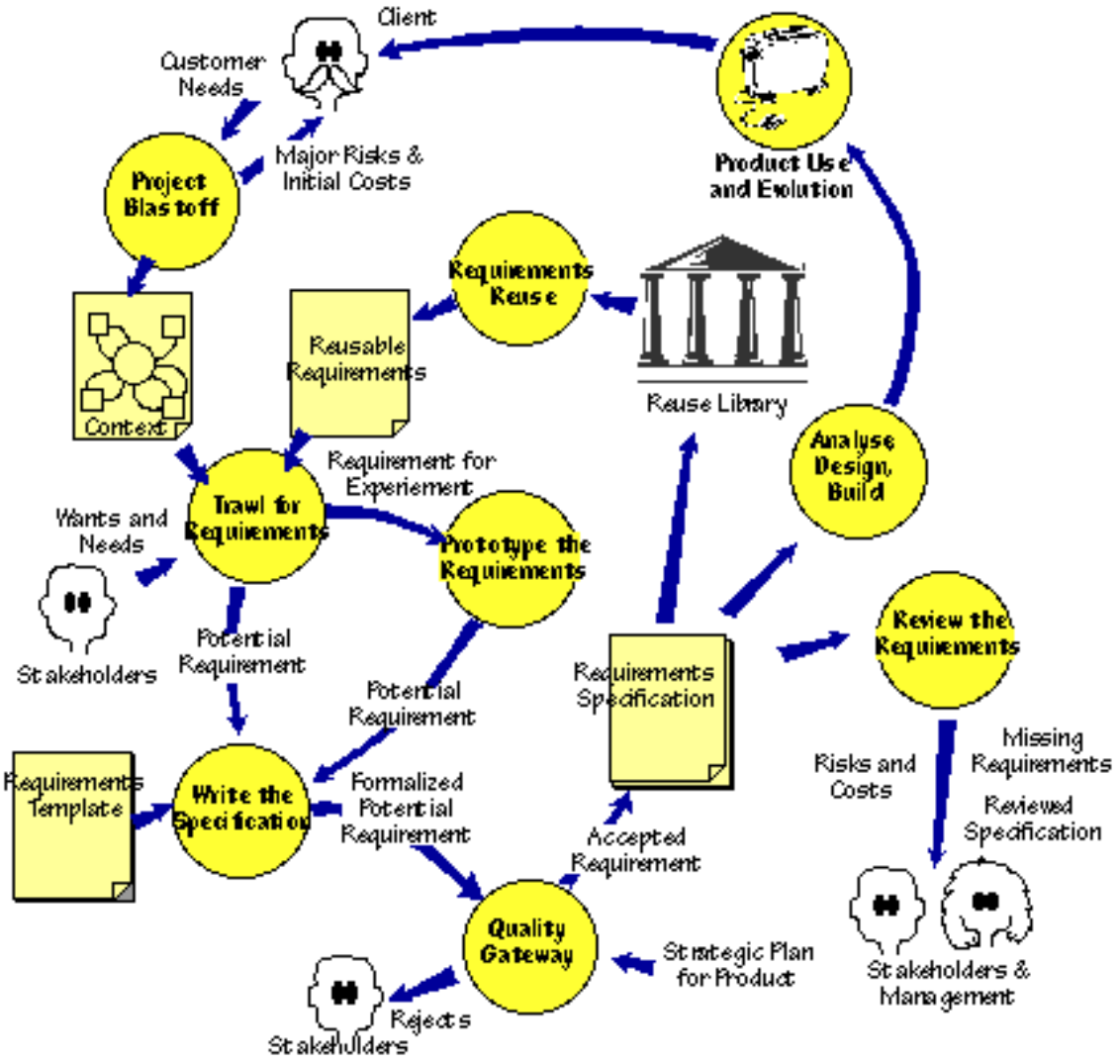
RE in V Model


Level of abstraction
system requirements
software requirements
preliminary design
detailed design
code & debug
unit test
component test
software integration
acceptance test
system integration
analyze and design
test and integrate
Time

- Appendix

RE Processes:
Volere Requirements Process

How many cycles? When to analyze and negotiate?



