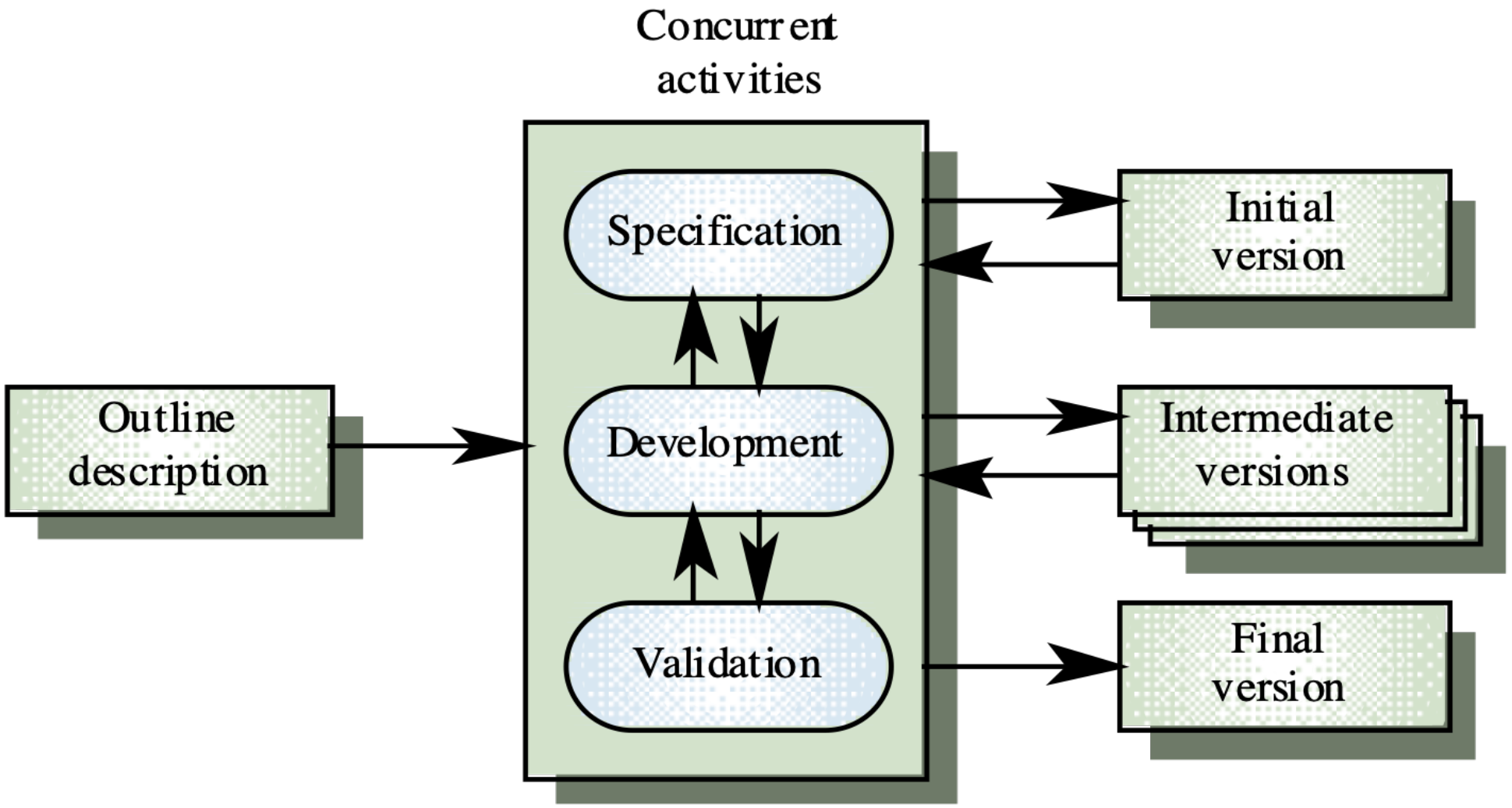
2. Evolutionary/Agile development

- **Exploratory development**
  - Aims to work with customers and to evolve to a final system from an initial **outline** specification.
  - Should start with **some** well-understood requirements.
  - The system evolves by adding new features as they are proposed by the customer.

- **Prototyping**
  - A software development technique, used to help understand system requirements. May start with poorly understood requirements
    - **Develop "quick and dirty" (or KISS: Keep It Simple and Stupid) system quickly;**
    - **Expose development to users' feedback continuously;**
    - **Refine and re-develop;**

    Until an adequate system is developed.

