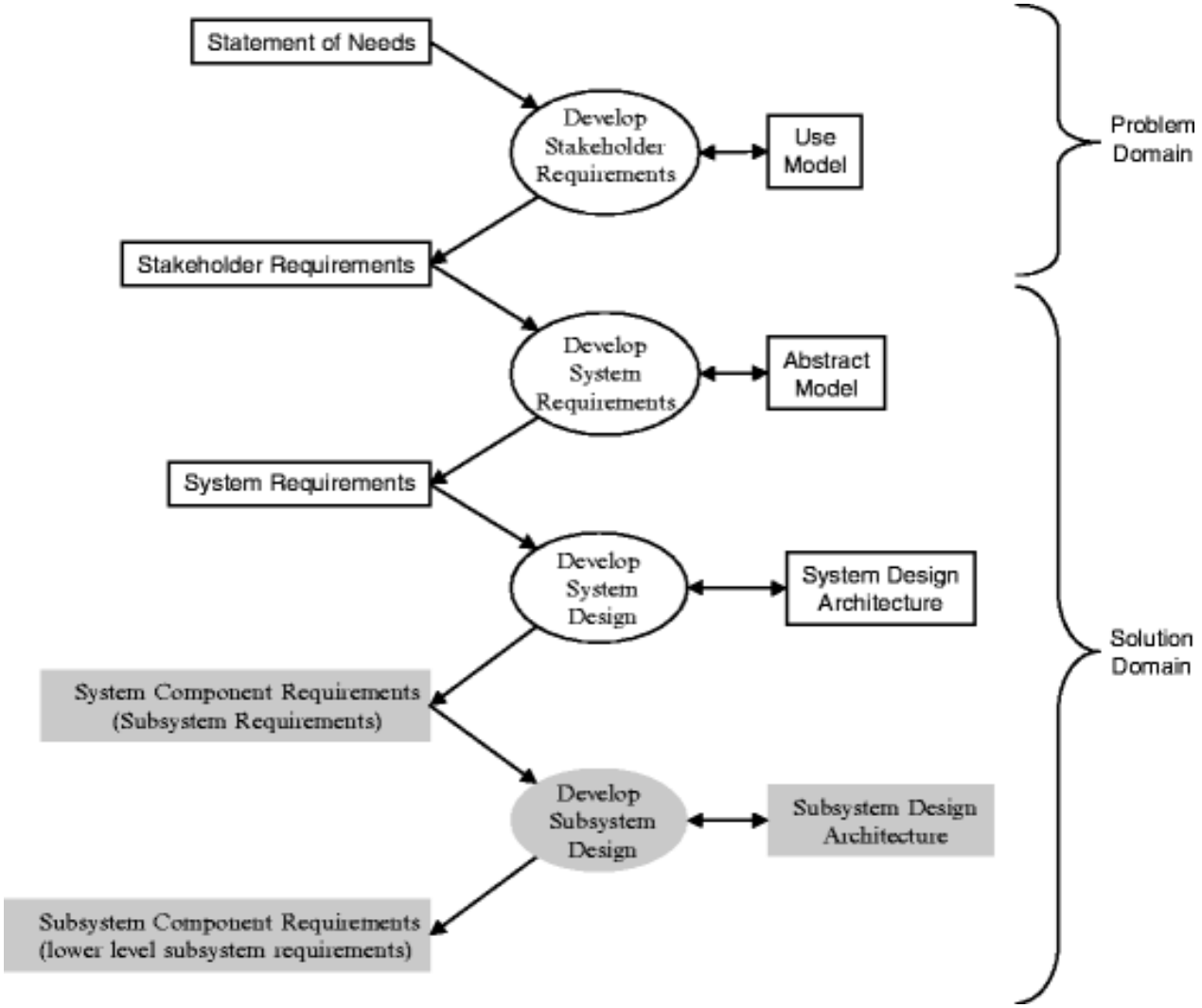
Types of Requirements and Requirement Boilerplating

The Purpose of Requirements

- Can act as a form of contract between the provider and their stakeholders.
- Describes what people want from their systems as well as what the systems have to do for them.
- Fulfills a need of some kind that is currently missing.
  - In the MoD this is called a “capability”

A simple model of system development



Statement of Needs (SON)

- A clear, concise statement about what is the overall goal of the system and how it will accomplish those goals.
- Usually determined by the client/sponsor early on.
  - Although it is often poorly done.
- Describes what capability the system being developed will provide.
- Provides a broad understanding of what the final outcome of the endeavour is going to be.
- Can act as a long term measuring stick regarding how well the requirements process has been completed.
  - If you as a requirements engineer cannot look back at the SON and decide whether your requirements describe a system that meets this need you have failed your stakeholders.

Example: Blizzard's World of Warcraft

- Utgarde Keep

- Do game designers have SON's?

Example of an SON: Blizzard

- *"[A goal of this] 'starter' dungeon is that you and your party can have a solid, memorable experience with plenty of story payoffs and some nice rewards to boot -- all in a single run."*
  - *Blizzard Development Team, Wrath of the Lich King (http://www.worldofwarcraft.com/wrath/features/dungeons/utgarde.xml)*
- Gives a clear idea of what they want to accomplish, but how are they going to do it?