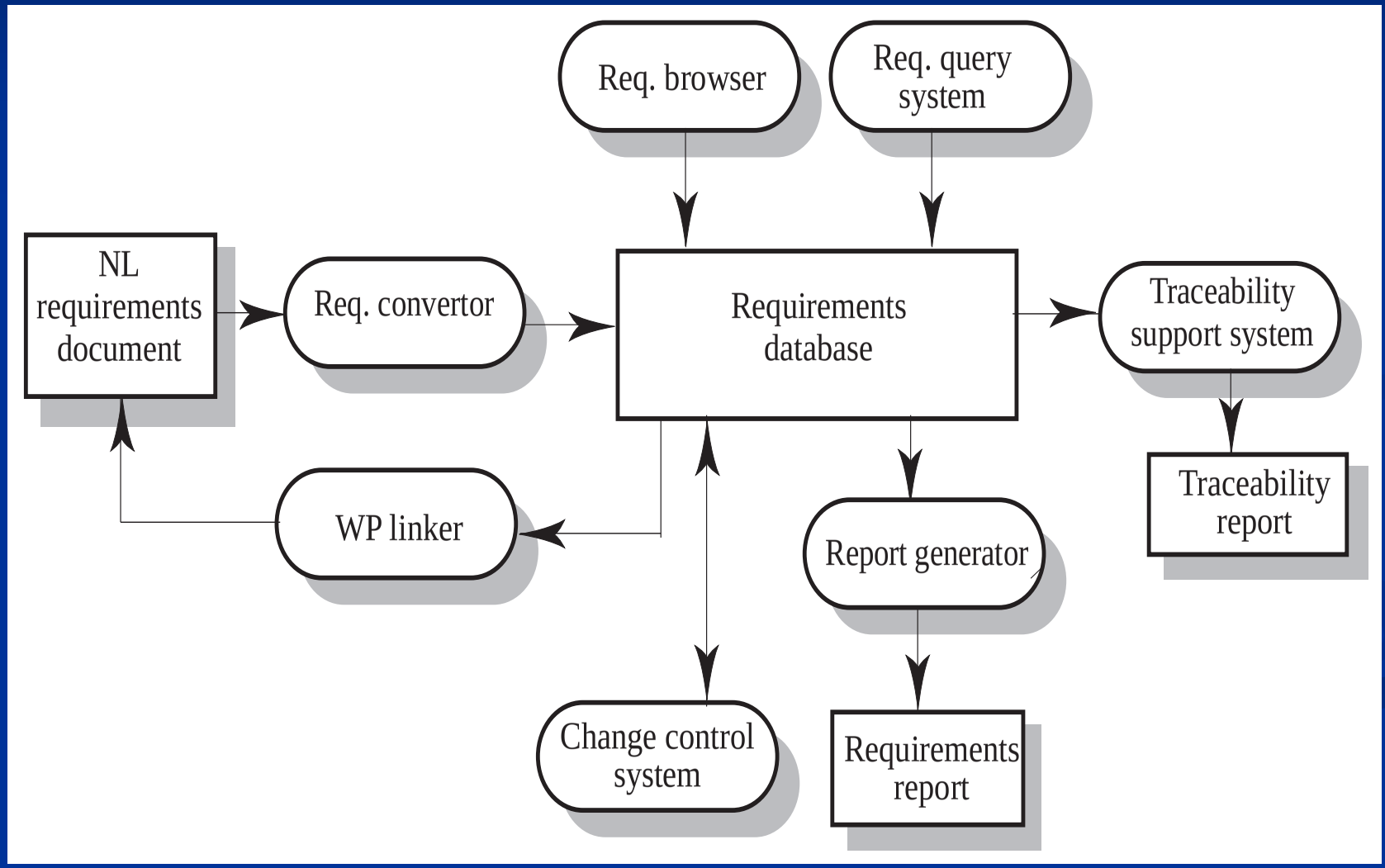
RE Processes:
RE Process Variability

Many Variety …and Evolution is inevitable

- RE processes vary radically from one organisation to another
- Factors contributing to this variability include
  - Technical maturity
  - Disciplinary involvement
  - Organisational culture
  - Application domain
  - …
- There is therefore no 'ideal' requirements engineering process [KotonyaSummerville98]

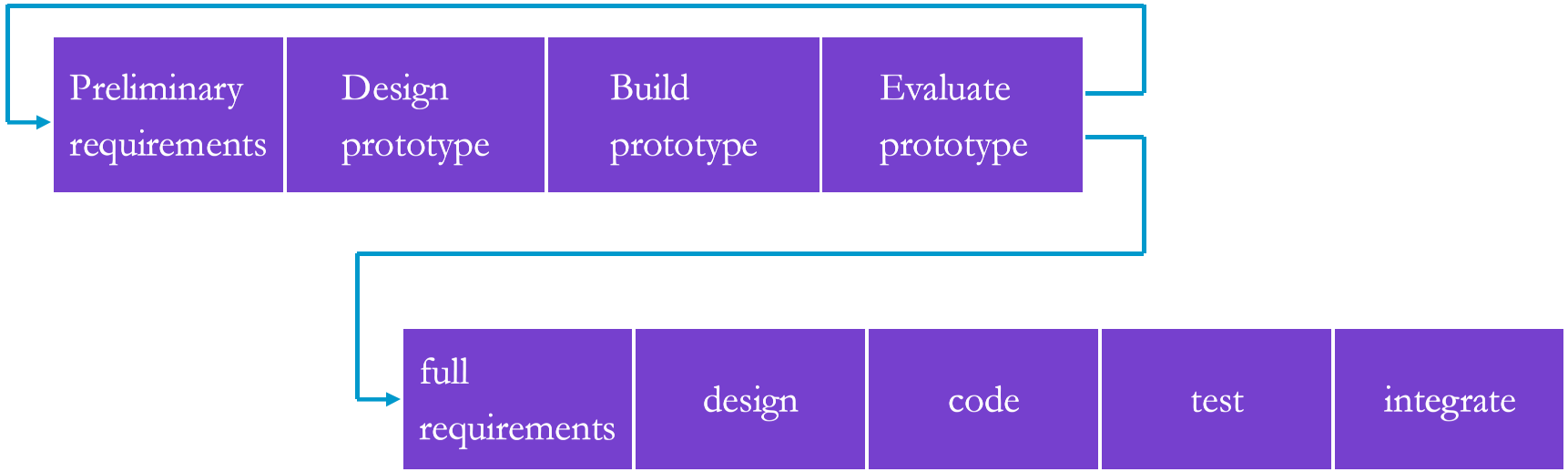
NFRs & RE Process: A Requirements Management System

Many variations and extensions



RE in Prototyping Lifecycle

[Dorfman, 1997, p9]



- **Prototyping is used for:**
  - understanding the requirements for the user interface
  - examining feasibility of a proposed design approach
  - exploring system performance issues
- **Problems:**
  - users treat the prototype as the solution
  - a prototype is only a partial specification