Iterative, Incremental, *Risk-Driven* Development

What is risk?

- A risk is a potential problem that could cause some loss or threaten the success of a project.

  It might have, for example, an adverse impact on the cost, schedule, or technical success of the project, or on the quality of the product.

- Risks are circumstances/events that could cause a project to fail to meet its objectives.

- Two principal characteristics:

  - *Uncertainty* - it may or may not happen; no risk carries a likelihood of 100% (if it did it would be a constraint)

  - *Impact* - if the risk becomes reality, something bad results

  - The product of these is the ***risk exposure***

How can we reduce our risk exposure?

Risk Mitigation: taking steps to reduce adverse effects.

Risk avoidance
- avoid all exposure to the risk

Risk limitation/reduction
- take action to reduce probability and/or impact

Risk transfer
- make it someone else's problem who can better manage the risk

Risk acceptance
- monitor and rely on contingency action if the risk materialises

How to use risk as a driver?

- Maintain a prioritized risk list throughout the project on the basis of exposure

Risk ID	Unique identifier of each risk item
Description	Summary of risk item
Exposure	Product of Probability and Impact.
Consequence	What is the damage if the risk becomes a problem
Indicator	Earliest indicator that the risk has become reality
Mitigation strategy	Strategy to mitigate the risk
Actions/Status	Actions taken to date

- Revise the list at the start of every iteration and then plan the iteration to deal with the highest priority risks

Risk-Driven Iterative Development

Determining increments based on risk

Each iteration is a mini-project

So how do we know a risk has been eliminated?

What kind of risk do we use to drive the iterations?

Technical Risk

- risks related to new technologies adopted or needed for this project
- risks related to architecture such as need to develop robust sub-system structure that can support required functionality, suitability of the framework to be used, ability to meet quality requirements,...
- risks related to building the right system - correct identification and understanding of true functional and non-functional requirements.

Rather than maintaining requirements in the risk list, with exposure related to the risk of the system failing to satisfy them, we often maintain a requirements in a separate list prioritized by importance to the client.

We choose from both lists when planning an iteration. So the iterations are actually:
Risk-driven and Client-driven.
But to do this well, we'll see that the requirements must be in a particular form.

The iterative cycle with risk-related actions added

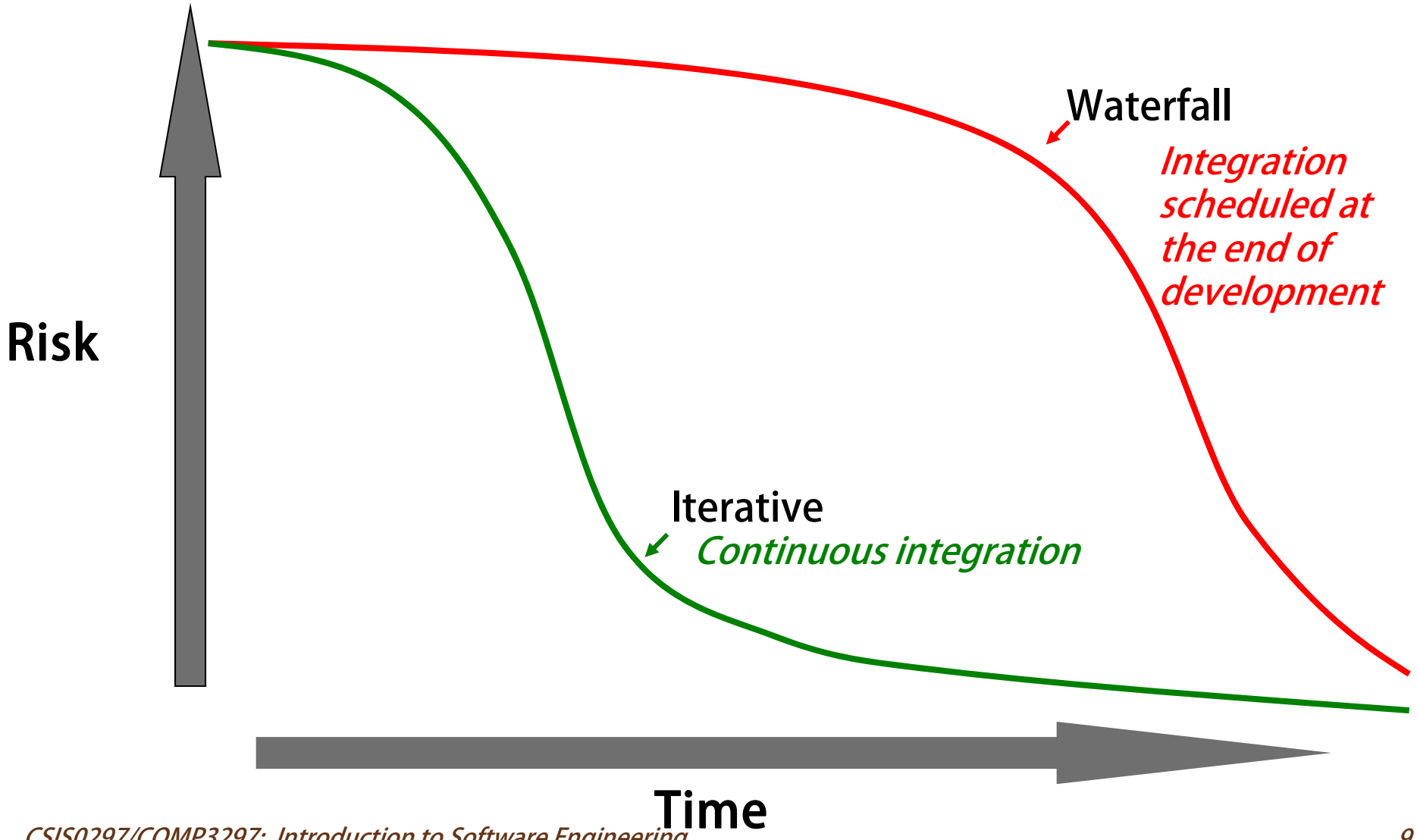
What about Non-technical risks?

- Our team doesn't have anyone with experience of the framework
- The schedule demanded by the client is too tight
- We don't have experience in system testing
- ...

These are real risks but not the kind of risk that is useful for driving our iterations (although work performed in the iterations may help mitigate them).

They are identified, tracked and handled by managers, and are out-of-scope for iteration planning.

Risk Reduction with an Iterative Model



What do we achieve in iterations at different stages?

First iteration: Understand what to build

- Vision, high-level requirements, business case
- Addresses business risks

Early iterations: Understand how to build it

- *Verified* baseline architecture, most requirements detailed
- Addresses architectural/technical risks

Main body of project: Build the product

- Working product, system test complete
- Addresses logistical risks

Final iteration(s): Validate solution

- Stakeholder acceptance
- Addresses delivery and roll-out risks

Summary of benefits of risk-driven Iterative and Incremental Process

- Early mitigation of big risks
- Early visible progress
- Early feedback, user engagement and adaptation – delivered system better meets needs of stakeholders
- Complexity is managed – e.g. no attempt to do *all* analysis up-front
- Experience during an iteration can be used to improve the process in later iterations