An SON outside of systems…

- Banking crisis SON (Telegraph 08-10-2008)

  The [£200 billion lending] proposals announced today [from the Bank of England] are intended to:

  - Provide sufficient liquidity in the short term;
  - Make available new capital to UK banks and building societies to strengthen their resources permitting them to restructure their finances, while maintaining their support for the real economy; and
  - Ensure that the banking system has the funds necessary to maintain lending in the medium term.

Kennedy Statement

- You may hear the term "a Kennedy statement" used from time to time (less these days). Why is that?
- John F. Kennedy gave a speech about landing a man on the moon …
  - "Before this decade is out to land a man on the Moon and return him safely to the Earth."

Stakeholder Requirements

- These are statements about what the stakeholders want to achieve through the system
  - They are written in the stakeholder's language in their domain
  - They do not make specific reference to a solution or technology
  - These are in the "problem domain" – something that needs to be addressed by some new or revised system

User Requirements

- Users are a special type of stakeholder – they are the ones who will be doing specific actions with the system
- Statements regarding the tasks of users that should be satisfied by the target system
- Driven from the high level goals of the users in organizations

System Requirements

- A description of how the system will deliver on the needs of the users.
  - Detailed descriptions of functionality, services and constraints.
  - Written for technical implementations.
- Multiple system requirements will often satisfy a single user requirement.

Stakeholder to System Requirements

Stakeholder Requirement

- The game must provide a means of securing a player's characters and possessions against malicious misuse.

System Requirements

- The game will provide authentication of a user name.
- The game will secure password information with one-way encryption.
- The game will secure password information retrieval with provision of a Q&A scheme.

Types of Requirements

Type taxonomy for requirements:

- Functional requirements
  - Things a system must do
  - An action that the system has to take if it is to provide useful functionality for its user
- Non-Functional Requirements
  - Design Characteristics
    - Qualities a system must have
    - Usability, performance, safety, security
    - Often critical to a system's success
  - Constraints
    - Global issues that shape the requirements
    - Preferably defined at the beginning of gathering requirements

Functional Requirements

- Transformation: Required response to a condition/event
  - E.g., "customer inserts card" shall lead to "request PIN" in the description of a cash machine
- Invariant: Property that must always hold
  - E.g., "angle of attack (angle between airplane wing and airflow) shall never exceed 27 degrees"
- Failures: Forbidden/permissible transformations
  - Failure modes to be avoided, such as failures leading to safety hazards
  - Failure modes to be allowed

Design Characteristics

- Optimisation
  - Maximise/minimise some property, e.g., "minimise power consumption"
- Reliability/Availability
  - Statistical qualification, e.g., invariant "shall hold 99% of the time"
- Timing
  - Temporal constraints on transformation, e.g., "request PIN within 2s"
- Effectiveness/Efficiency
  - The user will be able to withdraw money 95% of the time within 30 seconds

Constraint Requirements

Constraint requirements apply to the entire system

Examples of system-wide constraints:

- Project constraints
  - E.g., financial constraints such as "spend less than £5m"
- Process constraints
  - E.g., standardisation constraints such as "use Def Stan 00-55" *(standard for the development of safety critical systems)*
- Design constraints
  - E.g., technology constraints such as "shall run on a personal computer"

Non-Functional requirements

- There is quite a bit of debate in the field about what is a "non-functional requirement"
- Functional is easy – it is something that the system must do to satisfy stakeholders
- Non-functional requirements is the set of design characteristics and constraints the system has – sometimes they are specified on their own, sometimes they are attached to functional requirements (as we will see)
- We will talk about non-functional requirements later in the module and how hard they are to work with

Fit Criteria for Requirements

- A good quality requirement consists of two parts:
  - A description capturing a stakeholder's intention
  - A fit criterion that quantifies this intention and gives us something to measure against for success
- Fit criteria quantify our requirements, making them measurable in a qualification process

Examples of Fit Criteria

- *Description:* The product shall record the weather station readings.
- *Fit criterion:* The recorded weather station readings shall match the readings sent by the weather station.
- *Description:* The product must produce the road de-icing schedule in an acceptable time.
- *Fit criterion:* The road de-icing schedule shall be available to the engineer within 15 seconds for 90% of the times that it is produced; the remaining times it will be available within 20 seconds. [Robertson and Robertson 1999]

Writing Requirements

- We need methods that allow us to write requirements that are:
  - Readable to all parties involved in system development – including the stakeholders
  - That they can be processed in a variety of different ways – including tracing through to the final system
- This means they need to be precise, unambiguous and clearly worded

Requirements Boilerplates

- In requirements, there are a variety of words that are used over and over again:
  - Shall/must – a mandatory requirement
  - Should/may – a requirement that is of lower priority (but by how much isn't clear)
- For example, a basic functional requirement:
  - A person between 4'5" and 7'2" shall be able to sit and drive the car.
  - A <stakholder> shall be able to <capability>