Evolutionary development



Agile Process Models: Examples

- Extreme Programming (**XP**)
- Adaptive Software Development (**ASD**)
- **Scrum**
- Dynamic Systems Development Method (**DSDM**)
- **Crystal**
- Feature Driven Development (**FDD**)
- Lean Software Development (**LSD**)
- Agile Modeling (**AM**)
- Agile Unified Process (**AUP**)

Evolutionary/Agile development

- **Problems**
  - Lack of process visibility
  - Systems are often poorly structured
  - Special skills (e.g. rapid prototyping) may be required
  - Can be expensive, e.g. due to need for higher level of communication

- **Applicability**
  - For small or medium-size interactive systems
  - For parts of large systems (e.g. the user interface)
  - For short-lifetime systems
  - Particularly suitable where:
    - **requirements are not possible to detail at the start;**
    - **powerful development (e.g. visual) tools are available and could be used to aid development**

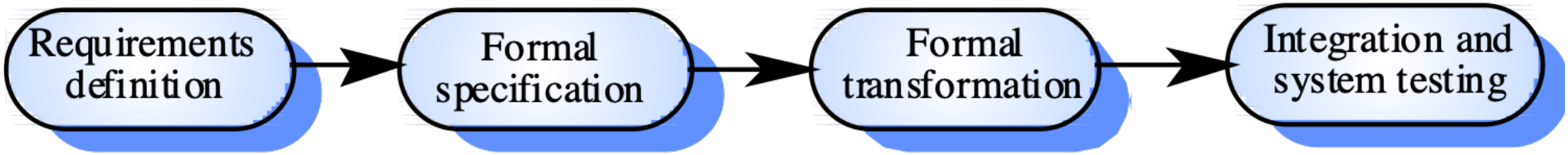
3. Formal systems development

Based on the transformation of a mathematical specification through different representations to an executable program

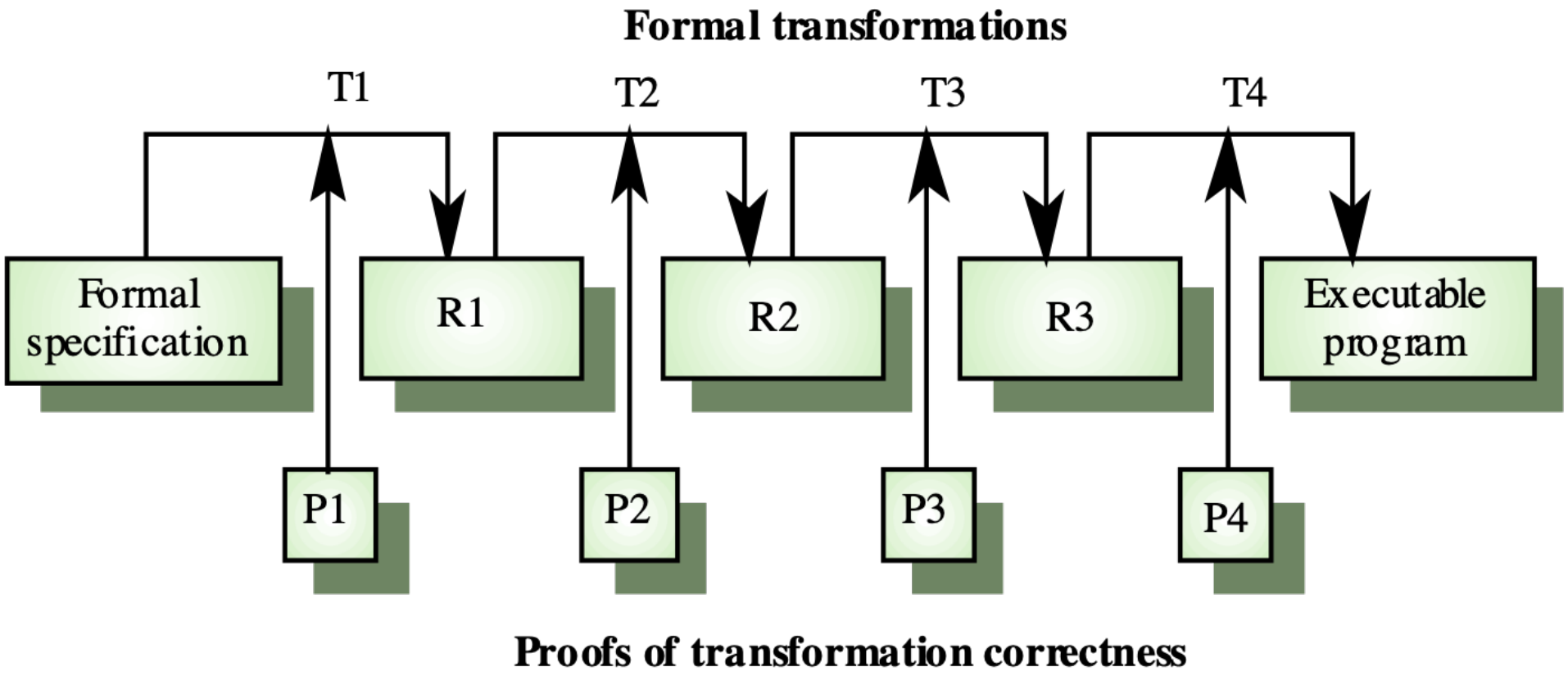
Transformations are 'correctness-preserving' so it is straightforward to show that the program conforms to its specification

Embodied in the 'Cleanroom' approach *(originally developed by IBM)* **to software development**

Formal systems development



Formal transformations



Formal systems development

- **Problems**
  - Need for specialised skills and training to apply the technique
  - Difficult to formally specify some aspects of the system (mathematically) such as the user interface
- **Applicability**
  - Critical systems, especially for those where a safety or security case must be made before the system is put into operation
  - Small systems or parts of a large system