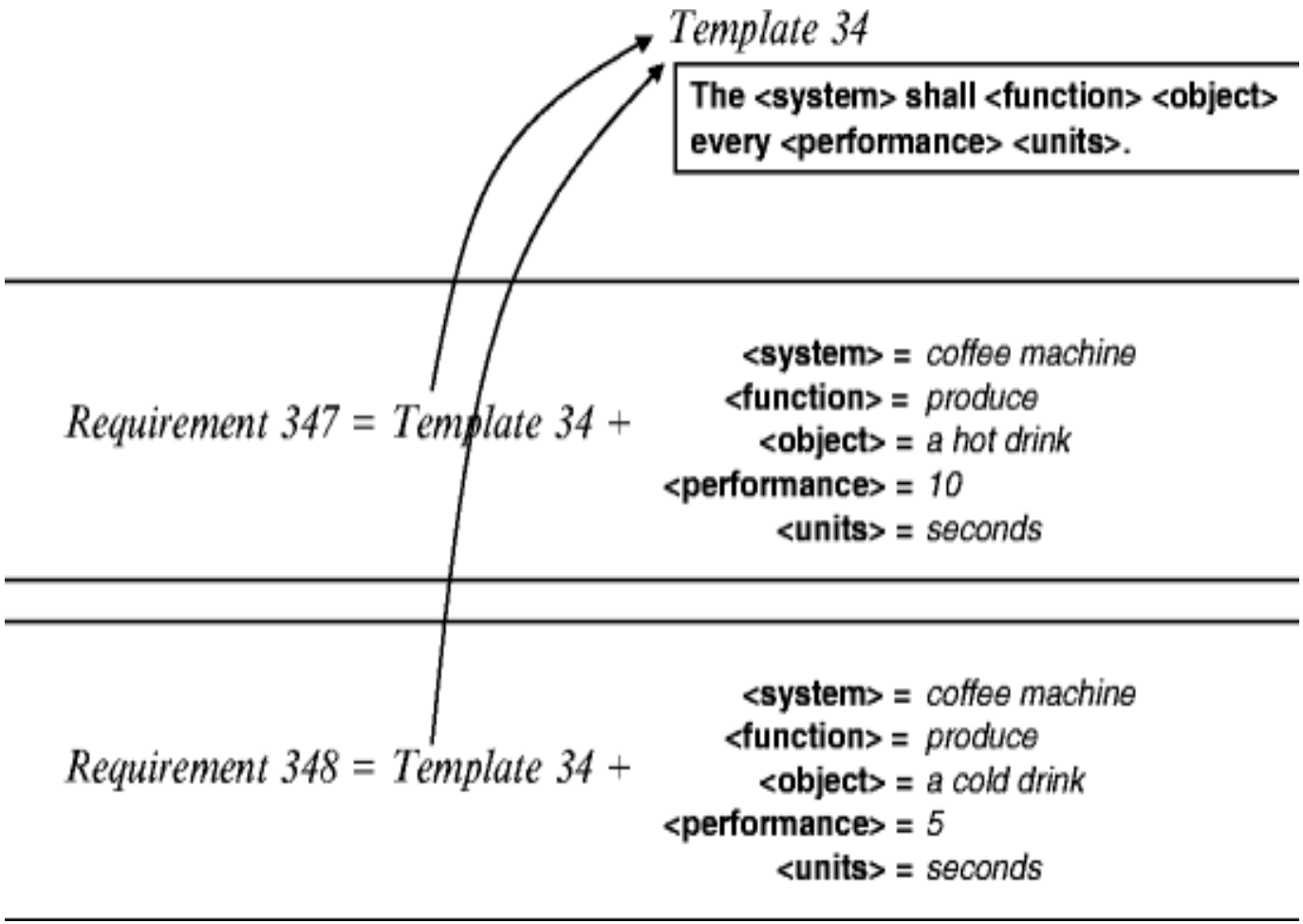
Requirements Boilerplates (2)

- We made add performance indicators or constraints to the requirements:
  - A person between 4'5" and 7'2" shall be able to sit and drive the car comfortably for 2 hours while driving in normal driving conditions.
  - A <stakeholder> shall be able to <capability> for <performance> while <operational condition>.

Requirements Boilerplates (3)

- We start to see more and more of these templates build up – in requirements books they are called "boilerplates"
- You can use templates to bring regularity and predictability to documents
- Many organisations will set up a central repository of boilerplates that can be used in requirements documents
- Can be implemented as macros in documents, or managed through complex requirements engineering systems

Boilerplate examples



Why use boilerplating?

- Why do we do this?
  - If we need to change the form of requirements later – we only need to change 1 template and all instances of that template will update automatically
  - Allows for complex querying and sorting of requirements if a requirements management system allows for it
  - In cases where you have confidential information, you can store the attributes of a template separate from the actual statement in the document – allowing you to redact it in public facing documents

Summary

- Requirements statements are at the heart of every specification.
  - They must be precise.
  - They must be unambiguous.
  - They must be clear.
- Building a set of requirements for a system is difficult – there could be thousands of individual requirements
  - Boilerplating is one means of standardizing how to express requirements

Reading

- Hull, Jackson and Dicks (2011) - Chapter 4, Section 4.8